Embarcadero Performance Center 2.5 Web Client Guide

Embarcadero Technologies, Inc.
100 California Street, 12th Floor
San Francisco, CA 94111 U.S.A.

Contents

Performance Center Web Client

Embarcadero Performance Center now offers a Web Client so DBAs and operators can review database performance statistics from any Web browser without requiring any additional software. The Web Client is automatically installed on the server machine and can be accessed from any client or from a remote computer when you type in the URL as long as the Performance Center is running.

The statistics you see from the “fat client” are all visible in the Web Client with the exception of Oracle’s OS page statistics, so you can how many, and where, alarms have been triggered. You can also easily review drill-down details by clicking a target statistic.

> **NOTE:** Performance Center Web Client is also fully integrated into DBArtisan, which means that a DBA can view the new Web Client from within DBArtisan and monitor all the datasources that are registered with Performance Center.

To access the Web Client from a Performance Center client or server machine, on the **Toolbar>View>Web Client.**

To access the Web Client remotely, type the URL, which would look like:

<http://SERVERMACHINE/webclient>

OR

<http://SERVERMACHINE/perfcntr/ServerManager.dll?RequestPage?Page=EnterpriseView>.

From the Web Client, you can see:

- Enterprise View
- Monitor Details
- Server Details
- Client Details
- Admin Login
- Reports

Enterprise View

The Enterprise view is a single view designed to give you a comprehensive picture of the overall health of all monitored datasources in your enterprise. It displays all currently registered datasources. The Enterprise view lets you know what databases are up and down and gives you other critical indicators regarding the status of your monitored datasources. The Enterprise view for the main or “fat” client and the Web client differ slightly:

The table below describes the Enterprise View columns for the Web Client:

Column	Description
Datasource	Displays the name of the datasource(s).
DBMS	Displays the datasource’s platform.
Version	Displays the platform’s version for each datasource.
Datasource Up	Indicates if the datasource is running.
Listener Up	**ORACLE ONLY** Indicates if the Oracle-specific datasource listener is functioning.

Column	Description
Health Index	Indicates the current overall performance level of the registered datasource. The Health Index is on a scale of 0-100 with 100 being the ideal.
Alarms	Indicates any alarms for the registered datasource.

Enterprise View Graphics

In addition to the information in the columns, the Enterprise view colors each graphic, for both the main Enterprise View and the Web client, to give you a general overview of the health of your datasources and to alert you to any threshold violations. The table below describes the color and associated severity level for your datasources:

Color	Severity
Green	Low
Yellow	Warning
Orange	High
Red	Critical

You can use the Enterprise view to quickly determine the status of the registered datasources in your enterprise.

Performance Center Web Client Views

The performance category views for the Web Client provide you with important statistics for monitoring your datasources. The overall categories are the same for all supported platforms, however, the information available is specific to the datasource currently selected. You can open more than one performance category view by selecting a datasource from the Enterprise view.

When you click a target datasource, you see the same statistics that you see from the “fat client.” You can also click to see the drill-down details for a target metric.

In addition to the general Enterprise View, the Web Client offers a Monitor Details view, a Server Details view, a Client Details view, and an Admin Login view.

Monitor Details

The Monitor Details view of the Web client lets you see performance details of all registered datasources at a glance. The table below describes the information available to you from this view:

Column	Description
Monitored Datasource	Lets you see the name of the datasources registered to the server you are monitoring.
Monitor	Lets you see the monitoring capability and whether or not the datasource is fully enabled or disabled. You enable or disable datasource monitoring from the Datasource Properties dialog box.
Status	Shows the current datasource status and whether or not it is OK. Other status possibilities are Error, Unknown, or Transition, but these are rarely, if ever seen.
Connection Detail	Lets you see that status of the connection of the datasource to your server. Ideally it is connected and deployed. Other possibilities are Connected/Deployed Failed, Not Connected, Connected/Deployed Execution Error, and so on.
User ID	Lets you see the user ID.

Column	Description
Query Timeout (seconds)	Lets you see the amount of time, in seconds, before a query times out. The default is 300.
Ping Interval (seconds)	Lets you see the amount of time between pings from the server to the datasource. The default is 300.
Primary Interval (seconds)	Lets you see the amount of time between primary data sampling intervals. The default is 300
Secondary Interval (seconds)	Lets you see the amount of time between secondary data fetch intervals. The default is 900 seconds.
Last Ping Time	Lets you see the date and time the datasource was most recently pinged.
Last Primary Fetch Time	Lets you see the date and time, and the amount of time it took to fetch the primary data most recently.
Last Secondary Fetch Time (seconds)	Lets you see the date and time, and the amount of time it took to fetch the secondary data most recently.
DBMS	Lets you see the database platform for the target datasource.
Version	Lets you see the database platform version for the target datasource.

Server Details

The Server Details view of the Web Client lets you see information about the server you are currently connected to.

TIP: To connect to another server, you can use the Select Performance Center Server dialog box or you can type the server name in the Web Client url:

<http://pc_host/perfcntr?ServerManager.dll?RequestPage?Page=default>

where "pc_host" equals the name or IP of the machine hosting the Embarcadero Performance Center server. You can open multiple windows to see details for more than one server simultaneously.

NOTE: To start or stop the target server from this page, go to the Admin Login view and enable your administrator status.

The table below describes the information available from this view:

Column	Description
Server Name	Lets you see the name of the server you are currently connected to.
Port	Displays the server's port number. The default is 80.
Web Server	Displays the web server. Currently only Apache 2.0.35 is available.
Status	Lets you see the server's status. The options are OK, Error, Unknown, or Transition.
Status Detail	Lets you see whether the server is started or not.
Jobs	Lets you see the number of jobs currently running on the server.
Repository Datasource	Displays the name of the datasource where the repository is stored.
Status	Displays the repository status and whether it is OK or not.
Connection Detail	Lets you see the connection status of the repository.
User ID	Lets you see the name of the repository.

Column	Description
DBMS	Lets you see the database platform for the repository.
Version	Displays the database platform version of the repository.

Client Details

The Client Details view of the Web Client gives you information about the client machine you are currently connected to. The table below describes the information you see:

Column	Description
Client ID	This is the number of times users launch the fat client.
Machine Name	Displays the name of the client machine. In those cases where the server and the client are on the same machine, you will see localhost.
IP Address	Displays the IP address of the client machine.
Jobs	Lets you see the number of jobs running on the client machine.
Msgs In	Lets you see the number of messages sent from the server to the client.
Msgs Out	Lets you see the number of messages sent by the client to the server.
Msg Queue Length	Lets you see how many messages are waiting in the queue.
Msgs Lost	Lets you see how many messages were lost. Client can lose messages because the Embarcadero Performance Center Server is producing too much data. If this happens, you can decrease the Polling Frequency to 2 or 1 second(s). However, decreasing the Polling Frequency causes the Embarcadero Performance Center Client to consume more CPU.

Admin Login

The Admin Login lets you sign in as an administrator, if that status is available to you. Simply type your user name and password and click Authorize to enable Administrator Status.

When you are logged in as an administrator, you can start or stop the server from the Server Details page.

Reports

The Reports function on the Web Client lets you generate reports remotely for data you have previously exported as a data extraction job. No special permissions are required. This gives you the opportunity to view reports wherever you have access to the Web Client.

It is important to keep this in mind when scheduling a data extraction job. If you are planning to run reports from a remote client, be aware that you cannot change the particulars of an extraction job from a distance. Also, the more data that's involved in the extraction job, the longer it will take to generate a report. From the fat client, you can always edit a data extraction job to focus on a smaller number of statistics or on a narrower time frame to ensure that generating a report will be less time-consuming and focused on areas of greatest interest to you.

NOTE: After you created a data extraction job, you can always get the most recent data by checking the refresh data box before you generate a report.

To Create or Rerun Report:

1. Click **Create Custom Report**.
2. Select the **Report Type:**

   You can choose Database Availability, Statistical Trend Analysis, or Alarms from the drop-down list.
3. Select the **Job Name:**

   A drop-down list of the Data Extraction Jobs you previously created will appear.
4. **OPTIONAL**: To refresh data before you generate the report, check **Refresh Job Data**.
5. Type a title in the **Report Title** box.
6. Optionally, add a description of the report.
7. Click **Generate**.

   NOTE: It can take a while for the report to generate, depending on how much data is involved. When the report is completed, the main report page opens. There's no overt indicator that the report is being generated, so you have to have faith that a report will come your way.

Reading Reports

After a report is generated, or when you open the Reports page, you see a list of file names and the date each report was created. Each report is available in three file forms: .csv, .htm, data.htm. If you do not have Microsoft Excel, you can open the file using Notepad or Wordpad.

The .csv (comma-separated variables) file displays each record as a single line. If you have Microsoft Excel on your client, the file will open in an Excel spreadsheet. You may need to increase column widths to see the data.

The .htm file presents an overview of the statistics.

The data.htm file gives you a drill-down view of each of the statistics.

Reports you create in the Web Client are automatically saved to the fat client in the PC's root directory/ReportOutput/. You can change that directory path in the Server Options editor. To open the Server Options Editor, from the fat client's toolbar, go to Monitor/Server Options/Report Tab.

Oracle Expert Guide

This section includes expert help for all Oracle categories and statistics in Performance Center views. For detailed information on using the application, see Using Embarcadero Performance Center. This guide includes the following sections:

- Home View Statistics
- Contention Page Statistics
- I/O Page Statistics
- Memory Page Statistics
- Network Statistics
- Objects Page Statistics
- OS Page Statistics
- Session Details
- Space Page Statistics
- Users Page Statistics
- Other Statistics

Home Statistics

The Home view lets you review availability and overall performance of all monitored databases from a single window. The Home page includes the following sections:

- Contention Vital Signs
- I/O Vital Signs
- Memory Vital Signs
- Network Vital Signs
- Space Vital Signs
- Users Vital Signs

Related Topics

I/O Page Statistics

Memory Page Statistics

Objects Page Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Users Page Statistics

Network Statistics

Contention Statistics

Other Statistics

Memory Vital Signs

The following memory statistics are on the Oracle Home view:

- Buffer Cache
- Dictionary Cache
- Library Cache
- Memory Sort Ratio

Contention Vital Signs

The following contention statistics are on the Oracle Home view:

- Enqueue Waits
- Redo Log Space Wait Time
- Users Blocked
- Users Waiting

Users Waiting

- Metrics
- Troubleshooting

User connections that are waiting on a system generally occur for two reasons:

1. A process waits because a requested resource is not available.
2. A process waits for Oracle to perform a prerequisite task for its given operation.

These two wait causes are worth your time and investigation; idle waits (processes waiting because they have no work) should not worry you at all.

Metrics

To determine the actual wait causes user connections are experiencing, drill down from the global count of users waiting into the actual system and user wait details of your target database. This lets you locate the exact causes of currently experienced waits.

Troubleshooting

If you find a problem, drill down into wait details to determine whether the waits are resource related.

Users Blocked

On a small system, a single blocking user has the potential to stop work for nearly all other processes. On larger systems, this can cause major headaches. Although Oracle supports unlimited row-level locking, blocking lock situations do occur. User processes holding exclusive locks and not releasing them via a proper COMMIT frequency, generally cause most blocks.

Metrics

You should investigate any blocking lock statistic indicator above zero to prevent a mushrooming situation.

Troubleshooting

You can quickly remedy a blocking lock situation by issuing a KILL against the offending process. This eliminates the user's stranglehold on the accessed objects and lets other user processes complete. Tools like Embarcadero Performance Center make it easier to discover the blocked lock situation, the tricky part is preventing the blocking lock situation in the first place.

The culprit of blocking lock scenarios is usually the application design, or the SQL used within the application itself. Properly coding an application to reference database objects in an efficient order and then using the right SQL to get the job done, is an art. Most DBAs who have had to face Oracle Forms applications have suffered through the dreaded SELECT…FOR UPDATE statements that place unnecessary restrictive locks on nearly every read operation, know all too well that good coding practice is important. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible, something not always easy to do.

Enqueue Waits

An enqueue is an advanced locking device that lets multiple database processes share certain resources. Enqueue waits typically occur when sessions wait to be granted a requested lock. Sometimes these locks are internal Oracle locks while other times they could be locks for rows of data in a table.

> **NOTE:** Enqueues are issued implicitly by Oracle.

Metrics

You should investigate any enqueue waits that read consistently above one or more (delta statistics).

Troubleshooting

Removing contention for enqueues is usually an application design issue. Many enqueue waits are either contention for certain rows in the database or the result of database-initiated lock escalation. You should examine the use of indexes to make sure all referencing foreign keys are indexes and that your SQL does not tarry over rows in the database during modification operations. If it does, you should tune your SQL.

Enqueue waits can also be the result of space management tasks (such as objects extending) and disk sorts (mainly in tablespaces that do not make use of the TEMPORARY tablespace parameter).

Redo Log Space Wait Time

- Troubleshooting

The Oracle RDBMS is able to manage recovery by recording all changes made to a database through the use of redo log files. Oracle writes modifications made to a database to the redo log files, which you can archive off to another medium for disaster recovery. The background process that performs these operations is Oracle's Log Writer (LGWR). There is a buffer area in Oracle's System Global Area (SGA) that is used to reduce redo log file I/O, whose size, or lack thereof, can affect performance in a busy system. Sometimes a user process must wait for space in the redo log buffer. Oracle uses the log buffer to cache redo entries prior to writing them to disk. If the buffer area is not large enough for the redo entry load, waits can occur. While waits can be an important indicator of contention, a better measure is the amount of wait time that your users experience.

Metrics

The two main numbers to watch are:

1 Redo log space requests

2 Redo log wait time

If either statistic strays too far from zero, you may want to increase the log_buffer parameter and add more memory to the redo log buffer.

Troubleshooting

If you find a problem, do the following:

1 Edit the Init.ora file for the database.

2 Increase the amount of log_buffer to a higher value (take care not to over-allocate; ensure enough free memory exists on server before increasing value.

3 Cycle the Oracle server when possible to allow the new value to take effect.

4 Monitor new value to see if performance improves.

When adjusting the log_buffer, make sure that the amount is a multiple of the block size. Otherwise, on database startup, Oracle returns an error stating that you have entered an invalid amount for the redo log buffer.

NOTE: If you make the log_buffer parameter smaller than its default size for a given platform, Oracle will silently increase it.

I/O Vital Signs

The following I/O statistics are on the Oracle Home view:

- Logical Changes
- Logical Reads
- Physical Reads
- Physical Writes

Physical Reads

This value reflects total number of physical reads performed on all datafiles since the last refresh.

Metrics

Large numbers of physical reads may indicate that your buffer cache is too small. You should examine the buffer cache hit ratio to determine the overall effectiveness of logical vs. physical I/O.

Physical Writes

This value reflects total number of times the DBWR process has performed writes to various database datafiles since the last refresh.

Metrics

Wait events related to I/O activity are good indicators of physical I/O problems. These events include db file parallel write and db file single write. You can view both of these events in Embarcadero Performance Center.

Logical Reads

This value reflects total number of db block gets and consistent gets (data read from memory) since the last refresh.

Metrics

Large numbers of logical reads may not indicate a problem, although extremely large amounts could indicate a runaway process. You should examine the buffer cache hit ratio to determine the overall effectiveness of logical vs. physical I/O. You could also examine the User I/O Detail view to see if any one process is churning up a lot of logical I/O.

Logical Changes

This is the total number of changes that were made to all blocks in the SGA that were part of an update or delete operation. These changes generate redo log entries and are permanent if the transaction is committed. The number of logical changes is an approximate indication of total database work.

Metrics

None.

Users Vital Signs

The following users statistics are on the Oracle Home view:

- Active Connections
- Current Locks
- Open Cursors
- Total Connections

Total Connections

- Metrics
- Troubleshooting

This statistic represents the total number of open threads, or connections, currently reported in the database. This number includes both active and inactive processes.

Metrics

You should view the total number of sessions in light of the maximum number of processes allowed to connect to Oracle. The processes parameter in the Init.ora file specifies the maximum number of operating system user processes that can simultaneously connect to an Oracle Server. Checking the Users performance category view can show just how close the system is coming to running out of processes.

Troubleshooting

If the total number of connections approaches the processes limit, do the following:

1. Edit the Init.ora file for the database.
2. Increase the amount of processes to a higher value.
3. Cycle the Oracle server when possible to allow the new value to take effect.

Active Connections

This statistic represents the total number of active and open threads, or connections, reported in the database. The number displays the number of processes actively performing work.

Metrics

None.

Current Locks

This statistic displays the total number of locks obtained by processes in the database.

Metrics

With respect to locks, the main thing to watch is that all DML locks currently held on the system do not approach the dml_locks limit specified in the Init.ora file. The parameter, dml_locks, limits how many locks can exist on the system at one time.

Troubleshooting

If the total number of locks approaches the dml_locks limit, do the following:

1. Ensure that user processes are efficiently using locks and are committing frequently to avoid excessive lock hold times before editing the Init.ora file.
2. Edit the Init.ora file for the database.
3. Increase the amount of dml_locks to a higher value.
4. Cycle the Oracle server when possible to allow the new value to take effect.

Space Vital Signs

The following space statistics are on the Oracle Home view:

- Free Space
- Used Space
- Tablespace Low on Space

Used Space and Free Space

- Metrics
- Troubleshooting

These statistics represent the total used and free space available in all tablespaces/datafiles in the database. Although this is good to know, you need a more detailed listing by tablespace to determine where any actual space shortages exist in the database.

You can view this information in the Embarcadero Performance Center Space performance category view.

Metrics

If any one tablespace begins to approach 90% used, the DBA should take action to prevent any future space allocation errors.

Troubleshooting

There are a couple of things a DBA can do to prevent a tablespace from running out of available free space:

1. Turn AUTOEXTEND on for the underlying tablespace's datafiles. This allows them to automatically grow when free space in the datafile has been exhausted.
2. Using the ALTER TABLESPACE…ADD DATAFILE... command, you can manually add a new datafile to a tablespace that is about to run out of available free space.

Tablespace Low on Space

- Metrics
- Troubleshooting

This statistic gives you a count of tablespaces with less than 10% free space.

Metrics

If any tablespace free space percent amount goes below 10%, you should take action to ensure that the tablespace does not run out of available free space.

NOTE: The default notification for Tablespace Low on Space is the designated Emergency Contact. If you do not want the notification to go to the designated Emergency Contact, you should create a Tablespace Low on Space notification.

Troubleshooting

There are two things you can do to ensure that a tablespace does not run out of available free space:

1 First, you should look into the use of Oracle's AUTOEXTEND feature. AUTOEXTEND lets you give an Oracle tablespace the ability to auto-grow when it has exhausted the free space contained within. You can let a tablespace grow in an unlimited fashion or put constraints on it to stop at a certain point. You can also dictate how much more free space the tablespace gets each time it needs more space than is available. However, AUTOEXTEND enabled for a tablespace does not mean that you cannot run out of space. Remember you still have the physical server limitations to contend with. Make sure you (or your sysadmin) keep a careful eye on the server drives that house your Oracle database files for available free space.

2 If the free space on a server drive nears its limit, disable AUTOEXTEND for the datafile(s) that are on that drive, and use the old-fashioned ALTER TABLESPACE... ADD DATAFILE command to place a new datafile for the tablespace on another drive that has more free space to offer.

TIP: AUTOEXTEND is not a replacement for proactive space planning on the part of the DBA. When the database needs extra space and AUTOEXTEND is activated by Oracle, you take a performance as Oracle allocates more space for the tablespace. Avoiding this type of extension aids performance, albeit in a small way.

Network Vital Signs

The following network statistics are on the Oracle Home view:

- Bytes from Client
- Bytes from DBLink
- Bytes to Client
- Bytes to DBLink

Bytes to Client

This statistic is the total number of bytes sent over the network to all Oracle client machines from the database since the last refresh.

Metrics

All users connected to a database via a client/server connection generate network traffic, so seeing decent volumes of network traffic is normal. Out of the ordinary numbers should raise alarms as possible network saturation could occur.

Bytes from Client

This statistic is the total number of bytes sent over the network to the Oracle database from all client machines since the last refresh.

Metrics

All users connected to a database via a client/server connection generate network traffic, so seeing decent volumes of network traffic is normal. Numbers out of the ordinary should raise alarms as possible network saturation could occur.

Bytes to DBLink

This statistic is the total number of bytes sent by all client machines over the network to any database link since the last refresh.

Metrics

All users connected to a database via a client/server connection generate network traffic, so seeing decent volumes of network traffic is normal. Numbers out of the ordinary should raise alarms as possible network saturation could occur.

NOTE: Database link traffic could experience longer response times depending on the distributed nature and setup of the involved databases.

Bytes from DBLink

This statistic is the total number of bytes received by all client machines over the network from any database link since the last refresh.

Metrics

All users connected to a database via a client/server connection generate network traffic, so seeing decent volumes of network traffic is normal. Numbers out of the ordinary should raise alarms as possible network saturation could occur.

NOTE: Database link traffic could experience longer response times depending on the distributed nature and setup of the involved databases.

- Metrics
- Troubleshooting

The buffer cache hit ratio is an indicator of how often user requests for data are satisfied through memory vs. being physically read from disk. Data read from memory produces user response times many times faster than when that same data is read from disk. Keeping physical I/Os to an absolute minimum is one of the Oracle buffer cache's purposes in life.

The table below describes the key counters Performance Analyst uses to calculate the buffer cache hit ratio:

Key Counter	Description
DB BLOCK GETS	Data read from memory for DML operations.
CONSISTENT GETS	Data read from rollback segments in memory.
PHYSICAL READS	Data read physically from disk.
Direct Reads	Data read physically from disk that bypasses the buffer cache. Direct reads are filtered out of overall physical reads so an accurate cache hit ratio can be determined.

Dividing the data read from memory by data read from disk yields the cache hit ratio.

NOTE: This statistic is also available on the Memory home page.

Metrics

To help ensure excellent performance, you want to keep your cache hit ratio in the neighborhood of 90% or higher. However, every database has its own ‘personality’ and can exhibit excellent performance with below average readings for the cache hit ratio. Excessive logical I/O activity can produce a very high cache hit ratio while actually degrading overall database performance.

Investigate consistent low readings of 60% or less.

NOTE: For Oracle8i or earlier, the adjustment of the db_block_buffers tuning parameter is required. For Oracle9i and later, the db_cache_size parameter is the parameter that needs attention. Any increases in db_block_buffers to take effect on Oracle8i or earlier, the database must be cycled. The db_cache_size parameter in Oracle9i or later, however, is dynamic and can be altered without stopping and starting the database instance.

Troubleshooting

If a problem is found in Oracle8i or earlier, do the following:

- Edit the Init.ora file for the database.
- Increase the amount of db_block_buffers to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor the new value to see if performance improves.

If a problem is found in Oracle9i or later, do the following:

- Increase the size of the db_cache_size parameter through use of the ALTER SYSTEM SET db_cache_size command value (take caution to not over-allocate; ensure enough free memory exists on server before increasing value).
- Monitor the new value to see if performance improves.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started.

Library Cache Hit Ratio

Oracle's shared pool offers a number of tuning possibilities and is made up of two main memory areas:

1. Library Cache
2. Data Dictionary Cache

The library cache hit ratio offers a key indicator in determining the performance of the shared pool. It holds commonly used SQL statements - basically database code objects. The library cache hit ratio shows how often SQL code is reused by other database users vs. the number of times an SQL statement is broken down, parsed, and then loaded (or reloaded) into the shared pool.

You can improve performance by encouraging the reuse of SQL statements so expensive parse operations can be avoided. The library cache assists this tuning effort.

NOTE: This statistic is available on the Home page, the Memory home page and on the Library Cache tab of the Memory Detail.

Metrics

A high library cache hit ratio is a desirable thing. Strive for a hit ratio between 95-100%, with 99% being a good performance benchmark for code reuse.

> **NOTE:** When a database is first started, the library cache hit ratio is not at an optimal level because all code being used is relatively new, and as such, must be parsed and placed into the shared pool. If, however, after a solid hour or two of steady database time, the library cache hit ratio has not increased to desirable levels, increase the shared_pool_size parameter.

Other red flags that can indicate a too small shared pool include:

- A wait count for the event 'latch free' of 10 or greater.
- The library cache wait count of two or greater.

These indicators can be tracked with Performance Analyst's Bottleneck and Wait detail views.

You can improve the library cache hit ratio by encouraging SQL code reuse through the implementation of bind variables. Discouraging hard coded literals in application code and instead making use of variables bound at run time aids in the reuse of SQL code that is maintained in Oracle's shared pool.

> **NOTE:** Bind variables can have an affect on the cost-based optimizer though.

A second way is to pin frequently used code objects in memory so they are available when needed, by using the system supplied DBMS_SHARED_POOL package. You can use Performance Analyst to view objects in the shared pool that are always present and/or have increasing reload numbers to help identify objects that are good candidates for pinning.

Troubleshooting

If a problem is found in Oracle8i or earlier, do the following:

- Edit the Init.ora file for the database.
- Increase the amount of shared_pool_size to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor the new value to see if performance improves.

If a problem is found in Oracle9i or later, do the following:

- Increase the size of the shared_pool_size parameter through use of the ALTER SYSTEM SET shared_pool_size command value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Monitor the new value to see if performance improves.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started.

If you determine that SQL literals are causing SQL to not be reused, do the following (in Oracle 8.1.6 and later):

- Change the cursor_sharing parameter to FORCE by using the ALTER SYSTEM SET cursor_sharing=FORCE command.
- Monitor database to see if parse activity is reduced and library cache reloads shrink.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started. If using an Init.ora file, add the cursor_sharing=FORCE parameter to the file.

Dictionary Cache Hit Ratio

Oracle's shared pool offers an number of tuning possibilities and is made up of two main memory areas:

1 Library Cache

2 Data Dictionary Cache

The dictionary cache hit ratio offers another key indicator in determining the performance of the shared pool. It shows how often object definitions are found in memory vs. having to read them in from disk. Because Oracle references the data dictionary many times when an SQL statement is processed, it is imperative that as much of this vital reference information be kept in RAM as possible.

> **NOTE:** This statistic is also available on the Memory home page.

Metrics

Just as with the library cache, a high data dictionary cache hit ratio is desirable. Strive for a hit ratio between 90-100%, with 95% being a good performance benchmark.

> **NOTE:** When a database is first started, the data dictionary cache hit ratio is not at an optimal level because all references to object definitions are relatively new, and as such, must be placed into the shared pool. Look for hit ratios in the eighty's for new database startups. If, however, after a solid hour or two of steady database time, the data dictionary cache hit ratio has not increased to desirable levels, increase the shared_pool_size parameter.

Databases supporting applications that involve large number of objects (such as an Oracle Financials installation) should have larger than normal shared pools to support the required object definitions.

Although each parameter is not individually tunable (it was in Oracle6), you can see which area of the dictionary cache could be pulling the overall hit ratio down.

Troubleshooting

If a problem is found in Oracle8i or earlier, do the following:

- Edit the Init.ora file for the database.
- Increase the amount of shared_pool_size to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor new value to see if performance improves.

If a problem is found in Oracle9i or later, do the following:

- Increase the size of the shared_pool_size parameter through use of the ALTER SYSTEM SET shared_pool_size command value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Monitor the new value to see if performance improves.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started.

Memory Sort Ratio

Oracle's SGA is not the only memory structure used by Oracle for database work. One of the other memory areas used by Oracle8i and earlier for normal activity is an area set aside for sort actions. When a sort operation occurs, Oracle attempts to perform the sort in a memory space that exists at the operating system level. If the sort is too large to be contained within this space, it continues the sort on disk - specifically, in the user's assigned TEMPORARY TABLESPACE. Oracle records the overall number of sorts that are satisfied in memory as well as those that end up being finalized on disk. Using these numbers, you can calculate the percentage of memory sorts vs. disk sorts and get a feel for how fast your sort activity is being resolved. Obviously, memory sorts completes many times faster than sorts forced to use physical I/O to accomplish the task at hand.

Oracle9i or later now has the option of running automatic PGA memory management. Oracle has introduced a new Oracle parameter called pga_aggregate_target. When the pga_aggregate_target parameter is set and you are using dedicated Oracle connections, Oracle ignores all of the PGA parameters in the Oracle file, including sort_area_size, hash_area_size and sort_area_retained_size. Oracle recommends that the value of pga_aggregate_target be set to the amount of remaining memory (less a 10% overhead for other server tasks) on a server after the instance has been started.

NOTE: This statistic is available on the Home page, Memory home page, and the Users home page.

Metrics

If the memory sort ratio falls below 90%, and you are on Oracle8i or earlier, increase the parameters devoted to memory sorts - sort_area_size and sort_area_retained_size.

For Oracle9i or later, investigate the use of pga_aggregate_target. Once the pga_aggregate_target has been set, Oracle automatically manages PGA memory allocation, based on the individual needs of each Oracle connection. Oracle9i or later allows the pga_aggregate_target parameter to be modified at the instance level with the alter system command, thereby lets you dynamically adjust the total RAM region available to Oracle9i.

Oracle9i also introduced a new parameter called workarea_size_policy. When this parameter is set to automatic, all Oracle connections benefits from the shared PGA memory. When workarea_size_policy is set to manual, connections allocates memory according to the values for the sort_area_size parameter. Under the automatic mode, Oracle tries to maximize the number of work areas that are using optimal memory and uses one-pass memory for the others.

Troubleshooting

If you find a problem, do the following:

- Edit the Init.ora or SPFILE file for the database.
- Increase the amount of sort_area_size to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value. EVERY user receives this amount for sorting.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor new value to see if performance improves.

In addition to increasing the amount of memory devoted to sorting, find inefficient SQL that cause needless sorts. For example, UNION ALL does not cause a sort whereas UNION does in an SQL query (to eliminate duplicate rows). DISTINCT is frequently misapplied to SQL statements and causes unnecessary sort actions.

There are times you simply cannot stop sort activity. This being the case, try to keep it in memory whenever possible. However, large data warehousing systems oftentimes simply exhaust RAM sort allotments, so if disk sorts must occur, try to ensure three things:

1 Your user's TEMPORARY TABLESPACE assignment is not the SYSTEM tablespace, which is the default assignment.

   NOTE: For Oracle9i or later, you can specify a default tablespace other than SYSTEM for every user account that is created.

2 The TEMPORARY TABLESPACE assigned to your users is placed on a fast disk.

3 The TEMPORARY TABLESPACE has the tablespace parameter TEMPORARY assigned to it, which allows sort activity to be performed in a more efficient manner.

Parse/Execute Ratio

Each time a new SQL statement is submitted to Oracle, the kernel must ‘parse’ the statement, which involves syntax checks, security checks, and object validations. The Parse/Execute Ratio shows the percentage of SQL executed that did not incur a hard parse.

NOTE: This statistic is available on the Home page, Memory home page, and the Users home page.

Metrics

Seeing low values might indicate that users are firing SQL with many hard-coded literals instead of using bind variables within an application. High values (90% and greater) generally indicate Oracle is saving precious CPU by avoiding heavy parse tasks.

Troubleshooting

The best way to reduce unnecessary parse activity is to encourage SQL statement reuse. This can be done by promoting SQL execution through the use of stored procedures or applications where bind variables can be used. Oftentimes, literals in otherwise identical SQL statements can cause unneeded parse work for Oracle. The use of bind variables can counter that problem.

If you determine that SQL literals are causing SQL to not be reused, do the following (in Oracle 8.1.6 and later):

- Change the cursor_sharing parameter to FORCE by using the ALTER SYSTEM SET cursor_sharing=FORCE command.
- Monitor database to see if parse activity is reduced and library cache reloads shrink.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started. If using an Init.ora file, add the cursor_sharing=FORCE parameter to the file.

Buffer Busy Waits

Buffer busy waits occur when a process needs to access a data block in the buffer cache, but cannot because it is being used by another process. So it must wait. A wait event generally happens because a buffer is being read into the buffer cache by another process or the buffer is in the buffer cache, but cannot be switched to a compatible mode immediately.

Metrics

Buffer busy waits normally center around contention for rollback segments, too small an INITRANS setting for tables, or insufficient free lists for tables.

Troubleshooting

On Oracle8i or earlier, the remedy for each situation would be increasing the number of rollback segments, or altering tables to have larger settings for INITRANS to allow for more transactions per data block, and more free lists.

For Oracle9i or later, you can use the automatic segment management feature in Oracle9i locally-managed tablespaces to help make free list problems a thing of the past. Using an UNDO tablespace in 9i or later can help remedy any rollback contention problem.

You can also obtain which objects have actually experienced buffer busy waits in Oracle9i or later by querying the sys.v_$segment_statistics. This view is not populated unless the configuration parameter statistics_level is set to TYPICAL or ALL.

Rollback Contention Ratio

Rollback segments are used by Oracle to hold data needed to rollback (or undo) any changes made through inserts, updates, or deletes to various Oracle objects. They also allow Oracle to have read consistency for long running queries, are used for recovery purposes, and play a role during exports of database information. In a heavy transaction processing environment, rollback segments are accessed continuously and therefore are subject to contention problems. The Rollback Contention Ratio helps identify contention occurring on the system relating to rollbacks.

Metrics

Overall, if the rollback contention ratio approaches 1% or more, create more rollback segments. Also consider creating a specialized, larger, rollback segment to be used by long running transactions. Doing so alleviates dynamic rollback extensions and cuts down heavily on ORA-01555 Snapshot Too Old errors.

Troubleshooting

Begin by creating new rollback segments and altering them to be online for use. Then monitor the overall contention ratio to see if it begins to drop.

If you are using Oracle9i or later, consider using an UNDO tablespace and allowing Oracle to automatically control rollback segment management.

Latch Miss Ratio

Protecting the many memory structures in Oracle's SGA are latches. They ensure that one and only one process at a time can run or modify any memory structure at the same instant. Much more restrictive than locks (which at least allow for some collective user interaction), latches have no queuing mechanism - so either you get it or you do not and are forced to continually retry.

The latch miss ratio defines the number of times a process obtained a willing-to-wait latch vs. missing the attempt.

Metrics

If the latch miss ratio exceeds 1%, you should take action to resolve the amount of latch contention.

Troubleshooting

Examine the details regarding the latch contention. Increasing the shared_pool_size can assist in latch problems also. The table below describes latches:

Latch	Description
Cache buffer chain latch	Protects paths to database block buffers in the buffer cache. High I/O loads tend to cause contention for this latch. You can alleviate contention somewhat by adding more buffers to the cache (through the db_block_buffers or db_cache_size parameter) or by adding more LRU latch chain latches with the db_block_lru_latches parameter.
Library cache latches	Protects cached SQL statements in the library cache area of the Oracle shared pool. Contention for this latch is the usual result of literals being used for SQL statements instead of bind variables.

Other routine latch contention problems used to include the redo allocation and redo copy latches, but these have pretty much been made obsolete in Oracle 8.1.5 and later.

Parallel Query Busy Ratio

Oracle's parallel query feature, when used properly, allows for terrific increases in performance for many SQL and utility operations. Parallel operations can be introduced through SQL hints, specified in an object's DDL, or used in command line arguments for utility programs (like SQL*Loader). To effectively service parallel query requests, ensure that enough query servers exist in the database instance. The Parallel Query Busy Ratio is an indicator of how busy all the servers are on the database in question.

Metrics

If the Parallel Query Busy Ratio approaches 80-90%, add more query servers to the database, or examining parallel requests to ensure they are being used in an efficient and necessary manner.

Troubleshooting

To fix, do the following:

- Edit the Init.ora file or SPFILE for the database.
- Increase the amount of parallel_max_servers to a higher value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor new value to see if performance improves.

You can also investigate the use of the parallel_automatic_tuning parameter in Oracle 8.1 and later.

Free Shared Pool Percent

Oracle's shared pool need not use all of the memory given to it through the shared_pool_size parameter. If the database does not have many object and code definitions to reference, then the shared pool can contain an abundance of free memory that is not being used.

Metrics

Under-allocating the shared pool size can have a serious impact on your database's performance, but over-allocating the shared pool can have run time ramifications as well. If you have a good chunk of memory allocated to the Oracle shared pool that is never used, it might be more of a performance enhancement to reduce the shared pool amount and instead give the memory to the buffer/data cache, or even back to the operating system itself. In terms of knowing when to reduce the shared pool, a good benchmark is continually seeing 2-3MB of free memory.

On the other hand, if after an hour or so of beginning database operation, you see that virtually no free memory is left in the shared pool, or you are seeing ORA-4031 errors (that indicate definitions cannot find enough contiguous free space in the shared pool), increase the pool by 10% or more.

Troubleshooting

If you continuously see little or no free memory in the shared pool, do the following:

For Oracle8i or earlier:

- Edit the Init.ora file for the database.
- Increase the amount of shared_pool_size to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor the new value to see if performance improves.

For Oracle9i or later:

- Increase the size of the shared_pool_size parameter through use of the ALTER SYSTEM SET shared_pool_size command value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Monitor the new value to see if performance improves.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started.

You can also use the ALTER SYSTEM FLUSHED SHARED_POOL command to remove all objects from the shared pool and start with a clean slate.

Problem Tablespaces

The Problem Tablespaces metric is a count of all tablespaces in the database whose free space percentage amount has fallen below a user-defined limit.

Metrics

A rule of thumb for dynamic tablespaces (those with growing objects) is to keep a minimum of 10-15% free space available for object growth.

Troubleshooting

There are two ways to prevent a tablespace from running out of available free space:

1. Turn AUTOEXTEND on for the underlying tablespace's datafiles. This allows them to automatically grow when free space in the datafile has been exhausted.
2. Using the ALTER TABLESPACE … ADD DATAFILE… command, you can manually add a new datafile to a tablespace that is about to run out of available free space.

Problem Objects

The Problem Objects statistic is a count of all objects in the database that are in danger of reaching their maximum extent limit or cannot allocate their next extent of free space because of a lack of overall or contiguous free space in their parent tablespace.

Metrics

Modify any object approaching their maximum extent limit or unable to allocate a new extent of space so they can continue to grow in size.

Troubleshooting

Depending on the situation, there are a number of things you can do to prevent object extent problems:

- Turn AUTOEXTEND on for the underlying parent tablespace's datafiles. This allows a tablespace to automatically grow when free space in the datafile has been exhausted, and allows an object to extend even when little or no current free space is available.
- Using the ALTER TABLESPACE … ADD DATAFILE… command, you can manually add a new datafile to a tablespace that is about to run out of available free space.
- You can alter an object that is at or near their maximum extent limit so that the object has unlimited extents.
- With Oracle 8.1.5 and later, you can use locally-managed tablespaces to ensure that no object ever reaches its maximum extent limit, because all objects are allowed unlimited extents.
- An object can be reorganized into another tablespace or reorganized in general to reduce the number of extents the object currently takes up.

Top System Bottlenecks

When viewing wait statistics, there are many levels of detail that you can view. The first level is the system view, which provides a global, cumulative snapshot of all the waits that have occurred on a system. Viewing these numbers can help you determine which wait events have caused the most commotion in a database thus far. The Top System Bottlenecks section identifies the top waits that have occurred on the Oracle database based on the number of waits per event.

Metrics

None.

Troubleshooting

Appendix A in the Oracle Reference manual contains a listing and description of every current wait event defined in Oracle. DBAs unfamiliar with what each event represents should keep this listing close by as they examine wait-based event metrics. For example, a ‘db file scattered read’ event is typically indicative of table scan operations. If you see many of these events, then you can begin to see if large table scans are occurring in the database. Like the ‘db file scattered read’ event, each wait event has its own meaning and individual end-resolution diagnosis.

After looking at system-level wait activity, you can discover which current connections are responsible for waits at the system level. Performance Analyst reports on historical and current wait events at the session level, making this investigation easy to accomplish.

Top Session Bottlenecks

When viewing wait statistics, there are many levels of detail that you can view. The first level is the system view, which provides a global, cumulative snapshot of all the waits that have occurred on a system. The second level is a historical look at waits from a session level. The third is which sessions are currently experiencing waits. The Top Session Bottlenecks section identifies the top sessions that are currently waiting, based on their wait time in seconds.

Metrics

None.

Troubleshooting

Appendix A in the Oracle Reference manual contains a listing and description of every current wait event defined in Oracle. DBAs unfamiliar with what each event represents should keep this listing close by as they examine wait-based event metrics.

The most common wait viewed at the session level is an ‘enqueue’ wait, which typically identifies lock contention. If enqueue waits are observed, then you can check the “Current Object Blocks” count on the Performance Analyst Home page, as well as the Users page which displays locks and blocking locks detail.

As with an enqueue event, each wait event has its own meaning and individual end-resolution diagnosis.

Current Object Blocks

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches even on large systems. Although Oracle supports unlimited row-level locking, blocking lock situations do crop up. Blocks are most often caused by user processes holding exclusive locks and not releasing them via a proper COMMIT frequency.

> **NOTE:** This statistic is also called Sessions Blocked on the Users home page and Session Blocks on the Objects home page.

Metrics

Investigate any indicator above zero immediately before the situation has a chance to mushroom.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied. You can issue a KILL against the offending process, which eliminates the user's stranglehold on the objects they were accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Analyst, but preventing the blocking lock situation in the first place is where it gets tricky. You can look at the Users Detail and view all current blocking locks to see exactly which sessions are holding the currently restrictive locks.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. Most DBAs who have had to face Oracle Forms applications have suffered through the dreaded SELECT … FOR UPDATE statements that place unnecessary restrictive locks on nearly every read operation, and know all too well that good coding practice is important. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

Enqueue Waits

An enqueue is an advanced locking device that allows multiple database processes to share certain resources. Enqueue waits typically occur when sessions wait to be granted a requested lock. Sometimes these locks are internal Oracle locks while other times they could be locks for rows of data in a table. Enqueues are issued implicitly by Oracle.

NOTE: This statistic is available on the Home page and the Objects home page.

Metrics

Investigate any enqueue waits that read consistently above one or more (delta statistics).

Troubleshooting

Removing contention for enqueues is almost always an application design issue. If many enqueue waits are seen, this normally indicates either contention for certain rows in the database, or the result of database-initiated lock escalation. Examine the use of indexes to make sure all referencing foreign keys are indexes and that SQL is tuned to not tarry over rows in the database during modification operations.

Enqueue waits can also be the result of space management tasks (such as objects extending) and disk sorts (mainly in tablespaces that do not make use of the TEMPORARY tablespace parameter).

Free List Waits

Free lists are lists of Oracle data blocks that contain free space for an Oracle object. Every table has at least one free list. Free lists are used to locate free blocks of space when a request is made of a table for the insertion of a row. Free list contention can reduce the performance of applications when many processes are involved in the insertion of data to the same table.

NOTE: This statistic is available on the Home page and the Objects home page.

Metrics

If consistent numbers for free list waits continues to appear, add additional free lists to the most dynamic database objects through the use of the STORAGE parameter. Another indicator of insufficient free lists are consistent, non-zero numbers for the buffer busy wait event.

Troubleshooting

If free list waits are observed, add more free lists to tables and indexes with high insert rates. For Oracle9i or later, objects can be created inside of locally-managed tablespaces that use the automatic segment management feature, which entirely does away with free lists.

Total Used Space/Total Free Space

These statistics represent the total used and free space available in all tablespaces/datafiles in the database. Although good to know, a more detailed listing by tablespace is needed to determine where any actual space shortages exist in the database.

NOTE: These statistics are in the Storage Analysis section of the Performance Analyst Home page and the Space home page.

Metrics

If any one tablespace begins to approach 90% used (and the objects contained within it are dynamic and growing as opposed to static), take action to prevent any future space allocation errors.

Troubleshooting

Here are a some things you can do to prevent a tablespace from running out of available free space:

- Turn AUTOEXTEND on for the underlying tablespace's datafiles. This allows them to automatically grow when free space in the datafile has been exhausted.
- Using the ALTER TABLESPACE … ADD DATAFILE… command, you can manually add a new datafile to a tablespace that is about to run out of available free space.

For more information, see Storage Analysis.

Archive Log

Oracle can be placed into archivelog mode, which tells the database to make copies of the online redo log files for point-in-time recovery purposes. The Archive Log statistic displays whether the database is running in archivelog mode or not. This information is displayed in the Storage Analysis section of the Performance Analyst Home page.

Metrics

None.

For more information, see Storage Analysis.

Top Processes

When the database population as a whole experiences a system slowdown, it is not uncommon to find one or two users who are responsible for bringing the system to its knees. In the best of worlds, users have an evenly divided amount of memory usage, disk I/O, CPU utilization, and parse activity. Unfortunately, this is not usually the case. Users submit large batch jobs during peak OLTP activity, or when sessions that are firing off untuned queries on a critical system.

If you are seeing a slowdown in your database, and cannot seem to find a root cause, one thing to examine is the resource consumption of the leading sessions on a system. A handful of users can overwhelm the I/O capabilities of Oracle (through untuned queries or runaway batch jobs) or hammer the CPU or memory structures of the database and server.

Performance Analyst makes it easy to pinpoint the top sessions by showing the leading processes at the physical I/O, logical I/O, memory, and CPU usage levels.

Metrics

If any one session uses more than 50% of a total resource (CPU, memory, etc.) go into the session to find out what they are currently executing.

Active User Processes

The Active User Processes statistic is the total number of active and open threads reported in the database. Active Sessions displays the number of processes actively performing work.

Metrics

None.

For more information, see Inactive User Processes.

Inactive User Processes

The Inactive User Processes statistic is the total number of threads logged on to the database that are idle at the current time.

Metrics

A large number of inactive users could indicate user sessions that have mistakenly been left logged on. Because each user thread consumes a portion of memory on the Oracle server, sever any sessions not needing a connection to reduce resource usage.

Troubleshooting

Double-click this statistic to open the Top Sessions tab of the Users Detail page. On this tab you can check sessions that have many seconds idle and/or that have been logged on for very long periods of time as indicated by the logon time column. After verifying that the session is no longer necessary, you can KILL the session.

For more information, see Active User Processes.

Memory Page Statistics

The Memory home page includes the following sections:

- Bottleneck Analysis
- Key Ratio Analysis
- SGA Analysis
- SQL Analysis
- Workload Analysis - Top Memory Hogs
- Buffer Cache Hit Ratio
- Buffer Pool Allocations
- Buffer Pool Usage
- Data Dictionary Cache Hit Ratio
- Free Memory in Shared Pool
- Library Cache Hit Ratio
- Memory Sort Ratio
- Disk Sort Ratio
- Session Leaders - Memory
- SGA

Related Topics

Memory Detail

Home Page Statistics

I/O Page Statistics

Objects Page Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Users Page Statistics

Network Statistics

Contention Statistics

Key Ratio Analysis - Memory

Object-related database activity can be examined using both ratio-based and wait/bottleneck-based analysis. Ratio-based analysis involves examining a number of key database ratios and statistical readings that can be used to indicate how active certain object types are. Performance ratios serve as roll-up mechanisms for busy DBAs to use for at-a-glance performance analysis.

When using ratio-based analysis, there are some standards to adhere to. To start with, many of the formulas that make up ratio-based analysis must be derived from delta measurements instead of cumulative statistics. Many of the global ratios that you examines come from the v$sysstat performance view. The performance view maintains a count of all the occurrences (in the VALUE column) of a particular database incident (in the NAME column) since the database was brought up. For databases that are kept up for long periods of time, these values can grow quite large and impacts how a particular ratio is interpreted. However, if delta statistics are used (taking, for a specified sampling period, the before and after counts of each statistic that make up a ratio), then an accurate and current portrayal of the various ratios can be had.

A final thing to remember about using ratio-based analysis is that, while there are a number of rules of thumb that can be used as starting points to begin the evaluation of database performance, DBAs must determine each database's individual 'personality' with respect to the key performance ratios. Some hard and fast rules simply do not apply to every database. The danger in using blanket ratio standards is that they can lead you to haphazardly take action, which can at times contribute nothing to the situation, and sometimes even degrade performance.

The following memory ratios are used on the Performance Analyst Memory home page to succinctly communicate the general overall memory performance levels of the monitored database:

- Buffer Cache Hit Ratio
- Library Cache Hit Ratio
- Dictionary Cache Hit Ratio
- Memory Sort Ratio
- Parse/Execute Ratio
- Free Shared Pool Percent

For more information, see Memory Page Statistics.

Bottleneck Analysis - Memory

When an Oracle database is up and running, every connected process is either busy doing work or waiting to perform work. A process that is waiting may mean nothing in the overall scheme of things or it can be an indicator that a database bottleneck exists. You can use Bottleneck Analysis to determine if perceived bottlenecks in a database are contributing to a performance problem.

Memory bottlenecks can definitely cause performance degradation in an otherwise well-running database. Typically, these bottlenecks center around Oracle's buffer/data cache, library cache, and occasionally log buffer memory regions. To help you identify such problems, the following statistics are presented on the Memory home page:

- Buffer Busy Waits
- Free Buffer Wait Average
- Object Reloads
- Redo Log Space Wait Time
- Top Latch Misses

For more information, see:

Memory Home Page

Buffer Busy Ratio

SQL Analysis - Memory

Much of a database's overall performance can be attributed to SQL statement execution. Poorly optimized SQL statements can drag an otherwise well-configured database down in terms of end user response times. SQL statements that use much memory can also cause a problem in a database. The SQL Analysis for memory shows what SQL statements have consumed the largest percentages of shareable, persistent, and runtime memory

Before you can identify problem SQL in your database, you have to ask the question, "What is bad SQL?" What criteria do you use when you begin the hunt for problem SQL in your critical systems? Understand that even the seasoned experts disagree on what constitutes efficient and inefficient SQL; so there is no way to sufficiently answer this question to every Oracle professional's satisfaction. The table lists some general criteria you can use when evaluating the output from various database monitors or personal diagnostic scripts:

Criteria	Description
Overall Response (Elapsed) Time	The time the query took to parse, execute, and fetch the data needed to satisfy the query. It should not include the network time needed to make the round trip from the requesting client workstation to the database server. This statistic is available in Oracle9i or later.
CPU Time	The CPU time the query took to parse, execute, and fetch the data needed to satisfy the query.
Physical I/O	This is often used as the major statistic in terms of identifying good vs. bad SQL, this is a measure of how many disk reads the query caused to satisfy the user's request. While you certainly want to control disk I/O where possible, it is important that you not focus solely on physical I/O as the single benchmark of inefficient SQL. Make no mistake, disk access is slower than memory access and also consumes processing time making the physical to logical transition, but you need to look at the entire I/O picture of a SQL statement, which includes looking at a statements' logical I/O as well.
Logical I/O	The number of memory reads the query took to satisfy the user's request. The goal of tuning I/O for a query should be to examine both logical and physical I/O, and use appropriate mechanisms to keep both to a minimum.
Repetition	The number of times the query has been executed. A problem in this area is not as easy to spot as the others unless you know your application well. A query that takes a fraction of a second to execute can be a headache on your system if it has executed erroneously (for example, a query that executes in a runaway PL/SQL loop) over and over again.

There are other criteria that you can examine like sort activity or access plan statistics (that show items like Cartesian joins and the like), but more often than not, these measures are reflected in the criteria listed above.

Fortunately, Oracle records all the above measures (some only in 9i), which makes tracking the SQL that has been submitted against an Oracle database much easier.

Metrics

When you begin to look for inefficient SQL in a database, there are two primary questions you want answered:

1 What HAS been the worst SQL that has historically been run in my database?

2 What IS the worst SQL that is running right now in my database?

When troubleshooting a slow system, you should be on the lookout for any query that shows an execution count that is significantly larger than any other query on the system. It could be that the query is in an inefficient PL/SQL loop, or other problematic programming construct. Only by bringing the query to the attention of the application developers will you know if the query is being mishandled from a programming standpoint.

There is the possibility that the SQL statement just is not tuned well. You can view Performance Analyst's Top SQL view and, if you have Embarcadero SQL Tuner installed, you can port the SQL over to SQL Tuner to better optimize the statement.

SGA Analysis - Memory

Most DBAs know all about the Oracle System Global Area (SGA). The SGA is Oracle's memory structural area devoted to facilitating the transfer of data and information between clients and the Oracle database. The table below describes Oracle memory structures:

Memory Structure	Description
Default buffer cache	Maintains data blocks when they are read from the database. If you do not specifically place objects in another data cache, then any data requested by clients from the database is placed into this cache. The memory area is controlled by the db_block_buffers parameter in Oracle8i and earlier and db_cache_size in Oracle9i or later.
Keep buffer cache	For Oracle 8 or later, you can assign various objects to a special cache that retains those object’s requested blocks in RAM for as long as the database remains up. The keep cache's main is for often-referenced lookup tables that should be kept in memory at all times for fast access. The buffer_pool_keep parameter controls the size of this cache in Oracle8, while the db_keep_cache_size parameter handles the cache in Oracle9i or later. The keep pool is a sub-pool of the default buffer cache.
Recycle buffer cache	The opposite of the keep cache. When large table scans occur, the data that fills a memory cache will likely not be needed again and should be quickly discarded from RAM so that they do not occupy memory space and prevent needed blocks from assuming their place. Objects containing such data can be assigned to the recycle pool to ensure that such a thing does indeed occur. The buffer_pool_recycle parameter controls the size of this cache in Oracle8 and earlier, while the db_recycle_cache_size parameter handles the cache in Oracle9i or later.
Specific block size caches	For Oracle 8 or later, you can create tablespaces whose blocksize differs from the overall database blocksize. When data is read into the SGA from these tablespaces, their data has to be placed into memory regions that can accommodate their special block size. Oracle9i or later has memory settings for 2K, 4K, 8K, 16K, and 32K caches. The configuration parameter names are in the pattern of db_nk_cache_size.
Shared pool	Holds object structure as well as code definitions, and other metadata. Setting the proper amount of memory in the shared pool assists a great deal in improving overall performance with respect to code execution and object references. The shared_pool_size parameter controls the memory region.

Memory Structure	Description
Large pool	For Oracle 8 or later, you can configure an optional, specialized memory region called the large pool that holds items for shared server operations, backup and restore tasks, and other miscellaneous things. The large_pool_size parameter controls the memory region. The large pool is also used for sorting when the multi-threaded server (MTS) is implemented.
Java pool	Handles the memory for Java methods, class definitions, etc. The java_pool_size parameter controls the amount of memory for this area.
Redo log buffer	Buffers modifications that are made to the database before they are physically written to the redo log files. The log_buffer configuration parameter controls the memory area.

Oracle also maintains a fixed area in the SGA that contains a number of atomic variables, pointers, and other miscellaneous structures that reference areas of the SGA.

For more information, see Memory Home Page.

Workload Analysis - Top Memory Hogs

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Oftentimes, user connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system level (ORA-4030 errors).

If your database server does not have an overabundance of memory, periodically check to see who your heavy memory users are along with the total percentage of memory each takes up. If you see one or two users who have more than 25-50% of the total memory usage, investigate the sessions further to see the activities they are performing.

For more information, see Memory Home Page.

Buffer Cache Hit Ratio

- Metrics
- Troubleshooting

The buffer cache hit ratio is an indicator of how often user requests for data are satisfied through memory vs. being physically read from disk. Data read from memory produces user response times many times faster than when that same data is read from disk. Keeping physical I/Os to an absolute minimum is one of the Oracle buffer cache's purposes in life.

The table below describes the key counters Performance Center uses to calculate the buffer cache hit ratio:

Key Counter	Description
DB BLOCK GETS	The number of times a CURRENT block was requested. You review values for this statistic in two places. First, go to Memory Detail>Session Detail>Session I/O. Second, you can add this statistic to the Trends view.
CONSISTENT GETS	Data read from rollback segments in memory.
PHYSICAL READS	Data read physically from disk.
Direct Reads	Data read physically from disk that bypasses the buffer cache. Direct reads are filtered out of overall physical reads so an accurate cache hit ratio can be determined.

Dividing the data read from memory by data read from disk yields the cache hit ratio.

TIP: Click this statistic to drill down to the Hit Ratio by User tab of the Memory Detail view.

Metrics

To help ensure excellent performance, you want to keep your cache hit ratio in the neighborhood of 90% or higher. However, every database has its own 'personality' and can exhibit excellent performance with below average readings for the cache hit ratio. Excessive logical I/O activity can produce a very high cache hit ratio while actually degrading overall database performance.

You should investigate consistent low readings of 60% or lower.

NOTE: You must cycle the Oracle server to allow any increases in db_block_buffers to take effect.

NOTE: For Oracle8i or lower, the adjustment of the `db_block_buffers` tuning parameter is required. For Oracle9i or higher, the `db_cache_size` parameter is the parameter that needs attention. Any increases in `db_block_buffers` to take effect on Oracle8i or lower, the database must be cycled. The `db_cache_size` parameter in Oracle9i or later, however, is dynamic and can be altered without stopping and starting the database instance.

To view individual session hit ratios that can be depressing the overall cache hit ratio for the database, drill down into the overall buffer cache hit ratio.

Troubleshooting

If a problem is found in Oracle8i or earlier:

- Edit the Init.ora file for the database.
- Increase the amount of db_block_buffers to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor the new value to see if performance improves.

If a problem is found in Oracle9i or later:

- Increase the size of the db_cache_size parameter through use of the ALTER SYSTEM SET db_cache_size command value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Monitor the new value to see if performance improves.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started.

Buffer Pool Usage

Buffer cache usage statistics show the state and use of the buffers currently in the buffer cache and they include a count of how many there are for each state. Typically, the statuses of the buffers are:

- Free
- Read and modified

- Read and not modified
- Currently being read

Metrics

This grouping of statistics is quite helpful in determining if you have an overabundance of block buffers allocated. Consistently seeing large number of free buffers clues you in to the fact that you can reduce the amount of block buffers in the cache and give memory back to the operating system.

If, however, you see no free buffers within the first hour of bringing your Oracle database up, you can consider adding more buffers to the cache.

Troubleshooting

If you find a problem, do the following:

1. Edit the Init.ora file for the database.
2. Increase the amount of db_block_buffers to a higher value if free buffers are not found (take care not to over-allocate; ensure enough free memory exists on server before increasing value). Reduce the number of an overabundance of FREE buffers are present.
3. Cycle the Oracle server when possible to allow the new value to take effect.
4. Monitor the new value to see if more numbers of free buffers show up for the FREE status.

Buffer Pool Allocations

Because you can reference data objects with different usage patterns, Oracle8 offers the option to intelligently place objects into one of three buffer caches:

Buffer Cache	Description
KEEP	Minimizes misses in the buffer cache. Small, frequently referenced objects are candidates for the KEEP buffer pool.
RECYCLE	Lets you avoid having large numbers of infrequently accessed blocks of data crowd out objects that need to be referenced in RAM. The RECYCLE pool is good for large objects that are scanned from beginning to end without the need for keeping all their data in RAM.
DEFAULT	The traditional cache for all other data objects.

NOTE: Unless you specify the KEEP or RECYCLE buffer cache, Oracle automatically places objects into the DEFAULT buffer cache.

TIP: Click this statistic to drill down to the Buffer Pool tab of the Memory Detail view.

Metrics

When looking at overall database I/O activity, you should keep an eye out for objects that can lend themselves to placement into a particular buffer cache. Consider using a KEEP cache for relatively small, frequently accessed tables that require fast response times. Large tables with random I/O activity and are scanned from beginning to end a lot are good candidates for a RECYCLE cache.

Troubleshooting

You can use the STORAGE...BUFFER _POOL option to place objects into different buffer pools at object creation time. You use the ALTER command to set existing objects different pool.

> **TIP:** Tables, partitions, and indexes can be placed into the different caches.

If you just want to use the DEFAULT buffer pool and not enable any special caches, you can still encourage Oracle to keep certain objects in the buffer cache as long as possible by using the CACHE parameter. For example, issuing the command ALTER TABLE…CACHE specifies that the blocks retrieved for this table be placed at the most recently used end of the LRU list in the DEFAULT buffer cache when a full table scan is performed. The CACHE hint can also be used in SQL statements to cache a table, but used in this form, Oracle only caches the blocks until the next time the database is shut down. Once the database comes back up, the CACHE hint would have to be issued in an SQL statement again to cache the needed blocks of data.

Free Memory and Used Memory in Shared Pool

Oracle's shared pool does not need to use all of the memory provided through the shared_pool_size parameter. If the database does not have many object and code definitions to reference, the shared pool can contain an abundance of unused free memory. This statistic is tracked by Oracle in its performance views.

> **TIP:** Click this statistic to drill down to the Shared Pool tab of the Memory Detail view.

Metrics

Under-allocating the shared pool size can have a serious impact on your database's performance and over-allocating the shared pool can have runtime ramifications. If you have a good chunk of memory allocated to the Oracle shared pool that is never used, it might be more of a performance enhancement to reduce the shared pool amount and instead give the memory to the buffer cache, or even back to the operating system itself. In terms of knowing when to reduce the shared pool, a good benchmark is continually seeing 2-3 megabytes (MB) of free memory.

On the other hand, if after an hour or so of beginning database operation, you see virtually no free memory is left in the shared pool, or you are seeing ORA-4031 errors (that indicate definitions cannot find enough contiguous free space in the shared pool), you should definitely look into increasing the pool by 10% or more.

Troubleshooting

If you continuously see little or no free memory in the shared pool then you can do the following on Oracle8i or lower:

- Edit the init.ora file for the database.
- Increase the amount of `shared_pool_size` to a higher value (take caution to not over-allocate; ensure enough free memory exists on server before increasing value.)
- Cycle the Oracle server when possible to allow the new value to take affect.
- Monitor the new value to see if performance improves.

If a problem is found in Oracle9i or later:

- Increase the size of the `shared_pool_size` parameter through use of the `ALTER SYSTEM SET shared_pool_size` command value (take caution to not over-allocate; ensure enough free memory exists on server before increasing value.)
- Monitor the new value to see if performance improves.

- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started.

You can also use the `ALTER SYSTEM FLUSHED SHARED_POOL` command to remove all objects from the shared pool and start with a clean slate.

Library Cache Hit Ratio

- Metrics
- Troubleshooting

Oracle's shared pool offers a number of tuning possibilities and is made up of two main memory areas:

1 Library Cache

2 Data Dictionary Cache

The library cache hit ratio offers a key indicator in determining the performance of the shared pool. It holds commonly used SQL statements - basically database code objects. The library cache hit ratio shows how often SQL code is reused by other database users vs. the number of times an SQL statement is broken down, parsed, and then loaded (or reloaded) into the shared pool.

You can improve performance by encouraging the reuse of SQL statements so expensive parse operations can be avoided. The library cache assists this tuning effort.

NOTE: This statistic is also available on the Home page, the Memory home page and on the Library Cache tab of the Memory Detail.

TIP: Click this statistic to drill down to the Library Cache tab of the Memory Detail view.

Metrics

A high library cache hit ratio is a desirable thing. Strive for a hit ratio between 95-100%, with 99% being a good performance benchmark for code reuse.

NOTE: When a database is first started, the library cache hit ratio is not at an optimal level because all code being used is relatively new, and as such, must be parsed and placed into the shared pool. If, however, after a solid hour or two of steady database time, the library cache hit ratio has not increased to desirable levels, increase the shared_pool_size parameter.

Other red flags that can indicate a too small shared pool include:

- A wait count for the event 'latch free' of 10 or greater.
- The library cache wait count of two or greater.

You can track these indicators in Embarcadero Performance Center's Contention performance category view.

A way of improving the library cache hit ratio is by encouraging code reuse through the implementation of bind variables. Discouraging hard coded literals in application code and instead making use of variables bound at run time aids in the reuse of SQL code that is maintained in Oracle's shared pool.

NOTE: Bind variables can have an affect on the cost-based optimizer though.

A second way is to pin frequently used code objects in memory so they are available when needed, by using the system supplied DBMS_SHARED_POOL package. You can use Performance Center to view objects in the shared pool that are always present and/or have increasing reload numbers to help identify objects that are good candidates for pinning.

You can use Embarcadero Performance Center to view objects in the shared pool that are always present and/or have increasing reload numbers to help identify objects that are good candidates for pinning.

Troubleshooting

If a problem is found in Oracle8i or earlier:

- Edit the Init.ora file for the database.
- Increase the amount of shared_pool_size to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor the new value to see if performance improves.

If a problem is found in Oracle9i or later:

- Increase the size of the shared_pool_size parameter through use of the ALTER SYSTEM SET shared_pool_size command value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Monitor the new value to see if performance improves.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started.

Dictionary Cache Hit Ratio

Oracle's shared pool offers a number of tuning possibilities and is made up of two main memory areas:

1. Library Cache
2. Data Dictionary Cache

The dictionary cache hit ratio offers another key indicator in determining the performance of the shared pool. It shows how often object definitions are found in memory vs. having to read them in from disk. Because Oracle references the data dictionary many times when an SQL statement is processed, it is imperative that as much of this vital reference information be kept in RAM as possible.

TIP: Click this statistic to drill down to the Data Dictionary tab of the Memory Detail view.

Metrics

Just as with the library cache, a high data dictionary cache hit ratio is desirable. Strive for a hit ratio between 90-100%, with 95% being a good performance benchmark.

NOTE: When a database is first started, the data dictionary cache hit ratio is not at an optimal level because all references to object definitions are relatively new, and as such, must be placed into the shared pool. Look for hit ratios in the eighty's for new database startups. If, however, after a solid hour or two of steady database time, the data dictionary cache hit ratio has not increased to desirable levels, increase the shared_pool_size parameter.

Databases supporting applications that involve large number of objects (such as an Oracle Financials installation) should have larger than normal shared pools to support the required object definitions.

Although each parameter is not individually tunable (it used to be in Oracle6), you can see which area of the dictionary cache could be pulling the overall hit ratio down.

Troubleshooting

If a problem is found in Oracle8i or earlier:

- Edit the Init.ora file for the database.
- Increase the amount of shared_pool_size to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor new value to see if performance improves.

If a problem is found in Oracle9i or later:

- Increase the size of the shared_pool_size parameter through use of the ALTER SYSTEM SET shared_pool_size command value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Monitor the new value to see if performance improves.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started.

Memory Sort Ratio

Oracle's SGA is not the only memory structure used by Oracle for database work. One of the other memory areas used by Oracle8i and earlier for normal activity is an area set aside for sort actions. When a sort operation occurs, Oracle attempts to perform the sort in a memory space that exists at the operating system level. If the sort is too large to be contained within this space, it continues the sort on disk - specifically, in the user's assigned TEMPORARY TABLESPACE. Oracle records the overall number of sorts that are satisfied in memory as well as those that end up being finalized on disk. Using these numbers, you can calculate the percentage of memory sorts vs. disk sorts and get a feel for how fast your sort activity is being resolved. Obviously, memory sorts completes many times faster than sorts forced to use physical I/O to accomplish the task at hand.

Oracle9i or later now has the option of running automatic PGA memory management. Oracle has introduced a new Oracle parameter called pga_aggregate_target. When the pga_aggregate_target parameter is set and you are using dedicated Oracle connections, Oracle ignores all of the PGA parameters in the Oracle file, including sort_area_size, hash_area_size and sort_area_retained_size. Oracle recommends that the value of pga_aggregate_target be set to the amount of remaining memory (less a 10% overhead for other server tasks) on a server after the instance has been started.

> **TIP:** Double-click these statistics to drill down to the Disk Sort Detail tab of the Users Detail view.

Metrics

If your memory sort ratio falls below 90%, and you are on Oracle8i or earlier, increase the parameters devoted to memory sorts - sort_area_size and sort_area_retained_size.

For Oracle9i or later, investigate the use of pga_aggregate_target. Once the pga_aggregate_target has been set, Oracle automatically manages PGA memory allocation, based upon the individual needs of each Oracle connection. Oracle9i or later allows the pga_aggregate_target parameter to be modified at the instance level with the alter system command, thereby lets you dynamically adjust the total RAM region available to Oracle9i.

Oracle9i also introduces a new parameter called workarea_size_policy. When this parameter is set to automatic, all Oracle connections benefits from the shared PGA memory. When workarea_size_policy is set to manual, connections allocates memory according to the values for the sort_area_size parameter. Under the automatic mode, Oracle tries to maximize the number of work areas that are using optimal memory and uses one-pass memory for the others.

Troubleshooting

If you find a problem, do the following:

- Edit the Init.ora or SPFILE file for the database.
- Increase the amount of sort_area_size to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value. EVERY user receives this amount for sorting.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor new value to see if performance improves.

In addition to increasing the amount of memory devoted to sorting, find inefficient SQL that cause needless sorts. For example, UNION ALL does not cause a sort whereas UNION does in an SQL query (to eliminate duplicate rows). DISTINCT oftentimes is misapplied to SQL statements and causes unnecessary sort actions.

There are times you simply cannot stop sort activity. This being the case, try to keep it in memory whenever possible. However, large data warehousing systems oftentimes simply exhaust RAM sort allotments, so if disk sorts must occur, try to ensure three things:

- Your user's TEMPORARY TABLESPACE assignment is not the SYSTEM tablespace, which is the default assignment.

  NOTE: For Oracle9i or later, you can specify a default tablespace other than SYSTEM for every user account that is created.

- The TEMPORARY TABLESPACE assigned to your users is placed on a fast disk.
- The TEMPORARY TABLESPACE has the tablespace parameter TEMPORARY assigned to it, which allows sort activity to be performed in a more efficient manner.

Free Memory and Used Memory in Shared Pool

Oracle's shared pool need not use all of the memory given to it through the shared_pool_size parameter. If the database does not have many object and code definitions to reference, then the shared pool can contain an abundance of free memory that is not being used.

TIP: Click this statistic to drill down to the Shared Pool tab of the Memory Detail view.

Metrics

Under-allocating the shared pool size can have a serious impact on your database's performance, but over-allocating the shared pool can have run time ramifications as well. If you have a good chunk of memory allocated to the Oracle shared pool that is never used, it might be more of a performance enhancement to reduce the shared pool amount and instead give the memory to the buffer/data cache, or even back to the operating system itself. In terms of knowing when to reduce the shared pool, a good benchmark is continually seeing 2-3MB of free memory.

On the other hand, if after an hour or so of beginning database operation, you see that virtually no free memory is left in the shared pool, or you are seeing ORA-4031 errors (that indicate definitions cannot find enough contiguous free space in the shared pool), increase the pool by 10% or more.

Troubleshooting

If you continuously see little or no free memory in the shared pool, you can do the following:

If a problem is found in Oracle8i or earlier:

- Edit the Init.ora file for the database.
- Increase the amount of shared_pool_size to a higher value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor the new value to see if performance improves.

If a problem is found in Oracle9i or later:

- Increase the size of the shared_pool_size parameter through use of the ALTER SYSTEM SET shared_pool_size command value. Take caution to not over-allocate; ensure enough free memory exists on server before increasing value.
- Monitor the new value to see if performance improves.
- If using an SPFILE, save the new configuration values so Oracle reuses them each time the database is stopped and re-started.

You can also use the ALTER SYSTEM FLUSHED SHARED_POOL command to remove all objects from the shared pool and start with a clean slate.

Session Leaders - Memory

Frequently one or two users can cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Frequently, user connections can get out of hand with memory consumption and extreme cases have caused headaches at both the database and operating system level (ORA-4030 errors).

NOTE: This statistic displays on both the Users performance category view and the Memory performance category view.

TIP: Click this statistic to drill down to the Leading Sessions tab of the Memory Detail view.

Metrics

If your database server does not have an overabundance of memory, you should periodically check to see two things:

1. Who are your heavy memory users
2. The total percentage of memory each user consumes.

If you see one or two users who have more than 5-15% of the total memory usage, you should further investigate the sessions to see what activities they are performing.

Troubleshooting

You can use the Session Leaders - Memory statistic to find the users with the greatest allocations of overall memory. Than you can drill down into the details to discover the memory components that make up each user's session. Pay particular attention to the amounts of memory usage associated with these areas:

Area	Definition
PGA	The Program Global Area is a private memory area devoted to housing the global variables and data structures for a single Oracle process.
UGA	The User Global Area contains session-specific information regarding open cursors, state information for packages, database link information, and more. Note that when using Oracle's Multi-threaded Server (MTS), the UGA can be moved up into the SGA.
Memory Sorts	Memory sorts contain a count of how many memory sorts a session has performed.

SGA

The SGA (System Global Area) is Oracle's memory structural area devoted to facilitating the transfer of data and information between clients and the Oracle database. The table below describes its main tunable components:

Component	Definition
Database Buffers	Maintains data blocks that are read from the database. Properly sizing the buffer cache goes a long way in improving response time performance.
Var Size	Holds object structure as well as code definitions. Setting the proper amount of memory in the variable size also assists a great deal in improving overall performance with respect to code execution and object references.
Redo Log Buffer	Buffers modifications that are made to the database before they are physically written to the redo log files.
Fixed Size	Contains a number of atomic variables, pointers, and other miscellaneous structures that reference areas of the SGA.

Metrics

None.

Memory Detail

The following tabbed pages are available in the Memory Detail view:

- Buffer Pool
- Data Dictionary
- Library Cache
- Leading Sessions
- Objects in Memory
- Sessions Overview
- Shared Pool

Data Dictionary Tab

- Metrics

Oracle's data dictionary is a component of the shared pool that contains system elements necessary for the parsing of SQL statements, security resolution, object definitions, and more.

The overall data dictionary cache hit ratio provides a key indicator in determining the performance of the shared pool, and shows how often object definitions are found in memory vs. having to read them in from disk. Because Oracle references the data dictionary many times when an SQL statement is processed, it is imperative that as much as possible of this vital reference information be kept in RAM.

The Data Dictionary tab shows the individual elements of the data dictionary along with their associated hit ratios. In versions 6.x of Oracle, these individual elements could be tuned, but in versions 7.x and later, the only main method for tuning them involves the adjustment of the entire shared pool setting. Although not tunable from an individual parameter level, each displayed element gives insight into which area of the data dictionary is either adding to or detracting from overall performance.

The table below describes the information available on this tab:

Column	Description
Parameter	The name of the individual data dictionary element.
Usage	The number of cache entries that contain valid data.
Gets	The number of requests for this element. To see what's happening to Dictionary Cache Gets in a given datasource, you can add this statistic to your Trends view.
Get Misses	The number of requests resulting in a cache miss To see what's happening to Dictionary Cache Misses in a given datasource, you can add this statistic to your Trends view.
Hit Ratio	The ratio of cache hits versus misses of total requests. The maximum is 100%.
Scans	The number of scan requests.
Scan Misses	The number of times that a scan failed to find the needed data in the cache.
Scan Completes	The number of times the list was scanned completely.
Modifications	The number of insert, update, and delete actions.
Flushes	The number of disk flushes.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

An overall high data dictionary cache hit ratio is desirable, as are high hit ratios in each individual parameter. You should strive for a hit ratio between 90-100%, with 95% being a good performance benchmark.

NOTE: When a database is first started, the data dictionary cache hit ratio is not at an optimal level because all references to object definitions are relatively new, and as such, must be placed into the shared pool. Look for hit ratios between 80-90% for new database startups. If, however, after an hour or two of steady database time, the data dictionary cache hit ratio and individual hit ratios, have not increased to desirable levels, you should look into the possibility of increasing the shared_pool_size parameter.

NOTE: Note that databases supporting applications that involve large number of objects (such as an Oracle Financials installation) should have larger than normal shared pools to support the required object definitions.

Buffer Pool Tab

- Metrics

Because data objects can be referenced with different usage patterns, Oracle8 and later offers the option to intelligently place objects into one of three buffer caches. The table below describes the types of buffer caches:

Cache	Description
KEEP	Designed to minimize misses in the buffer cache. Small objects that are frequently referenced are candidates for the KEEP buffer pool.
RECYCLE	Designed to avoid having large numbers of infrequently accessed blocks of data crowd out objects that need to be referenced in RAM, the RECYCLE pool is good for large objects that are scanned from beginning to end without the need for keeping all their data in RAM.
DEFAULT	The traditional cache for all other data objects.

NOTE: Unless you specify the KEEP or RECYCLE buffer cache, Oracle automatically places objects into the DEFAULT buffer cache.

The Buffer Pool Tab of the Memory Detail View calculates the hit ratios for each of the Oracle8 and later buffer caches so you can easily see how often the objects placed into the various caches are being referenced in memory. Examining how often data is satisfied from memory vs. disk will help you determine if the caches are large enough and if they are being used in an optimal manner. The table below describes the information available in this tab

Column	Description
Buffer pool name	The name of the Oracle buffer pool.
Buffer pool hit ratio	The overall hit ratio for the particular cache.

NOTE: The DEFAULT buffer cache will only be shown for those installations not using the specialized caches available in Oracle8 and later.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

The KEEP buffer pool should maintain a hit ratio as close to 100% as possible. However, the buffer pool hit ratio is not near 100% until the database has been up and running in a typical state for a short time.

A poor hit ratio for the RECYCLE buffer pool may not be a bad thing since there is little chance of reusing a block stored in the buffer pool before it is aged out.

NOTE: If you place objects into the KEEP buffer pool, you should periodically reexamine their object sizes to ensure that they are not growing to a physical state that will jeopardize the performance of the KEEP pool.

Library Cache Tab

- Metrics

The library cache holds commonly used SQL statements - basically database code objects. A method for improving performance in Oracle is to encourage the reuse of SQL statements so expensive parse operations may be avoided. The library cache assists this tuning effort.

The Library Cache tab provides insight into how efficiently the library cache is operating. The table below describes the information available in this section:

Column	Description
Namespace	The region of the library cache.
Gets	The number of times a lock was requested for objects in the particular namespace.
Get Hit Ratio	The percentage of times (with 100% being the maximum) that the object was found in the cache.
Pins	The number of times a pin was requested for objects of this namespace. To see what's happening to Library Cache Pins in a given datasource, you can add this statistic to your Trends view.
Pin Hit Ratio	The percentage of times (with 100% being the maximum) that pin requests were successful.
Reloads	The number of times a piece of the object had to be brought back in from disk to the cache, most likely because it was flushed from the shared pool. To see what's happening to Library Cache Reloads in a given datasource, you can add this statistic to your Trends view.
Invalidations	The number of times objects in this namespace were marked invalid because a dependent object was modified.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

NOTE: An overall high library cache hit ratio is a desirable thing. You should strive for a hit ratio between 95-100%, with 99% being a good performance benchmark for code reuse. When a database is first started, the library cache hit ratio, along with the individual region hit ratios, will not be at an optimal level because all code being used will be relatively new, and as such, must be parsed and placed into the shared pool. If, however, after a solid hour or two of steady database time, the library cache hit ratio has not increased to desirable levels, you should look into the possibility of increasing the shared_pool_size parameter.

To keep important code objects from being aged out of the library cache, you can use the DBMS_SHARED_POOL package to pin frequently used code objects in memory so they will always be there when needed.

Leading Sessions Tab

- Metrics

Frequently one or two users cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Sometimes, user connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system levels.

Embarcadero Performance Center displays information to help you find processes that are using the most memory on the server on the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view.

The table below describes the information available on the tabs for all supported database platforms:

Column	Description
User Name	The logon name the session is using.
SID	The unique Oracle identifier for the session.
Serial #	The serial number assigned to the session.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Machine	The name of the client machine name that the session is using.
PGA	The Program Global Area (PGA) is a private memory area devoted to housing the global variables and data structures for a single Oracle process, in KB.
UGA	The User Global Area (UGA) contains session specific information regarding open cursors, state information for packages, database link information, and more. When using Oracle's Multi-threaded Server (MTS), the UGA can be moved up into the SGA, in KB.
Memory Sorts	The number of memory sorts a session has performed.
Total Memory	The amount of memory (PGA + UGA) that the session is consuming, in KB.

NOTE: This information is available on both the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If your database server does not have an overabundance of memory, you should periodically check two things:

1 Who are your heavy memory users

2 The total percentage of memory each user consumes

If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing.

Objects in Memory Tab

- Metrics

The Objects in Memory tab of the Memory Detail view shows objects currently in the library cache. This view lets you see exactly which objects are in the shared pool as well as their resource usage and activity levels.

The table below describes the information available on the Objects in Memory tab of the Memory Detail view:

Column	Description
Object Owner	The user account that owns the object.
Object Name	The name of the object.
Type	The type of object: INDEX, TABLE, CLUSTER, VIEW, SET, SYNONYM, SEQUENCE, PROCEDURE, FUNCTION, PACKAGE, PACKAGE BODY, TRIGGER, CLASS, OBJECT, USER, or DBLINK.
Sharable Memory	The amount of memory consumed by the object in the shared pool.
Loads	The number of times the object has been loaded into the cache. This count increases when an object has been invalidated.
Executions	The number of times that the object has been executed by a session thread.
Locks	The number of users actively locking the object.
Pins	The number of user actively pinning the object.
Kept	Whether or not the object has been pinned in memory with the DBMS_SHARED_POOL package.

NOTE: This information is available on both the Objects in Memory tab of the Memory Detail view and the Objects in Memory tab of the Objects Detail view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If you see objects with large numbers of loads, this could signal that user sessions frequently use the objects causing the objects to prematurely age out of the cache. To fix this you could increase the size of the shared pool and/or pin the objects with the DBMS_SHARED_POOL package so they do not age out of the library cache.

Sessions Overview Tab

- Metrics

You can get more detail into dips in the overall buffer cache hit ratio by examining user I/O activity and users' individual cache hit ratios. Frequently you can pinpoint one or more accounts responsible for reducing performance and examine their SQL and other statistics to fix the situation.

The table below describes the information available on the Sessions Overview tab of the Memory Detail view:

Column	Description
Username	The logon name the session is using.
SID	The unique Oracle identifier for the session.
Serial #	The serial number assigned to the session.
Status	The status of the session, ACTIVE or INACTIVE.
Machine	The name of the client machine that the session is using.
O/S ID	The unique operating system identifier for the target session.
Logon Time	The timestamp when the session first logged on.
Blocked	Whether the session is blocked from doing work by another session.
Seconds Idle	The number of seconds since the last time the session actively performed work.
Hit Ratio	The percentage of times the session obtained data from memory vs. physical I/O activity. The maximum value is 100%.

NOTE: This information is available on both the Sessions Overview tab of the Memory Detail view and the Sessions Overview tab of the Users Detail view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Shared Pool Tab

What are the major components of the shared pool and how much memory does each take up? This question can be answered by viewing the Shared Pool section. You can easily see how much free memory is left in the pool as well as how much space all the other components consume. The table below describes the information available in this section:

Column	Description
Component	The major component of the SGA's shared pool.
Memory	The amount of memory the component is currently using, in bytes.
Percent of Shared Pool	The percentage of the shared pool the component uses.

I/O Statistics

The I/O home page includes the following sections:

- Archive Files Written Today
- Active Jobs
- Active Rollbacks
- Consistent Reads
- Index Scans
- Logical Changes

- Logical Reads
- Long Table Scans
- Physical Reads
- Physical Writes
- Redo Log Size
- Redo Wastage
- Redo Writes
- Rollback Gets
- Rollback Waits
- Session Leaders - I/O
- Short Table Scans
- User Commits
- User Rollbacks

Related Topics

I/O Detail

Home Page Statistics

Memory Page Statistics

Objects Page Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Users Page Statistics

Network Statistics

Contention Statistics

Archive Files Written Today

- Metrics
- Troubleshooting

To allow for point-in-time recovery, Oracle writes copies of redo log information to disk. When a database is running in archive log mode, a DBA can (with proper backup techniques in place) recovery nicely from a database error and roll forward to almost any point in time needed, as long as the proper archive logs are in place.

Oracle's ARCH process handles the I/O needed to write these archive logs. The archive files written today statistic provides a count of how many archive files have been written for the current date.

> **TIP:** Click this statistic to drill down to the Archive view.

Metrics

Archive files written more than one per hour or half-hour could indicate a too small redo size (or above average data modification load).

Troubleshooting

If the redo log size is too small for the database, you can redefine to a larger size. You must take care when performing this job and consult the Oracle administrator documentation exact procedures on altering the redo log file size.

User Object Accesses

This section of the I/O performance category view details some of the most important statistics relating to object accesses. The table below describes the statistics:

Statistic	Description
Long Table Scans	The number of scans of tables consisting of five or more data blocks. This statistic displays on both the I/O performance category view and the Objects performance category view. For more information on long table scan rows, see Long Table Scan Rows.
Short Table Scans	The number of scans of tables consisting of fewer than five data blocks.
Index Scans	The number of object access by a table's ROWID - usually accomplished through an index. For more information on index scans, see Index (ROWID) Accesses.
User Commits	A user commit, which normally signaling the end of a transaction, causes the LGWR process to write modified blocks to disk.
User Rollbacks	The number of times the user transaction manually issued the times the ROLLBACK SQL statement. Click this statistic to drill down to the Rollback Activity tab of the I/O Detail view.
Active Jobs	The number of active jobs, which are scheduled through the use of the Oracle job queue, indicate how many jobs are currently running in the queue. Click this statistic to drill down to the Active Jobs tab of the I/O Detail view.

Metrics

Depending on the statistics found, here are some general user I/O guidelines:

- Avoid unnecessary scans on large tables. Also, high counts of large table scans are a signal to you as a DBA to both investigate the use of more indexes and to review SQL access through EXPLAIN plans.
- Small table scans are actually a good thing because Oracle can often cache the entire table in a single I/O operation.
- Large numbers of index scans are normally desirable too, because they typically indicate the fastest possible resolution to data access requests.
- Keeping an eye on user commits helps give you one of the truest transaction rate indicators in your database.
- Large numbers of user rollbacks can be undesirable because they indicate that user transactions are not completing for one reason or another.
- An increasing number of active jobs can indicate a backup of job activity in the Oracle job queue.

Troubleshooting

Listed below are some methods you can use to avoid unnecessary large table scans:

- Try not to use SQL statements that include the NOT IN, NOT LIKE, <>, IS NULL operators because they typically suppress the use of indexes.
- When referencing concatenated indexes with queries, be sure that you use the leading column in the index. If you do not, the index will not be used at all.
- Avoid using functions in WHERE predicates. If you must use functions, and you are using Oracle8i, investigate the use of function-based indexes.

If you must use table scans, try and make use of the following Oracle features to increase scan performance as much as possible:

- Parallel query
- Partitioning
- Table CACHE parameter
- KEEP buffer pool (Oracle8)

Active Rollbacks

- Metrics

Oracle writes data to individual rollback segments to undo changes made to the Oracle database from within a transaction. You can also use these to maintain read consistency for multiple users of modified data. Because Rollback Segments read and write data, they can become very hot areas for I/O. This statistic is a count of how many rollback segments are actively involved in transactions for the database.

NOTE: This statistic displays on both the I/O performance category view and the Objects performance category view.

TIP: Click this statistic to drill down to the Rollback Activity tab of the I/O Detail view.

Metrics

When you drill down into this statistic and see that most or all rollback segments are involved in transactions, you should consider adding more rollback segments to the system.

Session Leaders - I/O

- Metrics

Heavy I/O activity in a system sometimes indicates that the user connections are contributing somewhat equally to the overall load. More often than not, however, one or two user connections are responsible for 75% or more of the I/O activity. It could be that your system is running a large batch load or other typical processes, which are perfectly okay. On the other hand, it could be a runaway process or other rogue connection, which you need to track down and possibly eliminate.

The leading I/O session's display lets you see who the leading sessions are in your system with respect to I/O.

NOTE: This statistic displays on both the Users performance category view and the I/O performance category view.

TIP: Click this statistic to drill down to the Leading Sessions tab of the I/O Detail view.

Metrics

One or two users consuming more than 75% of the total I/O load could indicate a runaway or improper process. You can drill into the I/O activity of all users and quickly see if this is the case.

Redo Wastage and Redo Log Size

- Metrics
- Troubleshooting

Oracle redo logs are hotbeds of I/O activity in databases that store heavily modified data. Oracle uses redo log files to perform database recovery in the event of a system crash and it writes redo logs in a cyclical fashion: each log file is filled up before Oracle moves on to the next file. The Redo Log Size statistic reflects the total amount of redo generated in bytes since the last refresh. Redo Wastage indicates the number of bytes that were wasted by redo log buffers being written before they were full.

> **TIP:** Double-click these statistics to drill down to the LGWR/DBWR Detail tab of the I/O Detail view.

Metrics

A high percentage of wasted bytes to overall redo bytes can help you identify if redo wastage is a problem on your system.

Troubleshooting

Sometimes heavy redo wastage occurs when the log_checkpoint_interval parameter is set too high. If you think this is the case for your system, then:

- Edit the Init.ora file for the database.
- Increase the amount of log_checkpoint_interval to a lower value.
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor new value to see if performance improves.

Rollback Gets and Rollback Waits

- Metrics

Oracle writes data to individual rollback segments to undo changes made to the Oracle database from a transaction. You can also use these segments to maintain read consistency of data modified by multiple users. Because rollback segments read and write data, they can become very hot areas for I/O.

A count of rollback gets defines the number of times a process went to a rollback header to glean information. The count of rollback writes is the number of bytes written to all rollback segments.

> **TIP:** Double-click these statistics to drill down to the Rollback Activity tab of the I/O Detail view.

Metrics

Taking the number of rollback writes and dividing them by the number of rollback gets and waits yields the overall rollback contention ratio, which indicates how bad contention is for rollback segments. A count greater than 1% typically means that you should add more rollback segments to the database.

Physical I/O

Physical I/O consists of Oracle going to disk to gather or write data. The database writer (DBWR) and log writer (LGWR) processes typically perform all the I/O work in the database. You can also use other processes like the checkpoint and archive processes (CKPT and ARCH). Embarcadero Performance Center shows three key indicators of physical I/O. The table below describes these indicators:

Indicator	Description
Physical Reads	Physical Reads is the total number of physical reads performed on all datafiles since the last refresh.
Physical Writes	Physical Writes is the total number of times the DBWR process has performed writes to various database datafiles since the last refresh
Redo Writes	Oracle's redo logs are hotbeds of I/O activity in databases that store heavily modified data. Redo log files are used to perform database recovery in the event of a system crash. Redo logs are written to in a cyclical fashion - each log file is filled up before Oracle moves on to the next file. The redo writes statistic reflects the total number of redo writes by the LGWR process since the last refresh

NOTE: For more information on physical I/O statistics, see Logical Vital Signs.

TIP: Click this statistic to drill down to the I/O by Tablespace tab of the I/O Detail view.

Metrics

The table below describes metrics for Physical I/O statistics:

Statistic	Metrics
Physical Reads	Large numbers of physical reads could reflect a too small data/buffer cache. The buffer cache hit ratio is a better indicator of overall logical vs. physical I/O.
Physical Writes	Wait events related to I/O activity are good indicators of physical I/O problems. These events include `db file parallel write` and `db file single write`.

Troubleshooting

Doing the following can negate large numbers of continuous physical reads:

- Increasing the size of the data/buffer cache.
- Pinning often-referenced objects in memory by using the KEEP buffer pool (Oracle 8 and higher.)
- Placing heavily scanned objects in larger blocksize tablespaces (16-32KB). For Oracle9i or later.
- Tune SQL statements for better efficiency.

Logical I/O

Logical I/O refers to data access performed in memory. The database writer (DBWR) and log writer (LGWR) processes typically perform all the I/O work in the database. You can also use other processes like the checkpoint and archive processes (CKPT and ARCH). Embarcadero Performance Center shows three key indicators of logical I/O. The table below describes these indicators:

Indicator	Description
Logical Reads	Logical Reads is the total number of db block gets and consistent gets (data read from memory) since the last refresh.
Logical Changes	Logical Changes is the total number of changes that were made to all blocks in the SGA that were part of an update or delete operation. These changes generate redo log entries and are permanent if the transaction is committed. The number of logical changes is an approximate indication of total database work.
Consistent Reads	Consistent Reads is the total number of times a consistent read was requested for a database block. Such a read is performed from Oracle's rollback segments.

NOTE: For more information on logical I/O statistics, see Logical Vital Signs.

TIP: Click this statistic to drill down to the I/O by Tablespace tab of the I/O Detail view.

Metrics

Regarding raw logical I/O counts, no hard-core metrics exist. However, because physical I/O takes longer to complete than logical (memory) I/O, you should minimize physical read operations when possible. The buffer cache hit ratio is a better indicator of overall logical vs. physical I/O.

Troubleshooting

While logical I/O is still up to 1,400 times faster than physical disk access, it would be wise to investigate the top logical I/O process using Performance Center and see what SQL it is executing. If one process on the system is consuming between 25-50% of the overall amount, their SQL might require tuning.

I/O Detail

The following tabbed pages are available on the I/O Detail page:

- I/O by Datafile
- Log I/O - DBWR/LGWR
- Rollback Activity
- I/O by Tablespace
- Leading Sessions
- Active Jobs Tab

Active Jobs Tab

- Metrics

Scheduled using the Oracle job queue, custom batch jobs can be set up to run whenever a DBA would like. The Active Jobs tab of the I/O Detail view presents the active jobs that are running and indicates the number of jobs currently running in the queue.

The table below describes the information available on the Active Jobs tab of the I/O Detail view:

Column	Description
SID	The identifier of the session executing the job.
Job ID	The identifier of the job.
Start Time	The time that the job was started.
Submitted By	The user account that submitted the job.
Run As	The user account whose privileges run the job.
Parse As	The user account that parsed the job.
Next Run	The next scheduled job run time.
Job Contents	The body of the anonymous PL/SQL block that this job executes.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Viewing jobs with long run times could indicate a job caught in a loop or experiencing a problem.

Leading Sessions Tab

- Metrics

When a system undergoes heavy I/O activity, sometimes you find that all the user connections are contributing somewhat equally to the overall load. Frequently, however, one or two user connections are responsible for 75% or more of the I/O activity. It could be that a large batch load or other typical process is running that is perfectly okay for your system. Alternatively, it could be a runaway process or other rogue connection that you should track down and possibly eliminate.

Embarcadero Performance Center displays information to help you find processes that are using the most memory on the server on the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view.

The table below describes the information available on the tabs for all supported database platforms:

Column	Description
Username	The logon name the session is using.
SID	The unique Oracle identifier for the session.
Serial #	The serial number assigned to the session.
Status	The status of the session, ACTIVE or INACTIVE.
Machine	The name of the client machine name that the session is using.
Reads	The number of physical reads.
Writes	The number of physical writes.
Total I/O	The total of all physical I/O operations for the session.

NOTE: This information is available on both the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Pinpointing sessions with abnormally high I/O activity relative to other sessions in the system could aid in ferreting out accounts dragging down overall system performance. You should examine the activity of each session to determine the system workload and to determine if you can reduce the workload or tune the system for better performance.

I/O by Tablespace Tab

- Metrics

Physical I/O consists of Oracle going to disk to gather or write data. Logical I/O refers to data access performed in memory. The database writer (DBWR) and log writer (LGWR) processes typically perform all the I/O work in the database. Other processes like the checkpoint and archive processes (CKPT and ARCH) may also be used.

The Tablespace I/O tab displays details concerning the physical I/O activity at the tablespace level. The table below lists the information available on this tab:

Column	Description
Tablespace	The name of the tablespace.
Physical Reads	The cumulative number of physical reads.
Physical Writes	The cumulative number of physical writes.
Block Reads	The cumulative number of physical block reads.
Blocks Written	The cumulative number of physical block writes.
Read Time	The time spent reading from the tablespace (in hundredths of seconds).
Write Time	The time spent writing to the tablespace (in hundredths of seconds).

Metrics

Generally, you want to see much more logical I/O activity than physical I/O, at least with respect to reads, although this in and of itself is a guarantee of good I/O performance. Seeing logical and physical reads keeping pace with one another is a sure sign that the Oracle SGA's buffer cache is sized too small or that needless, large table scans are occurring, which cause blocks of data to be continually read in and flushed out of the buffer cache.

Other telltale signs of trouble brewing are large amounts of activity visible in user's TEMP tablespaces. The normal interpretation of such a thing is that a large number of disk sorts are taking place (perhaps because the Init.ora/spfile parameter SORT_AREA_SIZE may be set too small).

For more information, see I/O Home Page.

I/O by DatafileTab

- Metrics

Physical I/O consists of Oracle going to disk to gather or write data. Logical I/O refers to data access performed in memory. The database writer (DBWR) and log writer (LGWR) processes typically perform all the I/O work in the database against the physical datafile used to hold information The checkpoint and archive processes (CKPT and ARCH), also perform I/O work in the database.

The Datafile I/O tab displays details concerning the physical I/O activity at the datafile level. The table below lists the information available on this tab:

Column	Description
Datafile	The name of the datafile.

Column	Description
Datafile Status	The availability of the datafile.
Tablespace	The name of the tablespace.
Physical Reads	The cumulative number of physical reads.
Physical Writes	The cumulative number of physical writes.
Block Reads	The cumulative number of physical block reads.
Blocks Written	The cumulative number of physical block writes.
Read Time	The time spent reading from the tablespace (in hundredths of seconds).
Write Time	The time spent writing to the tablespace (in hundredths of seconds).

Metrics

Generally, you will want to see much more logical I/O activity than physical I/O, at least with respect to reads, although this in and of itself is a guarantee of good I/O performance. Seeing logical and physical reads keeping pace with one another is a sure sign that the Oracle SGA's buffer cache is sized too small or that needless, large table scans are occurring, which cause blocks of data to be continually read in and flushed out of the buffer cache.

Other telltale signs of trouble brewing are large amounts of activity visible in user's TEMP tablespaces. The normal interpretation of such a thing is that a large number of disk sorts are taking place (perhaps because the Init.ora/spfile parameter SORT_AREA_SIZE may be set too small). One last thing to watch is high amounts of activity on datafiles housed on the same directories or file systems. This usually indicates a "hot" drive and could represent disk contention. If you see one or more hot drives, you should consider relocating one or more datafiles to lesser-used physical devices.

For more information, see I/O Home Page.

Rollback Activity Tab

The Rollback I/O tab includes the following sections:

- Rollback Activity
- Rollback/Optimal Detail

For more information, see I/O Home Page.

Rollback Activity Grid

- Metrics

To undo changes made to the Oracle database from within a transaction, Oracle writes data to individual rollback segments. Oracle also uses these segments to maintain read consistency for multiple users of data that is being modified. Because Oracle reads from and writes data to rollback segments, they can become very hot areas for I/O.

The Rollback Activity Detail grid presents everything necessary to view and troubleshoot rollback problems. The first grid displays rollback activity details and the second grid displays rollback/optimal details. The table below describes the information available in the Rollback Activity Detail grid on the Rollback Activity tab of the I/O Detail view:

Column	Description
Name	The name of the rollback segment.
Size	The size of the rollback segment in KBs.
Shrinks	The number of times the rollback segment has decreased in size.

Column	Description
Extends	The number of times the rollback segment has increased in size.
Gets	The number of header gets (the segment has been used).
Waits	The number of header waits.
Writes	The number of bytes written to the rollback segment.
Active	Indicates whether the rollback segment is active (non zero) or not (zero value).
Status	Indicates the status of the rollback segment, with the two main results being OFFLINE (a segment is offline and unavailable for transactions) and ONLINE (a rollback segment is online and available for transactional use).
High Water Mark	The largest size that the rollback segment has ever grown to.

Rollback/Optimal Detail Grid

- Metrics

The lower grid presents data on each rollback segment's current status with respect to its optimal size setting (the DBA-imposed size that the segment "should" be).

The table below describes the information available in the Rollback/Optimal Detail grid on the Rollback Activity tab of the I/O Detail view:

Column	Description
Rollback Name	The name of the rollback segment.
Size	The size of the rollback, in bytes.
Optimal Size	The optimal setting for the rollback segment. NULL if the DBA has not imposed an optimal setting on the rollback segment.
High-Water Mark	The largest size to which the rollback segment has ever grown, in bytes.
Pct Above Optimal	The percentage above the optimal setting to which the rollback segment ever grown.

NOTE: If a rollback segment has no OPTIMAL setting, it has NULL values for most values in the Rollback/Optimal Detail grid.

TIP: To configure a grid to display row numbers, use the Options Editor.

Metrics

To properly tune rollback I/O, you must first make sure that you have enough segments to accommodate the workload of the database. Constantly seeing a count of active rollback segments equal to or near the number of rollbacks defined for the database is an indicator that you should create more. An overall rollback contention ratio of 1% or greater is an indicator of too few rollbacks. Seeing wait counts greater than zero for each rollback segment is further evidence that you should create more rollback segments. Oracle9i provides the UNDO tablespace to automatically generate and eliminate the ‘correct’ number of rollback segments for a system given a certain workload.

After ensuring that enough rollback segments exist in the database, you should then turn your attention to the question of sizing. Dynamic rollback extension can take a toll on performance when rollback segments are consistently enlarged to accommodate heavy transaction loads. Seeing rollback segments undergoing numerous extends and shrinks (as Oracle returns a segment back to its OPTIMAL setting), as well as rollback segments having current or high-water mark sizes greater than their OPTIMAL setting usually is a good indicator that they should be permanently enlarged. Again, Oracle9i’s automatic undo management can assist in this process.

DBWR/LGWR Tab

The DBWR/LGWR tab includes Database Writer (DBWR/LGWR) detail and Redo Log detail.

For more information, see I/O Home Page.

Database Writer Detail

- Metrics

The database writer process (DBWR) handles the flow of information from Oracle's physical datafiles to and from the various memory structures in the system global area (SGA). On platforms that support it, you can configure and use multiple DBWR processes. The log writer (LGWR) process manages the information contained in Oracle's online redo log files and redo log buffer area. On the LGWR/DBWR tab of the I/O Detail view you can view statistics relating specifically to both of these critical Oracle background processes and see if they are running efficiently. The table below lists the information available in this section:

Column	Description
Statistic	The specific metric for the database writer.
Value	The value for the statistic.

Metrics

Of all the statistics presented for the DBWR process, the summed dirty queue length statistic deserves attention. Non-zero values typically indicate buffers left in the write queue after a write request and may indicate that the DBWR process is falling behind. For the LGWR process, nonzero values for the redo log space requests and redo log space wait time statistics could be a cause for concern. Redo log space requests reflect the number of times a user process waited for space in the redo log buffer, while the redo log space wait time is the total wait time in milliseconds. Both could indicate the presence of contention in the redo log buffer. Possible remedies include increasing the LOG_BUFFER size in the SGA.

Redo Wastage

Oracle's redo logs are hotbeds of I/O activity in databases that store heavily modified data. Redo log files are used to perform database recovery in the event of a system crash. Redo logs are written to in a cyclical fashion - each log file is filled up before Oracle moves on to the next file. The Redo Wastage section shows how many bytes were wasted by redo log buffers being written before they were full.

Metrics

Viewing a high percentage of wasted bytes to overall redo bytes can help you identify if redo wastage is a problem on your system.

Steps

Sometimes heavy redo wastage occurs when the log_checkpoint_interval parameter is set too high. If you think this is the case for your system, then

1. Edit the Init.ora file for the database.
2. Change the amount of log_checkpoint_interval to a lower value.
3. Cycle the Oracle server when possible to allow the new value to take effect.
4. Monitor new value to see if performance improves.

Space Statistics

The Space performance category view displays the following vital Oracle space statistics:

- Fragmentation Leaders
- Schema Leaders - Space
- Tablespace Overview

Related Topics

Space Detail

Home View Statistics

I/O Page Statistics

Memory Page Statistics

Objects Page Statistics

Users Page Statistics

Network Statistics

Contention Statistics

Tablespace Overview

- Metrics
- Troubleshooting

The Oracle tablespace is the logical container for the physical storage needs of a database. The tablespace overview displays details for all the tablespaces for a particular database, including their total, used, free space, and percent free, as well as if each can automatically extend to support incoming space requests.

> **TIP:** Click this statistic to drill down to the Space Usage tab of the Space Detail view.

Metrics

If any tablespace's free space percent amount goes below 10%, and at least one tablespace's datafiles does not have AUTOEXTEND enabled (or the datafile has reached its extend limit), you should take action to ensure that the tablespace does not run out of available free space.

Troubleshooting

There are two things you can do to ensure that a tablespace does not run out of available free space:

1 First, you should look into the use of Oracle's AUTOEXTEND feature. AUTOEXTEND lets you give an Oracle tablespace the ability to auto-grow when it has exhausted the free space contained within. You can let a tablespace grow in an unlimited fashion or put constraints on it to stop at a certain point. You can also dictate how much more free space the tablespace gets each time it needs more space than is available. However, AUTOEXTEND enabled for a tablespace does not mean that you cannot run out of space. Remember you still have the physical server limitations to contend with. Make sure you (or your sysadmin) keep a careful eye on the server drives that house your Oracle database files for available free space.

2 If the free space on a server drive nears its limit, disable AUTOEXTEND for the datafile(s) that are on the target drive, and use the ALTER TABLESPACE … ADD DATAFILE command to place a new datafile for the tablespace on another drive that has more free space to offer.

TIP: AUTOEXTEND is not a replacement for proactive space planning. When extra space is needed by the database, and AUTOEXTEND is activated by Oracle, performance slows as Oracle allocates more space for the tablespace. Avoiding AUTOEXTEND aids performance, albeit in a small way.

Fragmentation Leaders

The Fragmentation Leaders table shows the tablespaces datafiles suffering from the highest levels of free space fragmentation in the database. Tablespaces are made up of object segments and space extents. Extents are either allocated to object segments or are free. When a tablespace is initially populated, all objects are neatly packed together in the front of the tablespace and all remaining free space is in one free chunk at the end. As objects grow (or extend) they are given new extents of space in the tablespace. If objects are dropped, pockets of free space begin to appear throughout the tablespace. These pockets of space take one of two forms. The table below describes these forms:

Free Space	Description
Honeycombs	Pockets of free space that are adjacent to one another.
Bubbles	Pockets of free space that are trapped between object extents in the tablespace.

TIP: Click this statistic to drill down to the Fragmentation tab of the Space Detail view.

NOTE: This information is available on both the Fragmentation Leaders table of the Space performance category view and the Fragmentation tab of the Space Detail view.

Metrics

If you see a tablespace that has many chunks of free space, determine if the tablespace is experiencing honeycomb or bubble fragmentation. You can handle honeycomb fragmentation quite easily, whereas bubble fragmentation is more difficult to solve.

Troubleshooting

You can eliminate honeycomb fragmentation with the ALTER TABLESPACE…COALESCE command. Issuing this command combines all pockets of adjacent free space into one extent. Each database maintenance plan should include a job that coalesces all the free honeycombs in a tablespace into one free chunk. Although Oracle is supposed to perform this operation automatically through the SMON process, it requires you to have the PCTINCREASE parameter of the tablespace set to a nonzero value. Having PCTINCREASE set to a value greater than zero encourages tablespace fragmentation through disparately sized extents. Plus, using SMON in this way is not efficient or entirely reliable.

You can temporarily solve bubble fragmentation by performing a total tablespace reorganization. A better long-term solution for Oracle databases 8.1.5 or later is to convert tablespaces over to locally managed tablespaces. With locally managed tablespaces you either specify the initial extent size and let Oracle automatically size all other extents, or specify a uniform extent size for everything. Problems caused by fragmentation then become a thing of the past.

To help stave off fragmentation problems:

- Set PCTINCREASE to zero for all tablespaces and objects to promote same-sized extents.
- Specify equal-sized allotments for your INITIAL and NEXT object storage parameters.
- Group objects with like growth and storage needs together in their own tablespaces.

You should also avoid fragmentation in the SYSTEM tablespaces. The best ways to do this include:

- Ensure no user has a DEFAULT or TEMPORARY tablespace assignment of SYSTEM.
- Ensure no user has a quota set for SYSTEM.
- Ensure no user has been granted the UNLIMITED TABLESPACE privilege.

Schema Leaders - Space

The Space Leaders chart details the schemas in the database that own the most space. In databases where object owners can be many, it is frequently a good idea to take a quick look at which schemas are leading the way in terms of space usage. This is not restricted to production environments only, but can also be extended to dynamic development systems. This is useful when you have many developers in a database that have their own sets of objects. Developers sometimes have a bad habit of creating objects (or copies of objects) that they leave in a database even though they are not using them.

> **TIP:** Click this statistic to drill down to the Space Usage tab of the Space Detail view.

Metrics

None.

Troubleshooting

If you see a user who owns many objects, or copies of objects, reclaim space in your database by contacting the user to see if those objects can be dropped.

Space Detail

The following tabbed pages are available on the Space Detail page:

- Fragmentation Tablespace Growth
- Extents Deficits

- Object Summary
- Space Usage
- Tablespace Map

Tablespace Map Tab

- Metrics

Tablespaces comprise object segments and space extents. Extents are either free inside a tablespace or allocated to object segments. When a tablespace is initially populated, Oracle neatly packs all objects together in the front of the tablespace and all remaining free space is in one free “chunk” at the end. Unfortunately, this is not how things continue to be in a tablespace. As objects grow, or extend, the database gives them new extents of space in the tablespace. If you drop objects, pockets of free space begin to appear throughout the tablespace. These pockets of space are either honeycombs or bubbles.

The Tablespace Map tab of the Space Detail view lets you see all the extents each object in a tablespace takes up and all the free extents. The map is in block id order, letting you view the tablespace from beginning to end. The table below describes the information available on the Tablespace Map tab of the Space Detail view:

Column	Description
Owner	The owner of the object. NULL if free space occupies the extent.
Object Type	Type of object. NULL if free space occupies the extent.
Object Name	The name of the object occupying the extent. NULL if free space occupies the extent.
Block ID	The starting block number of the extent.
Bytes	The amount of space consumed by the extent, in bytes.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

The tablespace map identifies honeycomb problems with two adjacent chunks of free space. You can eliminate honeycomb fragmentation with the ALTER TABLESPACE…COALESCE command. This command combines all pockets of adjacent free space into one extent. This is important because when an object is newly introduced to a tablespace (or an existing object needs to extend), and a contiguous chunk of free space does not exist to accommodate an object's INITIAL or NEXT size allocation, Oracle must manually coalesce all available honeycombs to try and form a large enough free chunk. This is a performance hit. If possible, you should try to minimize performance hits.

Bubble fragmentation is a more serious matter and is normally only corrected through tablespace or database reorganization. The standard technique is to perform an export of all objects in the tablespace, drop the tablespace, and then import all the objects back into the tablespace. However, this technique may just treat the symptom and not the cause of bubble fragmentation. The real issue is to address the reuse of space extents within a tablespace so that bubble fragmentation does not occur in the first place.

Oracle8 offers the concept of locally managed tablespaces, which can all but eliminate tablespace fragmentation. It totally does away with the storage parameters of MINEXTENTS, MAXEXENTS, PCTINCREASE, and NEXT. With locally managed tablespaces you either specify the initial extent size and let Oracle automatically size all other extents, or specify a uniform extent size for everything. Problems caused by fragmentation then become outdated.

A few suggestions to manually to help stave off fragmentation problems include:

- Set PCTINCREASE to zero for all tablespaces and objects to promote same-size extents.
- Specify equal-size allotments for your INITIAL and NEXT object storage parameters.

- Group objects with like growth and storage needs together in their own tablespaces.

The SYSTEM tablespace is the major hotbed tablespace for Oracle activities. Therefore, more than any other tablespace, you should avoid fragmentation problems in your SYSTEM tablespace. The easiest way to avoid this is to not allow any user (even the default DBA ID's SYS and SYSTEM) to have access to it. There are three ways to accomplish this:

1 Ensure no user has a DEFAULT or TEMPORARY tablespace assignment of SYSTEM.

2 Ensure no user has a quota set for SYSTEM.

3 Ensure no user has been granted the UNLIMITED TABLESPACE privilege.

Object Summary Tab

- Metrics

What takes up space in a database? Simply, objects. In an Oracle database, the primary users of space are tables, indexes, and rollback segments. Under certain circumstances, other objects like materialized views and snapshots can come into play. With respect to space, as a DBA you should keep track of your object's growth and space characteristics. As with tablespaces, you want to keep an eye on fragmentation as well, unless locally managed tablespaces are being used (Oracle 8.1.5 or later).

The Object Summary tab of the Space Detail view displays object sizing information for selected tablespaces. The table below describes the information available on the Object Summary tab of the Space Detail view:

Column	Description
Owner	The user account who owns the object.
Object	The object name.
Object type	The type of object.
Kilobytes	The size of the object, in KB.
Extents	The number of extents used by the object.
Percent of tablespace used	The space in the tablespace consumed by the object, as a percentage.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

To avoid object growth errors in your database, you should consider setting the MAXEXTENTS parameter for your objects (which acts as a growth limit for the object) to a very high value or UNLIMITED if allowed by your version of Oracle. This lets you allocate as many extents as the objects need for growth. It is a good practice to monitor the amount of space your objects use and the number of extents they possess. This clues you in to what the dynamic objects in your database are.

One thing you can face with respect to allowing this type of unchecked extent growth is that your objects can reach a point of needing a new extent of space, but not enough free space exists in its parent tablespace to accommodate the enlargement request. It is good practice to periodically sweep your database for objects that can encounter this error on their next extension.

Understand that allowing unrestricted extent growth can save you from extent allocation errors, but introduces object fragmentation. What can you do if you have objects with high numbers of extents and are not using locally managed tablespaces? For tables and indexes, you have three options:

1. Use Oracle's export/import utility to export, drop, and then re-import the fragmented objects back into the database with the export parameter COMPRESS=Y. This brings the object back into the database with one large extent. Take care that large enough free chunks of available free space exist to accept the objects back in, or you can experience allocation errors.
2. With Oracle8, you can use the ALTER TABLE…MOVE command to reorganize a table back into one extent in a very safe and efficient manner.
3. Use ALTER INDEX…REBUILD to reorganize indexes that have moved into multiple extents.

For rollback segments, you can specify an OPTIMAL setting for each rollback segment. Setting OPTIMAL for the rollback gives the segment the opportunity to shrink back to whatever extent limit you would like to keep the rollback segment. The database periodically does this, but you can manually issue the ALTER ROLLBACK…SHRINK command.

Tablespace Growth Tab

- Metrics

Growing tablespaces do not have to spell problems for a DBA. Of course, up front planning is the key to sizing tablespaces correctly. Unfortunately, DBAs may not have all the information they need at tablespace creation time (or they may have the wrong information), so a tablespace can approach the end of its free space from time to time. The DBA can allow a tablespace to grow automatically (AUTOEXTEND) to prevent an out-of-space condition. Enabling AUTOEXTEND for a tablespace is quite reassuring for a DBA, but it introduces a new concept for the tablespace: monitoring data file growth. You should monitor your tablespaces that have AUTOEXTEND set so you can get an idea of the growth that is occurring in your database. Monitoring them lets you perform some mini-capacity planning and helps you get a jump-start on server upgrades.

The Tablespace Growth tab displays growth statistics for the database currently being monitored. The table below lists the information available on this tab:

Column	Description
Tablespace Name	The name of the tablespace.
Datafile Name	Thee name of the tablespace's datafile.
Auto Extend?	Indicates whether the tablespace has one or more datafiles that have the AUTOEXTEND feature set (allowing the datafile to automatically grow).
Original Size	The starting size for the physical datafile.
Current Size	The current size of the physical datafile.
Growth	The percentage of growth experienced for the datafile (if any).

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Using AUTOEXTEND can be especially important for tablespaces devoted to disk sorting. Large data warehouses often must endure large disk sort operations. Having AUTOEXTEND enabled for tablespaces used for temporary segments (sort activity) helps large sort queries to complete when they might otherwise fail due to running out of sort space.

Just having AUTOEXTEND enabled for a tablespace does not mean you cannot run out of space. Remember, you still have to contend with the physical server limitations. Make sure you (or your sysadmin) keep a careful eye on the server drives that house your Oracle database files for available free space. If the free space on a server drive nears its limit, disable AUTOEXTEND for the datafile(s) that are on that drive, and use the traditional ALTER TABLESPACE…ADD DATAFILE command to place a new datafile for the tablespace on another drive that has more free space to offer.

Fragmentation Tab

- Metrics
- Troubleshooting

Tablespaces are made up of object segments and space extents. Extents are either allocated to object segments or are free. When a tablespace is initially populated, all objects are neatly packed together in the front of the tablespace and all remaining free space are in one free chunk at the end. Unfortunately, this is not how things continue to be in a tablespace. As objects grow (or extend) they are given new extents of space in the tablespace. If you drop objects, pockets of free space begins to appear throughout the tablespace. These pockets of space are either honeycombs or bubbles. Honeycombs are not so difficult to deal with, but bubbles are another story.

The Fragmentation tab displays two different views of tablespace fragmentation. The first grid displays fragmentation at the tablespace level and the second grid displays fragmentation at the datafile level. The table below describes the information available in the Tablespace Fragmentation grid:

Column	Description
Tablespace Name	The tablespace name.
Free Space (MB)	The total amount of free space in MB for the tablespace.
Free Chunks	The total number of free chunks in the tablespace.
Largest Chunk (MB)	The largest free chunk (in MB) for the tablespace.

NOTE: Embarcadero Performance Center displays the fragmentation grid in two places, the Space performance category view and the Space Detail view.

The table below describes the information available in the Datafile Fragmentation grid on the Fragmentation tab:

Column	Description
Tablespace	The tablespace name.
Datafile	The name of the datafile.
Datafile Status	The status (AVAILABLE or INVALID) of the individual data file. A status of INVALID means that the file number is not in use. This could happen to a file that was part of a tablespace that was dropped.
Free Space	The amount of free space for the datafile, in MB.
Free Chunks	The number of free chunks in the datafile.
Largest Chunk	The largest free chunk for the datafile, in MB.

Column	Description
Datafile	The name of the datafile.
Autoextend	Indicates whether the datafile can automatically grow in size.
Tablespace	The tablespace name.
Free Chunks	The number of free chunks in the datafile.
Largest Chunk	The largest free chunk (in MB) for the datafile.

NOTE: This information is available on both the Fragmentation Leaders table of the Space performance category view and the Fragmentation tab of the Space Detail view.

TIP: To configure a grid to display row numbers, use the Options Editor.

Metrics

To spot and correct fragmentation in your tablespaces, you should periodically monitor the fragmentation levels of your tablespaces at a global level. Doing so helps you quickly spot tablespaces that are experiencing fragmentation issues. Seeing a tablespace with only one chunk of free space is a sign that a tablespace is not having fragmentation problems. Seeing a tablespace with a couple of free chunks may not be a big deal either, because the tablespace could be made up of more than one datafile. Each datafile has its own chunk or chunks of free space.

If you see a tablespace that has many chunks of free space, the next thing to do is drill down into it and find out if the tablespace is experiencing honeycomb or bubble fragmentation. Honeycomb fragmentation occurs when pockets of free space exist that are adjacent to one another. Bubbles are pockets of free space that are trapped between object segment extents.

You can eliminate honeycomb fragmentation with the ALTER TABLESPACE...COALESCE command. This command combines all pockets of adjacent free space into one extent. It is important to do this because when an object is newly introduced to a tablespace (or an existing object needs to extend), and a contiguous chunk of free space does not exist to accommodate an object's INITIAL or NEXT size allocation, Oracle must manually coalesce all available honeycombs to try and form a large enough free chunk. This is a performance hit. If possible, you should try to minimize performance hits.

Bubble fragmentation is a more serious matter and is normally only corrected through tablespace or database reorganization. The standard technique is to perform an export of all objects in the tablespace, drop the tablespace, and then import all the objects back into the tablespace. However, this technique may just treat the symptom and not the cause of bubble fragmentation. The real issue is to address the reuse of space extents within a tablespace so that bubble fragmentation does not occur in the first place.

Oracle8 and later offers the concept of locally-managed tablespaces, which can all but eliminate tablespace fragmentation. It totally does away with the storage parameters of MINEXTENTS, MAXEXTENTS, PCTINCREASE, and NEXT. With locally managed tablespaces you either specify the initial extent size and let Oracle automatically size all other extents, or specify a uniform extent size for everything. Problems caused by fragmentation then become a thing of the past.

Troubleshooting

What can you do manually to help stave off fragmentation problems? A few suggestions include:

- Set PCTINCREASE to zero for all tablespaces and objects to promote same-sized extents.
- Specify equal-sized allotments for your INITIAL and NEXT object storage parameters.
- Group objects with like growth and storage needs together in their own tablespaces.

Of all your tablespaces, you want to avoid fragmentation problems in your SYSTEM tablespace the most as this is the major hotbed tablespace for Oracle activities. The easiest way to avoid this is to not allow any user (even the default DBA ID's SYS and SYSTEM) to have access to it. There are three ways to do this:

1. Ensure no user has a DEFAULT or TEMPORARY tablespace assignment of SYSTEM.
2. Ensure no user has a quota set for SYSTEM.
3. Ensure no user has been granted the UNLIMITED TABLESPACE privilege.

Extents Deficits Tab

- Metrics

The Extents Deficits tab of the Space Detail view helps you determine if any object in your database is unable to extend into its next extent of space. Sometimes objects have NEXT extent parameters that are too large to fit into the parent tablespace's largest block of free space. For locally managed tablespaces, this is not a problem as the database handles object extent management, but for dictionary-managed tablespaces, the potential for problems exist.

By selecting a tablespace in the Extents Deficits tab, you can find out if any object extent problems are looming on the horizon of your database. The table below describes the information available on the Extents Deficits tab of the Space Detail view:

Column	Description
Owner	The user account that owns the object.
Object Name	The object name.
Object Type	The type of object.
Tablespace Name	The tablespace name.
Next Extent Size	The NEXT extent size of the object in bytes.
Max Contiguous Space	The maximum amount of contiguous space.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

You should ALTER any objects with excessive NEXT extent sizes to reduce the NEXT extent to a manageable size.

If you cannot use locally managed tablespaces to help stave off this problem, manually impose NEXT extent limits for all objects that are well within reason. Additionally, you should set the PCTINCREASE parameter for objects to zero to minimize the potential of stratospheric object growth.

Space Usage Tab

- Metrics

It is vital that a DBA know how the various accounts assigned to a database consume space. This is true in either production or development databases, because even in development databases, space can become an issue that causes major headaches. If development accounts are creating and cloning many objects without removing them, the result can be a lot of wasted space.

The Space Usage tab of the Space Detail view displays solid information to help with understanding database object ownership. The table below describes the information available on the Space Usage tab of the Space Detail view:

Column	Description
Schema	The name of the user account.
Tablespaces	The number of tablespaces in which the user currently has objects.
Spacial Objects Count	The number of space-related objects that the account owns.
Total Space Used	The amount of space that the account owns, in MB.
Percentage of Database	The database space that this account consumes, as a percentage.

Metrics

Seeing owner account with larger than expected data volumes could warrant further investigation. In addition, seeing accounts that have objects in more tablespaces than they have quota's for should also cause you to examine their account more closely to ensure the correct space privileges are in place.

Objects Page Statistics

The Objects performance category view displays the following vital Oracle objects statistics:

- Active Rollback Transactions
- Chained Row Tables
- Chained Row Fetches
- Index (ROWID) Access
- Invalid/Unusable Objects
- Locked Objects
- Long Table Scan Rows
- Object Buffer Pool Placement
- Objects With Space Deficits
- Objects With Extent Problems
- Objects Pinned

- Rollbacks at Optimal
- Rollbacks Beyond Optimal
- Total Row Fetches

Related Topics

Objects Detail

Home View Statistics

I/O Page Statistics

Memory Page Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Users Page Statistics

Contention Statistics

Network Statistics

Object/Buffer Pool Placement

- Metrics
- Troubleshooting

Because data objects can be referenced with different types of usage patterns, Oracle8 offers the option to intelligently place objects into one of three buffer caches. The table below describes the buffer caches:

Buffer Cache	Description
KEEP	Designed to minimize misses in the buffer cache. Small objects that are frequently referenced are candidates for the KEEP buffer pool.
RECYCLE	Designed to avoid having large numbers of infrequently accessed blocks of data crowd out objects that need to be referenced in RAM, the RECYCLE pool is good for large objects that are scanned from beginning to end without the need for keeping all their data in RAM.
DEFAULT	The traditional cache for all other data objects.

NOTE: Unless you specify the KEEP or RECYCLE buffer cache, Oracle automatically places objects into the DEFAULT buffer cache.

TIP: Click this category heading to drill down to the Buffer Pool tab of the Objects Detail view.

Metrics

When looking at overall database I/O activity, you should keep an eye out for objects that you can place into a particular buffer cache. Consider using a KEEP cache for relatively small, frequently accessed tables that require fast response times. Large tables with random I/O activity and are scanned from beginning to end are good candidates for a RECYCLE cache.

Troubleshooting

Objects can be placed into different buffer pools at object creation time (using the STORAGE... BUFFER_POOL option in Oracle8 or later) or existing objects can be set into a different pool with the ALTER command.

NOTE: Tables, partitions, and indexes can be placed into the different caches.

If you just want to use the DEFAULT buffer pool and not enable any special caches, you can still encourage Oracle to keep certain objects in the cache as long as possible using the CACHE parameter. For example, issuing the command ALTER TABLE…CACHE specifies that the blocks retrieved for a table be placed at the most recently used end of the LRU list in the DEFAULT buffer cache when a full table scan is performed. You can also use the CACHE hint in SQL statements to cache a table, but used in this form, the blocks are only cached until the next time the database is shut down.

Chained Table Placement

The chained row fetches statistic represents the rows fetched that were either the chained row type or the migrated row type. The total row fetches statistic represents the total number of rows accessed by user processes.

TIP: Double-click these statistics to drill down to the Chained Tables tab of the Objects Detail view.

Metrics

If you are seeing a high percentage (30% or greater) of data access operations being satisfied by chained rows, you should take positive steps toward the elimination of chained and migrated rows from your tables.

Troubleshooting

The best way to deal with chained and migrated rows is to prevent them from occurring in the first place. The table below describes some ways to do this:

Method	Solution
Use a Large Block Size	Because chaining and migrations occur when the Oracle block is too small to hold the rows in question, make sure you are using a large enough block size for each database you create. An 8 KB or greater block size is normally recommended to help stave off chained and migrated rows.
Use Proper Values of PCTFREE for Tables	The necessary amount of percent free in a table is what helps prevent row migrations from occurring. If you have a database that houses rows with the potential to grow substantially over their initially inserted size, you can provide a liberal amount of PCTFREE for each of your tables.

If chaining or migrations are already on the rise in your database, you can take steps to counter the attack. Chaining is usually hard to resolve because rows are too big for the originally specified block size. A rebuilt database is normally the only cure for them. Migrations are another story. You can usually back up the rows in your migrated-row tables, re-create them with larger amounts of PCTFREE, and then bring the rows back in. The ANALYZE…LIST CHAINED ROWS command is useful for locating chained and migrated rows in a table.

Active Rollback Transactions

- Troubleshooting

To undo changes made to the Oracle database from within a transaction, Oracle writes data to individual rollback segments. You can also use these to maintain read consistency for multiple users of modified data. Because Rollback Segments read and write data, they can become very hot areas for I/O. This statistic is a count of how many rollback segments are actively involved in transactions for the database.

NOTE: This statistic displays on both the I/O performance category view and the Objects performance category view.

TIP: Click this statistic to drill down to the Rollback Activity tab of the I/O Detail view.

Metrics

When you drill down into this statistic and see that most or all rollback segments are involved in transactions, you should consider adding more rollback segments to the system.

Troubleshooting

If you are using Oracle8i or earlier, begin by creating new rollback segments and altering them to be online for use. Then monitor the overall active rollback ratio to see if it begins to drop.

For Oracle9i or later, consider the use of UNDO management, which is where Oracle itself automatically manages rollback segments in special UNDO tablespaces.

Index (ROWID) Accesses and Long Table Scan Rows

- Metrics
- Troubleshooting

Encouraging index access vs. full table scans on large tables is something that is always sought after. Large tables typically make up more than five data blocks, and are usually accessed better with indexes. Small tables (those with five or fewer data blocks) are surprisingly accessed more efficiently in a full table scan, because Oracle usually caches the entire table in the buffer area of the SGA. The Index (ROWID) statistic counts the number of rows in tables accessed during index access. The Table Scan Rows statistic counts the number of rows accessed for table scan operations.

Metrics

The subject of proper indexing schemes is beyond the scope of this document, but as a DBA you should keep an eye on the counts of index accesses vs. table scans. Mounting counts of large table scans are indicative of a too restrictive indexing policy, and hints that you should revisit the physical design of the database.

Troubleshooting

Here are some ways to avoid unnecessary large table scans:

- Try not to use SQL statements that include the NOT IN, NOT LIKE, < >, IS NULL operators because they typically suppress the use of indexes.
- When referencing concatenated indexes with queries, be sure the leading column in the index is used. If it is not, the index will not be used at all.
- Avoid using functions in WHERE predicates. If functions must be used, and you are using Oracle8i, investigate the use of function-based indexes.

If table scans must be used, try and make use of the following Oracle features to increase scan performance:

- Parallel query

- Partitioning
- Table CACHE parameter
- KEEP buffer pool (Oracle8)

Just adding indexes to a table does not necessarily guarantee increased performance. For normal b-tree indexes to be effective, the selectivity of the column(s) being indexed must be high. Low selectivity columns are good candidates for bitmap indexing.

With either index, make sure that the additional indexes do not overly penalize data modification performance. This can especially be the case with bitmap indexes.

Chained Row Tables

- Metrics
- Troubleshooting

In normal circumstances, a row of data should fit completely inside one Oracle block. Sometimes, however, this is not the case and the table suddenly contains chained or migrated rows (rows that span more than one data block).

Chaining occurs when a row is initially too large to fit inside one block. Two or more blocks are used by Oracle to hold the row. Migration deals with rows that have grown so much that they can no longer be contained within their original block. When this occurs, Oracle relocates the row out of its original block into another block, but leaves a pointer behind to indicate the relocation.

Both chaining and migration force Oracle to perform more than one I/O to retrieve data that could normally be obtained with a single I/O operation, with the end result being degraded performance.

> **TIP:** Click this statistic to drill down to the Chained Tables tab of the Objects Detail view.

Metrics

If the amount of chained rows in your tables exceeds 25-30%, you should take steps to eliminate the problem. Further, if the amount of chained rows accessed in your system vs. total rows accessed approaches 20-30%, you can start the process of eliminating the chained and migrated rows.

Troubleshooting

You can locate tables that contain chained rows. Once found, there are a couple of ways to reorganize tables to remove the chaining/migration problem. However, the best way to deal with chained and migrated rows is to prevent them from occurring. The table below describes two methods:

Method	Description
Use a large block size	Because chaining and migrations occur when the Oracle block is too small to hold the rows in question, make sure you are using a large enough block size for each database you create. An 8KB block size or higher is normally recommended to help stave off chained and migrated rows. If you are using Oracle9i or later, you can create special tablespaces that have larger block sizes (16-32KB) than the overall database block size and place any table that is a candidate for chained/migrated rows into them.
Use proper values of PCTFREE for tables	The necessary amount of percent free in a table helps prevent row migrations from occurring. If you have a database that houses rows with the potential to grow substantially over their initially inserted size, provide a liberal amount of PCTFREE for each of your tables.

If chaining or migrations are already on the rise in your database, you can take steps to counter the attack. Chaining is usually hard to resolve because rows are too big for the originally specified block size. A rebuilt database is normally the only cure for them. Migrations are another story. You can usually back up the rows in your migrated-row tables, re-create them with larger amounts of PCTFREE, and then bring the rows back in. The ANALYZE…LIST CHAINED ROWS command is useful for locating chained and migrated rows in a table.

Objects With Space Deficits

If an existing Oracle object needs more space (due to incoming data additions, etc.) Oracle allocates new space based on the object's NEXT extent parameter. If an object's NEXT extent allocation is too large to fit into the parent tablespace's largest contiguous block of free space, and if the underlying tablespace's datafiles do not have their AUTOEXTEND property set or if they have hit their AUTOEXTEND limit, the object cannot grow in size and Oracle issues an error. For locally-managed tablespaces with generous settings for AUTOEXTEND, this problem is nonexistent as object extent management is handled by the database, but this is not the case for dictionary-managed tablespaces.

This statistic displays a count of all objects (including tables, indexes, rollback segments, etc.) that have a defined NEXT extent allocation that is larger than their tablespace's largest contiguous block of free space.

> **TIP:** Click this statistic to drill down to the Extent Deficits tab of the Space Detail view.

Metrics

You should investigate any count larger than zero, as this indicates a potential object/space problem for the database.

Troubleshooting

There are many solutions available for objects that have problematic NEXT extent sizes:

- You can ALTER the objects to reduce their NEXT extent size to an amount that is smaller than their tablespace's largest contiguous block of free space.
- If the parent tablespace suffers from heavy bubble-style fragmentation, the tablespace can be reorganized so that all pockets of free space are reclaimed.
- The parent tablespace's datafile(s) can have their AUTOEXTEND feature enabled. A side option is to have the datafile growth AUTOEXTEND option set to UNLIMITED.
- The object can be moved to a locally-managed tablespace or the parent tablespace can be converted to locally-managed (Oracle8i or later). If locally-managed tablespaces cannot be used, you can manually impose NEXT extent limits for all objects that are well within reason. In addition, you should set the PCTINCREASE parameter for objects to zero to minimize the potential of stratospheric object growth. To better control all object size/parameter allocations, you should set growth parameters for all objects at the tablespace level and not the object level.

Objects With Extent Problems

Objects With Extent Problems helps you determine how many objects in your database are at or approaching maximum extents. For locally managed tablespaces, this is not a problem as the database handles object extent management, but for dictionary-managed tablespaces, the potential for problems exist.

Metrics

Use the Object Extents tab of the Object Detail view to investigate which objects are at or near max extents.

Troubleshooting

If you cannot use locally managed tablespaces to help stave off this problem, increase the number of max extents for the object.

Objects Pinned

The Library Cache is an area of memory in Oracle's shared pool that permits users to share and reuse code objects like SQL statements, PL/SQL procedures, functions, packages, and the like. To avoid expensive parse operations, the Library Cache was introduced to let Oracle cache the code objects and shared SQL so other user processes can share the same SQL or PL/SQL.

> **TIP:** Click this statistic to drill down to the Shared Pool tab of the Objects Detail view.

Metrics

Objects that have many loads into the shared pool but are not marked KEPT are ideal candidates for being permanently pinned in memory using Oracle's DBMS_SHARED_POOL utility. Doing so helps reduce expensive load operations.

Troubleshooting

Use the DBMS_SHARED_POOL utility to pin an Oracle code object in the shared pool. Keep in mind that pinning objects in memory with the DBMS_SHARED_POOL utility is only good for the duration of the current Oracle instance's life. Once you bring Oracle down, and then start back up, you need to re-pin the objects again. You can add this operation to your database maintenance plan, which is invoked on any database startup.

Invalid/Unusable Objects

The statistic combines the number of objects with a status of INVALID with the number of indexes that have a status of UNUSABLE. Objects like procedures, packages, functions, triggers, and views can become invalidated for a variety of reasons. The main cause is generally a dependent object that has been altered or removed from the system. However, other objects, like indexes, can become invalid also due to scenarios like SQL*Loader problems. If an object that has become invalid is still referenced (through an application or SQL query tool), a variety of problems can result. Sometimes Oracle reports a clear error stating the problem, while other times seemingly odd behavior is exhibited by the database.

> **TIP:** Click this statistic to drill down to the Invalid Objects tab of the Objects Detail view.

Metrics

There is no reason to have invalid objects in a production database. If your production databases have invalid objects that are no longer needed, promptly remove them from each system. Any needed objects that are indicating an invalid status should quickly be fixed before access problems develop.

It is very normal for development databases to have invalid objects because developers create, modify, and compile objects all the time. The only invalid object that really should not be present in either a development or production database is an invalid index.

Troubleshooting

If code objects have become invalidated, you can issue an ALTER … COMPILE command to see if they compile properly and become valid once again. If they do not, then check the USER_ERRORS view for any resulting error messages. Indexes can be validated once more by using the ALTER INDEX … REBUILD command.

Locked Objects

- Metrics

Locked Objects is a count of all objects on the system that currently have some form of lock against them.

> **TIP:** Click this statistic to drill down to the All Locks tab of the Locks view.

Metrics

None.

> **NOTE:** Drilling down into the count of locked objects displays detail on each object that is locked, along with the user process holding the lock and the type of lock held.

Rollback Summary

- Metrics
- Troubleshooting

These statistics display how many rollback segments are currently setting at their OPTIMAL size setting and how many have extended into a larger size. Dynamic rollback extension can take a toll on performance when rollback segments are consistently enlarged to accommodate heavy transaction loads.

> **TIP:** Double-click these statistics to drill down to the Rollback Activity tab of the I/O Detail view.

Metrics

Seeing rollback segments undergoing numerous extends and shrinks (as Oracle returns a segment to its OPTIMAL setting), as well as rollback segments having current or high watermark sizes greater than their OPTIMAL setting usually is a good indicator that they should be permanently enlarged.

Troubleshooting

For rollback segments, you can specify an OPTIMAL setting for each rollback segment. Setting OPTIMAL for the rollback gives the segment the opportunity to be shrunk back to the extent boundary where you would like to keep the rollback segment. This is periodically done by the database, but can also be accomplished by manually issuing the ALTER ROLLBACK…SHRINK command.

Objects Detail

The following tabbed pages are available on the Objects Detail page:

- Invalid Objects

- Buffer Pool
- Chained Tables
- High Watermarks
- Objects Accessed
- Object Extents
- Objects in Memory

Invalid Objects Tab

- Metrics

Objects like procedures, packages, functions, triggers, and views can become invalidated for a variety of reasons, with the main cause being a dependent object that has been altered or removed from the system. However, other objects, like indexes, can become invalid also due to scenarios like SQL*Loader problems. If an object that has become invalid is still referenced (through an application or SQL query tool), a variety of problems can result. Sometimes Oracle will report a clear error stating the problem, while other times seemingly quirky behavior will be exhibited by the database. In any event, as a DBA you should be on the lookout for objects in your database that have suddenly become invalid.

The Invalid Objects tab of the Objects Detail view displays invalid objects found in the database. The first grid displays a summary of the invalid/unusable objects. The second grid displays the details of the invalid/unusable objects. The table below describes the information available on the Invalid/Unusable Objects Summary grid on the Invalid Objects tab of the Objects Detail view:

Column	Description
Owner	The user account that currently owns invalid database objects.
Count	The number of invalid objects found in the user account.

The table below describes the information available on the Invalid/Unusable Objects Detail grid on the Invalid Objects tab of the Objects Detail view:

Column	Description
Owner	The user account that owns the objects.
Object Name	The name of the invalid object.
Object Type	The type of object: PROCEDURE, VIEW, etc.
Status	The status of the object, INVALID or UNUSABLE.
Created	The date and time stamp when the object was created.
Last DDL Date	The last structural modification date for the object.

TIP: To configure a grid to display row numbers, use the Options Editor.

Metrics

There is no reason to have invalid objects in a production database. If your production databases have invalid/unusable objects that are no longer needed, you should promptly remove them from each system. Any needed objects that are indicating an invalid status should quickly be fixed before access problems develop.

Correcting invalid objects like procedures and views often involves performing an ALTER…COMPILE operation. If the object status does not return to VALID, then further examination is warranted.

It is very normal for development databases to have invalid objects because developers will no doubt be creating, modifying, and compiling objects all the time. The only invalid object that really should not be present in either a development or production database is an invalid index.

Buffer Pool Tab

- Metrics

Because data objects can be referenced with different types of usage patterns, Oracle8 offers the option to intelligently place objects into one of three distinct buffer caches. The table below describes these buffer caches:

Buffer Cache	Description
KEEP	Minimizes misses in the buffer cache. Small, frequently referenced objects are candidates for the KEEP buffer pool.
RECYCLE	Lets you avoid having large numbers of infrequently accessed blocks of data crowd out objects that need to be referenced in RAM. The RECYCLE pool is good for large objects that are scanned from beginning to end without the need for keeping all their data in RAM.
DEFAULT	The traditional cache for all other data objects.

NOTE: Unless you specify the KEEP or RECYCLE buffer cache, Oracle automatically places objects into the DEFAULT buffer cache.

The first grid displays a summary of the objects in the buffer pool. The second grid displays the details of the objects in the buffer pool. The table below describes the information available on the Buffer Pool Objects Summary grid on the Buffer Pool tab of the Objects Detail view:

Column	Description
Object Type	The type of object (TABLE, INDEX, etc.)
Buffer Pool	The name of the buffer pool (KEEP, RECYCLE, DEFAULT).
Object Count	The number of objects in the indicated pool for the object type.

The table below describes the information available on the Buffer Pool Objects Detail grid on the Buffer Pool tab of the Objects Detail view:

Column	Description
Buffer Pool	The name of the buffer pool (KEEP, RECYCLE, DEFAULT).
Owner	The account that owns the named object.
Object Name	The name of the object.
Object Type	The type of object (TABLE, INDEX, etc.)

TIP: To configure a grid to display row numbers, use the Options Editor.

Metrics

When looking at overall database I/O activity, you should keep an eye out for objects that you can place into a particular buffer cache. Consider using a KEEP cache for relatively small, frequently accessed tables that require fast response times. Large tables with random I/O activity and are scanned from beginning to end are good candidates for a RECYCLE cache.

Objects can be placed into different buffer pools at object creation time (using the STORAGE...BUFFER_POOL option) or objects can be moved into a different pool at any time with the ALTER command.

If you just want to use the DEFAULT buffer pool and not enable any special caches, you can still encourage Oracle to keep certain objects in the cache as long as possible using the CACHE parameter. For example, issuing the command ALTER TABLE…CACHE specifies that the blocks retrieved for a table be placed at the most recently used end of the LRU list in the DEFAULT buffer cache when a full table scan is performed. You can also use CACHE hint in SQL statements to cache a table, but used in this form, the blocks will only be cached until the next time the database is shut down.

Chained Tables Tab

- Metrics

In normal circumstances, a row of data should fit completely inside one Oracle block. Sometimes, however, this is not the case and the table suddenly finds itself containing chained or migrated rows, which are rows that span more than one data block.

Chaining occurs when a row is initially too large to fit inside one block. Two or more blocks are used by Oracle to hold the row. Migration deals with rows that have grown so much that they can no longer be contained within their original block. When this occurs, Oracle relocates the row out of its original block into another block, but leaves a pointer behind to indicate the relocation. Both chaining and migration force Oracle to perform more than one I/O to retrieve data that could normally be obtained with a single I/O operation, resulting in degraded performance.

The Chained Tables tab of the Objects Detail view displays tables that currently have chained rows. Note that for truly accurate data to be returned, the ANALYZE command needs to be routinely inside the database. This ensures that the data dictionary is up to date with respect to object demographics.

The table below describes the information available on the Chained Tables tab of the Objects Detail view:

Column	Description
Schema	The user account that owns the chained tables.
Table Name	The table name.
Rows	The number of rows that reside inside the table.
Chained Rows	The number of chained rows that reside inside the table.
Percent Chained	The percentage of chained rows to total rows in the table.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

How bad should chaining or migration get before you take action with a table? If the amount of chained rows in your tables exceeds 25-30%, you should take steps to begin eliminating the problem. Further, if the amount of chained rows accessed in your system vs. total rows accessed approaches 20-30%, you can start the process of eliminating the chained and migrated rows.

The best way to deal with chained and migrated rows is to prevent them from occurring in the first place. The table below describes some ways to do this:

Method	Solution
Use a Large Block Size	Because chaining and migrations occur when the Oracle block is too small to hold the rows in question, make sure you are using a large enough block size for each database you create. An 8 KB or greater block size is normally recommended to help stave off chained and migrated rows.

Method	Solution
Use Proper Values of PCTFREE for Tables	The necessary amount of percent free in a table is what helps prevent row migrations from occurring. If you have a database that houses rows with the potential to grow substantially over their initially inserted size, you can provide a liberal amount of PCTFREE for each of your tables.

If chaining or migrations are already on the rise in your database, you can take steps to counter the attack. Chaining is usually hard to resolve because rows are too big for the originally specified block size. A rebuilt database is normally the only cure for them. Migrations are another story. You can usually back up the rows in your migrated-row tables, re-create them with larger amounts of PCTFREE, and then bring the rows back in.

High Watermarks Tab

- Metrics

Tables that are the victim of much insert-delete activity can become performance problems due to their block high watermarks. A table's high watermark equals the last block in the table that was used to hold data. The problem is that the high watermark is not reset by DELETE activity, so it is possible for a table to have absolutely no data in it, but contain a high watermark that is many blocks high. When such a table is scanned, Oracle reads up to the high watermark even if no rows exist in the table at all. This can make for some unnecessarily large scan times.

The table below describes the information available on the High Watermarks tab of the Objects Detail view:

Column	Description
Owner	The user account that owns the tables.
Table Name	The name of the table.
Size (bytes)	The physical size of the table, in bytes.
Rows	The number of rows occupying the table.
Blocks	The number of blocks the table consumes.
Empty Blocks	The number of blocks assigned to the table that contain no data.
High Watermark	The last block in the table that was used to hold data.

NOTE: For truly accurate data to be returned, the ANALYZE command needs to be routinely run on the viewed tables. This ensures that the data dictionary is up to date with respect to object demographics.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Red flags should go up to you if you observe a table that contains no rows (or few rows) but has a high watermark that is nonzero for no rows, and much above zero for a few. Again, to effectively detect a table's high watermark, you must make timely use of the ANALYZE command to update the data dictionary. Doing so ensures a fresh supply of statistics available to analyze used and free data blocks in a table.

Objects Accessed Tab

- Metrics

There are two things to consider regarding the performance of objects and Oracle memory use. First, because data can be accessed many times faster in RAM than from disk, you should keep often-referenced data from your objects in memory as much as possible. It is also wise to keep object definitions in memory so that references to them are quick.

Second, to avoid expensive parsing operations and continuous loads into the Oracle shared pool, you should do everything possible to encourage code reuse and ensure that parsed code is ready and available in the library cache.

The Objects Accessed tab of the Objects Detail view displays what objects are currently being accessed in memory and users who are using them. The table below describes the information available on the Objects Accessed tab of the Objects Detail view:

Column	Description
Owner	The user account that owns the object.
Object name	The name of the object in memory.
Type	The type of object (TABLE, CURSOR, VIEW, etc.)
Accessing User	The user who is currently using the object.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Once you have an idea of what objects your users are most often after, you can do a couple of things to help their access process along.

With respect to data objects, you can use the Oracle8 concept of the KEEP buffer cache to force Oracle to hold often referenced data. The KEEP buffer cache is ideal for holding small lookup tables and the like. If you have an earlier version of Oracle or don't want to divide your current buffer cache, you can alternatively use the CACHE table attribute to encourage Oracle to keep data blocks of cached tables at the most recently used end of the LRU buffer cache chain.

Objects that have many loads but are not marked KEPT in the shared pool are ideal candidates for being permanently pinned in memory using the DBMS_SHARED_POOL utility. Doing so helps reduce expensive load operations. Keep in mind that pinning objects in memory with the DBMS_SHARED_POOL utility is only good for the duration of the current Oracle instance's life. Once Oracle is brought down and started back up, re-pin the objects.

Object Extents Tab

- Metrics

Object fragmentation results when objects consume multiple extents of space in a tablespace rather than a single block of space. Although performance problems with respect to object fragmentation are not as severe as they were in older versions of Oracle, response-time penalties can still be chalked up to this situation. When multiple extents exist for an object, the amount of time it takes Oracle to scan it can be longer than if the object was made up of only one extent. This typically holds true when extents are scattered on different parts of the physical server disk. In addition, a performance hit is taken each time an object must extend into another extent of space.

The Object Extents tab of the Objects Detail view displays objects whose extent count has exceeded a user-suggested numerical limit. If objects are found, Embarcadero Performance Center displays them on the this tab. The table below describes the information available on the Object Extents tab of the Objects Detail view:

Column	Description
Owner	The user account that owns the object.
Object Name	The name of the object.
Object Type	The type of object (TABLE, INDEX, etc.)
Extents	The number of extents for the object.
Limit	The MAXEXTENTS limit imposed on the object by the DBA.
Pct/Limit	How close the object is to reaching its extent limit, in percentage.

NOTE: For more information see, Object Extents Tab.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

When is object fragmentation a problem for an object? Opinions on this vary widely. As a general rule of thumb, if an object is exhibiting response time degradations, and no other factor can be attributed to the slowdown, examine the object to see how badly fragmented it is. If extent numbers exceed 20, you should consider reorganizing the object back into a single extent of space. This rule of thumb does not apply to objects assigned to locally managed tablespaces as extent fragmentation is expected and not a performance issue.

The best preventive for this problem is specifying the correct allocation of space for the object in the first place, but what can you do if you have objects with high numbers of extents? For tables and indexes you have three options:

1. Use Oracle's export/import utility to export, drop, and import the fragmented objects back into the database with the export parameter COMPRESS=Y. This brings the objects back into the database with one large extent. Make sure that large enough chunks of available free space exist to accept the object back in, or you may experience space allocation errors.
2. With Oracle8, you can use the ALTER TABLE…MOVE command to reorganize a table back into one extent in a very safe and efficient manner.
3. Use ALTER INDEX…REBUILD to reorganize indexes that have moved into multiple extents.

For rollback segments, you can specify an OPTIMAL setting for each rollback segment. Setting OPTIMAL for the rollback gives the segment the opportunity to be shrunk back to the extent boundary where you would like to keep the rollback segment. This is periodically done by the database, but can also be accomplished by manually issuing the ALTER ROLLBACK…SHRINK command.

Objects in Memory Tab

- Metrics

The Objects in Memory tab of the Objects Detail view shows objects currently in the library cache. This view lets you see exactly which objects are in the shared pool as well as their resource usage and activity levels.

The table below describes the information available on the Objects in Memory tab of the Objects Detail view:

Column	Description
Owner	The user account that owns the object.
Object Name	The name of the object.
Type	The type of object: INDEX, TABLE, CLUSTER, VIEW, SET, SYNONYM, SEQUENCE, PROCEDURE, FUNCTION, PACKAGE, PACKAGE BODY, TRIGGER, CLASS, OBJECT, USER, or DBLINK.
Sharable Memory	The amount of memory consumed by the object in the shared pool.
Loads	The number of times the object has been loaded into the cache. This count increases when an object has been invalidated.
Executions	The number of times that the object has been executed by a session thread.
Locks	The number of users actively locking the object.
Pins	The number of users actively pinning the object.
Kept	Whether or not the object has been pinned in memory with the DBMS_SHARED_POOL package.

NOTE: This information is available on both the Objects in Memory tab of the Memory Detail view and the Objects in Memory tab of the Objects Detail view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If you see objects with large numbers of loads, this could signal that the objects are being frequently by user sessions and can age out of the cache prematurely. To fix this you could increase the size of the shared pool and/or pin the objects with the DBMS_SHARED_POOL package so they do not age out of the library cache.

OS Page Statistics

In many scenarios, an optimally tuned database may not perform well because there are constraints imposed by the system where the database is running. These constraints may include processes competing with the database sever for resources (CPU, I/O, or Memory), a slow CPU, insufficient or slow I/O devices, and insufficient memory. The OS Statistics page of Performance Center lets you examine operating system metrics for the following platforms:

- AIX
- HP-UX

NOTE: To view processor info and swap disk info on an HP-UX box, you need to login as ROOT in the OS login.

- Linux
- Solaris
- Unix
- Windows XP, 2000, and NT

CAUTION: The OS statistics are not available from the Web Client.

- Average Disk Queue
- Network Output Queue (Windows)
- Network Queue (Solaris)
- Page Faults/Sec
- Processor Queue
- Processor Speed
- Processor
- Available Paged Memory (Windows)
- Available Physical Memory
- Available Swap Memory (AIX, HP-UX, Linux, Solaris, Unix)
- Total Paged Memory (Windows)
- Total Physical Memory
- Total Swap Memory (AIX, HP-UX, Linux, Solaris, Unix)
- Free Disk Space
- Total Disk Space
- Used Disk Space
- Number of Logins
- Number of Processes
- Number of Processors
- Top CPU Process
- Top I/O Process
- Top Memory Process

Processor Time

The Processor Time statistic indicates the percentage of time the processor is working. This counter is a primary indicator of processor activity.

Metrics

If your computer seems to be running sluggishly, this statistic could be displaying a high percentage.

Troubleshooting

Upgrade to a processor with a larger L2 cache, a faster processor, or install an additional processor.

Processor Speed

The Processor Speed statistic displays the speed of the active processor in MHz. The speed is approximate.

Processor

The Processor Statistic displays the type of processor currently in use, for example, GenuineIntel.

Disk Time

The Disk Time statistic is the percentage of elapsed time that the selected disk drive/device was busy servicing read or write requests.

Metrics

You should avoid consistently seeing values for this statistic greater then 90%.

Troubleshooting

Add more disk drives and partition the files among all of the drives.

Load Average

The Load Average statistic represents the system load averages over the last 1, 5, and 15 minutes.

Metrics

High load averages usually mean that the system is being used heavily and the response time is correspondingly slow.

Paged Memory Used

The Paged Memory Used statistic is the ratio of Commit Memory Bytes to the Commit Limit. Committed memory is where memory space has been reserved in the paging file if it needs to be written to disk. The commit limit is determined by the size of the paging file. As the paging file increases, so does the commit limit.

NOTE: This statistic is available for the Windows platform.

Metrics

This value displays the current percentage value only and not an average. If the percentage of paged memory used is above 90%, you may be running out of memory.

Troubleshooting

Increase the size of page file.

Number of Processors

This statistic displays the number of processors currently in use.

Swap Memory Used

The Swap Memory Used statistic is the percentage of swap space currently in use.

Metrics

If the percentage of swap memory used is above 90%, you may be running out of memory.

Troubleshooting

Increase the size of your swap files.

Average Disk Queue

The Average Disk Queue statistic is the average number of both read and write requests that were queued for the selected disk during the sample interval.

Metrics

This metric is useful in identifying I/O related bottlenecks. If the disk queue lengths for certain disks are consistently much higher than others, you may need to redistribute the load among available disks. If the disk queues lengths for all disks are consistently large, and you see a high amount of I/O activity, your disks may be inefficient.

Troubleshooting

Some things you can do if you have problems with this statistic include:

- Redistribute the data on the disk with the large average disk queue to other disks.
- Upgrade to faster disk(s).

Page Faults/Sec

The Page Faults/Sec statistic is the overall rate faulted pages are handled by the processor. It is measured in numbers of pages faulted per second. A page fault occurs when a process requires code or data that is not in its working set. This counter includes both hard faults and soft faults.

Metrics

This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

If the number of page faults remains consistently high, you can check with your Windows System Administrator for further investigation. Often, large numbers of page faults are not a problem so long as they are soft faults. However, hard faults, that require disk access, can cause delays.

Processor Queue

The Processor Queue Length statistic is the number of threads in the processor queue.

Metrics

Unlike the disk counters, this counter shows ready threads only, not threads that are running. There is a single queue for processor time even on computers with multiple processors. Therefore, if a computer has multiple processors, you need to divide this value by the number of processors servicing the workload. A sustained processor queue of less than 10 threads per processor is normally acceptable, dependent of the workload.

Troubleshooting

A sustained high value in the Processor Queue could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Network Output Queue/Network Queue

The Network Output Queue Length statistic is the number of threads in the processor queue.

NOTE: The name of this statistic depends on the platform of the operating system.

Metrics

Unlike the disk counters, this counter shows ready threads only, not threads that are running. There is a single queue for processor time even on computers with multiple processors. Therefore, if a computer has multiple processors, you need to divide this value by the number of processors servicing the workload. A sustained processor queue of less than 10 threads per processor is normally acceptable, dependent of the workload.

Troubleshooting

A sustained high value in the Processor Queue Length could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Available Physical Memory

The Available Physical Memory statistic represents the amount of RAM available to all processes.

Metrics

This counter displays the last observed value only and not an average. Use this value with the Total physical memory and paging metrics (Memory details page). If the available physical memory is very small compared to this value, and the paging activity is high, your system may be running low on memory.

Troubleshooting

Some things you can do if you have problems with this statistic include:

- Check the running processes to see if there are any memory leaks.
- Stop any services that are not required.
- Install additional RAM.

Available Paged Memory

The Available Paged Memory statistic shows the amount of virtual memory available for the processes.

NOTE: This statistic is available for the Windows platform.

Metrics

If the available virtual memory is less than 10% of the total virtual memory, your system may run out of memory.

Troubleshooting

Increase the size of page file.

Available Swap Memory

The Available Swap Memory statistic represents the amount of virtual memory available for the processes.

Metrics

If the available Available Swap Memory is less than 10% of the total Swap Memory, your system may run out of memory.

Troubleshooting

Increase the size of swap files.

Total Physical Memory

The Total Physical Memory statistic shows the amount of physical memory installed on your computer.

Metrics

This is an informational metric and displays the total amount installed on the machine. Use this value with the available physical memory and paging metrics (Memory details page). If the available physical memory is very small compared to this value, and the paging activity is high, your system may be running low on memory.

Total Paged Memory/Total Swap Memory

The Total Paged Memory statistic shows the maximum amount of virtual memory available to all processes.

NOTE: The name of this statistic depends on the platform of the operating system.

Metrics

It is recommended that this value is between 1.5 to 3 times the amount of RAM on the system.

Used Disk Space

The Used Disk Space statistic shows the amount of allocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations:

- If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
- You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Total Disk Space

Total Disk Space displays the total allocated and unallocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations, here are two:

1. If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
2. You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Free Disk Space

The Free Disk Space statistic shows the unallocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations:

- If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
- You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Top Memory Process

Top Memory Process shows the current process that is consuming the most amount of memory. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Metrics

If you are running out of memory on the system, this is a quick way to identify the top memory user. If the displayed process is using a significant portion of the total memory, it could be causing the memory issues.

Processes Overview

The Processes Overview of the OS Summary includes the following sections:

- Top CPU Process
- Top I/O Process
- Top Memory Process

Top CPU Process

Top CPU Process shows the current process that is consuming the most amount of CPU. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Metrics

If the amount of CPU time used by this process is close to 100% and the CPU usage is very high, this process may be the bottleneck on the server.

Troubleshooting

Investigate the process further to see if it is in an inconsistent state. Also, look at minimum requirements for CPU speed for the process. You may need to upgrade your CPU.

Top I/O Process

The Top I/O Process statistic shows the current process that is consuming the most amount of CPU. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Top Memory Process

The top I/O process identifies the Oracle process that currently is using the highest percentage of memory in the database.

Metrics

To obtain more details on the top memory process, locate the SID in the Top Sessions grid and drill down to obtain more granular information.

Number of Logins

This statistic displays the total number of logins on the server.

Number of Processes

This statistic displays the total number of processes on the server.

CPU Tab

The CPU tab of the OS Detail includes the following sections:

- Context Switches/Sec
- CPU Utilization
- Interrupts/Sec
- Processor Queue Length

CPU Utilization

The CPU Utilization section includes the following information:

- % Privileged Time
- % User Time

% Privileged Time

The % Privileged Time statistic is the percentage of elapsed time that the process threads spent executing code in privileged mode.

NOTE: For Windows systems, when a Windows system service is called, the service will often run in privileged mode to gain access to system-private data. Such data is protected from access by threads executing in user mode. Calls to the system can be explicit or implicit, such as page faults or interrupts. These kernel commands, are considered privileged to keep the low-level commands executing and prevent a system freeze. Unlike some early operating systems, Windows uses process boundaries for subsystem protection in addition to the traditional protection of user and privileged modes. Some work done by Windows on behalf of the application might appear in other subsystem processes in addition to the privileged time in the process.

Metrics

The ideal range should be 0-40% (less than 40% indicates excessive system activity).

Troubleshooting

If your CPU consistently runs at less than 40% you may need to upgrade your system to include a faster processor(s).

% User Time

The % User Time statistic is the percentage of elapsed time the processor spends in the user mode. User mode is a restricted processing mode designed for applications, environment subsystems, and integral subsystems. The alternative, privileged mode, is designed for operating system components and allows direct access to hardware and all memory. The operating system switches application threads to privileged mode to access operating system services. This counter displays the average busy time as a percentage of the sample time.

Metrics

If the Privileged Time is high in conjunction with Physical Disk Reads, consider upgrading the disk I/O subsystem.

Interrupts/Sec

The Interrupts/Sec statistic is the average rate, in incidents per second, at which the processor received and serviced hardware interrupts. It does not include deferred procedure calls (DPCs), which are counted separately. This value is an indirect indicator of the activity of devices that generate interrupts, such as the system clock, the mouse, disk drivers, data communication lines, network interface cards, and other peripheral devices. These devices normally interrupt the processor when they have completed a task or require attention. Normal thread execution is suspended. The system clock typically interrupts the processor every ten milliseconds, creating a background of interrupt activity. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Metrics

The ideal rage should be 0-5000. A number greater then 5000 indicates possible excessive hardware interrupts; justification is dependent on device activity.

Context Switches/Sec

The Context Switches/Sec section shows the combined rate at which all processors on the computer are switched from one thread to another. Context switches occur when a running thread voluntarily relinquishes the processor, is preempted by a higher priority ready thread, or switches between user-mode and privileged (kernel) mode to use an Executive or subsystem service.

Metrics

The ideal range should be between 0-10,000. GA number greater then 10,000 may indicate too many threads contending for resources.

Processor Queue Length

The Processor Queue Length statistic is the number of threads in the processor queue. There is a single queue for processor time even on computers with multiple processors.

> **NOTE:** For Windows systems, unlike the disk counters, this counter shows ready threads only, not threads that are running.

Metrics

A sustained high value in the Processor Queue Length could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Processes Tab

The Processes tab of the OS Detail page succinctly communicates the general overall performance levels of processes. The columns available in this table depend on the platform of operating system. The table below describes the information available in the table on this tab:

Column	Description
Process	The name of the process.
User	The user of the process.
ID	The ID Process is the unique identifier of this process. ID Process numbers are reused, so they only identify a process for the lifetime of that process.
CPU	The CPU is the percentage of elapsed time that all of process threads used the processor to execution instructions.
User Mode	The User Mode is the percentage of elapsed time that the process threads spent executing code in user mode.
Memory **WINDOWS ONLY**	Memory is the current size, in bytes, of the virtual address space the process is using. Use of virtual address space does not necessarily imply corresponding use of either disk or main memory pages. Virtual space is finite, and the process can limit its ability to load libraries.
Memory (MB)	Memory is the current size, in bytes, of the virtual address space the process is using. Use of virtual address space does not necessarily imply corresponding use of either disk or main memory pages. Virtual space is finite, and the process can limit its ability to load libraries.
Memory	Memory is the percentage of the memory used of the total memory.
Active Memory	Active Memory is the amount of committed virtual memory, in bytes for this process. Active memory is the physical memory which has space reserved on the disk paging file(s). There can be one or more paging files on each physical drive. This counter displays the last observed value only; it is not an average.

Column	Description
I/O Data	The rate at which the process is reading and writing bytes in I/O operations. This counter counts all I/O activity generated by the process to include file, network and device I/Os.
Elapsed Time	The total elapsed time, in seconds, that this process has been running.
Thread Count	The number of threads currently active in this process. An instruction is the basic unit of execution in a processor, and a thread is the object that executes instructions. Every running process has at least one thread.
Handle Count	The total number of handles currently open by this process. This number is equal to the sum of the handles currently open by each thread in this process.
Priority	The current base priority of this process. Threads within a process can raise and lower their own base priority relative to the process' base priority.
Creating Proc ID	The Creating Process ID value is the Process ID of the process that created the process. The creating process may have terminated, so this value may no longer identify a running process.
Page Faults/Sec	Page Faults/Sec is the rate at which page faults by the threads executing in this process are occurring. A page fault occurs when a thread refers to a virtual memory page that is not in its working set in main memory. This may not cause the page to be fetched from disk if it is on the standby list and hence already in main memory, or if it is in use by another process with whom the page is shared.
Page File	Page File is the current number of kilobytes that this process has used in the paging file(s). Paging files are used to store pages of memory used by the process that are not contained in other files. Paging files are shared by all processes, and the lack of space in paging files can prevent other processes from allocating memory.
Private	Private is the current size, in kilobytes, of memory that this process has allocated that cannot be shared with other processes.

I/O Tab

The table below describes the information available in this section:

Column	Description
Disk	The disk number assignment.
Reading (KB/s)	The amount of bytes read from the device.
Writing (KB/s)	The amount of bytes written to the device.
Disk Read Time	Disk Read Time is the percentage of elapsed time that the selected disk drive was busy servicing read requests.
Disk Write Time	Disk Write Time is the percentage of elapsed time that the selected disk drive was busy servicing write requests.
Disk Time	Disk Time is the percentage of elapsed time that the selected disk was busy servicing requests.
Avg. Read Queue	Avg. Disk Read Queue Length is the average number of read requests that were queued for the selected disk during the sample interval.
Avg. Write Queue	Avg. Disk Write Queue Length is the average number of write requests that were queued for the selected disk during the sample interval.
Disk Reads/Sec	Disk Reads/Sec is the rate of read operations on the disk.
Disk Writes/Sec	Disk Writes/Sec is the rate of write operations on the disk.

Memory Tab

The Memory tab of the OS Detail page includes the following sections:

- Cache Efficiency
- Free Physical
- Free Paged
- Paging Activity
- Page Faults
- Total Physical
- Total Paged

Paging Activity

The Paging Activity section includes the following statistics:

- Pages Input/Sec
- Pages Output/Sec

Pages Input/Sec

The Pages Input/Sec statistic is the number of pages read from disk to resolve hard page faults. Hard page faults occur when a process requires code or data that is not in its working set or elsewhere in physical memory, and must be retrieved from disk.

Metrics

This value was designed as a primary indicator of the kinds of faults that cause system-wide delays. It includes pages retrieved to satisfy faults in the file system cache (usually requested by applications) and in non-cached mapped memory files. This counter counts numbers of pages, and can be compared to other counts of pages, such as Memory: Page Faults/sec, without conversion. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

Although it never hurts to have as much physical memory as your system can handle, there are some things you can check within your system to alleviate the memory bottleneck.

- Check to see if you have any drivers or protocols that are running but not being used. They use space in all memory pools even if they are idle.
- Check to see if you have additional space on your disk drive that you could use to expand the size of your page file. Normally, the bigger the initial size of your page file, the better, in performance terms.

Pages Output/Sec

The Pages Output/Sec statistic is the number of pages written to disk to free up space in physical memory. Pages are written back to disk only if they are changed in physical memory. A high rate of pages output might indicate a memory shortage.

Metrics

Windows NT writes more pages back to disk to free up space when low in physical memory. This counter counts numbers of pages, and can be compared to other counts of pages, without conversion. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

Although it never hurts to have as much physical memory as your system can handle, there are some things you can check within your system to alleviate the memory bottleneck.

- Check to see if you have any drivers or protocols that are running but not being used. They use space in all memory pools even if they are idle.
- Check to see if you have additional space on your disk drive that you could use to expand the size of your page file. Normally, the bigger the initial size of your page file, the better, in performance terms.

Free Physical

The Free Physical statistic is the amount of physical memory that is uncommitted.

Metrics

None.

Free Paged

The Free Paged statistic is the amount of uncommitted virtual memory.

Metrics

None.

Total Physical

The Total Physical statistic is the total physical memory available.

Metrics

None.

Total Paged

The Total Paged statistic is the amount of total virtual memory, in bytes. Used Memory is the physical memory that has space reserved on the disk paging file(s). There can be one or more paging files on each physical drive.

Metrics

None.

Page Faults/Sec

The Page Faults/Sec statistic is the overall rate faulted pages are handled by the processor. It is measured in numbers of pages faulted per second. A page fault occurs when a process requires code or data that is not in its working set. This counter includes both hard faults and soft faults.

Metrics

This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

If the number of page faults remains consistently high, you can check with your Windows System Administrator for further investigation. Often, large numbers of page faults are not a problem so long as they are soft faults. However, hard faults, that require disk access, can cause delays.

Cache Efficiency

The Cache Efficiency section of the Memory tab succinctly communicates the general overall performance levels of the server's memory. The following statistics are available in this section:

- Copy Read Hits%
- Data Map Hits%
- MDL Read Hits%
- Pin Read Hits%

Copy Read Hits %

The Copy Read Hits % statistic is the percentage of cache copy read requests that hit the cache and does not require a disk read to provide access to the page in the cache.

Metrics

When the page is pinned in the memory, the page's physical address in the file system cache will not be altered. A copy read is a file read operation where a page in the cache is copied to the application's buffer. Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate

Data Map Hits %

The Data Map Hits % statistic is the percentage of data maps in the file system cache that could be resolved without having to retrieve a page from the disk.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

MDL Read Hits %

The MDL Read Hits % statistic is the percentage of Memory Descriptor List Read requests to the file system cache that hit the cache and does not require disk access to provide memory access to the pages in the cache.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

Pin Read Hits %

The Pin Read Hits % statistic is the percentage of pin read requests that hit the file system cache and does not require a disk read in order to provide access to the page in the file system cache.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

Space Tab

The Space tab of the OS Detail page includes the following sections:

- Disk Space Free
- Disk Space Detail

Disk Space Free

The Disk Space Free metric displays the amount of free disk space in megabytes.

Metric

None.

Disk Space Detail

The Disk Space Detail section of the Space tab succinctly communicates the general overall performance levels of the server’s disks and space allotment. The table below describes the statistics in this section:

Statistic	Description
Partition	The drive letter of the disk.
Local Filesystem	The name of the file system.
Type	The type of file system.
Total Space	Total size of the disk/device's capacity expressed in MBs.
Used Space	Amount of MBs currently allocated on the particular disk/device.
Free Space	Amount of MBs currently unallocated and free on the particular disk/device.

Statistic	Description
Capacity	The percentage of space used on the device.
Mounted On	The mount point of the device.

Network Tab

The Network tab of the OS Detail page succinctly communicates the general overall performance levels of the server's networking. The statistics available in this section depend on the platform of operating system. The table below describes the information available in this section:

Column	Description
Network Interface	The name of network interface.
INET Address/Address	The IP address assigned to the network interface.
Pkts Sent/Sec	The number of packets sent per second.
Pkts Received/Sec	The number of packets received per second.
Sent (KB/Sec)	The number of bytes sent per second.
Received (KB/Sec)	The number of bytes received per second.
Out Pkts Discarded	The number of outbound packets discarded.
In Pkts Discarded	The number of inbound packets discarded.
Out Pkt Errors	The number of outbound packet errors.
In Pkt Errors	The number of inbound packet errors.
Queue Length	The queue length.
Collisions	The number of collisions.
Packets Discarded	The number of packets discarded.

Contention Statistics - Oracle

The Contention performance category view displays the following vital Oracle contention statistics:

- Buffer Busy Wait Ratio
- Enqueue Waits
- Free List Waits
- Latch Immediate Miss Ratio
- Latch Miss Ratio
- Latch Sleep Ratio
- Max Query Slaves
- Parallel Query Busy Ratio

- Prolonged Lock Waits
- Redo Log Space Requests
- Redo Log Wait Time
- Rollback Contention Ratio
- Sessions in Buffer Busy Waits
- Sessions Waiting for Latches
- Used Query Slave
- Users Blocked
- Users Waiting (General)

Latch Miss Ratio

- Metrics
- Troubleshooting

The latch miss ratio defines the number of times a process obtained a willing-to-wait latch vs. missing the attempt. Latches protect the many memory structures in Oracle's SGA. They ensure that one and only one process at a time runs or modifies any memory structure at the same instant. Latches are much more restrictive than locks, which at least allow for some collective user interaction. They have no queuing mechanism, and therefore, you either get the latch or you are forced to continually retry.

> **TIP:** Click this statistic to drill down to the System Waits tab of the Contention Detail view.

Metrics

If the latch miss ratio exceeds 1%, you should take action to resolve the amount of latch contention occurring.

Troubleshooting

You should examine the details regarding the latch contention. Increasing the shared_pool_size can also assist in resolving latch problems. Here are a few specialized cases of latch contention that you can easily rectify:

Latch Contention Case	Description	Resolution
Cache buffer chain latch	This latch is responsible for protecting paths to database block buffers in the buffer cache. Very high I/O loads tend to cause contention for this latch.	You can alleviate contention somewhat by adding more buffers to the cache (through the db_block_buffers parameter) or by adding more LRU latch chain latches with the db_block_lru_latches parameter.
Library cache latches	Protects cached SQL statements in the library cache area of the Oracle shared pool. Contention for this latch is the usual result of literals being used for SQL statements instead of bind variables.	If possible, make SQL more reusable by using bind variables instead of literals. Doing so should alleviate contention for the library cache latch."

Other routine latch contention problems used to include the redo allocation and redo copy latches, however in Oracle 8.1.5 or later these are generally obsolete.

Latch Immediate Miss Ratio

The latch immediate miss ratio defines the number of times a process obtained a not-willing-to-wait latch vs. missing the attempt. Latches protect the many memory structures in Oracle's SGA. They ensure that one and only one process at a time runs or modifies any memory structure at the same instant. Latches are much more restrictive than locks, which at least allow some collective user interaction. They have no queuing mechanism, and therefore, you either get the latch or you are forced to continually retry.

> **TIP:** Click this statistic to drill down to the Latch Detail tab of the Contention Detail view.

Metrics

If the latch immediate miss ratio exceeds 1%, you should take action to resolve the amount of latch contention occurring.

Troubleshooting

You should examine the details regarding the latch contention. Increasing the shared_pool_size can also assist in resolving latch problems. Here are a few specialized cases of latch contention that you can easily rectify:

Latch Contention Case	Description	Resolution
Cache buffer chain latch	This latch is responsible for protecting paths to database block buffers in the buffer cache. Very high I/O loads tend to cause contention for this latch.	You can alleviate contention somewhat by adding more buffers to the cache (through the db_block_buffers parameter) or by adding more LRU latch chain latches with the db_block_lru_latches parameter.
Library cache latches	Protects cached SQL statements in the library cache area of the Oracle shared pool. Contention for this latch is the usual result of literals being used for SQL statements instead of bind variables.	If possible, make SQL more reusable by using bind variables instead of literals. Doing so should alleviate contention for the library cache latch."

Other routine latch contention problems previously included the redo allocation and redo copy latches. In Oracle 8.1.5 or later these are generally obsolete.

Latch Sleep Ratio

Latches protect the many memory structures in Oracle's SGA by ensuring that one and only one process at a time runs or modifies any memory structure at the same instant. Latches are much more restrictive than locks, which at least allow for some collective user interaction. They have no queuing mechanism, and therefore, you either get the latch or you are forced to continually retry.

A sleep indicates that a latch could not be obtained for a process, and that Oracle continues to retry. A high ratio indicates that many processes had to sleep multiple times before obtaining a requested latch.

> **TIP:** Click this statistic to drill down to the Latch Detail tab of the Contention Detail view.

Metrics

You should keep sleeps as low as possible. If the overall sleep ratio exceeds 1%, you should take action to resolve the amount of latch contention that is occurring.

Troubleshooting

You should examine the details regarding the latch contention. Increasing the shared_pool_size can also assist in resolving latch problems. Here are a few specialized cases of latch contention that you can easily rectify:

Latch Contention Case	Description	Resolution
Cache buffer chain latch	This latch is responsible for protecting paths to database block buffers in the buffer cache. Very high I/O loads tend to cause contention for this latch.	You can alleviate contention somewhat by adding more buffers to the cache (through the db_block_buffers parameter) or by adding more LRU latch chain latches with the db_block_lru_latches parameter.
Library cache latches	Protects cached SQL statements in the library cache area of the Oracle shared pool. Contention for this latch is the usual result of literals being used for SQL statements instead of bind variables.	If possible, make SQL more reusable by using bind variables instead of literals. Doing so should alleviate contention for the library cache latch."

Other routine latch contention problems previously included the redo allocation and redo copy latches. In Oracle 8.1.5 or later these are generally obsolete.

Buffer Busy Wait Ratio

- Metrics
- Troubleshooting

TIP: Click this statistic to drill down to the Buffer Busy Waits tab of the Contention Detail view.

Metrics

Buffer busy waits normally center around contention for rollback segments, too small an INITRANS setting for tables, or insufficient free lists for tables.

Troubleshooting

On Oracle8i or earlier, the remedy for each situation would be increasing the number of rollback segments, or altering tables to have larger settings for INITRANS to allow for more transactions per data block, and more free lists.

For Oracle9i or later, you can use the automatic segment management feature in Oracle9i locally-managed tablespaces to help make free list problems a thing of the past. Using an UNDO tablespace in 9i or later can help remedy any rollback contention problem.

You can also obtain which objects have actually experienced buffer busy waits in Oracle9i or later by querying the sys.v_$segment_statistics. This view is not populated unless the configuration parameter statistics_level is set to TYPICAL or ALL.

Rollback Contention Ratio

- Metrics

- Troubleshooting

Rollback segments are used by the Oracle RDBMS to hold data needed to rollback (or undo) any changes made through inserts, updates, or deletes to various Oracle objects. They also allow Oracle to have read consistency for long running queries. Rollback segments are used for recovery purposes and they play a role during exports of database information.

In a heavy transaction processing environment, rollback segments are accessed continuously and therefore are subject to contention problems. The rollback contention ratio helps identify contention occurring on the system relating to rollbacks.

> **TIP:** Click this statistic to drill down to the Rollback Waits tab of the Contention Detail view.

Metrics

Overall, if the rollback segment wait ratio approaches 1% or more, you should consider creating more rollback segments. You should also think about creating a specialized, larger, rollback segment to be used by long running transactions. Doing so alleviates dynamic rollback extensions and cuts down heavily on ORA-01555 snapshot too old errors.

Troubleshooting

Begin by creating new rollback segments and altering them to be online for use. Then monitor the overall contention ratio to see if it begins to drop.

Users Blocked

- Metrics
- Troubleshooting

On a small system, a single blocking user has the potential to stop work for nearly all other processes. On larger systems, this can cause major headaches. Although Oracle supports unlimited row-level locking, blocking lock situations do occur. User processes holding exclusive locks and not releasing them via a proper COMMIT frequency, generally cause most blocks.

> **TIP:** Click this statistic to drill down to the Blocking Locks tab of the Locks view.

Metrics

You should investigate any blocking lock statistic indicator above zero to prevent a mushrooming situation.

Troubleshooting

You can quickly remedy a blocking lock situation by issuing a KILL against the offending process. This eliminates the user's stranglehold on the accessed objects and lets other user processes complete. Tools like Embarcadero Performance Center make it easier to discover the blocked lock situation, the tricky part is preventing the blocking lock situation in the first place.

The culprit of blocking lock scenarios is usually the application design, or the SQL used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. Most DBAs who have had to face Oracle Forms applications have suffered through the dreaded SELECT…FOR UPDATE statements that place unnecessary restrictive locks on nearly every read operation, know all too well that good coding practice is important. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible, something not always easy to do.

Users Waiting (General)

- Metrics
- Troubleshooting

User connections that are waiting on a system generally occur for two reasons:

- A process waits because a requested resource is not available.
- A process waits for Oracle to perform a prerequisite task for its given operation.

Idle waits (processes waiting because they have no work) should not worry you. The two wait causes mentioned above are worth your time and investigation.

NOTE: This statistic displays on both the Contention performance category view and the Users performance category view.

TIP: Click this statistic to drill down to the Session Waits tab of the Contention Detail view.

Metrics

To determine the actual wait causes user connections are experiencing, drill down from the global count of users waiting into the actual system and user wait details of your target database. This lets you locate the exact causes of currently experienced waits.

Troubleshooting

If you find a problem, drill down into wait details to determine whether the waits are resource related.

Sessions Waiting for Latches

- Metrics

User connections that are waiting on a system generally occur for two reasons:

1. A process waits because a requested resource is not available.
2. A process waits for Oracle to perform a prerequisite task for its given operation.

Sessions Waiting for Latches displays the number of user processes specifically waiting for latches.

TIP: Click this statistic to drill down to the Latch Waits tab of the Contention Detail view.

Metrics

If you find nonzero values for this statistic, you should access the drill-down views to discover the exact latches that are making users wait.

Sessions in Buffer Busy Waits

- Metrics

User connections that are waiting on a system generally occur for two reasons:

- A process waits because a requested resource is not available.
- A process waits for Oracle to perform a prerequisite task for its given operation.

Sessions involved in buffer busy waits indicates contention for data blocks in the buffer cache.

> **TIP:** Click this statistic to drill down to the Buffer Busy Waits tab of the Contention Detail view.

Metrics

If you find nonzero values for this statistic, you should access the drill-down views to discover the exact latches that are making users wait.

Prolonged Lock Waits

- Metrics
- Troubleshooting

This statistic shows the number of sessions that have been blocked for over 300 seconds.

On a small system, a single blocking user has the potential to stop work for nearly all other processes. On larger systems, this can cause major headaches. Although Oracle supports unlimited row-level locking, blocking lock situations do occur. User processes holding exclusive locks and not releasing them via a proper COMMIT frequency generally cause most blocks. Click this statistic to drill down to the Blocking Locks tab of the Locks view.

Metrics

You should investigate any indicator above zero to prevent a mushrooming situation.

Troubleshooting

You can quickly remedy a blocking lock situation by issuing a KILL against the offending process. This eliminates the user's stranglehold on the accessed objects and lets other user processes complete. Embarcadero Performance Center make it easier to discover the blocked lock situation, but the tricky part is preventing the blocking lock situation in the first place.

The culprit of blocking lock scenarios is usually the application design, or the SQL used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible, something not always easy to do.

Redo Log Space Requests

- Metrics
- Troubleshooting

The Oracle RDBMS is able to manage recovery by recording all changes made to a database through the use of redo log files. Oracle writes modifications made to a database to the redo log files, which you can archive to another medium for disaster recovery. The background process that performs these operations is Oracle's Log Writer (LGWR). There is a buffer area in Oracle's System Global Area (SGA) that is used to reduce redo log file I/O, whose size, or lack thereof, can affect performance in a busy system. Sometimes a user process must wait for space in this redo log buffer. Oracle uses the log buffer area to cache redo entries prior to writing them to disk, and if the buffer is not large enough for the redo entry load, waits can occur.

> **TIP:** Click this statistic to drill down to the LGWR/DBWR Detail tab of the I/O Detail view.

Metrics

The two main numbers to watch are:

1 Redo log space requests

2 Redo log wait time

If either statistic strays too far from zero, increase the log_buffer parameter and add more memory to the redo log buffer.

Troubleshooting

If you find a problem, do the following:

1 Edit the Init.ora file for the database.

2 Increase the amount of log_buffer to a higher value (take care not to over-allocate; ensure enough free memory exists on the server before increasing the value).

3 Cycle the Oracle server when possible to allow the new value to take effect.

4 Monitor the new value to see if performance improves.

When adjusting the log_buffer parameter, make sure that you make the amount a multiple of the block size. Otherwise, on database startup, Oracle returns an error stating that you have entered an invalid amount for the redo log buffer.

NOTE: Oracle silently increases the log_buffer parameter if you make it smaller than its default size for a given platform.

Redo Log Space Wait Time

The Oracle RDBMS is able to manage recovery by recording all changes made to a database through the use of redo log files. Oracle writes modifications made to a database to the redo log files, which you can archive to another medium for disaster recovery. The background process that performs these operations is Oracle's Log Writer (LGWR). There is a buffer area in Oracle's System Global Area (SGA) that is used to reduce redo log file I/O, whose size, or lack thereof, can affect performance in a busy system. Sometimes a user process must wait for space in the redo log buffer. Oracle uses the log buffer to cache redo entries prior to writing them to disk. If the buffer area is not large enough for the redo entry load, waits can occur. While waits can be an important indicator of contention, a better measure is the amount of wait time that your users experience.

TIP: Click this statistic to drill down to the LGWR/DBWR Detail tab of the I/O Detail view.

Metrics

The two main numbers to watch are:

1 Redo log space requests

2 Redo log wait time

If either statistic strays too far from zero, increase the log_buffer parameter and add more memory to the redo log buffer.

Troubleshooting

If you find a problem, do the following:

1. Edit the Init.ora file for the database.
2. Increase the amount of log_buffer to a higher value (take care not to over-allocate; ensure enough free memory exists on the server before increasing the value).
3. Cycle the Oracle server when possible to allow the new value to take effect.
4. Monitor the new value to see if performance improves.

When adjusting the log_buffer parameter, make sure you make the amount a multiple of the block size. Otherwise, on database startup, Oracle returns an error stating that you have entered an invalid amount for the redo log buffer.

NOTE: If you make the log_buffer parameter smaller than its default size for a given platform, Oracle silently increases it.

Parallel Query Busy Ratio

Oracle's parallel query feature, when used properly, allows large increases in performance for many SQL and utility operations. Parallel operations can be introduced through SQL hints, specified in an object's DDL, or used in command line arguments for utility programs like SQL*Loader. To effectively service parallel query requests, you must ensure that enough query servers exist in the database instance. The parallel query busy ratio is an indicator of how busy all the servers are on the database in question.

TIP: Click this statistic to drill down to the Parallel Query tab of the Contention Detail view.

Metrics

If the parallel query busy ratio approaches 80-90%, you should think about:

- Adding more query servers to the database.
- Examining parallel requests to ensure they are being used in an efficient and necessary manner.

Troubleshooting

If you find a problem, do the following:

1. Edit the Init.ora file for the database.
2. Increase the amount of parallel_max_servers to a higher value.
3. Cycle the Oracle server when possible to allow the new value to take effect.
4. Monitor the new value to see if performance improves.

You can also investigate the use of the parallel_automatic_tuning parameter in Oracle 8.1 or later.

Used Query Slaves

Oracle's parallel query feature, when used properly, allows large increases in performance for many SQL and utility operations. Parallel operations can be introduced through SQL hints, specified in an object's DDL, or used in command line arguments for utility programs like SQL*Loader. To effectively service parallel query requests, you must ensure that enough query servers exist in the database instance. The used query slave statistic displays how many query servers are actively performing work in a database.

> **TIP:** Click this statistic to drill down to the Parallel Query tab of the Contention Detail view.

Metrics

If the parallel query busy ratio approaches 80-90%, or you see all the available slaves being used, you should think about:

- Adding more query servers to the database.
- Examining parallel requests to ensure they are being used in an efficient and necessary manner.

Troubleshooting

If you find a problem, do the following:

1. Edit the Init.ora file for the database.
2. Increase the amount of parallel_max_servers to a higher value.
3. Cycle the Oracle server when possible to allow the new value to take effect.
4. Monitor new value to see if performance improves.

You can also investigate the use of the parallel_automatic_tuning parameter in Oracle 8.1 or later.

Max Query Slaves

Oracle's parallel query feature, when used properly, allows large increases in performance for many SQL and utility operations. Parallel operations can be introduced through SQL hints, specified in an object's DDL, or used in command line arguments for utility programs like SQL*Loader. To effectively service parallel query requests, you must ensure that enough query servers exist in the database instance. The max query slave statistic displays the maximum number of query slaves allowed on the system as dictated by the parallel_max_servers parameter.

> **TIP:** Click this statistic to drill down to the Parallel Query tab of the Contention Detail view.

Metrics

If the parallel query busy ratio approaches 80-90%, or you see all the available slaves being used, you should think about:

- Adding more query servers to the database.
- Examining parallel requests to ensure they are being used in an efficient and necessary manner.

Troubleshooting

If you find a problem, do the following:

1. Edit the Init.ora file for the database.
2. Increase the amount of parallel_max_servers to a higher value.

3 Cycle the Oracle server when possible to allow the new value to take effect.

4 Monitor the new value to see if performance improves.

Free List Waits

- Metrics

Free lists are lists of Oracle data blocks that contain free space for an Oracle object. Every table has at least one free list. You can use Free lists to locate free blocks of space when a request is made of a table for the insertion of a row. Free list contention can reduce the performance of applications when many processes are involved in the insertion of data to the same table.

TIP: Click this statistic to drill down to the System Waits tab of the Contention Detail view.

Metrics

If free list contention approaches 1%, you should add additional free lists to the most dynamic database objects through the use of the STORAGE parameter.

Enqueue Waits

- Metrics
- Troubleshooting

An enqueue is an advanced locking device that lets multiple database processes share certain resources. Enqueue waits typically occur when sessions wait to be granted a requested lock. Sometimes these locks are internal Oracle locks while other times they could be locks for rows of data in a table.

NOTE: Enqueues are issued implicitly by Oracle.

TIP: Click this statistic to drill down to the System Waits tab of the Contention Detail view.

Metrics

You should investigate any enqueue waits that read consistently above one or more (delta statistics).

Troubleshooting

Removing contention for enqueues is usually an application design issue. Many enqueue waits are either contention for certain rows in the database or the result of database-initiated lock escalation. You should examine the use of indexes to make sure all referencing foreign keys are indexes and that your SQL does not tarry over rows in the database during modification operations. If it does, you should tune your SQL.

Enqueue waits can also be the result of space management tasks (such as objects extending) and disk sorts (mainly in tablespaces that do not make use of the TEMPORARY tablespace parameter).

Contention Detail View

The following tabbed pages are available on the Contention Detail view:

- Buffer Busy Waits
- Latch Detail
- Latch Waits
- Parallel Query
- Rollback Waits
- Session Waits
- System Waits

Buffer Busy Waits Tab

- Metrics

Buffer busy waits occur when a process needs to access a data block in the buffer cache, but cannot, because it is being used by another process. Therefore, it must wait. The Buffer Busy Waits tab of the Contention Detail view shows how long users are waiting for buffers in the buffer cache. The table below describes the information available on the Buffer Busy Waits tab of the Contention Detail view:

Column	Description
Username	The user account being used by the session.
SID	The unique Oracle identifier for the session.
OSUser	The operating system ID of the user session.
Machine	The client machine name used by the session.
Program	The program being executed against the Oracle database by the session.
Total Waits	The total number of buffer busy waits for the session.
Time Waited	The total amount of time waited for the event, in hundredths of a second.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Buffer busy waits normally center around contention for rollback segments, too small an INITRANS setting for tables, or insufficient free lists for tables. The remedy for each situation would be increasing the number of rollback segments, or altering tables to have larger settings for INITRANS to allow for more transactions per data block, and more free lists.

Latch Detail Tab

- Metrics

Protecting the many memory structures in Oracle's SGA are latches. They ensure that one and only one process at a time can run or modify any memory structure at the same instant. Much more restrictive than locks (which at least allow for some collective user interaction), latches have no queuing mechanism so either you get it or you do not and are forced to continually retry.

The Latch Detail tab of the Contention Detail view presents a detailed view of latch activity. The table below describes the information available on the Latch Detail tab of the Contention Detail view:

Column	Description
Name	The name of the latch.
Gets	The number of times the latch was requested by a process.
Misses	The number of failed attempts to acquire the latch on the first attempt.
Spins	The number of times a failed first attempt acquired the latch on a spin.
Immediate Gets	The total number of nowait requests for a latch.
Immediate Misses	The total number of failed nowait attempts to acquire the latch on the first attempt.
Sleeps	The total number of requests that "paused" while waiting for a latch.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Common indicators of latch contention are a latch miss ratio (which records willing-to-wait mode latch requests) and latch immediate miss ratio (which records no-wait mode latch requests). These statistics reflect the overall health of how often latch requests were made and satisfied without waiting. If either of these exceeds 1%, latch contention can be causing system slowdowns.

The table below describes two latch contention situations that you can recognize and quickly fix:

Situation	Remedy
Cache Buffer Chain Latch	This latch is responsible for protecting paths to database block buffers in the buffer cache. Very high I/O loads tend to cause contention for this latch. You can alleviate contention somewhat by adding more buffers to the cache (through the db_block_buffers parameter) or by adding more LRU latch chain latches with the db_block_lru_latches parameter.
Library Cache Latches	Protects cached SQL statements in the library cache area of the Oracle shared pool. Contention for this latch is the usual result of using literals for SQL statements instead of using bind variables.

NOTE: Other routine latch contention problems included the redo allocation and redo copy latches, but these have pretty much been made obsolete in Oracle 8.1.5 or later.

Latch Waits Tab

- Metrics

Protecting the many memory structures in Oracle's SGA are latches. They ensure that one and only one process at a time can run or modify any memory structure at the same instant. Much more restrictive than locks (which at least allow for some collective user interaction), latches have no queuing mechanism so either you get it or you do not and are forced to continually retry. The Latch Waits tab of the Contention Detail view shows detailed information about current latch wait situations. The table below describes the information available on the Latch Waits tab of the Contention Detail view:

Column	Description
Username	The user account being used by the session.
SID	The unique Oracle identifier for the session.
Latch Name	The name of the latch.
Parameter1 Text	The description of the first wait parameter.
Parameter1	The first wait/tuning parameter.
Parameter2 Text	The description of the second wait parameter.
Parameter2	The second wait/tuning parameter.
Seq #	The sequence number that uniquely identifies the wait. It is incremented for each wait.
Wait	The time waited, in hundredths of a second.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Common indicators of latch contention are a latch miss ratio (which records willing-to-wait mode latch requests) and latch immediate miss ratio (which records no-wait mode latch requests). These statistics reflect the overall health of how often latch requests were made and satisfied without waiting. By using latch information on this view (contained in the parameter1 and parameter2 boxes) you can locate the exact object experiencing contention.

Parallel Query Tab

- Metrics

Oracle's parallel query option offers great potential in pursuit of faster response times. The ability to split a request into multiple segments that are processed in parallel can greatly speed up the race to a result, however it is important that Oracle be set up with the proper resources necessary to use parallel functions in an efficient manner. The Parallel Query tab of the Contention Detail view displays information relating to parallel query performance. The table below describes the information available on the Parallel Query tab of the Contention Detail view:

Column	Description
Parallel Query Statistic	The name of the statistic affecting parallel query performance.
Value	The statistical value.

Metrics

Seeing a server's busy statistic that nears or matches the maximum values allotted for parallel query slaves is a sure sign that more query slaves need to be added to the system. Also, seeing a server's highwater value that matches the value for the PARALLEL_MAX_SERVERS parameter can indicate a need to raise the number of query slaves given to Oracle.

Rollback Waits Tab

- Metrics

In systems with heavy data modification loads, the rollback segments can become quite a hot spot. Contention for rollback segments can occur between competing database transactions and cause a performance slowdown.

The Rollback Waits tab of the Contention Detail view displays all the rollbacks available on the system and their wait statistics. The table below describes the information available on the Rollback Waits tab of the Contention Detail view:

Column	Description
Name	The name of the rollback segment.
Gets	The number of header gets (the segment has been used).
Waits	The number of header waits.
WaitRatio	The individual contention ratio for the rollback segment.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Overall, if the global rollback contention ratio approaches 1% or more, you should consider creating more rollback segments. Seeing many individual rollback segments with wait ratios above 1% could also be a cause for concern.

You should also think about creating a specialized, larger, rollback segment for use by long running transactions. Doing so alleviates dynamic rollback extensions and cuts down dramatically on ORA-01555 snapshot-too-old errors. For a database under heavy DML loads, consider creating two rollback tablespaces on separate drives to minimize disk contention among the rollback segments.

Session Waits Tab

- Metrics

Session contention is merely a subset of contention that is viewable at the global database level. Often, it takes analysis at the session level to pinpoint the exact source of contention that is occurring globally. Therefore, you need to become accustomed to viewing contention statistics and waits at the user process level.

When monitoring waits with respect to user sessions, there are two areas of interest:

1. What HAS the user session been waiting on?
2. What IS the user session waiting on?

Oracle records both sets of wait statistics for you. In reviewing previous waits for a session, you can see what types of things have caused a bottleneck in the session. The table below describes the information available on the Sessions Waits tab of the Contention Detail view:

Column	Description
User	The user account being used by the session.
SID	The unique Oracle identifier for the session.
Wait Cause	The name of the wait cause
Program	The program being executed against Oracle by the user session.
Time (sec)	The number of seconds that process has been waiting.
Wait State	The value of the wait state. WAITING indicates that the session is waiting. WAITED UNKNOWN TIME indicates that the duration of last wait is unknown. WAITED SHORT TIME indicates that the last wait was less than 1/100th of a second. WAITED KNOWN TIME indicates that the wait is equal to the time of the last wait.
Object Owner	The name of the object owner.
Object Name	The name of the object.

NOTE: This information is available on both the Session Waits tab of the Users Detail view and the Session Waits tab of the Contention Detail view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

To view the wait times for sessions and overall system wait events, you must set the TIMED_STATISTICS parameter to TRUE for your Oracle databases. You can either set this parameter in your Init.ora file or alter the system dynamically with the ALTER SYSTEM SET TIMED_STATISTICS=TRUE command.

Some waits can be ignored like SQL*Net more data from client and SQL*Net message from client. Others like enqueue waits can indicate a lock contention problem. Waits, like db file scattered reads, can indicate sessions involved in long table scan operations.

System Waits Tab

- Metrics

Waits on a system generally occur for three reasons:

1. A process waits because it has no work to do.
2. A process waits because a requested resource is not available.
3. A process waits for Oracle to perform a prerequisite task for its given operation.

Idle waits (processes waiting because they have no work) should not worry you at all, however the other two wait causes are the ones worth your time and investigation. From a global database level, there are many different types of waits and sources of contention. The System Waits tab of the Contention Detail view presents all the various system waits that have occurred on the system since startup. The table below describes the information available on the System Waits tab of the Contention Detail view:

Column	Description
Wait Event	The name of the wait event.
Total Waits	The total number of waits for the event.
Total Timeouts	The total number of timeouts for the event.
Time Waited	The total amount of time waited for the event in hundredths of a second.
Average Wait Time	The average amount of time waited for the event in hundredths of a second.
Percent Total	The total percentage of waits this event makes up relative to all waits on the system.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Which waits should concern you and which waits can you ignore (for the most part)? Common wait events to ignore include:

- SQL*Net more data from client
- SQL*Net message from client
- client message
- rdbms ipc message
- pipe get
- pmon timer
- smon timer
- Null event

If you see many enqueue waits, this normally indicates either contention for certain rows in the database, or the result of database-initiated lock escalation. You should examine the use of indexes to make sure all referencing foreign keys are indexes and that your SQL is tuned to not tarry over rows in the database during modification operations. Enqueue waits can also be the result of space management tasks (such as objects extending) and disk sorts (mainly in tablespaces that do not make use of the TEMPORARY tablespace parameter).

Buffer busy waits normally center around contention for rollback segments, too small an INITRANS setting for tables, or insufficient free lists for tables. The remedy for each situation would be increasing the number of rollback segments, or altering tables to have larger settings for INITRANS to allow for more transactions per data block, and more free lists.

An interesting wait event is the db file scattered read event. This event is indicative of table scans occurring on the system. Large numbers of them can indicate heavy scan activity and the need to revisit your indexing/physical design.

Network Statistics - Oracle

The Network performance category view displays the following vital network statistics:

- Bytes Received from Client

Bytes Sent to Client

- Metrics

The total number of bytes sent over the network to all Oracle client machines from the database since the last refresh.

Metrics

All users connected to a database via a client/server connection generate network traffic, so seeing decent volumes of network traffic is normal. Out of the ordinary numbers should raise alarms because possible network saturation could occur.

Bytes Received from Client

- Metrics

The total number of bytes sent over the network to the Oracle database from all client machines since the last refresh.

Metrics

All users connected to a database via a client/server connection generate network traffic, so seeing decent volumes of network traffic is normal. Out of the ordinary numbers should raise alarms because possible network saturation could occur.

Roundtrips to/from Client

- Metrics

The total number of Oracle network messages sent to and received from the client.

Metrics

All users connected to a database via a client/server connection generates network traffic, so seeing decent volumes of network traffic is normal. Out of the ordinary numbers should raise alarms because possible network saturation could occur.

Bytes Sent to DBLink

- Metrics

The total number of bytes sent by all client machines over the network to any database link since the last refresh.

Metrics

All users connected to a database via a client/server connection generate network traffic, so seeing decent volumes of network traffic is normal. Out of the ordinary numbers should raise alarms because possible network saturation could occur.

> **NOTE:** Database link traffic could experience longer response times depending on the distributed nature and set up of the involved databases.

Bytes Received from DBLink

- Metrics

The total number of bytes received by all client machines over the network from any database link since the last refresh.

Metrics

All users connected to a database via a client/server connection generate network traffic, so seeing decent volumes of network traffic is normal. Out of the ordinary numbers should raise alarms because possible network saturation could occur.

> **NOTE:** Database link traffic could experience longer response times depending on the distributed nature and setup of the involved databases.

Roundtrips to/from DBLink

- Metrics

The number of Oracle network messages sent over and received from a database link.

Metrics

All users connected to a database via a client/server connection generate network traffic, so seeing decent volumes of network traffic is normal. Out of the ordinary numbers should raise alarms because possible network saturation could occur. Also note that database link traffic could experience longer response times depending on the distributed nature and setup of the involved databases.

Network Contention

- Metrics
- Troubleshooting

Sometimes after sending a request to the database, a user experiences a slowdown in getting a request back from the Oracle server. These delays are usually negligible with respect to the actual amount of time that the user process actually waits. Examining the amount of network-related waits in light of all waits experienced by the database indicates the percentage of waits that the network is contributing to the overall contention scenario.

Metrics

The main thing to do with respect to your database and the network is to minimize the amount of data that travels over it. This means ensuring that only the amount of data your users truly need is being returned to their query tools or front-end application.

Troubleshooting

Eliminating unnecessary large volumes of rows being sent over the wire to clients from the Oracle database is one place to start, as overall traffic can be reduced.

MTS Response Activity

In the past, database professionals avoided Oracle's Multi-threaded server option unless they had a server that was low on horsepower. Instead, most people used the direct option of each user having his or her own SQL*Net thread. This was (and probably still is) the fastest way for Oracle networking to perform its job.

The MTS Response Activity area displays information regarding the amount of waits in the MTS response queues managed by Oracle's shared dispatcher processes.

NOTE: These statistics are only available for Oracle servers with the Multi-threaded server option.

Metrics

When the number of waits for MTS responses begins to climb, adding more shared dispatchers can increase performance.

Troubleshooting

You can add more dispatchers to Oracle both by editing the Init.ora file and changing the mts_dispatchers and mts_max_dispatchers parameters to higher amounts (and then cycling Oracle) or dynamically changing the mts_dispatchers parameter through use of the ALTER SYSTEM command.

MTS Request Activity

In the past, database professionals avoided Oracle's Multi-threaded server option unless they had a server that was low on horsepower. Instead, most people used the direct option of each user having his or her own SQL*Net thread. This was (and probably still is) the fastest way for Oracle networking to perform its job.

The MTS Request Activity area displays information regarding the amount of waits in the MTS request queues managed by Oracle's shared dispatcher processes.

NOTE: These statistics are only available for Oracle servers with the Multi-threaded server option.

Metrics

When the number of waits for MTS requests begins to climb, performance can be increased by adding more shared dispatchers.

Troubleshooting

You can add more dispatchers to Oracle by both editing the Init.ora file and changing the mts_dispatchers and mts_max_dispatchers parameters to higher amounts (and then cycling Oracle) or dynamically changing the mts_dispatchers parameter through use of the ALTER SYSTEM command.

Users Page Statistics

The Users home page includes the following sections:

- Active Connections
- Current Locks
- Current Transactions
- Inactive Connections
- Leading Sessions - CPU
- Leading Sessions - I/O
- Leading Sessions - Memory
- Max Connections
- Max Open Locks
- Max Sessions
- Max Transactions
- Open Cursors
- Sessions Involved in Disk Sorts
- Total Sessions
- Users Blocked
- Users Waiting

Related Topics

Users Detail

Home View Statistics

I/O Page Statistics

Memory Page Statistics

Objects Page Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Active Connections

The Active Connections statistic is the total number of active and open threads currently reported in the database as well as the number of processes actively performing work.

Metrics

None.

> **TIP:** Click this statistic to drill down to the Sessions Overview tab of the Users Detail view.

Current Locks and Max Open Locks

- Metrics
- Troubleshooting

This set of statistics reflects the total number of DML locks currently held on the database as well as the maximum number of DML locks allowed on the system at one time.

For more information, see Current Locks.

> **TIP:** Double-click these statistics to drill down to the All Locks tab of the Locks view.

Metrics

Seeing the current locks on the database approach 80-90% of the maximum limit allowed indicates that you should increase the Init.ora parameter dml_locks.

Troubleshooting

If the total number of locks on the database approaches the dml_locks limit, then:

1. Ensure that users are efficiently issuing COMMITs in transactions to release held locks before editing the Init.ora file.
2. Edit the Init.ora file for the database.
3. Increase the amount of dml_locks to a higher value.
4. Cycle the Oracle server when possible to allow the new value to take effect.

Current Transactions and Max Transactions

- Metrics
- Troubleshooting

This set of statistics reflects the total number of open transactions currently on the database as well as the maximum number of transactions allowed on the system at one time.

> **TIP:** Double-click these statistics to drill down to the Transaction Detail tab of the Users Detail view.

Metrics

Seeing the current transactions on the database approach 80-90% of the maximum allowed limit allowed indicates that you should increase the Init.ora parameter transactions.

Troubleshooting

If the total number of transactions on the database approaches the transactions limit, you should:

1. Edit the Init.ora file for the database.
2. Increase the amount of transactions to a higher value.
3. Cycle the Oracle server when possible to allow the new value to take effect.

Total Connections and Max Connections

This set of statistics reflects the total number of open sessions on the database (both active and inactive) as well as the maximum number of sessions allowed on the system at one time.

> **TIP:** Double-click Total Connections or Max Connections to drill down to the Sessions Overview tab of the Users Detail view.

Metrics

Seeing the open connections on the database approach 80-90% of the maximum process limit allowed indicates that you should increase the process limit imposed on the database.

Troubleshooting

If the total number of connections on the database approaches the processes limit, then:

1. Ensure that users are efficiently logging off and on the database before editing the Init.ora file.
2. Edit the Init.ora file for the database.
3. Increase the amount of processes to a higher value.
4. Cycle the Oracle server when possible to allow the new value to take effect.

Inactive Connections

The Inactive Connections statistic is the total number of threads logged on to the database that are currently idle.

> **TIP:** Click this statistic to drill down to the Sessions Overview tab of the Users Detail view.

Metrics

A large number of inactive users could indicate user sessions that have mistakenly been left logged on. Because each user thread consumes a portion of memory on the Oracle server, to reduce resource usage, you should sever any session that does not need a connection.

Troubleshooting

Drill down into the Sessions Overview tab of the Users Detail view and check sessions that have many seconds idle and/or have been logged on for very long periods of time (as indicated by the logon time column). After verifying that a session is no longer necessary, you can KILL it.

Open Cursors

- Metrics
- Troubleshooting

Open Cursors is the total number of all SQL open cursors that exist on the system. In some cases, Oracle cached cursors that have been open by PL/SQL procedures can be kept open for certain lengths of time, even though the actual activity has ceased.

TIP: Click this statistic to drill down to the Open Cursors tab of the Users Detail view.

Metrics

You should monitor sessions to make sure that they do not approach the Open Cursor limit (specified in the Init.ora file). The parameter, open_cursors, limits how many open cursors (context areas) a session can have open at one time.

Troubleshooting

If the total number of open cursors approaches the open_cursors limit on Oracle8i or earlier, then:

- Ensure that user processes are efficiently using cursors before editing the Init.ora file.
- Edit the Init.ora file for the database.
- Increase the amount of open_cursors to a higher value.
- Cycle the Oracle server when possible to allow the new value to take effect.

If the total number of open cursors approaches the open_cursors limit on Oracle9i or later then:

- Change the open_cursors parameter to FORCE by using the ALTER SYSTEM SET open_cursors=< new value > command.

Users Waiting

- Metrics
- Troubleshooting

User connections that are waiting on a system generally occur for two reasons:

1. A process waits because a requested resource is not available.
2. A process waits for Oracle to perform a prerequisite task for its given operation.

Idle waits (processes waiting because they have no work) are not usually a concern. However the two wait causes mentioned above are the ones worth your time and investigation.

NOTE: This statistic displays on both the Contention performance category view and the Users performance category view.

TIP: Click this statistic to drill down to the Session Wait Detail tab of the Users Detail view.

Metrics

To determine the actual wait causes currently experienced by user connections, you should drill down from the global count of users waiting, into the actual system and user wait details of a database.

Troubleshooting

If you find a problem, drill down into wait details to determine whether the waits are resource-related.

Users Blocked

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches even on large systems. Although Oracle supports unlimited row-level locking, blocking lock situations do crop up. User processes holding exclusive locks and not releasing them via a proper COMMIT generally cause most blocks.

> **TIP:** Click this statistic to drill down to the Blocking Locks tab of the Locks view.

Metrics

You should immediately investigate any indicator above zero for a blocking lock statistic before the situation has a chance to grow out of control.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied. You can issue a KILL against the offending process, which eliminates the user's stranglehold on the objects they were accessing. Other user processes then nearly almost always complete in an instant. Embarcadero Performance Center makes it easier to discover the blocked lock situation, but the trick is to prevent the blocking lock situation in the first place.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. Most DBAs who have had to face Oracle Forms applications have suffered through the dreaded SELECT … FOR UPDATE statements that place unnecessary restrictive locks on nearly every read operation, and know all too well that good coding practice is important. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

Sessions Involved in Disk Sorts

Oracle's SGA is not the only memory structure used by Oracle for database work. One of the other memory areas used by Oracle8i and earlier for normal activity is an area set aside for sort actions. When a sort operation occurs, Oracle attempts to perform the sort in a memory space that exists at the operating system level. If the sort is too large to be contained within this space, it continues the sort on disk - specifically, in the user's assigned TEMPORARY TABLESPACE. Oracle records the overall number of sorts that are satisfied in memory as well as those that end up being finalized on disk. Using these numbers, you can calculate the percentage of memory sorts vs. disk sorts and get a feel for how fast your sort activity is being resolved. Obviously, memory sorts completes many times faster than sorts forced to use physical I/O to accomplish the task at hand.

Oracle9i or later now has the option of running automatic PGA memory management. Oracle has introduced a new Oracle parameter called pga_aggregate_target. When the pga_aggregate_target parameter is set and you are using dedicated Oracle connections, Oracle ignores all of the PGA parameters in the Oracle file, including sort_area_size, hash_area_size and sort_area_retained_size. Oracle recommends that the value of pga_aggregate_target be set to the amount of remaining memory (less a 10% overhead for other server tasks) on a server after the instance has been started.

> **TIP:** Click this statistic to drill down to the Disk Sort Detail tab of the Users Detail view.

Metrics

Consistently seeing non-zero numbers for this statistic (as well as low values observed for the memory sort ratio) can indicate excessive disk sort activity. If you are on Oracle8i or earlier, increase the parameters devoted to memory sorts - sort_area_size and sort_area_retained_size.

If you are using Oracle9i or later, investigate the use of pga_aggregate_target. Once the pga_aggregate_target has been set, Oracle automatically manages PGA memory allocation, based upon the individual needs of each Oracle connection. Oracle9i or later allows the pga_aggregate_target parameter to be modified at the instance level with the alter system command, thereby lets you dynamically adjust the total RAM region available to Oracle9i.

Oracle9i also introduces a new parameter called workarea_size_policy. When this parameter is set to automatic, all Oracle connections benefit from the shared PGA memory. When workarea_size_policy is set to manual, connections allocate memory according to the values for the sort_area_size parameter. Under the automatic mode, Oracle tries to maximize the number of work areas that are using optimal memory and uses one-pass memory for the others.

Troubleshooting

If you find a problem, do the following:

- Edit the Init.ora or SPFILE file for the database.
- Increase the amount of sort_area_size to a higher value (take care not to not over-allocate; ensure enough free memory exists on server before increasing value). Realize that EVERY user receives this amount for sorting).
- Cycle the Oracle server when possible to allow the new value to take effect.
- Monitor new value to see if performance improves.

In addition to increasing the amount of memory devoted to sorting, you should also locate inefficient SQL that causes needless sorts. For example, in an SQL query (to eliminate duplicate rows) UNION ALL does not cause a sort whereas UNION does. People frequently code DISTINCT inappropriately (especially people transferring from Microsoft Access, which uses DISTINCT for most SELECT queries).

There are times you simply cannot stop sort activity. When this happens, you should try to keep it in memory whenever possible. However, large data warehousing systems frequently exhaust RAM sort allotments, so if disk sorts must occur, ensure three things:

- Your user's TEMPORARY TABLESPACE assignment is not the SYSTEM tablespace, which is the default assignment. In Oracle9i or later, you can specify a default tablespace other than SYSTEM for every user account that is created.
- The TEMPORARY TABLESPACE assigned to your users is placed on a fast disk.
- The TEMPORARY TABLESPACE has the tablespace parameter TEMPORARY assigned to it, which allows sort activity to be performed in a more efficient manner.

Leading Sessions - CPU

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Often, user connections can get out of hand with CPU use, and extreme cases have caused headaches at both the database and operating system levels.

The leading CPU session's display allows you to see who runs the leading sessions in your system with respect to CPU usage.

Metrics

Finding one or two users who use the majority of the CPU can indicate runaway or improper processes. By drilling down into the CPU activity of all users, you can quickly see if this is the case.

Leading Sessions - Memory

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problems could include a runaway process, an untuned batch procedure, or some other user-initiated operation. Frequently, user connections can get out of hand with memory consumption, with extreme cases causing headaches at both the database and operating system levels (ORA-4030 errors).

NOTE: This statistic displays on both the Users performance category view and the Memory performance category view.

TIP: Click this statistic to drill down to the Leading Sessions tab of the Memory Detail view.

Metrics

If your database server does not have an overabundance of memory, you should periodically check to see who your heavy memory users are and the total percentage of memory they take. If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing.

Troubleshooting

The Leading Sessions - Memory statistic lets you find the users with the greatest allocations of overall memory. You can drill down into the details to discover the memory components that make up each user's session. The table below describes areas where you should pay particular attention to the amounts of memory usage:

Area	Description
PGA	The Program Global Area is a private memory area devoted to housing the global variables and data structures for a single Oracle process.
Memory Sorts	Contains a count of how many memory sorts a session has performed.
UGA	The User Global Area contains session-specific information regarding open cursors, state information for packages, database link information, and more.

TIP: When using Oracle's Multi-threaded Server (MTS), the UGA can be moved up into the SGA.

Leading Sessions - I/O

When a system undergoes heavy I/O activity, you find that all the user connections are contributing somewhat equally to the overall load. Often however, one or two user connections are responsible for 75% or more of the I/O activity. It can be that a large batch load or another typical process is running that is perfectly okay for your system or there can be a runaway process or rogue connection that you need to track down and possibly eliminate.

The Leading I/O Session statistic displays, with respect to I/O, the leading sessions in your system.

NOTE: This statistic displays on both the Users performance category view and the I/O performance category view.

TIP: Click this statistic to drill down to the Leading Sessions tab of the I/O Detail view.

Metrics

Finding one or more users who are consuming more than 75% of the total I/O load can indicate a runaway or improper process. By drilling down into the I/O activity of all users, you can quickly see if this is the case.

Users Detail

The following tabbed pages are available on the Users Detail page:

- Disk Sort Detail
- Open Cursors
- Session Waits
- Sessions Overview
- System Tablespace
- Transaction Detail

Disk Sort Detail Tab

- Metrics

Something that can degrade a user's or overall database performance is disk sort activity. When a sort operation occurs, Oracle attempts to perform the sort in a memory space, assigned by the DBA, which exists at the operating system level. If the sort is too large to be contained within this space, it continues the sort on disk - specifically, in the user's assigned TEMPORARY TABLESPACE.

The Disk Sort Detail tab of the Users Detail view displays information relating to active disk sorts on a system. The table below describes the information available on the Disk Sort Detail tab of the Users Detail view:

Column	Description
User	The user account a session is using.
SID	The unique Oracle identifier for the session.
Tablespace	The tablespace being used for the disk sort.
Contents	Whether the tablespace has been set to PERMANENT or TEMPORARY.
Extents	The number of extents involved in the sort.
Blocks	The number of blocks involved in the sort.
Bytes	The total number of bytes used in the sort.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Techniques to include in your overall performance strategy are those that relate to minimizing the amount of sort activity overall and specifically sort activity that takes place on disk. A good place to start is by understanding what things cause sorts in the first place. A list of sort-related commands and SQL-related options include:

- CREATE INDEX, ALTER INDEX…REBUILD
- DISTINCT
- ORDER BY
- GROUP BY
- UNION
- INTERSECT
- MINUS
- IN, NOT IN
- Certain unindexed joins
- Certain correlated subqueries

All of these SQL commands have the potential to create a sort. As a DBA, you probably won't know which query sorts perform entirely in memory and which ones go to disk.

There are times you simply cannot stop disk sort activity (such as in data warehousing environments). That being the case, you should ensure the following are true in your database:

- Your users' TEMPORARY TABLESPACE assignment is not the SYSTEM tablespace, which is the default assignment.
- The TEMPORARY TABLESPACE assigned to your users is placed on a fast disk.
- The TEMPORARY TABLESPACE has the tablespace parameter TEMPORARY assigned to it, which allows sort activity to be performed in a more efficient manner.

If your overall memory sort ratio falls below 90%, increase the parameters devoted to memory sorts - sort_area_size and sort_area_retained_size. Keep in mind that Individual users might have the ability to alter their own sessions and increase their sort_area_size assignments. As a DBA, restrict users that have the ALTER SESSION privilege.

In addition to increasing the amount of memory devoted to sorting, you should also locate inefficient SQL that causes needless sorts. For example, in an SQL query (to eliminate duplicate rows) UNION ALL does not cause a sort whereas UNION does. People frequently code DISTINCT inappropriately (especially people transferring from Microsoft Access, which uses DISTINCT for most SELECT queries).

Open Cursors Tab

- Metrics

Programs opening cursors consume Oracle resources and have the potential to degrade performance, especially if their SQL code is inefficient. The Open Cursors tab of the Users Detail view allows you to quickly spot user accounts that have many cursors opened as well as the actual performance statistics for each opened cursor. The first grid displays a summary of the open cursors. The second grid displays the details of open cursors.

The table below describes the information available in the Open Cursor Summary grid on the Open Cursors tab of the Users Detail view:

Column	Description
User	The user account owning the cursors.
SID	The unique Oracle identifier for the session.
Open Cursors	The number of open cursors for the session.

The table below describes the information available in the Open Cursor Detail grid on the Open Cursors tab of the Users Detail view:

Column	Description
SID	The unique Oracle identifier for the session.
SQL Text	The SQL text being used for the cursor.
Disk reads	The number of physical I/O reads.
Buffer gets	The number of buffer gets.
Rows	The number of rows returned for the statement.
Parse Calls	The total of all parse calls to all the child cursors under this parent.
Sharable Memory	The sum of all sharable memory of all child cursors under this parent, in bytes.
Persistent Memory	The sum of all persistent memory of all child cursors under this parent, in bytes.
Runtime Memory	The sum of all ephemeral frame sizes of all children.
Sorts	The sum of sorts that were done for all the children of the parent code.
Loaded Versions	The number of children that are present in the cache and have their context heap loaded.
Loads	The number of times the object was loaded or reloaded.
Executions	The number of times the code was executed since being brought into the shared pool.
First Load Time	The creation timestamp of the parent code.

TIP: To configure a grid to display row numbers, use the Options Editor.

Metrics

The Init.ora parameter OPEN_CURSORS controls the maximum number of open cursors (context areas) a session can have at one time. Seeing individual sessions approaching this limit should concern you. To avoid cursor allocation errors, set it to a very high number. The cursor limit for the Init.ora file was 255, but this limit has been increased in Oracle8.

With respect to individual cursor performance statistics, be on the lookout for cursors with abnormally high physical I/O counts (which can indicate inefficient SQL or indexing schemes), and cursors with high load counts (which can indicate an undersized library cache).

Session Waits Tab

Session contention is merely a subset of contention that is viewable at the global database level. Often, it takes analysis at the session level to pinpoint the exact source of contention that is occurring globally. Therefore, you need to become accustomed to viewing contention statistics and waits at the user process level.

When monitoring waits with respect to user sessions, there are two areas of interest:

- What HAS the user session been waiting on?
- What IS the user session waiting on?

Oracle records both sets of wait statistics for you. In reviewing previous waits for a session, you can see what types of things have caused a bottleneck in the session. The table below describes the information available on the Sessions Waits tab of the Users Detail view:

Column	Description
User	The user account being used by the session.
SID	The unique Oracle identifier for the session.
Wait Cause	The name of the wait cause
Program	The program being executed against Oracle by the user session.
Time (sec)	The number of seconds that process has been waiting.
Wait State	The value of the wait state. WAITING indicates that the session is waiting. WAITED UNKNOWN TIME indicates that the duration of last wait is unknown. WAITED SHORT TIME indicates that the last wait was less than 1/100th of a second. WAITED KNOWN TIME indicates that the wait is equal to the time of the last wait.
Object Owner	The name of the object owner.
Object Name	The name of the object.

NOTE: This information is available on both the Session Waits tab of the Contention Detail view and the Session Waits tab of the Users Detail view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

To view the wait times for sessions and overall system wait events, you must set the TIMED_STATISTICS parameter to TRUE for your Oracle databases. You can either set this parameter in your Init.ora file or alter the system dynamically with the ALTER SYSTEM SET TIMED_STATISTICS=TRUE command.

Some waits can be ignored like SQL*Net more data from client and SQL*Net message from client. Others like enqueue waits can be indicative of a lock contention problem. Waits, like db file scattered reads, can indicate sessions involved in long table scan operations.

Sessions Overview Tab

- Metrics

You can get more detail into dips in the overall buffer cache hit ratio by examining user I/O activity and users' individual cache hit ratios. Frequently you can pinpoint one or more accounts responsible for reducing performance and examine their SQL and other statistics to fix the situation.

The table below describes the information available on the Sessions Overview tab of the Users Detail view:

Column	Description
User	The logon name the session is using.
SID	The unique Oracle identifier for the session.
Serial #	The serial number assigned to the session.
Status	The status of the session, ACTIVE or INACTIVE.
Machine	The name of the client machine that the session is using.
O/S ID	The unique operating system identifier for the target session.
Logon Time	The date/timestamp of when the session first logged on.
Blocked	Whether the session is blocked from doing work by another session.
Seconds Idle	The number of seconds since the last time the session actively performed work.
Hit Ratio	The percentage of times the session obtained data from memory vs. physical I/O activity. The maximum value is 100%.
Program	The program being executed against Oracle by the user session.

NOTE: This information is available on both the Sessions Overview tab of the Memory Detail view and the Sessions Overview tab of the Users Detail view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Observing a poor hit ratio for one or more threads can help you pinpoint sessions that are dragging overall database performance down. Obtaining the current SQL for those sessions could yield information into the exact nature of the problem. Typically, you should investigate hit ratios under 85%. Another critical piece of information is sessions blocked. You should investigate these statistics using the Embarcadero Performance Center Locks view. Two other interesting pieces of information include the logon time and seconds idle. Sessions with high idle times and logon time stamps that are many hours or days old could be candidates for removal from the system.

System Tablespace Tab

- Metrics

Of all your tablespaces, you want to avoid fragmentation problems in your SYSTEM tablespace the most, as this is the major hotbed tablespace for Oracle activities. The easiest way to avoid this is to not allow any user (even the default DBA ID's SYS and SYSTEM) to have access to it. There are three ways to do this:

1 Ensure no user has a DEFAULT or TEMPORARY tablespace assignment of SYSTEM.

2 Ensure no user has a quota set for SYSTEM.

3 Ensure no user has been granted the UNLIMITED TABLESPACE privilege.

TIP: The System Tablespace tab of the Users Detail view identifies users that have privileges on the SYSTEM tablespace.

Metrics

Seeing any user with SYSTEM tablespace privileges should be cause for alarm. To rectify the situation, you can do one of the following:

- Use the ALTER command to change a user's DEFAULT or TEMPORARY tablespace assignment from SYSTEM to a user-defined tablespace.
- Remove any system tablespace quotas assigned to users.
- Revoke the UNLIMITED TABLESPACE privilege from any identified user.

Transaction Detail Tab

- Metrics

Long running transactions have the potential to consume large amounts of resources and cause lock contention headaches. The Transaction Detail tab of the Users Detail view provides a snapshot of transactions in progress. The table below describes the information available on the Transaction Detail tab of the Users Detail view:

Column	Description
User	The user account used by the session.
SID	The unique Oracle identifier for the session.
Rollback Segment	The name of the rollback segment.
Machine	The client machine name the session is using.
Program	The program being executed against Oracle by the session.
Status	The current status of the transaction.
Start Time	The beginning timestamp for the transaction.
Logical	The total logical I/O for the transaction.
Physical	The total physical I/O for the transaction.
Gets	The total consistent gets used by the transaction.
Changes	The total consistent changes produced by the transaction.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Seeing a long running, persistent transaction might indicate a session caught in a loop or other coding mess. Watching a transaction sit idle could clue you into a lock contention problem that exists.

Session Details

The following tabbed pages are available on the Session Detail view for Oracle:

- Session Contention
- Session I/O
- Session Memory
- Session Network
- Session Objects
- Session SQL
- Session Statistics

Session Memory Tab for Oracle

- Metrics

The Session Memory tab of the Session Detail view presents the statistics surrounding a session's memory usage. The table below describes the information available on the Session Memory tab of the Session Detail view for Oracle:

Column	Description
Statistic	The name of the memory related statistic.
Value	The cumulative value for the memory statistic.
Cache Hit Ratio	The percentage of data obtained from memory access vs. physical I/O.

Metrics

Sessions with abnormally high memory usage can affect overall performance at the server level, as this memory (PGA and UGA) is allocated outside of the Oracle SGA (unless the multi-threaded server option is being used). If cache hit ratios at the session level are lower than 85 percent, data access can be inefficient.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session I/O Tab for Oracle

- Metrics

The Session I/O tab of the Session Detail view presents the statistics surrounding a session's I/O activity. The table below describes the information available on the Session I/O tab of the Session Detail view for Oracle:

Column	Description
Statistic	The name of the I/O related statistic.
Value	The cumulative value for the I/O statistic.

Metrics

Seeing high values for physical reads and writes can indicate an inefficient session. Large numbers of physical reads can imply a session with too many large table scans or inefficient SQL operations. Large numbers of physical writes can be okay for sessions inputting large volumes of data into the database. Or, they could also indicate a session involved in heavy disk sort activity.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session Contention Tab for Oracle

- Metrics

The Session Contention tab of the Session Detail view presents information relating to resources on which the current session is waiting. The first grid displays user waits. The second grid displays user locks. The table below describes the information available in the User Waits grid on the Session Contention tab of the Session Detail view for Oracle:

Column	Description
Wait Cause	The wait event being experienced by the session.
Program	The program the session is executing against Oracle.
Seconds	The number of seconds the session has spent in the wait.
State	The status of the wait event (WAITING, etc.)

The table below describes the information available in the User Locks grid on the Session I/O tab of the Session Detail view for Oracle:

Column	Description
User	The user account being used by the session.
Terminal	The client machine name used by the session.
SID	The unique Oracle identifier for the session.
Serial #	The serial number for the session.
Table	The object locked by the session.
Lock Mode	The lock mode used by the session.
Title	The lock request issued by the session.

Metrics

You can ignore some waits, like the SQL*Net more data from client and SQL*Net message from client. Others, like enqueue waits, can be indicative of a lock contention problem. Waits, like db file scattered reads, can indicate sessions involved in long table scan operations.

Locks that are held for unusually long periods require further investigation. The application logic can be inefficient or the program is not issuing COMMITs frequently enough. The culprit of blocking-lock scenarios is usually the application design, or the SQL used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the correct SQL to get the job done is an art. Most DBAs who have had to face Oracle Forms applications have suffered through the dreaded SELECT…FOR UPDATE statements. These place unnecessary restrictive locks on nearly every read operation. The key to avoiding lock contention is to process user transactions as quickly and efficiently as possible.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session Objects Tab for Oracle

- Metrics

The Session Objects tab of the Session Detail view presents information regarding the objects in use by a session. The table below describes the information available in the Objects Accessed grid on the Session Objects tab of the Session Detail view for Oracle:

Column	Description
Owner	The owner of the object.
Type	The type of object (TABLE, VIEW, etc.).
Object	The name of the object.

The Rollback Segments Accessed grid displays the names of rollback segments currently being used by the session.

Metrics

Once you have an idea of which objects your users access most often, you can refine some processes to facilitate access to them. You can use the Oracle 8 concept of the KEEP buffer cache to force Oracle to hold often-referenced data for data objects. The KEEP buffer cache is ideal for holding small look-up tables. If you have an earlier version of Oracle or do not want to split up your current buffer cache, you can use the CACHE table attribute to encourage Oracle to keep data blocks of CACHE'd tables at the most recently used end of the LRU buffer cache chain.

If you consistently see a session with active rollbacks, it can indicate locks are being held for long durations. It can also indicate that a session is using code without frequent enough COMMIT points.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session Network Tab for Oracle

- Metrics

The Session Network tab of the Session Detail view presents information about requests being sent to and from the database with respect to the current session. The table below describes the information available on the Session Network tab of the Session Detail view for Oracle:

Column	Description
Statistic	The name of the SQL*Net related statistic.
Value	The cumulative value of the SQL*Net related statistic.

Metrics

None.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session SQL Tab for Oracle

- Metrics

The Session SQL tab of the Session Detail view presents information relating to any current SQL issued by the session as well as any SQL previously issued by the session.

Metrics

To determine access paths, you should export and run through an EXPLAIN PLAN session any SQL that you suspect of inefficient access.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session Statistics Tab for Oracle

- Metrics

The Session Statistics tab of the Session Detail view presents information relating to all recorded performance and miscellaneous statistics for the current session. The table below describes the information available on the Session Statistics tab of the Session Detail view for Oracle:

Column	Description
Statistic	The name of the session related statistic.
Value	The cumulative value of the session related statistic.

Metrics

None.

> **NOTE:** To configure the grid to show/hide row numbers, use the Options Editor.

Microsoft SQL Server Expert Guide

This section includes expert help for all Microsoft SQL Server statistics in the Embarcadero Performance Center views:

- Home View Statistics
- Database Page Statistics
- Contention Statistics
- I/O Page Statistics
- Memory Page Statistics
- Network Statistics
- OS Page Statistics
- Space Page Statistics
- Users Page Statistics

Home View Statistics

The Embarcadero Performance Center Home view lets you review availability and overall performance of all monitored databases from a single window. Statistics on the Home view are organized into the following categories:

- Contention Vital Signs
- I/O Vital Signs
- Memory Vital Signs
- Network Vital Signs
- Space Vital Signs
- Users Vital Signs

Related Topics

Database Page Statistics

Contention Statistics

I/O Page Statistics

Memory Page Statistics

Network Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Users Page Statistics

Memory Vital Signs

The following memory statistics are on the SQL Server Home view:

- Ad Hoc SQL Hit Ratio
- Buffer Cache Hit Ratio
- Log Cache Hit Ratio
- Procedure Plan Hit Ratio

Buffer Cache Hit Ratio

- Metrics
- Troubleshooting

Data read from memory will produce end-user response times many times faster than when that same data is read from disk. Keeping physical I/Os to an absolute minimum is one of the purposes of the SQL Server buffer/data cache.

The buffer cache hit ratio is a terrific indicator of how often user requests for data are satisfied through memory vs. being physically read from disk.

> **TIP:** Click this statistic to drill down to the Buffer Cache tab of the Memory Detail view.

Metrics

To help ensure excellent performance, you want to keep your cache hit ratio in the neighborhood of 90% or higher. However, you should be aware that every server has its own 'personality' and might exhibit excellent performance with below average readings for the cache hit ratio. You should also be aware that excessive logical I/O activity can produce a very high cache hit ratio while actually degrading overall database performance.

Consistently viewed low readings (60% or lower) might require tuning attention on the part of the DBA.

Troubleshooting

Ensure SQL Server is configured to use as much physical memory as possible by checking the Max Server memory configuration option. Also, consider increasing your SQL Server Min. Memory parameter to allocate more memory to SQL Server. (Note that to obtain optimal values for these parameters, an option is to install more physical RAM to your server.) Check for any large objects that are pinned in memory that could possibly be removed.

Often a user process is taking a large amount of memory due to an inordinate amount of I/O.

Procedure Hit Ratio

- Metrics
- Troubleshooting

The SQL Server procedure cache is used to hold the execution plans for all Transact-SQL statements currently executing in the server. When a user executes a Transact-SQL statement, SQL Server looks in the procedure cache for a query plan to use.

This statistic is the percentage of query plan requests generated by stored procedures that are found in the procedure cache area. The percentage of times that a statement's plan and definition can be referenced in memory, the better the procedure execution time.

> **TIP:** Click this statistic to drill down to the SQL Cache tab of the Memory Detail view.

Metrics

A high procedure cache hit rate is a desirable thing. You should strive for a hit ratio between 95-100%, with 95% being a good performance benchmark for code reference. Note that when a database is first started, the procedure cache hit rate will not be at an optimal level because all code being used will be relatively new, and as such, must be read in from disk and placed into the cache. If, however, after a solid hour or two of steady database time, the procedure cache hit rate has not increased to desirable levels, you should look into the possibility of increasing the amount of memory allocated to the cache.

Troubleshooting

First, ensure SQL Server is configured to use as much physical memory as possible by checking the Max Server Memory configuration option. Also, consider increasing your SQL Server Min Memory parameter to allocate more memory to SQL Server. (Note that to obtain optimal values for these parameters, an option is to install more physical RAM to your server.) Check for any large objects that are pinned in memory that could possibly be removed.

Ad Hoc SQL Hit Ratio

When an ad hoc SQL statement is issued, the query plan is then stored in the SQL Server procedure cache area. If the identical ad hoc statement is launched in the future, SQL Server uses the query plan already stored in the procedure cache if it is still there. The Ad Hoc SQL Hit Ratio statistic defines the percentage of times that a query plan for an ad hoc SQL statement is found in the procedure cache.

> **TIP:** Click this statistic to drill down to the SQL Cache tab of the Memory Detail view.

Metrics

A high ad hoc hit rate is desirable, but is harder to maintain at a high level than something like a procedure cache hit rate. Therefore, an 80% or greater ad hoc cache hit rate is a good performance benchmark for code reference. Note that when a database is first started, the ad hoc cache hit rate will not be at an optimal level because all code being used will be relatively new, and as such, must be read in from disk and placed into the cache. If, however, after a solid hour or two of steady database time, the ad hoc cache hit rate has not increased to desirable levels, you should look into the possibility of increasing the amount of memory allocated to the cache.

Troubleshooting

First, ensure SQL Server is configured to use as much physical memory as possible by checking the Max Server Memory configuration option. Also, consider increasing your SQL Server Min Memory parameter to allocate more memory to SQL Server. (To obtain optimal values for these parameters, an option is to install more physical RAM to your server.) Check for any large objects that are pinned in memory that could possibly be removed.

Log Cache Hit Ratio

The Log Cache Hit Ratio represents the percentage of log cache reads satisfied from the log cache.

> **TIP:** Click this statistic to drill down to the Log Cache tab of the Memory Detail view.

Metrics

None.

Troubleshooting

A low percentage on this statistic is not necessarily a bad sign, as it is possible that the information needed from the log will not be readily available in memory.

I/O Vital Signs

The following I/O statistics are on the SQL Server Home view:

- I/O Errors
- Log Flushes
- Total Server Reads
- Total Server Writes

> **TIP:** Click any of these statistics to open the I/O performance category view.

Total Server Reads

- Metrics

Total Server Reads reflect total number of physical reads performed by the database server since the last refresh inside Embarcadero Performance Center.

Metrics

Large numbers of physical reads could reflect a data or procedure cache that is too small. You should examine the data and procedure cache hit rates to determine the overall effectiveness of logical vs. physical I/O.

Total Server Writes

- Metrics

The Total Server Writes value reflects the total number of physical writes performed by the database server since the last refresh inside Embarcadero Performance Center.

Metrics

None.

Log Flushes

- Metrics

The Log Flushes statistic represents the total number of log pages for all databases written to disk by the log writer process. A log flush occurs when SQL Server writes all changes from the database's log cache out to the database's log files on disk.

Metrics

Increasing numbers observed for log flushes should not cause concern unless the I/O subsystem of the server appears overwhelmed. In addition, to minimize I/O contention between a database and its accompanying log, it is wise to place database files and log files on separate disks.

I/O Errors

- Metrics
- Troubleshooting

I/O Error Rate reflects total number of I/O errors (errors during read and write operations) encountered by the server since the last refresh inside Performance Center. I/O Error Rate is a percentage based on Total I/O (the sum the physical reads and writes).

Metrics

You should observe few, if any errors.

Troubleshooting

If you notice any errors, you should check the SQL Server error log for details.

Contention Vital Signs

The following contention statistics are on the SQL Server Home view:

- Blocked Users
- Deadlocks
- Latch Waits
- Lock Timeouts

Blocked Users

- Metrics
- Troubleshooting

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches even on large systems. Blocks are most often caused by user processes holding exclusive locks and not releasing them via a proper COMMIT frequency. Unless a process times out via an application timeout mechanism, or the process has specified a timeout period via the SET LOCK_TIMEOUT command, a process waiting for a lock will wait indefinitely.

Metrics

You should immediately investigate any indicator above zero, before the situation has a chance to mushroom.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects they were accessing. Then other user processes, almost always, complete in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Center, but preventing the blocking lock situation in the first place is where it gets tricky. The DBA can drill down into Users Detail and see all current blocking locks, learning exactly which sessions are holding the currently restrictive locks.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process will wait for a lock before timing out.

Deadlocks

The Deadlocks statistic represents the number of deadlocks per second detected by SQL Server. Deadlocks occur when processes cannot proceed because they are waiting on a set of resources held by each other or held by other processes.

> **TIP:** Click this statistic to open the Contention performance category view.

Metrics

Consistently seeing deadlock counts greater than zero will indicate that some user processes are experiencing delays in completing their work. When SQL Server identifies a deadlock, it resolves the situation by choosing the process that can break the deadlock. This process is termed the deadlock victim. SQL Server rolls back the deadlock victim's transaction, and then notifies the process' application by returning an error message. It also cancels the process' request and allows the transactions of the remaining processes to continue.

SQL Server always attempts to choose the least expensive thread running the transaction as the deadlock victim.

Troubleshooting

Because SQL Server automatically resolves deadlock situations, you should do proactive work to prevent them in the first place.

The culprit of most blocking lock and deadlock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

You can change default deadlock behavior by using the SET DEADLOCK_PRIORITY command, which reprioritizes a process' position in a deadlock situation.

Lock Timeouts

- Metrics

Number of lock requests per second that timed out, including internal requests for NOWAIT locks.

> **TIP:** Click this statistic to open the Contention performance category view.

Metrics

None.

Latch Waits

- Metrics
- Troubleshooting

The Latch Waits statistic represents the number of latches per second that could not be satisfied immediately by SQL Server. Latches can be thought of as lightweight, mini-locks that are used to protect actions and resources used inside transactions.

> **TIP:** Click this statistic to drill down to the Waits tab of the Contention Detail view.

Metrics

Latch waits rarely impact performance, however seeing latch waits with high wait time might indicate an area that needs further investigation.

Troubleshooting

Wait events can be hard to interpret at times. If you see a particular event that has caused a lot of wait time, you can review the information in this link (Microsoft Knowledge Base Article - 244455) to help understand the cause and potential remedy:

http://support.microsoft.com/default.aspx?scid=kb;en-us;Q244455

Space Vital Signs

The following space statistics are on the SQL Server Home view.

- Databases Low on Space
- Logs Low on Space

TIP: Click either of these statistics to open the Space performance category view.

Databases Low on Space

A SQL server contains many databases, some of which are devoted to system-level activities (the master and tempdb databases, for example) and others that hold user data. The Databases Low on Space statistic indicates databases that have fallen below a recommended percentage of free space.

Metrics

This might or might not be a problem. Some databases are not dynamic in nature (meaning they are not expected to grow in size) and are sized so very little free space is present. However, growing databases are another situation and might require DBA intervention if their free space amounts run low.

Troubleshooting

If the percent used amount of a database is approaching problematic levels, there are three ways a DBA can rectify the situation:

1. The DBA can resize the current file(s) used by the database via an ALTER DATABASE … MODIFY FILE command
2. The DBA can add a new file to the database via the ALTER DATABASE … ADD FILE command.
3. The DBA can modify the file(s) used by the database to automatically grow by using the ALTER DATABASE … MODIFY FILE … FILEGROWTH command. You should also ensure that the MAXSIZE setting for each file is set appropriately.

Of course, the DBA should also ensure that enough physical space exists on the server to accommodate additional database space.

Logs Low on Space

Each database in SQL Server has a transaction log, which is a serial record of all modifications that have occurred in a database as well as the transactions that caused each change. The Logs Low on Space statistic indicates transaction logs that have fallen below a recommended percentage of free space.

Metrics

If any log's used space exceeds the Performance Center recommended thresholds, then the DBA should take action to ensure that the log does not run out of available free space.

Troubleshooting

There are several things a DBA can do to ensure that a database's log does not run out of available free space:

1. First, most transactional-oriented databases should have their logs assigned to a separate physical drive than the database. Reasons for doing so include:
   - It prevents competition for space between the log and the database itself.
   - It allows the log to be monitored for space more effectively.
   - It improves performance.
2. If the database is not critical in nature, you can set the truncate log on checkpoint option trunc log on chkpt, which will eliminate any non-active space in the log when a database checkpoint occurs.
3. Critical databases needing higher levels of recovery should have schedules established that regular perform transaction log dumps. Doing so ensures better recovery scenarios as well as a reduced risk of the transaction log running out of space.
4. If a critical transaction log becomes full, it might be impossible to use standard procedures to dump transactions and reclaim space. The dump operation will likely have to incorporate the no log or truncate only options.
5. If a transaction log continuously approaches dangerously low levels of free space, then the DBA should allow the underlying file(s) of the log to automatically grow to meet the demand. This can be accomplished by using the ALTER DATABASE … MODIFY FILE … FILEGROWTH command. You should also ensure that the MAXSIZE setting for each file is set appropriately.

The DBA should also be on the lookout for large load or data modification operations that do not make use of prudently timed commit points. A single, large transaction has the ability to overwhelm any transaction log since only non-active space in the transaction log is removed from log dumps or truncation operations.

Users Vital Signs

The following users statistics are on the SQL Server Home view:

- Active Connections
- Current Locks
- Total Connections
- Transactions

Total Connections

- Metrics
- Troubleshooting

Every connection to SQL Server spawns one or more processes, each uniquely identified by what SQL Server calls a SPID. The Total Connections display shows the number of user and system processes currently reported in the SQL Server instance. This number includes both active and inactive processes.

TIP: Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

You should view the total number of connections in light of the maximum number of processes allowed to connect to SQL Server. The user connections parameter specifies the maximum number of user processes that can simultaneously connect to a SQL Server instance.

Troubleshooting

If the total number of connections approaches the number of user connections limit then:

1. Edit the configuration file for SQL Server.
2. Increase the amount of user connections to a higher value.
3. Cycle SQL Server when possible to allow the new value to take effect.

Active Connections

The Active Connections statistic represents the total number of active and open threads reported on the server. This number displays the number of processes actively performing work.

TIP: Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

None.

Transactions

- Metrics

This statistic represents the total number of active transactions in the SQL Server instance that have not yet been committed, or are waiting on a blocking lock to complete.

TIP: Click this statistic to open the Users performance category view.

NOTE: Embarcadero Performance Center displays this statistic on the Users performance category view as Active Transactions.

Metrics

None.

Current Locks

This statistic represents the total number of granted, converting, and waiting lock requests.

TIP: Click this statistic to drill down to the All Lock tab of the Locks view.

Network Vital Signs

The following network statistics are on the SQL Server Home view:

- Logins
- Logouts
- Packets Received
- Packets Sent

> **TIP:** Click any of these statistics to open the Network performance category view.

Packets Received

- Metrics

SQL Server counts incoming packets from the time the server is initiated. This statistic displays the delta count of all incoming packets received over the network since the last Embarcadero Performance Center sampling.

Metrics

None.

Packets Sent

- Metrics

SQL Server counts outgoing packets from the time the server is initiated. This statistic displays the delta count of all outbound packets written to the network the last Embarcadero Performance Center sampling.

Metrics

None.

Logins

- Metrics

This statistic represents the total number of logins started per second.

Metrics

None.

Logouts

- Metrics

This statistic represents the total number of logout operations started per second.

Metrics

None.

Memory Page Statistics

The Memory view includes the following statistics:

- Ad Hoc SQL Hit Ratio
- Buffer Cache
- Buffer Cache Hit Ratio
- Buffer Cache Usage
- Memory Usage
- Procedure Hit Ratio
- Session Leaders - Memory
- SQL Cache
- SQL Cache Usage

Related Topics

Memory Detail View

Home View Statistics

Database Page Statistics

Contention Statistics

I/O Page Statistics

Network Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Users Page Statistics

Buffer Cache Hit Ratio

- Metrics
- Troubleshooting

Data read from memory will produce end-user response times many times faster than when that same data is read from disk. Keeping physical I/Os to an absolute minimum is one of the purposes of the SQL Server buffer/data cache.

The buffer cache hit ratio is a terrific indicator of how often user requests for data are satisfied through memory vs. being physically read from disk.

> **TIP:** Click this statistic to drill down to the Buffer Cache tab of the Memory Detail view.

Metrics

To help ensure excellent performance, you want to keep your cache hit ratio in the neighborhood of 90% or higher. However, you should be aware that every server has its own 'personality' and might exhibit excellent performance with below average readings for the cache hit ratio. You should also be aware that excessive logical I/O activity can produce a very high cache hit ratio while actually degrading overall database performance.

Consistently viewed low readings (60% or lower) might require tuning attention on the part of the DBA.

Troubleshooting

Ensure SQL Server is configured to use as much physical memory as possible by checking the Max Server memory configuration option. Also, consider increasing your SQL Server Min. Memory parameter to allocate more memory to SQL Server. (Note that to obtain optimal values for these parameters, an option is to install more physical RAM to your server.) Check for any large objects that are pinned in memory that could possibly be removed.

Often a user process is taking a large amount of memory due to an inordinate amount of I/O.

Procedure Plan Hit Ratio

The SQL Server procedure cache is used to hold the execution plans for all Transact-SQL statements currently executing in the server. When a user executes a Transact-SQL statement, SQL Server looks in the procedure cache for a query plan to use.

This statistic is the percentage of query plan requests generated by stored procedures that are found in the procedure cache area. The percentage of times that a statement's plan and definition can be referenced in memory, the better the procedure execution time.

> **TIP:** Click this statistic to drill down to the SQL Cache tab of the Memory Detail view.

Metrics

A high procedure cache hit rate is a desirable thing. You should strive for a hit ratio between 95-100%, with 95% being a good performance benchmark for code reference. Note that when a database is first started, the procedure cache hit rate will not be at an optimal level because all code being used will be relatively new, and as such, must be read in from disk and placed into the cache. If, however, after a solid hour or two of steady database time, the procedure cache hit rate has not increased to desirable levels, you should look into the possibility of increasing the amount of memory allocated to the cache.

Troubleshooting

First, ensure SQL Server is configured to use as much physical memory as possible by checking the Max Server Memory configuration option. Also, consider increasing your SQL Server Min Memory parameter to allocate more memory to SQL Server. (Note that to obtain optimal values for these parameters, an option is to install more physical RAM to your server.) Check for any large objects that are pinned in memory that could possibly be removed.

Ad Hoc SQL Hit Ratio

When an ad hoc SQL statement is issued, the query plan is then stored in the SQL Server procedure cache area. If the identical ad hoc statement is launched in the future, SQL Server uses the query plan already stored in the procedure cache if it is still there. This statistic defines the percentage of times that a query plan for an ad hoc SQL statement is found in the procedure cache.

TIP: Click this statistic to drill down to the SQL Cache tab of the Memory Detail view.

Metrics

A high ad hoc hit rate is desirable, but is harder to maintain at a high level than something like a procedure cache hit rate. Therefore, an 80% or greater ad hoc cache hit rate is a good performance benchmark for code reference. Note that when a database is first started, the ad hoc cache hit rate will not be at an optimal level because all code being used will be relatively new, and as such, must be read in from disk and placed into the cache. If, however, after a solid hour or two of steady database time, the ad hoc cache hit rate has not increased to desirable levels, you should look into the possibility of increasing the amount of memory allocated to the cache.

Troubleshooting

First, ensure SQL Server is configured to use as much physical memory as possible by checking the Max Server Memory configuration option. Also, consider increasing your SQL Server Min Memory parameter to allocate more memory to SQL Server. (To obtain optimal values for these parameters, an option is to install more physical RAM to your server.) Check for any large objects that are pinned in memory that could possibly be removed.

SQL Cache

- Metrics

The total percentage of memory that SQL Server is using in its SQL/procedure cache. Microsoft has begun to transition from the term “procedure cache” to “SQL cache” to define this area of memory. The reason being that in SQL Server's past, this area was devoted exclusively to holding query plans for stored procedures only.

The SQL Cache (procedure cache) is the part of the SQL Server memory pool that is used to store execution plans for Transact-SQL batches, stored procedures, and triggers. Execution plans record the steps that SQL Server must take to produce the results specified by the Transact-SQL statements contained in the batches, stored procedures, or triggers.

Metrics

None.

Memory Usage

- Metrics
- Troubleshooting

SQL Server is allocated memory at start up that will fall between the range of two configurable parameters: Min Memory and Max Memory. This statistic shows you the amount of SQL Server memory allocated at present. In a normal SQL Server system, this number fluctuates as SQL Server uses more memory and releases unused memory back to the operating system. SQL Server also works in conjunction with the Windows memory manager, deallocating space when it detects that the operating system is unable to satisfy memory requests of other operating system processes.

Metrics

An excessively high memory allocation may mean that the SQL Server instance is working in a memory deficient environment.

Troubleshooting

Increase the Max Memory parameter to provide SQL Server with more memory resources. Note that this may require adding physical RAM to the machine to satisfy the requests of both SQL Server and the Windows operating system.

Buffer Cache

- Metrics
- Troubleshooting

The total amount of memory, in megabytes, that is allocated for the SQL Server buffer cache. Each instance of SQL Server has its own buffer cache where it stores recently used data pages to reduce physical I/O. The goal is to make the buffer cache large enough to maximize the ratio of logical reads to physical reads, but not so large that excessive memory swapping starts generating physical I/O to the pagefile. (Instances of SQL Server 2000 do this automatically under the default configuration settings.)

> **TIP:** Click either of these statistics to drill down to the Buffer Cache tab of the Memory Detail view.

Metrics

A percentage used consistently remaining close to 100% indicates a deficient amount of memory available to SQL Server.

Troubleshooting

First, ensure SQL Server is configured to use as much physical memory as possible by checking the Max Server Memory configuration option. Also, consider increasing your SQL Server Min Memory parameter to allocate more memory to SQL Server. (Note that to obtain optimal values for these parameters, an option is to install more physical RAM to your server.) Check for any large objects that are pinned in memory that could possibly be removed.

Session Leaders - Memory

- Metrics
- Troubleshooting

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Frequently, user connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system level.

The leading memory session's display identifies the top users in the server with respect to memory consumption.

Metrics

If your SQL Server machine does not have an overabundance of memory, you should periodically check to see who your heavy memory users are along with the total percentage of memory each takes up. If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing.

Troubleshooting

You can use the Session Leaders - Memory statistic to find the users with the greatest current allocations of overall memory. Runaway processes can be immediately terminated from within the Embarcadero Performance Center Client.

Cache Usage

- Metrics
- Troubleshooting

The total MB of space that SQL Server is using in the buffer cache. Each instance of SQL Server has its own buffer cache where it stores recently used data pages to reduce physical I/O. The goal is to make the buffer cache large enough to maximize the ratio of logical reads to physical reads, but not so large that excessive memory swapping starts generating physical I/O to the pagefile. (Instances of SQL Server 2000 do this automatically under the default configuration settings.)

Metrics

A percentage used consistently remaining close to 100% indicates a deficient amount of memory available to SQL Server.

Troubleshooting

First, ensure SQL Server is configured to use as much physical memory as possible by checking the Max Server Memory configuration option. Also, consider increasing your SQL Server Min Memory parameter to allocate more memory to SQL Server. (Note that to obtain optimal values for these parameters, an option is to install more physical RAM to your server.) Check for any large objects that are pinned in memory that could possibly be removed.

SQL Cache Size

- Metrics

The total MB of space that SQL Server is using in its SQL/procedure cache. Microsoft has begun to transition from the term “procedure cache” to “SQL cache” to define this area of memory. The reason being that in SQL Server's past, this area was devoted exclusively to holding query plans for stored procedures only.

The SQL Cache (procedure cache) is the part of the SQL Server memory pool that is used to store execution plans for Transact-SQL batches, stored procedures, and triggers. Execution plans record the steps that SQL Server must take to produce the results specified by the Transact-SQL statements contained in the batches, stored procedures, or triggers.

Metrics

None.

SQL Cache Free

- Metrics

The total MB of space that is free in the SQL Server SQL/procedure cache. Microsoft has begun transitioning from the term “procedure cache” to “SQL cache” to define this area of memory. In the past this area was devoted exclusively to holding query plans for stored procedures only, but this is no longer the case.

The SQL Cache (procedure cache) is responsible for storing exection plans for Transact-SQL batches, stored procedures, and triggers. Execution plans record the steps that SQL Server must take to produce the results specified by the Transact-SQL statements contained in the batches, stored procedures, or triggers.

Metriics

None.

Memory Detail View

The Memory Detail view includes the following tabbed pages:

- Buffer Cache
- Log Cache
- SQL Cache
- Leading Sessions

Buffer Cache Tab

- Metrics

The buffer cache is a memory pool of buffer pages into which SQL Server reads data pages. The Buffer Cache tab of the Memory Detail view displays the top twenty objects that reside in the buffer cache. The table below describes the information available on the Buffer Cache tab of the Memory Detail view:

Information	Description
Database	The database name of where the object resides.
Owner	The owner of the object.
Table Name	The name of the table whose data pages have been read into the buffer cache.
Index Name	The name of the index whose pages have been read into the buffer cache.
Size (KB)	The size of the table or index in kilobytes.
Pinned	Whether or not the table has been pinned in memory.

Metrics

Keep a close eye on tables that have been pinned in memory. While it is wise to pin frequently-used, small lookup tables in memory, pinning larger tables can lead to overcrowding in the buffer cache and might result in thrashing behavior as SQL Server constantly moves pages in and out of the cache to satisfy user requests for data.

SQL Cache Tab

- Metrics

Beginning with SQL Server version 7.0, other SQL-related code objects might be placed in memory for reuse. Once SQL Server parses through and places a set of SQL of program code in memory, response time can be increased for subsequent calls to the same set of SQL or SQL code object. The SQL Cache tab of the Memory Detail view uses two grids to communicate the contents of the SQL/Procedure cache. The first grid displays the summary information and the second grid displays detail information about the SQL/Procedure cache. The table below describes the information available in the SQL Cache Summary grid on the SQL Cache tab of the Memory Detail view:

Column	Description
Cache Type	The category of SQL code inside the cache (ad hoc SQL, executable plan, etc.)
Object Count	The total objects for a particular cache type.
Cache Used (KB)	The total amount of the SQL/Procedure cache used by the particular cache type.

The table below describes the information available in the SQL Cache Details grid on the SQL Cache tab of the Memory Detail view:

Information	Description
Object Type	The type of object (ad hoc SQL, etc.)
Cache Type	The category of SQL code inside the cache (executable plan, etc.)
Name	The name of the object (if applicable).
Database	The database where the code originated.
User	The owner of the object or code.
Use Count	The number of uses for the object.
SQL Bytes	The amount of SQL bytes used by the code object.
Size (KB)	The size used by the object in kilobytes.
SQL	The actual SQL statement or code being executed.

Metrics

Seeing many objects of the ad hoc SQL type might indicate an environment where sessions are submitting a lot of SQL requests. Many SQL Server DBAs like to control code submissions through stored procedures.

Log Cache Tab

- Metrics
- Troubleshooting

Before ever writing transactions to disk, the log manager of SQL Server formats everything in memory. This area of memory is known as the log cache. The log writer of SQL Server moves through the log caches when transactions are committed (as well as other events) and flushes each cache out to disk. SQL Server also reads from the cache when log records are needed. The table below describes the information available on the Log Cache tab of the Memory Detail view:

Information	Description
Database	The name of the SQL Server database.
Log Cache Reads	The number of reads from the log cache.
Log Flushes	The number of times data was flushed from the log to disk.
Log Flush Waits	The number of times the log had to wait before flushing data to disk.
Log Flush Wait Time	The amount of time the log waited before flushing data to disk in milliseconds.
Log Growths	The number of times the log had to physically grow in size to meet the need for more space.
Log Shrinks	The number of times the log contracted in physical size.

Metrics

Seeing high amounts of wait time for log flushes could indicate a bottleneck at the disk level. A log that shows high number of growths likely indicates an undersized log. While automatic growth can alleviate out-of-space conditions, many growth operations can slow down overall operations. It is better to have a properly sized transaction log that allows SQL Server to continually enlarge it in size when needed.

Troubleshooting

Consider relocating logs showing high amounts of wait time to faster disks. For logs showing high numbers of growths, permanently enlarge the log(s) via an ALTER DATABASE command that will resize the log files.

Leading Sessions Tab

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. User connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system levels.

Embarcadero Performance Center displays information to help you find processes that are using the most memory on the server on the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view.

The table below describes the information available on these tabs:

Column	Description
PID	The process ID of the connected session.
User Name	The user name assigned to the process.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The machine name that originated the process.
Program	The program the process has invoked against SQL Server.
Memory Usage	The total number of memory pages allocated to the process.
Pct Mem Used	The percentage of overall memory among all processes that can be attributed to the process.
Database	The database in which the process is currently running.
Command	The command the process is currently issuing.
Open Trans	The number of open transactions for the process.
Blocked	The PID of any process blocking the current process.
Wait Time	The current amount of wait time for the process, in milliseconds.

NOTE: This information is available on both the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If your database server does not have an overabundance of memory, you should periodically check to see who your heavy memory users are along with the total percentage of memory each takes up. If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing.

I/O Page Statistics

The I/O view includes the following sections and statistics:

- Checkpoint Pages
- Forwarded Record Fetches
- Full Scans
- I/O Error Rate
- Index Searches
- Log Cache Reads
- Log Flushes
- Page Reads
- Page Writes
- Probe Scans
- Range Scans
- Read Ahead Pages
- Server I/O Busy Rate
- Session Leaders - I/O
- Worktables Created

Related Topics

I/O Detail View

Home View Statistics

Database Page Statistics

Contention Statistics

Memory Page Statistics

Network Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Users Page Statistics

Page Reads

- Metrics
- Troubleshooting

The Page Reads statistic represents that number of physical database page reads that are issued per second by SQL Server.

> **TIP:** Click this statistic to drill down to the Systems I/O tab of the I/O Detail view.

Metrics

Page reads are to be expected, especially after initial server start up. This is because SQL Server must first satisfy requests for data and meta-data by reading information in from physical disk. Numerous page reads can also be expected if the physical server does not contain an adequate amount of memory to hold repetitively requested blocks of information.

No hard-and-fast rules exist for how many page reads per second is too much. You can cross-reference this statistic with the physical server disk statistics to see if physical page reads and accompanying physical disk I/O is approaching the server's premium capacity levels. And because logical I/O is always many times faster than physical I/O, you should also evaluate the buffer cache hit ratio to determine overall memory vs. physical read efficiency.

Troubleshooting

If you find that the server is becoming overworked from a physical I/O standpoint, there are several courses of action you can take:

- Examine index usage to ensure that unnecessary table scans are not occurring.
- Check the physical database design to see if table objects have been over-normalized.
- Ensure that SQL Server is configured to use sufficient amounts of memory. Examine the min server memory (MB) and max server memory (MB) parameters to see if SQL Server is constrained on either end of the memory spectrum.
- Check for large pinned table objects that could be using excessive amounts of space in the buffer cache.
- Last, but not least, investigate the possibility of adding more RAM to the physical server.

Page Writes

- Metrics

The Page Writes statistic represents the number of physical database page writes issued by SQL Server. Page Writes take place during operations such as checkpoints, lazywriter writes, index creations, and BCP routines.

> **TIP:** Click this statistic to drill down to the Systems I/O tab of the I/O Detail view.

Metrics

Page Writes can give you an idea of overall physical write activity, however there are a number of statistics that pertain specifically to certain write activities like checkpoints, etc., that you can examine to determine the amount of physical writes caused by distinct SQL Server processes.

With respect to performance, response times experienced by SQL Server users are normally not impacted by write operations unless the writes are synchronous in nature. These are typically BCPs, database recovery operations, and index creations.

Read Ahead Pages

- Metrics

The Read Ahead Pages statistic represents the number of physical database page read in anticipation of use by SQL Server. If SQL Server senses that database pages are being read in a sequential manner, it will institute a pre-fetch mechanism that moves pages into the buffer cache before they are actually needed by a process.

> **TIP:** Click this statistic to drill down to the Systems I/O tab of the I/O Detail view.

Metrics

If data is accessed sequentially (for example, through the use of a clustered index), the read ahead mechanism of SQL Server can increase performance by needed database pages already in the buffer cache before they are actually requested.

However, because the read ahead mechanism is typically triggered by full table or index range scans, if the read ahead pages are actually required to satisfy a user's query, then performance might actually suffer. In these cases, the judicious use of indexes is a better route to take.

Log Flushes

- Metrics
- Troubleshooting

The Log Flushes statistic represents the total number of log pages for all databases written to disk by the log writer process. A log flush occurs when SQL Server writes all changes from the database's log cache out to the database's log files on disk.

> **TIP:** Click this statistic to open the Contention performance category view.

Metrics

Increasing numbers observed for log flushes should not cause concern unless the I/O subsystem of the server appears overwhelmed. In addition, to minimize I/O contention between a database and its accompanying log, it is wise to place database files and log files on separate disks.

Troubleshooting

If you have placed a very active database on the same physical file as its log, you can look into moving the log to a separate physical device by adding new log files to a new drive and subsequently removing the old log files when they are not being used.

Log Cache Reads

- Metrics

The Log Cache Reads statistic represents the reads performed per second through the log manager cache. Before ever writing transactions to disk, the log manager of SQL Server formats them in memory. This area of memory is known as the log cache and only contains log records for SQL Server 2000 and later. The log writer of SQL Server moves through the log caches when transactions are committed (as well as other events) and flushes each cache out to disk.

Metrics

None.

Checkpoint Pages

- Metrics
- Troubleshooting

The Checkpoint Pages statistic represents the number of pages flushed to disk per second by a checkpoint or other operation that require all dirty (modified) pages to be flushed.

> **TIP:** Click this statistic to open the Contention performance category view.

Metrics

Checkpoint operations are used by SQL Server to minimize the amount of work the server must perform when databases are recovered during system startup. Checkpoints periodically write out modified pages that are found in the buffer cache to disk. Afterwards, SQL Server records the operation in the log to signify that the operation succeeded.

Checkpoints can be explicitly performed by a database owner issuing the checkpoint command. SQL Server also performs checkpoints automatically for databases that have the trunc log on chkpt option set.

Large SQL Servers have the potential to generate lots of checkpoint write operations. Although SQL Server will do what it can to minimize checkpoint activity, you can also set the recovery interval server parameter to influence how often checkpoints should run.

Troubleshooting

If you believe excessive checkpoint activity is occurring, you can take the following steps:

- Set the recovery interval server parameter to a larger value with sp_configure.
- Restart SQL Server so the change will take affect or use the RECONFIGURE option to make the change immediately.

Full Scans

The Full Scans statistic represents the total number of full table or index scans per second.

Metrics

Full scans occur if a table is inadequately indexed or if SQL Server truly needs to access all rows in a table or index to satisfy a query. Other operations that can cause full scans include UPDATE STATISTICS calls.

Unnecessary scans on large tables is something to avoid, and can be a signal to you as a DBA to investigate the use of more indexes and to review SQL access through EXPLAIN plans. Small table scans are actually a good thing because SQL Server can often cache the entire table in a single I/O operation. Large numbers of index scans are normally desirable too, because it typically indicates the fastest possible resolution to data access requests.

Troubleshooting

Here are some methods you can use to avoid unnecessary large table scans:

- Try not to use SQL statements that include the NOT IN, NOT LIKE, <>, IS NULL operators because they typically suppress the use of indexes.
- When referencing concatenated indexes with queries, be sure the leading column in the index is used. If it is not, the index will not be used at all.
- Avoid using functions in WHERE predicates.

Range Scans

- Metrics

The Range Scans statistic represents the total number of qualified range scans through indexes per second.

Metrics

Large numbers of index scans are normally desirable, because it typically indicates the fastest possible resolution to data access requests is being taken.

Index Searches

- Metrics

The Index Searches statistic represents the total number of index searches per second. Index searches are normally used to start range scans, for single index record fetches and can be used to reposition an index.

> **TIP:** Click this statistic to drill down to the Systems I/O tab of the I/O Detail view.

Metrics

Large numbers of index searches are normally desirable because they typically indicate the fastest possible resolution to data access requests is being taken.

Probe Scans

- Metrics

The Probe Scans statistic represents the total number of probe scans per second. Probe scans are used in SQL Server to directly find rows in an index or base table.

> **TIP:** Click this statistic to open the Contention performance category view.

Metrics

Large numbers of probe scans are normally desirable, because they typically indicate the fastest possible resolution to data access requests is being taken.

Worktables Created

- Metrics

The Worktables Created statistic represents the total number of work tables created per second. Worktables are used many times by SQL Server to perform a logical operation specified in an end user SQL statement. GROUP BY, ORDER BY, or UNION queries can cause worktables to be created as can specific CREATE statements used in Transact SQL processing.

Worktables are built in the tempdb database and are dropped automatically at the end of the statement or procedure run.

> **TIP:** Click this statistic to open the Contention performance category view.

Metrics

The only thing to keep in mind with respect to worktables is that the tempdb database should be large enough to hold large worktables.

Forwarded Record Fetches

- Metrics
- Troubleshooting

The Forwarded Record Fetches statistic represents the total number of records per second fetched by reading forwarded record pointers.

SQL Server moves rows in a table under certain conditions. One situation might arise when you update a row in a table that has a variable-length column to a larger size that no longer fits on its original page. Another situation would be if SQL Server moves a row when the clustered index column changes.

> **TIP:** Click this statistic to open the Contention performance category view.

Metrics

When SQL Server creates a forwarding pointer, it remains in place unless one of two things happens. The first is when a row shrinks enough to move back to its original location. The second is when the entire database shrinks. When a database file shrinks, SQL Server reassigns the row identifiers, which are used as the row locators, so the shrink process never generates forwarded rows.

Forwarded records can reduce performance at times because additional I/O is involved to first obtain the record pointer to the relocated row, and then the row itself is read.

Troubleshooting

If consistent numbers are present for Forward Record Fetches, examine your databases to see which tables have forwarded records.

To see the total count of forwarded records in a table, you can enable trace flag 2509, and then execute the DBCC CHECKTABLE command. The output should display the number of forwarded records in that table. Tables with many forwarded records could be candidates for table reorganization.

Server I/O Busy Rate

- Metrics

The Server I/O Busy or I/O Busy statistic represents the time in milliseconds that the database server has spent performing input and output operations since the last refresh in Embarcadero Performance Center.

> **TIP:** Click this statistic to drill down to the Systems I/O tab of the I/O Detail view.

Metrics

None.

Session Leaders - I/O

- Metrics

Heavy I/O activity in a system can indicate that the user connections are contributing somewhat equally to the overall load. More often than not, however, one or two user connections are responsible for 75% or more of the I/O activity. Your system may be running a large batch load or other typical processes, which are perfectly okay. On the other hand, it may be a runaway process or other rogue connection, which you need to track down and possibly eliminate.

The leading I/O session's display lets you see who the leading sessions are in your system with respect to I/O.

NOTE: This statistic displays on both the Users performance category view and the I/O performance category view.

TIP: Click this statistic to drill down to the Systems I/O tab of the I/O Detail view.

TIP: Click an item in the chart to open the Overview tab of the Session Detail view.

Metrics

One or two users consuming more than 75% of the total I/O load can indicate a runaway or improper process. You can drill into the I/O activity of all users and quickly see if this is the case.

Total I/O

The Total I/O statistic represents the total number of physical reads and writes.

Metrics

None.

I/O Error Rate

- Metrics
- Troubleshooting

I/O Error Rate reflects total number of I/O errors (errors during read and write operations) encountered by the server since the last refresh inside Performance Center. I/O Error Rate is a percentage based on Total I/O (the sum the physical reads and writes).

TIP: Click this statistic to drill down to the User I/O tab of the I/O Detail view.

Metrics

You should observe few, if any errors.

Troubleshooting

If you notice any errors, you should check the SQL Server error log for details.

I/O Detail View

The I/O Detail view includes the following tabbed pages:

- Database I/O
- File I/O

- Leading Sessions
- System I/O
- User I/O

System I/O Tab

- Metrics
- Troubleshooting

SQL Server performs many system-related I/O functions to keep data moving into and out of the server. The System I/O tab of the I/O Detail view details space-related I/O operations. The table below describes the information available in the System I/O tab:

Information	Description
Pages/Extents Allocated	The number of space extents that SQL Server allocated. Rapidly increasing numbers for these statistics indicates that SQL Server is receiving large volumes of incoming data and is allocating space to make room.
Page/Extent Deallocations	This indicates that SQL Server is reclaiming space from database objects due to shrinking database volumes.
Freespace Scans	The number of scans performed by SQL Server to locate free space for an incoming record.
Page Splits	When data is inserted or updated in a table, SQL Server might reorganize the storage of the data in the table's index pages. When an index page becomes full, but a DML operation demands room on that page, SQL Server moves about half the rows to a new page to accommodate the request. This reorganization is known as a page split. Performance for DML actions can be impaired from page split operations. In addition, more index pages can make for longer index scan times.

Metrics

Increasing numbers for extent and page allocation, and freespace operations likely indicates aggressive volumes of data being inserted or modified in SQL Server.

Page splits cause additional overhead in the form of CPU usage and I/O. Observing large numbers of page splits can signal a resource bottleneck in your server.

Troubleshooting

To avoid page splits, you can look into tuning the FILLFACTOR property of an index, which controls the percentage of the index page that is filled during creation. The default, 100, tells SQL Server to completely fill each page, whereas lower numbers tell SQL Server to leave room for additional index rows or updates to existing rows.

Database I/O Tab

SQL Server performs many system-related I/O functions to keep data moving into and out of the server. The Database I/O tab of the I/O Detail view details maintenance-related I/O operations. The table below describes the information available in the Database I/O tab

Information	Description
Database	The name of the database.
DBCC Logical Scans	The number of logical read scan bytes per second caused by DBCC operations.

Information	Description
Bulk Copy Rows	The number of rows copied either into or out of the database via the BCP utility.
Bulk Copy Throughput	The amount of data (in KB) copied via BCP operations.
Backup/Restore Throughput	The read/write throughput for backup and restore operations of a database.
Log Cache Reads	The number of reads performed through the log manager cache.
Log Flushes	The number of log flushes for the server.
Backup/Restore T-Put	Defines the read/write throughput for backup and restore operations.
Log Growths	The number of times the transaction log for the database has been expanded.
Log Shrinks	The number of times the transaction log for the database has been reduced in size.
Log Truncations	The number of times the transaction log for the database has been truncated (possibly by log backup operations).

User I/O Tab

- Metrics
- Troubleshooting

The User I/O tab of the I/O Detail view displays statistics that track various user-related I/O operations. The I/O function along with its counter value is presented. The User I/O tab details performance statistics that reflect how SQL Server is performing object access operations. The table below describes the information available on the User I/O tab of the I/O Detail view:

Information	Description
Forwarded Records	The number of records per second fetched through forwarded record pointers. At times forwarded records can reduce performance because additional I/O is involved to first obtain the record pointer to the relocated row, and then the row itself is read.
Full Scans	Full scans of moderately sized indexes or tables are generally okay. SQL Server can scan and cache a small table much faster than using its index to navigate to any requested data. Full, unrestricted, large table scans, however, are typically not good and degrade overall system performance and response time.
Index Searches	The total number of index searches per second. Index searches are normally used to start range scans, for single index record fetches and can be used to reposition an index.
Probe Scans	The total number of probe scans per second. Probe scans are used in SQL Server to directly find rows in an index or base table.
Range Scans	The total number of qualified range scans through indexes per second.
Skipped Ghosted Records	The number of ghosted records per second skipped during scans.

Metrics

Full scans occur if a table is inadequately indexed or if SQL Server truly needs to access all rows in a table or index to satisfy a query. UPDATE STATISTICS calls can also cause full scans.

Unnecessary scans on large tables is something to avoid, and can be a signal to you as a DBA to investigate the use of more indexes and to review SQL access through EXPLAIN plans. Small table scans are actually a good thing because SQL Server can often cache the entire table in a single I/O operation. Large numbers of index scans are normally desirable too, because this typically indicates the fastest possible resolution to data access requests.

When SQL Server creates a forwarding pointer, it remains in place unless one of two things happens. The first is when a row shrinks enough to move back to its original location. The second is when the entire database shrinks. When a database file shrinks, SQL Server reassigns the row identifiers, which are used as the row locators, so the shrink process never generates forwarded rows. Forwarded records can reduce performance at times because additional I/O is involved to first obtain the record pointer to the relocated row, and then the row itself is read.

Large numbers of index searches and probe scans are normally desirable because they typically indicate the fastest possible resolution to data access requests is being taken.

Troubleshooting

Here are some methods you can use to avoid unnecessary large table scans:

- Try not to use SQL statements that include the NOT IN, NOT LIKE, <>, IS NULL operators because they typically suppress the use of indexes.
- When referencing concatenated indexes with queries, be sure the leading column in the index is used. If it is not, the index will not be used at all.
- Avoid using functions in WHERE predicates.

If consistent numbers are present for Forward Record Fetches, examine your databases to see which tables have forwarded records. This can easily be done with the Embarcadero Space Analyst component of DBArtisan. If you do not have Space Analyst, then to see the total count of forwarded records in a table, enable trace flag 2509, and then execute the DBCC CHECKTABLE command. The output should display the number of forwarded records in that table. Tables with many forwarded records could be candidates for table reorganization.

File I/O Tab

- Metrics

The File I/O tab of the I/O Detail view summarizes I/O activity for each database file, letting you quickly spot the “hot” databases and files on your server. The table below describes the information available on the File I/O tab of the I/O Detail view for SQL Server 2000 and later:

Information	Description
Database	The database name.
File ID	The file identifier for the target file.
Logical Name	The name given the file by the DBA.
File Name	The physical file name of the file.
Timestamp	The internal time stamp of when the data was obtained.
Reads	The number of reads issued against the database file.
Writes	The number of writes issued against the database file.
Bytes Read	The total number of bytes read for the database file.
Bytes Written	The total number of bytes written for the database file.
I/O Stall	The total amount of time that processes have waited for I/O operations to complete, in milliseconds.

NOTE: Data is only available for SQL Server 2000 and later.

Metrics

Consider moving databases and/or files with lots of I/O activity and wait time onto separate drives/devices.

Leading Sessions Tab

- Metrics

When a system undergoes heavy I/O activity, sometimes you find that all the user connections are contributing somewhat equally to the overall load. Frequently, however, one or two user connections are responsible for 75% or more of the I/O activity. It may be that a large batch load or other typical process is running that is perfectly okay for your system. Alternatively, it may be a runaway process or other rogue connection that you should track down and possibly eliminate.

Embarcadero Performance Center displays information to help you find processes that are using the most memory on the server on the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view.

The table below describes the information available on these tabs:

Column	Description
PID	The process ID.
User Name	The user name assigned to the process.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Physical I/O	The current cumulative number of reads and writes issued by the process.
Pct I/O Used	The percentage of overall I/O that can be attributed to the process.
Database	The database where the process is running.
Command	The command the process is currently issuing.
Open Trans	The number of open transactions for the process.
Blocked	The PID of any process blocking the current process.
Wait Time	The current amount of wait time for the process, in milliseconds.

NOTE: This information is available on both the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Finding one more two users that are consuming more than 75% of the total I/O load can indicate a runaway or improper process. By drilling down into the I/O activity of all users, you can quickly see if this is the case.

Space Page Statistics

The Space view includes the following statistics:

- Database Overview
- Log Overview
- Disk Free Space

Related Topics

Space Detail View

Home View Statistics

Database Page Statistics

Contention Statistics

I/O Page Statistics

Memory Page Statistics

Network Statistics

OS Page Statistics

Top SQL Page Statistics

Users Page Statistics

Disk Free Space

- Metrics
- Troubleshooting

The Disk Free Space statistic provides a summary of the total database and log space used per server disk drive, as well as a summary of free disk space for each server drive.

Metrics

If any database or transaction log file has been set up to automatically grow, the DBA should ensure there is enough server disk space to accommodate any new, additional requests for space.

Troubleshooting

If you see any drive that has reached zero free space (or is close), you might want to add new files, on other disks with abundant free space, to any databases or transaction logs so that no out-of-space errors result.

Database Overview

- Metrics
- Troubleshooting

A SQL server contains many databases, some of which are devoted to system-level activities (the master and tempdb databases, for example) and others that hold user data. The Databases Low on Space statistic represents a count of databases that have fallen below the Performance Center recommended free space limit.

> **TIP:** Click this category heading to drill down to the Fragmentation tab of the Space Detail view.

Metrics

If a database's free space goes below the Performance Center recommended threshold, (and either the database or transaction log does not have their automatic growth property enabled or the files have reached their growth limit) then the DBA should take action to ensure that the database or transaction log does not run out of available free space.

Troubleshooting

If the percent used amount of a database is approaching problematic levels, then there are three ways a DBA can rectify the situation:

1. The DBA can resize the current file(s) used by the database via an ALTER DATABASE … MODIFY FILE command.
2. The DBA can add a new file to the database via the ALTER DATABASE … ADD FILE command.
3. The DBA can modify the file(s) used by the database to automatically grow by using the ALTER DATABASE … MODIFY FILE … FILEGROWTH command. You should also ensure that the MAXSIZE setting of each file is set appropriately.

Of course, the DBA should also ensure that enough physical space exists on the server to accommodate additional database space.

Log Overview

A SQL server contains many databases, some of which are devoted to system-level activities (the master and tempdb databases, for example) and others that hold user data. The Log Overview statistic represents a count of database transaction logs that have fallen below the Performance Center recommended free space limit.

Metrics

If a database's transaction log free space goes below the Performance Center recommended threshold, (and the transaction log does not have their automatic growth property enabled or the files have reached their growth limit) then the DBA should take action to ensure that the transaction log does not run out of available free space.

Troubleshooting

There are several things a DBA can do to ensure that a database's log does not run out of available free space:

1. First, most transactional-oriented databases should have their logs assigned to a separate physical drive than the database. Reasons for doing so include:
   - It prevents competition for space between the log and the database itself.
   - It allows the log to be monitored for space more effectively.
   - It improves performance.
2. If the database is not critical in nature, you can set the truncate log on checkpoint option (trunc log on chkpt), which will eliminate any non-active space in the log when a database checkpoint occurs.
3. Critical databases needing higher levels of recovery should have schedules established that regular perform transaction log dumps. Doing so ensures better recovery scenarios as well as a reduced risk of the transaction log running out of space.
4. If a critical transaction log becomes full, it might be impossible to use standard procedures to dump transactions and reclaim space. The dump operation will likely have to incorporate the no log or truncate only options.
5. If a transaction log continuously approaches dangerously low levels of free space, then the DBA should allow the underlying file(s) of the log to automatically grow to meet the demand. This can be accomplished by using the ALTER DATABASE … MODIFY FILE … FILEGROWTH command. You should also ensure that the MAXSIZE setting for each file is set appropriately.

The DBA should also be on the lookout for large load or data modification operations that do not make use of prudently timed commit points. A single, large transaction has the ability to overwhelm any transaction log since only non-active space in the transaction log is removed from log dumps or truncation operations.

Space Detail View

The Space Detail view includes the following tabbed pages:

- Disk Space
- Indexes
- File Groups/Files
- Fragmentation
- Tables
- Virtual Log Files

File Groups/Files Tab

- Metrics
- Troubleshooting

SQL Server manages physical storage space through files and file groups. The file groups tab displays detailed information regarding storage usage in each of the SQL Server file groups.The File Groups/Files tab of the Space Detail view displays space measures for the file groups and files of a particular database. The first grid presents file group information and the second grid displays data for the individual files in a database. The table below describes the information available in the File Group Summary grid on the File Group/Files tab of the Space Detail view:

Information	Description
File Group	The name of the file group.
Files	The number of files that make up the file group.
Size (MB)	The total physical size of the file group.
Table Reserved	The amount of reserved space consumed by tables.
Index Reserved	The amount of reserved space consumed by indexes.

The table below describes the information available in the Files Summary section of the Files tabin the File Summary grid on the File Group/Files tab of the Space Detail view:

Information	Description
Logical Name	The nickname given to the file by the DBA.
File Group	The name of the file group.
File Name	The physical name of the file.
File Name	The name and location of the file.
Size (MB)	The total physical size of the file.
Growth	This indicates if the file can automatically grow in size.
Growth Amount	This indicates how much the file will grow in size.

Information	Description
Max Size	This indicates the maximum file size that the file can grow.
Total Extents	The total number of extents taken up by the file.
Used Extents	The actual number of used extents in the file.

Metrics

Unless space is tight on a server, it is normally wise practice to allow your files to automatically grow to meet demand for more incoming data. It is also smart to physically separate your database and log files onto separate physical drives.

Troubleshooting

To let your files automatically grow until out of space, you can easily do this by setting the file's growth option in Embarcadero's DBArtisan or Embarcadero's Rapid SQL, or by using the ALTER DATABASE... MODIFY FILE command.

Virtual Log Files Tab

- Metrics

The Virtual Log Files tab of the Space Detail view displays an internal structure of sorts for a each database's log. The presented information is helpful when trying to shrink a database's log because you can see how much of the log is active and exactly where the active portion resides. The table below describes the information available on the Virtual Log Files tab of the Space Detail view:

Information	Description
Database Name	The name of database.
File Name	The name of the log file.
Status	This indicates if this portion of the log is active or inactive (not being used).
Size	The size of this portion of the log in MBs.

Metrics

None.

Tables Tab

- Metrics

Tables and indexes consume the storage in all databases. The Tables tab displays summary information regarding table and index storage for all databases.

The Tables tab of the Space Detail view displays the space details for tables in a selected database. The table below describes the information available on the Tables tab of the Space Detail view:

Column	Description
Owner	The owner of the table.
Table Name	The name of the table.
File Group	The file group where the table resides.

Column	Description
Rows	The number of rows in the table.
Reserved (KB)	The amount of space reserved for the table in kilobytes.
Used (KB)	The amount of space the table is using.
Free (KB)	The amount of free space that the table possesses.
% of Database	The percentage of the database that the table consumes.

Metrics

Negative numbers viewed for index space information can be caused by inaccuracies contained in the SQL Server data dictionary. Frequently, running a DBCC UPDATEUSAGE command against the database resolves the problem. However, there are bugs in SQL Server that sometimes caused erroneous numbers to the reported for the reserved space amounts used by tables and indexes.

Indexes Tab

- Metrics

Tables and indexes consume the storage in all databases. The Database Object Detail grid displays object space details for the database selected in the Database Object Summary grid.

The Indexes tab of the Space Detail view displays the space details for indexes in a selected database. The table below describes the information available on the Indexes tab of the Space Detail view:

Information	Description
Table Name	The owner/table that is the parent of the index.
Index Name	The name of the index.
Index Type	The type of index (clustered, etc.)
File Group	The file group where the object resides.
Reserved (KB)	The amount of space reserved by the object in kilobytes.
Used (KB)	The amount of space used by the object in kilobytes.
Free (KB)	The amount of free space used by the object in kilobytes.
% of Database	The percentage of the database that the index consumes.

Metrics

Negative numbers viewed for index space information can be caused by inaccuracies contained in the SQL Server data dictionary. Frequently, running a DBCC UPDATEUSAGE command against the database resolves the problem. However, there are bugs in SQL Server that sometimes caused erroneous numbers to the reported for the reserved space amounts used by tables and indexes.

Fragmentation Tab

- Metrics
- Troubleshooting

Object fragmentation can hinder the performance of SQL Server in a couple of ways. The first type of fragmentation is internal fragmentation. In a nutshell, this means that an index is taking up more space than it really needs. Larger indexes make for longer scan times, so performance degradation can result for indexes with high amounts of internal fragmentation. External fragmentation is the second form of fragmentation, and occurs when the logical order of pages for an index do not match the actual physical order. It also presents itself when the extents belonging to a table are not contiguous. External fragmentation generally only degrades performance when SQL Server is performing an ordered scan of a table or index.

The Fragmentation tab of the Space Detail view presents fragmentation information for the tables and indexes of a selected database. The table below describes the information available on the Fragmentation tab of the Space Detail view:

Column	Description
Table Name	The name of the table.
Index Name	The name of the index.
Object Type	The type of object (table or index).
Local Fragmentation	The percentage of extent fragmentation found after scanning the leaf pages of an index. Logical fragmentation is not relevant to tables or text indexes. Lesser values indicate little fragmentation.
Scan Density	Cannot be calculated adequately if the table resides on more than one file. However, if the table is placed on only one file, the Scan Density lists a percentage of the amount of external/extent fragmentation. A value of 100 represents a perfectly contiguous object. Values of less than 100 indicate that some fragmentation exists.
Average Page Density	In general, a low value indicates little internal fragmentation. Page Density represents how full the actual pages are.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

In general, you want to see low values for logical fragmentation and page density, and high values for scan density.

Troubleshooting

If you observe threatening fragmentation levels, you can take many courses of action:

1. You can use DBCC DBREINDEX to rebuild all the tables for a selected table or selectively rebuild just the indexes you want.
2. For SQL Server 2000 and later, you can use DBCC INDEXDEFRAG to rebuild the indexes for a table online. Users can perform DML operations on the objects involved in the DBCC operation.
3. You can drop and recreate the indexes that appear fragmented.

Disk Space Tab

- Metrics
- Troubleshooting

The Disk Space tab of the Space Detail view displays the amounts of space used by SQL Server per physical drive and by database per physical drive. The first section displays summary information and the second section contains space information per database.

The table below describes the information available in the Disk Summary by Space section on the Disk Space tab of the Space Detail view:

Column	Description
Disk Drive	The physical drive letter.
SQL/Data Space (MB)	The amount of reserved database space on the drive.
SQL/Log Space (MB)	The amount of reserved log space on the drive.
Disk Free Space (MB)	The total amount of free space that remains on the drive.

The table below describes the information available in the Disk Summary By Database section on the Disk Space tab of the Space Detail view:

Column	Description
Disk Drive	The physical drive letter.
Database	The database name.
SQL/Data Space (MB)	The amount of reserved database space on the drive.
SQL/Log Space (MB)	The amount of reserved log space on the drive.

Metrics

If you allow your database and/or log files to automatically grow, and you see little or no free space left on their physical drives, an option is to add new files to the database or log on different drives to avoid any out of space errors.

It is also smart to physically separate your database and log files onto separate physical drives.

Troubleshooting

If you need to add new files to your databases or logs, you can do so easily by using the ALTER DATABASE… ADD FILE and ALTER DATABASE… ADD LOG FILE commands.

Users Page Statistics

The Users view includes the following statistics:

- Active Connections
- Active Transactions
- Auto-Param Attempts
- Failed Auto-Param Attempts
- Inactive Connections
- Session Leaders - Memory
- Session Leaders - I/O
- Session Leaders - CPU
- SQL Compilations
- SQL Re-Compilations

- System Connections
- T-SQL Batches
- Total Connections
- Users Blocked

Related Topics

Users Detail View

Home View Statistics

Database Page Statistics

Contention Statistics

I/O Page Statistics

Memory Page Statistics

Network Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Total Connections

- Metrics
- Troubleshooting

Every connection to SQL Server spawns one or more processes, each uniquely identified by what SQL Server calls a SPID. The Total Connections display shows the number of user and system processes currently reported in the SQL Server instance. This number includes both active and inactive processes.

> **TIP:** Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

You should view the total number of connections in light of the maximum number of processes allowed to connect to SQL Server. The user connections parameter specifies the maximum number of user processes that can simultaneously connect to a SQL Server instance.

Troubleshooting

If the total number of connections approaches the number of user connections limit then:

1. Edit the configuration file for SQL Server.
2. Increase the amount of user connections to a higher value.
3. Cycle SQL Server when possible to allow the new value to take effect.

Active Transactions

- Metrics

The active transactions statistic represents a count of the number of in-process transactions for SQL Server.

> **TIP:** Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

None.

T-SQLBatches

- Metrics

The T-SQL batches statistic refers to the number of transact SQL batches processed by SQL Server.

Metrics

None.

Active Connections

- Metrics

The Active Connections statistic represents the total number of active and open threads reported on the server. This number displays the number of processes actively performing work.

> **TIP:** Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

None.

Inactive Connections

- Metrics

The Inactive Connections statistic represents the total number of threads logged on to the server that are idle at the current time.

> **TIP:** Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

None.

System Connections

- Metrics

The System Connections statistic represents the total number of threads logged on to the server that are SQL Server internal processes.

> **TIP:** Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

None.

- Troubleshooting

SQL Compilations

- Metrics
- Troubleshooting

The number of SQL compilations performed per second, indicating the number of times the compile code path is entered. This also includes compiles due to recompiles. When SQL Server user activity levels become stable, this value reaches a steady state.

Metrics

Because compilation is a significant part of a query's turnaround time, you should strive to have as many compilations stored in the cache as possible. If this number does not stabilize in direct proportion to user activity stabilizing, you should investigate your SQL Cache to see if it has adequate memory assigned to it.

Troubleshooting

If you discover that the SQL Cache area is undersized, first ensure SQL Server is configured to use as much physical memory as possible by checking the Max Server Memory configuration option. Also, consider increasing your SQL Server Min Memory parameter to allocate more memory to SQL Server. (Note that to obtain optimal values for these parameters, one option is to install more physical RAM to your server.) Check for any large objects that are pinned in memory that could possibly be removed.

SQL Re-Compilations

- Metrics
- Troubleshooting

The SQL re-compilations statistic represents the total number of recompiles triggered per second in a SQL Server instance. Recompiles occur when SQL Server determines that the currently defined execution plan for an executing stored procedure might no longer be the best possible plan. SQL Server pauses the query execution and recompiles the stored procedure.

Metrics

Recompiles slow down the process that is executing the procedure and increase the load on the CPU. By extension, the more recompiles that are occurring on your system, the more overall load increases, resulting in poor performance. In general, you want to keep the number of recompiles low. The most common reasons SQL Server would issue a recompile are: Running sp_recompile against any table referenced in the stored procedure, significant data changes in a referenced table, schema changes to referenced objects, the use of the WITH RECOMPILE clause in the CREATE PROCEDURE or EXECUTE statement, and a plan no longer available in the system cache.

Troubleshooting

Try to practice coding standards that eliminate the most frequent causes detailed above. Also, try to:

- Use temporary tables only in the stored procedure that created them.
- Minimize creating temporary tables in control block structures.
- Use the KEEP PLAN option on references to static temporary tables.
- Issue the CREATE TABLE statement before any other references to the created table.
- Minimize the use of temporary tables.

Auto-Param Attempts

Auto-parameterization occurs when an instance of SQL Server attempts to reuse a cached plan for a previously executed query that is similar to, but not the same as, the current query. This statistic shows the number of auto-parameterization attempts per second and includes failed, safe, and unsafe auto-parameterizations.

Metrics

SQL Server's ability to match new SQL statements with existing, unused execution plans is increased when parameters or parameter markers are used in Transact-SQL statements. If an SQL statement is executed without parameters, SQL Server parameterizes the statement internally to increase the possibility of matching it against an existing execution plan. A high number for this statistic shows that SQL Server is efficiently reusing existing cached plans.

Troubleshooting

You can increase the ability of the relational engine to match complex SQL statements to existing, unused execution plans, by explicitly specify the parameters using either sp_executesql or parameter markers in your T-SQL code.

Failed Auto-Param Attempts

Auto-parameterization occurs when an instance of SQL Server attempts to reuse a cached plan for a previously executed query that is similar to, but not the same as, the current query. The Failed Auto-Param Attempts statistic shows the number of failed auto-parameterization attempts per second.

Metrics

SQL Server's ability to match new SQL statements with existing, unused execution plans increases when parameters or parameter markers are used in Transact-SQL statements. If an SQL statement is executed without parameters, SQL Server parameterizes the statement internally to increase the possibility of matching it against an existing execution plan. A small number for this statistic shows that SQL Server is efficiently reusing existing cached plans.

Troubleshooting

You can increase the ability of the relational engine to match complex SQL statements to existing, unused execution plans, by explicitly specify the parameters using either sp_executesql or parameter markers in your T-SQL code. Doing so helps lower this number.

Users Blocked

- Metrics
- Troubleshooting

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches even on large systems. Blocks are most often caused by user processes holding exclusive locks and not releasing them via a proper COMMIT frequency. Unless a process times out via an application timeout mechanism, or the process has specified a timeout period via the SET LOCK_TIMEOUT command, a process waiting for a lock will wait indefinitely.

TIP: Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

You should immediately investigate any indicator above zero, before the situation has a chance to mushroom. If this number is consistently greater than zero, investigate by using the Blocking Locks tab of the Locks view.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects they were accessing. Other user processes then almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Center, but preventing the blocking lock situation in the first place is where it gets tricky. The DBA can drill down into user detail and view all current blocking locks to see exactly which sessions are holding the currently restrictive locks.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process will wait for a lock before timing out.

Session Leaders - Memory

- Metrics
- Troubleshooting

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Frequently, user connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system levels.

The leading memory session's display identifies the top users in the server with respect to memory consumption.

NOTE: This statistic displays on both the Memory performance category view and the Users performance category view.

Metrics

If your SQL Server machine does not have an overabundance of memory, you should periodically check to see who your heavy memory users are along with the total percentage of memory each takes up. If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing.

Troubleshooting

You can use the Session Leaders - Memory statistic to find the users with the greatest current allocations of overall memory. Runaway processes can be immediately terminated from within the Embarcadero Performance Center Client.

Session Leaders - I/O

- Metrics

When a system undergoes heavy I/O activity, sometimes you finds that all the user connections are contributing somewhat equally to the overall load. More often than not, however, one or two user connections are responsible for 75% or more of the I/O activity. This may be because It may be that a large batch load or other typical process is running that is perfectly okay for your system. Or it may be a runaway process or other rogue connection that needs to be tracked down and possibly eliminated.

The leading I/O session's display allows you to see who runs the leading sessions in your system with respect to I/O.

NOTE: This statistic displays on both the I/O performance category view and the Users performance category view.

TIP: Click this statistic to drill down to the Leading Sessions tab of the I/O Detail view.

Metrics

Finding one more two users that are consuming more than 75% of the total I/O load can indicate a runaway or improper process. By drilling down into the I/O activity of all users, you can quickly see if this is the case.

Session Leaders - CPU

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Often, user connections can get out of hand with CPU use, and extreme cases have caused headaches at both the database and operating system levels.

The leading CPU session's display allows you to see who runs the leading sessions are in your system with respect to CPU usage.

Metrics

Finding one or two users who use the majority of the CPU can indicate runaway or improper processes. By drilling down into the CPU activity of all users, you can quickly see if this is the case.

Users Detail View

The Users view Detail includes the following tabbed pages:

- All Locks
- Blocking Locks
- Processes

Processes Tab

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Often, user connections can get out of hand with memory consumption, CPU, or physical I/O, and extreme cases have caused headaches at both the database and operating system levels.

The Processes tab of the Users Detail view displays information to help pinpoint problem processes. The table below describes the information available on the Processes tab of the Users Detail view:

Column	Description
SPID	The unique SQL Server process ID of the process holding the lock.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Login	The SQL Server login name of the process.
NT User	If using Windows Authentication, this is the name of the Windows NT user for this process. Also appears as the Windows NT user name if using a trusted connection.
Host	The machine name that originated the process.
Program	The name of the application program that initiated the process.
Memory	Number of pages in the procedure cache (SQL Cache) that are currently allocated to this process. A negative number indicates that pages are being released (freed) from this process.
CPU	Cumulative CPU time for the process. The CPU usage is updated regardless of the value of the SET STATISTICS TIME ON option.
Physical I/O	The current cumulative number of reads and writes issued by the process.
Blocked	If the process is being blocked, this value contains the SPID of the blocking process.
Database	The database where the process is running.
Command	The command the process is currently issuing.
Last Batch	For a client process, this value represents the last time a remote stored procedure call or an EXECUTE statement was executed. For system processes, it represents the time at which SQL Server was started.
Login Time	For client process, this value represents the last time a client logged into the SQL Server instance. For system processes, it represents the time at which SQL Server was started.
Wait Time	Represents the number of milliseconds that the process has been waiting to be serviced. If the process is not waiting, a zero (0) is displayed.
Wait Type	The last or current SQL Server wait type.
Open Xacts	The current number of open transactions owned by the process.
NT Domain	If using Windows Authentication or a trusted connection, this value contains the name of the Windows Domain of the user who owns the process.
Net Address	The unique identifier of the network card in the client machine that owns the process.

Metrics

The key to spotting problem processes is to view their information in light of the total activity on a server. For example, if your database server does not have an overabundance of memory, you should periodically check to see who your heavy memory users are. You should also check the total percentage of memory each takes up. If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing and take action if necessary.

All Locks Tab

- Metrics

To modify database information or structures, a user session must obtain a lock on the object to perform its task. In addition to user locks, SQL Server itself issues lock requests to carry out its internal duties. The All Locks tab gives information the locks currently on the system and also indicates if any blocking situations are occurring. The table below describes the information available on the All Locks tab of the Users Detail view:

Information	Description
SPID	The process id of the process holding the lock.
Login	The login name of the process.
NT User	The operating system name of the process.
Database	The database in which the process is running.
Table Name	The name of the table involved in a lock. This will be NULL for non-table locks or table locks that take place in the tempdb database.
Ndx ID	The index ID involved in the lock.
Lock Type	The type of the lock (database, table, row id, etc.)
Lock Mode	The lock's mode (shared, exclusive, etc.)
Lock Status	The lock's status (waiting or granted).
Lock Owner Type	Whether the lock came from a regular session or a transaction.
User Program	The executable the process is using against the server.
BLK SPID	If zero, the process is not being blocked. If non-zero, this column represents the process ID of the process blocking the requested lock.
Wait Time	The current amount of wait time for the process, in milliseconds.
SPID Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
SPID Command	The command the process is currently issuing.
NT Domain	The name of Windows 2000/NT domain.

Metrics

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the stranglehold on the objects the user was accessing. Other user processes then almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process waits for a lock before timing out.

Blocking Locks Tab

- Metrics
- Troubleshooting

Metrics

Without a doubt, blocking lock situations can give the appearance of a frozen database almost more than anything else can. A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems. Although SQL Server supports row-level locking, blocking lock situations do crop up - sometimes frequently.

The Blocking Locks tab contains information relating to user accounts that are currently blocked and the sessions that are blocking them. The table below describes the information available on the Blocking Locks tab of the Users Detail view:

Column	Description
SPID	The process id of the process holding the lock.
Login	The login name of the process.
NT User	The operating system name of the process.
BLK SPID	If zero, the process is not being blocked. If nonzero, this column represents the process ID of the process blocking the requested lock.
Database	The database in which the process is running.
Table Name	The name of the table involved in a lock. This will be NULL for non-table locks or table locks that take place in the tempdb database.
Ndx ID	The index ID involved in the lock.
Lock Type	The type of the lock (database, table, row id, etc.)
Lock Mode	The lock's mode (shared, exclusive, etc.)
Lock Status	The lock's status (waiting or granted).
Owner Type	Whether the lock came from a regular session or a transaction.
Program	The executable the process is using against the server.
Wait Time	The current amount of wait time for the process, in milliseconds.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Command	The command the process is currently issuing.
NT Domain	The name of Windows 2000/NT domain.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects the user was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process waits for a lock before timing out.

Contention Page Statistics

The Contention performance category view displays the following vital SQL Server contention statistics:

Blocked Users

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems. Although SQL Server supports flexible locking mechanisms, blocking lock situations do crop up. User processes holding exclusive locks and not releasing them via a proper COMMIT frequency most often cause blocks.

The Blocked Users statistic displays the number of current process blocked by other processes. Unless a process times out via an application timeout mechanism, or the process has specified a timeout period via the SET LOCK_TIMEOUT command, a process waiting for a lock waits indefinitely.

> **TIP:** Click this statistic to drill down to the Blocking Lock tab of the Locks view.

Metrics

Consistently seeing positive numbers for the blocked statistic should also clue you into the fact that a bottleneck exists for some processes. You can easily drill down and discover the exact process(es) holding locks that are blocking out other user activity.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects he or she was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process waits for a lock before timing out.

Page Deadlocks

- Troubleshooting

The Page Deadlocks statistic represents the number of deadlocks per second detected by SQL Server. Page Deadlocks occur when processes cannot proceed because they are waiting on a set of resources held by each other or held by other processes.

Metrics

Consistently seeing page deadlock counts greater than zero indicates that some user processes are experiencing delays completing their work. When SQL Server identifies a page deadlock, it resolves the situation by choosing the process that can break the deadlock. This process is termed the deadlock victim. SQL Server rolls back the deadlock victim's transaction, and then notifies the process' application by returning an error message. It also cancels the process' request and allows the transactions of the remaining processes to continue.

SQL Server always attempts to choose the least expensive thread running the transaction as the deadlock victim.

Troubleshooting

Because SQL Server automatically resolves deadlock situations, you should work proactively to prevent them in the first place.

The culprit of most blocking lock and deadlock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

You can change default deadlock behavior by using the SET DEADLOCK_PRIORITY command, which reprioritizes a process' position in a deadlock situation.

Number of Deadlocks

The number of deadlocks statistic is the measure of the lock requests per second that resulted in a deadlock.

The culprit of most blocking lock and deadlock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

You can change default deadlock behavior by using the SET DEADLOCK_PRIORITY command, which reprioritizes a process' position in a deadlock situation.

Latch Waits

- Metrics

The Latch Waits statistic represents the number of latches per second that could not be satisfied immediately by SQL Server. Latches can be thought of as lightweight, mini-locks that are used to protect actions and resources used inside transactions.

> **TIP:** Click this statistic to drill down to the Waits tab of the Contention Detail view.

Metrics

Unless accompanied by long wait times, latch waits should not pose too much of a performance problem in normal SQL Server installations.

Log Flush Waits

- Metrics
- Troubleshooting

The Log Flush Waits statistic represents the number of commits per second waiting for a log flush. SQL Server records every modification to data in a database's log cache. After a transaction is committed, SQL Server writes everything from the database's log cache to the physical log files. Before a process invoking the COMMIT can continue work, it must wait until a log flush finishes. Reduced performance results when a process must wait for a log flush to complete.

> **TIP:** Click this statistic to drill down to the Waits tab of the Contention Detail view.

Metrics

Log flush waits with no measurable wait time are not considered worrisome. Waits accompanied by considerable wait times are an indicator of contention that degrades overall performance.

Troubleshooting

Drilling down into the log flush detail quickly identifies databases that are experiencing log flush waits. Placing these logs on faster physical devices (and separating them from their parent database) should reduce wait activity.

Table Lock Escalations

- Metrics

The Table Lock Escalations statistic represents the number of times locks on a table were escalated.

> **TIP:** Click this statistic to drill down to the All Lock tab of the Locks view.

Metrics

Many table lock escalations could indicate contention problems. If increasing numbers of table lock escalations are viewed at the same time as blocking or deadlock problems, the application design could be at fault.

Lock Overview

- Metrics
- Troubleshooting

The Lock Overview tab displays the total number of acquired locks and blocking locks being experienced per database.

Metrics

Consistently seeing positive numbers for the blocked statistic should also clue you into the fact that a bottleneck exists for some processes. You can easily drill down and discover the exact process(es) holding locks that are blocking out other user activity.

Another situation to look for with respect to locking is when the total number of acquired locks reaches the maximum lock limit currently set on SQL Server.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects the user was accessing. Other user processes then almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process waits for a lock before timing out.

If the total amount of acquired/requested locks has exceeded the maximum allowed on SQL Server, you should increase the configuration parameter locks.

CPU Busy

- Metrics

The CPU Busy statistic represents the time in milliseconds that the CPU has spent working since the last refresh in Embarcadero Performance Center.

Metrics

None.

I/O Busy

- Metrics

The Server I/O Busy statistic represents the time in milliseconds that SQL Server has spent performing input and output operations since the last refresh in Embarcadero Performance Center.

Metrics

None.

Server Idle

- Metrics

The Server Idle statistic represents the time in milliseconds that SQL Server has spent idle since the last refresh in Embarcadero Performance Center.

Metrics

None.

Contention Detail View

The following tabbed pages are available on the Contention Detail view:

- Blocking Locks
- Locks
- Waits

Locks Tab

- Metrics

To modify database information or structures, a user session must obtain a lock on the object to perform its task. In addition to user locks, SQL Server itself issues lock requests to carry out its internal duties. The Locks tab of the Contention Detail view displays two grids that contain information about all locks currently on the system. The first grid displays an overview of lock activity by database. The second grid displays a detailed view of lock activity.

The table below describes the information available in the Lock Summary grid on the Locks tab of the Contention Detail view:

Column	Description
Database	The database containing the locks.
Lock Type	The type of lock acquired.
Lock Count	The number of locks for the database/lock type combination.

The table below describes the information available in the Lock Detail grid on the Locks tab of the Contention Detail view:

Column	Description
SPID	The process id of the process holding the lock.
Login	The login name of the process.
NT User	The operating system name of the process.
Database	The database where the locks are occurring.
Table Name	The name of the table involved in a lock. This will be NULL for non-table locks or table locks that take place in the tempdb database.
Ndx ID	The index ID involved in the lock.
Lock Type	The type of lock (database, table, row id, etc.)
Lock Mode	The mode of the lock (shared, exclusive, etc.)
Lock Status	The status of the lock (waiting or granted).
Owner Type	Whether the lock came from a regular session or a transaction.
Program	The executable the process is using against the server.
BLK SPID	If nonzero, the process ID of the process blocking the requested lock. If zero, the process is not being blocked.
Wait Time	The amount of time the process has waited for the lock, in milliseconds.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Command	The command the process is currently issuing.
NT Domain	The name of the Windows 2000/NT domain.

NOTE: The information in the Lock Detail grid is available in the Lock Detail grid on the Locks tab and the All Locks tab of the Locks view.

TIP: To configure a grid to display row numbers, use the Options Editor.

Metrics

Locks that are held for unusually long periods could be candidates for further investigation. The application logic could be inefficient or perhaps the program is not issuing frequent enough COMMITs.

Blocking Locks Tab

- Metrics
- Troubleshooting

Without a doubt, blocking lock situations can give the appearance of a frozen database almost more than anything else. A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches even on large systems. Although SQL Server supports row-level locking, blocking lock situations do crop up - sometimes frequently.

Metrics

The Blocking Locks tab of the Contention Detail view displays two grids that contain information about all blocking lock situations on the system. The first grid displays the database name and count of all blocking locks inside the database. The second grid contains information relating to user accounts that are currently blocked and the sessions that are blocking them.

The table below describes the information available in the Blocking Lock Summary grid on the Blocking Locks tab of the Contention Detail view:

Column	Description
Database	The database containing the locks.
Locks Blocked	The number of blocked locks.

The table below describes the information available in the Blocking Lock Detail grid on the Blocking Locks tab of the Contention Detail view:

Column	Description
SPID	The process ID of the process holding the lock.
Login	The login name of the process.
NT User	The operating system name of the process.
BLK SPID	If nonzero, the process ID of the process blocking the requested lock. If zero, the process is not being blocked.
Database	The database where the locks are occurring.
Table Name	The name of the table involved in a lock. This will be NULL for non-table locks or table locks that take place in the tempdb database.
Ndx ID	The index ID involved in the lock.
Lock Type	The type of lock (database, table, row id, etc.)
Lock Mode	The mode of the lock (shared, exclusive, etc.)
Lock Status	The status of the lock (waiting or granted).
Owner Type	Whether the lock came from a regular session or a transaction.
Program	The executable the process is using against the server.
Wait Time	The amount of time the process has waited for the lock, in milliseconds.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Command	The command the process is currently issuing.
NT Domain	The name of the Windows 2000/NT domain.

NOTE: The information in the Blocking Lock Detail grid is available in the Blocking Lock Detail grid on the Locks tab and the Blocking Locks tab of the Locks view.

TIP: To configure a grid to display row numbers, use the Options Editor.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects the user was accessing. Other user processes then almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process waits for a lock before timing out.

Waits Tab

- Metrics

Understanding where SQL Server is delayed in its processing can yield big benefits in increasing overall response time. The Waits tab of the Contention Detail view contains two grids that display wait information. The first grid is a summary of various wait type categories, along with their overall wait counts and wait times. The second grid displays individual wait statistics, along with their individual wait counts and wait times.

> **TIP:** To configure a grid to display row numbers, use the Options Editor.

Metrics

Waits could not be causing actual delays in SQL Server processing unless they are accompanied by substantial wait times. The ones displaying high wait times are the ones likely to be causing transactional and system slowdowns.

Database Page Statistics

The Database view includes the following statistics:

- Database Summary
- Databases
- Databases Needing Backup
- Databases Without Auto-Create Statistics
- Databases Without Auto-Growth
- Databases Without Auto-Update Statistics
- Errors in Current Error Log
- Files
- File Groups
- Logs Without Auto-Growth
- Offline Databases
- Suspect Databases

Related Topics

Database Detail View

Home View Statistics

Contention Statistics

I/O Page Statistics

Memory Page Statistics

Network Statistics

OS Page Statistics

Space Page Statistics

Top SQL Page Statistics

Users Page Statistics

Errors in Current Error Log

- Metrics
- Troubleshooting

SQL Server records various system events in its system or error log. The majority of messages placed into the log are informational in nature, however since some contain critical messages, you should immediately review them. This section indicates the number of actual error messages in the most recent error log so you know if there are potential events that require your attention.

> **TIP:** Click this statistic to open the I/O performance category view.

Metrics

You should investigate any positive values for errors.

Troubleshooting

If non-zero values are found for this statistic, you should drill down into the current error log and examine the detail found that accompanies each error issued by SQL Server.

Databases

- Metrics

The Database statistic represents total number of databases defined on a SQL Server.

> **TIP:** Click this statistic to drill down to the SQL Cache tab of the Memory Detail view.

Metrics

None.

Files

- Metrics

The Files statistic represents total number of files (both database and log) defined on a SQL Server.

> **TIP:** Click this statistic to open the Memory performance category view.

Metrics

None.

File Groups

- Metrics

The File Groups statistic represents total number of file groups (that contain both database and log files) defined on a SQL Server.

> **TIP:** Click this statistic to open the Memory performance category view.

Metrics

None.

Suspect Databases

- Metrics
- Troubleshooting

The Suspect Databases statistic represents the number of databases SQL Server has marked as suspect. Databases are marked suspect by SQL Server if they fail during automatic recovery, which is performed at server startup. If serious damage is experienced by a database during regular uptime, SQL server will also mark a database as suspect.

Metrics

You should not find any suspect databases on any production server. You should immediately investigate any non-zero numbers, for this statistic.

Troubleshooting

The steps to handling a suspect database will vary from one installation to another. However, here are some general guidelines you can use to troubleshoot a suspect database:

- Begin by examining the SQL Server error log for clues as to what caused the database to be marked as suspect.
- It is not unusual for a server to run out of physical disk space on drives used by SQL Server. When this happens, recovery for databases can sometimes fail with the end result being SQL Server marking a database as suspect. To remedy this situation, you should free up space on the identified drives or add files to the newly marked suspect database. For SQL Server 2000, this can be accomplished by utilizing the following stored procedures: sp_add_data_file_recover_suspect_db and sp_add_log_file_recover_suspect_db. For version 7.0 of SQL Server, you will need to use the sp_resetstatus stored procedure to reset the suspect status flag for the database in question, use the alter database command to add new datafiles to the database, and then stop/start the SQL Server.
- Many times, suspect databases are caused by SQL Server not being able to access a database or log file. This happens if a particular physical hard drive has become unavailable, but also can occur if another operating system process has obtained exclusive access to a file. If this scenario proves to be true, once you have ensured that the file(s) are available once again to the operating system, you can use the sp_resetstatus stored procedure to reset the suspect status flag for the database and then stop/start the SQL Server.

If none of these solutions are possible, you will likely have to restore your database using the last full and transaction log backups.

Offline Databases

The Offline Databases statistic represents the number of databases SQL Server has offline, meaning that no database modifications can occur.

> **TIP:** Click this statistic to open the I/O performance category view.

Metrics

You should not find any offline databases on any production server. You should immediately investigate any non-zero numbers, for this statistic.

Troubleshooting

Should an offline database be found by Performance Center, you can easily place it back online by utilizing either the sp_dboption stored procedure or the alter database command.

Databases Needing Backup

The Databases Needing Backup statistic represents the number of databases in SQL Server that have not been backed up for more than seven days. This statistic excludes the pubs, tempdb, Northwind, msdb, and model databases.

> **TIP:** Click this statistic to open the I/O performance category view.

Metrics

To ensure proper protection for most databases, it is recommended that even the most static databases be backed up at least once a week. You should frequently backup critical, dynamic databases and include transaction log backups to enable point-in-time recovery ability.

Troubleshooting

Any critical databases found with obsolete or no backups should immediately be backed up. Moreover, to ensure proper data protection for each database, you should institute a planned backup schedule. The timing and repetition of the backups depend on the critical recovery needs of each database.

Databases Without Auto-Growth

Beginning in SQL Server 7.0, a DBA was given the ability to tell SQL Server to automatically grow a database in size when more space was required. This feature can save a critical transaction or other database request from failing due to a lack of free space in the database. It is recommended that critical databases have this feature enabled.

The Databases Without Auto-Growth statistic provides a count of databases that do not have their automatic growth property enabled.

TIP: Click this statistic to drill down to the Blocking Lock tab of the Locks view.

Metrics

Static databases (those not expected to grow in size) will likely not need their auto-growth property enabled. Growing, dynamic databases should almost always be allowed to automatically grow when needed.

Troubleshooting

If any critical, dynamic database is found to not have its auto-growth feature enabled, then the DBA can modify the file(s) used by the database to automatically grow by using the ALTER DATABASE … MODIFY FILE … FILEGROWTH command. You should also ensure that the MAXSIZE setting of each file is set appropriately.

Logs Without Auto-Growth

Beginning in SQL Server 7.0, a DBA was given the ability to tell SQL Server to automatically grow a database or transaction log in size when more space was required. This feature can save a critical transaction or other database request from failing due to a lack of free space in the database or transaction log. It is recommended that critical databases and their transaction logs have this feature enabled.

The Logs Without Auto-Growth statistic provides a count of transaction logs that do not have their automatic growth property enabled.

TIP: Click this statistic to open the Contention performance category view.

Metrics

Static databases (those not expected to grow in size) will likely not need their transaction log's auto-growth property enabled. Growing, dynamic databases should almost always have their transaction log be set to automatically grow when needed.

Troubleshooting

If any critical, dynamic database is found to not have their transaction log auto-growth feature enabled, then the DBA can modify the file(s) used by the database's transaction log to automatically grow by using the ALTER DATABASE … MODIFY FILE … FILEGROWTH command. You should also ensure that the MAXSIZE setting for each file is set appropriately.

Databases Without Auto-Update Statistics

The Databases Without Auto-Update Statistics statistic represents the total number of databases defined on SQL Server that do not have the AUTO_UPDATE_STATISTICS option enabled.

TIP: Click this statistic to open the Contention performance category view.

Metrics

When the AUTO_UPDATE_STATISTICS option is enabled, SQL Server automatically updates existing statistics when the statistics become out-of-date because data in the tables has changed enough to affect the optimizer's decision-making process.

Troubleshooting

If possible, a DBA should keep databases in AUTO_UPDATE_STATISTICS mode. If a database is found without this option set, you can easily change its auto-update statistics to true by using the command:

```
EXEC sp_dboption '(database name)',
```

Databases Without Auto-Create Stats

The Databases Without Auto-Create Stats statistic represents the total number of databases defined on SQL Server that do not have the AUTO_CREATE_STATISTICS option enabled.

> **TIP:** Click this statistic to open the Contention performance category view.

Metrics

When the AUTO_CREATE_STATISTICS option is enabled, statistics are automatically created on columns used in a SQL query predicate. Keeping object statistics fresh in the SQL Server data dictionary improves query performance because the optimizer can better determine how to evaluate a query and return the requested data. If the statistics are not used, SQL Server should automatically delete them.

Troubleshooting

If possible, a DBA should keep their databases in AUTO_CREATE_STATISTICS mode. If a database is found without this option set, you can easily change it by using the command:

```
EXEC sp_dboption '<database name>','auto create statistics',true
```

Database Summary

The Database Summary gives a bird's eye view of situations that could require a DBA's attention.

> **TIP:** Click the table heading to drill down to the Overview tab of the Database Detail view.

Metrics

You should investigate any blocking lock situations and databases that are using an inappropriate amount of memory, CPU, or I/O. In addition, you should reserve off-hours submissions for various utilities such as DBCCs, BCPs, or backup and restore operations.

Troubleshooting

If a blocking lock situation is observed, you should drill down and find the blocking locks that exist using Embarcadero Performance Center's Lock/Blocking Locks view. Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects the user was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds a process waits for a lock before timing out.

Database Detail View

The Database view Detail includes the following tabbed pages:

- Backup History
- Index Details
- Options
- Overview
- Replication
- SQL Agent
- Table Details

Overview Tab

- Metrics
- Troubleshooting

The Overview tab of the Database Detail view displays summarized metadata for all defined databases on SQL Server. The table below describes the information available on the Overview tab of the Database Detail view:

Information	Description
Database	The name of the database.
Created	The date/time when the database was created.
Status	The status of the database (online, offline, suspect, etc.)
DB Size (MB)	The total size of the database in MB.
Log Size (MB)	The total size of the log in MB.
Tables	The number of all tables in the database.
Indexes	The number of all indexes in the database.
Users	The number of all defined user accounts in the database.
Last Full Backup	The date/time of the last full backup for the database.

Metrics

The following are things to take note of:

- Any critical database that shows a last full backup date that is older than the database's required backup needs.
- Any database that shows a status of offline or suspect.
- Any growing database that does not have its database or log files set to automatically grow in size.
- Any dynamic database that does not have its object statistics set to automatically update.

Troubleshooting

Depending on the situation, you should take the following actions:

For databases that require a full backup, perform a full backup of the database when appropriate.

For suspect databases, the steps to handling will vary from one installation to another. However, here are some general guidelines you can use to troubleshoot a suspect database:

- Begin by examining the SQL Server error log for clues as to what caused the database to be marked as suspect.
- It is not unusual for a server to run out of physical disk space on drives used by SQL Server. When this happens, recovery for databases can sometimes fail with the end result being SQL Server marking a database as suspect. To remedy this situation, you should free up space on the identified drives or add files to the newly marked suspect database. For SQL Server 2000, this can be accomplished by utilizing the following stored procedures: sp_add_data_file_recover_suspect_db and sp_add_log_file_recover_suspect_db. For SQL Server version 7.0, you will need to use the sp_resetstatus stored procedure to reset the suspect status flag for the database in question, use the alter database command to add new datafiles to the database, and then stop/start SQL Server.
- Many times, suspect databases are caused by SQL Server not being able to access a database or log file. This happens if a particular physical hard drive has become unavailable, but also can occur if another operating system process has obtained exclusive access to a file. If this scenario proves to be true, once you have ensured that the file(s) are available once again to the operating system, you can use the sp_resetstatus stored procedure to reset the suspect status flag for the database and then stop/start SQL Server.

Should an offline database be found by Performance Center, you can easily place it back online by utilizing either the sp_dboption stored procedure or the alter database command.

If any critical, dynamic database or log is found to not have their auto-growth feature enabled, then the DBA can modify the file(s) used by the database to automatically grow by using the ALTER DATABASE … MODIFY FILE … FILEGROWTH command. You should also ensure that the MAXSIZE setting for each file is set appropriately.

If possible, a DBA should keep their databases in AUTO_CREATE_STATISTICS and AUTO_UPDATE_STATISTICS mode. If a database is found without this option set, you can easily change it by using the following commands:

```
EXEC sp_dboption '<database name>','auto create statistics',true

EXEC sp_dboption '<database name>','auto update statistics',true
```

SQL Agent Tab

SQL Server provides the ability to submit and run jobs as well as be notified about certain SQL Server-related events. The SQL Agent tab of the Database Detail view displays information regarding the status of the server's SQL Agent as well as details for all jobs and alerts that are defined to the system.

SQL Server Job Summary

The SQL Server Job Summary section graphically displays the outcome of all jobs for the last 25 job outcomes.

Metrics

You should investigate any counts noted for failed jobs.

SQL Server Alert Summary

The SQL Server alert summary section graphically displays the number of times that performance and event alerts have fired.

Backups History Tab

The Backups tab of the Database Detail includes the following:

- Backup Detail

Backup Detail

- Metrics

The Backup History tab of the Database Detail view displays the most recent 25 backups for a selected database. The table below describes the information available on the Backup History tab of the Database Detail view:

Information	Description
Database	The name of the database.
Backup Start	The time stamp when the backup began.
Backup Finish	The time stamp when the backup finished.
Backup Type	The type of backup (FULL, INCREMENTAL, etc.)
Backup Size (KB)	The size of the backup, in kilobytes.
Expiration Date	The expiration date for the backup, if any.

Metrics

Although the needs of an application determine the frequency and type of backup, it is generally recommended that most dynamic databases have a solid plan in place for full and differential backups. For databases requiring point-in-time recovery, a backup plan should also include log backups.

Replication Tab

The Replication tab of the Database Detail view provides basic information regarding replication activities that are occurring on the monitored SQL Server. The replication tab includes the following sections:

- Replication Agent Details
- Replication Throughput Details

The first grid displays information about replication agent activities. The second grid delivers information regarding replication throughput for the server.

The table below describes the information available in the Agent Information grid on the Replication tab of the Database Detail view

Information	Description
Agent	The agent performing the replication work.
Publication	The object being published to a subscribing database.
Publisher	The SQL Server providing the source material.
Publisher DB	The database of the SQL Server providing the source material.
Status	The current status of the replication agent's task.
Subscriber	The SQL Server requesting the source material.
Subscriber DB	The database of the SQL Server requesting the source material.
Type	The type of replication activity (push, pull, etc.)
Start Time	The start time for the replication task.
Duration	The duration of the replication task.
Last Action	The output message for the replication task.

The table below describes the information available in the SQL Cache Details grid on the Replication tab of the Database Detail view:

Information	Description
Owner	The owner of the table.
Table Name	The name of the table.
Pinned	This indicates if the table is pinned in memory.
File Group	The filegroup name that contains the table.
Rows	The number of rows in the table.
Clustered	This indicates if the table has a clustered index.
Indexes	This provides a count of the number of indexes for the table.
FK Constraints	This indicates if the table has any defined foreign keys.
Chk Constraints	This indicates if the table has any defined check constraints.
Has Identity	This indicates if the table has an identity column.

NOTE: For space-related table information, use the Tables tab of the Space Detail view.

Index Details Tab

- Metrics
- Troubleshooting

The Index Details tab of the Database Detail view provides an overview of the attributes of every user index defined in the database. The table below describes the information available on the Index Details tab of the Database Detail view:

Information	Description
Table Name	The name of the table.
Index Name	The name of the index.
File Group	The filegroup name that contains the index.
Clustered	This indicates if the index is clustered.
Unique	This indicates if the index is unique.
Depth	This indicates the depth/level of the index.
Auto-Stats	This indicates if statistics are automatically recomputed for the index.
Max Nonleaf Index Row Sizes	This indicates the longest size of a non-leaf index row.
Max Length	This indicates the maximum size of a row.

NOTE: For space-related index information, use the Indexes tab of the Space Detail view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Seeing an index with a depth of more than three could indicate the index has become fragmented and is need of a rebuild. Indexes built on dynamic objects should have their statistics automatically collected by SQL Server to ensure the optimizer has the most up-to-date information to work with when making access path decisions.

Troubleshooting

Embarcadero Performance Center offers several options to rebuild an index:

1. You can use DBCC DBREINDEX to rebuild all the indexes for a selected table or selectively rebuild just the indexes you want.
2. For SQL Server 2000 and later, you can use DBCC INDEXDEFRAG to rebuild the indexes for a table online. Users can perform DML operations on the objects involved in the DBCC operation.
3. You can drop and re-create the indexes that appear fragmented.

For indexes that are not having statistics automatically gathered by SQL Server, you can drop and recreate the index with the option of having statistics recomputed by the server.

Table Details Tab

- Metrics

The Table Details tab of the Database Detail view provides an overview of the attributes of every user table defined in the database. The table below describes the information available on the Table Details tab of the Database Detail view:

Column	Description
Owner	The owner of the table.
Table Name	The name of the table.
Pinned	Indicates if the table has been pinned in the buffer cache.
File Group	The file group where the table resides.
Rows	The number of rows in the table (could be inaccurate if statistics are not kept up to date).
Clustered	Whether the table has a clustered index defined or not.
Indexes	The number of indexes defined to the table.
FK Constraints	Whether the table has foreign key constraints defined to it.
Chk Constraints	Whether the table has check constraints defined to it.
Identity	Whether the table has an IDENTITY column defined to it

NOTE: For space-related table information, use the Tables tab of the Space Detail view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

None.

Network Statistics

The Network view displays the following vital network statistics:

- Network Bytes Read
- Network Bytes Written
- Network I/O Wait Requests
- Network I/O Wait Time
- Network Reads
- Network Writes
- Packet Errors
- Packets Received
- Packets Sent
- Total Traffic

Related Topics

Home View Statistics

Database Statistics

Packets Received

- Metrics

SQL Server counts incoming packets from the time the server is initiated. This statistic displays the delta count of all incoming packets received over the network since the last Embarcadero Performance Center sampling.

Metrics

None.

Packets Sent

- Metrics

SQL Server counts outgoing packets from the time the server is initiated. This statistic displays the delta count of all outbound packets written to the network since the last Embarcadero Performance Center sampling.

Metrics

None.

Network Reads

- Metrics

The number of network reads performed by SQL Server since the last Embarcadero Performance Center sampling.

Metrics

None.

Network Writes

- Metrics

The number of network writes performed by SQL Server since the last Embarcadero Performance Center sampling.

Metrics

None.

Network Bytes Read

- Metrics

The number of bytes that SQL Server read from the network since the last Embarcadero Performance Center sampling.

Metrics

None.

Network Bytes Written

- Metrics

The number of bytes that SQL Server wrote to the network since the last Embarcadero Performance Center sampling.

Metrics

None.

Packet Errors

- Metrics
- Troubleshooting

SQL Server counts packet errors from the time the server is first initiated. This statistic displays the delta count of all packet errors that occurred since the last Embarcadero Performance Center sampling.

Metrics

This value should remain low relative to your overall network stability and efficiency. Networks with great amounts of congestion and other traffic problems causes these errors to occur.

Troubleshooting

In the event that packet errors continues to remain elevated beyond what you would expect in your particular environment, contact a Network Administrator for assistance in tracking the problem.

Total Traffic

- Metrics

The total number of packets that SQL Server received and sent over the network since the last Embarcadero Performance Center sampling.

Metrics

None.

Network I/O Wait Requests

- Metrics

The total number of times that SQL Server generated an I/O wait request since the last Embarcadero Performance Center sampling.

Metrics

None.

Network I/O Wait Time

- Metrics
- Troubleshooting

The number of milliseconds that SQL Server remained in a wait state due to Network I/O Wait Requests since the last Embarcadero Performance Center sampling.

Metrics

None.

OS Page Statistics

In many scenarios, an optimally tuned database may not perform well because there are constraints imposed by the system where the database is running. These constraints may include processes competing with the database sever for resources (CPU, I/O, or Memory), a slow CPU, insufficient or slow I/O devices, and insufficient memory. The OS Statistics page of Performance Center lets you examine operating system metrics for the following platforms:

- AIX
- HP-UX

  NOTE: To view processor info and swap disk info on an HP-UX box, you need to login as ROOT in the OS login.

- Linux
- Solaris
- Unix
- Windows XP, 2000, and NT

Summary Tab

The OS Summary tab displays the following statistics to communicate the general overall performance levels of the operating system:

- Disk Time
- Load Average
- Processor Time
- Paged Memory Used (Windows)
- Swap Memory Used (AIX, HP-UX, Linux, Solaris, Unix)

- Average Disk Queue
- Network Output Queue (Windows)
- Network Queue (Solaris)
- Page Faults/Sec
- Processor Queue
- Processor Speed
- Processor
- Available Paged Memory (Windows)
- Available Physical Memory
- Available Swap Memory (AIX, HP-UX, Linux, Solaris, Unix)
- Total Paged Memory (Windows)
- Total Physical Memory
- Total Swap Memory (AIX, HP-UX, Linux, Solaris, Unix)
- Free Disk Space
- Total Disk Space
- Used Disk Space
- Number of Logins
- Number of Processes
- Number of Processors
- Top CPU Process
- Top I/O Process
- Top Memory Process

Processor Time

The Processor Time statistic indicates the percentage of time the processor is working. This counter is a primary indicator of processor activity.

Metrics

If your computer seems to be running sluggishly, this statistic could be displaying a high percentage.

Troubleshooting

Upgrade to a processor with a larger L2 cache, a faster processor, or install an additional processor.

Processor Speed

The Processor Speed statistic displays the speed of the active processor in MHz. The speed is approximate.

Processor

The Processor Statistic displays the type of processor currently in use, for example, GenuineIntel.

Disk Time

The Disk Time statistic is the percentage of elapsed time that the selected disk drive/device was busy servicing read or write requests.

Metrics

You should avoid consistently seeing values for this statistic greater then 90%.

Troubleshooting

Add more disk drives and partition the files among all of the drives.

Load Average

The Load Average statistic represents the system load averages over the last 1, 5, and 15 minutes.

Metrics

High load averages usually mean that the system is being used heavily and the response time is correspondingly slow.

Paged Memory Used

The Paged Memory Used statistic is the ratio of Commit Memory Bytes to the Commit Limit. Committed memory is where memory space has been reserved in the paging file if it needs to be written to disk. The commit limit is determined by the size of the paging file. As the paging file increases, so does the commit limit.

NOTE: This statistic is available for the Windows platform.

Metrics

This value displays the current percentage value only and not an average. If the percentage of paged memory used is above 90%, you may be running out of memory.

Troubleshooting

Increase the size of page file.

Number of Processors

This statistic displays the number of processors currently in use.

Swap Memory Used

The Swap Memory Used statistic is the percentage of swap space currently in use.

Metrics

If the percentage of swap memory used is above 90%, you may be running out of memory.

Troubleshooting

Increase the size of your swap files.

Average Disk Queue

The Average Disk Queue statistic is the average number of both read and write requests that were queued for the selected disk during the sample interval.

Metrics

This metric is useful in identifying I/O related bottlenecks. If the disk queue lengths for certain disks are consistently much higher than others, you may need to redistribute the load among available disks. If the disk queues lengths for all disks are consistently large, and you see a high amount of I/O activity, your disks may be inefficient.

Troubleshooting

Some things you can do if you have problems with this statistic include:

- Redistribute the data on the disk with the large average disk queue to other disks.
- Upgrade to faster disk(s).

Page Faults/Sec

The Page Faults/Sec statistic is the overall rate faulted pages are handled by the processor. It is measured in numbers of pages faulted per second. A page fault occurs when a process requires code or data that is not in its working set. This counter includes both hard faults and soft faults.

Metrics

This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

If the number of page faults remains consistently high, you can check with your Windows System Administrator for further investigation. Often, large numbers of page faults are not a problem so long as they are soft faults. However, hard faults, that require disk access, can cause delays.

Processor Queue

The Processor Queue Length statistic is the number of threads in the processor queue.

Metrics

Unlike the disk counters, this counter shows ready threads only, not threads that are running. There is a single queue for processor time even on computers with multiple processors. Therefore, if a computer has multiple processors, you need to divide this value by the number of processors servicing the workload. A sustained processor queue of less than 10 threads per processor is normally acceptable, dependent of the workload.

Troubleshooting

A sustained high value in the Processor Queue could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Network Output Queue/Network Queue

The Network Output Queue Length statistic is the number of threads in the processor queue.

NOTE: The name of this statistic depends on the platform of the operating system.

Metrics

Unlike the disk counters, this counter shows ready threads only, not threads that are running. There is a single queue for processor time even on computers with multiple processors. Therefore, if a computer has multiple processors, you need to divide this value by the number of processors servicing the workload. A sustained processor queue of less than 10 threads per processor is normally acceptable, dependent of the workload.

Troubleshooting

A sustained high value in the Processor Queue Length could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Available Physical Memory

The Available Physical Memory statistic represents the amount of RAM available to all processes.

Metrics

This counter displays the last observed value only and not an average. Use this value with the Total physical memory and paging metrics (Memory details page). If the available physical memory is very small compared to this value, and the paging activity is high, your system may be running low on memory.

Troubleshooting

Some things you can do if you have problems with this statistic include:

- Check the running processes to see if there are any memory leaks.
- Stop any services that are not required.
- Install additional RAM.

Available Paged Memory

The Available Paged Memory statistic shows the amount of virtual memory available for the processes.

NOTE: This statistic is available for the Windows platform.

Metrics

If the available virtual memory is less than 10% of the total virtual memory, your system may run out of memory.

Troubleshooting

Increase the size of page file.

Available Swap Memory

The Available Swap Memory statistic represents the amount of virtual memory available for the processes.

Metrics

If the available Available Swap Memory is less than 10% of the total Swap Memory, your system may run out of memory.

Troubleshooting

Increase the size of swap files.

Total Physical Memory

The Total Physical Memory statistic shows the amount of physical memory installed on your computer.

Metrics

This is an informational metric and displays the total amount installed on the machine. Use this value with the available physical memory and paging metrics (Memory details page). If the available physical memory is very small compared to this value, and the paging activity is high, your system may be running low on memory.

Total Paged Memory/Total Swap Memory

The Total Paged Memory statistic shows the maximum amount of virtual memory available to all processes.

NOTE: The name of this statistic depends on the platform of the operating system.

Metrics

It is recommended that this value is between 1.5 to 3 times the amount of RAM on the system.

Used Disk Space

The Used Disk Space statistic shows the amount of allocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations:

- If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
- You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Total Disk Space

Total Disk Space displays the total allocated and unallocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations, here are two:

1. If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
2. You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Free Disk Space

The Free Disk Space statistic shows the unallocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations:

- If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
- You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Top Memory Process

Top Memory Process shows the current process that is consuming the most amount of memory. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Metrics

If you are running out of memory on the system, this is a quick way to identify the top memory user. If the displayed process is using a significant portion of the total memory, it could be causing the memory issues.

Processes Overview

The Processes Overview of the OS Summary includes the following sections:

- Top CPU Process
- Top I/O Process
- Top Memory Process

Top CPU Process

Top CPU Process shows the current process that is consuming the most amount of CPU. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Metrics

If the amount of CPU time used by this process is close to 100% and the CPU usage is very high, this process may be the bottleneck on the server.

Troubleshooting

Investigate the process further to see if it is in an inconsistent state. Also, look at minimum requirements for CPU speed for the process. You may need to upgrade your CPU.

Top I/O Process

The Top I/O Process statistic shows the current process that is consuming the most amount of CPU. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Number of Logins

This statistic displays the total number of logins on the server.

Number of Processes

This statistic displays the total number of processes on the server.

CPU Tab

The CPU tab of the OS Detail includes the following sections:

- Context Switches/Sec
- CPU Utilization
- Interrupts/Sec
- Processor Queue Length

CPU Utilization

The CPU Utilization section includes the following information:

- % Privileged Time
- % User Time

% Privileged Time

The % Privileged Time statistic is the percentage of elapsed time that the process threads spent executing code in privileged mode.

> **NOTE:** For Windows systems, when a Windows system service is called, the service will often run in privileged mode to gain access to system-private data. Such data is protected from access by threads executing in user mode. Calls to the system can be explicit or implicit, such as page faults or interrupts. These kernel commands, are considered privileged to keep the low-level commands executing and prevent a system freeze. Unlike some early operating systems, Windows uses process boundaries for subsystem protection in addition to the traditional protection of user and privileged modes. Some work done by Windows on behalf of the application might appear in other subsystem processes in addition to the privileged time in the process.

Metrics

The ideal range should be 0-40% (less than 40% indicates excessive system activity).

Troubleshooting

If your CPU consistently runs at less than 40% you may need to upgrade your system to include a faster processor(s).

% User Time

The % User Time statistic is the percentage of elapsed time the processor spends in the user mode. User mode is a restricted processing mode designed for applications, environment subsystems, and integral subsystems. The alternative, privileged mode, is designed for operating system components and allows direct access to hardware and all memory. The operating system switches application threads to privileged mode to access operating system services. This counter displays the average busy time as a percentage of the sample time.

Metrics

If the Privileged Time is high in conjunction with Physical Disk Reads, consider upgrading the disk I/O subsystem.

Interrupts/Sec

The Interrupts/Sec statistic is the average rate, in incidents per second, at which the processor received and serviced hardware interrupts. It does not include deferred procedure calls (DPCs), which are counted separately. This value is an indirect indicator of the activity of devices that generate interrupts, such as the system clock, the mouse, disk drivers, data communication lines, network interface cards, and other peripheral devices. These devices normally interrupt the processor when they have completed a task or require attention. Normal thread execution is suspended. The system clock typically interrupts the processor every ten milliseconds, creating a background of interrupt activity. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Metrics

The ideal rage should be 0-5000. A number greater then 5000 indicates possible excessive hardware interrupts; justification is dependent on device activity.

Context Switches/Sec

The Context Switches/Sec section shows the combined rate at which all processors on the computer are switched from one thread to another. Context switches occur when a running thread voluntarily relinquishes the processor, is preempted by a higher priority ready thread, or switches between user-mode and privileged (kernel) mode to use an Executive or subsystem service.

Metrics

The ideal range should be between 0-10,000. GA number greater then 10,000 may indicate too many threads contending for resources.

Processor Queue Length

The Processor Queue Length statistic is the number of threads in the processor queue. There is a single queue for processor time even on computers with multiple processors.

> **NOTE:** For Windows systems, unlike the disk counters, this counter shows ready threads only, not threads that are running.

Metrics

A sustained high value in the Processor Queue Length could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Processes Tab

The Processes tab of the OS Detail page succinctly communicates the general overall performance levels of processes. The columns available in this table depend on the platform of operating system. The table below describes the information available in the table on this tab:

Column	Description
Process	The name of the process.
User	The user of the process.
ID	The ID Process is the unique identifier of this process. ID Process numbers are reused, so they only identify a process for the lifetime of that process.
CPU	The CPU is the percentage of elapsed time that all of process threads used the processor to execution instructions.
User Mode	The User Mode is the percentage of elapsed time that the process threads spent executing code in user mode.
Memory **WINDOWS ONLY**	Memory is the current size, in bytes, of the virtual address space the process is using. Use of virtual address space does not necessarily imply corresponding use of either disk or main memory pages. Virtual space is finite, and the process can limit its ability to load libraries.
Memory (MB)	Memory is the current size, in bytes, of the virtual address space the process is using. Use of virtual address space does not necessarily imply corresponding use of either disk or main memory pages. Virtual space is finite, and the process can limit its ability to load libraries.
Memory	Memory is the percentage of the memory used of the total memory.
Active Memory	Active Memory is the amount of committed virtual memory, in bytes for this process. Active memory is the physical memory which has space reserved on the disk paging file(s). There can be one or more paging files on each physical drive. This counter displays the last observed value only; it is not an average.
I/O Data	The rate at which the process is reading and writing bytes in I/O operations. This counter counts all I/O activity generated by the process to include file, network and device I/Os.
Elapsed Time	The total elapsed time, in seconds, that this process has been running.
Thread Count	The number of threads currently active in this process. An instruction is the basic unit of execution in a processor, and a thread is the object that executes instructions. Every running process has at least one thread.

Column	Description
Handle Count	The total number of handles currently open by this process. This number is equal to the sum of the handles currently open by each thread in this process.
Priority	The current base priority of this process. Threads within a process can raise and lower their own base priority relative to the process' base priority.
Creating Proc ID	The Creating Process ID value is the Process ID of the process that created the process. The creating process may have terminated, so this value may no longer identify a running process.
Page Faults/Sec	Page Faults/Sec is the rate at which page faults by the threads executing in this process are occurring. A page fault occurs when a thread refers to a virtual memory page that is not in its working set in main memory. This may not cause the page to be fetched from disk if it is on the standby list and hence already in main memory, or if it is in use by another process with whom the page is shared.
Page File	Page File is the current number of kilobytes that this process has used in the paging file(s). Paging files are used to store pages of memory used by the process that are not contained in other files. Paging files are shared by all processes, and the lack of space in paging files can prevent other processes from allocating memory.
Private	Private is the current size, in kilobytes, of memory that this process has allocated that cannot be shared with other processes.

I/O Tab

The table below describes the information available in this section:

Column	Description
Disk	The disk number assignment.
Reading (KB/s)	The amount of bytes read from the device.
Writing (KB/s)	The amount of bytes written to the device.
Disk Read Time	Disk Read Time is the percentage of elapsed time that the selected disk drive was busy servicing read requests.
Disk Write Time	Disk Write Time is the percentage of elapsed time that the selected disk drive was busy servicing write requests.
Disk Time	Disk Time is the percentage of elapsed time that the selected disk was busy servicing requests.
Avg. Read Queue	Avg. Disk Read Queue Length is the average number of read requests that were queued for the selected disk during the sample interval.
Avg. Write Queue	Avg. Disk Write Queue Length is the average number of write requests that were queued for the selected disk during the sample interval.
Disk Reads/Sec	Disk Reads/Sec is the rate of read operations on the disk.
Disk Writes/Sec	Disk Writes/Sec is the rate of write operations on the disk.

Memory Tab

The Memory tab of the OS Detail page includes the following sections:

- Cache Efficiency
- Free Physical

- Free Paged
- Paging Activity
- Page Faults
- Total Physical
- Total Paged

Paging Activity

The Paging Activity section includes the following statistics:

- Pages Input/Sec
- Pages Output/Sec

Pages Input/Sec

The Pages Input/Sec statistic is the number of pages read from disk to resolve hard page faults. Hard page faults occur when a process requires code or data that is not in its working set or elsewhere in physical memory, and must be retrieved from disk.

Metrics

This value was designed as a primary indicator of the kinds of faults that cause system-wide delays. It includes pages retrieved to satisfy faults in the file system cache (usually requested by applications) and in non-cached mapped memory files. This counter counts numbers of pages, and can be compared to other counts of pages, such as Memory: Page Faults/sec, without conversion. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

Although it never hurts to have as much physical memory as your system can handle, there are some things you can check within your system to alleviate the memory bottleneck.

- Check to see if you have any drivers or protocols that are running but not being used. They use space in all memory pools even if they are idle.
- Check to see if you have additional space on your disk drive that you could use to expand the size of your page file. Normally, the bigger the initial size of your page file, the better, in performance terms.

Pages Output/Sec

The Pages Output/Sec statistic is the number of pages written to disk to free up space in physical memory. Pages are written back to disk only if they are changed in physical memory. A high rate of pages output might indicate a memory shortage.

Metrics

Windows NT writes more pages back to disk to free up space when low in physical memory. This counter counts numbers of pages, and can be compared to other counts of pages, without conversion. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

Although it never hurts to have as much physical memory as your system can handle, there are some things you can check within your system to alleviate the memory bottleneck.

- Check to see if you have any drivers or protocols that are running but not being used. They use space in all memory pools even if they are idle.
- Check to see if you have additional space on your disk drive that you could use to expand the size of your page file. Normally, the bigger the initial size of your page file, the better, in performance terms.

Free Physical

The Free Physical statistic is the amount of physical memory that is uncommitted.

Metrics

None.

Free Paged

The Free Paged statistic is the amount of uncommitted virtual memory.

Metrics

None.

Total Physical

The Total Physical statistic is the total physical memory available.

Metrics

None.

Total Paged

The Total Paged statistic is the amount of total virtual memory, in bytes. Used Memory is the physical memory that has space reserved on the disk paging file(s). There can be one or more paging files on each physical drive.

Metrics

None.

Page Faults/Sec

The Page Faults/Sec statistic is the overall rate faulted pages are handled by the processor. It is measured in numbers of pages faulted per second. A page fault occurs when a process requires code or data that is not in its working set. This counter includes both hard faults and soft faults.

Metrics

This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

If the number of page faults remains consistently high, you can check with your Windows System Administrator for further investigation. Often, large numbers of page faults are not a problem so long as they are soft faults. However, hard faults, that require disk access, can cause delays.

Cache Efficiency

The Cache Efficiency section of the Memory tab succinctly communicates the general overall performance levels of the server's memory. The following statistics are available in this section:

- Copy Read Hits%
- Data Map Hits%
- MDL Read Hits%
- Pin Read Hits%

Copy Read Hits %

The Copy Read Hits % statistic is the percentage of cache copy read requests that hit the cache and does not require a disk read to provide access to the page in the cache.

Metrics

When the page is pinned in the memory, the page's physical address in the file system cache will not be altered. A copy read is a file read operation where a page in the cache is copied to the application's buffer. Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate

Data Map Hits %

The Data Map Hits % statistic is the percentage of data maps in the file system cache that could be resolved without having to retrieve a page from the disk.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

MDL Read Hits %

The MDL Read Hits % statistic is the percentage of Memory Descriptor List Read requests to the file system cache that hit the cache and does not require disk access to provide memory access to the pages in the cache.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

Pin Read Hits %

The Pin Read Hits % statistic is the percentage of pin read requests that hit the file system cache and does not require a disk read in order to provide access to the page in the file system cache.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

Space Tab

The Space tab of the OS Detail page includes the following sections:

- Disk Space Free
- Disk Space Detail

Disk Space Free

The Disk Space Free metric displays the amount of free disk space in megabytes.

Metric

None.

Disk Space Detail

The Disk Space Detail section of the Space tab succinctly communicates the general overall performance levels of the server’s disks and space allotment. The table below describes the statistics in this section:

Statistic	Description
Partition	The drive letter of the disk.
Local Filesystem	The name of the file system.
Type	The type of file system.
Total Space	Total size of the disk/device's capacity expressed in MBs.
Used Space	Amount of MBs currently allocated on the particular disk/device.
Free Space	Amount of MBs currently unallocated and free on the particular disk/device.
Capacity	The percentage of space used on the device.
Mounted On	The mount point of the device.

Network Tab

The Network tab of the OS Detail page succinctly communicates the general overall performance levels of the server's networking. The statistics available in this section depend on the platform of operating system. The table below describes the information available in this section:

Column	Description
Network Interface	The name of network interface.
INET Address/Address	The IP address assigned to the network interface.
Pkts Sent/Sec	The number of packets sent per second.
Pkts Received/Sec	The number of packets received per second.
Sent (KB/Sec)	The number of bytes sent per second.
Received (KB/Sec)	The number of bytes received per second.
Out Pkts Discarded	The number of outbound packets discarded.
In Pkts Discarded	The number of inbound packets discarded.
Out Pkt Errors	The number of outbound packet errors.
In Pkt Errors	The number of inbound packet errors.
Queue Length	The queue length.
Collisions	The number of collisions.
Packets Discarded	The number of packets discarded.

Sybase Expert Guide

This section includes expert help for all Sybase ASE categories and statistics in the Embarcadero Performance Center views and pages. For detailed information on using the application, see Using Embarcadero Performance Center. This help is divided into the following sections:

Home View Statistics - Sybase

The Embarcadero Performance Center Home view lets you review availability and overall performance of all monitored databases from a single window. Statistics on the Home view are organized into the following categories:

- Contention Vital Signs
- I/O Vital Signs
- Memory Vital Signs
- Network Vital Signs
- Space Vital Signs
- Users Vital Signs

Memory Vital Signs

The following memory statistics are on the Sybase Home view:

- Data Cache Hit Rate
- Procedure Cache Hit Rate
- Large I/O Hit Rate
- Clean Buffer Grab Rate

Data Cache Hit Rate

- Metrics

- Troubleshooting

Data read from memory produces user response times many times faster than when that same data is read from disk. The Sybase data cache assists with keep physical I/Os to an absolute minimum.

The data cache hit rate is an excellent indicator of how often user requests for data are satisfied through memory vs. being physically read from disk. The table below describes the three key counters in Sybase used to arrive at this statistic:

Counter	Description
LOGICAL READS	Data read from memory for user requests.
PAGES PER I/O	The number of pages retrieved in a single I/O operation.
PHYSICAL READS	Data read physically from disk.

TIP: Click this statistic to drill down to the Cache Activity tab of the Memory Detail view.

Metrics

To help ensure excellent performance, keep your cache hit rate in the neighborhood of 90% or greater. Lower amounts can be okay for user ad hoc databases where sporadic, large table scan operations occur. However, anything below this general threshold for normal databases can require tuning attention, and the adjustment of the Sybase memory tuning parameters.

If you are using named caches, you can drill down into the cache hit rates for each named cache. This helps you understand which objects/operations are depressing the overall cache hit rate for the server.

Troubleshooting

If a problem is found in Sybase servers, versions 11-12, you can increase the amount of the total memory configuration parameter or reduce the percentage of memory allocated to the procedure cache (by default, the data cache assumes any free memory left over after Sybase has met its kernel and procedure cache needs). Take care when reducing the procedure cache, as this could reduce performance in the server as it relates to reading procedures in from disk.

For Sybase 12.5, the total memory configuration parameter can again be increased to provide more memory for the data cache (and any named caches), but in 12.5, if you want to reduce the size of the procedure cache, note that it is now configured in terms of literal size instead of a percentage of the overall configured memory.

Once the data cache has been adjusted, monitor Sybase to see if the cache hit rate improves. If it does not, another increase may be necessary and examination of unnecessary large table scan operations. Also, keep a careful eye on the actual machine's memory limits and swap activity. Increased swap activity can be indicative of too little memory left for the server machine.

Procedure Cache Hit Rate

- Metrics
- Troubleshooting

The Sybase procedure cache is used to hold the definitions and query plans of stored procedures and triggers. It is used for short-term memory needs like statistics and query plans for parallel queries. When a user executes a stored procedure, Sybase looks in the procedure cache for a query plan to use. If a query plan is available, Sybase places it on the most recently used (MRU) end of the memory chain and the procedure begins to execute. If no execution plan is in memory, or if all copies of the plan are currently being used, the query tree for the procedure is read in again from the data dictionary, optimized, put on the MRU end of the chain, and executed. Note that other operations, like CREATE INDEX, can also use the procedure cache even when no procedure is referenced.

The more often that a procedure's plan and definition can be referenced in memory, the better the procedure execution time.

TIP: Click this statistic to drill down to the Cache Activity tab of the Memory Detail view.

NOTE:

Metrics

A high procedure cache hit rate is a desirable thing. You should strive for a hit ratio between 95-100%, with 95% being a good performance benchmark for procedure code reference. Note that when a database is first started, the procedure cache hit rate is not at an optimal level because all code being used is relatively new, and as such, must be read in from disk and placed into the cache. If, however, after a solid hour or two of steady database time, the procedure cache hit rate has not increased to desirable levels, you should look into the possibility of increasing the amount of memory allocated to the cache.

You can drill down into the procedure cache to view the procedures currently in memory along with how much memory they are consuming.

If there is not enough memory to load a requested procedure, or the maximum number of compiled objects is already in use, Sybase returns an error (normally a 701).

Troubleshooting

If a problem is found in Sybase servers, versions 11-12, you can increase the amount of the total memory configuration parameter or increase the percentage of memory allocated to the procedure cache (by default, the data cache assumes any free memory left over after Sybase has met its kernel and procedure cache needs). You should be careful when increasing the procedure cache alone, as this could increase query response times due to more physical I/O being performed.

For Sybase 12.5, the total memory configuration parameter can again be increased to provide more memory for the Sybase server, but in 12.5, if you want to increase the size of the procedure cache, note that it is now configured in terms of literal size instead of a percentage of the overall configured memory.

Once the procedure cache has been adjusted, monitor Sybase to see if the cache hit rate improves. If it does not, another increase may be necessary. Also, keep a careful eye on the actual machine's memory limits and swap activity. Increased swap activity can be indicative of too little memory left for the server machine.

Large I/O Hit Rate

Large I/O can be enabled by splitting the default or any named cache into pools. By default, Sybase performs I/O operations based on a 2-KB page size. For queries where pages are stored and accessed in a sequential manner, it is possible to read many more data pages in a single I/O operation. Large I/O can greatly reduce disk access time when the right situations exist. Operations that routinely perform large table scans, access image or text data, do bulk copies, scan the leaf level of nonclustered indexes, or initiate DBCC tasks can benefit from large I/O.

If large I/O has been configured and is being used, you should observe a high percentage of hits (the number of times large I/O could be performed vs. the number of times large I/O requests were denied by the server). If large I/O is not configured, no large I/O activity should be present.

Metrics

As you might expect, if large I/O is in use, a high hit rate is desirable. You should strive for a hit ratio between 90-100%, with 90% being a good performance benchmark.

Troubleshooting

If large I/O is configured, but you see a low hit rate, you should configure more caches for large I/O use.

Clean Buffer Grab Rate

- Metrics
- Troubleshooting

As information is requested from users, buffers are moved into and out of the Sybase data cache. Pages are also modified in the cache (termed dirty buffers) and need to be written out to disk. If Sybase has to wait for a dirty buffer to be written out to disk before a requested buffer is placed into the cache, performance can suffer.

The clean buffer grab rate represents the percentage of time clean buffers were found and referenced in the cache as opposed to Sybase finding dirty buffers.

Metrics

Ideally, the clean buffer grab rate should stay at or near 100%.

Troubleshooting

Seeing a poor clean buffer grab rate for either the default or named caches could indicate that the cache size is too small. You can look into adjusting the total memory configuration parameter higher. Keep a careful eye on the actual machine's memory limits and swap activity. Increased swap activity can be indicative of too little memory left for the server machine.

- Suspect Databases

Contention Vital Signs

The following contention statistics are on the Sybase Home view:

- Blocking Lock Rate
- Deadlock Rate
- Device I/O Contention
- Network Contention Rate

Blocking Lock Rate

- Metrics
- Troubleshooting

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems. Although Sybase supports flexible locking mechanisms, blocking lock situations do crop up. Blocks are most often caused by user processes holding exclusive locks and not releasing them via a proper COMMIT frequency.

The blocking lock rate displays the percentage of times blocks occurred vs. the percentage of locks requested and immediately granted.

> **TIP:** Click this statistic to drill down to the Blocking Lock Rate tab of the Locks view.

Metrics

Immediately investigate a percentage much above zero so the situation does not mushroom.

You can easily drill down with Embarcadero Performance Center and discover the exact process(es) holding locks that are blocking out other user activity.

Troubleshooting

Once you discover a blocking lock situation, you can normally remedy it by issuing a KILL against the offending process. This eliminates the user's stranglehold on the objects the user was accessing, and usually results in other user processes completing in an instant. Embarcadero Performance Center makes discovering the blocked lock situation easier; preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in Sybase. You can change this behavior by modifying the Lock Wait Period configuration parameter, which limits the number of seconds that a process waits for a lock before timing out.

Deadlock Rate

A deadlock occurs when two processes have a lock on a separate page or object and each wants to acquire a lock on the other process' page or object. Each waits for the other to release the necessary lock. Sybase constantly checks for deadlocks and, when found, chooses the transaction that has accumulated the least amount of CPU time and terminates the transaction. The server then rolls back that transaction and issues a notification of the event. The other process gets to move forward.

A deadlock occurs when two processes have a lock on a separate page or object and each wants to acquire a lock on the other process' page or object. Each waits for the other to release the necessary lock. Sybase constantly checks for deadlocks and, when found, chooses the transaction that has accumulated the least amount of CPU time and terminates it (the transaction). The server then rolls back that transaction and issues a notification of the event. The other process gets to move forward.

The deadlock rate displays the percentage of times deadlocks occurred vs. the percentage of locks requested and immediately granted.

Metrics

You should immediately investigate a percentage much above zero to prevent the situation from mushrooming. You can easily drill down and discover the exact process(es) holding locks and deadlocks that are blocking out other user activity.

Troubleshooting

Well-designed applications can minimize deadlocks by always acquiring locks in the same order. You should always do updates to multiple tables in the same order.

Once Sybase discovers a deadlock, it takes action and remedies the situation. Embarcadero Performance Center makes it easier to discover how prevalent deadlock activity is on a system; preventing deadlocks from occurring in the first place is more difficult.

Those responsible for writing systems can minimize deadlocks by ensuring that applications acquire locks in the same order. Likewise, you should always do updates and other DML that act on multiple tables in the same order.

You can also shrink the amount of time that Sybase waits to check for deadlocks by modifying the deadlock checking period configuration parameter.

Device I/O Contention

- Metrics
- Troubleshooting

Device I/O Contention reflects the number of times a task or process was put to sleep while waiting for a semaphore for a particular database device.

When a task or process involves physical I/O, Sybase first fills out the block I/O structure and links it to a per engine I/O queue. If two or more Sybase engines request an I/O structure from the same device at the exact same time, the server puts one of them to sleep where it waits for the semaphore it needs.

Metrics

An I/O contention percentage consistently greater than 10% should be cause for concern.

Troubleshooting

Distributing the accessed objects across multiple devices can lessen contention for I/O device semaphores. You can also place devices on different physical drives to lessen contention at the operating system level as well.

Task Context Switches

- Metrics

Task Context Switches summarizes task-switch activity for all engines on SMP servers. You can use this statistic to observe the effect of reconfigurations.

Metrics

You might reconfigure a cache or add memory if tasks appear to block on cache search misses and to be switched out often. Then, check the data to see if tasks tend to be switched out more or less often.

Network Contention Rate

- Metrics
- Troubleshooting

The Sybase server normally sends and receives network packets at a regular rate. If the network begins to become saturated, Sybase could experience delays in network I/O. The network contention rate indicates the percentage of times network I/O activity was delayed.

Metrics

Seeing a network contention rate greater than 1% could indicate a challenged network structure.

Troubleshooting

Outside of ensuring that the existing network is sufficient for handling the current load (database and non-database), other items to look at from a Sybase specific standpoint include validating that only the necessary amount of data is being sent and returned to requesting users. You can also examine the default packet size and see if it is too small for the average packet size being sent/received by the Sybase server.

I/O Vital Signs

The following I/O statistics are on the Sybase Home view:

- Total Server Reads
- Total Server Writes
- Transaction Log Writes
- I/O Errors

Total Server Reads

- Metrics

Total Server Reads reflect the total number of physical reads performed by the database server since the last refresh inside Embarcadero Performance Center.

Metrics

Large numbers of physical reads could reflect a too small data or procedure cache. You should examine the data and procedure cache hit rates to determine the overall effectiveness of logical vs. physical I/O.

Total Server Writes

The Total Server Writes value reflects total number of physical writes performed by the database server since the last refresh inside Embarcadero Performance Center.

Metrics

None.

Transaction Log Writes

Transaction Log Writes refers to the number of times Sybase wrote a transaction log page to disk since the last refresh inside Embarcadero Performance Center. When the current log page becomes full, Sybase writes it out to disk. Sybase also writes transaction log pages to disk after a transaction commits.

Metrics

None.

I/O Errors

- Metrics

The I/O Errors value reflects total number of I/O errors (errors during read and write operations) encountered by the database server since the last refresh inside Embarcadero Performance Center.

Metrics

You should investigate large numbers of I/O errors by examining the database error log.

Users Vital Signs

The following user statistics are on the Sybase Home view:

- Total Connections
- Active Connections
- Committed Transactions
- Current Locks

Total Connections

- Metrics
- Troubleshooting

The Total Connections statistic represents the total number of open threads, or connections, currently reported in the Sybase server. This number includes both active and inactive processes.

> **TIP:** Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

You should view the total number of connections in light of the maximum number of processes allowed to connect to Sybase. The Number of User Connections parameter specifies the maximum number of user processes that can simultaneously connect to a Sybase server.

Troubleshooting

If the total number of connections approaches the number of user connections limit, you should:

1. Edit the configuration file for Sybase.
2. Increase the amount of number of user connections to a higher value.
3. Cycle the Sybase server when possible to allow the new value to take effect.

Active Connections

The Active Connections statistic represents the total number of active and open threads reported in the Sybase server. This number displays the number of processes actively performing work.

> **TIP:** Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

None.

Committed Transactions

- Metrics

Committed Transactions reflects the number of transactions committed since the last refresh inside Embarcadero Performance Center. This includes transactions that meet explicit, implicit, and ANSI definitions for committed transactions. Note also that Sybase counts multidatabase transactions.

> **TIP:** Click this statistic to drill down to the Transactions tab of the Users Detail view.

Metrics

Multidatabase transactions generally incur more overhead than single database transactions (for example, a transaction that modifies two databases is counted as two transactions). They usually involve more log activity, cause more ULC flushes, and involve two-phase commits between the different databases. You can improve performance by reducing the number of multidatabase transactions.

Current Locks

- Metrics
- Troubleshooting

The Current Locks statistic displays the total number of locks obtained/requested by processes in the database.

> **TIP:** Click this statistic to drill down to the All Locks tab of the Locks view.

Metrics

The main thing to watch with respect to current locks is that no DML locks currently held on the system approach the number of locks limit specified in the Sybase server's configuration. The parameter number of locks limits how many locks can exist on the system at one time.

Troubleshooting

If the total number of locks approaches the number of locks limit, you should:

1. Ensure that user processes are efficiently using locks and are committing frequently to avoid excessive lock hold times before editing the current Sybase configuration.
2. Edit the configuration file for the Sybase server.
3. Increase the amount of number of locks to a higher value.
4. Cycle the Sybase server when possible to allow the new value to take effect.

Space Vital Signs

The following space statistics are on the Sybase Home view:

- Databases Low on Space
- Logs Low on Space

Databases Low on Space

- Metrics

- Troubleshooting

The Databases Low on Space statistic indicates if any databases in the server are approaching low levels of available free space. Although good to know, you need a more detailed listing by database to determine where any actual space shortages exist in the server. You can view this information in the Embarcadero Performance Center Space performance category view.

Metrics

If any one database begins to approach 90% used space, and is continuing to dynamically grow, you should take action to prevent any future space allocation errors.

Troubleshooting

If a database is approaching its limit on space, a DBA can either extend the database onto a new device, or if space exists on the device where the database currently resides, the DBA can allocate more space for the database on the current device.

Logs Low on Space

- Metrics
- Troubleshooting

This statistic indicates if any log for a database in the Sybase server is approaching low levels of available free space. Although good to know, a more detailed listing by database is needed to determine where any actual space shortages exist in the server. This information can be viewed in the Embarcadero Performance Center Space performance category view.

Metrics

If a database log's used space begins to approach 90%, you should take action to prevent any future space allocation errors. If the transaction log runs out of space, no transactions can take place in the database until you free up space in the log.

Troubleshooting

If a database log consistently approaches its limit on used space, there are a few actions a DBA can take to prevent a database from freezing.

A backup (dump) of the log can be taken:

- If log backups are not required for disaster recovery, the truncate log on checkpoint option can be set for the database. Setting this option deletes any space devoted to transactions in the log that have already completed when a database checkpoint occurs.
- You can extend the log onto a new device. Or, if space exists on the device on which the database log currently resides, you can allocate more space for the log on the current device.

Network Vital Signs

The following network statistics are on the Sybase Home view:

- Network Requests
- Network Delays
- Bytes Received

- Bytes Sent

Network Requests

The Network Requests statistic represents the total TDS packets received and sent since Embarcadero Performance Center was last refreshed.

Metrics

None.

Network Delays

- Metrics
- Troubleshooting

Network Delays represents the number of times I/O was delayed since Embarcadero Performance Center was last refreshed.

Metrics

Seeing a value much above zero could indicate a challenged network structure.

Troubleshooting

Outside of ensuring that the existing network is sufficient for handling the current load (database and non-database), other items to look at from a Sybase specific standpoint include validating that only the necessary amount of data is being sent and returned to requesting users. You can also examine the default packet size and see if it is too small for the average packet size being sent/received by the Sybase server.

Bytes Received

This statistic represents the number of bytes received by Sybase since Embarcadero Performance Center was last refreshed.

Metrics

None.

Bytes Sent

Bytes Sent represents the number of bytes sent by Sybase to client applications since Embarcadero Performance Center was last refreshed.

Metrics

None.

Memory Statistics - Sybase

The Memory performance category view displays the following vital Sybase memory statistics:

- Cache Allocations
- Dirty Buffers Grabbed
- Data Cache Activity
- Dirty Read Requests
- Dirty Read Restarts
- Large I/O Hit Rate
- Large I/O Acquired
- Large I/O Denied
- New Pages Allocated
- Procedure Cache Activity
- Procedure Reads from Disk
- Procedure Requests
- Procedure Removals
- Procedure Writes to Disk
- Session Leaders - Memory

Data Cache Activity

- Metrics
- Troubleshooting

Data read from memory produces end-user response times many times faster than when that same data is read from disk. Keeping physical I/Os to an absolute minimum is something that the Sybase data cache tries to assist with.

The data cache hit activity rate is an excellent indicator of how often user requests for data are satisfied through memory vs. being physically read from disk. The table below lists three key counters in Sybase used to arrive at this statistic:

Counter	Description
LOGICAL READS	Data read from memory for user requests.
PAGES PER I/O	The number of pages retrieved in a single I/O operation.
PHYSICAL READS	Data read physically from disk.

TIP: Click this category heading to drill down to the Cache Activity tab of the Memory Detail view.

Metrics

To help ensure excellent performance, you want to keep your cache hit rate in the neighborhood of 90% or greater. Lower amounts can be okay for user ad hoc databases where sporadic, large table scan operations occur. However, anything below this general threshold for normal databases can require tuning attention, and the adjustment of the Sybase memory tuning parameters.

If you are using named caches, you can drill down into the cache hit rates for each named cache. This helps you understand which objects/operations are depressing the overall cache hit rate for the server.

Troubleshooting

If a problem is found in Sybase servers, versions 11-12, you can increase the amount of the total memory configuration parameter and/or reduce the percentage of memory allocated to the procedure cache (by default, the data cache assumes any free memory left over after Sybase has met its kernel and procedure cache needs). Take care when reducing the procedure cache, as this could reduce performance in the server as it relates to reading procedures in from disk.

For Sybase 12.5, the total memory configuration parameter can again be increased to provide more memory for the data cache (and any named caches), but in 12.5, if you want to reduce the size of the procedure cache, note that it is now configured in terms of literal size instead of a percentage of the overall configured memory.

Once the data cache has been adjusted, monitor Sybase to see if the cache hit rate improves. If it does not, another increase may be necessary as will an examination of unnecessary large table scan operations.

New Pages Allocated

The statistic reports the number of times that a new page was allocated in memory for Sybase since the last refresh in Embarcadero Performance Center.

Metrics

None.

Dirty Buffers Grabbed

As information is requested from users, buffers are moved into and out of the Sybase data cache. Pages are also modified in the cache (termed dirty buffers) and need to be written out to disk. If Sybase has to wait for a dirty buffer to be written out to disk before a requested buffer is placed into the cache, performance can suffer.

This statistic represents the number of times Sybase found dirty buffers since the last refresh in Embarcadero Performance Center.

Metrics

Ideally, the dirty buffer grab statistic should stay close to zero.

Troubleshooting

Seeing high numbers for this statistic could indicate that the cache size is too small. You can want to look into carefully adjusting the total memory configuration parameter higher. However, keep a careful eye on the actual machine's memory limits and swap activity. Increased swap activity can be indicative of too little memory left for the server machine.

Dirty Read Requests

Sybase allows dirty reads, which are reads of uncommitted data. To accomplish a dirty read, Sybase uses a special lightweight protection mechanism to gain access to an object without using actual page locks. This statistic displays the number of dirty reads that occurred since the last refresh in Embarcadero Performance Center.

Metrics

Dirty read page requests can incur significant overhead if they are observed with many dirty read restarts.

Dirty Read Restarts

Sybase allows dirty reads, which are reads of uncommitted data. To accomplish a dirty read, Sybase uses a special lightweight protection mechanism to gain access to an object without using actual page locks. A dirty read restart occurs when a dirty read is active on an object page, and another process makes changes to the page that cause the page to be deallocated in memory. The scan for the dirty read must be restarted. The amount shown for dirty read restarts are the number of restarts that occurred since the last Embarcadero Performance Center refresh.

Metrics

Values observed much above zero should serve as a signal that application modifications can be in order. Most applications should do everything possible to avoid restarts because of the large overhead they incur.

Troubleshooting

If the numbers observed for dirty read restarts are significant, you can want to look into modifying applications that use dirty reads to accomplish data acquisition.

ProcedureCache Activity

The Sybase procedure cache is used to hold the definitions and query plans of stored procedures and triggers and is used for short-term memory needs like statistics and query plans needed for parallel queries. When a user executes a stored procedure, Sybase looks in the procedure cache for a query plan to use. If a query plan is available, Sybase places it on the most recently used (MRU) end of the memory chain and execution of the procedure begins. If no execution plan is in memory, or if all copies of the plan are currently being used, the query tree for the procedure is read in again from the data dictionary, optimized, put on the MRU end of the chain, and executed.

> **NOTE:** Other operations, like CREATE INDEX, can also use the procedure cache even when no procedure is referenced.

The percentage of times that a procedure's plan and definition can be referenced in memory, the better the procedure execution time.

Metrics

A high procedure cache hit rate is a desirable thing. You should strive for a hit ratio between 95-100%, with 95% being a good performance benchmark for procedure code reference. Note that when a database is first started, the procedure cache hit rate is not at an optimal level because all code being used is relatively new, and as such, must be read in from disk and placed into the cache. If, however, after a solid hour or two of steady database time, the procedure cache hit rate has not increased to desirable levels, you should look into the possibility of increasing the amount of memory allocated to the cache.

> **NOTE:** You can drill down into the procedure cache to view the procedures currently in memory along with how much memory they are consuming.

If there is not enough memory to load a requested procedure, or the maximum number of compiled objects is already in use, Sybase returns an error (normally a 701).

Troubleshooting

If a problem is found in Sybase servers, versions 11-12, you can increase the amount of the total memory configuration parameter and/or increase the percentage of memory allocated to the procedure cache (by default, the data cache assumes any free memory left over after Sybase has met its kernel and procedure cache needs). Take care when increasing the procedure cache alone, as this could increase query response times due to more physical I/O being performed.

For Sybase 12.5, the total memory configuration parameter can again be increased to provide more memory for the Sybase server, but in 12.5, if you want to increase the size of the procedure cache, note that it is now configured in terms of literal size instead of a percentage of the overall configured memory.

Once the procedure cache has been adjusted, monitor Sybase to see if the cache hit rate improves. If it does not, another increase may be necessary. Also, keep a careful eye on the actual machine's memory limits and swap activity. Increased swap activity can be indicative of too little memory left for the server machine.

Procedure Requests

The Procedure Requests statistic reports the number of times that stored procedures were executed since Embarcadero Performance Center was last refreshed. Such a request could use either an unused copy of the procedure's query plan in memory or if no such copy exists, the procedure must be read in from disk.

Metrics

None.

Procedure Reads from Disk

The Procedure Reads from Disk statistic reports the number of times since Embarcadero Performance Center was last refreshed that stored procedures were read from disk rather than copied in the procedure cache.

Metrics

You should examine this number in conjunction with the overall procedure cache hit rate. Observing large numbers in this statistic indicates a lower than ideal procedure cache hit rate. Note that when a database is first started, this statistic is likely larger than desired because all code being used is relatively new and as such, must be read in from disk and placed into the cache. If, however, after a solid hour or two of steady database time, the procedure cache hit rate has not increased to desirable levels and this statistic continues to sport high numbers, you should look into the possibility of increasing the amount of memory allocated to the cache.

Troubleshooting

If a problem is found in Sybase servers, versions 11-12, you can increase the amount of the total memory configuration parameter and/or increase the percentage of memory allocated to the procedure cache (by default, the data cache assumes any free memory left over after Sybase has met its kernel and procedure cache needs). Take care when increasing the procedure cache alone, as this could increase query response times due to more physical I/O being performed.

For Sybase 12.5, the total memory configuration parameter can again be increased to provide more memory for the Sybase server, but in 12.5, if you want to increase the size of the procedure cache, note that it is now configured in terms of literal size instead of a percentage of the overall configured memory.

Once the procedure cache has been adjusted, monitor Sybase to see if the cache hit rate improves. If it does not, another increase may be necessary. Also, keep a careful eye on the actual machine's memory limits and swap activity. Increased swap activity can be indicative of too little memory left for the server machine.

Procedure Writes to Disk

This statistic reports the number of times since Embarcadero Performance Center was last refreshed that stored procedures were created.

Metrics

None.

Procedure Removals

This statistic reports the number of times since Embarcadero Performance Center was last refreshed that stored procedures were aged out of the procedure cache.

Metrics

High numbers, along with a lower than desired procedure cache hit rate, could indicate too small a procedure cache.

Troubleshooting

If a problem is found in Sybase servers, versions 11-12, you can increase the amount of the total memory configuration parameter and/or increase the percentage of memory allocated to the procedure cache (by default, the data cache assumes any free memory left over after Sybase has met its kernel and procedure cache needs). Take care when increasing the procedure cache alone, as this could increase query response times due to more physical I/O being performed.

For Sybase 12.5, the total memory configuration parameter can again be increased to provide more memory for the Sybase server, but in 12.5, if you want to increase the size of the procedure cache, note that it is now configured in terms of literal size instead of a percentage of the overall configured memory.

Once the procedure cache has been adjusted, monitor Sybase to see if the cache hit rate improves. If it does not, another increase may be necessary. Also, keep a careful eye on the actual machine's memory limits and swap activity. Increased swap activity can be indicative of too little memory left for the server machine.

Large I/O Hit Rate

Large I/O can be enabled by splitting the default or any named cache into pools. By default, Sybase performs I/O operations based on a 2-KB page size. For queries where pages are stored and accessed in a sequential manner, it is possible to read many more data pages in a single I/O operation. Large I/O can greatly reduce disk access time when the right situations exist. Operations that routinely perform large table scans, access image or text data, do bulk copying, scan the leaf level of nonclustered indexes, or initiate DBCC tasks can benefit from large I/O.

If large I/O has been configured and is being used, you should observe a high percentage of hits (the number of times large I/O could be performed vs. the number of times large I/O requests were denied by the server). If large I/O is not configured, no large I/O activity should be present.

The table below describes the other large I/O statistics available on the Memory Detail view:

Statistic	Description
Large I/O Acquired	This statistic measures the number of times that a requested large I/O operation was performed since the last Embarcadero Performance Center refresh.
Large I/O Denied	This statistic measures the number of times since the last Embarcadero Performance Center refresh that large I/O could not be performed since the last Embarcadero Performance Center refresh. You should examine nonzero numbers in light of the overall large I/O hit rate. If large I/O is configured, but you observe nonzero values for Large I/O Denied and a low Large I/O hit rate is also present, you should configure more caches for large I/O use.

Metrics

As you might expect, if large I/O is in use, a high hit rate is desirable. You should strive for a hit ratio between 90-100%, with 90% being a good performance benchmark.

Troubleshooting

If large I/O is configured, but a low hit rate is being observed, you should configure more caches for large I/O use.

Session Leaders - Memory

It is common for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Frequently, user connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system levels.

The leading memory session's display identifies the top users in the server with respect to memory consumption.

> **TIP:** Click this category heading to drill down to the Leading Sessions tab of the I/O Detail view.

Metrics

If your server does not have an overabundance of memory, you should periodically check to see who your heavy memory users are along with the total percentage of memory each takes up. If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing.

Troubleshooting

You can use the Session Leaders - Memory statistic to find the users with the greatest current allocations of overall memory. Runaway processes can be immediately terminated from within the Embarcadero Performance Center Client.

Cache Allocations

The Sybase cache areas are devoted to facilitating the transfer of data and information between clients and the Sybase database. The table below describes their main tunable components:

Component	Description
Procedure Cache	Is used to hold the definitions and query plans of stored procedures and triggers and is used for short-term memory needs like statistics and query plans needed for parallel queries.
Data Cache	Maintains data blocks that are read from the database. Properly sizing the data cache (both default and named caches) goes a long way to improving response time performance.

Metrics

None.

Memory Detail View

The Memory Detail view includes the following tabbed pages:

- Cache Activity
- Leading Sessions
- Page Activity

Leading Sessions Tab

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. User connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system levels.

Embarcadero Performance Center displays information to help you find processes that are using the most memory on the server on the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view.

The table below describes the information available on these tabs:

Column	Description
PID	The process ID.
User Name	The logon name the session is using.
FID	The process ID of the worker process' parent.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The client machine name the session is using.
Program	The executable the process is using against the Sybase server.
Memory	The amount of memory currently allocated to the process.
Pct Mem Used	The percentage of memory currently used by the process.
Database	The database the process is attached to.
Command	The command the process is currently issuing.
Trans	The name of any active transaction.

NOTE: This information is available on both the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If your database server does not have an overabundance of memory, you should periodically check to see who your heavy memory users are along with the total percentage of memory each takes up. If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing.

Cache Activity Tab

- Metrics

Database administrators can configure multiple caches for the Sybase server. These can be used judiciously by a DBA to hold objects being requested for data and speed end use response times. The Cache Activity tab of the Memory Detail view contains information that you can use to see if configured caches are being used the way they were intended. The table below describes the information available on the Cache Activity tab of the Memory Detail view:

Column	Description
Cache Name	The name of the configured cache.
Hit Rate	The percentage of times that a data page was found in the particular cache vs. having to be read in from disk. Higher percentages are desirable because data read from memory can be accomplished in much less time than data read from disk.
Pct Used	The percentage of the cache currently being used.
Spinlocks	How many spinlocks the cache has experienced.
LRU Buffers	Indicates the number of buffers acting in accordance with the normal cache strategy of moving to the most recently used (MRU) end of the cache.
MRU Buffers	The number of buffers following a fetch-and-discard cache strategy.
Large I/O	The number of times large I/O requests were performed.
Dirty Reads	The number of dirty reads (reads of uncommitted data) performed.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Seeing low hit rates for any configured dynamic caches indicates that they are not large enough and that users experience slower than desirable response times. On the other hand, caches with zero hit rates can indicate that they are not being used and that their memory could be released to another cache or the operating system itself. Another indicator that this can be the case is a very low percent used reading.

Page Activity Tab

- Metrics

Pages are read into and deallocated from memory constantly in a dynamic Sybase environment. The Page Activity tab of the Memory Detail view displays how many pages were moved into and out of memory during a particular monitoring interval. The table below describes the information available on the Page Activity tab of the Memory Detail view:

Column	Description
Page Action	The type of paging action that occurred, either Pages Allocated or Pages Released. Allocated pages are ones that were moved into memory during the monitoring interval. Released pages are ones that were deallocated from memory during the monitoring interval.
Count	The number of pages moved.

Metrics

Seeing allocation and deallocation numbers that are close can indicate objects that have a fetch-and-discard cache strategy. If large table scans are continuously needed on a system, this can be a valid strategy (because large amounts of unnecessary data kept in a cache can crowd out needed data pages). If this is not the case, investigate further to ensure that objects have a valid cache strategy (most recently used or MRU) in place.

I/O Statistics - Sybase

The I/O performance category view displays the following vital Sybase I/O statistics:

- Completed Disk I/Os
- Delayed Disk I/Os
- I/O Error Rate
- Outstanding Disk I/Os
- Requested Disk I/Os
- Server I/O Busy Rate
- Session Leaders - I/O
- Transaction Log Allocations
- Transaction Log Writes

Requested Disk I/Os

- Metrics

This statistic reports the number of times that Sybase requested disk I/Os since Performance Center was last refreshed.

> **TIP:** Click this statistic to drill down to the Devices tab of the I/O Detail view.

Metrics

The numbers for requested disk I/Os and completed I/Os should be very close. If there are large differences in these two values, it is likely that network contention or saturation is occurring.

Note that the value for requested I/Os includes every request that was initiated during the sample period. It is possible, however, that some of them completed after the sample period ended. These I/Os would then be excluded from the total number of completed I/Os, and can give the appearance that network problems exist when that is not the case. The number for completed I/Os can also be larger than the value for requested I/Os if I/O requests were made before Performance Center polled the server but were completed during the polling interval.

Completed Disk I/Os

- Metrics

The Completed Disk I/Os statistic reports the number of times that Sybase completed disk I/Os operations since Performance Center was last refreshed.

> **TIP:** Click this statistic to drill down to the Engines tab of the I/O Detail view.

Metrics

The numbers for requested disk I/Os and completed I/Os should be very close. If there are large differences in these two values, it is likely that network contention or saturation is occurring.

Note that the value for requested I/Os includes every request that was initiated during the sample period. It is possible, however, that some of them completed after the sample period ended. These I/Os would then be excluded from the total number of completed I/Os, and can give the appearance that network problems exist when that is not the case. The number for completed I/Os can also be larger than the value for requested I/Os if I/O requests were made before Performance Center polled the server but completed during the polling interval.

Outstanding Disk I/Os

- Metrics

The Outstanding Disk I/Os statistic reflects the maximum number of I/Os pending for Sybase since the last refresh in Embarcadero Performance Center.

> **TIP:** Click this statistic to drill down to the Engines tab of the I/O Detail view.

Metrics

If nonzero numbers are observed for both outstanding and delayed disk I/Os, there could be a problem in many areas. The table below describes these areas:

Area	Description
Delayed Disk I/O	This statistic indicates the number of I/Os delayed for Sybase since the last refresh in Embarcadero Performance Center.
Transaction Log Writes	This statistic indicates the number of times Sybase wrote a transaction log page to disk since the last refresh in Performance Center. When the current log page becomes full, it is written to disk. Transaction log pages are also written to disk after a transaction commits.
Transaction Log Allocations	This statistic indicates the number of times since Performance Center was last refreshed that additional pages were allocated to the transaction log. You can use this statistic for getting a feel for the rate of transaction log growth.
Server I/O Busy Rate	This statistic indicates number of seconds in CPU time that Sybase has spent doing input and output operations. Seeing a consistent rate of 90-100% could indicate a CPU bound server.

Transaction Log Writes

Transaction Log Writes refers to the number of times Sybase wrote a transaction log page to disk since the last refresh inside Embarcadero Performance Center. When the current log page becomes full, Sybase writes it out to disk. Sybase also writes transaction log pages to disk after a transaction commits.

Metrics

None.

Transaction Log Allocations

Transaction Log Allocations refers to the number of times since Performance Center was last refreshed that additional pages were allocated to the transaction log. This statistic gives you a feel for the rate of transaction log growth.

Metrics

None.

Session Leaders - I/O

- Metrics

Heavy I/O activity in a system can indicate that the user connections are contributing somewhat equally to the overall load. More often than not, however, one or two user connections are responsible for 75% or more of the I/O activity. It may be that your system is running a large batch load or other typical processes, which are perfectly okay. On the other hand, it may be a runaway process or other rogue connection, which you need to track down and possibly eliminate.

The leading I/O session's display lets you see who the leading sessions are in your system with respect to I/O.

NOTE: This statistic displays on both the Users performance category view and the I/O performance category view.

TIP: Click this statistic to drill down to the Leading Sessions tab of the I/O Detail view.

Metrics

One or two users consuming more than 75% of the total I/O load can indicate a runaway or improper process. You can drill into the I/O activity of all users and quickly see if this is the case.

I/O Error Rate

- Metrics

The I/O Error Rate statistic represent the percentage of times I/O errors occurred in the server. I/O Error Rate is a percentage based on Total I/O (the sum the physical reads and writes.)

Metrics

You should investigate large numbers of I/O errors by examining the error log.

- Delayed Disk I/Os
- Total I/O

Delayed Disk IOs

- Metrics
- Troubleshooting

The Delayed Disk IOs value is the number of I/Os delayed when the limit on disk I/O structures is reached.

Metrics

When Adaptive Server exceeds the number of available disk I/O control blocks, I/O is delayed because Adaptive Server requires that tasks get a disk I/O control block before initiating an I/O request.

Troubleshooting

If you see a non-zero value for delayed disk I/Os, try to add to the number of available disk I/O control blocks by increasing the configuration parameter.

Total I/O

- Metrics

The Total I/O statistic represents the total number of physical reads and writes.

Metrics

None.

I/O Errors

- Metrics
- Troubleshooting

I/O error rate reflects total number of I/O errors (errors during read and write operations) encountered by the server since the last refresh inside Performance Center. The I/O errors rate is a percentage based on Total I/O (the sum of the physical reads and writes).

Metrics

You should observe few, if any errors.

Troubleshooting

If you notice any errors, you should check the Sybase error log for details.

I/O Busy

- Metrics
- Troubleshooting

The I/O Busy statistic represents the number of clock ticks in the sample interval during which the user task performed I/O operations.

Metrics

High numbers indicate an I/O-intensive process. If idle time is also high, the application could be I/O bound.

Troubleshooting

The application might achieve better throughput if you assign it a higher priority, bind it to a lightly loaded engine or engine group, or partition the application's data onto multiple devices.

- Average Number of Writes per Log Page

Average Number of Writes per Log Page

- Metrics

The Average Number of Writes is the average number of times each log page was written to disk.

Metrics

In high throughput applications, this number should be as low as possible. If the transaction log uses 2K I/O, the lowest possible value is 1; with 4K log I/O, the lowest possible value is .5, since one log I/O can write 2 log pages.

In low throughput applications, the number will be significantly higher. In very low throughput environments, it may be as high as one write per completed transaction.

The following statistics are used on the Performance Center for Sybase I/O page to succinctly communicate the general overall performance levels of I/O:

- Top I/O Hogs

Top I/O Hogs

- Metrics
- Troubleshooting

The Top I/O Process statistic identifies the processes that has currently caused the most I/O usage on the database.

The table below describes the information available from the Top I/O Hogs view of the I/O page:

Column	Description
SPID	The process ID.
Login	The login name the session is using.
Physical I/O	The physical amount of I/O the process is using.
% Used	Percent of total I/O on the server this process is consuming.

Metrics

None.

Troubleshooting

If any one session uses more than 50% of a total resource (CPU, memory, etc.) you should drill down into that particular session and investigate the cause.

I/O Detail View

The following tabbed pages are available on the I/O Detail view:

- Devices
- Engines
- Index Scans
- Leading Sessions

Leading Sessions Tab

- Metrics

When a system undergoes heavy I/O activity, sometimes you find that all the user connections are contributing somewhat equally to the overall load. Frequently, however, one or two user connections are responsible for 75% or more of the I/O activity. It may be that a large batch load or other typical process is running that is perfectly okay for your system. Alternatively, it may be a runaway process or other rogue connection that you should track down and eliminate.

Embarcadero Performance Center displays information to help you find processes that are using the most memory on the server on the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view.

The table below describes the information available on these tabs:

Column	Description
PID	The process ID.
User Name	The logon name the session is using.
FID	The process ID of the worker process' parent.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The client machine name the session is using.
Program	The executable the process is using against the Sybase server.
Physical I/O	The amount of I/O the process has currently accumulated.
Pct I/O Used	The percentage of overall I/O that can be attributed to the process.
Database	The database to which the process is attached.
Command	The command the process is currently issuing.
Transaction	The name of any transaction.
Blocked	Indicates if the process is currently blocked by another process.
Time Blocked	The amount of time that the process has been blocked, in seconds.

NOTE: This information is available on both the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Pinpointing sessions with abnormally high I/O activity relative to other sessions in the system helps you identify accounts dragging down overall system performance. You should examine the activity of each session to determine the system workload and to determine if you can reduce the workload or tune the system for better performance.

Devices Tab

- Metrics
- Troubleshooting

Devices are accessed repeatedly in a dynamic Sybase environment to satisfy end user requests for data, to handle write activity that records transactions in a database's transaction log, and to manage other I/O operations. Viewing the I/O activity for each device is a good way to see what the "hot" devices are in a Sybase server with respect to I/O usage. The same information can be used to spot heavy database usage in systems where the device-to-database mappings are one-to-one.

The Devices tab of the I/O Detail view displays information that is useful in determining device I/O patterns. The table below describes the information available on the Devices tab of the I/O Detail view:

Column	Description
Device Name	The name of the device.
Physical Name	The name used by the operating system to identify the device.
Hit Rate	The rate at which accesses to a device were granted immediately compared to the total number of requests.
Reads	The total number of read operations for the device.
Writes	The total number of write operations for the device.
Total I/O	The combined total of read and write operations.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If devices have one-to-one relationships with user databases, you can quickly tell which databases are experiencing the most demand on a server. If the device-to-physical drive/file system is a one-to-one relationship, you can also spot which physical drives on a server are getting the heaviest workouts. For servers that have many drives, it is normally desirable to spread devices across different physical drives and controllers so contention does not occur at the disk level. In addition, separating databases and their corresponding logs is normally recommended so that each is located on a distinct physical drive. If possible, write-intensive devices, like log devices, are best suited for non-RAID5 scenarios.

Troubleshooting

If device hit rates are low, you can add more devices or redistribute objects among different devices. Typically, segments can be used to redistribute objects among different devices or physical hard disks. Common techniques include placing tables and indexes on different segments and partitioning large tables.

If device loads appear skewed (one device has much more activity than others), you should focus attention on that device. Again, redistributing objects can lessen the device's workload.

Engines Tab

In symmetric multiprocessing (SMP) environments, a DBA can configure the Sybase server to use more than one "engine," which represents a certain amount of CPU power. By default, Sybase configures one engine for use. If you have a server machine with multiple CPUs, you can enable more engines to take advantage of the machine's additional processing ability.

The Engines tab of the I/O Detail view displays information with respect to how each engine is handling I/O. The table below describes the information available on the Engines tab of the I/O Detail view:

Column	Description
Engine	The name of the configured engine.
Completed I/Os	The number of I/Os completed during the sample interval.
Outstanding I/Os	The number of I/Os left outstanding during the sample interval.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If the number of outstanding I/Os remains high or increases during periods of heavy activity, there may not be enough engines configured for the system.

Troubleshooting

If the server machine has multiple CPUs, you can configure more engines for Sybase to use by following this process:

1. Use the sp_configure procedure to change the current engine configuration. For example, to change the number of engines from one to two, you would run: 'sp_configure "max online engines",2'
2. Stop and restart the Sybase server.

Index Scans Tab

- Metrics

Indexes are accessed frequently in dynamic Sybase server environments. The type of index access often determines the response time an end user experiences. Single row index accesses are the quickest, and complete index scans are the most time consuming (for large indexes at least).

The Index Scans tab of the I/O Detail view presents information with respect to index scans. The table below describes the information available on the Index Scans tab of the I/O Detail view:

Column	Description
Scan Type	The type of index scan.
Count	The number of scans per type for the sample interval.

Metrics

There are two basic scan types - ascending and descending. For ascending scans, Sybase moves up the index page chain from beginning to end. Sybase follows the page chain in reverse order for descending scans. Descending scans are normally the result of requests made for data by descending column order.

Within ascending and descending scans, a data-only lock, or DOL, styled access can also occur.

Space Statistics - Sybase

The Space performance category view displays the following vital Sybase space statistics:

- Database Overview
- Log Overview
- Device Overview

Database Overview

- Metrics
- Troubleshooting

A Sybase server contains many databases, some of which are devoted to system-level activities (the master and tempdb databases, for example) and others that hold user data. The database overview displays details about the space situation for each database in a Sybase server, including the total, used, and free space. The percentage used amount for each database is also shown.

> **TIP:** Click this statistic to drill down to the Object Detail tab of the Space Detail view.

Metrics

If a database's used space percent amount goes above 90%, and the database is dynamic in nature (meaning that users are constantly adding and modifying data), then you should take action to ensure that the database does not run out of available free space.

Troubleshooting

If the percent used amount of a database is approaching problematic levels, there are two ways you can rectify the situation:

1. If the database device that the database currently resides on contains additional free space, you can ALTER the database to consume more available space on that device.
2. If the database device that the database currently resides on does not contain additional free space, you can do one of the following:
   - Create a new device and issue an ALTER for the database to use that device for space in addition to those currently used.
   - Choose another existing device that has free space and ALTER the database to use that device in addition to those currently in use.

Log Overview

Each database in Sybase has a transaction log, which can be placed on the same device as its database or assigned to a separate device for backup and performance purposes. The log overview displays details concerning each transaction log in the Sybase server and includes the total, used, and free space amounts. The percent used amount for each log is also shown.

Metrics

If any log's used space exceeds 90%, you should take action to ensure that the log does not run out of available free space.

Troubleshooting

There are many things a DBA can do to ensure that a database's log does not run out of available free space:

1. First, most transactional-oriented databases should have their logs assigned to a device separate from the database. Reasons for doing so include:
   - It lets you have more granular and point-in-time backups because a log separate from its database's device can make use of the dump transaction command.
   - It prevents competition for space between the log and the database itself.
   - It allows the log to be monitored for space more effectively.
   - It improves performance.
   - It enables better recovery from hard disk crashes because you can dump your transaction log, even when your data device is on a server's damaged hard disk.
2. If the database is not critical in nature, you can set the truncate log on checkpoint option (trunc log on chkpt), which eliminates any inactive space in the log when a database checkpoint occurs.
3. Critical databases needing higher levels of recovery should have schedules established that regularly perform transaction log dumps. Doing so ensures better recovery scenarios as well as a reduced risk of the transaction log running out of space.
4. If a critical transaction log becomes full, it can be impossible to use standard procedures to dump transactions and reclaim space. The dump operation incorporates the no log or truncate only options.
5. If a transaction log continuously approaches dangerously low levels of free space, you should expand the database (if the transaction log is on the same device as the database) or expand the log onto its current device (if room exists) or a new device.

You should also be on the lookout for large load or data modification operations that do not make use of prudently timed commit points. A single, large transaction has the ability to overwhelm any transaction log because only inactive space in the transaction log is removed from log dumps or truncation operations.

Device Overview

Database devices are the logical containers for databases and transaction logs. Each device is mapped to a physical file that is located on a database server's hard drive. The device overview displays details about every defined device on a Sybase server, including the total, used, and free space amounts. The percent of used space is also displayed.

> **TIP:** Click this statistic to drill down to the Device Detail tab of the Space Detail view.

Metrics

The environment of the particular Sybase server as well as your work style dictate the metrics to use in evaluating a device that is running into trouble with space. Many DBAs create devices that parallel a single, corresponding database in size, and therefore, such devices show 100% utilization. Other DBAs create large devices that are not completely utilized.

Troubleshooting

If a device has become too full, a DBA can begin the process of manually relocating databases from it onto other devices. The process of moving logs is somewhat easier and can be accomplished via singular commands (sp_logdevice).

Space Detail View

The following tabbed pages are available on the Space Detail view:

- Cache Detail
- Device Detail
- Object Detail
- Segment Detail

Device Detail Tab

- Metrics
- Troubleshooting

The Device Detail tab of the Space Detail view presents details about a selected device located on the monitored Sybase server. The table below describes the information available on the Device Detail tab of the Space Detail view:

Column	Description
Database Name	The name of a database that uses all or part of the device for its database space, transaction log space, or both.
Space	The amount of space used by the database on the device.
Space Type	The type of space allocated (database, log, or both).

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

The environment of the particular Sybase server as well as the DBA's work style dictate the metrics to use in evaluating a device that is running into trouble with space. Many DBAs create devices that parallels a single, corresponding database in size, and therefore, such devices always show 100% utilization. Other DBAs create large devices that are not completely utilized.

Troubleshooting

If a device has become too full, a DBA can begin the process of manually relocating databases from it onto other devices. The process of moving logs is somewhat easier and can be accomplished via singular commands (sp_logdevice).

Cache Detail Tab

- Metrics
- Troubleshooting

Caches retain frequently requested data and object information in memory. They can potentially boost performance by allowing users to read or write data in memory instead of to and from physical disk. Information requests satisfied in memory equate to much better response times than if that same data requires physical I/O access.

Named data caches can help achieve substantially greater concurrency and increase the likelihood that Sybase reads data from memory and not disk by letting you bind particularly active database objects to dedicated data caches. In addition, you can boost performance by creating memory pools, each configured for different I/O throughput (2 KB-16 KB), within every named data cache.

The table below describes the information available on the Cache Detail tab of the Space Detail view:

Column	Description
Cache Name	The name of the data cache.
Status	The status of the cache (Active, Activation Pending, Deletion Pending).
Mode	Mixed or log only.
Configured Size	The defined size of the cache.
Running Size	The actual running size of the cache.
Overhead	The amount of overhead required to manage the cache.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

The data cache hit rate is an excellent indicator of how often user requests for data are satisfied through memory vs. being physically read from disk. To help ensure maximum performance, you want to keep your cache hit rate in the neighborhood of 90% or greater. Lower amounts can be okay for user ad hoc databases where sporadic, large table scan operations occur. But anything below this general threshold for normal databases can require tuning attention and the adjusting the Sybase memory tuning parameters.

If you are using named caches, you can drill down into the cache hit rates for each named cache (from the Memory performance category view). This helps you understand which objects/operations are depressing the overall cache hit rate for the server.

Troubleshooting

If a problem is found in Sybase servers, versions 11-12, you can increase the amount of the total memory configuration parameter or reduce the percentage of memory allocated to the procedure cache (by default, the data cache assumes any free memory left over after Sybase has met its kernel and procedure cache needs). Take care when reducing the procedure cache, as this could reduce performance in the server as it relates to reading procedures in from disk.

For Sybase 12.5, the total memory configuration parameter can again be increased to provide more memory for the data cache (and any named caches), but in 12.5, if you want to reduce the size of the procedure cache, note that it is now configured in terms of literal size instead of a percentage of the overall configured memory.

Once the data cache has been adjusted, monitor Sybase to see if the cache hit rate improves. If it does not, then another increase may be necessary as will an examination of unnecessary large table scan operations. Also, keep a careful eye on the actual machine's memory limits and swap activity. Increased swap activity can be indicative of too little memory left for the server machine.

Segment Detail Tab

- Metrics
- Troubleshooting

Segments are instruments used for mapping objects (tables and indexes) to specific physical locations on disk. Mapping certain objects to defined segments allows you to potentially increase I/O throughput by placing heavily used tables and indexes on different physical devices. You can place tables and indexes on segments by including placement statements in your CREATE TABLE or CREATE INDEX statements.

The Segment Detail tab of the Space Detail view lets you view information about segments for selected databases. The table below describes the information available on the Segment Detail tab of the Space Detail view:

Column	Description
Segment Name	The name of the segment.
Total (KB)	Represents the total amount of space defined for the segment, in KB. Typically this equates to the amount of space on the device where the segment resides.
Used (KB)	Represents the amount of space used by the segment alone, in KB. Segments containing no objects will show zero space used.

NOTE: This information is available on both the Segment Detail tab of the Space Detail view and the Segment Detail grid on the Storage tab of the Database Detail view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

It is often good practice to separate tables and their corresponding indexes onto different physical devices to minimize I/O contention at the server level. Segments can be used to accomplish this within Sybase. If you have hot tables and indexes within your system, investigate the use of segments to place such objects on separate, faster physical devices.

Troubleshooting

Hot objects can be relocated onto different devices through the use of the sp_placeobject procedure.

Object Detail Tab

Tables and indexes comprise every Sybase database. The Object Detail tab of the Space Detail view presents space-related information about tables and indexes for a selected database. The table below describes the information available on the Object Detail tab of the Space Detail view:

Column	Description
Owner	The owner of the object.
Object Name	The name of the table or index.
Type	The type of object (table or index).
Rows	The number of rows currently in the table.
Reserved (KB)	The amount of space reserved for the object, in KB.
Data (KB)	The amount of space consumed by data, in KB.
Indexes (KB)	The amount of space used by an index, in KB.
Unused (KB)	The amount of unused space (free space) that the object contains, in KB.
Segment Name	The segment name that the object is assigned to.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Databases Statistics - Sybase

Databases with Suspect Pages

Suspect objects normally indicate an internal problem in a server. They can also indicate that physical damage has occurred to part of a database.

Metrics

Suspect objects have no place in a production database. If you identify any suspect objects, you should take immediate action.

Troubleshooting

If the suspect object is an index, you could try dropping and recreating it, or use the DBCC REINDEX command. Other damaged objects can complete rebuilding the database. A suspect database can be a difficult thing to recover from quickly. There are times when the cache of the database is suspect and not the database itself. Stopping and starting the Sybase server can verify if this is the case. If the database itself is actually damaged, there could be a true recovery situation. The suspect database can be dropped using the DBCC DBREPAIR DROPDB command. You would then need to recreate the database and perform a recovery operation using the most recent database dump.

Suspect Databases

The Suspect Databases value represents the number of suspect databases on the server.

Metrics

None.

Troubleshooting

It can be difficult to recover from a suspect database quickly. There are times when the cache of the database is suspect and not the actual database itself. Stopping and starting the Sybase server can verify if this is the case. If the database itself is actually damaged, there could be a true recovery situation. The suspect database can be dropped using the DBCC DBREPAIR DROPDB command. You would then need to recreate the database and perform a recovery operation using the most recent database dump.

The Databases performance category view displays Sybase Database Summary statistics.

Database Summary

The Database Summary display lists every database in a Sybase server and includes an overview of the activity currently being experienced in each database. It also shows demographic information about each database, which includes the database owner, the last transaction dump date, and if the database itself or any object contained within it is suspect.

> **TIP:** Click this statistic to drill down to the Suspect Objects tab of the Database Detail view.

Metrics

Databases with high numbers of blocked users could signal a contention problem. You should immediately investigate suspect databases or database objects.

Troubleshooting

The action you need to take depends on the situation you are watching. The table below describes four situations that you can recognize and fix:

Situation	Description
Blocked Users	You should investigate lock contention problems in detail. If the blocking lock situation has been long in duration, or involves many users, you should consider terminating the process holding the offending locks.
Suspect Objects	If Embarcadero Performance Center has indicated that a database contains suspect objects, the first thing to do is drill down to find exactly which objects are suspect. You can also do this manually by running a DBCC against the database. If the object is an index, you could try dropping and recreating it, or use the DBCC REINDEX command.
Suspect Database	A suspect database can be a difficult thing from which to quickly recover. There are times when the actual cache of the database is suspect and not the actual database itself. Stopping and starting the Sybase server can verify if this is the case. If the database itself is actually damaged, there could be a true recovery situation. The suspect database can be dropped using the DBCC DBREPAIR DROPDB command. You would then need to recreate the database and perform a recovery operation using the most recent database dump.
Suspect Pages	If you have to drop a table or index containing suspect pages, you must use a transaction in the master database. If you don't, the drop fails because it needs to delete entries for the suspect pages from the master database.

Database Detail View

The following tabbed pages are available on the Database Detail view:

- Index Statistics
- Options
- Storage
- Suspect Objects
- Table Statistics

Suspect Objects Tab

- Metrics
- Troubleshooting

Suspect objects normally indicate an internal problem in a server or that physical damage has occurred to part of a database. The Suspect Objects tab of the Database Detail view presents objects in a Sybase server that have been marked suspect for a selected database. The table below describes the information available on the Suspect Objects tab of the Database Detail view:

Column	Description
Owner	The owner of the suspect object.
Table Name	The name of the table.
Index Name	The name of any suspect index.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Suspect objects have no place in a production database. If any suspect objects are identified, you should immediately take action.

Troubleshooting

If the suspect object is an index, you could try dropping and recreating it, or use the DBCC REINDEX command. Other damaged objects can complete rebuilding the database. A suspect database can be a difficult thing to recover from quickly. There are times when the cache of the database is suspect and not the database itself. Stopping and starting the Sybase server can verify if this is the case. If the database itself is actually damaged, there could be a true recovery situation. The suspect database can be dropped using the DBCC DBREPAIR DROPDB command. You would then need to re-create the database and perform a recovery operation using the most recent database dump.

Table Statistics Tab

Understanding the placement, shape, and organization of your tables enhances your ability to make correct decisions regarding their use and maintenance. The Table Statistics tab of the Database Detail view communicates useful measures to gauge the overall condition of the tables in a selected database. The first grid displays table demographic information. The second grid displays cache binding information. The table below describes the information available in the Table Demographics grid on the Table Statistics tab of the Database Detail view:

Column	Description
Owner	The owner of the table.
Table Name	The name of the table.
Segment Name	The segment that holds the table.
Last Statistic Date	The date and time when statistics were last gathered for the table.
Row Count	The reported number of rows in the table.
Reserved	The amount of space reserved (in KB) of the table.
Avg Row Size	The average size of a row in the table.

The table below describes the information available in the Cache Bindings grid on the Table Statistics tab of the Database Detail view:

Column	Description
Owner	The owner of the table.
Table Name	The name of the table.
Cache Name	The cache to which the table has been bound.

TIP: To configure a grid to display row numbers, use the Options Editor.

Metrics

There are a few data points worth noting with regard to the information presented in the Table Statistics tab of the Database Detail view:

- Heavily used tables and indexes can benefit from being separated onto different segments that correspond to different physical hard drives.
- The Sybase cost-based optimizer may not be choosing the proper access path for tables with out-of-date, or without any, statistics. Keeping dynamic objects current with respect to their internal statistics helps ensure that the Sybase kernel makes the proper decisions on how to satisfy user requests for data.
- Frequently accessed small - or look up - tables can benefit from being bound to a special cache that is used to retain their often-accessed data in memory.

Troubleshooting

To change the segment a table or index belongs to, you can use the sp_placeobject procedure. Keeping table statistics current can be handled by judicious use of the UPDATE STATISTICS command. Binding frequently accessed tables can be accomplished via the sp_bindcache procedure.

Index Statistics Tab

Understanding the placement, shape, and organization of your table's indexes enhances your ability to make the right decisions about using and maintaining them. The Index Statistics tab of the Database Detail view communicates a number of useful measures to gauge the overall condition of the indexes in a selected database. The first grid displays index demographics. The second grid displays cache bindings. The table below describes the information available in the Index Demographics grid on the Index Statistics tab of the Database Detail view:

Column	Description
Owner	The owner of the table.
Index Name	The name of the index in <table name>.<index name> fashion.
Segment Name	The segment that holds the table.
Reserved (KB)	The amount of space reserved of the table, in KB.
Tree Depth	The height of the index. Can hold bogus values for indexes where statistics have not been gathered.
Leaf Count	Number of leaf pages in the index.
Leaf Row Size	The average size of a leaf row in the index.

The table below describes the information available in the Cache Bindings grid on the Index Statistics tab of the Database Detail view:

Column	Description
Owner	The owner of the table.
Table Name	The name of the table.
Index Name	The name of the index.
Cache Name	The cache to which the table has been bound.

TIP: To configure a grid to display row numbers, use the Options Editor.

Metrics

There are a few points worth noting with regard to the information presented in the Index Statistics view:

- Heavily used tables and indexes can benefit from being separated onto different segments that correspond to different physical hard drives.
- The Sybase cost-based optimizer could not be choosing the proper access path for tables that have indexes with out-of-date, or without any, statistics. Keeping dynamic objects current with respect to their internal statistics helps ensure that the Sybase kernel makes the proper decisions on how to satisfy user requests for data.
- Indexes that are frequently scanned can benefit from being bound to a special cache that can be used to retain their often-accessed data in memory.

Troubleshooting

To change the segment a table or index belongs to, you can use the sp_placeobject procedure. Keeping index statistics current can be handled by judicious use of the UPDATE STATISTICS command. Binding frequently scanned indexes can be accomplished via the sp_bindcache procedure.

Storage Tab

A database can contain a number of segments you can use to physically separate tables and indexes onto different physical devices. A database can also be bound to a specific data cache in hopes of improving performance through more frequent memory access. The first grid displays segment use by a database. The second grid displays cache use by a database.

The table below describes the information available in the Segment Detail grid on the Storage tab of the Database Detail view:

Column	Description
Segment Name	The name of the database segment.
Total (KB)	The total amount of space allocated to the segment, in KB. Note that this normally corresponds to the device allocations for a database.
Used (KB)	The amount of space used by the segment to hold tables and indexes, in KB. Note that space used by other segments that make use of the same device are not included in the total.

NOTE: The information in the Segment Detail grid is also available on the Segment Detail tab of the Space Detail view.

The table below describes the information available in the Cache Bindings grid on the Storage tab of the Database Detail view:

Column	Description
Cache Name	The name of the data cache.
Status	The status of the cache (Active, Activation Pending, Deletion Pending).
Configured Size (MB)	The defined size of the cache, in MB.
Running Size (MB)	The actual running size of the cache, in MB.
Overhead (MB)	The amount of overhead required to manage the cache, in MB.

TIP: To configure a grid to display row numbers, use the Options Editor.

Metrics

None.

Options Tab

- Metrics

A database can have a number of options set to determine behavior like transaction log usage, database access, etc. The Options tab of the Database Detail view presents all available database options for a selected database along with which options are in use. The table below describes the information available on the Options tab of the Database Detail view:

Column	Description
Option	The name of the database option.
Setting	Indicates whether the option is in use.

Metrics

Depending on how it is used, a database can have many or few options set. For example, a production database that needs near point-in-time recovery abilities would not have the truncate log on checkpoint option set, because doing so would negate the use of transaction log dumps. However a development database that is loaded/reloaded many times a day with test data would likely make use of the option to prevent the transaction log from filling up. You should examine the specific needs of each database under your care to determine which options are right for it.

Contention Statistics - Sybase

The Contention performance category view displays the following vital Sybase contention statistics:

- Logical Lock Contention
- Address Lock Contention
- Group Commit Sleeps
- Modify Conflicts
- Device I/O Contention
- Disk I/O Structures

- Server Configuration Limit
- Engine Configuration Limit
- Operating System Limit
- Lock Contention
- Blocked
- Current Locks
- Deadlock Rate
- Deadlocks
- Network Contention Rate
- Network Requests

Logical Lock Contention

- Metrics
- Troubleshooting

A single blocking user has the potential to stop work for nearly all processes on a small system, and can cause major headaches on large systems. Although Sybase supports flexible locking mechanisms, blocking lock situations do crop up. Blocks are most often caused by user processes holding exclusive locks and not releasing them via a proper COMMIT frequency.

The Logical Lock Contention statistic reflects the number of times a task/process was switched out because of contention over table/page locks.

Metrics

Seeing consistent high numbers for this statistic should result in you investigating the lock details of your server before the situation has a chance to mushroom.

You can easily drill down and discover the exact process(es) holding locks that are blocking out other user activity.

Troubleshooting

Once you discover a blocking lock situation, you can normally remedy it by issuing a KILL against the offending process. This eliminates the user's stranglehold on the objects the user was accessing, and usually results in other user processes completing in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Center, but preventing the blocking lock situation in the first place is where it gets tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do. You can also check to see if your queries are doing deferred and direct expensive updates, which can cause additional index locks.

By default, all processes wait indefinitely for locks in Sybase. You can change this behavior by modifying the lock wait period configuration parameter, which limits the number of seconds that a process waits for a lock before timing out.

Address Lock Contention

This statistic reflects the number of times a task/process was switched out because of memory address lock problems. Sybase acquires address locks during update operations, and for index pages, OAM pages, allocation pages, and occasionally on data pages when page splits are performed.

Metrics

None.

Group Commit Sleeps

For databases needing high throughput, a large log I/O size is very important to prevent disk queuing problems on the transaction log. Group commit sleeps reflect the number of times a task performed a transactional commit operation and was put to sleep by the server until data in the log was written to disk.

Metrics

You should examine group commit sleeps in conjunction with the number of committed transactions (found on the Users performance category view).

Troubleshooting

A high count for group commit sleeps is not necessarily a problem if the server's transaction rate is low. If there are a significant number of transactions that result in group commit sleeps, and the log I/O size is greater than 2 KB, a smaller log I/O size can help to reduce commit time by causing more frequent page flushes.

Other factors that can effect group commit sleeps are the size of the server run queue and the speed of the hard disk where the log is located.

Modify Conflicts

For some database operations, Sybase uses a specialized protection mechanism to get exclusive access to a page without using actual restrictive page locks. Examples include accessing certain system tables and dirty reads on data pages. These actions need exclusive access to the page in question, even though they do not actually modify it. Modify conflicts record how many times these actions conflicted with other processes needing true database locks.

Metrics

None.

Device I/O Contention

Device I/O Contention reflects the number of times a task or process was put to sleep while waiting for a semaphore for a particular database device.

When a task or process involves physical I/O, Sybase first fills out the block I/O structure and links it to a per engine I/O queue. If two or more Sybase engines request an I/O structure from the same device at the exact same time, the server puts one of them to sleep where it waits for the semaphore it needs.

Metrics

An I/O contention percentage consistently greater than 10% should be cause for concern.

Troubleshooting

Distributing the accessed objects across multiple devices can lessen contention for I/O device semaphores. You can also place devices on different physical drives to lessen contention at the operating system level as well.

Disk I/O Structures

The user community can become very dissatisfied when a Sybase server begins to experience I/O delays. When such I/O delays occur, you should investigate the various Sybase or operating system limits. It could be that I/O operations are being blocked by one or both.

The Disk I/O Structures statistic represents the number of I/O delays caused by Sybase when it reaches the disk I/O structures limit. When Sybase exceeds the number of available disk I/O control blocks, I/O is deferred. This is because Sybase requires that any task get a disk I/O control block before it begins an I/O request.

Metrics

You should be concerned if you consistently see numbers above zero.

Troubleshooting

If you continue to see nonzero numbers for this statistic, you can try increasing the number of available disk I/O control blocks. Do this by increasing the configuration parameter disk I/O structures.

Server Configuration Limit

When a Sybase server begins to experience I/O delays, the result can be a very dissatisfied user community. When such problems begin to occur, things to investigate include the various Sybase or operating system limits. It could be that I/O operations are being blocked by one or both.

The Server Configuration Limit statistic shows nonzero numbers if Sybase has exceeded its limit for the number of asynchronous disk I/O requests that can be outstanding for the entire server at one time.

Metrics

You should be concerned if you consistently see numbers above zero.

Troubleshooting

If you continue to see nonzero numbers for this statistic, you can raise this limit using sp_configure with the max async I/Os per server parameter.

Engine Configuration Limit

When a Sybase server begins to experience I/O delays, the result can be a very dissatisfied user community. When such problems begin to occur, things to investigate include the various Sybase or operating system limits. It could be that I/O operations are being blocked by one or both.

The Engine Configuration Limit statistic shows nonzero numbers if Sybase has exceeded its limit for the number of asynchronous disk I/O requests that can be outstanding for a Sybase engine at one time.

Metrics

Consistent numbers above zero should be a cause for concern.

Troubleshooting

If you continue to see nonzero numbers for this statistic, you can raise this limit using sp_configure with the max async I/Os per engine parameter.

Operating System Limit

The user community can become very dissatisfied when a Sybase server begins to experience I/O delays. When such problems begin to occur, things to investigate include the Sybase or operating system limits. It could be that I/O operations are being blocked by one or both.

The Operating System Limit statistic shows nonzero numbers if Sybase has detected that the limit for asynchronous I/Os has been exceeded.

Metrics

You should be concerned if you consistently see numbers greater than zero.

Troubleshooting

In most UNIX operating systems, there is a kernel parameter that limits the number of asynchronous I/Os that can occur at one time. If you continue to see nonzero numbers for this statistic, you should look into raising this limit.

Deadlock Rate

A deadlock occurs when two processes have a lock on a separate page or object and each wants to acquire a lock on the other process' page or object. Each waits for the other to release the necessary lock. Sybase constantly checks for deadlocks and, when found, chooses the transaction that has accumulated the least amount of CPU time and terminates the transaction. The server then rolls back that transaction and issues a notification of the event. The other process gets to move forward.

The deadlock rate displays the percentage of times deadlocks occurred vs. the percentage of locks requested and immediately granted.

Metrics

You should immediately investigate a percentage much above zero to prevent the situation from mushrooming. You can easily drill down and discover the exact process(es) holding locks and deadlocks that are blocking out other user activity.

Troubleshooting

Well-designed applications can minimize deadlocks by always acquiring locks in the same order. You should always do updates to multiple tables in the same order.

Once Sybase discovers a deadlock, it takes action and remedies the situation. Embarcadero Performance Center makes it easier to discover how prevalent deadlock activity is on a system; preventing deadlocks from occurring in the first place is more difficult.

Those responsible for writing systems can minimize deadlocks by ensuring that applications acquire locks in the same order. Likewise, you should always do updates and other DML that act on multiple tables in the same order.

You can also shrink the amount of time that Sybase waits to check for deadlocks by modifying the deadlock checking period configuration parameter.

Deadlocks

A deadlock occurs when two processes have a lock on a separate page or object and each wants to acquire a lock on the other process' page or object. Each waits for the other to release the necessary lock. Sybase constantly checks for deadlocks and, when found, chooses the transaction that has accumulated the least amount of CPU time and terminates it (the transaction). The server then rolls back that transaction and issues a notification of the event. The other process gets to move forward.

The deadlock statistic displays the number of current deadlocks in a Sybase server.

Metrics

A nonzero number should clue you into the fact that application conflicts are likely being experienced by your user community.

Troubleshooting

Well-designed applications can minimize deadlocks by always acquiring locks in the same order. Updates to multiple tables should always be performed in the same order.

Once Sybase discovers a deadlock, it takes action and remedies the situation. Discovering how prevalent deadlock activity is on a system is made easier by using tools like Performance Center, but preventing deadlocks from occurring in the first place is more difficult.

Those responsible for writing systems can minimize deadlocks by ensuring that applications acquire locks in the same order. Likewise, updates and other DML that act on multiple tables should always be performed in the same order.

You can also shrink the amount of time that Sybase waits before performing a deadlock check by reducing the deadlock checking period parameter.

Current Locks

- Troubleshooting

There are varieties of operations in Sybase that require the use of locks. The Current Locks statistic represents the number of total locks currently active in Sybase.

TIP: Click this statistic to drill down to the All Locks tab of the Locks view.

NOTE: This statistic is also available on the Users performance category view.

Metrics

The only thing to watch with respect to locks is if the number approaches the Sybase limit for available locks.

Troubleshooting

If the number of current locks in a Sybase server approaches the Sybase limit for available locks, you can look into increasing the Number of Locks configuration parameter.

Lock Contention

- Metrics
- Troubleshooting

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems. Although Sybase supports flexible locking mechanisms, blocking lock situations do crop up. Blocks are most often caused by user processes holding exclusive locks and not releasing them via a proper COMMIT frequency.

The blocking lock rate displays the percentage of times blocks occurred vs. the percentage of locks requested and immediately granted.

Metrics

To prevent a mushrooming situation, you should immediately investigate any percentages greater than zero for this statistic.

You can easily drill down and discover the exact process(es) holding locks that are blocking out other user activity.

Troubleshooting

Once you discover a blocking lock situation, you can normally remedy it by issuing a KILL against the offending process. This eliminates the user's stranglehold on the objects the user was accessing, and usually results in other user processes completing in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in Sybase. You can change this behavior by modifying the lock wait period configuration parameter, which limits the number of seconds that a process waits for a lock before timing out.

Blocked

- Metrics

- Troubleshooting

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches even on large systems. Although Sybase supports flexible locking mechanisms, blocking lock situations do crop up. Blocks are most often caused by user processes holding exclusive locks and not releasing them via a proper COMMIT frequency.

The Blocked statistic displays the number of current processes blocked by other processes.

TIP: Click this statistic to drill down to the Blocking Locks tab of the Locks view.

NOTE: This statistic is also available on the Users performance category view.

Metrics

While the Lock Contention rate is a better measure of the overall lock contention situation, consistently seeing positive numbers for the Blocked statistic should also clue you into the fact that a bottleneck exists for some processes. You can easily drill down and discover the exact process(es) holding locks that are blocking other user activity.

Troubleshooting

Once you discover a blocking lock situation, you can normally remedy it by issuing a KILL against the offending process. This eliminates the user's stranglehold on the objects the user was accessing, and usually results in other user processes completing in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in Sybase. You can change this behavior by modifying the lock wait period configuration parameter, which limits the number of seconds that a process waits for a lock before timing out.

Network Contention Rate

- Metrics
- Troubleshooting

The Sybase server normally sends and receives network packets at a regular rate. If the network begins to become saturated, Sybase could experience delays in network I/O. The Network Contention rate indicates the percentage of times network I/O activity was delayed. The following network contention rate statistics are available on the Contention performance category view:

- Network Delays
- Network Requests

Metrics

Seeing a network contention rate greater than 1% could indicate a challenged network structure.

Troubleshooting

Outside of ensuring that the existing network is sufficient for handling the current load (database and non-database), other items to look at from a Sybase specific standpoint include validating that only the necessary amount of data is being sent and returned to requesting users. You can also examine the default packet size and see if it is too small for the average packet size being sent/received by the Sybase server.

Network Delays

- Metrics
- Troubleshooting

The Sybase server normally sends and receives network packets at a regular rate. If the network begins to become saturated, Sybase could experience delays in network I/O. The Network Delays statistic reflects the number of times network I/O activity was delayed.

> **NOTE:** This statistic displays on both the Network performance category view and the Contention performance category view.

Metrics

Seeing a network contention rate greater than 1% could indicate a challenged network structure.

Troubleshooting

Beyond ensuring that the existing network is sufficient for handling the current load (database and nondatabase), other items to look at from a Sybase-specific standpoint include validating that only the necessary amount of data is being sent and returned to requesting users. You can also examine the default packet size and see if it is too small for the average packet size being sent/received by the Sybase server.

Network Requests

Network Requests represents the total TDS packets received and sent by Sybase.

> **NOTE:** This statistic displays on both the Network performance category view and the Contention performance category view.

Contention Detail View

The following tabbed pages are available on the Contention Detail view:

- All Locks
- Blocking Locks

All Locks Tab

- Metrics
- Troubleshooting

Data integrity is maintained in a database through the use of locks. There are many variations of locks that can be applied to data objects, with some being very restrictive. The All Locks tab of the Locks view presents granular information relating to locks that currently exist on the Sybase server. The table below describes the information available on the All Locks tab of the Locks view:

Column	Description
PID	The process of the view.
User Name	The login name of the user.
Database	The database that contains the locks.
Lock Type	The type of lock being applied.
Object Name	The object name being locked.
Status	The status of the transaction.
Lock Page	The page number where the lock is currently applied.
Lock Class	The name of the cursor the lock is associated with (if any).
Host	The name of the host computer.
Program	The program the process is running.
Command	The command being issued by the process.
CPU Time	The amount of CPU time accumulated for the current command.
Physical I/O	The amount of physical I/O accumulated for the current command.
Mem Usage	The amount of memory accumulated for the current command.
FID	The process ID of the worker process' parent.
Transaction	The name of the associated transaction (if any).

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

The main thing to watch for with respect to locks is that all DML locks currently held on the system do not approach the number of locks limit specified in the Sybase server's configuration. The parameter number of locks limits how many locks can exist on the system at one time.

Troubleshooting

If the total number of locks approaches the number of locks limit, you should:

- Ensure that user processes are efficiently using locks and are committing frequently to avoid excessive lock hold times before editing the current Sybase configuration.
- Edit the configuration file for the Sybase server.
- Increase the amount of number of locks to a higher value.
- Cycle the Sybase server when possible to allow the new value to take effect.

Blocking Locks Tab

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems. Although Sybase supports flexible locking mechanisms, blocking lock situations crop up. User processes that hold exclude locks without releasing them via a proper COMMIT frequency most often cause blocks.

The Blocking Locks tab of the Locks view shows granular data concerning current blocking lock scenarios. The table below describes the information available on the Blocking Locks tab of the Locks view:

Column	Description
Holding PID	The process ID of the user holding the blocking lock.
Holding User	The login name of the user holding the blocking lock.
Waiting PID	The process ID of the user waiting for a lock.
Waiting User	The login name of the user waiting for a lock.
Database	The database that contains the locks.
Status	The status of the transaction.
Lock Type	The type of lock being applied.
Time Blocked	The amount of time that a process has been waiting for the lock, in seconds.
Lock Page	The page number where the lock is currently applied.
Lock Class	The name of the cursor the lock is associated with (if any).
Object Name	The object name being locked.
Holding Host	The name of the host computer with the blocking lock.
Waiting Host	The name of the host computer waiting for the lock.
Holding Program	The program the process is running that has the lock.
Waiting Program	The program the process is running that is waiting for the lock.
Holding Command	The command being issued by the process holding the lock.
Waiting Command	The command being issued by the process waiting for the lock.
CPU Time	The amount of CPU time accumulated for the current command.
Physical I/O	The amount of physical I/O accumulated for the current command.
Mem Usage	The amount of memory accumulated for the current command.
Holding FID	The process ID of the worker process' parent that has the lock.
Waiting FID	The process ID of the worker process' parent that is waiting for the lock.
Transaction	The name of the associated transaction (if any).

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

To prevent a mushrooming situation, you should investigate blocking locks that persist for noticeable periods of time, especially if the object(s) in question are hot objects for a database.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects he or she was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in Sybase. You can change this behavior by modifying the lock wait period configuration parameter, which limits the number of seconds that a process waits for a lock before timing out.

Users Statistics - Sybase

The Users performance category view displays the following vital Sybase user statistics:

- Active Connections
- Current Locks
- Inactive Connections
- Rows Deleted
- Rows Inserted
- Rows Updated
- Row Updated (Data Locks Only)
- Session Leaders - CPU
- Session Leaders - I/O
- Session Leaders - Memory
- Total Rows Affected
- Committed Transactions
- ULC Flushes
- Users Blocked
- Lock Promotions

Lock Promotions

- Metrics
- Troubleshooting

The Lock Promotions value represents the average number of lock promotion types combined per second and per transaction.

Metrics

Lock promotions occur when the following escalations take place:

- "Ex-Page to Ex-Table" - Exclusive page to exclusive table.
- "Sh-Page to Sh-Table" - Shared page to shared table.
- "Ex-Row to Ex-Table" - Exclusive row to exclusive table.

- "Sh-R to Sh-Table - Shared row to shared table.
- "Sh-Next-Key to Sh-Table" - Shared next-key to shared table.

Troubleshooting

If lock contention is high and lock promotion is frequent, you should consider changing the lock promotion thresholds for the tables involved. You can configure the lock promotion threshold either server-wide or for individual tables.

Active Connections

The Active Connections statistic represents the total number of active and open threads reported in the Sybase server. This number displays the number of processes actively performing work.

TIP: Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

None.

Inactive Connections

- Metrics

The Inactive Connections statistic represents the total number of Sybase threads that are not actively running on one of the server engines.

TIP: Click this statistic to drill down to the Processes tab of the Users Detail view.

Metrics

Seeing a large number of inactive connections is not generally a cause for concern. If the threads are in a sleeping state (waiting for a resource or performing large volumes of disk I/O), further investigation can be warranted.

Current Locks

- Metrics
- Troubleshooting

The Current Lock statistic displays the total number of locks obtained/requested by processes in the database.

TIP: Click this statistic to drill down to the All Locks tab of the Locks view.

NOTE: This statistic is also available on the Contention performance category view.

Metrics

The main thing to watch with respect to current locks is that all DML locks currently held on the system do not approach the Number of Locks limit specified in the Sybase server's configuration. The parameter for number of locks limits how many locks can exist on the system at once.

Troubleshooting

If the total number of locks approaches the number of locks limit, you should:

1. Ensure that user processes are efficiently using locks and are committing frequently to avoid excessive lock hold times before editing the current Sybase configuration.
2. Edit the configuration file for the Sybase server.
3. Increase the number of locks to a higher value.
4. Cycle the Sybase server when possible to allow the new value to take effect.

Users Blocked

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems. Although Sybase supports flexible locking mechanisms, blocking lock situations do crop up. Blocks are most often caused by user processes holding exclusive locks and not releasing them via a proper COMMIT frequency.

The blocked statistic displays the number of current processes blocked by other processes.

NOTE: Click this statistic to drill down to the Blocking Locks tab of the Locks view.

NOTE: This statistic is also available on the Contention performance category view.

Metrics

While the lock contention rate is a better measure of the overall lock contention situation, consistently seeing positive numbers for the blocked statistic should clue you into the fact that there is a bottleneck for some processes. You can easily drill down and discover the exact process(es) holding locks that are blocking out other user activity.

Troubleshooting

Once you discover a blocking lock situation, you can normally remedy it by issuing a KILL against the offending process. This eliminates the user's stranglehold on the objects the user was accessing, and usually results in other user processes completing in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in Sybase. You can change this behavior by modifying the lock wait period configuration parameter, which limits the number of seconds that a process waits for a lock before timing out.

Committed Transactions

Committed Transactions gives the number of transactions committed since the last refresh inside Performance Center. This includes transactions that meet explicit, implicit, and ANSI definitions for committed transactions. Note that multidatabase transactions are counted.

> **TIP:** Click this statistic to drill down to the Transactions tab of the Users Detail view.

Metrics

Multidatabase transactions generally incur more overhead than single database transactions (for example, a transaction that modifies two databases is counted as two transactions). They usually involve more log activity and two-phase commits between the different databases, as well as cause more ULC flushes. You can improve performance by reducing the number of multidatabase transactions.

ULC Flushes

- Metrics

The total number of times that user log caches (ULCs) were flushed to a transaction log during the sample interval.

Metrics

None.

Rows Inserted

- Metrics
- Troubleshooting

The Rows Inserted statistic reflects the number of row inserts to heap and clustered tables. Note that it does not include the number of fast bulk copy inserts, because these are written directly to the data pages themselves.

> **TIP:** Click this statistic to drill down to the DML Detail tab of the Users Detail view.

Metrics

Large numbers of inserts are no cause for concern. The only thing to keep an eye on is that heavy insert activity on heap tables can potentially cause lock contention. It is easy to investigate lock contention in the Embarcadero Performance Center's Locks view.

Troubleshooting

If insert activity seems to be causing lock contention, you can consider creating a clustered index that randomizes insert activity on the tables in question. Alternatives are establishing partitions on an unpartitioned table or increasing the number of partitions on a partitioned table.

Keep in mind that an unwanted by-product of clustered indexes is occasional page splitting.

Rows Updated

- Metrics

The Rows Updated statistic represents all deferred and direct update activity.

> **TIP:** Click this statistic to drill down to the DML Detail tab of the Users Detail view.

Metrics

None.

Rows Updated (Data Only Locks)

- Metrics

The Rows Updated (Data Only Locks) statistic represents all deferred and direct update activity.

> **TIP:** Click this statistic to drill down to the DML Detail tab of the Users Detail view.

Metrics

None.

Rows Deleted

- Metrics

The Rows Deleted statistic represents all deferred and direct delete activity.

> **TIP:** Click this statistic to drill down to the DML Detail tab of the Users Detail view.

Metrics

None.

Total Rows Affected

- Metrics

The Total Rows Affected statistic represents all rows impacted by some form of DML activity (inserts, updates, deletes).

> **TIP:** Click this statistic to drill down to the DML Detail tab of the Users Detail view.

Metrics

None.

Session Leaders - Memory

- Metrics
- Troubleshooting

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problems could be runaway processes, untuned batch procedures, or user-initiated operations. Often, user connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system levels.

NOTE: This statistic displays on both the Memory performance category view and the Users performance category view.

TIP: Click this statistic to drill down to the Leading Sessions tab of the Memory Detail view.

Metrics

If your Sybase server does not have an overabundance of memory, you should check periodically to see who your heavy memory users are. You should also check the total percentage of memory each takes up. If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing.

Troubleshooting

You can use the Session Leaders - Memory statistic to find the users with the greatest current allocations of overall memory. Runaway processes can be immediately terminated from within the Embarcadero Performance Center Client.

Session Leaders - I/O

- Metrics

When a system undergoes heavy I/O activity, sometimes you find that all user connections are contributing equally to the overall load. More often than not, however, one or two user connections are responsible for 75% or more of the I/O activity. It can be that a large batch load or other typical process is running that is perfectly okay for your system. Alternatively, it can be a runaway process or other rogue connection that needs to be tracked down and possibly eliminated.

The leading I/O session's display allows you to see who runs the leading sessions in your system with respect to I/O.

NOTE: This statistic displays on both the I/O performance category view and the Users performance category view.

TIP: Click this statistic to drill down to the Leading Sessions tab of the I/O Detail view.

Metrics

Finding one or two users who consume more than 75% of the total I/O load can indicate runaway or improper processes. By drilling down into the I/O activity of all users, you can quickly see if this is the case.

Session Leaders - CPU

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Often, user connections can get out of hand with CPU use, and extreme cases have caused headaches at both the database and operating system levels.

The leading CPU session's display allows you to see who runs the leading sessions are in your system with respect to CPU usage.

Metrics

Finding one or two users who use the majority of the CPU can indicate runaway or improper processes. By drilling down into the CPU activity of all users, you can quickly see if this is the case.

Users Detail View

The following tabbed pages are available on the Users Detail view:

- Applications
- Blocking Locks
- DML Detail
- Locks
- Processes
- Transactions

Top Memory Processes

- Metrics

The Top Memory Process statistic identifies the Sybase process that currently is using the highest percentage of memory in the database.

Metrics

None.

Applications Tab

Different programs/applications can access the Sybase server at the same time, with some being more resource-intensive than others. The Applications Detail tab on the Users Detail view summarizes resource consumption at the application level. The table below describes the information available on the Applications tab of the Users Detail view:

Column	Description
Application	The application name. **NOTE**: This can be blank if Sybase cannot determine the application accessing the server.
Application Count	The total number of like applications accessing the server.
Active	The count of active applications accessing the server.
Inactive	The count of inactive applications accessing the server.
Memory Used	The total amount of memory used by each application's current command.
CPU Used	The total amount of CPU used by each application's current command.
Physical I/O	The total amount of Physical I/O used by each application's current command.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

None.

Processes Tab

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Often, user connections can get out of hand with memory consumption, CPU, or physical I/O, and extreme cases have caused headaches at both the database and operating system levels.

The Processes tab of the Users Detail view displays information to help pinpoint problem processes. The table below describes the information available on the Processes tab of the Users Detail view:

Column	Description
User Name	The logon name the session is using.
PID	The process ID.
FID	The process ID of the worker process' parent.
Database	The database the process is attached to.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The client machine name the session is using.
Program	The executable the process is using against the Sybase server.
Memory	The amount of memory currently allocated to the process.
CPU	The amount of CPU currently in use by the process.
Physical I/O	The amount of physical I/O currently accumulated by the process.
Blocked	Whether the process is blocked by another process.
Command	The command the process is currently issuing.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

The key to spotting problem processes is to view their information in light of the total activity on a server. For example, if your database server does not have an overabundance of memory, you should periodically check to see who your heavy memory users are. You should also check the total percentage of memory each takes up. If you see one or two users who have more than 5-15% of the total memory usage, you should investigate the sessions further to see what activities they are performing and take action if necessary.

Locks Tab

- Metrics
- Troubleshooting

Data integrity is maintained in a database through the use of locks. There are many variations of locks that can be applied to data objects, with some being very restrictive. The Locks tab of the Users Detail view presents granular information relating to locks that currently exist on the Sybase server. The table below describes the information available on the Locks tab of the Users Detail view:

Column	Description
PID	The process of the user.
User Name	The login name of the user.
Database	The database that contains the locks.
Lock Type	The type of lock being applied.
Object Name	The object name being locked.
Status	The status of the transaction.
Lock Page	The page number where the lock is currently applied.
Lock Class	The name of the cursor the lock is associated with (if any).
Host	The name of the host computer.
Program	The program the process is running.
Command	The command being issued by the process.
CPU Time	The amount of CPU time accumulated for the current command.
Physical I/O	The amount of physical I/O accumulated for the current command.
Mem Usage	The amount of memory accumulated for the current command.
FID	The process ID of the worker process' parent.
Transaction	The name of the associated transaction (if any).

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

The main thing to watch with respect to current locks is that all DML locks currently held on the system do not approach the number of locks limit specified in the Sybase server's configuration. The parameter number of locks limits how many locks can exist on the system at once.

Troubleshooting

If the total number of locks approaches the number of locks limit, then:

- Before editing the current Sybase configuration, ensure that user processes are efficiently using locks and are committing frequently to avoid excessive lock hold times.
- Edit the configuration file for the Sybase server.
- Increase the amount of number of locks to a higher value.
- Cycle the Sybase server when possible to allow the new value to take effect.

Blocking Locks Tab

A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches even on large systems. Although Sybase supports flexible locking mechanisms, blocking lock situations do crop up. User processes holding exclusive locks and not releasing them via a proper COMMIT frequency most often cause blocks.

The Blocking Locks tab of the Users Detail view shows granular data concerning current blocking lock scenarios. The table below describes the information available on the Blocking Locks tab of the Users Detail view:

Column	Description
Holding PID	The process ID of the user holding the blocking lock.
Holding User	The login name of the user holding the blocking lock.
Waiting PID	The process ID of the user waiting for a lock.
Waiting User	The login name of the user waiting for a lock.
Database	The database that contains the locks.
Object Name	The object name being locked.
Status	The status of the transaction.
Lock Type	The type of lock being applied.
Time Blocked	The amount of time that a process has been waiting for the lock, in seconds.
Lock Page	The page number where the lock is currently applied.
Lock Class	The name of the cursor the lock is associated with (if any).
Holding Host	The name of the host computer with the blocking lock.
Waiting Host	The name of the host computer waiting for the lock.
Holding Program	The program the process is running that has the lock.
Waiting Program	The program the process is running that is waiting for the lock.
Holding Command	The command being issued by the process holding the lock.
Waiting Command	The command being issued by the process waiting for the lock.
CPU Time	The amount of CPU time accumulated for the current command.
Physical I/O	The amount of physical I/O accumulated for the current command.
Mem Usage	The amount of memory accumulated for the current command.
Holding FID	The process ID of the worker process' parent that has the lock.
Waiting FID	The process ID of the worker process' parent that is waiting for the lock.
Transaction	The name of the associated transaction (if any).

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

To prevent a mushrooming situation, you should investigate blocking locks that persist for noticeable periods of time should, especially if the object(s) in question are hot objects for a database.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects he or she was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Performance Center, but preventing the blocking lock situation in the first place is where it gets tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in Sybase. You can change this behavior by modifying the lock wait period configuration parameter, which limits the number of seconds that a process waits for a lock before timing out.

DML Detail Tab

- Metrics

Dynamic Sybase environments typically have large amounts of DML (data manipulation language) activity that occurs. Knowing the volume of INSERT, UPDATE, and DELETE activity can help you make correct decisions regarding sizing, object options, etc. The DML Detail tab of the Users Detail view provides a quick snapshot of the amount of object/data manipulation that is occurring. The table below describes details of the DML Detail tab of the Users Detail view:

Detail	Description
Activity Type	The type of INSERT, UPDATE, or DELETE.
Amount	The number of operations that have taken place since the last refresh.

The table below describes the statistics available on the DML Detail tab of the Users Detail view:

Statistic	Description
Heap Table Inserts	Total rows inserted into heap tables during sample interval.
Clustered Table Inserts	Total rows inserted into clustered tables during sample interval.
Deferred Updates	Total number of updates that were accomplished in two steps during sample interval.
Direct In Place Updates	Total number of in-place updates (no data rows are moved on the data page) during sample interval.
Direct Cheap Updates	Total number of on-page updates (the length of the data row changes) during sample interval.
Direct Expensive Updates	Total number of delete/insert direct updates (rows are deleted from their original location and inserted at a new location) during sample interval.
Deferred Deletes	Total rows deleted during the sample interval.
Direct Deletes	Total rows directly deleted during the sample interval.

Metrics

Response times can differ greatly for various types of DML activity. For example, deferred UPDATE and DELETE operations (which are often caused by join conditions or because referential integrity conditions exist) normally perform slower than direct UPDATE and DELETEs. Deferred operations can involve rereading the transaction log to finalize the operation whereas direct operations have no such need. Another example would be that a direct expensive update involves additional index locks (and is therefore more restrictive), while direct cheap updates do not.

Transactions Tab

- Metrics

Transactions are used in Sybase to preserve data integrity and keep logical units of work together. Long-running transactions, however, run the risk of holding data and object locks longer than necessary and possibly degrading overall system response times. The Transactions tab of the Users Detail view shows all running transactions in a database and their current state. The table below describes the information available on the Transactions tab of the Users Detail view:

Column	Description
SPID	The process ID.
Login	The login name of the user initiating the transaction.
Database	The database involving the transaction.
Status	The status of the transaction.
Start Time	The time the transaction began.
Type	The type of transaction (local, remote, etc.)
State	The point where the transaction is (In Command, Committed, etc.)
Command	The current command being issued.
Connection	Whether the transaction is current attached or detached.
CPU	The amount of CPU currently being used by the transaction.
Memory	The amount of memory used by the current transaction.
Physical I/O	The current cumulative number of reads and writes issued by the process.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

You should examine transactions that have been running for excessive periods of time to see if they are being blocked by other transactions. If you find this is the case, Embarcadero Performance Center offers mechanisms that allow you to kill any process that is blocking another. If blocking lock activity is occurring, a small redesign of the application or more frequent COMMIT TRANSACTION points can prove to be useful preventive measures.

Network Statistics - Sybase

The Network performance category view displays the following vital network statistics:

- Total Packets Received
- Total Packets Sent
- Total Bytes Received
- Total Bytes Sent
- Network Errors
- Network Contention Rate
- Network Delays
- Network Requests

Average Bytes Received Per Packet/Average Bytes Sent Per Packet

These statistics represent the average number of bytes received or sent per packet, respectively, by the Sybase Adaptive Server engine during the sample interval.

Metrics

None.

Total Packets Received

- Metrics

Total Packets Received reflects the number of times Sybase received a packet from a client application.

Metrics

None.

Total Packets Sent

- Metrics

Total Packets Sent reflects the number of times Sybase sent a packet to a client application.

Metrics

None.

Total Bytes Received

- Metrics

Total Bytes Received reflects the number of bytes received by Sybase since the last refresh in Embarcadero Performance Center.

Metrics

None.

Total Bytes Sent

- Metrics

Total Bytes Sent reflects the number of bytes sent to Sybase since the last refresh in Embarcadero Performance Center.

Metrics

None.

Network Errors

The Network Errors statistic reflects the number of times that network errors were detected by Sybase while reading and writing packets.

Metrics

Seeing a consistent value much above zero could indicate a challenged network structure.

Troubleshooting

Beyond ensuring that the existing network is sufficient for handling the current load (database and non-database), other items to look at from a Sybase-specific standpoint include validating that only the necessary amount of data is being sent and returned to requesting users. You can also examine the default packet size and see if it is too small for the average packet size being sent/received by the Sybase server.

Network Contention Rate

The Sybase server normally sends and receives network packets at a regular rate. If the network begins to become saturated, Sybase can experience delays in network I/O. The network contention rate indicates the percentage of times network I/O activity was delayed.

Metrics

Seeing a network contention rate greater than 1% could indicate a challenged network structure.

Troubleshooting

Outside of ensuring that the existing network is sufficient for handling the current load (database and non-database), other items to look at from a Sybase-specific standpoint include validating that only the necessary amount of data is being sent and returned to requesting users. You can also examine the default packet size and see if it is too small for the average packet size being sent/received by the Sybase server.

Network Delays

The Sybase server normally sends and receives network packets at a regular rate. If the network begins to become saturated, Sybase can experience delays in network I/O. The Network Delays statistic reflects the number of times network I/O activity was delayed.

Metrics

Seeing a network contention rate greater than 1% could indicate a challenged network structure.

Troubleshooting

Beyond ensuring that the existing network is sufficient for handling the current load (database and non-database), other items to look at from a Sybase-specific standpoint include validating that only the necessary amount of data is being sent and returned to requesting users. You can also examine the default packet size and see if it is too small for the average packet size being sent/received by the Sybase server.

Network Requests

Network Requests represents the total TDS packets received and sent by Sybase.

Metrics

None.

OS Page Statistics

In many scenarios, an optimally tuned database may not perform well because there are constraints imposed by the system where the database is running. These constraints may include processes competing with the database sever for resources (CPU, I/O, or Memory), a slow CPU, insufficient or slow I/O devices, and insufficient memory. The OS Statistics page of Performance Center lets you examine operating system metrics for the following platforms:

- AIX
- HP-UX
- Linux
- Solaris
- Unix
- Windows XP, 2000, and NT

Summary Tab

The OS Summary tab displays the following statistics to communicate the general overall performance levels of the operating system:

- Disk Time
- Load Average
- Processor Time
- Paged Memory Used (Windows)
- Swap Memory Used (AIX, HP-UX, Linux, Solaris, Unix)
- Average Disk Queue
- Network Output Queue (Windows)
- Network Queue (Solaris)
- Page Faults/Sec
- Processor Queue

- Processor Speed
- Processor
- Available Paged Memory (Windows)
- Available Physical Memory
- Available Swap Memory (AIX, HP-UX, Linux, Solaris, Unix)
- Total Paged Memory (Windows)
- Total Physical Memory
- Total Swap Memory (AIX, HP-UX, Linux, Solaris, Unix)
- Free Disk Space
- Total Disk Space
- Used Disk Space
- Number of Logins
- Number of Processes
- Number of Processors
- Top CPU Process
- Top I/O Process
- Top Memory Process

Processor Time

The Processor Time statistic indicates the percentage of time the processor is working. This counter is a primary indicator of processor activity.

Metrics

If your computer seems to be running sluggishly, this statistic could be displaying a high percentage.

Troubleshooting

Upgrade to a processor with a larger L2 cache, a faster processor, or install an additional processor.

Processor Speed

The Processor Speed statistic displays the speed of the active processor in MHz. The speed is approximate.

Processor

The Processor Statistic displays the type of processor currently in use, for example, GenuineIntel.

Disk Time

The Disk Time statistic is the percentage of elapsed time that the selected disk drive/device was busy servicing read or write requests.

Metrics

You should avoid consistently seeing values for this statistic greater then 90%.

Troubleshooting

Add more disk drives and partition the files among all of the drives.

Load Average

The Load Average statistic represents the system load averages over the last 1, 5, and 15 minutes.

Metrics

High load averages usually mean that the system is being used heavily and the response time is correspondingly slow.

Paged Memory Used

The Paged Memory Used statistic is the ratio of Commit Memory Bytes to the Commit Limit. Committed memory is where memory space has been reserved in the paging file if it needs to be written to disk. The commit limit is determined by the size of the paging file. As the paging file increases, so does the commit limit.

NOTE: This statistic is available for the Windows platform.

Metrics

This value displays the current percentage value only and not an average. If the percentage of paged memory used is above 90%, you may be running out of memory.

Troubleshooting

Increase the size of page file.

Number of Processors

This statistic displays the number of processors currently in use.

Swap Memory Used

The Swap Memory Used statistic is the percentage of swap space currently in use.

Metrics

If the percentage of swap memory used is above 90%, you may be running out of memory.

Troubleshooting

Increase the size of your swap files.

Average Disk Queue

The Average Disk Queue statistic is the average number of both read and write requests that were queued for the selected disk during the sample interval.

Metrics

This metric is useful in identifying I/O related bottlenecks. If the disk queue lengths for certain disks are consistently much higher than others, you may need to redistribute the load among available disks. If the disk queues lengths for all disks are consistently large, and you see a high amount of I/O activity, your disks may be inefficient.

Troubleshooting

Some things you can do if you have problems with this statistic include:

- Redistribute the data on the disk with the large average disk queue to other disks.
- Upgrade to faster disk(s).

Page Faults/Sec

The Page Faults/Sec statistic is the overall rate faulted pages are handled by the processor. It is measured in numbers of pages faulted per second. A page fault occurs when a process requires code or data that is not in its working set. This counter includes both hard faults and soft faults.

Metrics

This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

If the number of page faults remains consistently high, you can check with your Windows System Administrator for further investigation. Often, large numbers of page faults are not a problem so long as they are soft faults. However, hard faults, that require disk access, can cause delays.

Processor Queue

The Processor Queue Length statistic is the number of threads in the processor queue.

Metrics

Unlike the disk counters, this counter shows ready threads only, not threads that are running. There is a single queue for processor time even on computers with multiple processors. Therefore, if a computer has multiple processors, you need to divide this value by the number of processors servicing the workload. A sustained processor queue of less than 10 threads per processor is normally acceptable, dependent of the workload.

Troubleshooting

A sustained high value in the Processor Queue could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Network Output Queue/Network Queue

The Network Output Queue Length statistic is the number of threads in the processor queue.

NOTE: The name of this statistic depends on the platform of the operating system.

Metrics

Unlike the disk counters, this counter shows ready threads only, not threads that are running. There is a single queue for processor time even on computers with multiple processors. Therefore, if a computer has multiple processors, you need to divide this value by the number of processors servicing the workload. A sustained processor queue of less than 10 threads per processor is normally acceptable, dependent of the workload.

Troubleshooting

A sustained high value in the Processor Queue Length could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Available Physical Memory

The Available Physical Memory statistic represents the amount of RAM available to all processes.

Metrics

This counter displays the last observed value only and not an average. Use this value with the Total physical memory and paging metrics (Memory details page). If the available physical memory is very small compared to this value, and the paging activity is high, your system may be running low on memory.

Troubleshooting

Some things you can do if you have problems with this statistic include:

- Check the running processes to see if there are any memory leaks.
- Stop any services that are not required.
- Install additional RAM.

Available Paged Memory

The Available Paged Memory statistic shows the amount of virtual memory available for the processes.

NOTE: This statistic is available for the Windows platform.

Metrics

If the available virtual memory is less than 10% of the total virtual memory, your system may run out of memory.

Troubleshooting

Increase the size of page file.

Available Swap Memory

The Available Swap Memory statistic represents the amount of virtual memory available for the processes.

Metrics

If the available Available Swap Memory is less than 10% of the total Swap Memory, your system may run out of memory.

Troubleshooting

Increase the size of swap files.

Total Physical Memory

The Total Physical Memory statistic shows the amount of physical memory installed on your computer.

Metrics

This is an informational metric and displays the total amount installed on the machine. Use this value with the available physical memory and paging metrics (Memory details page). If the available physical memory is very small compared to this value, and the paging activity is high, your system may be running low on memory.

Total Paged Memory/Total Swap Memory

The Total Paged Memory statistic shows the maximum amount of virtual memory available to all processes.

NOTE: The name of this statistic depends on the platform of the operating system.

Metrics

It is recommended that thisvalue is between 1.5 to 3 times the amount of RAM on the system.

Used Disk Space

The Used Disk Space statistic shows the amount of allocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations:

- If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
- You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Total Disk Space

Total Disk Space displays the total allocated and unallocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations, here are two:

1. If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
2. You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Free Disk Space

The Free Disk Space statistic shows the unallocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations:

- If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
- You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Top Memory Process

Top Memory Process shows the current process that is consuming the most amount of memory. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Metrics

If you are running out of memory on the system, this is a quick way to identify the top memory user. If the displayed process is using a significant portion of the total memory, it could be causing the memory issues.

Processes Overview

The Processes Overview of the OS Summary includes the following sections:

- Top CPU Process
- Top I/O Process
- Top Memory Process

Top CPU Process

Top CPU Process shows the current process that is consuming the most amount of CPU. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Metrics

If the amount of CPU time used by this process is close to 100% and the CPU usage is very high, this process may be the bottleneck on the server.

Troubleshooting

Investigate the process further to see if it is in an inconsistent state. Also, look at minimum requirements for CPU speed for the process. You may need to upgrade your CPU.

Top I/O Process

The Top I/O Process statistic shows the current process that is consuming the most amount of CPU. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Number of Logins

This statistic displays the total number of logins on the server.

Number of Processes

This statistic displays the total number of processes on the server.

CPU Tab

The CPU tab of the OS Detail includes the following sections:

- Context Switches/Sec
- CPU Utilization
- Interrupts/Sec
- Processor Queue Length

CPU Utilization

The CPU Utilization section includes the following information:

- % Privileged Time
- % User Time

% Privileged Time

The % Privileged Time statistic is the percentage of elapsed time that the process threads spent executing code in privileged mode.

> **NOTE:** For Windows systems, when a Windows system service is called, the service will often run in privileged mode to gain access to system-private data. Such data is protected from access by threads executing in user mode. Calls to the system can be explicit or implicit, such as page faults or interrupts. These kernel commands, are considered privileged to keep the low-level commands executing and prevent a system freeze. Unlike some early operating systems, Windows uses process boundaries for subsystem protection in addition to the traditional protection of user and privileged modes. Some work done by Windows on behalf of the application might appear in other subsystem processes in addition to the privileged time in the process.

Metrics

The ideal range should be 0-40% (less than 40% indicates excessive system activity).

Troubleshooting

If your CPU consistently runs at less than 40% you may need to upgrade your system to include a faster processor(s).

% User Time

The % User Time statistic is the percentage of elapsed time the processor spends in the user mode. User mode is a restricted processing mode designed for applications, environment subsystems, and integral subsystems. The alternative, privileged mode, is designed for operating system components and allows direct access to hardware and all memory. The operating system switches application threads to privileged mode to access operating system services. This counter displays the average busy time as a percentage of the sample time.

Metrics

If the Privileged Time is high in conjunction with Physical Disk Reads, consider upgrading the disk I/O subsystem.

Interrupts/Sec

The Interrupts/Sec statistic is the average rate, in incidents per second, at which the processor received and serviced hardware interrupts. It does not include deferred procedure calls (DPCs), which are counted separately. This value is an indirect indicator of the activity of devices that generate interrupts, such as the system clock, the mouse, disk drivers, data communication lines, network interface cards, and other peripheral devices. These devices normally interrupt the processor when they have completed a task or require attention. Normal thread execution is suspended. The system clock typically interrupts the processor every ten milliseconds, creating a background of interrupt activity. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Metrics

The ideal rage should be 0-5000. A number greater then 5000 indicates possible excessive hardware interrupts; justification is dependent on device activity.

Context Switches/Sec

The Context Switches/Sec section shows the combined rate at which all processors on the computer are switched from one thread to another. Context switches occur when a running thread voluntarily relinquishes the processor, is preempted by a higher priority ready thread, or switches between user-mode and privileged (kernel) mode to use an Executive or subsystem service.

Metrics

The ideal range should be between 0-10,000. GA number greater then 10,000 may indicate too many threads contending for resources.

Processor Queue Length

The Processor Queue Length statistic is the number of threads in the processor queue. There is a single queue for processor time even on computers with multiple processors.

> **NOTE:** For Windows systems, unlike the disk counters, this counter shows ready threads only, not threads that are running.

Metrics

A sustained high value in the Processor Queue Length could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Processes Tab

The Processes tab of the OS Detail page succinctly communicates the general overall performance levels of processes. The columns available in this table depend on the platform of operating system. The table below describes the information available in the table on this tab:

Column	Description
Process	The name of the process.
User	The user of the process.
ID	The ID Process is the unique identifier of this process. ID Process numbers are reused, so they only identify a process for the lifetime of that process.
CPU	The CPU is the percentage of elapsed time that all of process threads used the processor to execution instructions.
User Mode	The User Mode is the percentage of elapsed time that the process threads spent executing code in user mode.
Memory **WINDOWS ONLY**	Memory is the current size, in bytes, of the virtual address space the process is using. Use of virtual address space does not necessarily imply corresponding use of either disk or main memory pages. Virtual space is finite, and the process can limit its ability to load libraries.
Memory (MB)	Memory is the current size, in bytes, of the virtual address space the process is using. Use of virtual address space does not necessarily imply corresponding use of either disk or main memory pages. Virtual space is finite, and the process can limit its ability to load libraries.
Memory	Memory is the percentage of the memory used of the total memory.
Active Memory	Active Memory is the amount of committed virtual memory, in bytes for this process. Active memory is the physical memory which has space reserved on the disk paging file(s). There can be one or more paging files on each physical drive. This counter displays the last observed value only; it is not an average.
I/O Data	The rate at which the process is reading and writing bytes in I/O operations. This counter counts all I/O activity generated by the process to include file, network and device I/Os.
Elapsed Time	The total elapsed time, in seconds, that this process has been running.
Thread Count	The number of threads currently active in this process. An instruction is the basic unit of execution in a processor, and a thread is the object that executes instructions. Every running process has at least one thread.

Column	Description
Handle Count	The total number of handles currently open by this process. This number is equal to the sum of the handles currently open by each thread in this process.
Priority	The current base priority of this process. Threads within a process can raise and lower their own base priority relative to the process' base priority.
Creating Proc ID	The Creating Process ID value is the Process ID of the process that created the process. The creating process may have terminated, so this value may no longer identify a running process.
Page Faults/Sec	Page Faults/Sec is the rate at which page faults by the threads executing in this process are occurring. A page fault occurs when a thread refers to a virtual memory page that is not in its working set in main memory. This may not cause the page to be fetched from disk if it is on the standby list and hence already in main memory, or if it is in use by another process with whom the page is shared.
Page File	Page File is the current number of kilobytes that this process has used in the paging file(s). Paging files are used to store pages of memory used by the process that are not contained in other files. Paging files are shared by all processes, and the lack of space in paging files can prevent other processes from allocating memory.
Private	Private is the current size, in kilobytes, of memory that this process has allocated that cannot be shared with other processes.

I/O Tab

The table below describes the information available in this section:

Column	Description
Disk	The disk number assignment.
Reading (KB/s)	The amount of bytes read from the device.
Writing (KB/s)	The amount of bytes written to the device.
Disk Read Time	Disk Read Time is the percentage of elapsed time that the selected disk drive was busy servicing read requests.
Disk Write Time	Disk Write Time is the percentage of elapsed time that the selected disk drive was busy servicing write requests.
Disk Time	Disk Time is the percentage of elapsed time that the selected disk was busy servicing requests.
Avg. Read Queue	Avg. Disk Read Queue Length is the average number of read requests that were queued for the selected disk during the sample interval.
Avg. Write Queue	Avg. Disk Write Queue Length is the average number of write requests that were queued for the selected disk during the sample interval.
Disk Reads/Sec	Disk Reads/Sec is the rate of read operations on the disk.
Disk Writes/Sec	Disk Writes/Sec is the rate of write operations on the disk.

Memory Tab

The Memory tab of the OS Detail page includes the following sections:

- Cache Efficiency
- Free Physical

- Free Paged
- Paging Activity
- Page Faults
- Total Physical
- Total Paged

Paging Activity

The Paging Activity section includes the following statistics:

- Pages Input/Sec
- Pages Output/Sec

Pages Input/Sec

The Pages Input/Sec statistic is the number of pages read from disk to resolve hard page faults. Hard page faults occur when a process requires code or data that is not in its working set or elsewhere in physical memory, and must be retrieved from disk.

Metrics

This value was designed as a primary indicator of the kinds of faults that cause system-wide delays. It includes pages retrieved to satisfy faults in the file system cache (usually requested by applications) and in non-cached mapped memory files. This counter counts numbers of pages, and can be compared to other counts of pages, such as Memory: Page Faults/sec, without conversion. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

Although it never hurts to have as much physical memory as your system can handle, there are some things you can check within your system to alleviate the memory bottleneck.

- Check to see if you have any drivers or protocols that are running but not being used. They use space in all memory pools even if they are idle.
- Check to see if you have additional space on your disk drive that you could use to expand the size of your page file. Normally, the bigger the initial size of your page file, the better, in performance terms.

Pages Output/Sec

The Pages Output/Sec statistic is the number of pages written to disk to free up space in physical memory. Pages are written back to disk only if they are changed in physical memory. A high rate of pages output might indicate a memory shortage.

Metrics

Windows NT writes more pages back to disk to free up space when low in physical memory. This counter counts numbers of pages, and can be compared to other counts of pages, without conversion. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

Although it never hurts to have as much physical memory as your system can handle, there are some things you can check within your system to alleviate the memory bottleneck.

- Check to see if you have any drivers or protocols that are running but not being used. They use space in all memory pools even if they are idle.
- Check to see if you have additional space on your disk drive that you could use to expand the size of your page file. Normally, the bigger the initial size of your page file, the better, in performance terms.

Free Physical

The Free Physical statistic is the amount of physical memory that is uncommitted.

Metrics

None.

Free Paged

The Free Paged statistic is the amount of uncommitted virtual memory.

Metrics

None.

Total Physical

The Total Physical statistic is the total physical memory available.

Metrics

None.

Total Paged

The Total Paged statistic is the amount of total virtual memory, in bytes. Used Memory is the physical memory that has space reserved on the disk paging file(s). There can be one or more paging files on each physical drive.

Metrics

None.

Page Faults/Sec

The Page Faults/Sec statistic is the overall rate faulted pages are handled by the processor. It is measured in numbers of pages faulted per second. A page fault occurs when a process requires code or data that is not in its working set. This counter includes both hard faults and soft faults.

Metrics

This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

If the number of page faults remains consistently high, you can check with your Windows System Administrator for further investigation. Often, large numbers of page faults are not a problem so long as they are soft faults. However, hard faults, that require disk access, can cause delays.

Cache Efficiency

The Cache Efficiency section of the Memory tab succinctly communicates the general overall performance levels of the server's memory. The following statistics are available in this section:

- Copy Read Hits%
- Data Map Hits%
- MDL Read Hits%
- Pin Read Hits%

Copy Read Hits %

The Copy Read Hits % statistic is the percentage of cache copy read requests that hit the cache and does not require a disk read to provide access to the page in the cache.

Metrics

When the page is pinned in the memory, the page's physical address in the file system cache will not be altered. A copy read is a file read operation where a page in the cache is copied to the application's buffer. Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate

Data Map Hits %

The Data Map Hits % statistic is the percentage of data maps in the file system cache that could be resolved without having to retrieve a page from the disk.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

MDL Read Hits %

The MDL Read Hits % statistic is the percentage of Memory Descriptor List Read requests to the file system cache that hit the cache and does not require disk access to provide memory access to the pages in the cache.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

Pin Read Hits %

The Pin Read Hits % statistic is the percentage of pin read requests that hit the file system cache and does not require a disk read in order to provide access to the page in the file system cache.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

Space Tab

The Space tab of the OS Detail page includes the following sections:

- Disk Space Free
- Disk Space Detail

Disk Space Free

The Disk Space Free metric displays the amount of free disk space in megabytes.

Metric

None.

Disk Space Detail

The Disk Space Detail section of the Space tab succinctly communicates the general overall performance levels of the server's disks and space allotment. The table below describes the statistics in this section:

Statistic	Description
Partition	The drive letter of the disk.
Total Space	Total size of the disk/device's capacity expressed in MBs.
Used Space	Amount of MBs currently allocated on the particular disk/device.
Free Space	Amount of MBs currently unallocated and free on the particular disk/device.

Network Tab

The Network tab of the OS Detail page succinctly communicates the general overall performance levels of the server's networking. The statistics available in this section depend on the platform of operating system. The table below describes the information available in this section:

Column	Description
Network Interface	The name of network interface.
INET Address/Address	The IP address assigned to the network interface.

Column	Description
Pkts Sent/Sec	The number of packets sent per second.
Pkts Received/Sec	The number of packets received per second.
Sent (KB/Sec)	The number of bytes sent per second.
Received (KB/Sec)	The number of bytes received per second.
Out Pkts Discarded	The number of outbound packets discarded.
In Pkts Discarded	The number of inbound packets discarded.
Out Pkt Errors	The number of outbound packet errors.
In Pkt Errors	The number of inbound packet errors.
Queue Length	The queue length.
Collisions	The number of collisions.
Packets Discarded	The number of packets discarded.

DB2 Expert Guide

This section includes expert help for all DB2 UDB LUW categories and statistics in the Embarcadero Performance Center views and pages. For detailed information on using the application, see Using Embarcadero Performance Center. This help is divided into the following sections:

- Home View Statistics
- Memory Page Statistics
- Object Page Statistics
- Contention Page Statistics
- I/O Page Statistics
- Users Page Statistics
- Space Page Statistics
- Instance Page Statistics

Home View Statistics

The Embarcadero Performance Center Home view lets you review availability and overall performance of all monitored databases from a single window. Statistics on the Home view are organized into the following categories:

- Memory Vital Signs
- Contention Vital Signs
- I/O Vital Signs
- Users Vital Signs
- Space Vital Signs
- Instance Vital Signs

Memory Vital Signs

The following memory statistics are on the DB2 Home view:

- Buffer Pool Hit Ratio
- Catalog Cache Hit Ratio
- Package Cache Overflows
- Shared Workspace Hit Ratio

NOTE: For the complete set of available memory statistics, see Memory Page Statistics.

Contention Vital Signs

The following contention statistics are on the DB2 Home view:

- Deadlocks
- Lock Timeouts
- Lock Escalations
- Applications Waiting on Locks

NOTE: For the complete set of available Contention statistics, see Contention Page Statistics.

I/O Vital Signs

The following I/O statistics are located on the DB2 Home view:

- Logical Read Ratio
- Physical Read Ratio
- Synchronous Read Ratio
- Asynchronous Read Ratio

NOTE: For the complete set of available I/O statistics, see I/O Page Statistics.

Users Vital Signs

The following users statistics are located on the DB2 Home view:

- Connections
- Max. Connections
- Percent Executing
- Percent of Maximum

NOTE: For the complete set of available Users statistics, see Users Page Statistics.

Space Vital Signs

The following space statistics are located on the DB2 Home view:

- DMS Utilization
- SMS Utilization
- Active Log Utilization
- Tablespaces Low on Space

NOTE: For the complete set of available Space statistics, see Space Page Statistics.

Instance Vital Signs

The following instance statistics are located on the DB2 Instance view:

- Agent Utilization

- Sort Heap Utilization
- Percent of Agents Stolen
- Waiting for Tokens

NOTE: For the complete set of available Instance statistics, see Instance Page Statistics.

Memory Page Statistics

The Memory view includes the following statistics:

- Buffer Pool Hit Ratio
- Buffer Pool Index Hit Ratio
- Database Heap Utilization
- Catalog Cache Hit Ratio
- Memory Overflows
  - Sort Overflow Percentage
  - Hash Join Overflow Percentage
  - Catalog Cache Overflows
  - Package Cache Overflows
  - Private Workspace Overflows
  - Shared Workspace Overflows
- Lock List Utilization
- Package Cache Hit Ratio
- Session Leaders – Memory
- Sort Heap Utilization
- Shared Workspace Hit Ratio

Buffer Pool Hit Ratio

The DB2 database server reads and updates all data from a bufferpool because memory access is much faster than disk access. Data is copied from disk to a bufferpool as needed by the applications using the database. When the server needs to read/write data and the data is already in the bufferpool, no disk access is required. However, if the data is not in the bufferpool, it needs to be read from the disk, which is a significantly slower process.

The buffer pool hit ratio indicates the percentage of time that the database server did not need to load a page to service the request for a page. The calculation takes into account all the index and data pages that were requested.

Metrics

Avoiding disk I/O is the main issue when you try to improve the performance tuning. A high buffer pool hit ratio is desirable because it indicates a lower frequency of synchronous disk I/O. A database where data is accessed uniformly from very large tables will have a poor hit ratio. There is little you can do to improve the performance in such cases.

Troubleshooting

The buffer pool hit ratio on the database Home page is the overall hit ratio of the database. First, drill down to the bufferpool level and check the individual buffer pool hit ratios to identify the elements that may require tuning. Increasing the bufferpool size generally improves the bugger pool hit ratio. You can use one of the following methods to increase the bufferpool size depending on the layout of the data.

1. If the tablespaces using the bufferpool have tables and indexes on them, increase the bufferpool size incrementally until the index page hit ratio stops increasing for that bufferpool. You will not usually get any benefit increasing the bufferpool size after you reach that point.
2. If the tablespaces associated with the bufferpool have only *indexes* on them, increase the bufferpool size until the index page hit ratio stops increasing.
3. If the tablespaces associated with the bufferpool have only *data* on them, increase the bufferpool size until the data page hit ratio stops increasing.

One general design guideline is to try and use different tablespaces for indexes and tables and associate different bufferpools with them. This generally improves performance and makes bufferpool tuning much easier.

Buffer Pool Index Hit Ratio

The buffer pool index hit ratio is calculated by separating the pages and indexes that are cached by the buffer pool: (1 - ((pool_index_p_reads) / (pool_index_l_reads)))*100%.

Troubleshooting

By tuning the buffer pool index hit ratio separately, you may be able to improve the buffer pool hit ratio.

Database Heap Utilization

There is one database heap per database, and the database manager uses it on behalf of all applications connected to the database. The database heap utilization is the percentage of database heap that is currently being used.

Metrics

The database heap contains control block information for tables, indexes, table spaces, and bufferpools. It also contains space for event monitor buffers, the log buffer, and temporary memory used by utilities. Thus, the heap utilization can increase or decrease whenever any of the constituent elements change. If the utilization goes above 85% several times or stays above 85% for a sustained period, it may mean that you need to increase the maximum database heap size.

Troubleshooting

The *dbheap* database configuration parameter determines the maximum amount of heap memory that can be allocated. Examine the individual memory pools and how they are using the memory before increasing this parameter. An excessively large use of one of the memory pools may cause this problem.

Catalog Cache Hit Ratio

Catalog cache is used to cache the following types of catalog information:

- Metadata for tables, views, and aliases.
- Database authorization information that is used to check authorization when performing operations like CONNECT, LOAD, CREATE, BIND, and so on.

- Execute privileges for user-defined functions and stored procedures.

When a database operation accesses catalog information, it inserts this information into the catalog cache so operations accessing the same information can read it from the cache and avoid disk reads.

The catalog cache hit ratio indicates how well the catalog cache avoids accesses to the catalog.

Metrics

A high catalog cache hit ratio (> 80%) is desirable and it indicates that the catalog cache is working well. A smaller ratio can indicate that this parameter needs tuning. You may see a smaller ratio immediately following the first connection to the database and execution of DDL/DCL statements since these require heavy catalog accesses.

Troubleshooting

If the catalog cache hit ratio is consistently small, the database configuration parameter *catalogcache_sz* should be increased. When you increase the value of this parameter, pause to consider whether it would be more effective to allocate the extra memory you are reserving to another purpose such as package cache or bufferpools.

In a partitioned database environment, make the *catalog cache_sz* larger on the catalog partition than on other partitions because the catalog information required for all partitions will go through this partition.

When tuning this parameter, it is advisable to monitor changes to the database catalog via DDL statements. During the execution of DDL statements, there may be a large drop in the catalog cache hit ratio due to invalidation of the cached data.

Sort Overflow Percentage

The sort overflows statistic is the total number of sorts that ran out of sort heap and that may have required disk space for temporary storage. Sort Overflow Percentage is the percentage of sorts that need more heap space.

Metrics

When a sort overflows, additional overhead is incurred. The sort requires a merge phase and can potentially require more I/O if data needs to be written to disk.

Troubleshooting

Sort overflows can be reduced by increasing the *sortheap* database configuration parameter.

Hash Join Overflow Percentage

Hash join is an option for the DB2 optimizer. A hash join compares the hash codes before comparing the predicates for tables involved in a join. This reduces the number of comparisons. The hash join overflows percentage metric gives the percentage of all hash join operations that ran out of sort heap and may have required disk space for temporary storage since the current instance of Performance Analyst started monitoring the database.

Metrics

An overflowing hash join incurs a large overhead because of the amount of disk I/O required to complete the operation. If this value crosses the 30% mark, the DBA should take action.

Troubleshooting

Increase the *sortheap* database configuration parameter to reduce the number of overflows.

Catalog Cache Overflows

The catalog cache overflow statistic is the number of times the catalog cache overflowed the bounds of its allocated memory.

Metrics

Catalog cache overflows can cause unnecessary lock escalations. This can result in loss of concurrency, or ‘out of memory’ errors from other heaps allocated to the database’s shared memory. Overflows of the catalog cache can also cause performance degradation.

DB2 reclaims the catalog cache space by evicting table descriptor information for tables, views, or aliases and/or authorization information that is not currently being used by any transaction.

Troubleshooting

Use this element with the catalog cache high watermark to determine whether the size of the catalog cache needs to be increased to avoid overflowing.

If the number of overflows is large, the catalog cache may be too small for the workload. Enlarging the catalog cache can improve its performance. If the workload includes transactions that compile a large number of SQL statements referencing many tables, views, aliases, user-defined functions, or stored procedures in a single unit of work, then compiling fewer SQL statements in a single transaction can improve the performance of the catalog cache. Or, if the workload includes binding packages that contain many SQL statements referencing many tables, views, aliases, user-defined functions, or stored procedures, you can try splitting packages so that they include fewer SQL statements to improve performance.

Package Cache Overflows

The package cache overflows metric is the number of times that the package cache overflowed the bounds of its allocated memory.

Metrics

Package cache overflows can cause unnecessary lock escalations. This can result in loss of concurrency, or ‘out of memory’ errors from other heaps allocated to the database’s shared memory. Overflows of the package cache can also cause performance degradation.

Troubleshooting

Use this element with the package cache high watermark to determine whether the size of the package cache needs to be increased to avoid overflowing.

Private Workspace Overflows

The private workspace overflows statistic is the number of times that private workspaces overflowed the bounds of their allocated memory.

Metrics

Private workspace overflows can cause performance degradation as well as ‘out of memory’ errors from other heaps allocated to the agent’s private memory.

Troubleshooting

Use this element with the private workspace high watermark to determine whether the size of the private workspace needs to be increased to avoid overflowing.

Shared Workspace Overflows

The shared workspace overflows metric is the number of times that shared workspaces overflowed the bounds of their allocated memory.

Metrics

Overflows of shared workspaces can cause performance degradation. Overflows can also ‘out of memory’ errors from the other heaps allocated out of application’s shared memory.

Troubleshooting

Use this element with the shared workspace high watermark to determine whether the size of the shared workspaces need to be increased to avoid overflowing.

Lock List Utilization

Lock list utilization is the percentage of total database memory allocated for locks that is currently being used.

Metrics

There is only one lock list for each database and it contains the locks held by all applications connected to the database. Once the lock list is full, the database manager starts escalating row locks to table locks to free up space. This escalation may result in serious performance degradation because of reduced concurrency. Additionally, the number of deadlocks and transaction rollbacks may go up.

If this metric reaches the 75% mark, you should consider bringing this percentage down with tuning.

Troubleshooting

Depending on the database’s activity level, you may be able to reduce the lock utilization by following these recommendations:

Increase size of lock list: If there is not enough lock list space available, lock escalations will occur, thereby increasing contention and reducing concurrency. Update the *locklist* database configuration parameter to increase this value.

Tune applications to reduce locking: On the Locks tab of the Users detail section, identify the applications that are holding many locks and then consider the following steps for controlling the size of the lock list:

- Make the transactions shorter by increasing the COMMIT frequency. This ensures that locks are released frequently, thus freeing up lock space.
- Before you update many rows in a single table, lock the entire table (using the SQL LOCK TABLE statement). This prevents many row-level locks from being obtained (although this decreases concurrency)
- To control how locking is done for a specific table, use the LOCKSIZE parameter of the ALTER TABLE.
- To decrease the number of share locks held, use the Cursor Stability isolation level when possible. If the applications’ integrity requirements are not compromised, use Uncommitted Read instead of Cursor Stability to further decrease the amount of locking.

Decrease percentage of lock list: If a small number of applications are consuming most of the lock space, decrease the percentage of lock list for each application. You can throttle back those applications by decreasing the maxlocks database configuration parameter. This reduces the amount of lock list memory available to each application thereby allowing for better distribution of lock list memory.

> **NOTE:** Decreasing the percentage of lock list should be the last resort, and used only if you cannot decrease utilization with the other recommendations. It can cause a large number of lock escalations.

Package Cache Hit Ratio

The sections for frequently used dynamic and static SQL statements are cached in the package cache.

The package cache hit ratio indicates how well the package cache is avoiding catalog accesses to packages and recompilations.

Metrics

A high package cache hit ratio (> 80%) is a good thing. It indicates that the package cache is working well. In the case of static SQL statements, package caching allows the Database Manager to reduce the internal overhead by eliminating the need to access system catalogs when reloading a package. For dynamic SQL, the benefit of package caching is even greater since it a cache hit eliminates the need for recompilation.

The package cache hit ratio metric is particularly important for transaction-processing applications since a typical workload involves repeatedly executing the same SQL statements.

Troubleshooting

Executing DDL statements can invalidate sections of SQL statements in the cache, causing the hit ratio to decrease dramatically. Before attempting to tune this parameter, you should check the DDL activity to see if that is causing a skew. No amount of tuning will improve the package cache performance if the DDL activity is causing package invalidations in the cache.

If the DDL activity is minimal and package cache hit ratio is consistently small, consider increasing the package cache size (pckcachesz) in the database configuration. When increasing the value of this parameter, consider whether it might be more effective to allocate the extra memory being reserved to another purpose such as catalog cache or bufferpools.

Session Leaders – Memory

Session Leaders – Memory shows the top five memory consuming applications that are running in the database. The details include:

Agent ID: A system-wide unique identifier for the application. On a single-partitioned database, this identifier consists of a 16 bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16 bit counter. In addition, this identifier will be the same on every partition where the application may make a secondary connection.

Auth ID: The authorization ID of the user who invoked the application that is being monitored.

Memory: The total memory pool usage for the application.

> **NOTE:** Click the **Session Leaders -- Memory** heading to view more details on the Top Sessions View.

Sort Heap Utilization

The amount of memory allocated for each sort may be some or all of the available sort heap size. Sort heap size is the amount of memory available for each sort as defined in the database configuration parameter *sortheap*. Shared sort memory utilization gives the percentage of the sort heap being used.

It is possible for a single application to have concurrent sorts active. For example, in some cases a SELECT statement with a subquery can cause concurrent sorts.

Metrics

Memory estimates do not usually include sort heap space. If excessive sorting is occurring, the extra memory used for the sort heap should be added to the base memory requirements for running the database manager. Generally, the larger the sort heap, the more efficient the sort. Typically the shared sort utilization should be less than or equal to 70%. You should consider tuning the database if you see a utilization value greater than this.

Troubleshooting

To bring the sort heap utilization to an acceptable level, use the following guidelines:

- Examine the queries you are running on the database to see if you can add indexes to columns frequently accessed in the WHERE clause. This minimizes the use of the sort heap.
- If you are using dynamic bitmaps or hash join buffers in your queries, or your queries frequently require large sorts, increase the *sortheap* size.
- If you adjust the *sortheap* size, also look at the *sheapthres* database manager configuration parameter to see if it too needs to be adjusted

If you are adjusting the *sortheap* size, you may also benefit from rebinding your packages since the optimizer takes this parameter into account when determining the access paths.

Shared Workspace Hit Ratio

When sections are required by an application for executing dynamic or static SQL statements, they are placed in the shared workspace. The shared workspace exists at the application level and is shared among applications using the database.

The hit ratio is a percentage indicating how well the shared SQL workspace is helping to avoid initialization of sections for SQL statements that are about to be executed. A high ratio indicates the shared workspace is successful in avoiding this action.

Metrics

A shared workspace is shared by many applications. If applications have similar database access patterns, they can benefit greatly if they find required sections in the shared workspace. If an application finds a section in the shared workspace (e.g., a hit), that application avoids the setup and initialization cost for that section. A high hit ratio is desirable for this metric. Typically, you should expect to see a high ratio (>80%) in transaction processing environments.

Troubleshooting

Shared workspace memory is allocated from the application control heap (app_ctl_heap_sz database configuration parameter) and increasing this may improve the hit ratio.

I/O Page Statistics

The I/O statistics page displays the following vital DB2 I/O statistics:

- Direct Read Ratio
- Logical Read Ratio
- Physical Read Ratio
- Synchronous Read Ratio

- Asynchronous Read Ratio
- Direct Write Ratio
- Synchronous Write Ratio
- Asynchronous Write Ratio
- Log Read Rate
- Direct Read Rate
- Synchronous Read Rate
- Asynchronous Read Rate
- Log Write Rate
- Direct Write Rate
- Synchronous Write Rate
- Asynchronous Write Rate
- I/O Agent Statistics
  - Prefetchers
  - Page Cleaners
  - Log Space Cleans
  - Dirty Page Cleans
  - Prefetch Wait Time
- Key I/O Ratios
  - Buffer Pool Hit Ratio
  - Buffer Pool Index Hit Ratio
  - Victim Cleans Ratio
  - Threshold Cleans Ratio
  - Log Space Cleans Ratio
- Session Leaders – I/O

Direct Read Ratio

Direct Read Ratio is the percentage of all reads that were direct reads. Direct reads are read operations that do not use the buffer pool.

Logical Read Ratio

Logical Read Ratio is the percentage of all reads that were logical reads. Logical Reads is the sum of all Buffer Pool Data Logical Reads and Buffer Pool Index Logical Reads.

Physical Read Ratio

Physical Read Ratio is the percentage of all reads that were physical reads. Physical Reads is the sum of all Buffer Pool Data Physical Reads and Buffer Pool Index Physical Reads.

Synchronous Read Ratio

Synchronous Read Ratio is the percentage of all index and data reads that performed synchronously by the database manager prefetchers.

Asynchronous Read Ratio

Asynchronous Read Ratio is the percentage of all index and data reads that performed asynchronously by the database manager prefetchers.

Metrics

Prefetching refers to the technique whereby the Database Manager can read several pages on the physical reads device simultaneously into a bufferpool in anticipation of access to these pages. Since each physical read operation is costly, a 50% async read ratio is desirable. It shows that the prefetchers are working well and read waits are being minimized.

Troubleshooting

The *num_ioserver* database configuration parameter defines the number of prefetchers that are available for the database. To get an optimal async read ratio, you should set the *num_ioserver* database configuration parameter to 1-2 more than the number of physical storage devices used by the database to ensure that asynchronous I/O is occurring. This parameter should be changed judiciously. Having too many prefetchers invariably results in high system I/O because prefetchers can read many more pages than required into the bufferpools.

Direct Write Ratio

Direct Write Ratio is the percentage of all writes that were direct writes. Direct Writes are write operations that do not use the buffer pool.

Synchronous Write Ratio

Synchronous Write Ratio is the percentage of all index and data writes that were performed synchronously by the database manager bufferpool page cleaners.

Asynchronous Write Ratio

Asynchronous Write Ratio is the percentage of all index and data writes that were performed asynchronously by the database manager bufferpool page cleaners.

Metrics

Page cleaners write changed pages from bufferpool to disk before the space in the bufferpool is needed by a database agent. If the page cleaners are not working well, the problem may manifest itself in two ways:

- The database agents need to synchronously free space in bufferpools resulting in poorer response time.

- If the system crashes, the recovery time of the system is greater because there will be a large number of pages that would not have been committed to disk at the time of the crash and they require processing for a large number of log records.

An overall async write ratio of 50% is desirable for most transactional database systems. If your database is 'query only' (i.e., updates are rarely or never performed), it is fine to have an async write ratio of 0 to less than 50%. Even in 'query only' databases if the queries create temp tables on the database for intermediate results, a 30-40% async write ratio is desirable.

Troubleshooting

The async write ratio can be tuned using the *num_iocleaners* database configuration parameter, which specifies the number of asynchronous page cleaners for a database. Increasing the number of page cleaners generally results in a higher async write ratio. The following rules of thumb should be followed when tuning this parameter:

- For most transactional systems set this parameter to between one and the number of physical storage devices used by the database.
- For workloads that consists mainly of simple reads from the database, set this parameter to zero.
- When you define very large bufferpools on your database you may need to increase the *num_iocleaners*
- Monitor the victim page cleaners% metric when tuning the *num_iocleaners*.

Log Read Rate

Log Read Rate is the number of pages (per second) the logger read from disk during the last monitoring interval.

Direct Read Rate

Direct Reads Rate is the number of read operations not using the buffer pool (per second) during the last monitoring interval.

Synchronous Read Rate

Synchronous Read Rate is the number of synchronous reads (per second) in the last monitoring interval.

Asynchronous Read Rate

Asynchronous Read Rate is the number of asynchronous reads (per second) in the last monitoring interval.

Log Write Rate

Log Write Rate is the number of pages (per second) the logger read from disk during the last monitoring interval.

Direct Write Rate

Direct Write Rate is the number of write operations not using the buffer pool (per second) during the last monitoring interval.

Synchronous Write Rate

Synchronous Write Rate is the number of synchronous writes (per second) in the last monitoring interval.

Asynchronous Write Rate

Asynchronous Write Rate is the number of asynchronous writes (per second) in the last monitoring interval.

Prefetchers

The *num_ioserver* database configuration parameter defines the number of prefetchers that are available for the database.

Troubleshooting

To get an optimal async read ratio, you should set the *num_ioserver* database configuration parameter to 1-2 more than the number of physical storage devices used by the database to ensure that asynchronous I/O is occurring. This parameter should be changed judiciously since having too many prefetchers invariably results in high system I/O because prefetchers can read many more pages than required into the bufferpools.

Page Cleaners

The *num_iocleaners* database configuration parameter specifies the number of asynchronous page cleaners for a database. Increasing the number of page cleaners generally results in a higher async write ratio.

Troubleshooting

The following rules of thumb can be followed when tuning this parameter:

- For most transactional systems set this parameter to between 1 and the number of physical storage devices used by the database.
- For workloads that consists of mainly simple reads from the database, set this parameter to 0.
- When you define very large bufferpools on your database you may need to increase the *num_iocleaners*.

Monitor the victim page cleaners% metric when tuning the *num_iocleaners*.

Log Space Cleans

Log Space Cleans is the number of times a page cleaner was invoked because the logging space used reached a predefined criterion for the database.

Metrics

This element can help you determine whether you have enough space for logging, and whether you need more log files or larger log files. The page cleaning criterion is set using the *softmax* configuration parameter. Page cleaners are triggered when the oldest page in the buffer pool contains an update described by a log record that is older than the current log position by the criterion value.

Dirty Page Cleans

Dirty Page Cleans is the sum of all buffer pool victim page cleaners triggered and buffer pool threshold cleaners triggered. *Buffer pool victim page cleaners triggered* is the number of times a page cleaner was invoked because a synchronous write was needed during the victim buffer replacement for the database. *Buffer pool threshold cleaners triggered* is the number of times a page cleaner was invoked because a buffer pool had reached the dirty page threshold criterion for the database.

Prefetch Wait Time

This is the total time database applications spent waiting for prefetchers to finish loading.

Metrics

None.

Troubleshooting

If you see a high number for this metric, it indicates there is a bottleneck in the prefetcher operations. Experiment with changing the number of I/O servers *(num_ioserver)* and the I/O server sizes.

Buffer Pool Hit Ratio

The DB2 database server reads and updates all data from a bufferpool because memory access is much faster than disk access. Data is copied from disk to a bufferpool as needed by the applications using the database. When the server needs to read/write data and the data is already in the bufferpool, no disk access is required. However, if the data is not in the bufferpool, it needs to be read from the disk, which is significantly slower process.

The buffer pool hit ratio indicates the percentage of time that the database server did not need to load a page to service the request for a page. The calculation takes into account all the index and data pages that were requested.

Metrics

Avoiding disk I/O is the main issue when you try to improve the performance tuning. A high buffer pool hit ratio is desirable because it indicates a lower frequency of synchronous disk I/O. A database where data is accessed uniformly from very large tables will have a poor hit ratio. There is little you can do to improve the performance in such cases.

Troubleshooting

The buffer pool hit ratio on the database Home page is the overall hit ratio of the database. First, drill down to the bufferpool level and check the individual buffer pool hit ratios to identify the elements that may require tuning. Increasing the bufferpool size generally improves the buffer pool hit ratio. You can use one of the following methods to increase the bufferpool size depending on the layout of the data:

1. If the tablespaces using the bufferpool have tables and indexes on them, increase the bufferpool size incrementally until the index page hit ratio stops increasing for that bufferpool. You will not usually get any benefit increasing the bufferpool size after you reach that point.
2. If the tablespaces associated with the bufferpool have only *indexes* on them, increase the bufferpool size until the index page hit ratio stops increasing.
3. If the tablespaces associated with the bufferpool have only *data* on them, increase the bufferpool size until the data page hit ratio stops increasing.

One general design guideline is to try and use different tablespaces for indexes and tables and associate different bufferpools with them. This generally improves performance and makes bufferpool tuning much easier.

Buffer Pool Index Hit Ratio

The buffer pool index hit ratio is calculated by separating the pages and indexes that are cached by the buffer pool: (1 - ((pool_index_p_reads) / (pool_index_l_reads))) *100%.

Troubleshooting

By tuning the buffer pool index hit ratio separately, you may be able to improve the buffer pool hit ratio.

Victim Cleans Ratio

Victim Cleans Ratio is the percentage of all page cleans that were victim page cleans. Victim page cleans are triggered when a synchronous write is needed during the victim buffer replacement for the database.

Troubleshooting

If this ratio is low, you may have defined too many page cleaners. Because the buffer pool is meant to defer writing to the last possible moment, having too many page cleaners is counterproductive. If the ratio is high, it might mean that you have too few page cleaners defined, which will increase recovery time after failures.

Threshold Cleans Ratio

Threshold Cleans Ratio is the percentage of all page cleans that were invoked because a buffer pool had reached the dirty page threshold criterion for the database.

Metrics

This dirty page threshold is set by the *chngpgs_thresh* configuration parameter. Cleaners are triggered when the number of dirty pages in the pool exceeds the percentage (threshold) applied to the buffer pool size.

Troubleshooting

If this ratio is too low, pages could be written out too early, and they will need to be read back in. If the ratio is too high, too many pages may accumulate, and users will have to write out pages synchronously.

Log Space Cleans Ratio

The log space cleaners triggered metric is the number of times a page cleaner was triggered because log space usage reached a predefined threshold for the database.

Metrics

The log space cleaners are triggered every time the space used by the DB2 log reaches the limit set in the *softmax* database configuration parameter. This parameter specifies the percentage of primary log size at which the cleaners are triggered. By default this parameter is set to 100.

A low rate of log space cleaners getting triggered may indicate that logs are not being written to disk frequently enough and that you may need to process a large number of log records and redundant log records in the event of a crash recovery.

High rates of log space cleaners getting triggered can indicate that your primary log is too small or that you have set the *softmax* too high. In either case, a very high rate of log cleaners being triggered may adversely impact database performance.

It is important to keep in mind, however, that more page cleaner triggers and more frequent soft checkpoints increase the overhead associated with database logging.This can have an impact on the performance of the Database Manager. Also, more frequent soft checkpoints may not reduce the time required to restart a database, if you have:

- Very long transactions with few commit points.
- A very large bufferpool and the pages containing the committed transactions are not written back to disk very frequently. The use of asynchronous page cleaners can help avoid this situation.

In both of these cases, the log control information kept in memory does not change frequently and there is no advantage in writing the log control information to disk, unless it has changed

Troubleshooting

Examine this metric together with other page cleaner metrics and the async write ratio to determine if excessive numbers of log page cleaners are being triggered. If this is true, you need to either increase your primary log size or adjust the *softmax* parameter to a higher value.

> **NOTE:** Decreasing the log page cleaner triggering rate can impact the overall crash recovery time so it needs to be done judiciously.

Session Leaders – I/O

Session Leaders – I/O shows the top five applications consuming the most I/O. The details include:

Agent ID – A system-wide unique ID for the application. On a single-partitioned database, this identifier consists of a 16 bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16 bit counter. In addition, this identifier will be the same on every partition where the application may make a secondary connection.

Auth ID – The authorization ID of the user who invoked the application that is being monitored. On a DB2 Connect gateway node, this is the user's authorization ID on the host.

Total I/O (ms) – The total amount time (in milliseconds) spent performing buffered and direct reads and writes.

Space Page Statistics

- Database Overview
  - DMS Space (Total Used and Total Free)
  - DMS Utilization
  - SMS Space (Total Used and Total Free)
  - SMS Utilization
- Log Overview
  - Active Log Size
  - Active Log Used
  - Active Log Free
  - Active Log Utilization-+
  - Secondary Logs Allocated
  - Secondary Log Used HWM

- Exception Summary
  - Inaccessible Containers
  - Tablespaces Low on Space
  - Abnormal State Tablespaces
- Tablespace Overview

DMS Space (Total Used and Total Free)

The total used and total free metrics show the space details for DMS tablespaces. Specifically, they show the amount of used and free space on the DMS tablespaces in the database.

Metrics

Unlike SMS tablespaces, the total available space to the DMS tablespaces is defined at the time the database is defined. The DBA needs to explicitly increase the amount of space on DMS tables by adding/extending tablespace containers (using the ALTER TABLESPACE statement). If left unattended, a DMS tablespace can either remain underutilized or fill up completely. Keeping an eye on the DMS space is important because the once it fills up, the applications trying to write to the database will come to a halt suddenly.

Troubleshooting

Go to the Space home page or drilldown to see the usage metrics for individual tablespaces and allocate/deallocate space to containers accordingly using ALTER TABLESPACE command.

DMS Utilization

DMS utilization is the percentage of all DMS tablespaces that are free.

SMS Space (Total Used and Total Free)

This metric shows the total amount of free and used SMS space by the database.

Metrics

The maximum size of SMS tablespaces is not set at the time the tablespaces are created. The maximum size that can be reached is the space available on the drives/volumes that the SMS tablespace containers are defined on. You need to periodically examine the available space of the drives/volumes (using OS metrics Space page) to make sure there is enough space available for your database requirements. Since the space is determined by the space available on the drives/volumes, remember that if other applications are reading and writing to the same devices (especially logs, traces, etc), DB2 may be periodically competing for space with these applications.

Troubleshooting

Ideally, try and isolate the DB2 SMS drives/volumes from other applications. If the OS level metrics show that you are running out of space on a volume that is used by a tablespace's containers, you can add more containers defined on volumes with more space, to the tablespace using the ALTER TABLESPACE command.

SMS Utilization

SMS utilization is the percentage of all SMS tablespaces that are free.

Active Log Size

Active Log Size is the total amount of active log space (in megabytes) available in the database.

Active Log Used

Active Log Used is the current amount of active log space (in megabytes) being used in the database.

Active Log Free

Active Log Free is the active log space (in megabytes) in the database that is not currently being used.

Active Log Utilization

Active Log Utilization is the percentage of space consumed in the database. A very high percentage may cause an alert.

Secondary Logs Allocated

This section shows the secondary logs allocations by the database over the monitoring period.

Metrics

When the primary log files become full, the secondary log files are allocated one at a time as needed, up to a maximum number as controlled by the logsecond database configuration parameter. Secondary log files are useful in preventing occasional log fill ups but they may not be as efficient as primary log and they also increase the recovery time. If you see a constant reliance on secondary log files, it may indicate infrequent commits by database applications or insufficient primary log space.

Troubleshooting

First examine the database applications to see if secondary log files are due to long running transactions and whether these transactions can be shortened. If the transactions cannot be shortened or you still see a very frequent use of secondary logs after transaction tuning, increase the primary log size (logprimary, logfilsiz Database configuration parameters) to reduce the dependence on secondary logs.

If there is an occasional long running transaction, and you see your see transaction log full (SQL0964C) error messages, you can either increase the number of secondary log files (logsecond database configuration parameter) or set the number to -1 (no limit on number of secondary log files).

Secondary Log Used HWM

Secondary Log Used HWM is the maximum amount of secondary log space used (in megabytes).

Troubleshooting

If this value is high, you can try several options:

- Increase the size of the log files.
- Create more primary logs.

The following configuration parameters may need to be updated as a result:

- logfilsiz
- logprimary
- logsecond
- logretain

NOTE: If there are no secondary log files defined, the value will be zero.

Inaccessible Containers

The inaccessible containers metric identifies the number of containers on all tablespaces that are currently inaccessible.

Metrics

Inaccessible containers represent a serious problem on the database. The database is unable to access/write any data on an inaccessible container. Usually this is caused by either media errors or modifications to the container files/directories from outside DB2.

Troubleshooting

Drill down to the Space metrics to examine the inaccessible containers and the tablespaces on which they are contained. You may need to perform a redirected restore on the tablespace from a backup to correct this.

Tablespaces Low on Space

Tablespaces Low on Space is the number of tablespaces in the database where the used space on the tablespace is more than 80% of the total available space.

Metrics

This metric is an indicator that some of the tablespaces in your database may be running out of space. You should drill down to identify the tablespaces that may be at the critical utilization level. In tablespaces where there are little or no inserts after the initial data is loaded, little or no free space may have been left on purpose. In such cases, it is normal to see a high utilization.

Troubleshooting

If this metric goes beyond the normal operational value for your database, consider creating additional space on the tablespaces that are running out of space. You can do this by performing one or more one of the following tasks:

- Clean up unused space by deleting the tables/indexes that are no longer required.
- Resize or extend the existing containers in the tablespaces.
- Add new containers to the tablespaces.

Abnormal State Tablespaces

The tablespaces in abnormal state metric is a count of the number of tablespaces in your database that are not in a 'Normal State'.

Metrics

Tablespaces in not normal state are not necessarily a cause for alarm. They may have been transitioned to that state intentionally by the DBA for maintenance related tasks. If this metric shows a value that is higher than you anticipated, you may need to drilldown to the Space metrics to see which tablespaces are not normal.

A tablespace can be in one or more of the following states when it is not 'Normal':

- Quiesced: SHARE, UPDATE, EXCLUSIVE
- Pending: Load, delete, backup, roll forward, restore, disable, drop
- In progress: Roll forward, reorg, backup, TBS deletion, TBS creation
- Storage must be defined
- Restore in progress
- Offline and not accessible
- Storage may be defined
- Storage definition is in 'final' state
- Storage definition was changed prior to rollforward
- DMS rebalancer is active

Tablespace Overview

The Tablespace Overview section includes a list of all of the tablespaces in the database. The following information is presented for each tablespace:

- **Tablespace:** The name of the tablespace.
- **Type:** system managed (SMS) or database managed (DMS) tablespace.
- **State:** This element describes the current state of the tablespace.
- **Size (MB):** Total space (Used and Free) used by the tablespace on the storage device(s). This will be the same as used space for SMS tables.
- **Used (MB):** The space currently in use on tablespace.
- **Free (MB):** The space currently free on the tablespace. (This is not applicable to SMS tablespaces.)
- **% Used:** The percentage of used space on the tablespace.

Object Page Statistics

The Object view includes the following statistics:

- Buffer Pools Tab – Overview
- Buffer Pools Tab – I/O Analysis
- Tablespaces Tab – Overview
- Tablespaces Tab – I/O Analysis
- Containers Tab

- Tables Tab

Buffer Pools Tab – Overview

The Buffer Pools tab displays details about the buffer pools in the database being monitored:

Node: The unique identification number assigned to the node.

Bufferpool: The name of the buffer pool.

Size (MB): The current buffer pool size.

Page Size (KB): The default page size of the buffer pool.

Files Closed: The number of times a database file was closed because the limit for concurrently open files was reached.

Overall Hit Ratio: Indicates the percentage of time that the database server did not need to load a page from in order to service the request for a page.

Index Hit Ratio: The percentage of all index reads that were satisfied because the page was already available in a bufferpool.

Avg Read Time (ms): The average amount of time (in milliseconds) spent reading in data and index pages from table space containers.

Avg Write Time (ms): The average amount of time (in milliseconds) spent writing data and index pages from the buffer pool to disk.

Pages Read Per Minute: The number of physical reads per minute.

Buffer Pools Tab – I/O Analysis

The Buffer Pools tab displays details about the buffer pools in the database being monitored:

Node: The unique identification number assigned to the node.

Bufferpool: The name of the buffer pool.

% Async Reads: The percentage of all index and data reads performed asynchronously by the database manager prefetchers.

% Sync Reads: The percentage of all index and data reads that performed synchronously by the database manager prefetchers.

% Async Writes: The percentage of all index and data writes performed asynchronously by the database manager bufferpool page cleaners.

% Sync Writes: The percentage of all index and data writes that were performed synchronously by the database manager bufferpool page cleaners.

% Block I/O: The percentage of all block/vectored I/O that is block I/O. Block I/O represents the number of times DB2 performs sequential prefetching of pages into the block area of the buffer pool.

% Vectored I/O: The percentage of all block/vectored I/O that is vectored I/O. Vectored I/O represents the number of times DB2 performs sequential prefetching of pages into the page area of the buffer pool.

Avg Async Read (ms): The average amount of time spent reading data and index pages.

Avg Sync Read (ms): The average amount of time spent performing synchronous reads.

Avg Async Write (ms): The average amount of time spent writing data and index pages to disk.

Avg Sync Write (ms): The average amount of time spent performing synchronous writes.

Tablespaces Tab – Overview

The Tablespaces tab displays details about the tablespaces in the database being monitored:

Tablespace: The name of the tablespace.

Type: system managed (SMS) or database managed (DMS) tablespace.

State: This element describes the current state of the tablespace.

Size (MB): The current buffer pool size.

Used (MB): The space currently in use on tablespace.

Free (MB): The space currently free on the tablespace. (This is not applicable to SMS tablespaces.)

% Used: The percentage of space currently in use on tablespace.

Page Size (KB): The default page size of the buffer pool.

Files Closed: The number of times a database file was closed because the limit for concurrently open files was reached.

Overall Hit Ratio: Indicates the percentage of time that the database server did not need to load a page from in order to service the request for a page.

Index Hit Ratio: The percentage of all index reads that were satisfied because the page was already available in a bufferpool.

Avg Read Time (ms): The average amount of time (in milliseconds) spent reading in data and index pages from table space containers.

Avg Write Time (ms): The average amount of time (in milliseconds) spent writing data and index pages from the buffer pool to disk.

Pages Read Per Minute: The number of physical reads per minute.

Tablespaces Tab – I/O Analysis

The Tablespaces tab displays details about the tablespaces in the database being monitored:

Tablespace: The name of the tablespace.

Bufferpool: The name of the buffer pool.

% Async Reads: The percentage of all index and data reads performed asynchronously by the database manager prefetchers.

% Sync Reads: The percentage of all index and data reads that performed synchronously by the database manager prefetchers.

% Async Writes: The percentage of all index and data writes performed asynchronously by the database manager bufferpool page cleaners.

% Sync Writes: The percentage of all index and data writes that were performed synchronously by the database manager bufferpool page cleaners.

% Block I/O: The percentage of all block/vectored I/O that is block I/O. Block I/O represents the number of times DB2 performs sequential prefetching of pages into the block area of the buffer pool.

% Vectored I/O: The percentage of all block/vectored I/O that is vectored I/O. Vectored I/O represents the number of times DB2 performs sequential prefetching of pages into the page area of the buffer pool.

Avg Async Read (ms): The average amount of time spent reading data and index pages.

Avg Sync Read (ms): The average amount of time spent performing synchronous reads.

Avg Async Write (ms): The average amount of time spent writing data and index pages to disk.

Avg Sync Write (ms): The average amount of time spent performing synchronous writes.

Async Pages Read Per Request: The average number of pages read per asynchronous request.

Containers Tab

The Containers tab displays details about the containers for a tablespace:

Node: The unique identification number assigned to the node.

Tablespace: The name of the tablespace.

ID: A value that uniquely identifies a tablespace used by the current database.

Type: System managed (SMS) or database managed (DMS) tablespace.

Container: The name of the container. Typically this is the full path of the file/directory/device where the container exists.

ID: A value that uniquely identifies the container within the tablespace

Type: The type of container. An SMS Container will be a directory. A DMS Containers will be a raw device/file/stripped disk/ or stripped file. Together with Container name, and partition, this metric identifies the physical location of the container.

Accessible: This element describes if a container is accessible or not.

Size (MB): The size of the container (in megabytes).

Total Pages: Total pages in the container.

Useable Pages: Usable pages in the container (applicable to DMS Tablespaces only).

Node Name: The name of the node.

FS Type: This is the type of file system that the container is defined on such as 'NTFS'. It is obtained from the IBM supplied user defined function "SYSPROC.SNAPSHOT_CNTRFS" and is only available for DB2 v8.2 databases and above.

FS Size (GB): This is the total size in gigabytes (GB) of the file system that the container is defined on. It is obtained from the IBM supplied user defined function "SYSPROC.SNAPSHOT_CNTRFS" and is only available for DB2 v8.2 databases and above.

FS Free (GB): This is the total amount of space in gigabytes (GB) that is available on the file system that the container is defined on. It is obtained from the IBM supplied user defined function "SYSPROC.SNAPSHOT_CNTRFS" and is only available for DB2 v8.2 databases and above.

NOTE: If the DB2 database is not a DB2 v8.2 database the values in these columns will be N/A or zero.

Tables Tab

The Tables tab displays details about the tables in the database being monitored:

Schema: The schema name for the table.

Table Name: The name of the table.

Type: The type of table for which information is returned.

Tablespace: Tablespace where the table data resides.

Bufferpool: The name of the buffer pool.

Rows Read Per Tran: The average number of rows read per transaction.

Rows Read: The number of rows read from the table.

Rows Written: The number of rows written to each table.

Overflow Accesses: The number of reads and writes to overflowed rows in this table.

Page Reorgs: The number of page reorganizations executed for a table.

Data Pages (MB): The amount of tablespace used by data pages (in megabytes).

Index Pages (MB): The amount of tablespace used by index pages (in megabytes).

LOB Pages (MB): The amount of tablespace used by LOB pages (in megabytes).

Long Pages (MB): The amount of tablespace used by long pages (in megabytes).

Contention Page Statistics

The Contention view includes the following statistics:

- System Contention
  - Deadlocks
  - Lock Waits
  - Lock Timeouts
  - Lock Escalations
  - Applications Waiting on Locks
- Lock Overview
- Lock List Utilization
- Lock Wait Time
- Average Lock Wait Time

Deadlocks

Deadlocks shows the total number of deadlocks that have occurred since this instance of Performance Center started monitoring the database.

Metrics

If a large number of deadlocks are detected, it can indicate that applications are experiencing lock contention problems. Deadlocks are usually caused by one of the following situations:

- Lock escalations on the database.
- Catalog tables locked for Repeatable Read.
- Applications are using inappropriate isolation levels at bind time.
- Applications are obtaining the same locks in a different order.
- Applications are locking tables explicitly where row level locks are sufficient.

Troubleshooting

You may be able to modify the applications causing lock contentions for better concurrency.

To identify the applications that may be causing contentions, go to the Lock View.

Lock Waits

At the database level, this is the total number of times that applications have had to wait for locks within this database. At the application-connection level, this is the total number of times that this connection requested a lock but had to wait because another connection was already holding a lock on the data.

Metrics

This element may be used with lock_wait_time to calculate, at the database level, the average wait time for a lock. This calculation can be done at either the database or the application-connection level.

Troubleshooting

If the average lock wait time is high, you should look for applications that hold many locks, or have lock escalations, with a focus on tuning your applications to improve concurrency, if appropriate. If escalations are the reason for a high average lock wait time, then the values of one or both of the locklist and maxlocks configuration parameters may be too low.

Lock Timeouts

The lock timeouts metric identifies the number of times that a request to lock an object timed out without being granted.

Metrics

If the number of lock timeouts becomes excessive when compared to the acceptable range for your database, it can indicate that an application is holding locks for long durations. It can also indicate that the amount of time an application waits for a lock before timing out is too short.

If you have too few lock timeouts and the average lock wait time is too high, it can indicate that the lock timeout configuration parameter is set to an excessively high value.

Troubleshooting

First you should examine the lock activity at the application level to identify any particular application that is causing excessive lock contentions. If so, you can tune the application to provide better concurrency. If lock timeouts are excessive, and average lock wait times are very short, you can increase the *locktimeout* database configuration parameter to make the applications wait longer before timing out.

Lock Escalations

The lock escalations metric indicates the number of times that locks have been escalated from row locks to table locks, since this instance of Performance Center started monitoring the database.

Metrics

A lock is escalated when the total number of locks held by an application reaches the maximum amount it is allowed in the lock list memory. There are several possible causes of lock escalations:

- The database lock list size is too small for the concurrent workload
- The maximum percentage of lock list memory allowed for each application is too small
- One or more applications are obtaining an excessive number of locks

Monitor the lock escalations over a period of time to determine what levels are acceptable in your environment. If the escalations are excessive, or are accompanied by deadlocks or long lock waits, consider tuning the database.

Applications Waiting on Locks

The Apps Waiting on Locks metric gives the percentage of all currently connected applications that are waiting for locks.

Metrics

If this number is high, you should investigate whether the applications are having concurrency problems.

Troubleshooting

Compare this metric with the lock escalations metric to identify if the lock list memory is too small.

Go to the Locks Waiting Details tab of the Lock View and examine the lock activity at application level to identify the applications that are holding a large number of row locks and table-level locks. You may be able to tune the applications with a high number of locks.

Lock Overview

Agent ID – A system-wide unique ID for the application. On a single-partitioned database, this identifier consists of a 16 bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16 bit counter. In addition, this identifier will be the same on every partition where the application may make a secondary connection.

Auth ID – The authorization ID of the user who invoked the application that is being monitored. On a DB2 Connect gateway node, this is the user's authorization ID on the host.

Application – The name of the application running at the client, as known to the database or DB2 Connect server.

Locks Waiting – The total number of times that applications or connections waited for locks.

Lock Wait Time (ms) – The total elapsed time applications waited for a lock (given in milliseconds).

Timeouts – Indicates the timeout value (in seconds) when an application has issued a SET CURRENT LOCK TIMEOUT statement. In cases where the statement has not been executed, the database level lock timeout will be shown.

Deadlocks – Shows the total number of deadlocks that have occurred since this instance of Performance Center started monitoring the database.

Lock List Utilization

Lock list utilization is the percentage of total database memory allocated for locks that is currently being used.

Metrics

There is only one lock list for each database and it contains the locks held by all applications connected to the database. Once the lock list is full, the database manager starts escalating row locks to table locks to free up space. This escalation may result in serious performance degradation because of reduced concurrency. Additionally, the number of deadlocks and transaction rollbacks may go up.

If this metric reaches the 75% mark, you should consider bringing this percentage down with tuning.

Troubleshooting

Depending on the database's activity level, you may be able to reduce the lock utilization by following these recommendations:

Increase size of lock list: If there is not enough lock list space available, lock escalations will occur, thereby increasing contention and reducing concurrency. Update the *locklist* database configuration parameter to increase this value.

Tune applications to reduce locking: On the Lock View, identify the applications that are holding many locks and then consider the following steps for controlling the size of the lock list:

- Make the transactions shorter by increasing the COMMIT frequency. This ensures that locks are released frequently, thus freeing up lock space.
- Before you update many rows in a single table, lock the entire table (using the SQL LOCK TABLE statement). This prevents many row-level locks from being obtained (although this decreases concurrency).
- To control how locking is done for a specific table, use the LOCKSIZE parameter of the ALTER TABLE.
- To decrease the number of share locks held, use the Cursor Stability isolation level when possible. If the applications' integrity requirements are not compromised, use Uncommitted Read instead of Cursor Stability to further decrease the amount of locking.

Decrease percentage of lock list: If a small number of applications are consuming most of the lock space, decrease the percentage of lock list for each application. You can throttle back those applications by decreasing the maxlocks database configuration parameter. This reduces the amount of lock list memory available to each application thereby allowing for better distribution of lock list memory.

> **NOTE:** Decreasing the percentage of lock list should be the last resort, and used only if you cannot decrease utilization with the other recommendations. It can cause a large number of lock escalations.

Lock Wait Time

Lock Wait Time is the elapsed time (in milliseconds) that all applications were waiting for a lock.

Metrics

At the database level, this is the total amount of elapsed time that all applications were waiting for a lock within this database. At the application-connection and transaction levels, this is the total amount of elapsed time that this connection or transaction has waited for a lock to be granted to it.

This element may be used in conjunction with the lock_waits monitor element to calculate the average wait time for a lock. This calculation can be performed at either the database or the application-connection level.

When using monitor elements providing elapsed times, you should consider:

- Elapsed times are affected by system load, so the more processes you have running, the higher this elapsed time value.
- To calculate this element at the database level, the database system monitor sums the application-level times. This can result in double counting elapsed times at a database level, since more than one application process can be running at the same time.

Troubleshooting

None.

Average Lock Wait Time

This is the average time, applications waited for locks in the database.

Metrics

This metric helps you determine if the applications are spending a large amount of time waiting to obtain locks.

Troubleshooting

If the average lock wait time is high, you should look for applications that hold many locks or have lock escalations. If appropriate, you may need to tune such applications for better concurrency. If escalations are the reason for high average lock wait time, the values of one or both of the *locklist* and *maxlocks* database configuration parameters may be too low.

Users Page Statistics

The Users view includes the following statistics:

- Session Activity Summary
  - Connections
  - Connections Idle
  - Connections Active
  - Connections Waiting
  - Static SQL Statements
  - Dynamic SQL Statements

- Session Transaction Summary
  - Transactions
  - Transactions Per Second
  - Sorts Per Transaction
  - Lock Waits Per Transaction
  - Selects Per Transaction
  - Rows Selected Per Transaction
- Session Leaders – Memory
- Session Leaders – I/O
- Session Leaders – CPU
- Max. Connections
- Percent Executing
- Percent of Maximum

Related Topics

- Users Detail

Connections

The total connections metric is the number of applications currently connected to the database.

Metrics

You can use this metric to help you get an overview of the level of database activity and the amount of system resources in use.

Troubleshooting

This metric can help you adjust the setting of the *maxappls* and *max_coordagents* configuration parameters. For example, if this value is always the same as *maxappls*, you consider increasing the value of *maxappls*.

Connections Idle

The connections idle metric indicates the number of applications that are currently connected to the database for which the database manager is not executing any requests.

Metrics

You may use this element to help you understand the level of activity within a database and the amount of system resource being used.

Connections Active

The connections active statistic indicates the number of applications for which the database manager is currently executing requests.

Metrics

You can use this element to understand how many of the database manager agent tokens are being used by applications connected to this database.

Connections Waiting

This metric indicates the percentage of all connected applications waiting on locks.

Metrics

If this number is high, the application may have concurrency problems.

Troubleshooting

You can identify the applications that are holding locks or exclusive locks for a long time from the Users Detail>Locks page and tune such applications for better concurrency.

Static SQL Statements

The static SQL metric is the number of static SQL statement executions attempted on the database each second.

Dynamic SQL Statements

The dynamic SQL metric is the number of dynamic SQL statement executions being attempted on the database each second.

Transactions

Transactions is the number of transactions executed on the database since the start of this instance of Performance Center.

Metrics

A low number of units of work compared to the overall SQL activity (static + dynamic SQL statements) indicate long transactions. This may in turn be an indicator of poor concurrency and heavy log usage.

Transactions Per Second

Transactions per second is the number of transactions (units of work) completed per second on the database.

Metrics

A small rate of transactional activity on the database can indicate that applications are not doing frequent commits, which may lead to logging and concurrency problems.

Troubleshooting

Drill down to the Users Detail>SQL Activity page to check which applications are running their transactions for long periods of time.

Sorts Per Transaction

Sorts per transaction is the number sorts that have been executed during per transaction.

Lock Waits Per Transaction

Lock Waits Per transaction is the number of times per transaction applications or connections had to wait because another connection was holding a lock on the data.

Selects Per Transaction

Selects per transaction is the number of SQL SELECT statements executed per transaction.

Rows Selected Per Transaction

Rows selected per transaction is the number of rows per transaction that have been selected and returned to the application.

Session Leaders – Memory

Session Leaders – Memory shows the top five memory consumption applications that are running in the database. The details include:

Agent ID: A system-wide unique ID for the application. On a single-partitioned database, this identifier consists of a 16 bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16 bit counter. In addition, this identifier will be the same on every partition where the application may make a secondary connection.

Auth ID – The authorization ID of the user who invoked the application that is being monitored. On a DB2 Connect gateway node, this is the user's authorization ID on the host.

Memory: Total memory pool usage for the application.

Session Leaders – I/O

Session Leaders – I/O shows the top five applications consuming the most I/O. The details include:

Agent ID – A system-wide unique ID for the application. On a single-partitioned database, this identifier consists of a 16 bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16 bit counter. In addition, this identifier will be the same on every partition where the application may make a secondary connection.

Auth ID – The authorization ID of the user who invoked the application that is being monitored. On a DB2 Connect gateway node, this is the user's authorization ID on the host.

Total I/O (ms) – The total amount time (in milliseconds) spent performing buffered and direct reads and writes.

Session Leaders – CPU

Session Leaders – CPU shows the top five applications with the highest CPU usage. The details include:

Agent ID: A system-wide unique ID for the application. On a single-partitioned database, this identifier consists of a 16 bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16 bit counter. In addition, this identifier will be the same on every partition where the application may make a secondary connection.

Auth ID – The authorization ID of the user who invoked the application that is being monitored. On a DB2 Connect gateway node, this is the user's authorization ID on the host.

CPU (sec): The total user and system CPU time used by the application agents.

Max. Connections

This parameter (*maxappls*) is the maximum number of concurrent applications that can be connected locally and remotely to a database. Allowing a higher number of concurrent applications to connect increases the potential for more memory use.

Metrics

If you set *maxappls* to *automatic*, any number of applications will be allowed to connect. Otherwise, you can set the maxappls parameter to a specific number that allows for all of the following:

- The sum of the connected applications.
- The number of these connected applications that might be concurrently in the process of completing a two-phase commit or rollback.
- The expected number of indoubt transactions that may exist at any given time.

Percent Executing

Percent Executing is the percentage of applications that are currently executing in the database, and for which the database manager is currently processing a request.

Percent of Maximum

Percent of Maximum is the percentage of the maximum number of allowed concurrent applications that are currently connected to the database.

Users Detail

The following tabbed pages are available on the Users Detail page:

Overview Tab

Column	Description
Agent ID	This is a system-wide unique indentifier for the application.
Auth ID	The authorization ID of the user who invoked the application that is being monitored.
Client PID	The process ID of the client application that made the connection to the database.
OS User ID	The authorization ID used to access the operating system.
Application	Name of the application executable.
Status	The current status of the application.
Coord Node	In a multi-node system, this is the node number of the node where the selected application connected or attached to the instance.
UOW Elapsed Time (sec)	The elapsed execution time of the most recently completed unit of work.
Memory Used (KB)	Total memory pool usage for the application.
Lock Wait Time (ms)	The current amount of wait time for the process, in milliseconds.
Total Sort Time (ms)	The total elapsed time (in milliseconds) for all sorts that have been executed.
User CPU Time (sec)	The total user CPU time used by the application agents.
System CPU Time (sec)	The total system CPU time used by the application agents.
Buffered I/O Time (ms)	The total time spent by application in performing buffered reads and writes.
Direct I/O Time (ms)	The total time spent by application in performing non-buffered reads and writes.

Locking Tab

Column	Description
Agent ID	This is a system-wide unique indentifier for the application.
Auth ID	The authorization ID of the user who invoked the application that is being monitored.
Client PID	The process ID of the client application that made the connection to the database.
OS User ID	The authorization ID used to access the operating system.
Application	Name of the application executable.
Status	The current status of the application.
Coord Node	In a multi-node system, this is the node number of the node where the selected application connected or attached to the instance.
Lock Wait Time (ms)	The current amount of wait time for the process, in milliseconds.
Lock Waits	At the database level, this is the total number of times that applications have had to wait for locks within this database. At the application-connection level, this is the total number of times that this connection requested a lock but had to wait because another connection was already holding a lock on the data.
Locks Waiting	The number of agents waiting on a lock.
Locks Held	The number of locks currently held either by all applications in the database (database level) or by all agents for the application (application level).
Deadlocks	The total number of deadlocks that have occurred since this instance of Performance Center started monitoring the database.

Column	Description
Timeouts	The number of times that a request to lock an object timed out without being granted.
Timeout Value (sec)	Indicates the timeout value (in seconds) when an application has issued a SET CURRENT LOCK TIMEOUT statement.
Lock Escalations	The number of times that locks have been escalated from row locks to table locks, since this instance of Performance Center started monitoring the database.
X Lock Escalations	The number of times that locks have been escalated from several row locks to one exclusive table lock, or the number of times an exclusive lock on a row caused the table lock to become an exclusive lock.

Sorting Tab

Column	Description
Agent ID	This is a system-wide unique indentifier for the application.
Auth ID	The authorization ID of the user who invoked the application that is being monitored.
Client PID	The process ID of the client application that made the connection to the database.
OS User ID	The authorization ID used to access the operating system.
Application	Name of the application executable.
Status	The current status of the application.
Coord Node	In a multi-node system, this is the node number of the node where the selected application connected or attached to the instance.
Total Sort Time (ms)	The total elapsed time (in milliseconds) for all sorts that have been executed.
Total Sorts	The total number of sorts that have been executed.
Sort Overflows	The total number of sorts that ran out of sort heap and may have required disk space for temporary storage.
Hash Joins	The total number of hash joins executed.
Hash Join Loops	The total number of times that a single partition of a hash join was larger than the available sort heap space.
Hash Join Overflows	The number of times that hash join data exceeded the available sort heap space.
Hash Join Small Overflows	The number of times that hash join data exceeded the available sort heap space by less than 10%.

I/O Tab

Column	Description
Agent ID	This is a system-wide unique indentifier for the application.
Auth ID	The authorization ID of the user who invoked the application that is being monitored.
Client PID	The process ID of the client application that made the connection to the database.
OS User ID	The authorization ID used to access the operating system.

Column	Description
Application	Name of the application executable.
Status	The current status of the application.
Coord Node	In a multi-node system, this is the node number of the node where the selected application connected or attached to the instance.
Total I/O Time (ms)	The total time spent by application in performing buffered and non-buffered reads and writes.
Buffered I/O Time (ms)	The total time spent by application in performing buffered reads and writes.
Direct I/O Time (ms)	The total time spent by application in performing non-buffered reads and writes.
Total Reads	The number of reads issued against the database.
Total Writes	The number of writes issued against the database.
Logical Reads	The total number of db block gets and consistent gets (data read from memory) since the last refresh.
Physical Reads	The total number of physical reads performed on all datafiles since the last refresh.
Direct Reads	This is the number of read operations that do not use the buffer pool.
Direct Writes	This is the number of write operations that do not use the buffer pool.
Buffered Writes	This indicates the number of times a buffer pool data page was physically written to disk.
Pre-Fetch Wait Time (ms)	This is the time an application spent waiting for an I/O server (prefetcher) to finish loading pages into the buffer pool.
Pre-Fetch Unread Pgs	This indicates the number of pages that the prefetcher read in that were never used.

Memory Tab

Column	Description
Agent ID	This is a system-wide unique indentifier for the application.
Auth ID	The authorization ID of the user who invoked the application that is being monitored.
Client PID	The process ID of the client application that made the connection to the database.
OS User ID	The authorization ID used to access the operating system.
Application	Name of the application executable.
Status	The current status of the application.
Coord Node	In a multi-node system, this is the node number of the node where the selected application connected or attached to the instance.
Memory Used (KB)	Total memory pool usage for the application.
Priv Work HWM (KB)	This is the largest size (in kilobytes) reached by the private workspace.
Shr Work HWM (KB)	This is the largest size (in kilobytes) reached by shared workspaces.
Cache Overflows	This is the number of times that the catalog cache overflowed the bounds of its allocated memory.
Cache Lookups	This is the number of times that the catalog cache was referenced to obtain table descriptor information or authorization information.
Cache Inserts	This is the number of times that the system tried to insert table descriptor or authorization information into the catalog cache.

Column	Description
Catlg Cache Hit Ratio	The catalog cache hit ratio indicates how well the catalog cache is helping to avoid actual accesses to the catalog on disk. A high ratio indicates successful avoidance of actual disk I/O accesses.
Pkg Cache Hit Ratio	The package cache hit ratio indicates how well the package cache is helping to avoid reloading packages and sections for static SQL from the system catalogs as well as helping to avoid recompiling dynamic SQL statements. A high ratio indicates successful avoidance of these activities.
Priv Work Hit Ratio	The private workspace hit ratio indicates how well the private SQL workspace is helping to avoid having to initialize sections for SQL statements that are about to be executed. A high ratio indicate successful avoidance of this action.
Shr Work Hit Ratio	The shared workspace hit ratio indicates how well the shared SQL workspace is helping to avoid having to initialize sections for SQL statements that are about to be executed. A high ratio indicate successful avoidance of this action.
Appl Sect Hit Ratio	The application workspace hit ratio indicates how well the application SQL workspace is helping to avoid having to initialize sections for SQL statements that are about to be executed. A high ratio indicate successful avoidance of this action.

SQL Activity Tab

Column	Description
Agent ID	This is a system-wide unique indentifier for the application.
Auth ID	The authorization ID of the user who invoked the application that is being monitored.
Client PID	The process ID of the client application that made the connection to the database.
OS User ID	The authorization ID used to access the operating system.
Application	Name of the application executable.
Status	The current status of the application.
Coord Node	In a multi-node system, this is the node number of the node where the selected application connected or attached to the instance.
Rows Read	This is the number of rows read from the table.
Rows Written	This is the number of rows changed (inserted, deleted, or updated) in the table.
Rows Read Per Tran	The average number of rows read per transaction.
Rows Written Per Tran	The average number of rows read per transaction.
Transactions	The total number of active transactions in that have not yet been committed, or are waiting on a blocking lock to complete.
Commits	This is the total number of commits initiated by the database manager.
Rollbacks	This is the total number of rollbacks initiated by the database manager.
Rows Selected	This is the number of rows that have been selected and returned to the application.
Rows Inserted	This is the number of row insertions attempted.
Rows Updated	This is the number of row updates attempted.
Rows Deleted	This is the number of row deletions attempted.
Select Stmts	This is the percentage of all executed statements that were SQL Select Statements.

Column	Description
DDL Stmts	This is the percentage of all executed statements that were SQL DDL Statements.
Failed Stmts	This is the percentage of all executed statements that were SQL Failed Statements.

Instance Page Statistics

Summary Tab

- Memory
  - Sort Heap Utilization
  - Monitor Heap Utilization
  - Sort Heap (MB)
  - Private Memory (MB)
- Sorts/Joins
  - Piped Sorts Requested
  - Piped Sorts Rejected
  - Post Threshold Sorts
  - Post Threshold Hash Joins
- FCM
  - Buffer Utilization
  - Request Block Utilization
  - Message Anchor Utilization
  - Connection Entry Utilization
- Instance Identification
  - Product Name
  - Service Level
  - Instance Type
  - Instance Name
  - Last Reset Time
  - Active Databases
  - Number of Nodes
  - Operating System

- Monitor Switches
  - Lock
  - Sort
  - Table
  - Bufferpool
  - Statement
  - Timestamp
  - Unit of Work
- Agent Statistics
  - Idle
  - Stolen
  - Requests
  - Registered
  - Request Overflows
  - Waiting for Tokens
  - Assigned from Pool
  - Created Empty Pool

Configuration Tab

- Database Configuration

Memory Pool Tab

- Memory Pools
- Memory Pool Utilization
- Memory Pool Size

Utilities Tab

- Instance Utilities

FCM Tab

- FCM Throughput
- FCM Resource Utilization
- Buffer Utilization
- Message Anchor Utilization
- Connection Entry Utilization
- Request Block Utilization

Other

- Agent Utilization
- Percent of Agents Stolen

Sort Heap Utilization

The amount of memory allocated for each sort may be some or all of the available sort heap size. Sort heap size is the amount of memory available for each sort as defined in the database configuration parameter *sortheap*. Shared sort memory utilization gives the percentage of the sort heap being used.

It is possible for a single application to have concurrent sorts active. For example, in some cases a SELECT statement with a subquery can cause concurrent sorts.

Metrics

Memory estimates do not usually include sort heap space. If excessive sorting is occurring, the extra memory used for the sort heap should be added to the base memory requirements for running the database manager. Generally, the larger the sort heap, the more efficient the sort. Typically the shared sort utilization should be less than or equal to 70%. You should consider tuning the database if you see a utilization value greater than this.

Troubleshooting

To bring the sort heap utilization to an acceptable level, use the following guidelines:

- Examine the queries you are running on the database to see if you can add indexes to columns frequently accessed in the WHERE clause. This minimizes the use of the sort heap.
- If you are using dynamic bitmaps or hash join buffers in your queries, or your queries frequently require large sorts, increase the *sortheap* size.
- If you adjust the *sortheap* size, also look at the *sheapthres* database manager configuration parameter to see if it too needs to be adjusted

If you are adjusting the *sortheap* size, you may also benefit from rebinding your packages since the optimizer takes this parameter into account when determining the access paths.

Monitor Heap Utilization

The Monitor Heap Utilization statistic measures the consumption of monitor heap memory based on the memory pool. The utilization is calculated from using the following equation (db2.pool_cur_size / db2.pool_max_size) * 100.

Metric

The rate of utilization is measured by looking at the current memory heap pool size being used in relation to the maximum memory heap pool size. When the percentage reaches the maximum 100%, monitor operations may fail.

Troubleshooting

If you are experiencing trouble, consider increasing the monitor heap memory size.

Sort Heap (MB)

The Sort Heap metric is the total number of allocated pages of sort heap space for all sorts at the database manager level and at the moment the snapshot was captured.

Metric

Sort heap size is the amount of memory available for each sort as defined in the *sortheap* database configuration parameter. At the database manager level, it is the total sort heap space allocated in all active databases in the database manager.

Troubleshooting

If excessive sorting is occurring, you can add the extra memory used for the sort heap to the base memory requirements. Often, the larger the sort heap, the more efficient the sort. Use the information returned at the database manager level to help you adjust the *sheapthres* parameter.

Private Memory (MB)

This is the amount of private memory that the instance of the database manager has allocated at the time the snapshot was taken.

To change the amount of private memory allocated, change the minimum committed private memory configuration parameter (*min_priv_mem*). The default value is recommended, but you can use it to commit more memory to the database server. If you set the allocation too high, you can have a negative impact on the performance of non-DB2 applications.

Piped Sorts Requested

This metric gives the number of piped sorts that have been requested.

Metrics

Each active sort on the system allocates memory, which may result in sorting taking up too much of the available system memory. A piped sort is not accepted if the sort heap threshold is exceeded when the sort heap is allocated for the sort.

The sort list heap (*sortheap*) and sort heap threshold (*sheapthres*) configuration parameters help control the amount of memory used for sort operations. These parameters are also used to determine whether a sort will be piped.

Since piped sorts may reduce disk I/O, allowing more piped sorts can improve the performance of sort operations and possibly the performance of the overall system.

Troubleshooting

If piped sorts are being rejected, you might consider decreasing your sort heap or increasing your sort heap threshold. You should be aware of the possible implications of either of these options. If you increase the sort heap threshold there is the possibility that more memory will remain allocated for sorting. This could cause paging memory to disk. If you decrease the sort heap, you might require an extra merge phase that could slow down the sort.

Piped Sorts Rejected

The piped sorts rejects statistic is the number of piped sorts that were rejected by the database manager.

Metrics

Each active sort on the system allocates memory, which may result in sorting taking up too much of the available system memory. A piped sort is not accepted if the sort heap threshold is exceeded when the sort heap is allocated for the sort.

The sort list heap (*sortheap*) and sort heap threshold (*sheapthres*) configuration parameters help control the amount of memory used for sort operations. These parameters are also used to determine whether a sort will be piped.

Since piped sorts may reduce disk I/O, allowing more piped sorts can improve the performance of sort operations and possibly the performance of the overall system.

Troubleshooting

If piped sorts are being rejected, you might consider decreasing your sort heap or increasing your sort heap threshold. You should be aware of the possible implications of either of these options. If you increase the sort heap threshold there is the possibility that more memory will remain allocated for sorting. This could cause paging memory to disk. If you decrease the sort heap, you might require an extra merge phase that could slow down the sort.

Post Threshold Sorts

The post threshold sorts is the number of sorts that have requested heaps after the sort heap threshold has been exceeded.

Metrics

Under normal conditions, the Database Manager allocates sort heap using the value specified by the *sortheap* configuration parameter. If the amount of memory allocated to sort heaps exceeds the sort heap threshold (*sheapthres* configuration parameter), the database manager allocates sort heap using a value less than that specified by the *sortheap* configuration parameter.

Each active sort on the system allocates memory, which may result in sorting taking up too much of the system memory available. Sorts that start after the sort heap threshold has been reached may not receive an optimum amount of memory to execute. As a result, however, the entire system may benefit.

Troubleshooting

By modifying the sort heap threshold and sort heap size configuration parameters, the performance of sort operations and/or the overall system can be improved. If this element's value is high, you can:

- Increase the sort heap threshold (*sheapthres*) or,
- Adjust applications to use fewer or smaller sorts via SQL query changes.

Post Threshold Hash Joins

The post threshold hash joins statistic is the number of times that a hash join heap request was limited because of concurrent use of shared or private sort heap space.

Metrics

Each active sort on the system allocates memory, which may result in sorting taking up too much of the available system memory. A piped sort is not accepted if the sort heap threshold is exceeded when the sort heap is allocated for the sort.

The sort list heap (*sortheap*) and sort heap threshold (*sheapthres*) configuration parameters help control the amount of memory used for sort operations. These parameters are also used to determine whether a sort will be piped.

Since piped sorts may reduce disk I/O, allowing more piped sorts can improve the performance of sort operations and possibly the performance of the overall system.

Troubleshooting

If this value is large (greater than 5% of hash join overflows), the sort heap threshold should be increased.

Buffer Utilization

This element indicates the percentage of all FCM buffers that are currently being used by the fast communication manager.

Metrics

None.

Troubleshooting

You can use this information to tune *fcm_num_anchors.* If the utilization percentage is high, you should increase the *fcm_num_anchors* to ensure that operations do not run out of FCM message anchors. If the utilization is low, you can decrease *fcm_num_anchors* to conserve system resources.

Request Block Utilization

This element indicates the percentage of all request blocks that are currently being used by the fast communication manager.

NOTE: This metric is only applicable to DB2 version 7. In DB2 version 8, the maximum request blocks are adjusted dynamically and automatically

Troubleshooting

You can use this information to tune *fcm_num_rqb.* If the utilization percentage is high, you should increase the *fcm_num_rqb* to ensure that operations do not run out of FCM request blocks. If the utilization is low, you can decrease *fcm_num_rqb* to conserve system resources.

Message Anchor Utilization

This element indicates the percentage of all message anchors that are currently being used by the fast communication manager.

NOTE: This metric is only applicable to DB2 version 7. In DB2 version 8, the maximum message anchors are adjusted dynamically and automatically

Metrics

None.

Troubleshooting

You can use this information to tune *fcm_num_buffers.* If the utilization percentage is high, you should increase the *fcm_num_buffers* to ensure that operations do not run out of FCM buffers. If the utilization is low, you can decrease *fcm_num_buffers* to conserve system resources.

Connection Entry Utilization

The connection entry utilization element indicates the percentage of all connection entries that are currently being used by the fast communication manager.

NOTE: This metric is only applicable to DB2 version 7. In DB2 version 8, the maximum connection entries are adjusted dynamically and automatically.

Troubleshooting

You can use this information to tune *fcm_num_connect.* If the utilization percentage is high, you should increase the *fcm_num_connect* to ensure that operations do not run out of FCM connection entries. If the utilization is low, you can decrease *fcm_num_connect* to conserve system resources.

Instance Identification

The following information about the database instance is displayed.

Product Name: The name of the database software.

Service Level: The FixPack level.

Instance Type: The type of DB2 server.

Instance Name: The name of the DB2 instance.

DB2 Start Time: The date and time that the database manager was started using the `db2start` command.

Active Databases: The number of active databases running on the server.

Number of Nodes: The number of nodes running on the instance.

Operating System: The operating system the database is running on.

Monitor Switches

Monitor switches control the collection of potentially expensive data by the database manager. Each switch can be set to ON or OFF.

Lock

The lock monitor switch controls the collection of data related to the lock wait times and deadlocks.

Sort

The sort monitor switch controls the collection of data related to the number of heaps used and sort performance.

Table

The table monitor switch controls the collection of data related to the measure of activity (rows read/written).

Bufferpool

The bufferpool monitor switch controls the collection of data related to the number of reads and writes and the amount of time taken.

Statement

The statement monitor switch controls the collection of data related to the start/stop time and statement identification.

Timestamp

The timestamp monitor switch controls the collection of timestamps. This switch is ON by default.

Unit of Work

The unit of work monitor switch controls the collection of data related to start and end times and completion status.

Idle

An idle agent is one type of worker agent, that is an agent that carries out application requests but that has no fixed attachment to any given application. More specifically, an idle agent doesn't have a local database or outbound connection.

Metrics

The idle agents metric is the number of agents in the agent pool that are currently unassigned to an application and are, therefore, "idle."

Having idle agents available to service requests for agents can improve performance.

Troubleshooting

You can use this element to help set the *num_poolagents* configuration parameter.

Stolen

This is the number of times agents are stolen. Agents are stolen when an idle agent is reassigned from one application to another.

Metric

Agents stolen can be used along with associate_agents_top to evaluate the load the application places on the system.

Troubleshooting

If *agents_stolen* is high, try increasing the *num_poolagents* configuration parameter.

Requests

This metric is the number of requests for agents from the agent pool. As requests are made, idle agents are deployed.

Troubleshooting

Adjust the *num_poolagents* configuration parameter if there are there are more requests than agents available.

Registered

The agents registered metric is the number of agents registered in the Database Manager instance that is being monitored (coordinator agents and subagents).

Metrics

None.

Troubleshooting

You can use this element to help evaluate your setting for the *maxagents* configuration parameter.

Request Overflows

Request Overflows is the maximum number of agents associated with an application. The *max_agent_overflows* metric is the number of requests received after the *maxagents* configuration parameter has been reached.

Waiting for Tokens

The agents waiting for tokens statistic is the number of agents waiting for a token so they can execute a transaction in the database manager.

Metrics

Each application has a dedicated coordinator agent to process database requests within the Database Manager. Furthermore, each agent has to get a token before it can execute a transaction. The maximum number of agents that can execute Database Manager transactions is limited by the configuration parameter *maxcagents*.

Troubleshooting

You can use this element to help evaluate your setting for the *maxcagents* configuration parameter.

Assigned from Pool

This is the number of agents that have been assigned from the pool of agents available.

Troubleshooting

Use the *agents_created_empty_pool* in conjunction with the *agents_from_pool* metric to determine how often an agent has to be created because the pool is empty. If you see the ratio (Agents Created Due to Empty Agent Pool / Agents Assigned from Pool) is high, consider increasing the *num_poolagents* configuration parameter.

A low ratio suggests *num_poolagents* is set too high and rarely used agents are wasting resources. A high ratio may indicate the workload is too high for that node.

Created Empty Pool

This is the number of agents created because the agent pool ran dry. The metric includes the number of agents at startup (*num_intiagents*).

Troubleshooting

Use the *agents_created_empty_pool* in conjunction with the *agents_from_pool* metric to determine how often an agent has to be created because the pool is empty. If you see the ratio (Agents Created Due to Empty Agent Pool / Agents Assigned from Pool) is high, consider increasing the *num_poolagents* configuration parameter.

A low ratio suggests *num_poolagents* is set too high and rarely used agents are wasting resources. A high ratio may indicate the workload is too high for that node.

Database Configuration

The following information about the Database Manager (Instance) is available on the configuration tab:

Parameter: The parameter name.

Description: The description of the parameter.

Value: The value set for the parameter.

Modifiable: Whether the parameter is modifiable or not.

Memory Pools

This section shows the memory pool usage details of the selected memory pool over time.

- **Node:** Indicates the database partition number.
- **Memory Pool:** This is the kind of memory pool. Each pool type is only shown once.
- **Utilization:** The percentage of utilization of the selected memory pool.
- **High Watermark (MB):** The largest size of a memory pool since its creation.
- **Current Size (MB):** The current size of a memory pool.
- **Max Size (MB):** The internally configured size of a memory pool in DB2 UDB.

Metrics

The nature of memory_pool data elements varies between platforms. On Windows systems, no memory usage is reported at the database level, while on UNIX systems, memory is reported at the database level. Instead of reporting this memory at the database level, the system monitor for Windows systems reports it in instance -level snapshots. This divergence in reporting is due to differences in the underlying memory architecture between Windows systems and UNIX systems.

Memory Pool Utilization

This section shows the percentage of utilization of the selected memory pool.

Metrics

You can use this metric to see if a memory pool is nearly full. You can diagnose the problems with specific memory pools by monitoring their utilization over time.

Troubleshooting

If the value of the pool utilization is consistently close to or exceeds 100%, you may need to increase the configuration parameter associated with that pool.

Memory Pool Size

This section shows the current memory pool size for the selected application (or database level).

Instance Utilities

The Utilities tab of the Instance Page gives information on the utilities currently executing in the database manager (available in DB2 version 8 only). The Utilitity details are as follows:

Node: Indicates the database partition number.

Database: Database on which the utility is operating.

ID: The unique identifier corresponding to the utility invocation.

Type: Class of the utility (Rebalance, Backup, Restore, Reorg, etc.).

Description: A brief description of the work the utility is performing.

Start Time: The time the utility was invoked.

% Complete: Shows the completion status of the utility.

FCM Throughput

This section gives a detailed view of inter-node communication in a multipartition environment. The information shown here includes the connection status and buffers sent and received between various combinations of partitions in a multi-partitioned environment. The columns presented here include:

Source Node: Partition that sends the information.

Target Node: Partition that received the information.

Buffers Sent: Number of Buffers sent from source node to the target node.

Buffers Received: Number of Buffers sent from source node to the target node.

Connection Status: Status of the connection between nodes.

Metrics

If the total number of FCM buffers sent or received between selected nodes is high, you may want to redistribute the database, or move tables to reduce the inter-node traffic.

FCM Resource Utilization

The FCM Resource Utilization chart shows the following:

Node: Indicates the database partition number.

FCM Buffer Utilization: The current utilization and low watermark of FCM buffers.

FCM Message Anchor Utilization: The current utilization and low watermark of FCM message anchors.

FCM Connection Entries Utilization: The current utilization and low watermark of FCM connection entries.

FCM Request Block Utilization: The current utilization and low watermark of FCM request blocks.

Agent Utilization

Agent Utilization is the percentage of available database manager agents (maxagents) that are currently registered on the database manager (agents_registered).

Percent of Agents Stolen

Percent of Agents Stolen is the percentage of all agent requests—including the number of agents assigned from the agent pool (agents_from_pool), the number of agents created because the agent pool was empty (agents_created_empty_pool), and the number of agents stolen (agents_stolen)—that were agents stolen.

Other Views and Statistics

In addition to the Home view, Enterprise view and the performance category views, Embarcadero Performance Center offers many other views. The tables below lists, by database platform, the other views available in Embarcadero Performance Center:

View	Oracle	SQL Server	Sybase	DB2
Alert Log	x	x	x	x
Archive	x			
Configuration Parameters		x	x	x
Health Index	x	x	x	x
Hot Objects	x			
Instance Parameters	x			
Lock	x	x	x	x
Operating System	x	x	x	x
Session Detail	x	x	x	x
SQL Server Logs		x		
Top Sessions	x	x	x	x
Top SQL	x	x		x
Trends	x	x	x	x

Archive View

- Metrics

To allow for point-in-time recovery, Oracle writes copies of redo log information to disk. When a database is running in archive log mode, a DBA, with proper backup techniques in place, can recover nicely from a database error and roll forward to almost any point in time as long as the proper archive logs are in place.

The I/O needed to write these archive logs is handled by Oracle's ARCH process. The Archive view allows archive files written by the ARCH process to be viewed by user-specified time frames. The table below describes the information available on the Archive view:

Column	Description
Date/Time	The timestamp of the archive log (when the log was written).
Title	The actual archive log file name and path.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Numerous archive files can be written to disk if there is heavy redo log activity. Batch jobs have the potential to move very fast; sometimes so fast that the online redo logs wrap back around before they have a chance to be archived. Messages indicating this has happened show up in the Oracle alert log. If this happens frequently, you should think about increasing the size of the online redo log files, or increasing the number of redo logs in general.

Seeing archive files written at a rate of more than one every 30-60 minutes can indicate the redo size is too small (or there is an above-average data modification load).

If you do not want to lose an archive file that can be needed for recovery and you are using Oracle 8 or later, you can take advantage of the feature that lets you write archive files to more than one destination on disk. This feature also allows multiple ARCH processes to be invoked. Investigate using the Init.ora parameters log_archive_dest_n and log_archive_max_processes.

Always remember one thing with respect to archive files and running Oracle in archive log mode: Running out of archive file space on the server can halt all activity in a database. Make sure you have ample free space available on your archive drives. And, you should also implement a purge procedure for older archives in conjunction with your backup routine.

Health Index View

- Metrics

Embarcadero Performance Center's global and category-specific health indexes are fast and efficient indicators you can use to determine if a database is experiencing problems. They also locate the most problematic area(s). With the Health Index view, you can scan individual category indexes simultaneously and see, over time, where the problem areas reside.

Metrics

Generically speaking, you should investigate any index that falls below 90. Temporary dips in a health index graph should not be a cause for concern unless the dips form a pattern and occur on a predictable and continuous basis.

Hot Objects

The following tabbed pages are available on the Hot Objects view:

- Hot Tables
- Hot Code

NOTE: The Hot Objects view is for Oracle datasources.

Hot Tables

- Metrics

Certain objects in an Oracle database are accessed more than others. These objects can become a source of contention given certain conditions. The Hot Tables tab of the Hot Objects view identifies tables that are being frequently accessed through various SQL statements. The table below describes the information that Performance Center displays on the Hot Tables tab of the Hot Objects view:

Column	Description
Table Owner	The owner of the table.
Table Name	The name of the table.
Command Issued	The SQL statement command issued against the table.
Executions	The number of SQL executions the object has experienced.

Column	Description
Disk Reads	The number of estimated disk reads from the object.
Buffer Gets	The number of estimated logical I/O's from the object.
Rows Processed	The number of estimated rows processed from the object.

Metrics

DML activity against tables can cause contention for space management objects like free lists. Oracle9i and above provides automatic segment management, which can remove problems with free lists and the like.

Hot Code

- Metrics

Certain objects in an Oracle database are accessed more than others. Data objects can become a source of contention given certain conditions, while code objects rarely cause contention issues. The Hot Code tab of the Hot Objects view identifies code objects (procedure, functions, etc.) that are being frequently accessed through various SQL statements. The table below describes the information that Performance Center displays on the Hot Code tab of the Hot Objects view:

Column	Description
Object Owner	The owner of the object.
Object Name	The name of the object.
Object Type	The type of object (package, etc.)
# of Executions	The number of estimated executions for the object.
Loads	The number of times the object was loaded into the shared pool.
Locks	The number of locks the object has experienced.
Pins	The number of times the object was pinned in the shared pool.

Metrics

Often referenced code objects should be pinned in the shared pool using the Oracle DBMS_SHARED_POOL package. Objects with many executions and loads should be considered candidates for pinning.

Lock View

The following tabbed pages are available on the Lock view:

- All Locks
- All User Locks (Oracle only)
- Blocking Locks
- Locks View for DB2

All Locks Tab

The information on the All Locks tab of the Locks view depends on the target DBMS:

- Oracle
- SQL Server
- Sybase

All Locks Tab for Oracle

- Metrics

To modify database information or structures, a user session must obtain a lock on the object to perform its task. In addition to user locks, Oracle issues lock requests to carry out its internal duties. The All Locks tab of the Locks view displays information about all locks currently on a system. The table below describes the information available on the All Locks tab of the Locks view for Oracle:

Column	Description
SID	The session identifier of the session holding the lock.
User Name	The user account of the session holding the lock. NULL if it is a background process.
Lock Mode	The lock mode (EXCLUSIVE, SHARE, etc.)
Request Type	The type of lock requested by the session.
Object Name	The name of the object being locked.
Object Type	The type of object being locked (TABLE, etc.)
Lock Type	The type of lock (TRANSACTION, DML, etc.)
Lock ID 1	The lock identifier #1 (depends on type).
Lock ID 2	The lock identifier #2 (depends on type).

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Locks held for unusually long periods are candidates for further investigation. The application logic can be inefficient or the program may not be issuing COMMIT frequently enough.

All Locks Tab for SQL Server

- Metrics

To modify database information or structures, a user session must obtain a lock on the object to perform its task. In addition to user locks, SQL Server issues lock requests to carry out its internal duties. Embarcadero Performance Center displays information about all locks currently on a system on the All Locks tab of the Locks view and the Lock Detail grid on the Lock tab of the Contention Detail view.

The table below describes the information available on the tab and the grid:

Column	Description
SPID	The process ID of the process holding the lock.
Login	The login name of the process.
NT User	The operating system name of the process.
Database	The database where the locks are occurring.
Table Name	The name of the table involved in a lock. This will be NULL for non-table locks or table locks that take place in the tempdb database.
Ndx ID	The index ID involved in the lock.
Lock Type	The type of lock (database, table, row ID, etc.)
Lock Mode	The mode of the lock (shared, exclusive, etc.)
Lock Status	The status of the lock (waiting or granted).
Owner Type	Whether the lock came from a regular session or a transaction.
Program	The executable the process is using against the server.
BLK SPID	If nonzero, the process ID of the process blocking the requested lock. A value of zero indicates that the process is not blocked.
Wait Time	The time the process has waited for the lock, in milliseconds.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Command	The command the process is currently issuing.
NT Domain	The name of the Windows 2000/NT domain.

NOTE: The information in the Lock Detail grid is available in the Lock Detail grid on the Locks tab of the Contention Detail view and the All Locks tab of the Locks view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Locks held for unusually long periods are candidates for further investigation. The application logic may be inefficient or the program may not be issuing COMMIT frequently enough.

All Locks Tab for Sybase

- Metrics

To modify database information or structures, a user session must obtain a lock on the object to perform its task. In addition to user locks, Sybase issues lock requests to carry out its internal duties. The All Locks tab of the Locks view displays information about all locks currently on a system. The table below describes the information available on the All Locks tab of the Locks view for Sybase:

Column	Description
PID	The process ID.
User Name	The user name assigned to the process.
Database	The database in which the process is running.
Lock Type	The type of lock (database, table, row ID, etc.)
Object Name	The name of the object being locked.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Lock Page	The page number where the lock is currently applied.
Lock Class	The name of the cursor the lock is associated with (if any).
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Command	The command the process is currently issuing.
CPU Time	The CPU time accumulated for the current command.
Physical I/O	The current cumulative number of reads and writes issued by the process.
Mem Usage	The memory accumulated for the current command.
FID	The process ID of the worker process' parent.
Transaction	The name of any transaction.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Locks held for unusually long periods are candidates for further investigation. The application logic may be inefficient or the program may not be issuing COMMIT frequently enough.

All User Locks Tab

- Metrics

To modify database information or structures, a user session must obtain a lock on the object to perform its task. The All User Locks tab of the Locks view displays information about all user locks currently on a system. The table below describes the information available on the All User Locks tab of the Locks view:

Column	Description
User Name	The user account that holds the lock.
Terminal	The machine name of the client session.
SID	The unique Oracle identifier for the session.
Serial #	The serial number of the lock.
Table	The name of the object being locked.
Lock Mode	The lock mode (EXCLUSIVE, SHARE, etc.)
Request	The type of lock requested by the session.

NOTE: The All User Locks tab of the Locks view is only available for Oracle datasources.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Locks held for unusually long periods are candidates for further investigation. The application logic may be inefficient or the program may not be issuing COMMITs frequently enough.

Blocking Locks Tab

The information on the Blocking Locks tab of the Locks view depends on the target DBMS.

- Oracle
- SQL Server
- Sybase

Blocking Locks Tab for Oracle

- Metrics

Blocking-lock situations can make the database appear frozen, rivalling only a stuck archive in effect. A single blocking user has the potential to stop work for nearly all other processes on a small system, or can cause major headaches on large systems. Although Oracle supports unlimited row-level locking, blocking-lock situations do occur - sometimes frequently.

The Blocking Locks tab of the Locks view contains information relating to user accounts that are currently blocked and the sessions that are blocking them. The table below describes the information available on the Blocking Locks tab of the Locks view for Oracle:

Column	Description
Blocked SID	The session ID of the session waiting for the lock.
Blocked User	The user account of the session waiting for the lock.
Wait Time (sec)	The current wait time for the session, in seconds.
Blocking SID	The session ID of the session holding the offending lock.
Blocking User	The user account of the session holding the offending lock.
Lock Type	The type of lock (TRANSACTION, DML, etc.)
Lock Mode	The lock mode (EXCLUSIVE, SHARE, etc.)
Request Type	The type of lock being requested by the session.
Locked Object	The name of the object being locked.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

When a blocking lock is discovered, the DBA can quickly remedy the situation by issuing a KILL against the offending process. This eliminates the user's stranglehold on the objects the user was accessing. Other user processes then usually complete in an instant. Tools like Embarcadero Performance Center make it easier to discover the blocking-lock situation.

The culprit of blocking-lock scenarios is often the application design, or the SQL used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. Most DBAs who have had to face Oracle Forms applications have suffered through the dreaded SELECT…FOR UPDATE statements that place unnecessary restrictive locks on nearly every read operation. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

Data warehouses, whose data is mostly read, can benefit from tablespaces set in read-only mode. Read-only mode signals to the other databases that exclusive locks need not be used for the data contained within the tablespace. This is especially helpful in Oracle Parallel Server environments and drastically reduces ping activity.

Blocking Locks Tab for SQL Server

- Metrics

Embarcadero Performance Center displays information about all blocking locks currently on a system on the Blocking Locks tab of the Locks view and the Blocking Lock Detail grid on the Blocking Lock tab of the Contention Detail view.

The table below describes the information available on the tab and the grid:

Column	Description
SPID	The process ID of the process holding the lock.
Login	The login name of the process.
NT User	The operating system name of the process.
Database	The database where the locks are occurring.
Table Name	The name of the table involved in a lock. This will be NULL for non-table locks or table locks that take place in the tempdb database.
Ndx ID	The index ID involved in the lock.
Lock Type	The type of lock (database, table, row ID, etc.).
Lock Mode	The mode of the lock (shared, exclusive, etc.).
Lock Status	The status of the lock (waiting or granted).
Owner Type	Whether the lock came from a regular session or a transaction.
Program	The executable the process is using against the server.
BLK SPID	If nonzero, the process ID of the process blocking the requested lock. A value of zero indicates that the process is not blocked.
Wait Time	The time the process has waited for the lock, in milliseconds.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Command	The command the process is currently issuing.
NT Domain	The name of the Windows 2000/NT domain.

NOTE: The information in the Blocking Lock Detail grid is available in the Blocking Lock Detail grid on the Blocking Locks tab of the Contention view and the Blocking Locks tab of the Locks view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

When a blocking lock is discovered, the DBA can quickly remedy the situation by issuing a KILL against the offending process. This eliminates the user's stranglehold on the objects the user was accessing. Other user processes then usually complete in an instant. Tools like Embarcadero Performance Center make it easier to discover the blocking-lock situation.

Blocking Locks Tab for Sybase

- Metrics

The Blocking Locks tab of the Locks view contains information relating to user accounts that are currently blocked and the sessions that are blocking them. The table below describes the information available on the Blocking Locks tab of the Locks view for Sybase:

Column	Description
Holding PID	The process ID that owns the blocking lock.
Holding User	The user account of the session holding the offending lock.
Waiting PID	The session PID of the session waiting for the lock.
Waiting User	The user account of the session waiting for the lock.
Database	The database in which the process is running.
Object Name	The table on which the lock is being held.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Lock Type	The type of lock being applied.
Time Blocked	The time that a process has been waiting for the lock, in seconds.
Lock Page	The page number where the lock is currently applied.
Lock Class	The name of the cursor the lock is associated with (if any).
Holding Host	The name of the host computer with the blocking lock.
Waiting Host	The name of the host computer waiting for the lock.
Holding Program	The program the process is running that has the lock.
Waiting Program	The program the process is running that is waiting for the lock.
Holding Command	The command being issued by the process holding the lock.
Waiting Command	The command being issued by the process waiting for the lock.
CPU Time	The CPU time accumulated for the current command.
Physical I/O	The physical I/O accumulated for the current command.
Mem Usage	The memory accumulated for the current command.
Holding FID	The process ID of the worker process' parent that has the lock.
Waiting FID	The process ID of the worker process' parent that is waiting for the lock.
Transaction	The name of the associated transaction (if any).

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

When a blocking lock is discovered, the DBA can quickly remedy the situation by issuing a KILL against the offending process. This eliminates the user's stranglehold on the objects the user was accessing. Other user processes then usually complete in an instant. Tools like Embarcadero Performance Center make it easier to discover the blocking-lock situation.

Locks View for DB2

The Locks view displays all processes that are currently holding locks on an IBM DB2 UDB database. The following sections of this view are available to display lock information:

- Applications

Applications

This section lists the following lock information for all applications:

Agent ID – The application handle of the agent holding a lock for which this application is waiting.

Auth ID – The authorization ID of the user who invoked the application that is being monitored.

OS User ID – The authorization ID used to access the operating system.

Client PID – The process ID of the client application that made the connection to the database.

Application – Name of the application executable.

Status – The lock's status (waiting or granted).

Locks Held – The number of locks on the lock being held.

Locks Waiting – Indicates the number of agents waiting on a lock

Lock Wait Time (ms) – The current amount of wait time for the process, in milliseconds.

Timeouts – The number of times that a request to lock an object timed out without being granted.

Deadlocks – Processes that cannot proceed because they are waiting on a set of resources held by each other or held by other processes.

Locks Held Tab

This tab displays all the locks held by the selected application in the Applications list. The following data is available:

Lock Mode – The type of lock being held.

Object Type – The type of object against which the application holds a lock (for object-lock-level information), or the type of object for which the application is waiting to obtain a lock (for application-level and deadlock-level information).

Table Schema – Schema of the table that the lock is on.

Table Name – Name of the table that the lock is on. This element is only set if Object Type indicates Table.

Tablespace – The name of the table space against which the lock is held.

Lock Status – The lock's status (waiting or granted).

Escalation – Indicates whether a lock request was made as part of a lock escalation.

Locks Waiting Tab

This tab displays all the locks waiting by the selected application in the Applications list. The following data is available:

Agent ID – The application handle of the agent holding a lock for which this application is waiting.

Application ID – The application ID of the application that is holding a lock on the object that this application is waiting to obtain.

Lock Mode – The type of lock being held.

Mode Requested – The lock mode being requested by the application.

Object Type – The type of object against which the application holds a lock.

Table Schema – Schema of the table that the lock is on.

Table Name – Name of the table that the lock is on. This element is only set if Object Type indicates Table.

Tablespace – The name of the table space against which the lock is held.

Wait Start Time – The date and time that this application started waiting to obtain a lock on the object that is currently locked by another application

Escalation – Indicates whether a lock request was made as part of a lock escalation.

Unit of Work Tab

This tab displays the SQL statement text for the selected application. The statement is available if the selected application is in lock wait status or is the thread blocking other applications. This enables you to easily identify what SQL statements are causing lock wait conditions and to help diagnose deadlock scenarios.

Click **Explain SQL** to view an explain plan for the statement.

Operating System View

The Operating System view displays vital operating system statistics on the following tabbed pages:

- Summary
- CPU
- Processes
- I/O
- Memory
- Space
- Network

To use integrated security, or gather certain operating system statistics, there are two main things to know:

1 You must enable the Embarcadero Performance Center Server for operating system monitoring. To enable it, you must select the Enable operating system monitoring check box on the Machine tab of the Datasource Properties dialog box.

2 You must supply the credentials necessary to view these statistics. To be able to view performance counters on a remote computer, Microsoft requires specific permissions on the remote computer that you want to monitor.

   Because the Embarcadero Performance Center Server collects data using the registry, monitoring a remote computer requires the use of the Remote Registry Service. If the service stops due to failure, the system restarts it automatically only once. Therefore, if the service stops again, you must manually restart it. You can change this default behavior by modifying the properties for Remote Registry Service. To access service properties, see Services under Services and Applications in Computer Management or see Administrative Tools. You can also check the Event Viewer's Application and System Logs for events that might have stopped the service.

   Remote data collection also requires access to specific registry subkeys and system files. To provide remote access to the registry to collect data on remote systems, Microsoft requires that users have a minimum of Read access to the Winreg subkey in HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurePipeServers. By default, members of the Administrators group have Full Control access and members of the Backup Operators group have Read access. Microsoft also requires that users have Read access to the registry subkey that stores counter names and descriptions used by System Monitor, HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Perflib*LanguageID*, where *LanguageID* is the numeric code for the spoken language for the operating system installation. (For English, the subkey is Perflib\009.) By default, Microsoft gives Full Control access to the System account and members of the Administrators and Creator Owners groups. Therefore, a local user on a server who is not logged in as an administrator cannot see performance counters.

   In addition, users might also require read access to the files that supply counter names and descriptions to the registry, Perfc*.dat and Perfh*.dat. (where the asterisk is a wildcard character representing the specific language code; for English, these are Perfc009.dat and Perfh009.dat.) If these files reside on an NTFS volume, to have access to them, the access control lists (ACLs) on these files must specify that the user has such access. By default, members of the Administrators and Interactive groups have sufficient access.

OS Page Statistics

In many scenarios, an optimally tuned database may not perform well because there are constraints imposed by the system where the database is running. These constraints may include processes competing with the database sever for resources (CPU, I/O, or Memory), a slow CPU, insufficient or slow I/O devices, and insufficient memory. The OS Statistics page of Performance Center lets you examine operating system metrics for the following platforms:

- AIX
- HP-UX

  NOTE: To view processor info and swap disk info on an HP-UX box, you need to login as ROOT in the OS login.

- Linux
- Solaris
- Unix
- Windows XP, 2000, and NT

Summary Tab

The OS Summary tab displays the following statistics to communicate the general overall performance levels of the operating system:

- Disk Time
- Load Average
- Processor Time
- Paged Memory Used (Windows)
- Swap Memory Used (AIX, HP-UX, Linux, Solaris, Unix)
- Average Disk Queue
- Network Output Queue (Windows)
- Network Queue (Solaris)
- Page Faults/Sec
- Processor Queue
- Processor Speed
- Processor
- Available Paged Memory (Windows)
- Available Physical Memory
- Available Swap Memory (AIX, HP-UX, Linux, Solaris, Unix)
- Total Paged Memory (Windows)
- Total Physical Memory
- Total Swap Memory (AIX, HP-UX, Linux, Solaris, Unix)
- Free Disk Space
- Total Disk Space
- Used Disk Space
- Number of Logins
- Number of Processes
- Number of Processors
- Top CPU Process
- Top I/O Process
- Top Memory Process

Processor Time

The Processor Time statistic indicates the percentage of time the processor is working. This counter is a primary indicator of processor activity.

Metrics

If your computer seems to be running sluggishly, this statistic could be displaying a high percentage.

Troubleshooting

Upgrade to a processor with a larger L2 cache, a faster processor, or install an additional processor.

Processor Speed

The Processor Speed statistic displays the speed of the active processor in MHz. The speed is approximate.

Processor

The Processor Statistic displays the type of processor currently in use, for example, GenuineIntel.

Disk Time

The Disk Time statistic is the percentage of elapsed time that the selected disk drive/device was busy servicing read or write requests.

Metrics

You should avoid consistently seeing values for this statistic greater then 90%.

Troubleshooting

Add more disk drives and partition the files among all of the drives.

Load Average

The Load Average statistic represents the system load averages over the last 1, 5, and 15 minutes.

Metrics

High load averages usually mean that the system is being used heavily and the response time is correspondingly slow.

Paged Memory Used

The Paged Memory Used statistic is the ratio of Commit Memory Bytes to the Commit Limit. Committed memory is where memory space has been reserved in the paging file if it needs to be written to disk. The commit limit is determined by the size of the paging file. As the paging file increases, so does the commit limit.

NOTE: This statistic is available for the Windows platform.

Metrics

This value displays the current percentage value only and not an average. If the percentage of paged memory used is above 90%, you may be running out of memory.

Troubleshooting

Increase the size of page file.

Number of Processors

This statistic displays the number of processors currently in use.

Swap Memory Used

The Swap Memory Used statistic is the percentage of swap space currently in use.

Metrics

If the percentage of swap memory used is above 90%, you may be running out of memory.

Troubleshooting

Increase the size of your swap files.

Average Disk Queue

The Average Disk Queue statistic is the average number of both read and write requests that were queued for the selected disk during the sample interval.

Metrics

This metric is useful in identifying I/O related bottlenecks. If the disk queue lengths for certain disks are consistently much higher than others, you may need to redistribute the load among available disks. If the disk queues lengths for all disks are consistently large, and you see a high amount of I/O activity, your disks may be inefficient.

Troubleshooting

Some things you can do if you have problems with this statistic include:

- Redistribute the data on the disk with the large average disk queue to other disks.
- Upgrade to faster disk(s).

Page Faults/Sec

The Page Faults/Sec statistic is the overall rate faulted pages are handled by the processor. It is measured in numbers of pages faulted per second. A page fault occurs when a process requires code or data that is not in its working set. This counter includes both hard faults and soft faults.

Metrics

This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

If the number of page faults remains consistently high, you can check with your Windows System Administrator for further investigation. Often, large numbers of page faults are not a problem so long as they are soft faults. However, hard faults, that require disk access, can cause delays.

Processor Queue

The Processor Queue Length statistic is the number of threads in the processor queue.

Metrics

Unlike the disk counters, this counter shows ready threads only, not threads that are running. There is a single queue for processor time even on computers with multiple processors. Therefore, if a computer has multiple processors, you need to divide this value by the number of processors servicing the workload. A sustained processor queue of less than 10 threads per processor is normally acceptable, dependent of the workload.

Troubleshooting

A sustained high value in the Processor Queue could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Network Output Queue/Network Queue

The Network Output Queue Length statistic is the number of threads in the processor queue.

NOTE: The name of this statistic depends on the platform of the operating system.

Metrics

Unlike the disk counters, this counter shows ready threads only, not threads that are running. There is a single queue for processor time even on computers with multiple processors. Therefore, if a computer has multiple processors, you need to divide this value by the number of processors servicing the workload. A sustained processor queue of less than 10 threads per processor is normally acceptable, dependent of the workload.

Troubleshooting

A sustained high value in the Processor Queue Length could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Available Physical Memory

The Available Physical Memory statistic represents the amount of RAM available to all processes.

Metrics

This counter displays the last observed value only and not an average. Use this value with the Total physical memory and paging metrics (Memory details page). If the available physical memory is very small compared to this value, and the paging activity is high, your system may be running low on memory.

Troubleshooting

Some things you can do if you have problems with this statistic include:

- Check the running processes to see if there are any memory leaks.
- Stop any services that are not required.
- Install additional RAM.

Available Paged Memory

The Available Paged Memory statistic shows the amount of virtual memory available for the processes.

NOTE: This statistic is available for the Windows platform.

Metrics

If the available virtual memory is less than 10% of the total virtual memory, your system may run out of memory.

Troubleshooting

Increase the size of page file.

Available Swap Memory

The Available Swap Memory statistic represents the amount of virtual memory available for the processes.

Metrics

If the available Available Swap Memory is less than 10% of the total Swap Memory, your system may run out of memory.

Troubleshooting

Increase the size of swap files.

Total Physical Memory

The Total Physical Memory statistic shows the amount of physical memory installed on your computer.

Metrics

This is an informational metric and displays the total amount installed on the machine. Use this value with the available physical memory and paging metrics (Memory details page). If the available physical memory is very small compared to this value, and the paging activity is high, your system may be running low on memory.

Total Paged Memory/Total Swap Memory

The Total Paged Memory statistic shows the maximum amount of virtual memory available to all processes.

NOTE: The name of this statistic depends on the platform of the operating system.

Metrics

It is recommended that this value is between 1.5 to 3 times the amount of RAM on the system.

Used Disk Space

The Used Disk Space statistic shows the amount of allocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations:

- If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
- You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Total Disk Space

Total Disk Space displays the total allocated and unallocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations, here are two:

1. If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
2. You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Free Disk Space

The Free Disk Space statistic shows the unallocated space, in megabytes on all logical disk drives.

Troubleshooting

There are many things a DBA can do to ensure that a database does not encounter a space problem due to physical space limitations:

- If a database currently resides on a disk that has little free space, you can add more files to the database. Of course, you should add the new files to other physical hard disks that can accommodate a growing database.
- You should examine hard disks with shrinking disk space to see if you can relocate or delete files to allow more free space.

Top Memory Process

Top Memory Process shows the current process that is consuming the most amount of memory. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Metrics

If you are running out of memory on the system, this is a quick way to identify the top memory user. If the displayed process is using a significant portion of the total memory, it could be causing the memory issues.

Processes Overview

The Processes Overview of the OS Summary includes the following sections:

- Top CPU Process
- Top I/O Process
- Top Memory Process

Top CPU Process

Top CPU Process shows the current process that is consuming the most amount of CPU. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Metrics

If the amount of CPU time used by this process is close to 100% and the CPU usage is very high, this process may be the bottleneck on the server.

Troubleshooting

Investigate the process further to see if it is in an inconsistent state. Also, look at minimum requirements for CPU speed for the process. You may need to upgrade your CPU.

Top I/O Process

The Top I/O Process statistic shows the current process that is consuming the most amount of CPU. The information displayed is dependent on the platform of the operating system. Information displayed includes the name of the process, process ID, amount of memory consumed expressed in KB, amount of CPU expressed as a percentage, the amount of Major Page Faults, and the amount of I/O expressed in KB/sec.

Number of Logins

This statistic displays the total number of logins on the server.

Number of Processes

This statistic displays the total number of processes on the server.

CPU Tab

The CPU tab of the OS Detail includes the following sections:

- Context Switches/Sec
- CPU Utilization
- Interrupts/Sec
- Processor Queue Length

CPU Utilization

The CPU Utilization section includes the following information:

- % Privileged Time
- % User Time

% Privileged Time

The % Privileged Time statistic is the percentage of elapsed time that the process threads spent executing code in privileged mode.

NOTE: For Windows systems, when a Windows system service is called, the service will often run in privileged mode to gain access to system-private data. Such data is protected from access by threads executing in user mode. Calls to the system can be explicit or implicit, such as page faults or interrupts. These kernel commands, are considered privileged to keep the low-level commands executing and prevent a system freeze. Unlike some early operating systems, Windows uses process boundaries for subsystem protection in addition to the traditional protection of user and privileged modes. Some work done by Windows on behalf of the application might appear in other subsystem processes in addition to the privileged time in the process.

Metrics

The ideal range should be 0-40% (less than 40% indicates excessive system activity).

Troubleshooting

If your CPU consistently runs at less than 40% you may need to upgrade your system to include a faster processor(s).

% User Time

The % User Time statistic is the percentage of elapsed time the processor spends in the user mode. User mode is a restricted processing mode designed for applications, environment subsystems, and integral subsystems. The alternative, privileged mode, is designed for operating system components and allows direct access to hardware and all memory. The operating system switches application threads to privileged mode to access operating system services. This counter displays the average busy time as a percentage of the sample time.

Metrics

If the Privileged Time is high in conjunction with Physical Disk Reads, consider upgrading the disk I/O subsystem.

Interrupts/Sec

The Interrupts/Sec statistic is the average rate, in incidents per second, at which the processor received and serviced hardware interrupts. It does not include deferred procedure calls (DPCs), which are counted separately. This value is an indirect indicator of the activity of devices that generate interrupts, such as the system clock, the mouse, disk drivers, data communication lines, network interface cards, and other peripheral devices. These devices normally interrupt the processor when they have completed a task or require attention. Normal thread execution is suspended. The system clock typically interrupts the processor every ten milliseconds, creating a background of interrupt activity. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Metrics

The ideal rage should be 0-5000. A number greater then 5000 indicates possible excessive hardware interrupts; justification is dependent on device activity.

Context Switches/Sec

The Context Switches/Sec section shows the combined rate at which all processors on the computer are switched from one thread to another. Context switches occur when a running thread voluntarily relinquishes the processor, is preempted by a higher priority ready thread, or switches between user-mode and privileged (kernel) mode to use an Executive or subsystem service.

Metrics

The ideal range should be between 0-10,000. GA number greater then 10,000 may indicate too many threads contending for resources.

Processor Queue Length

The Processor Queue Length statistic is the number of threads in the processor queue. There is a single queue for processor time even on computers with multiple processors.

NOTE: For Windows systems, unlike the disk counters, this counter shows ready threads only, not threads that are running.

Metrics

A sustained high value in the Processor Queue Length could indicate that a processor bottleneck has developed due to threads of a process requiring more process cycles than are available. If this is the case, you should look at installing a faster (or an additional) processor.

Processes Tab

The Processes tab of the OS Detail page succinctly communicates the general overall performance levels of processes. The columns available in this table depend on the platform of operating system. The table below describes the information available in the table on this tab:

Column	Description
Process	The name of the process.
User	The user of the process.
ID	The ID Process is the unique identifier of this process. ID Process numbers are reused, so they only identify a process for the lifetime of that process.
CPU	The CPU is the percentage of elapsed time that all of process threads used the processor to execution instructions.
User Mode	The User Mode is the percentage of elapsed time that the process threads spent executing code in user mode.
Memory **WINDOWS ONLY**	Memory is the current size, in bytes, of the virtual address space the process is using. Use of virtual address space does not necessarily imply corresponding use of either disk or main memory pages. Virtual space is finite, and the process can limit its ability to load libraries.
Memory (MB)	Memory is the current size, in bytes, of the virtual address space the process is using. Use of virtual address space does not necessarily imply corresponding use of either disk or main memory pages. Virtual space is finite, and the process can limit its ability to load libraries.
Memory	Memory is the percentage of the memory used of the total memory.
Active Memory	Active Memory is the amount of committed virtual memory, in bytes for this process. Active memory is the physical memory which has space reserved on the disk paging file(s). There can be one or more paging files on each physical drive. This counter displays the last observed value only; it is not an average.

Column	Description
I/O Data	The rate at which the process is reading and writing bytes in I/O operations. This counter counts all I/O activity generated by the process to include file, network and device I/Os.
Elapsed Time	The total elapsed time, in seconds, that this process has been running.
Thread Count	The number of threads currently active in this process. An instruction is the basic unit of execution in a processor, and a thread is the object that executes instructions. Every running process has at least one thread.
Handle Count	The total number of handles currently open by this process. This number is equal to the sum of the handles currently open by each thread in this process.
Priority	The current base priority of this process. Threads within a process can raise and lower their own base priority relative to the process' base priority.
Creating Proc ID	The Creating Process ID value is the Process ID of the process that created the process. The creating process may have terminated, so this value may no longer identify a running process.
Page Faults/Sec	Page Faults/Sec is the rate at which page faults by the threads executing in this process are occurring. A page fault occurs when a thread refers to a virtual memory page that is not in its working set in main memory. This may not cause the page to be fetched from disk if it is on the standby list and hence already in main memory, or if it is in use by another process with whom the page is shared.
Page File	Page File is the current number of kilobytes that this process has used in the paging file(s). Paging files are used to store pages of memory used by the process that are not contained in other files. Paging files are shared by all processes, and the lack of space in paging files can prevent other processes from allocating memory.
Private	Private is the current size, in kilobytes, of memory that this process has allocated that cannot be shared with other processes.

I/O Tab

The table below describes the information available in this section:

Column	Description
Disk	The disk number assignment.
Reading (KB/s)	The amount of bytes read from the device.
Writing (KB/s)	The amount of bytes written to the device.
Disk Read Time	Disk Read Time is the percentage of elapsed time that the selected disk drive was busy servicing read requests.
Disk Write Time	Disk Write Time is the percentage of elapsed time that the selected disk drive was busy servicing write requests.
Disk Time	Disk Time is the percentage of elapsed time that the selected disk was busy servicing requests.
Avg. Read Queue	Avg. Disk Read Queue Length is the average number of read requests that were queued for the selected disk during the sample interval.
Avg. Write Queue	Avg. Disk Write Queue Length is the average number of write requests that were queued for the selected disk during the sample interval.
Disk Reads/Sec	Disk Reads/Sec is the rate of read operations on the disk.
Disk Writes/Sec	Disk Writes/Sec is the rate of write operations on the disk.

Memory Tab

The Memory tab of the OS Detail page includes the following sections:

- Cache Efficiency
- Free Physical
- Free Paged
- Paging Activity
- Page Faults
- Total Physical
- Total Paged

Paging Activity

The Paging Activity section includes the following statistics:

- Pages Input/Sec
- Pages Output/Sec

Pages Input/Sec

The Pages Input/Sec statistic is the number of pages read from disk to resolve hard page faults. Hard page faults occur when a process requires code or data that is not in its working set or elsewhere in physical memory, and must be retrieved from disk.

Metrics

This value was designed as a primary indicator of the kinds of faults that cause system-wide delays. It includes pages retrieved to satisfy faults in the file system cache (usually requested by applications) and in non-cached mapped memory files. This counter counts numbers of pages, and can be compared to other counts of pages, such as Memory: Page Faults/sec, without conversion. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

Although it never hurts to have as much physical memory as your system can handle, there are some things you can check within your system to alleviate the memory bottleneck.

- Check to see if you have any drivers or protocols that are running but not being used. They use space in all memory pools even if they are idle.
- Check to see if you have additional space on your disk drive that you could use to expand the size of your page file. Normally, the bigger the initial size of your page file, the better, in performance terms.

Pages Output/Sec

The Pages Output/Sec statistic is the number of pages written to disk to free up space in physical memory. Pages are written back to disk only if they are changed in physical memory. A high rate of pages output might indicate a memory shortage.

Metrics

Windows NT writes more pages back to disk to free up space when low in physical memory. This counter counts numbers of pages, and can be compared to other counts of pages, without conversion. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

Although it never hurts to have as much physical memory as your system can handle, there are some things you can check within your system to alleviate the memory bottleneck.

- Check to see if you have any drivers or protocols that are running but not being used. They use space in all memory pools even if they are idle.
- Check to see if you have additional space on your disk drive that you could use to expand the size of your page file. Normally, the bigger the initial size of your page file, the better, in performance terms.

Free Physical

The Free Physical statistic is the amount of physical memory that is uncommitted.

Metrics

None.

Free Paged

The Free Paged statistic is the amount of uncommitted virtual memory.

Metrics

None.

Total Physical

The Total Physical statistic is the total physical memory available.

Metrics

None.

Total Paged

The Total Paged statistic is the amount of total virtual memory, in bytes. Used Memory is the physical memory that has space reserved on the disk paging file(s). There can be one or more paging files on each physical drive.

Metrics

None.

Page Faults/Sec

The Page Faults/Sec statistic is the overall rate faulted pages are handled by the processor. It is measured in numbers of pages faulted per second. A page fault occurs when a process requires code or data that is not in its working set. This counter includes both hard faults and soft faults.

Metrics

This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval.

Troubleshooting

If the number of page faults remains consistently high, you can check with your Windows System Administrator for further investigation. Often, large numbers of page faults are not a problem so long as they are soft faults. However, hard faults, that require disk access, can cause delays.

Cache Efficiency

The Cache Efficiency section of the Memory tab succinctly communicates the general overall performance levels of the server's memory. The following statistics are available in this section:

- Copy Read Hits%
- Data Map Hits%
- MDL Read Hits%
- Pin Read Hits%

Copy Read Hits %

The Copy Read Hits % statistic is the percentage of cache copy read requests that hit the cache and does not require a disk read to provide access to the page in the cache.

Metrics

When the page is pinned in the memory, the page's physical address in the file system cache will not be altered. A copy read is a file read operation where a page in the cache is copied to the application's buffer. Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate

Data Map Hits %

The Data Map Hits % statistic is the percentage of data maps in the file system cache that could be resolved without having to retrieve a page from the disk.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

MDL Read Hits %

The MDL Read Hits % statistic is the percentage of Memory Descriptor List Read requests to the file system cache that hit the cache and does not require disk access to provide memory access to the pages in the cache.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

Pin Read Hits %

The Pin Read Hits % statistic is the percentage of pin read requests that hit the file system cache and does not require a disk read in order to provide access to the page in the file system cache.

Metrics

Because this value reflects hits, it ideally should be close to 100%. An amount below 100% indicates misses.

Troubleshooting

Adding physical memory to a server results in a larger file system cache, which is generally more efficient. Defragmenting disks also helps, putting related pages in the cache together and thereby improving the cache hit rate.

Space Tab

The Space tab of the OS Detail page includes the following sections:

- Disk Space Free
- Disk Space Detail

Disk Space Free

The Disk Space Free metric displays the amount of free disk space in megabytes.

Metric

None.

Disk Space Detail

The Disk Space Detail section of the Space tab succinctly communicates the general overall performance levels of the server’s disks and space allotment. The table below describes the statistics in this section:

Statistic	Description
Partition	The drive letter of the disk.
Total Space	Total size of the disk/device's capacity expressed in MBs.
Used Space	Amount of MBs currently allocated on the particular disk/device.
Free Space	Amount of MBs currently unallocated and free on the particular disk/device.

Network Tab

The Network tab of the OS Detail page succinctly communicates the general overall performance levels of the server's networking. The statistics available in this section depend on the platform of operating system. The table below describes the information available in this section:

Column	Description
Network Interface	The name of network interface.
INET Address/Address	The IP address assigned to the network interface.
Pkts Sent/Sec	The number of packets sent per second.
Pkts Received/Sec	The number of packets received per second.
Sent (KB/Sec)	The number of bytes sent per second.
Received (KB/Sec)	The number of bytes received per second.
Out Pkts Discarded	The number of outbound packets discarded.
In Pkts Discarded	The number of inbound packets discarded.
Out Pkt Errors	The number of outbound packet errors.
In Pkt Errors	The number of inbound packet errors.
Queue Length	The queue length.
Collisions	The number of collisions.
Packets Discarded	The number of packets discarded.

Session Detail View

The Session Detail view is available for the following database platforms:

- Oracle Session Detail View
- SQL Server Session Detail View
- Sybase Session Detail View
- DB2 Session Detail View

Oracle Session Detail View

The following tabbed pages are available on the Session Detail view for Oracle:

- Session Contention
- Session I/O
- Session Memory
- Session Network
- Session Objects
- Session SQL
- Session Statistics

Session Memory Tab for Oracle

- Metrics

The Session Memory tab of the Session Detail view presents the statistics surrounding a session's memory usage. The table below describes the information available on the Session Memory tab of the Session Detail view for Oracle:

Column	Description
Statistic	The name of the memory related statistic.
Value	The cumulative value for the memory statistic.
Cache Hit Ratio	The percentage of data obtained from memory access vs. physical I/O.

Metrics

Sessions with abnormally high memory usage can affect overall performance at the server level, as this memory (PGA and UGA) is allocated outside of the Oracle SGA (unless the multi-threaded server option is being used). If cache hit ratios at the session level are lower than 85 percent, data access can be inefficient.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session I/O Tab for Oracle

- Metrics

The Session I/O tab of the Session Detail view presents the statistics surrounding a session's I/O activity. The table below describes the information available on the Session I/O tab of the Session Detail view for Oracle:

Column	Description
Statistic	The name of the I/O related statistic.
Value	The cumulative value for the I/O statistic.

Metrics

Seeing high values for physical reads and writes can indicate an inefficient session. Large numbers of physical reads can imply a session with too many large table scans or inefficient SQL operations. Large numbers of physical writes can be okay for sessions inputting large volumes of data into the database. Or, they could also indicate a session involved in heavy disk sort activity.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session Contention Tab for Oracle

- Metrics

The Session Contention tab of the Session Detail view presents information relating to resources on which the current session is waiting. The first grid displays user waits. The second grid displays user locks. The table below describes the information available in the User Waits grid on the Session Contention tab of the Session Detail view for Oracle:

Column	Description
Wait Cause	The wait event being experienced by the session.
Program	The program the session is executing against Oracle.
Seconds	The number of seconds the session has spent in the wait.

Column	Description
State	The status of the wait event (WAITING, etc.)

The table below describes the information available in the User Locks grid on the Session I/O tab of the Session Detail view for Oracle:

Column	Description
User	The user account being used by the session.
Terminal	The client machine name used by the session.
SID	The unique Oracle identifier for the session.
Serial #	The serial number for the session.
Table	The object locked by the session.
Lock Mode	The lock mode used by the session.
Title	The lock request issued by the session.

Metrics

You can ignore some waits, like the SQL*Net more data from client and SQL*Net message from client. Others, like enqueue waits, can be indicative of a lock contention problem. Waits, like db file scattered reads, can indicate sessions involved in long table scan operations.

Locks that are held for unusually long periods require further investigation. The application logic can be inefficient or the program is not issuing COMMITs frequently enough. The culprit of blocking-lock scenarios is usually the application design, or the SQL used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the correct SQL to get the job done is an art. Most DBAs who have had to face Oracle Forms applications have suffered through the dreaded SELECT…FOR UPDATE statements. These place unnecessary restrictive locks on nearly every read operation. The key to avoiding lock contention is to process user transactions as quickly and efficiently as possible.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session Objects Tab for Oracle

- Metrics

The Session Objects tab of the Session Detail view presents information regarding the objects in use by a session. The table below describes the information available in the Objects Accessed grid on the Session Objects tab of the Session Detail view for Oracle:

Column	Description
Owner	The owner of the object.
Type	The type of object (TABLE, VIEW, etc.).
Object	The name of the object.

The Rollback Segments Accessed grid displays the names of rollback segments currently being used by the session.

Metrics

Once you have an idea of which objects your users access most often, you can refine some processes to facilitate access to them. You can use the Oracle 8 concept of the KEEP buffer cache to force Oracle to hold often-referenced data for data objects. The KEEP buffer cache is ideal for holding small look-up tables. If you have an earlier version of Oracle or do not want to split up your current buffer cache, you can use the CACHE table attribute to encourage Oracle to keep data blocks of CACHE'd tables at the most recently used end of the LRU buffer cache chain.

If you consistently see a session with active rollbacks, it can indicate locks are being held for long durations. It can also indicate that a session is using code without frequent enough COMMIT points.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session Network Tab for Oracle

- Metrics

The Session Network tab of the Session Detail view presents information about requests being sent to and from the database with respect to the current session. The table below describes the information available on the Session Network tab of the Session Detail view for Oracle:

Column	Description
Statistic	The name of the SQL*Net related statistic.
Value	The cumulative value of the SQL*Net related statistic.

Metrics

None.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session SQL Tab for Oracle

- Metrics

The Session SQL tab of the Session Detail view presents information relating to any current SQL issued by the session as well as any SQL previously issued by the session.

Metrics

To determine access paths, you should export and run through an EXPLAIN PLAN session any SQL that you suspect of inefficient access.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

Session Statistics Tab for Oracle

- Metrics

The Session Statistics tab of the Session Detail view presents information relating to all recorded performance and miscellaneous statistics for the current session. The table below describes the information available on the Session Statistics tab of the Session Detail view for Oracle:

Column	Description
Statistic	The name of the session related statistic.

Column	Description
Value	The cumulative value of the session related statistic.

Metrics

None.

> **TIP:** To configure the grid to show/hide row numbers, use the Options Editor.

SQL Server Session Detail View

The Session Detail view down provides a granular look at the details when a process is acting in a way that merits further investigation. It also presents data about a particular process in a way that makes it easier to understand and view apart from all other processes that SQL Server is running.

The following tabbed pages are available on the Session Detail view for SQL Server:

- All Locks
- Blocked By
- Blocking
- Overview
- SQL

Overview Tab for SQL Server

- Metrics

The Overview tab of the Session Detail view displays information to analyze the details of a particular process. The tables below describe the statistics for each category on the Overview tab of the Session Detail view for SQL Server. The available categories are:

- Contention
- General
- I/O
- Memory
- Network
- Users

General Statistics

The table below describes the statistics in the General category on the Overview tab of the Session Detail view:

Statistic	Description
SPID	The SQL Server process ID. Unique value across all processes.
Login Name	The SQL Server login name of the process.
NT User	If using Windows Authentication, the name of the Windows NT user for this process. If using a trusted connection, the Windows NT user name.

Statistic	Description
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Database	The database in which the process is running.
Program	The executable the process is using against the server.
Host	The machine name that originated the process.

Memory Statistics

The table below describes the statistics in the Memory category on the Overview tab of the Session Detail view:

Statistic	Description
Memory Usage	The number of pages in the procedure cache (SQL Cache) that are currently allocated to this process. A negative number indicates that pages are being released (freed) from this process.
Buffer Cache Hit Ratio	The percentage of data page requests by this process that is available in memory as opposed to performing a physical I/O to disk.

Contention Statistics

The table below describes the statistics in the Contention category on the Overview tab of the Session Detail view:

Statistic	Description
Blocked By	If the process is being blocked, the SPID of the blocking process. A value of zero means that the process is not blocked.
Wait Time	The number of milliseconds that the process has been waiting to be serviced. A value of zero indicates that the process is not waiting.
Last Wait Type	The last or current SQL Server wait type.
Wait Resource	The SQL Server's textual representation of a lock resource.

I/O Statistics

The table below describes the statistics in the I/O category on the Overview tab of the Session Detail view:

Statistic	Description
Physical I/O	The number of physical and logical reads performed by this session.
Physical Reads	The number of physical reads from disk performed by this session.
Logical Reads	The number of reads from memory performed by this process.
Logical Writes	The number of writes to memory performed by this process.

Users Statistics

The table below describes the statistics in the Users category on the Overview tab of the Session Detail view:

Statistic	Description
CPU Usage	The cumulative CPU time for the process. The CPU usage is updated regardless of the value of the SET STATISTICS TIME ON option.
Open Trans	The current number of open transactions owned by the process.
Login Time	For client process, the last time a client logged into the SQL Server instance. For system processes, the time that SQL Server was started.

Statistic	Description
Last Batch	For client process, the last time a remote stored procedure call or an EXECUTE statement was executed. For system processes, the time that SQL Server was started.

Network Statistics

The table below describes the statistics in the Network category on the Overview tab of the Session Detail view:

Statistic	Description
Net Address	The unique identifier of the network card in the client machine that owns the process.
Net Library	When a process is initiated from a client, the controlling mechanism is the network connection. Each network connection has a library associated with it. This value is the name library associated with the network connection responsible for this process.

Metrics

High memory usage and a low cache hit ratio for a given process over a sustained period of time could indicate that the process is using poorly written code. Check the SQL tab to investigate further.

Also, watch for an unusually high percentage of CPU use over a long period of time. This could indicate a rogue process that must be terminated by the DBA using a KILL command for the session.

SQL Tab for SQL Server

- Metrics

The SQL tab of the Session Detail view presents information relating to any current SQL issued by the session as well as any SQL previously issued by the session.

Metrics

To determine access paths, you should export and run through a QUERY PLAN session, any SQL suspect of inefficient access.

Blocked By Tab for SQL Server

- Metrics
- Troubleshooting

Without a doubt, blocking lock situations can give the appearance of a “frozen” database almost more than anything else. A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems. Although SQL Server supports row-level locking, blocking lock situations do crop up - sometimes frequently.

Metrics

The Blocked By tab contains information relating to processes that are currently blocking the process featured in the Session Detail view. The table below describes the information available on the Blocked By tab of the Session Detail view for SQL Server:

Column	Description
SPID	The SQL Server process ID. It is a unique value across all processes.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Login	The SQL Server login name of the process.
NT User	If using Windows Authentication, the name of the Windows NT user for this process. If using a trusted connection, the Windows NT user name.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Memory	The number of pages in the procedure cache (SQL Cache) that are currently allocated to this process. A negative number indicates that pages are being released (freed) from this process.
CPU	The cumulative CPU time for the process. The CPU usage is updated regardless of the value of the SET STATISTICS TIME ON option.
Physical I/O	The current cumulative number of physical disk reads and writes issued by the process.
Blocked	If the process is being blocked, the SPID of the blocking process.
Database	The database in which the process is running.
Command	The current command being executed by the process.
Last Batch	For a client process, the last time a remote stored procedure call or an EXECUTE statement was executed. For system processes, the time that SQL Server was started.
Login Time	For a client process, the last time a client logged into the SQL Server instance. For system processes, the time that SQL Server was started.
Wait Time	The time that the process has been waiting to be serviced, in milliseconds. A value of zero indicates that the process is not waiting.
Wait Type	The last or current SQL Server wait type.
Open Xacts	The current number of open transactions owned by the process.
NT Domain	If using Windows Authentication or a trusted connection, the name of the Windows domain of the user who owns the process.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects the user was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process waits for a lock before timing out.

Blocking Tab for SQL Server

Without a doubt, blocking lock situations can give the appearance of a “frozen” database almost more than anything else. A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems. Although SQL Server supports row-level locking, blocking lock situations do crop up - sometimes frequently.

Metrics

The Blocking tab contains information on blocks issued by the featured process that are blocking other processes. The table below describes the information available on the Blocking tab of the Session Detail view for SQL Server:

Column	Description
SPID	The SQL Server process ID. It is a unique value across all processes.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Login	The SQL Server login name of the process.
NT User	If using Windows Authentication, the name of the Windows NT user for this process. If using a trusted connection, the Windows NT user name.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Memory	The number of pages in the procedure cache (SQL Cache) that are currently allocated to this process. A negative number indicates that pages are being released (freed) from this process.
CPU	The cumulative CPU time for the process. The CPU usage is updated regardless of the value of the SET STATISTICS TIME ON option.
Physical I/O	The current cumulative number of physical disk reads and writes issued by the process.
Database	The database in which the process is running.
Command	The current command being executed by the process.
Last Batch	For client process, this value represents the last time a remote stored procedure call or an EXECUTE statement was executed. For system processes, it represents the time at which SQL Server was started.
Login Time	For client process, the last time a client logged into the SQL Server instance. For system processes, the time that SQL Server was started.
Wait Time	The time that the process has been waiting to be serviced, in milliseconds. A value of zero indicates that the process is not waiting.
Wait Type	The last or current SQL Server wait type.
Open Xacts	The current number of open transactions owned by the process.
NT Domain	If using Windows Authentication or a trusted connection, the name of the Windows Domain of the user who owns the process.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects the user was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process waits for a lock before timing out.

All Locks Tab for SQL Server

- Metrics
- Troubleshooting

Without a doubt, blocking lock situations can give the appearance of a “frozen” database almost more than anything else. A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches even on large systems. Although SQL Server supports row-level locking, blocking lock situations do crop up - sometimes frequently.

Metrics

The All Locks tab contains information on all locks associated with the featured SPID, including locks that are held by the process and locks that are blocking the process. The table below describes the information available on the All Locks tab of the Session Detail view for SQL Server:

Column	Description
Database	The database where the locks are occurring.
Table Name	The name of the table involved in a lock. NULL for non-table locks or table locks that take place in the tempdb database.
Ndx ID	The index ID involved in the lock.
Lock Type	The type of lock (database, table, row ID, etc.).
Lock Mode	The mode of the lock (shared, exclusive, etc.).
Lock Status	The status of the lock (waiting or granted).
Owner Type	Whether the lock came from a regular session or a transaction.
Program	The executable the process is using against the server.
BLK SPID	If nonzero, the process ID of the process blocking the requested lock. A value of zero indicates that the process is not blocked.
Wait Time	The time the process has waited for the lock, in milliseconds.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Command	The command the process is currently issuing.
NT Domain	The name of the Windows 2000/NT domain.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects the use was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

By default, all processes wait indefinitely for locks in SQL Server. You can change this behavior by using the SET LOCK_TIMEOUT command, which limits the number of seconds that a process waits for a lock before timing out.

Sybase Session Detail View

The Session Detail view down provides a granular look at the details when a process is acting in a way that merits further investigation. It also presents data about a particular process in a way that makes it easier to understand and view apart from all other processes that Sybase is running.

The following tabbed pages are available on the Session Detail view for Sybase:

- Overview
- SQL
- Blocked By
- Blocking
- All Locks

Overview Tab for Sybase

The Overview tab of the Session Detail view displays information to analyze the details of a particular process. The tables below describe the statistics for each category on the Overview tab of the Session Detail view for Sybase. The available categories are:

- Contention
- Execution
- General
- I/O
- Memory
- Users

General Statistics

The table below describes the statistics in the General category on the Overview tab of the Session Detail view:

Statistic	Description
SPID	The Sybase process ID. Unique value across all processes.
Login Name	The Sybase login name of the process.
Family ID	The ID of the coordinating process and all of its worker processes.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Database	The database in which the process is running.
Program	The executable the process is using against the server.

Statistic	Description
Host	The machine name that originated the process.

Memory Statistics

The Memory Usage statistic in the Memory category on the Overview tab of the Session Detail view is the number of pages in the procedure cache (SQL Cache) that are currently allocated to this process. A negative number indicates that pages are being released (freed) from this process.

Contention Statistics

The table below describes the statistics in the Contention category on the Overview tab of the Session Detail view:

Statistic	Description
Blocked By	If the process is being blocked, the SPID of the blocking process. A value of zero indicates that the process is not blocked.
Time Blocked	The SQL Server's wait time.

I/O Statistics

The Physical I/O statistic in the I/O category on the Overview tab of the Session Detail view is the number of physical and logical reads performed by this session.

Users Statistics

The table below describes the statistics in the Users category on the Overview tab of the Session Detail view:

Statistic	Description
CPU Usage	The cumulative CPU time for the process. The CPU usage is updated regardless of the value of the SET STATISTICS TIME ON option.
Active Trans	The current number of open transactions owned by the process.

Execution Statistics

The table below describes the statistics in the Execution category on the Overview tab of the Session Detail view:

Statistic	Description
Exec Class	The execution class for the current process.
Priority	The priority of the current process.
Affinity	The affinity level for the current process.

SQL Tab for Sybase

The SQL tab of the Session Detail view presents information relating to any current SQL issued by the session as well as any SQL previously issued by the session.

Metrics

To determine access paths, you should export and run through a QUERY PLAN session, any SQL suspect of inefficient access.

Blocked By Tab for Sybase

- Metrics
- Troubleshooting

Without a doubt, blocking lock situations can give the appearance of a “frozen” database almost more than anything else. A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches even on large systems.

Metrics

The Blocked By tab contains information relating to processes that are currently blocking the process featured in the Session Detail view. The table below describes the information available on the Blocked By tab of the Session Detail view for Sybase:

Column	Description
SPID	The Sybase process ID. This is a unique value across all processes.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Login	The Sybase login name of the process.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Memory	The number of pages in the procedure cache (SQL Cache) that are currently allocated to this process. A negative number indicates that pages are being released (freed) from this process.
CPU	The cumulative CPU time for the process. The CPU usage is updated regardless of the value of the SET STATISTICS TIME ON option.
Physical I/O	The current cumulative number of physical disk reads and writes issued by the process.
Database	The database in which the process is running.
Command	The current command being executed by the process.
Time Blocked	The time that the process has been blocked, in seconds.
Transaction	The name of any transaction.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects the user was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

Blocking Tab for Sybase

- Metrics
- Troubleshooting

Without a doubt, blocking lock situations can give the appearance of a “frozen” database almost more than anything else. A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems.

Metrics

The Blocking tab contains information on blocks issued by the featured process that are blocking other processes. The table below describes the information available on the Blocking tab of the Session Detail view for Sybase:

Column	Description
SPID	The Sybase process ID. This is a unique value across all processes.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Login	The Sybase login name of the process.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Memory	The number of pages in the procedure cache (SQL Cache) that are currently allocated to this process. A negative number indicates that pages are being released (freed) from this process.
CPU	The cumulative CPU time for the process. The CPU usage is updated regardless of the value of the SET STATISTICS TIME ON option.
Physical I/O	The current cumulative number of physical disk reads and writes issued by the process.
Database	The database in which the process is running.
Command	The current command being executed by the process.
Time Blocked	The time that the process has been blocked, in seconds.
Transaction	The name of any transaction.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied - the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects they were accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

All Locks Tab for Sybase

Without a doubt, blocking lock situations can give the appearance of a “frozen” database almost more than anything else. A single blocking user has the potential to stop work for nearly all other processes on a small system, and can cause major headaches on large systems.

Metrics The All Locks tab contains information on all locks associated with the featured SPID, including locks that are held by the process and locks that are blocking the process. The table below describes the information available on the All Locks tab of the Session Detail view for Sybase:

Column	Description
Database	The database where the locks are occurring.
Lock Type	The type of lock (database, table, row ID, etc.)
Object Name	The name of the object being blocked.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Lock Page	The page number where the lock is currently applied.
Lock Class	The name of the cursor the lock is associated with (if any).
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Command	The command the process is currently issuing.
Transaction	The name of any transaction.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Troubleshooting

Once discovered, a blocking lock situation can normally be quickly remedied—the DBA issues a KILL against the offending process, which eliminates the user's stranglehold on the objects the user was accessing. Other user processes then nearly almost always complete in an instant. Discovering the blocked lock situation is made easier by using tools like Embarcadero Performance Center, but preventing the blocking lock situation in the first place is tricky.

The culprit of blocking lock scenarios is usually the application design, or the SQL being used within the application itself. Properly coding an application to reference database objects in an efficient order, and then using the right SQL to get the job done, is an art. The key to avoiding lock contention is to process user transactions in the quickest and most efficient manner possible - something not always easy to do.

DB2 Session Detail View

The following tabbed pages are available on the Session Detail view for DB2:

- Application Tab for DB2
- Unit of Work Tab for DB2
- Locking Tab for DB2
- Memory Tab for DB2
- I/O Tab for DB2
- SQL Statistics Tab for DB2

The following information about the selected application is available on each page:

Field	Description
Agent ID	This is a system-wide unique indentifier for the application.
Auth ID	This is the authorization ID of the user who invoked the application being monitored.

Field	Description
OS User ID	This is the authorization ID used to access the operating system.
Application	This is the name of the application executable.

Application Tab for DB2

The Application tab contains application and client detail information from the application snapshot record for the selected application.

Application Details

Statistic	Description
Application Agent ID	This is a system-wide unique indentifier for the application.
Application Agent PID	The process ID (UNIX) or thread ID (Windows) of a DB2 agent.
Application Agent System CPU (sec)	This is the total system CPU time (in seconds) used by the database manager agent process.
Application Agent User CPU (sec)	This is the total CPU time (in seconds) used by database manager agent process.
Application Agents	This is the number of agents currently executing statements or subsections.
Application Agents Stolen	This is the number of times agents are stolen from the application. Agents are stolen when an idle agent is reassigned from one application to another.
Application Assoc Agents HWM	This is the maximum number of subagents associated with the application.
Application Assoc Sub-Agents	This is the number of subagents associated with the application.
Application Authorization ID	This is the authorization ID of the user who invoked the application being monitored.
Application Codepage	This is the code page identifier.
Application Connect Complete Time	This is the date and time a connection request was granted.
Application Connect Start Time	This is the date and time an application started a connection request.
Application Coord Agent PID	This is the process ID (UNIX) or thread ID (Windows) of the coordinator agent for the selected application.
Application Coord Node	In a multi-node system, this is the node number of the node where the selected application connected or attached to the instance.
Application Coord Token	This is the DRDA AS correlation token.
Application ID	This is generated when the application connects to the database at the database manager or when DDCS receives a request to connect to a DRDA database.
Application Idle Time	This is the number of seconds since the selected application last issued any requests to the server. This includes applications that have not terminated a transaction, for example not issued a commit or rollback.
Application Name	Name of the application executable.
OS User ID	The authorization ID used to access the operating system.
Application Priority	This is the priority of the agents working for this application.
Application Priority Type	This is the operating system priority type for the agent working on behalf of the application.
Application Sequence Number	This identifier is incremented when a unit of work ends (when a COMMIT or ROLLBACK terminates a unit of work). Together, the application ID and application sequence number uniquely identify a transaction.

Statistic	Description
Application Session Auth ID	This is the current authorization ID for the session being used by this application.
Application Status	This is the current status of the application.
Application Status Change Time	This is the date and time the application entered its current status.
Application Territory Code	This is the territory code (formerly country code) of the database for which the monitor data is being collected.

Application Authorities

Column	Description
Authority	This is the highest authority level granted to the application.
Explicit	Authorizations granted explicitly to a user.
Indirect	Indirect authorizations inherited from group or public.

Client Details

Statistic	Description
Client Database Alias	This is the alias of the database provided by the application to connect to the database.
Client Inbound Comm Address	This is the communication address of the client.
Client Node Name	This is the node name (nname) in the database manager configuration file at the client database partition.
Client PID	This is the process ID of the client application that made the connection to the database.
Client Platform	This is the operating system on which the client application is running.
Client Product and Version	This is the product and version that is running on the client.
Client Protocol	This is the communication protocol that the client application is using to communicate with the server.
TP Client Accounting String	This is the data passed to the target database for logging and diagnostic purposes (if the sqleseti API was issued in this connection).
TP Client Application Name	This name identifies the server transaction program performing the transaction (if the sqleseti API was issued in this connection).
TP Client User ID	This is the client user ID generated by a transaction manager and provided to the server (if the sqleseti API is used).
TP Client Workstation Name	This name identifies the client's system or workstation, for example CICS EITERMID, if the sqleseti API was issued in this connection.

Unit of Work Tab for DB2

The Unit of Work tab contains the SQL statement for the selected application, as well as statistics related to that statement.

SQL Statement

The SQL Statement field displays the SQL statement for the application. You can click **Explain SQL** to view an explain plan for the statement.

Statement Detail

Statistic	Description
SQL Statement	This is the text of the dynamic SQL statement.
Node	This is the node where the statement was executed.
Type	This is the type of statement processed.
Creator	This is the authorization ID of the user that pre-compiled the application.
Operation	This is the statement operation currently being processed or that was most recently processed (if none are currently running).
Agents Created	This is the maximum number of agents that were used when executing the statement.
Agents Working	This is the number of concurrent agents currently executing a statement or subsection.
Cursor Name	This is the name of the cursor corresponding to this SQL statement.
Blocking Cursor	This indicates if the statement being executed is using a blocking cursor.
Package Name	This is the name of the package that contains the SQL statement that is currently executing.
Package Version	This identifies the version identifier of the package that contains the currently executing SQL statement.
Section Number	This is the internal section number in the package for the SQL statement currently processing or most recently processed.
Parallelism Degree	This is the degree of parallelism requested when the query was bound.
Query Cost Estimate	This is the estimated cost for a query (in timerons) as determined by the SQL compiler.
Query Card Estimate	This is an estimate of the number of rows that will be returned by a query.

Statement Statistics

Statistic	Description
Statement Start Time	This is the date and time when the statement operation (stmt_operation) monitor element started executing.
Statement Stop Time	This is the date and time when the statement operation (stmt_operation) monitor element stopped executing.
Statement Sort Time (sec)	This is the total elapsed time (in seconds) for all sorts that have been executed.
Statement User CPU (sec)	This is the total user CPU time (in seconds) used by the currently executing statement.
Statement System CPU (sec)	This is the total system CPU time (in seconds) used by the currently executing statement.
Statement Elapsed Time (sec)	This is the elapsed execution time (in seconds) of the most recently completed statement.
UOW Start Time	This is the date and time that the unit of work first required database resources.
UOW Stop Time	This is the date and time that the most recent unit of work completed. This occurs when database changes are committed or rolled back.
UOW Prev Stop Time	This is the time the previous unit of work completed.
UOW Elapsed Time (sec)	This is the elapsed execution time (in seconds) of the most recently completed unit of work.
UOW Lock Wait Time (sec)	This is the total amount of elapsed time (in seconds) this unit of work has spent waiting for locks.
UOW Log Space Used (KB)	This is the amount of log space (in kilobytes) used in the current unit of work of the monitored application.
UOW Completion Status	This is the status of the unit of work and how it stopped.

Statement Activity

Statistic	Description
Sorts	This is the total number of times a set of data was sorted in order to process the statement operation (stmt_operation).
Sort Overflows	This is the total number of sorts that ran out of sort heap and may have used disk space for temporary storage.
Fetches	This is the number of successful fetches that were performed on a specific cursor.
Rows Read	This is the number of rows read from the table.
Rows Written	This is the number of rows changed (inserted, deleted, or updated) in the table.
Internal Rows Deleted	This is the number of rows deleted from the database as a result of internal activity.
Internal Rows Updated	This is the number of rows updated from the database as a result of internal activity.
Internal Rows Inserted	This is the number of rows inserted into the database as a result of internal activity that was caused by triggers.
Temporary Data Logical Reads	This indicates the number of data pages that have been requested from the buffer pool (logical) for temporary table spaces. **NOTE:** The functionality to record buffer pool information at the statement level is supported for API and CLP snapshot requests.
Temporary Data Physical Reads	Indicates the number of data pages that have been read in from the table space containers (physical) for temporary table spaces. **NOTE:** The functionality to record buffer pool information at the statement level is supported for API and CLP snapshot requests.
Temporary Index Logical Reads	Indicates the number of index pages that have been requested from the buffer pool (logical) for temporary table spaces. **NOTE:** The functionality to record buffer pool information at the statement level is supported for API and CLP snapshot requests.
Temporary Index Physical Reads	Indicates the number of index pages that have been read in from the table space containers (physical) for temporary table spaces. **NOTE:** The functionality to record buffer pool information at the statement level is supported for API and CLP snapshot requests.

Locking Tab for DB2

The Locking tab is a single application view of the Lock View. Information that is found in the Locks Held and Locks Waiting tabs of the Lock View will be available here for the selected application.

Locks Held

Column	Description
Lock Mode	This is the type of lock being held.
Object Type	This is the type of object against which the application holds a lock (for object-lock-level information), or the type of object for which the application is waiting to obtain a lock (for application-level and deadlock-level information).
Table Schema	This is the schema of the table that the lock is on.
Table Name	This is the name of the table that the lock is on. This element is only set if Object Type indicates Table.
Tablespace	This is the name of the table space against which the lock is held.
Lock Status	This is the lock's status (waiting or granted).

Column	Description
Escalation	This indicates whether a lock request was made as part of a lock escalation.

Locks Waiting

Column	Description
Agent ID	This is the application handle of the agent holding a lock for which this application is waiting.
Application ID	This is the application ID of the application that is holding a lock on the object that this application is waiting to obtain.
Lock Mode	This is the type of lock being held.
Mode Requested	This is the lock mode being requested by the application.
Object Type	This is the type of object against which the application holds a lock.
Table Schema	This is the schema of the table that the lock is on.
Tablespace	This is the name of the table that the lock is on. This element is only set if Object Type indicates Table.
Wait Start Time	This is the date and time that this application started waiting to obtain a lock on the object that is currently locked by another application.
Escalation	This indicates whether a lock request was made as part of a lock escalation.

Memory Tab for DB2

The Memory tab displays all the memory pools for the application and key cache statistics specific to the application.

Memory Pools

Column	Description
Node	This is the number assigned to the node in the db2nodes.cfg file.
Memory Pool	This is the type of memory pool.
Utilization	This is the percentage of memory pool used.
High Watermark (MB)	This is the largest size of a memory pool (in megabytes) since its creation.
Current Size (MB)	This is the current size of a memory pool (in megabytes).
Max Size (MB)	This is the internally configured size of a memory pool (in megabytes) in DB2.

Memory Statistics

Statistic	Description
Application Section Inserts	This is the number of inserts of SQL sections by an application from its SQL work area.
Application Section Lookups	This is the number of lookups of SQL sections by an application from its SQL work area.
Catalog Cache Inserts	This is the number of times that the system tried to insert table descriptor or authorization information into the catalog cache.
Catalog Cache Lookups	This is the number of times that the catalog cache was referenced to obtain table descriptor information or authorization information.
Catalog Cache Overflows	This is the number of times that the catalog cache overflowed the bounds of its allocated memory.

Statistic	Description
Package Cache Inserts	This is the total number of times that a requested section was not available for use and had to be loaded into the package cache. This includes any implicit prepares performed by the system.
Package Cache Lookups	This is the number of times an application looked for a section or package in the package cache.
Private Workspace Inserts	This is the number of inserts of SQL sections by an application into the private workspace.
Private Workspace Lookups	This is the number of lookups of SQL sections by an application in its agents' private workspace.
Private Workspace Overflows	This is the number of times that the private workspaces overflowed the bounds of its allocated memory.
Private Workspace HWM (KB)	This is the largest size (in kilobytes) reached by the private workspace.
Shared Workspace Inserts	This is the number of inserts of SQL sections by applications into shared workspaces.
Shared Workspace Lookups	This is the number of lookups of SQL sections by applications in shared workspaces.
Shared Workspace Overflows	This is the number of times that shared workspaces overflowed the bounds of their allocated memory.
Shared Workspace HWM (KB)	This is the largest size (in kilobytes) reached by shared workspaces.

Cache Hit Ratios

Ratio	Description
Catalog Cache	The catalog cache hit ratio indicates how well the catalog cache is helping to avoid actual accesses to the catalog on disk. A high ratio indicates successful avoidance of actual disk I/O accesses.
Package Cache	The package cache hit ratio indicates how well the package cache is helping to avoid reloading packages and sections for static SQL from the system catalogs as well as helping to avoid recompiling dynamic SQL statements. A high ratio indicates successful avoidance of these activities.
Shared Workspace	The shared workspace hit ratio indicates how well the shared SQL workspace is helping to avoid having to initialize sections for SQL statements that are about to be executed. A high ratio indicate successful avoidance of this action.
Private Workspace	The private workspace hit ratio indicates how well the private SQL workspace is helping to avoid having to initialize sections for SQL statements that are about to be executed. A high ratio indicate successful avoidance of this action.
Application Workspace	The application workspace hit ratio indicates how well the application SQL workspace is helping to avoid having to initialize sections for SQL statements that are about to be executed. A high ratio indicate successful avoidance of this action.

I/O Tab for DB2

The I/O tab displays detailed I/O information from the application snapshot record for the selected application.

I/O Statistics

Statistic	Description
Buffered Data Logical Reads	This indicates the number of data pages that have been requested from the buffer pool (logical) for regular and large table spaces.

Statistic	Description
Buffered Data Physical Reads	This indicates the number of data pages that have been read from the table space containers (physical) for regular and large table spaces.
Buffered Data Writes	This indicates the number of times a buffer pool data page was physically written to disk.
Buffered Index Logical Reads	This indicates the number of index pages that have been requested from the buffer pool (logical) for regular and large table spaces.
Buffered Index Physical Reads	This indicates the number of index pages that have been read from the table space containers (physical) for regular and large table spaces.
Buffered Index Writes	This indicates the number of times a buffer pool index page was physically written to disk.
Buffered Read Time (sec)	This indicates the total amount of time (in seconds) spent reading data and index pages from the table space containers (physical) for all types of table spaces.
Buffered Write Time (sec)	This indicates the total amount of time (in seconds) spent physically writing data or index pages from the buffer pool to disk.
Direct Reads	This is the number of read operations that do not use the buffer pool.
Direct Read Requests	This is the number of requests to perform a direct read of one or more sectors of data.
Direct Writes	This is the number of write operations that do not use the buffer pool.
Direct Write Requests	This is the number of requests to perform a direct write of one or more sectors of data.
Direct Read Time (sec)	This is the elapsed time (in seconds) required to perform the direct reads.
Direct Write Time (sec)	This is the elapsed time (in seconds) required to perform the direct writes.
Extended Storage – Data Pages From	This is the number of buffer pool data pages copied from extended storage.
Extended Storage – Data Pages To	This is the number of buffer pool data pages copied to extended storage.
Extended Storage – Index Pages From	This is the number of buffer pool index pages copied from extended storage.
Extended Storage – Index Pages To	This is the number of buffer pool index pages copied to extended storage.
Prefetch Pages Unread	This indicates the number of pages that the prefetcher read in that were never used.
Prefetch Wait Time (sec)	This is the time an application spent waiting for an I/O server (prefetcher) to finish loading pages into the buffer pool.
Temporary Data Logical Reads	This indicates the number of data pages which have been requested from the buffer pool (logical) for temporary table spaces.
Temporary Data Physical Reads	This indicates the number of data pages read in from the table space containers (physical) for temporary table spaces.
Temporary Index Logical Reads	This indicates the number of index pages which have been requested from the buffer pool (logical) for temporary table spaces.
Temporary Index Physical Reads	This indicates the number of index pages read in from the table space containers (physical) for temporary table spaces.

I/O Distribution Ratios

Ratio	Description
Direct Read Ratio	Direct Reads are read operations that do not use the buffer pool. Direct Read Ratio is the percentage of all reads that were direct reads.

Ratio	Description
Logical Read Ratio	Logical Reads is the sum of all Buffer Pool Data Logical Reads and Buffer Pool Index Logical Reads. Logical Read Ratio is the percentage of all reads that were logical reads.
Physical Read Ratio	Physical Reads is the sum of all Buffer Pool Data Physical Reads and Buffer Pool Index Physical Reads. Physical Read Ratio is the percentage of all reads that were physical reads.
Direct Write Ratio	Direct Writes are write operations that do not use the buffer pool. Direct Write Ratio is the percentage of all writes that were direct writes.
Buffered Write Ratio	Buffered Writes is the sum of all Buffer Pool Data Writes and Buffer Pool Index Writes. Buffered Write Ratio is the percentage of all writes that were buffered writes.

SQL Statistics Tab for DB2

The SQL Statistics tab contains statistics about the SQL statement for the selected application.

SQL Statistics

Statistic	Description
Binds and Pre-Compiles	This is the number of binds and pre-compiles attempted.
Cursor Block Requests Accepted	This is the number of times a request for an I/O block was accepted.
Cursor Block Requests Rejected	This is the number of times a request for an I/O block at the server was rejected and the request was converted to non-blocked I/O.
Cursors Open Local	This is the number of local cursors currently open for this application, including those cursors counted by Cursors Open Local with Blocking.
Cursors Open Local with Blocking	This is the number of local blocking cursors currently open for this application.
Cursors Open Remote	This is the number of remote cursors currently open for this application, including those cursors counted by Cursors Open Remote with Blocking.
Cursors Open Remote with Blocking	This is the number of remote blocking cursors currently open for this application.
Internal Automatic Rebinds	This is the number of automatic rebinds (or recompiles) that have been attempted.
Internal Commits	This is the total number of commits initiated internally by the database manager.
Internal Rollbacks	This is the total number of rollbacks initiated internally by the database manager.
Internal Deadlock Rollbacks	This is the total number of forced rollbacks initiated by the database manager due to a deadlock. A rollback is performed on the current unit of work in an application selected by the database manager to resolve the deadlock.
Internal Rows Deleted	This is the number of rows deleted from the database as a result of internal activity.
Internal Rows Inserted	This is the number of rows inserted into the database as a result of internal activity caused by triggers.
Internal Rows Updated	This is the number of rows updated from the database as a result of internal activity.
Rows Deleted	This is the number of row deletions attempted.
Rows Inserted	This is the number of row insertions attempted.
Rows Read	This is the number of rows read from the table.
Rows Selected	This is the number of rows that have been selected and returned to the application.
Rows Updated	This is the number of row updates attempted.
Rows Written	This is the number of rows changed (inserted, deleted, or updated) in the table.

Statistic	Description
SQL DDL Statements	This indicates the number of SQL Data Definition Language (DDL) statements that were executed.
SQL Commit Statements	This indicates the total number of SQL COMMIT statements that have been attempted.
SQL Dynamic Statements	This indicates the number of dynamic SQL statements that were attempted.
SQL Failed Statements	This indicates the number of SQL statements that were attempted and failed.
SQL Requests Since Last Commit	This indicates the number of SQL requests submitted since the last commit.
SQL Rollback Statements	This indicates the total number of SQL ROLLBACK statements that have been attempted.
SQL Select Statements	This indicates the number of SQL SELECT statements that were executed.
SQL Static Statements	This indicates the number of static SQL statements that were attempted.
SQL UID Statements	This indicates the number of SQL UPDATE, INSERT, and DELETE statements that were executed.

SQL Distribution Ratios

Column	Description
DDL Statements	This is the percentage of all executed statements that were SQL DDL Statements.
UID Statements	This is the percentage of all executed statements that were SQL UID Statements.
Failed Statements	This is the percentage of all executed statements that were SQL Failed Statements.
Select Statements	This is the percentage of all executed statements that were SQL Select Statements.

Top Sessions View

The following tabbed pages are available on the Top Sessions view:

- Memory Tab
- I/O Tab
- CPU Tab

Memory Tab

The information on the Memory tab of the Top Sessions view depends on the target DBMS:

- Memory Tab for Oracle
- Memory Tab for SQL Server
- Memory Tab for Sybase
- Memory Tab for DB2

Memory Tab for Oracle

- Metrics

It is frequently the case that one or two users cause the majority of run-time problems. The problem could originate with a runaway process, an untuned batch procedure, or other user-initiated operation. User connections can get out of hand with memory consumption, and extreme cases have caused difficulties at both the database and operating system levels (ORA-4030 errors).

The table below describes the information available on the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view for Oracle:

Column	Description
User Name	The logon name the session is using.
SID	The unique Oracle identifier for the session.
Serial #	The serial number assigned to the session.
Status	The status of the session, ACTIVE or INACTIVE.
Machine	The name of the client machine name that the session is using.
PGA Memory (KB)	The Program Global Area (PGA) is a private memory area devoted to housing the global variables and data structures for a single Oracle process, in KB.
UGA Memory (KB)	The User Global Area (UGA) contains session specific information regarding open cursors, state information for packages, database link information, and more. When using Oracle's Multi-threaded Server (MTS), the UGA can be moved up into the SGA, in KB.
Memory Sorts (KB)	The total number of memory sorts a session has performed.
Total Memory (KB)	The memory (PGA + UGA) that the session is consuming, in KB.

NOTE: This information is available on both the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If your database server does not have an overabundance of memory, you should check periodically to see who your heavy memory users are and the total percentage of memory each takes up. If you see that one or two users use more than 5-15 percent of the total memory, you should investigate the sessions further to see what activities they are performing.

Memory Tab for SQL Server

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. User connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system levels.

Embarcadero Performance Center displays information to find processes that are using the most memory on the server on the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view.

The table below describes the information available on these tabs:

Column	Description
PID	The process ID of the connected session.
User Name	The user name assigned to the process.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The machine name that originated the process.
Program	The program the process has invoked against SQL Server.
Memory Usage	The total number of memory pages allocated to the process.
Pct Mem Used	The percentage of overall memory among all processes that can be attributed to the process.
Database	The database in which the process is currently running.
Command	The command the process is currently issuing.
Open Trans	The number of open transactions for the process.
Blocked	The PID of any process blocking the current process.
Wait Time	The current wait time for the process, in milliseconds.

NOTE: This information is available on both the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If your database server does not have an overabundance of memory, you should periodically check to see who your heavy memory users are along with the total percentage of memory each takes up. If you see one or two users who have more than 5-15 percent of the total memory usage, you should investigate the sessions further to see what activities they are performing.

Memory Tab for Sybase

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Often, user connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system levels.

This tab displays information to find processes that are using the most memory on the server. The table below describes the information available on the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view for Sybase:

Column	Description
PID	The process ID.
User Name	The user name assigned to the process.
FID	The process ID of the worker process' parent.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Memory Usage	The memory currently used by the process.
Pct Mem Used	The percentage of memory currently used by the process.
Database	The database in which the process is running.
Command	The command the process is currently issuing.
Transaction	The name of any transaction.
Blocked	The PID of any process blocking the current process.
Time Blocked	The time that the process has been blocked, in seconds.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If your database server does not have an overabundance of memory, you should periodically check to see who your heavy memory users are along with the total percentage of memory each takes up. If you see one or two users who have more than 5-15 percent of the total memory usage, you should investigate the sessions further to see what activities they are performing.

Memory Tab for DB2

- Metrics

It is not uncommon for one or two users to cause the majority of runtime problems that plague a database. The problem could be a runaway process, an untuned batch procedure, or other user-initiated operation. Often, user connections can get out of hand with memory consumption, and extreme cases have caused headaches at both the database and operating system levels.

This tab displays information to find processes that are using the most memory on the server. The table below describes the information DB2 on the Leading Sessions tab of the Memory Detail view and the Memory tab of the Top Sessions view for DB2:

Column	Description
Agent ID	The application handle of the agent holding a lock for which this application is waiting.
Auth ID	The authorization ID of the user who invoked the application that is being monitored.
OS User ID	The operating system ID of the process.
Client PID	The process ID of the client application that made the connection to the database.
Application	The name of the application executable.
Status	The current status of the application.
UOW Elapsed Time (sec)	The elapsed execution time of the most recently completed unit of work.
Memory Overflows	Total number of memory overflows.
Memory Used (KB)	Total memory pool usage for the application.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

If your database server does not have an overabundance of memory, you should periodically check to see who your heavy memory users are along with the total percentage of memory each takes up. If you see one or two users who have more than 5-15 percent of the total memory usage, you should investigate the sessions further to see what activities they are performing.

I/O Tab

The information on the I/O tab of the Top Sessions view depends on the target DBMS:

- I/O Tab for DB2
- I/O Tab for SQL Server
- I/O Tab for Sybase
- I/O Tab for DB2

I/O Tab for Oracle

- Metrics

When a system undergoes heavy I/O activity, all the user connections can contribute somewhat equally to the overall load. Frequently, however, one or two user connections are responsible for 75 percent or more of the I/O activity. It may be that a large batch load or other typical process is running, and that is perfectly okay for your system. Or, it may be a runaway process or rogue connection that you should track down and eliminate.

It is a good idea to periodically check the leading sessions in your system with respect to I/O and make sure all is well. You can use Embarcadero Performance Center to perform this function with the information available on this tab. The table below describes the information available on the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view for Oracle:

Column	Description
User Name	The logon name the session is using.
SID	The unique Oracle identifier for the session.
Serial #	The serial number assigned to the session.
Status	The status of the session, ACTIVE or INACTIVE.
Machine	The name of the client machine name that the session is using.
Reads	The number of physical reads.
Writes	The number of physical writes.
Total I/O	The total of all physical I/O operations for the session.

NOTE: This information is available on both the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Pinpointing sessions with abnormally high I/O activity relative to other sessions in the system can aid in ferreting out accounts that are dragging down overall system performance. You should examine the activity of each session to determine the system workload and to see if you can reduce the workload or tune the system for better performance.

I/O Tab for SQL Server

- Metrics

When a system undergoes heavy I/O activity, sometimes you find that all the user connections are contributing somewhat equally to the overall load. Frequently, however, one or two user connections are responsible for 75 percent or more of the I/O activity. It can be that a large batch load or other typical process is running that is perfectly okay for your system. Alternatively, it can be a runaway process or other rogue connection that you should track down and possibly eliminate.

Embarcadero Performance Center displays information to find processes that are using the most memory on the server on the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view.

The table below describes the information available on these tabs:

Column	Description
PID	The process ID.
User Name	The user name assigned to the process.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Physical I/O	The current cumulative number of reads and writes issued by the process.
Pct I/O Used	The percentage of overall I/O that can be attributed to the process.
Database	The database in which the process is running.
Command	The command the process is currently issuing.
Open Trans	The number of open transactions for the process.
Blocked	The PID of any process blocking the current process.
Wait Time	The current wait time for the process, in milliseconds.

NOTE: This information is available on both the Leading Sessions tab of the I/O Detail view and the I/O tab of the Top Sessions view.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Finding one more two users that are consuming more than 75 percent of the total I/O load can indicate a runaway or improper process. By drilling down into the I/O activity of all users, you can quickly see if this is the case.

I/O Tab for Sybase

- Metrics

When a system undergoes heavy I/O activity, sometimes you find that all the user connections are contributing somewhat equally to the overall load. More often than not, however, one or two user connections are responsible for 75 percent or more of the I/O activity. It can be that a large batch load or other typical process is running that is perfectly okay for your system. Or it can be a runaway process or other rogue connection that needs to be tracked down and possibly eliminated.

It is a good idea to periodically check who the leading sessions are in your system with respect to I/O and make sure all is well. You can use Performance Center to easily perform this function with the leading sessions tab of the I/O Detail view. The table below describes the information available on the I/O tab of the Top Sessions view for Sybase:

Column	Description
PID	The process ID.
User Name	The user name assigned to the process.
FID	The process ID of the worker process' parent.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
Physical I/O	The current cumulative number of reads and writes issued by the process.
Pct I/O Used	The percentage of overall I/O that can be attributed to the process.
Database	The database in which the process is running.
Command	The command the process is currently issuing.
Transaction	The name of any transaction.
Blocked	The PID of any process blocking the current process.
Time Blocked	The time that the process has been blocked, in seconds.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Pinpointing sessions with abnormally high I/O activity relative to other sessions in the system help you ferret out accounts dragging down overall system performance. You should examine the activity of each session to determine the workload being placed on the system and if you can reduce or tune that workload for better performance.

I/O Tab for DB2

- Metrics

When a system undergoes heavy I/O activity, sometimes you find that all the user connections are contributing somewhat equally to the overall load. More often than not, however, one or two user connections are responsible for 75 percent or more of the I/O activity. It can be that a large batch load or other typical process is running that is perfectly okay for your system. Or it can be a runaway process or other rogue connection that needs to be tracked down and possibly eliminated.

It is a good idea to periodically check who the leading sessions are in your system with respect to I/O and make sure all is well. You can use Performance Center to easily perform this function with the leading sessions tab of the I/O Detail view. The table below describes the information available on the I/O tab of the Top Sessions view for DB2:

Column	Description
Agent ID	The application handle of the agent holding a lock for which this application is waiting.
Auth ID	The authorization ID of the user who invoked the application that is being monitored.
OS User ID	The operating system ID of the process.
Client PID	The process ID of the client application that made the connection to the database.
Application	The name of the application executable.
Status	The current status of the application.
UOW Elapsed Time (sec)	The elapsed execution time of the most recently completed unit of work.
Buffered I/O Time (ms)	The total time spent by application in performing buffered reads and writes.
Direct I/O Time (ms)	The total time spent by application in performing non-buffered reads and writes.
Total I/O Time (ms)	The total time spent by application in performing buffered and non-buffered reads and writes.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

Metrics

Pinpointing sessions with abnormally high I/O activity relative to other sessions in the system help you ferret out accounts dragging down overall system performance. You should examine the activity of each session to determine the workload being placed on the system and if you can reduce or tune that workload for better performance.

CPU Tab

The information on the CPU tab of the Top Sessions view depends on the target DBMS:

- CPU Tab for Oracle
- CPU Tab for SQL Server
- CPU Tab for Sybase
- CPU Tab for DB2

CPU Tab for Oracle

The table below describes the information available on the CPU tab of the Top Sessions view for Oracle:

Column	Description
User Name	The logon name the session is using.
SID	The unique Oracle identifier for the session.
Serial #	The serial number assigned to the session.
Status	The status of the session, ACTIVE or INACTIVE.
Machine	The name of the client machine name that the session is using.
Program	The executable the process is using against the server.
CPU	The CPU used by the process when the call started.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

CPU Tab for SQL Server

The table below describes the information available on the CPU tab of the Top Sessions view for SQL Server:

Column	Description
PID	The process ID.
User Name	The user name assigned to the process.
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
CPU	The cumulative CPU time for the process.
Pct CPU Used	The percentage of the CPU dedicated to this process.
Database	The database in which the process is running.
Command	The command the process is currently issuing.
Open Trans	The number of open transactions for the process.
Blocked	The PID of any process blocking the current process.
Wait Time	The current wait time for the process, in milliseconds.

TIP: To configure the grid to show/hide row numbers, use the Options Editor.

CPU Tab for Sybase

The table below describes the information available on the CPU tab of the Top Sessions view for Sybase:

Column	Description
PID	The process ID.
User Name	The user name assigned to the process.
FID	The process ID of the worker process' parent.

Column	Description
Status	Indicates if the process is actively performing work, is idle, blocked by another process, etc.
Host	The machine name that originated the process.
Program	The executable the process is using against the server.
CPU	The cumulative CPU time for the process in ticks.
Pct CPU Used	The percentage of the CPU dedicated to this process.
Database	The database in which the process is running.
Command	The command the process is currently issuing.
Transaction	The name of any transaction.
Blocked	The PID of any process blocking the current process.
Time Blocked	The time that the process has been blocked, in seconds.

To configure the grid to show/hide row numbers, use the Options Editor.

CPU Tab for DB2

The table below describes the information available on the CPU tab of the Top Sessions view for DB2:

Column	Description
Agent ID	The application handle of the agent holding a lock for which this application is waiting.
Auth ID	The authorization ID of the user who invoked the application that is being monitored.
OS User ID	The operating system ID of the process.
Client PID	The process ID of the client application that made the connection to the database.
Application	The name of the application executable.
Status	The current status of the application.
UOW Elapsed Time (sec)	The elapsed execution time of the most recently completed unit of work.
Agent ID	The application handle of the agent holding a lock for which this application is waiting.
User CPU Time (sec)	The total user CPU time used by the application agents.
System CPU Time (sec)	The total system CPU time used by the application agents.
Total CPU Time (sec)	The total user + system CPU time used by the application agents.

To configure the grid to show/hide row numbers, use the Options Editor.

Top SQL View

- Metrics

One or two bad queries can cause a lot of trouble for the remaining sessions in a database. It is important to find them before they get into a production system, but sometimes a few sneak through.

By applying custom filters and performance-related thresholds, the Top SQL view locates inefficient SQL. By applying filters to certain I/O and statistical counters, you hope to isolate queries that far exceed their nearest competitors in the same area (like disk reads). When you find them, you should reduce the number of sorts a query performs. Or, for a query that returns only a few records, you should try to minimize the number of rows a query processes.

The Top SQL view displays requested SQL for SQL Server, Oracle, DB2, and Sybase datasources.

Metrics

When you begin to look for inefficient SQL in a database, there are two primary questions you need to answer:

1 What has been the worst SQL that has historically been run in my database?

2 What is the worst SQL that's running right now in my database?

When troubleshooting a slow system, you should be on the lookout for any query that shows an execution count that is significantly larger than any other query on the system. It may be that the query is in an inefficient Transact SQL loop, or other problematic programming construct. Only by bringing the query to the attention of the application developers will you know if the query is being mishandled from a programming standpoint.

Glossary

A

Action

A predefined response to a threshold violation.

Alarms View

An HTML window where you can see alarms, for all monitored datasources. In the Alarms view, you can set alarms for any threshold violation. When a violation occurs, Embarcadero Performance Center initiates an alarm and logs it for viewing.

Alert Logs View

An HTML window where you can see a list of your Oracle Alert Logs from the Embarcadero Performance Center application. It includes filters, which let you customize your display for specific Alert Logs. The Embarcadero Performance Center Alert Log view lets you view a list of your Oracle Alert.

Archive View

An HTML window where you can see lists of all archived log files written out by the database.

Analyzer

The Embarcadero Performance Center Server internal process that performs threshold validation.

Apache

A public-domain Web server developed by a loosely-knit group of programmers known as the Apache Group. Apache httpd Server is based on a popular, public-domain HTTP daemon developed in 1995 by Rob McCool at the National Center for Supercomputing Applications, University of Illinois, Urbana-Champaign. Because it was developed from existing NCSA code and various patches, it was called “a patchy” server - hence the name “Apache”.

Embarcadero Performance Center uses Apache 2.0.35 to communicate between the Embarcadero Performance Center Server and Embarcadero Performance Center Clients.

B

Blackout

A schedule that is a time interval where Performance Center will not send notifications about any normal threshold violations or performance problems.

C

Capture Agent

The Embarcadero Performance Center Server internal process that retrieves values for a certain list of statistics and then writes them to the repository.

Category

Type of statistic.

Embarcadero Performance Center breaks up all of the statistics into eight categories, Memory, I/O, Space, Objects, Databases, Contention, Users, and Network. Each of these categories has a performance category view. All of the statistics for each category are available on the individual performance category views.

Clone

A copy of a datasource including all of the properties (thresholds, notifications etc.) from the original datasource.

D

Datasource Manager

The Embarcadero Performance Center Server internal datasource connection pool manager.

DBWR

Database Writer Process.

The DBWR process handles the flow of information from Oracle physical datafiles to and from the various memory structures in the system global area (SGA).

E

Enterprise

The network (or interconnected networks) of computer systems. It can span diverse geographical locations and usually encompasses a range of platforms, operating systems, protocols, and network architectures.

Enterprise View

An HTML window designed to give you a comprehensive picture of the overall health of all monitored datasources in your enterprise. It displays all currently registered datasources. The Enterprise view lets you know what databases are up and down and gives you other critical indicators regarding the status of your monitored datasources.

F

There are no entries for this letter.

G

There are no entries for this letter.

H

Health Index

The set of statistics used to define the various performance indexes used to globally measure database performance. The Health Index provides important statistics for monitoring the performance of datasources in an enterprise.

Health Index Template

Logical containers that define the various performance indexes used to globally measure database performance.

Home View

An HTML window where you can see the statistic category vital signs. The Home view is the default performance category view.

I

I/O

Input/Output.

Physical I/O is read to/from a disk. Logical I/O is read to/from memory.

Init.ora

The Oracle Initialization Parameter File. It is a text file containing a list of parameters and values for each parameter. You can modify many initialization parameters to improve database performance.

J

There are no entries for this letter.

K

There are no entries for this letter

L

LGWR

Log Writer Process.

The LGWR manages the information contained in Oracle online redo log files and redo log buffer area.

Login

A login is a form of identification that lets you connect to a datasource. Permissions within specific databases are controlled by user accounts. The database administrator maps your login account to a user account in any database you are authorized to access.

Logins are one of the fundamental security mechanisms of Embarcadero Performance Center. Users who connect to an Embarcadero Performance Center Server must identify themselves using a specific login identifier (ID). Users can then only see the tables and views they are authorized to see. To view Embarcadero Performance Center, a user need not login to the Embarcadero Performance Center Server. However, to add, modify, delete and access other functionality, a user MUST login to the target Embarcadero Performance Center Server WITH administrative privileges.

M

MDAC

Microsoft Data Access Component.

Embarcadero Performance Center Server uses MDAC to establish a connection to the repository database and monitored datasources.

MTS

Multi-Threaded Server.

Oracle MTS configuration permits many user processes to share very few server processes. The user processes connect to a dispatcher background process, which routes client requests to the next available shared server process.

N

Notification

Reusable messages that you tell Embarcadero Performance Center to send in response to threshold violations.

Notifier

The Embarcadero Performance Center Server internal process that is responsible for sending the emergency contact messages.

O

Oracle Alert Log

An operating system file containing vital information on the Oracle database. The alert log contains information messages, including messages regarding the startup and shutdown of the database, errors that have occurred in the database. When a database is online Oracle continuously writes to the alert log.

P

Performance Category View

HTML windows where you can see statistics for each category of statistic. Every performance category view displays the Health Index and includes links to the Alert Log and Alarm views.

Pin

To hold fast; immobilize.

Marking an object to be pinned means Embarcadero Performance Center Server does not flush the pages for the object from memory. Pinning an object loads the object instance into the object cache, letting you access and modify the instance's attributes and follow references from that object to other objects.

Purge

To systematically and permanently remove old and unneeded data from an Embarcadero Performance Center Server. Purge is different from delete in that it is possible to regain deleted objects but purged objects are gone forever.

Q

There are no entries for this letter.

R

Repository

The Microsoft SQL Server, Oracle, DB2, or Sybase database server that Embarcadero Performance Center uses as a repository to store information used during monitoring.

Requestor

The Embarcadero Performance Center Server internal process that is responsible for collecting datasource data.

S

SID

System identifier.

A unique number that Oracle gives to a session.

Statistic

Individual performance measurements that Embarcadero Performance Center uses to gauge the overall health of the database.

T

Threshold
Default or user-defined values that mark performance boundaries. Thresholds are the point that must be exceeded to begin producing a user-defined effect or result or to elicit a response.

Threshold Template
Logical containers used to define and deploy thresholds to monitored datasources.

U

There are no entries for this letter.

V

Views
HTML windows in Embarcadero Performance Center that are designed to provide you with several different ways to monitor your enterprise. The views include a combination of graphics and a color coded severity level, which let you quickly determine the health of all registered datasources in your enterprise.

W

WSH
Windows Scripting Host.

A Microsoft tool that the Embarcadero Performance Center Server uses to run Visual Basic scripts.

X

There are no entries for this letter.

Y

There are no entries for this letter.

Z

There are no entries for this letter.

Index

H

I

K

L

M

N

O

P

R

S

T