Name	Description
background checkpoints completed	Number of checkpoints completed by the background process. This statistic is incremented when the background process successfully advances the thread checkpoint.
background checkpoints started	Number of checkpoints started by the background process. This statistic can be larger than "background checkpoints completed" if a new checkpoint overrides an incomplete checkpoint or if a checkpoint is currently under way. This statistic includes only checkpoints of the redo thread. It does not include: Individual file checkpoints for operations such as offline or begin backup Foreground (user-requested) checkpoints (for example, performed by ALTER SYSTEM CHECKPOINT LOCAL statements)
background timeouts	This is a count of the times where a background process has set an alarm for itself and the alarm has timed out rather than the background process being posted by another process to do some work.
branch node splits	Number of times an index branch block was split because of the insertion of an additional value
buffer is not pinned count	Number of times a buffer was free when visited. Useful only for internal debugging purposes.
buffer is pinned count	Number of times a buffer was pinned when visited. Useful only for internal debugging purposes.
bytes received via SQL*Net from client	Total number of bytes received from the client over Oracle Net Services
bytes received via SQL*Net from dblink	Total number of bytes received from a database link over Oracle Net Services
bytes sent via SQL*Net to client	Total number of bytes sent to the client from the foreground processes
bytes sent via SQL*Net to dblink	Total number of bytes sent over a database link
Cached Commit SCN referenced	Useful only for internal debugging purposes
calls to get snapshot scn: kcmgss	Number of times a snapshot system change number (SCN) was allocated. The SCN is allocated at the start of a transaction.
calls to kcmgas	Number of calls to routine kcmgas to get a new SCN
calls to kcmgcs	Number of calls to routine kcmgcs to get a current SCN
calls to kcmgrs	Number of calls to routine kcsgrs to get a recent SCN
change write time	Elapsed redo write time for changes made to CURRENT blocks in 10s of milliseconds. This statistic is populated only if TIME_STATISTICS is set to true.
cleanouts and rollbacks - consistent read gets	Number of consistent gets that require both block rollbacks and block cleanouts.
cleanouts only - consistent read gets	Number of consistent gets that require only block cleanouts, no rollbacks.
cluster key scan block gets	Number of blocks obtained in a cluster scan
cluster key scans	Number of cluster scans that were started
cold recycle reads	Number of buffers that were read through the least recently used end of the recycle cache with fast aging strategy
commit cleanout failures: block lost	Number of times Oracle attempted a cleanout at commit but could not find the correct block due to forced write, replacement, or switch CURRENT
commit cleanout failures: buffer being written	Number of times Oracle attempted a cleanout at commit, but the buffer was currently being written
commit cleanout failures: callback failure	Number of times the cleanout callback function returns FALSE
commit cleanout failures: cannot pin	Total number of times a commit cleanout was performed but failed because the block could not be pinned

commit cleanout failures: hot backup in progress	Number of times Oracle attempted block cleanout at commit during hot backup. The image of the block needs to be logged before the buffer can be made dirty.
commit cleanout failures: write disabled	Number of times a cleanout block at commit was performed but the writes to the database had been temporarily disabled
commit cleanouts	Total number of times the cleanout block at commit function was performed
commit cleanouts successfully completed	Number of times the cleanout block at commit function completed successfully
Commit SCN cached	Number of times the system change number of a commit operation was cached
consistent changes	Number of times a user process has applied rollback entries to perform a consistent read on the block Work loads that produce a great deal of consistent changes can consume a great deal of resources. The value of this statistic should be small in relation to the "consistent gets" statistic.
consistent gets	Number of times a consistent read was requested for a block.
consistent gets direct	Number of times a consistent read was requested for a block bypassing the buffer cache (for example, direct load operation). This is a subset of "consistent gets" statistics value.
consistent gets from cache	Number of times a consistent read was requested for a block from buffer cache. This is a subset of "consistent gets" statistics value.
CPU used by this session	Amount of CPU time (in 10s of milliseconds) used by a session from the time a user call starts until it ends. If a user call completes within 10 milliseconds, the start and end user-call time are the same for purposes of this statistics, and 0 milliseconds are added. A similar problem can exist in the reporting by the operating system, especially on systems that suffer from many context switches.
CPU used when call started	The CPU time used when the call is started
CR blocks created	Number of CURRENT blocks cloned to create CR (consistent read) blocks. The most common reason for cloning is that the buffer is held in a incompatible mode.
current blocks converted for CR	Number CURRENT blocks converted to CR state
cursor authentications	Number of privilege checks conducted during execution of an operation
data blocks consistent reads - undo records applied	Number of undo records applied to data blocks that have been rolled back for consistent read purposes
db block changes	Closely related to "consistent changes", this statistic counts the total number of changes that were part of an update or delete operation that were made to all blocks in the SGA. Such changes generate redo log entries and hence become permanent changes to the database if the transaction is committed. This approximates total database work. It statistic indicates the rate at which buffers are being dirtied (on a per-transaction or per-second basis, for example).
db block gets	Number of times a CURRENT block was requested
db block gets direct	Number of times a CURRENT block was requested bypassing the buffer cache (for example, a direct load operation). This is a subset of "db block gets" statistics value.
db block gets from cache	Number of times a CURRENT block was requested from the buffer cache. This is a subset of "db block gets" statistics value.

DBWR buffers scanned	Total number of dirty and clean buffers Oracle looks at when scanning LRU sets for dirty buffers to clean. Divide by "DBWR lru scans" to find the average number of buffers scanned.
DBWR checkpoint buffers written	Number of buffers that were written for checkpoints
DBWR checkpoints	Number of times the DBWR was asked to scan the cache and write all blocks marked for a checkpoint or the end of recovery. This statistic is always larger than "background checkpoints completed".
DBWR cross instance writes	Real Application Clusters only: Number of blocks written to satisfy a lock request from another instance
DBWR free buffers found	Number of clean buffers that DBWR found when it was requested to make free buffers. Divide by "DBWR make free requests" to find the average number of reusable buffers at the end of each LRU.
DBWR lru scans	Number of times that DBWR scans the LRU queue looking for buffers to write. This count includes scans to fill a batch being written for another purpose (such as a checkpoint). This statistic is always greater than or equal to "DBWR make free requests".
DBWR make free requests	Number of requests to DBWR to make some free buffers for the LRU
DBWR revisited being-written buffer	Number of times that DBWR tried to save a buffer for writing and found that it was already in the write batch. This statistic measures the amount of "useless" work that DBWR had to do in trying to fill the batch. Many sources contribute to a write batch. If the same buffer from different sources is considered for adding to the write batch, then all but the first attempt will be "useless" because the buffer is already marked as being written.
DBWR summed scan depth	The current scan depth (number of buffers examined by DBWR) is added to this statistic every time DBWR scans the LRU for dirty buffers. Divide by "DBWR lru scans" to find the average scan depth.
DBWR transaction table writes	Number of rollback segment headers written by DBWR. This statistic indicates how many "hot" buffers were written, causing a user process to wait while the write completed.
DBWR undo block writes	Number of rollback segment blocks written by DBWR
DDL statements parallelized	Number of DDL statements that were executed in parallel
deferred (CURRENT) block cleanout applications	Number of times cleanout records are deferred, piggyback with changes, always current get
DFO trees parallelized	Number of times a serial execution plan was converted to a parallel plan
dirty buffers inspected	Number of dirty buffers found by the user process while the it is looking for a buffer to reuse
DML statements parallelized	Number of DML statements that were executed in parallel
enqueue conversions	Total number of conversions of the state of table or row lock
enqueue deadlocks	Total number of deadlocks between table or row locks in different sessions
enqueue releases	Total number of table or row locks released
enqueue requests	Total number of table or row locks acquired
enqueue timeouts	Total number of table and row locks (acquired and converted) that timed out before they could complete
enqueue waits	Total number of waits that occurred during an enqueue convert or get because the enqueue get was deferred

exchange deadlocks	Number of times that a process detected a potential deadlock when exchanging two buffers and raised an internal, restartable error. Index scans are the only operations that perform exchanges.
execute count	Total number of calls (user and recursive) that executed SQL statements
free buffer inspected	Number of buffers skipped over from the end of an LRU queue in order to find a reusable buffer. The difference between this statistic and "dirty buffers inspected" is the number of buffers that could not be used because they had a user, a waiter, or were being read or written, or because they were busy or needed to be written after rapid aging out.
free buffer requested	Number of times a reusable buffer or a free buffer was requested to create or load a block
global cache blocks corrupt	Real Application Clusters only: Number of blocks that encountered a corruption or checksum failure during interconnect
global cache convert time	Real Application Clusters only: Total time elapsed during lock converts
global cache convert timeouts	Number of times lock converts in the global cache timed out
global cache converts	Number of lock converts in the global cache
global cache cr block log flushes	Number of log flushes of the consistent-read block
global cache cr block log flush time	Total time spent by the BSP process in log flushes after sending a constructed consistent-read (CR) block. This statistic divided by "global cache cr blocks served" = log flush time per CR block.
global cache cr block receive time	Total amount of time foreground processes waited for a CR block to be sent through the interconnect. This statistic divided by "global cache cr blocks received" = time waited per block.
global cache cr block send time	Total time spent by the BSP process in sending constructed consistent-read (CR) blocks. This statistic divided by "global cache cr blocks served" = send time per CR block.
global cache cr block serve time	Total amount of time the BSP process took to construct consistent-read (CR) blocks. This statistic divided by "global cache cr blocks served" = construction time per CR block.
global cache cr blocks received	Total number of blocks received
global cache cr blocks served	Total number of blocks constructed by the BSP process
global cache cr requests blocked	Number of times foreground attempt to request a cr block and failed
global cache cr timeouts	Number of times a foreground process requested a consistent-read (CR) block when the request timed out
global cache defers	Number of times a lock was requested and the holder of the lock deferred the release
global cache freelist waits	System configured with fewer lock elements than buffers. Number of times foreground has to wait for a lock element.
global cache get time	Total time spent waiting. This divided by global cache gets = time waited per request.
global cache gets	Number of locks acquired
global cache prepare failures	Number of times a failure occurred during preparation for interconnect transfer
global lock async converts	Total number of asynchronous global lock converts
global lock async gets	Total number of asynchronous global lock gets
global lock convert time	Total elapsed time in 10s of milliseconds of all synchronous and asynchronous global lock converts
global lock get time	Total elapsed time in 10s of milliseconds of all synchronous and asynchronous global lock gets
global lock releases	Total number of synchronous global lock releases
global lock sync converts	Total number of synchronous global lock converts
global lock sync gets	Total number of synchronous global lock gets

hot buffers moved to head of LRU	When a hot buffer reaches the tail of its replacement list, Oracle moves it back to the head of the list to keep it from being reused. This statistic counts such moves.
immediate (CR) block cleanout applications	Number of times cleanout records are applied immediately during consistent-read requests
immediate (CURRENT) block cleanout applications	Number of times cleanout records are applied immediately during current gets. Compare this statistic with "deferred (CURRENT) block cleanout applications"
index fast full scans (direct read)	Number of fast full scans initiated using direct read
index fast full scans (full)	Number of fast full scans initiated for full segments
index fast full scans (rowid ranges)	Number of fast full scans initiated with rowid endpoints specified
instance recovery database freeze count	Number of times the database is frozen during instance recovery
kcmccs called get current scn	Number of times the kernel got the CURRENT SCN when there was a need to casually confirm the SCN
kcmgss read scn without going to DLM	Number of times the kernel got a snapshot SCN without going to the distributed lock manager (DLM)
kcmgss waited for batching	Number of times a database process is blocked waiting for a snapshot SCN
leaf node splits	Number of times an index leaf node was split because of the insertion of an additional value
lob reads	Number of LOB API read operations performed in the session/system. A single LOB API read may correspond to multiple physical/logical disk block reads.
lob writes	Number of LOB API write operations performed in the session/system. A single LOB API write may correspond to multiple physical/logical disk block writes.
lob writes unaligned	Number of LOB API write operations whose start offset or buffer size is not aligned to the internal chunk size of the LOB. Writes aligned to chunk boundaries are the most efficient write operations. The internal chunk size of a LOB is available through the LOB API (for example, DBMS_LOB.GETCHUNKSIZE()).
logons cumulative	Total number of logons since the instance started. Useful only in V$SYSSTAT. It gives an instance overview of all processes that logged on.
logons current	Total number of current logons. Useful only in V$SYSSTAT.
messages received	Number of messages sent and received between background processes
messages sent	Number of messages sent and received between background processes
native hash arithmetic execute	Number of hash operations performed using native arithmetic rather than Oracle NUMBERs
native hash arithmetic fail	Number of has operations performed using native arithmetic that failed, requiring the hash operation to be performed with Oracle NUMBERs
next scns gotten without going to DLM	Number of system change numbers obtained without going to the distributed lock manager or server
no buffer to keep pinned count	Number of times a visit to a buffer attempted, but the buffer was not found where expected. Like "buffer is not pinned count" and "buffer is pinned count", this statistic is useful only for internal debugging purposes.
no work - consistent read gets	Number consistent gets that require neither block cleanouts nor rollbacks.
opened cursors cumulative	In V$SYSSTAT: Total number of cursors opened since the instance started. In V$SESSTAT: Total number of cursors opened since the start of the session.
opened cursors current	Total number of current open cursors

opens of replaced files	Total number of files that had to be reopened because they were no longer in the process file cache
opens requiring cache replacement	Total number of file opens that caused a current file in the process file cache to be closed
OS All other sleep time	Time spent sleeping for reasons other than misses in the data segment (see "OS Data page fault sleep time"), kernel page faults (see "OS Kernel page fault sleep time"), misses in the text segment (see "OS Text page fault sleep time"), or waiting for an OS locking object (see "OS User lock wait sleep time"). An example of such a reason is expiration of quanta.
OS Chars read and written	Number of bytes read and written
OS Data page fault sleep time	Time spent sleeping due to misses in the data segment
OS Input blocks	Number of read I/Os
OS Involuntary context switches	Number of context switches that were enforced by the operating system
OS Kernel page fault sleep time	Time spent sleeping due to OS kernel page faults
OS Major page faults	Number of page faults that resulted in I/O
OS Messages received	Number of messages received
OS Messages sent	Number of messages sent
OS Minor page faults	Number of page faults that did not result in an actual I/O
OS Other system trap CPU time	Total amount of time to process system traps (as distinct from system calls)
OS Output blocks	Number of write I/Os
OS Process heap size	Size of area in memory allocated by the process. Typically this represents memory obtained by way of malloc().
OS Process stack size	Size of the process stack segment
OS Signals received	Number of signals received
OS Swaps	Number of swap pages
OS System call CPU time	Total amount of time spent executing in system mode
OS System calls	Number of system calls
OS Text page fault sleep time	Time spent sleeping due to misses in the text segment
OS User level CPU time	Total amount of time spent executing in user mode
OS User lock wait sleep time	Total amount of time sleeping while waiting for an OS locking object
OS Voluntary context switches	Number of voluntary context switches (for example, when a process gives up the CPU by a SLEEP() system call)
OS Wait-cpu (latency) time	Time spent sleeping while waiting for a CPU to become available
Parallel operations downgraded 1 to 25 pct	Number of times parallel execution was requested and the degree of parallelism was reduced because of insufficient parallel execution servers
Parallel operations downgraded 25 to 50 pct	Number of times parallel execution was requested and the degree of parallelism was reduced because of insufficient parallel execution servers
Parallel operations downgraded 50 to 75 pct	Number of times parallel execution was requested and the degree of parallelism was reduced because of insufficient parallel execution servers
Parallel operations downgraded 75 to 99 pct	Number of times parallel execution was requested and the degree of parallelism was reduced because of insufficient parallel execution servers
Parallel operations downgraded to serial	Number of times parallel execution was requested but execution was serial because of insufficient parallel execution servers
Parallel operations not downgraded	Number of times parallel execution was executed at the requested degree of parallelism
parse count (hard)	Total number of parse calls (real parses). A hard parse is a very expensive operation in terms of memory use, because it requires Oracle to allocate a workheap and other memory structures and then build a parse tree.

parse count (total)	Total number of parse calls (hard and soft). A soft parse is a check on an object already in the shared pool, to verify that the permissions on the underlying object have not changed.
parse time cpu	Total CPU time used for parsing (hard and soft) in 10s of milliseconds
parse time elapsed	Total elapsed time for parsing, in 10s of milliseconds. Subtract "parse time cpu" from the this statistic to determine the total waiting time for parse resources.
physical reads	Total number of data blocks read from disk. This value can be greater than the value of "physical reads direct" plus "physical reads cache" as reads into process private buffers also included in this statistic.
physical read bytes	Total size in bytes of all disk reads by application activity (and not other instance activity) only.
physical read IO requests	Number of read requests for application activity (mainly buffer cache and direct load operation) which read one or more database blocks per request. This is a subset of "physical read total IO requests" statistic.
physical read total bytes	Total size in bytes of disk reads by all database instance activity including application reads, backup and recovery, and other utilities. The difference between this value and "physical read bytes" gives the total read size in bytes by non-application workload.
physical read total IO requests	Number of read requests which read one or more database blocks for all instance activity including application, backup and recovery, and other utilities. The difference between this value and "physical read total multi block requests" gives the total number of single block read requests.
physical read total multi block requests	Total number of Oracle instance read requests which read in two or more database blocks per request for all instance activity including application, backup and recovery, and other utilities.
physical reads cache	Total number of data blocks read from disk into the buffer cache. This is a subset of "physical reads" statistic.
physical reads direct	Number of reads directly from disk, bypassing the buffer cache. For example, in high bandwidth, data-intensive operations such as parallel query, reads of disk blocks bypass the buffer cache to maximize transfer rates and to prevent the premature aging of shared data blocks resident in the buffer cache.
physical reads prefetch warmup	Number of data blocks that were read from the disk during the automatic prewarming of the buffer cache.
physical write bytes	Total size in bytes of all disk writes from the database application activity (and not other kinds of instance activity).
physical write IO requests	Number of write requests for application activity (mainly buffer cache and direct load operation) which wrote one or more database blocks per request.
physical write total bytes	Total size in bytes of all disk writes for the database instance including application activity, backup and recovery, and other utilities. The difference between this value and "physical write bytes" gives the total write size in bytes by non-application workload.
physical write total IO requests	Number of write requests which wrote one or more database blocks from all instance activity including application activity, backup and recovery, and other utilities. The difference between this stat and "physical write total multi block requests" gives the number of single block write requests.

physical write total multi block requests	Total number of Oracle instance write requests which wrote two or more blocks per request to the disk for all instance activity including application activity, recovery and backup, and other utilities.
physical writes	Total number of data blocks written to disk. This statistics value equals the sum of "physical writes direct" and "physical writes from cache" values.
physical writes direct	Number of writes directly to disk, bypassing the buffer cache (as in a direct load operation)
physical writes from cache	Total number of data blocks written to disk from the buffer cache. This is a subset of "physical writes" statistic.
physical writes non checkpoint	Number of times a buffer is written for reasons other than advancement of the checkpoint. Used as a metric for determining the I/O overhead imposed by setting the FAST_START_IO_TARGET parameter to limit recovery I/Os. (Note that FAST_START_IO_TARGET is a deprecated parameter.) Essentially this statistic measures the number of writes that would have occurred had there been no checkpointing. Subtracting this value from "physical writes" gives the extra I/O for checkpointing.
pinned buffers inspected	Number of times a user process, when scanning the tail of the replacement list looking for a buffer to reuse, encountered a cold buffer that was pinned or had a waiter that was about to pin it. This occurrence is uncommon, because a cold buffer should not be pinned very often.
prefetched blocks	Number of contiguous and noncontiguous blocks that were prefetched
prefetched blocks aged out before use	Number of contiguous and noncontiguous blocks that were prefetched but aged out before use
process last non-idle time	The last time this process executed
PX local messages recv'd	Number of local messages received for parallel execution within the instance local to the current session
PX local messages sent	Number of local messages sent for parallel execution within the instance local to the current session
PX remote messages recv'd	Number of remote messages received for parallel execution within the instance local to the current session
PX remote messages sent	Number of remote messages sent for parallel execution within the instance local to the current session
queries parallelized	Number of SELECT statements executed in parallel
recovery array read time	Elapsed time of I/O during recovery
recovery array reads	Number of reads performed during recovery
recovery blocks read	Number of blocks read during recovery
recursive calls	Number of recursive calls generated at both the user and system level. Oracle maintains tables used for internal processing. When Oracle needs to make a change to these tables, it internally generates an internal SQL statement, which in turn generates a recursive call.
recursive cpu usage	Total CPU time used by non-user calls (recursive calls). Subtract this value from "CPU used by this session" to determine how much CPU time was used by the user calls.
redo blocks written	Total number of redo blocks written. This statistic divided by "redo writes" equals number of blocks per write.
redo buffer allocation retries	Total number of retries necessary to allocate space in the redo buffer. Retries are needed either because the redo writer has fallen behind or because an event such as a log switch is occurring.
redo entries	Number of times a redo entry is copied into the redo log buffer

redo log space requests	Number of times the active log file is full and Oracle must wait for disk space to be allocated for the redo log entries. Such space is created by performing a log switch. Log files that are small in relation to the size of the SGA or the commit rate of the work load can cause problems. When the log switch occurs, Oracle must ensure that all committed dirty buffers are written to disk before switching to a new log file. If you have a large SGA full of dirty buffers and small redo log files, a log switch must wait for DBWR to write dirty buffers to disk before continuing. Also examine the log file space and log file space switch wait events in V$SESSION_WAIT
redo log space wait time	Total elapsed waiting time for "redo log space requests" in 10s of milliseconds
redo log switch interrupts	Number of times that another instance asked this instance to advance to the next log file
redo ordering marks	Number of times that a system change number was allocated to force a redo record to have an higher SCN than a record generated in another thread using the same block
redo size	Total amount of redo generated in bytes
redo synch writes	Number of times a change being applied to the log buffer must be written out to disk due to a commit. The log buffer is a circular buffer that LGWR periodically flushes. Usually, redo that is generated and copied into the log buffer need not be flushed out to disk immediately.
redo synch time	Elapsed time of all "redo synch writes" calls in 10s of milliseconds
redo wastage	Number of bytes wasted because redo blocks needed to be written before they are completely full. Early writing may be needed to commit transactions, to be able to write a database buffer, or to switch logs.
redo write time	Total elapsed time of the write from the redo log buffer to the current redo log file in 10s of milliseconds
redo writer latching time	Elapsed time in 10s of milliseconds needed by LGWR to obtain and release each copy latch
redo writes	Total number of writes by LGWR to the redo log files. "redo blocks written" divided by this statistic equals the number of blocks per write
remote instance undo block writes	Number of times this instance wrote a rollback segment so that another instance could read it
remote instance undo header writes	Number of times this instance wrote a undo header block so that another instance could read it
rollback changes - undo records applied	Number of undo records applied to user-requested rollback changes (not consistent-read rollbacks)
rollbacks only - consistent read gets	Number of consistent gets that require only block rollbacks, no block cleanouts.
rows fetched via callback	Rows fetched via callback. Useful primarily for internal debugging purposes.
serializable aborts	Number of times a SQL statement in a serializable isolation level had to abort
session connect time	The connect time for the session in 10s of milliseconds. This value is useful only in V$SESSTAT. It is the wall clock time since the logon to this session occurred.

session cursor cache count	Total number of cursors cached. This statistic is incremented only if SESSION_CACHED_CURSORS > 0. This statistic is the most useful in V$SESSTAT. If the value for this statistic in V$SESSTAT is close to the setting of the SESSION_CACHED_CURSORS parameter, the value of the parameter should be increased.
session cursor cache hits	Number of hits in the session cursor cache. A hit means that the SQL statement did not have to be reparsed. Subtract this statistic from "parse count (total)" to determine the real number of parses that occurred.
session logical reads	The sum of "db block gets" plus "consistent gets". This includes logical reads of database blocks from either the buffer cache or process private memory.
session pga memory	Current PGA size for the session. Useful only in V$SESSTAT; it has no meaning in V$SYSSTAT.
session pga memory max	Peak PGA size for the session. Useful only in V$SESSTAT; it has no meaning in V$SYSSTAT.
session stored procedure space	Amount of memory this session is using for stored procedures
session uga memory	Current UGA size for the session. Useful only in V$SESSTAT; it has no meaning in V$SYSSTAT.
session uga memory max	Peak UGA size for a session. Useful only in V$SESSTAT; it has no meaning in V$SYSSTAT.
sorts (disk)	Number of sort operations that required at least one disk write Sorts that require I/O to disk are quite resource intensive. Try increasing the size of the initialization parameter SORT_AREA_SIZE. For more information, see "SORT_AREA_SIZE".
sorts (memory)	Number of sort operations that were performed completely in memory and did not require any disk writes You cannot do much better than memory sorts, except maybe no sorts at all. Sorting is usually caused by selection criteria specifications within table join SQL operations.
sorts (rows)	Total number of rows sorted
SQL*Net roundtrips to/from client	Total number of Net8 messages sent to and received from the client
SQL*Net roundtrips to/from dblink	Total number of Net8 messages sent over and received from a database link
summed dirty queue length	The sum of the dirty LRU queue length after every write request. Divide by write requests to get the average queue length after write completion.
switch current to new buffer	Number of times the CURRENT block moved to a different buffer, leaving a CR block in the original buffer
table fetch by rowid	Number of rows that are fetched using a ROWID (usually recovered from an index) This occurrence of table scans usually indicates either non-optimal queries or tables without indexes. Therefore, this statistic should increase as you optimize queries and provide indexes in the application.
table fetch continued row	Number of times a chained or migrated row is encountered during a fetch

	Retrieving rows that span more than one block increases the logical I/O by a factor that corresponds to the number of blocks than need to be accessed. Exporting and re-importing may eliminate this problem. Evaluate the settings for the storage parameters PCTFREE and PCTUSED. This problem cannot be fixed if rows are larger than database blocks (for example, if the LONG datatype is used and the rows are extremely large).
table scan blocks gotten	During scanning operations, each row is retrieved sequentially by Oracle. This statistic counts the number of blocks encountered during the scan. This statistic tells you the number of database blocks that you had to get from the buffer cache for the purpose of scanning. Compare this value with the value of "consistent gets" to determine how much of the consistent read activity can be attributed to scanning.
table scan rows gotten	Number of rows that are processed during scanning operations
table scans (cache partitions)	Number of range scans performed on tables that have the CACHE option enabled
table scans (direct read)	Number of table scans performed with direct read (bypassing the buffer cache)
table scans (long tables)	Long (or conversely short) tables can be defined as tables that do not meet the short table criteria as described in table scans (short tables)
table scans (rowid ranges)	During parallel query, the number of table scans conducted with specified ROWID ranges
table scans (short tables)	Long (or conversely short) tables can be defined by optimizer hints coming down into the row source access layer of Oracle. The table must have the CACHE option set.
total file opens	Total number of file opens performed by the instance. Each process needs a number of files (control file, log file, database file) in order to work against the database.
transaction lock background get time	Useful only for internal debugging purposes
transaction lock background gets	Useful only for internal debugging purposes
transaction lock foreground requests	Useful only for internal debugging purposes
transaction lock foreground wait time	Useful only for internal debugging purposes
transaction rollbacks	Number of transactions being successfully rolled back
transaction tables consistent read rollbacks	Number of times rollback segment headers are rolled back to create consistent read blocks
transaction tables consistent reads - undo records applied	Number of undo records applied to transaction tables that have been rolled back for consistent read purposes
Unnecessary process cleanup for SCN batching	Total number of times that the process cleanup was performed unnecessarily because the session or process did not get the next batched SCN. The next batched SCN went to another session instead.
user calls	Number of user calls such as login, parse, fetch, or execute When determining activity, the ratio of user calls to RPI calls, give you an indication of how much internal work gets generated as a result of the type of requests the user is sending to Oracle.
user commits	Number of user commits. When a user commits a transaction, the redo generated that reflects the changes made to database blocks must be written to disk. Commits often represent the closest thing to a user transaction rate.
user rollbacks	Number of times users manually issue the ROLLBACK statement or an error occurs during a user's transactions

write clones created in background	Number of times a background or foreground process clones a CURRENT buffer that is being written. The clone becomes the new, accessible CURRENT buffer, leaving the original buffer (now the clone) to complete writing.
write clones created in foreground	Number of times a background or foreground process clones a CURRENT buffer that is being written. The clone becomes the new, accessible CURRENT buffer, leaving the original buffer (now the clone) to complete writing.