Oracle 8i

BACKUP AND RECOVERY
Performance Tuning
Network Administration

Collection from Troytec books
By Ayman_tamim

BACKUP AND RECOVERY

Table of Contents

Performance Tuning

Table of Contents

Network Administration

Table of Contents

BACKUP AND RECOVERY CONSIDERATIONS

One of the DBA's major responsibilities is to keep the database available for use.

- Protect the database from numerous types of failures.
- Increase Mean-Time-Between-Failures (MTBF)
- Decrease Mean-Time-To-Recover (MTTR)
- Minimize data loss.

The strategy for backup and recovery depends upon business, technical, operational requirements and management concurrence.

The business requirements: Consists of Mean-Time-To-Recover (MTTR), Mean-Time-Between-Failures (MTBF) and evolving processes.

The operational requirements: Consists of Mean-Time-To-Recover (MTTR), Mean-Time-Between-Failures (MTBF), 24-hour operations and testing and validating backups.

The technical requirements: Physical image copies of the operating system files, logical copies of the objects in the database, database configurations and transaction volume affects desired frequency of backups.

Disaster Recovery Issues: Earthquake, flood, fire, complete loss of machine and loss of key professional.

ORACLE RECOVERY STRUCTURES AND PROCESSES

Oracle uses many memory components, background processes and file structures for backup and recovery mechanisms.

Memory Structures

Data Buffer Cache: Memory area used to store blocks read from data files.

Log Buffer: Memory containing before and after image copies changed data to be written to the redo logs.

Large Pool: An optional memory area used for IO by RMAN backup and restore.

Stored Pool: Stores parsed version of SQL Statement, PL/SQL procedures and data dictionary information.

Background Processes

Database Writer (DBW0): Writes dirty buffers from the data buffer cache to the data files.

Log Writer (LGWR): Writes data from the log buffer to the redo log files.

System Monitor (SMON): Performs automatic instance recovery. Recovers space in temporary segments when they are no longer in use.

Process Monitor (PMON): Cleans up the connection/server process dedicated to an abnormally terminated user process. Performs rollback and release the resources held by the failed process.

Checkpoint (CKPT): Synchronizes the headers of the data files and control files with the current redo log and checkpoint members.

Archiver (ARCH): An optional process that when enabled automatically copies redo logs that have been marked for archiving.

Oracle Database Physical Files

Data Files: Physical store of data. At least one file is required to store SYSTEM tablespace.

Redo Logs: Contain before and after image copies of changed data for recovery purposes. At least two groups are required.

Control Files: Record the physical structure and status of database.

Parameter File: Stores parameters required for instance startup.

Password File: Stores information on users who can start, stop and recover the database.

Archive Logs: Physical copies of the online redo log files created when the database is set in archivelog mode and used in recovery,

Standard Data Dictionary Views

V$SGA - Queries the size of the instance for shared pool, log buffer, data buffer cache and fixed memory sizes.

V$INSTANCE - Queries the status of the instance such as instance mode, instance name, startup time and host name.

V$PROCESS - Queries the background and server processes created for the instance.

V$DATABASE - Lists status and recovery information about the database.

V$DATAFILE - Lists the location and names of the data files that are contained in the database.

- **Large Pool -** used to allocate sequential IO buffers from shared memory. Recovery Manager RMAN uses the large pool for backup and restore when you set the BACKUP_DISK_IO_SLAVES parameter. If the LARGE_POOL_SIZE parameter is not set then there is no large pool and oracle attempts to allocate shared memory from the shared pool in SGA.
- **Archive Process -** an optional process, that when enabled, archives the redo log files to designated storage area. The arch process initiates when a log switch occurs and copies one member of the last redo log group to the destination specified by the init.ora parameter LOG_ARCHIVE_DEST.
- **CKPT Process -** ensures that all modified database buffers are written to the database files.
- **Data Files -** store both system and user data on the disk whether it is committed or uncommitted.
- **Redo log files -** store all changes made to database. If the database is to be recovered to a point in time when it was operational, redo log groups are used to make sure all committed transactions are committed to disk and all uncommitted transactions are rolled back. LGWR writes to redo log files in a circular fashion and there should be at least two redo log groups to store the cyclic nature. Redo logs always should be multiplexed.

ə **Archive Log Files -** When a database is set to ARCHIVE log mode, the LGWR waits for the online redo log files to be archived before they can be reused. A database backup combined with archived redo log files guarantees that all committed data be recovered to the point of failure and valid database backups can be taken while the database is online.

ə **The control file** - A small binary file that describes the structure of database.

ORACLE BACKUP AND RECOVERY CONFIGURATION

Redo Log History

All transactions are recorded in online redo log files.

ə If the database is configured for **Noarchivelog Mode,** no redo log history is saved to archived log files and recovery operations are limited and a loss of transaction may occur. The database by default is configured for Noarchivelog Mode. In this mode the log files are used in a circular fashion and once redo logs are overwritten, media recovery is only possible to the last full backup.

ə You can configure a database in **Archivelog Mode,** so that the history of redo information is maintained in archived files, allowing for a complete recovery up to the point of failure. A filled redo log file can not be reused until a checkpoint has taken place and the redo log files has been backed up by the ARCH background process. You can enable the ARCH process in an open instance through the initialization parameter. You can always stop archive process regardless of how you started it by using the ALTER SYSTEM command in server manager or by using Backup Manager.

ə **V$ARCHIVED_LOG -** Displays archived log information from control file

ə **V$ARCHIVE_DEST -** Describes for the current instance, all archive destinations, the current value, mode and status.

ə **V$LOG_HISTORY -** Contains log file information from the control file.

ə **V$DATABASE -** Current state of archiving.

PHYSICAL BACKUPS WITHOUT ORACLE RECOVERY MANAGER

The physical backup methods are:

ə Operating system backup without archiving used to recover to the point of the last backup after a media failure.

ə Operating system backup with archiving used to recover to the point of failure after a media failure.

CLOSED DATABASE BACKUP

A closed database backup is an operating system backup of all the data files, control files, parameter files and the password file that constitute an oracle database. It is conceptually simple, easy to perform, requires little operator interaction and is reliable.

OPEN DATABASE BACKUP

A DBA can perform backups of all the tablespaces or individual datafiles while the database is in use using the open database backup method. This method maintains high database availability, can be done at a tablespace or datafile level and supports non-stop business operations.

ə Information in the control file is required at instance startup time. It should be protected against loss.

TYPES OF FAILURES AND TROUBLESHOOTING

It can be:

ə Statement failure

ə User process failure

ə User error

ə Instance failure

ə Media failure

A **statement failure** occurs due to logic error in an application, an attempt to enter bad data into table, when attempting an operation with insufficient privileges and attempting to create a table but exceed allotted quota limits.

A **user process failure** occurs when a user performs an abnormal disconnect in the session, when the user was abnormally terminated and when the user's program raised an address exception terminating the session.

A **user error failure** occurs when a user accidentally drops or truncates a table, when the user deleted all rows in a table which are required and when a user committed data but discovered an error in the committed data.

An **instance failure** may occur due to power outage, when server is unavailable due to hardware problems and when oracle server background processes experienced a failure.

A **media failure** occurs when there is a head crash on a disk, physical problem in reading from or writing to database files and when the file was accidentally erased.

The alert.log is a file written by the oracle server that includes the status information for the instance and database. Oracle trace files are created by background processes and written by BACKGROUND_DUMP_DEST and are the source of additional diagnostic information.

COMPLETE RECOVERY OF ORACLE DATABASE

Noarchivelog mode can be suitable when:

ə Data loss between backups can be tolerated.

ə Recovery is faster by reapplying transactions.

ə Data rarely changes.

Archivelog mode should be preferred when:

ə Database cannot be shutdown for closed backup.

ə Data loss cannot be tolerated.

ə Easier to recover using archive logs than applying transactions.

RECOVERY WITH ARCHIVING(COMPLETE RECOVERY)

Its advantages are:

ə Only need to restore lost files.

ə Recovers all data to the time of failure.

ə Recovery time=Time to restore lost datafiles and apply all archived logs.

ə Its disadvantages are:

ə Must have all archived logs created since the backup from which you are restoring.

COMPLETE RECOVERY METHODS

ə Closed database recovery

ə Open database recovery

ə Open database recovery with database initially closed

ə Recover a datafile with no backup

DATA DICTIONARY VIEWS

V$RECOVER_FILE: to locate datafiles needing recovery.

V$LOG_HISTORY: for a list of all archived logs for the database.

V$RECOVER_LOG: for a list of all archived logs required for recovery.

INCOMPLETE ORACLE RECOVERY WITH ARCHIVING

ə Requires a valid offline or online backup of all database files.

ə Needs all archived logs since the backup until the specified time of recovery.

ə When server failures occur.

ə The reason for incomplete recovery are: due to user error, when complete recovery fails because an archived log is lost, and loss of all unarchived redo logs and the datafile.

TYPES OF INCOMPLETE RECOVERY

ə Time-based recovery

ə Cancel-based recovery

ə Recovery using a backup control file.

ə Change-based recovery

ORACLE EXPORT AND INPORT UTILITIES

These utilities provide the following:

ə Historical archive

ə Save table definitions

ə Move data between machines and databases

ə Provide user error failure protection

Table Mode Export: This mode exports specified tables in the user's schema, rather than all tables.
User Mode Export: This mode exports all objects for a user's schema. Privileged users can export all objects in the schema of a specified set of users.
Full Database Mode: This mode exports all database objects, except those in SYS schema.
Complete Export: If you use cumulative and incremental exports you should periodically perform a complete export to create a base backup.
Incremental Export: It includes objects that have changed since the last export of any kind.
Cumulative Export: It backups the tables that have changed since the last cumulative or complete export.

USES OF IMPORT UTILITY FOR RECOVERY

ə Create table definitions.
ə Extract data from a valid Export file.
ə Import from a complete, incremental or cumulative export file.
ə Recover from user-error failures.

ORACLE RECOVERY MANAGER

Recovery manager helps DBAs manage the process of backup, restore and recovery through a powerful, operating system independent scripting language.
FEATURES:
ə Backup the database, tablespaces, control files and archive logs.
ə Store frequently executed backup and recovery operations.
ə Perform incremental block level backup.
ə Compress unused blocks.
ə Specify limits for backups.
COMPONENTS
ə RMAN
ə Command line interface
ə Target Database
ə Recovery Catalog
ə Channel
MODES
ə A command line interpreter:
ə Interactive mode
ə Batch mode
ə A graphical user interface:

ə OEM
ə A library suitable for link-editing with other applications

ORACLE RECOVERY CATALOG MAINTENANCE

UPDATED RESYN RECORDS:

The target database must be mounted or open.
The updated recovery catalog records are:
ə Log switch
ə Archivelog copy
ə Backup history
ə Physical schema

Backup Set: A logical structure recording all backup pieces.
Backup piece: A physical file on a disk or tape, which stores one or more datafiles.
Datafile Copy: A physical copy of a datafile or control file stored on disk.
Archive Logs: Only created if archiving has been enabled for the database. These files are copies of redo log files.

"Change" Command

ə Only operates on backup set, backup pieces, datafile copy, archivelog and control files.
ə Marks file as UNAVAILABLE
ə Removes reference to files.
ə Determines which files in the recovery catalog physically exist.
ə Removes datafiles no longer needed.

"Catalog" Command

ə Archive logs, datafile copies and control file copies.
ə File copies not created by RMAN.
ə Only files with same database incarnation number.
ə Only files that belong to the databases.

"Run" Command

ə Compiles command into blocks of PL/SQL called steps.
ə Steps executed immediately from memory.
ə Can also be stored in a script.

PHYSICAL BACKUPS USING ORACLE RECOVERY MANAGER

Whole Backup: The backup of all data files and the control files.
Full Backup: The backup of one or more one or more files, which is not incremental.
Incremental Backup: The Backup of data changed since the last incremental backup.
Operating System Backup: The Backup performed using OS utilities.

TAGS: A tag is logical name assigned to a backup set or image copy.
BACKUP SETS: A backup set consists of one or more physical files stored in an oracle propriety format on either disk or tape.
BACKUP PIECE: A backup piece is a file in backup set. It may contain blocks from more than one datafile.
V$ARCHIVED_LOG: Shows which archives has been created, backed up and cleared in database.
V$BACKUP_DATAFILE: Useful for creating equal size backup sets by determining the number of blocks in each datafile.

RESTORE AND RECOVER USING RMAN

RMAN is used to perform:

- Restore and recovery of database in noarchivelog mode.
- Recovery of a datafile in open database.
- Recovery of a tablespace when there is a disk failure.
- Incomplete recovery.

ADDITIONAL ORACLE RECOVERY ISSUES

FASTER WARMSTART:

- Allows for a fast instance recovery.
- Database is available at the end of the roll forward phase of recovery.
- Individual user process will perform rollback when blocks are requested.

READ-ONLY TABLESPACE RECOVERY ISSUES

- Recreating a control file.
- Renaming data files.
- Using a backup control file.

Note:

- You can start the database and make it available for use if datafile are missing.
- Minimize downtime by determining the nature of tablespace essential or nonessential.
- Use parallel recovery operations to minimize recovery time.
- Prevent recoveries by having a good strategy for example mirrored control file.

Performance Tuning Key Concepts

Business Requirements and Tuning

Before beginning Oracle specific tuning you should check for bottlenecks on the operating system level. In particular, determine if there is excessive paging to the swap file and whether other applications are contending for system resources (such as RAM, CPU and disks) with Oracle.

Database Design

Many times you will not be able to participate at this level but when possible, being able to lay down a good design initially will help prevent a lot of performance nightmares from ever occurring.

Memory Tuning

Make sure the SGA is tuned correctly. The SGA is composed of three parts: 1) the data block buffer cache, 2) the redo log cache and 3) the shared pool. All three need to be tuned properly for optimal performance.

Disk I/O

Make sure your tablespaces are laid out correctly so that you don't have one type of tablespace adversely affecting another. An example of this is putting a rollback segment tablespace on the same disk as your data tablespace. Also verify you have storage parameters set optimally.

Internal Memory Structure Contention

Verify one internal process isn't waiting upon another for long periods of time. This is done by monitoring and adjusting latches correctly.

Oracle Alert, Trace Files, and Events

Trace files are Oracle log files which give the status of Oracle events. The main overall trace file which monitors the instance is called the alert.log. It is located in the directory specified by the BACKGROUND_DUMP_DEST initialization parameter. Background process (e.g., SMON, DBWR, etc.) trace files go here too.

Trace files (except for alert.log) usually end with a .trc extension. User process trace files go in the directory specified by the **USER_DUMP_DEST** initialization parameter. The initialization parameter **MAX_DUMP_FILE_SIZE** specifies the maximum size for user process trace files. In contrast, the alert.log trace file cannot have a specified size limit and must be truncated manually.

The V$ views are dynamic performance views based on the X$ tables. X$ tables are internal tables which hold information about the instance. Both are owned by SYS and populated at instance startup. V$ views can be seen by anyone with the "SELECT ANY TABLE" object privilege while X$ tables can only be viewed by SYS.
V$SYSTEM_EVENT, V$SESSION_EVENT, V$SESSION_WAIT - views which show information about processes waiting for resources.
V$SYSTEM_EVENT - shows total waits for events since the database started.
V$SESSION_EVENT - shows total waits for events since the database started grouped by session.
V$SESSION_WAIT - shows the most information about event waits per session. If the **WAIT_TIME** column of V$SESSION_WAIT has a value over 0 this indicates the session's last wait time. If the value is zero, the session is currently waiting. If the value is -1, the last wait was so small it wasn't measurable. If the value is -2, you need to enable **TIMED_STATISTICS** (see below) because Oracle isn't currently gather any time information.

Utilities and Dynamic Performance Views

Utilities

ULTBSTAT.SQL and UTLESTAT.SQL (B for begin and E for end) are scripts for collecting a variety of performance statistics. These scripts are located in $ORACLE_HOME/rdbms/admin. Before running UTLBSTAT.SQL and UTLESTAT.SQL you should set the initialization parameter **TIMED_STATISTICS** equal to TRUE. This will give CPU time statistics. Alternatively you can enable and disable **TIMED_STATISTICS** on the fly by issuing: "**ALTER SYSTEM SET TIMED_STATISTICS=TRUE**" (or FALSE).
Report.txt contains the following sections:
1. Library cache statistics.
2. Numbers on a variety of statistics for the duration of the time UTLBSTAT.SQL was gathering data.
3. System wide wait events.
4. Average length of the dirty buffer write queue.
5. I/O statistics by datafile and tablespace.
6. Latch statistics.
7. Rollback segment statistics.
8. The current init<sid>.ora parameters.
9. Row (dictionary) cache statistics.

There is a set of GUI applications that come with the Oracle Enterprise Edition version of Oracle Enterprise Manager called the Oracle Performance Pack. These programs help monitor and optimize database performance:
Oracle Expert - assists in configuring and tuning your database (see below section for more detail).
Lock Manager - monitors locks.

Performance Manager - lets you monitor real time performance statistics and view them in various ways.
Tablespace Manager - lets you monitor segment storage in tablespaces. Also lets you defragment tablespaces (know as coalescing).
TopSessions - like the Unix 'top' command, lets you monitor the top resource intensive user sessions.
Oracle Trace - lets you monitor performance within user applications.

Performance Views

V$SYSSTAT is the main view for system performance on the instance level. You can also use V$SYSSTAT to monitor client-server traffic.
V$SESSION gives connection information for all user sessions.
Performance Ratios

ə V$LIBRARYCACHE gives the pins/reload ratio for the library cache. The GETHITRATIO column should be .95 or higher. The RELOADS column should be equal to or less than one percent of the PINS column.
ə The GETMISSES to GETS columns in V$ROWCACHE should have a ratio less than 15% percent.
ə The buffer cache hit ratio should be 90% or higher.
ə The hit ratio for a latch should be .98 or higher. Anything lower indicates latch contention.
ə If in V$LATCH the MISSES to GETS ratio is higher than 1%, there is a problem with redo allocation latch contention.
ə If in V$LATCH the IMMEDIATE_MISSES to IMMEDIATE_GETS ratio is higher than 1%, there is a problem with redo copy latch contention.
ə Query V$WAITSTAT to see if there is rollback segment header block contention. A value over zero in the UNDO HEADER column (undo = rollback) indicates contention.
ə Query V$SYSTAT to see whether sorts are occurring on disk or in memory. For OLTP systems 95% of sorts should be happening in memory.

Tuning Considerations For Different Applications

Know the difference between OLTP, DSS and hybrid systems. OLTP stands for online transaction processing system and is characterized by a high volume of inserts, updates and deletes (DML). DSS stands for decision support system. Is also commonly known as a data warehouse or OLAP: online analytical processing system. DSS is a read-only database consisting of massive amounts of data.
A *hybrid* system would be one that is both an OLTP database and a DSS database but at different times. An example of a hybrid database is a purchase order database that is used for querying during the daytime. But at night, new orders are inserted, current ones are modified, and fulfilled orders are deleted.

SQL Tuning

There are two optimizers built into Oracle: Rules Based and Cost Based. The Rules Based Optimizer (RBO) is a set of 15 rules Oracle uses to determine the fastest path of execution for a given SQL statement. The Cost Based Optimizer (CBO) determines all possible paths of execution and assigns a cost to each one. It chooses what it finds to be the least expensive path. You must be running ANALYZE on your tables and indexes to generate statistics to use the Cost Based Optimizer.

It is very important to update statistics on a regular basis to provide meaningful data for the CBO work with. Performance can suffer tremendously by using outdated statistics. You can set the CBO goal to either; 1) fastest overall throughput for a SQL statement or, 2) fastest initial response time for a SQL statement.

You can set the optimizer at the session level by issuing the command:

ALTER SESSION SET OPTIMIZER GOAL=*optimizer mode*

Use EXPLAIN PLAN FOR *sql statement* to see how the optimizer is executing your statement. This puts the execution plan in the PLAN_TABLE.

SQL Trace

SQL Trace is used to gather user session statistics. It generates a trace file which is generated in **USER_DUMP_DEST**. It contains the same data as AUTOTRACE with the following additional information:

- parse, fetch and execute counts
- cpu and elapsed time
- number of logical and physical reads
- number of rows processed
- library cache misses

SQL Trace can be turned on at the instance level by setting the initialization parameter SQL_TRACE=TRUE. SQL Trace can also be turned on for an individual session by issuing:

ALTER SESSION SET SQL_TRACE=TRUE;

You can also turn SQL Trace on in the current session by executing the procedure SYS.DBMS_SESSION.SET_SQL_TRACE(TRUE).

Finally, you can turn on SQL Trace for another session by executing the procedure:

SYS.DBMS_SESSION.SET_SQL_TRACE_IN_SESSION(*sid*,*serial#*,TRUE).

TKPROF is used to read the output of a trace file created by SQL Trace. The DMBS_APPLICATION_INFO package is created by the DBMSUTIL.SQL. It allows you to track resource usage and performance data for PL/SQL procedures. DBMSUTIL.SQL is called by the CATPROC.SQL script so it should already exist in your database.
Each lookup table has a *primary key* to *foreign key* relationship to the fact table (a child-parent relationship). However, in a star schema there is no relation between the various lookup tables among themselves. Hence, the "star" formation.
A *hash join* is a special type of equijoin that works well with joins between a large and small table. The initialization parameter **HASH_JOINED_ENABLED** controls whether hash joins are allowed. The default value is TRUE. This can also be changed at the session level via ALTER SESSION or at the statement level via the /*+ USE_HASH (*tablename*) */ hint.

Tuning the Shared Pool

The shared pool is composed of the library cache, row cache and if MTS is used, the User Global Area (UGA). The size of the shared pool is set by the initialization parameter **SHARED_POOL_SIZE** (in bytes).
The library cache contains shared parse information for SQL statements. This is the chief area to monitor in the shared pool.
V$LIBRARYCACHE - gives the pins to reload ratio for the library cache. The GETHITRATIO column should be .95 or higher. In V$LIBRARYCACHE, the value for RELOADS should be equal to or less than one percent of the value of PINS. If it is higher, you should increase the size of the shared pool.
V$SQLAREA - gives information about shared cursors and the first part of the SQL statements they contain.
V$DB_OBJECT_CACHE - gives information about PL/SQL packages and views. The SHAREABLE_MEM column specifies how much memory of the library cache an object is consuming.
V$SQLTEXT - gives the full SQL statements contained in shared cursors.
V$DB_OBJECT_CACHE - gives detailed information about cached tables, rows and PL/SQL objects.
The row (dictionary) cache stores date dictionary information.
V$ROWCACHE - view gives the hit/miss ratio for the dictionary cache. The GETMISSES to GETS columns in V$ROWCACHE should have a ratio of less than 15% percent. This ratio is expressed in report.txt as GET_MISS to GET_REQS.
You can pin packages, procedures, triggers and cursors into the shared pool using **DBMS_SHARED_POOL.KEEP( )**. The **DBMS_SHARED_POOL** package is created by running DBMSPOOL.SQL and PRVTPOOL.SQL.

The initialization parameter **SHARED_POOL_RESERVED_SIZE** specifies how much of the shared pool you want to set aside for the reserved list. This is an area of the library cache to store large objects in. Objects smaller that the value specified by the initialization parameter **SHARED_POOL_RESERVED_MIN_ALLOC** will not be allowed on the reserved list.
The User Global Area (UGA)is an additional third area of the shared pool when Oracle runs in Multithreaded Server (MTS) mode. When running in MTS mode, user information stored in the PGA in dedicated server mode is stored in the UGA instead. Hence, the UGA needs to be taken into consideration for overall sizing of the shared pool. Specifically you need to calculate the additional amount of memory required by the shared server sessions and open cursors.

Tuning the Buffer Cache

The size of the buffer cache is determined by: DB_BLOCK_SIZE * DB_BLOCK_BUFFERS. DB_BLOCK_SIZE cannot be changed without recreating the database. The buffer cache contains the dirty buffer write queue, which holds dirty block buffers (i.e., modified blocks) until they can be written out to disk. A least recently used (LRU) algorithm is used to decide which buffers to move out of the buffer cache so that new buffers can be read in.
DBWR is the background process that writes the dirty block buffers (located in the dirty buffer write queue) to disk. Server processes (S000..S999) read data blocks from data files into the buffer cache. Server processes check first to determine if the data blocks already exist in the buffer cache.
V$SYSSTAT - view that shows the *buffer cache hit ratio*. The buffer cache hit ratio should be 90% or higher.
V$RECENT_BUCKET - view that shows the effects of increasing the buffer cache by x amount of blocks. The value in the ROWNUM column shows the number of proposed block additions minus one (it starts with zero). The value in the COUNT column shows how many more buffer cache hits you will get by increasing the buffer cache by the proposed amount of blocks. To use V$RECENT_BUCKET, stop the database and set the **DB_BLOCK_LRU_EXTENDED_STATISTICS** initialization parameter to the maximum number of blocks you are considering adding. Next, start the database. After the instance has been running for a while under normal conditions query V$RECENT_BUCKET.
V$CURRENT_BUCKET - shows the effect of decreasing the buffer cache. The value in the ROWNUM column shows the number of proposed block subtractions. The value in COUNT column shows how many buffer cache hits you would still have after this decrease. To use V$CURRENT_BUCKET, stop the database and set the **DB_BLOCK_LRU_STATISTICS** initialization parameter to TRUE. Next, start the database. After the instance has been running for a while under normal conditions query V$CURRENT_BUCKET.
Gathering statistics for V$RECENT_BUCKET and V$CURRENT_BUCKET consumes CPU cycles and hence is a performance hit to the database. Therefore, you should disable these views by resetting their respective initialization parameters to the default value (i.e., 0 and FALSE) as soon as you are finished using them.

You should increase the buffer cache if:
1) The cache hit ratio is too low (below 90%).
2) There are increasing values for free buffer inspected (check V$SYSSTAT or report.txt).
3) There is contention for the cache buffer's chain latch (check V$LATCH).
4) There are buffer busy waits (check V$SYSTEM_EVENT or V$SESSION_WAIT).
On multi-CPU systems, **DB_BLOCK_LRU_LATCHES** specifies how many LRU latches (buffer chain latches) are available for the database. The default value is one half the number of CPUs and can be increased to double the number of CPUs. On single CPU systems you can only have one LRU latch.
A new feature in Oracle 8 is optional *multiple buffer pools* within the buffer cache. Using these buffer pools lets you explicitly define which objects you want to remain in the buffer cache and which you want to quickly unload. There are three buffer pools:
э Keep Buffer Pool: For objects you want to remain in the buffer cache as long as possible.
э Recycle Buffer Pool: For objects you want unloaded from the buffer cache as soon as the transaction that uses them is complete.
э Default Buffer Pool: The default buffer pool that DB blocks are stored in if not otherwise specified. Applications based on Oracle 7.3 and earlier will always use the default buffer pool. By default, the keep and recycle buffer pools are not enabled. To specify space for them, you must define the initialization parameters **BUFFER_POOL_KEEP** and **BUFFER_POOL_RECYCLE**. You must also specify how many buffer chain latches you want allocated for each pool. You must have at least 50 buffer blocks for every LRU latch you assign.
Syntax: BUFFER_POOL_KEEP(buffers:50000, lru_latches:5)
The size of the default buffer pool is not explicitly defined. Instead it is equal to the size of the entire buffer cache (value of **DB_BLOCK_BUFFERS**) minus the number of blocks allocated to the keep buffer pool and/or the recycle buffer pool.
The buffer blocks for the keep and recycle buffer pools are taken from the buffer cache. Likewise, their LRU latches are taken from the latches allocated for the entire buffer cache as specified by **DB_BLOCK_LRU_LATCHES**. Therefore, neither the number of buffer blocks nor the number of LRU latches allocated to the keep and recycle buffer pools can, taken together, equal or exceed the values allocated to the buffer cache as a whole. If you do over-allocate either DB blocks or LRU latches by accident, the database will not mount.
Use the CACHE hint (/*+ CACHE */) in your SQL statements to ensure that the selected table will be cached. You can also use the CACHE *clause* when creating a table to make sure it will always be cached.

Only cache small, frequently used tables. By default, large tables will fill the buffer cache (unless assigned to the keep or recycle buffer pool).
V$CACHE shows what objects are currently in the buffer cache. Run CATPARR.SQL to create this view. While CATPARR.SQL is for Parallel Server environments, V$CACHE is useful in single instance environments as well. The initialization parameter
CACHE_SIZE_THRESHOLD specifies how many blocks per table will be cached. This is helpful to prevent buffer cache crowding.
Latch and Contention Issues
What locks are to tables, latches are to internal memory structures. Latches regulate access to instance resources by system and user processes. The V$LATCH view gives real time detailed latch information. The HIT_RATIO column gives the hit ratio for each latch. The SLEEPS column indicates how many times a process has had to wait to obtain a latch. Latch statistics are also listed in report.txt. The hit ratio for a latch should be .98 or higher. Anything lower indicates latch contention.
Tuning the Redo Log Buffer
The LOG_BUFFER initialization parameter specifies how big (in bytes) the redo log buffer is. Query V$SYSTEM_EVENT to see if there are waits for ‘log buffer space’. If so, increase the redo log buffer size. You can also obtain this information from V$SYSSTAT by looking at the number of 'redo buffer allocation retries' there are for a given user process.
Query V$SYSTEM_EVENT to determine if there are waits for ‘log file parallel write’. If so, this means there is I/O contention for redo log files. Stripe the redo log files across several disks to alleviate this problem.
You can forego creating redo log information during ALTER and CREATE statements as well as SQL*Loader direct-path loads by specifying the NOLOGGING clause. This greatly speeds up creation of large tables and indexes. NOLOGGING replaces the UNRECOVERABLE keyword used in Oracle 7.3.
There are two types of latches associated with the redo log buffer. The first is the *redo allocation latch*. The second type of latch is the *redo copy latch*. This latch exists only in multi-CPU systems.
The redo copy latch is used to break up the allocation and writing processes into two steps. The V$LATCH view gives statistics about redo allocation latch and redo copy latch contention. For redo allocation latch requests (WILLING_TO_WAIT type) the columns to monitor are GETS, MISSES and SLEEPS. For redo copy latch requests (IMMEDIATE type) the columns to monitor are IMMEDIATE_GETS and IMMEDIATE_MISSES.

If the MISSES to GETS ratio is higher than 1%, there is a problem with redo allocation latch contention. If the IMMEDIATE_MISSES to IMMEDIATE_GETS ratio is higher than 1%, there is a problem with redo copy latch contention.

Database Configuration and I/O Issues

Set the **LOG_CHECKPOINTS_TO_ALERT** initialization parameter to TRUE so that checkpoint beginning and ending times are logged to the alert.log file.

The **DB_BLOCK_CHECKPOINT_BATCH** specifies the maximum number of blocks that a DBWR process can write in a single batch during a checkpoint. Increasing this value can speed up checkpoint times, but making it too large can give poor response times as well.

The initialization parameters **DISK_ASYNCH_IO** and **TAPE_ASYNCH_IO** specify whether the operating system supports asynchronous I/O for hard drives and tape drives respectively (most do). The default value is TRUE.

If your operating system doesn't support asynchronous disk and/or tape I/O, you can enable them on the Oracle level by specifying the following initialization parameters:

- **ARCH_IO_SLAVES**
- **LGWR_IO_SLAVES**
- **DBWR_IO_SLAVES**
- **BACKUP_DISK_IO_SLAVES**
- **BACKUP_TAPE_IO_SLAVES**

The SYSTEM tablespace should only contain data dictionary objects, PL/SQL packages, triggers and the initial rollback segment. Your database should have a minimum of six tablespaces:

SYSTEM - for the data dictionary and initial rollback segment.
DATA - for tables.
INDEX - for indexes.
TEMP - for sorting.
RBS - for rollback segments.
USER - for user created tables.

V$FILESTAT tells the number of reads and writes done to each data file. Use this view and report.txt to determine if I/O is evenly distributed or not.

Avoid heavy I/O on the SYSTEM tablespace. Heavy I/O may occur by not specifying the default and temporary tablespace when creating users. If so, users object and query sorts will be written and read from SYSTEM.

If the row cache isn’t big enough to allow the entire data dictionary to exist in memory, additional I/O will occur in the SYSTEM tablespace due to data dictionary reads.

When using hardware or software striping, make sure the stripe block size is a multiple of the **DB_FILE_MULTIBLOCK_READ_COUNT** initialization parameter. Manually stripe tablespaces only if hardware or the operating system doesn't support it. You should only consider it if you are in an OLAP environment where a lot of full table scans are happening and you are using Parallel Query.
You can manually stripe by spreading the tablespace over two data files, one on a different drive. Set MINEXTENTS greater than 1 and then create the table so that each extent fills up more than half of each data file. You can also allocate extents to data files explicitly.

Using Oracle Blocks Efficiently

PCTUSED is relevant for deletes. If you have a lot of inserts and deletes on a table, set PCTUSED high. On DSS systems PCTUSED is irrelevant. PCTFREE and PCTUSED taken together should be less than 100.
Migrated and chained rows cause excessive disk I/O. So does inefficient use of block space which causes more blocks to be read than is necessary. The goal is to determine how your tables are being used and set the block storage parameters optimally, neither overcrowding nor wasting space.
The CHAIN_CNT column of **DBA_TABLES** tells you how many chained or migrated rows a table has. The CHAINED_ROWS table gives detailed information about each chained or migrated rows in a table. This table is created by the UTLCHAIN.SQL script. Use the LIST CHAINED ROWS clause of the ANALYZE command to populate the CHAINED_ROWS table.
DB_BLOCK_SIZE should be a multiple of the OS block size. Make it as large as possible.
Set **DB_FILE_MULTIBLOCK_READ_COUNT** to a multiple of **DB_BLOCK_SIZE**. This determines how many blocks are read at once in a full table scan. This is very important for DSS systems.
The *highwater* mark for a table is increased 5 blocks at a time. Full table scans read up through the highwater mark. DELETE does not reduce the highwater mark count on a table , but TRUNCATE does. So does ALTER TABLE *tablename* DEALLOCATE UNUSED.

Optimize Sort Operations

The following clauses cause sorting to occur: ORDER BY, GROUP BY, DISTINCT, UNION, INTERSECT and MINUS. Oracle will sort in memory if the sort can fit in the value specified by the initialization parameter **SORT_AREA_SIZE** (in bytes). If not, Oracle will break the sort into multiple *sort runs*. MAXEXTENTS is not a valid storage parameter for temporary tablespaces.
When a process needs to sort on disk it looks in the sort extent pool (SEP) in the SGA to get a free extent. V$SORT_SEGMENT gives detailed information regarding sort extents. The EXTENT_HITS column tells how many times a free segment was found in the sort extent pool.
Query V$SYSTAT to see whether sorts are occurring on disk or in memory. At least 95% of sorts should be happening in memory for OLTP systems.

The initialization parameters **SORT_WRITE_BUFFERS** and
SORT_WRITE_BUFFER_SIZE specify how much memory to allocate for direct write sorts.
The **SORT_WRITE_BUFFERS** should be between 2 and 8.
SORT_WRITE_BUFFER_SIZE should be between 32K and 64K.

Rollback Segment Tuning

Rollback segments are used for the three Rs: transaction rollback, instance recovery and query read consistency. There is a *Rule of 4* for sizing the number of rollback segments: # of concurrent transactions divided by 4. If less than 4+4, round up to a multiple of 4. No more than 50 total.
If you get an "ORA-01555**:** snapshot too old (rollback segment too small**)**" error, this means a rollback segment is too small for a given transaction to complete and that it has overwritten itself. You should increase the number of rollback segments or assign the transaction a large rollback segment.
Use "SET TRANSACTION USE ROLLBACK SEGMENT *rollback segment name*" to assign a specific rollback segment to a transaction. This should be done for long running batch jobs.
MINEXTENTS for rollback segments should be 20. INITIAL for rollback segments should be set to the power of 2. OPTIMAL should be set high enough to accommodate normal database activity.
DELETEs use the most rollback space. Next UPDATEs, then INSERTs.
V$SESSION and V$TRANSACTION show how much rollback information a transaction is creating.
The rollback transaction table is in the rollback segment header. Query V$WAITSTAT to see if there is rollback segment header block contention. A value over zero in the UNDO HEADER column (undo = rollback) indicates contention.

Monitoring and Detecting Lock Contention

Enqueues are what Oracle uses to manage locks. DML locks occur at the row level. The V$LOCK view gives detailed lock information.
A query does not hold a lock unless the SELECT FOR UPDATE statement is issued. The SELECT FOR UPDATE statements give the user a Row Shared (RS) lock. This gives a shared table lock and an exclusive row lock. A Share lock on a table prevents DML operations happening to it. A Share Row Exclusive lock prevents DML and SELECT...FOR UPDATE on a table.
Query V$SESSION to see which row is causing lock contention. DDL locks are not usually in contention because they are held only briefly.

The three types of locks mainly in use by applications are: TM (row exclusive table lock), TX (row exclusive lock) and UL (PL/SQL user locks). High level locking (i.e., locking a whole table instead of just a row) in application code can cause the majority of lock contention in a database. This may be because the application was coded for another rdbms such as Sybase or SQL Server.
To kill a user session which is holding a lock, query V$SESSION and get the SERIAL# and SID (session ID). Next, issue ALTER SYSTEM KILL SESSION *serial #, sid*.
UTLLOCKT.SQL shows a graph of which locks are being held. CATBLOCK.SQL must be run first to set up the views for this.
When a deadlock is detected, Oracle will roll back the statement that caused the deadlock and issue an ORA-00060 error message. Oracle does not rollback the entire transaction, only the statement that detected the deadlock. You will have to manually rollback the rest of the transaction if necessary. When a deadlock occurs the rowid of the locking row is recorded in a trace file located in **USER_DUMP_DEST**.

Tuning with Oracle Expert

Oracle Expert uses the concept of a *tuning session* to focus in on a given portion of the database you want to examine. All data is collected and stored on the tuning session level. These sessions are saved to disk with an .XDL extension. There are three scopes for tuning sessions:
1) *Application level* - examines SQL statements and access methods (index usage).
2) *Instance level* - examines SGA, operating system and I/O parameters.
3) *Structure level* - examines storage parameters and placement of tablespaces.
Oracle Expert has seven different classes of input data:
1) *Database* - users, tablespaces, public synonyms.
2) *Instance* - data gathered from the X$ tables and V$ views.
3) *Schema* - tables, indexes, sequences, views, etc.
4) *Environment* - physical server attributes (RAM, CPU, disks).
5) *Workload* - what normal transactions are performed on the database. Can be gathered by Oracle Trace.
6) *Rules* - guidelines to tell Oracle how to treat collected data. Change rules if you don't like a given recommendation.
7) *Control parameters* - specifies whether the database is an OLTP, DSS, or hybrid, so Oracle Expert can set goals and use specific features accordingly.
Oracle Expert generates two different types of output:
1) *Reports* - includes a) Analysis, b) Session Data and c) Recommendation Summary.
2) *Implementation Files* - includes proposed changes to the init<sid>.ora file and SQL scripts to put into effect recommended changes (e.g. index creation).

Oracle 8: Network Administration Concepts

Types of Networks

There are 3 basic network types. Simple or 2-tier, Simple/Complex or N-tier, and Complex.

SIMPLE (2-TIER) NETWORK

In two-tier network, a client communicates directly with a server. This is also called client/ server architecture. The client and server communicate over the network by using a given protocol and it must be installed on both the client and the server. This configuration works best with only small networks.

SIMPLE/COMPLEX (N-TIER) NETWORK

In n-tier networks, one or more agents are introduced between the client and server. This agent can provide:

- Translation Services
- Scalability Services
- Intelligent Agent Services

This middle tier contains applications and services. The Server holds the actual data.

COMPLEX NETWORK

In a complex network you have different hardware platforms, multiple protocols, and different geographical locations.

Net8 Overview

Oracle Net8 offers comprehensive platform support. This includes:

- Protocol independence
- Integrated GUI administration tools
- Multiple configuration options
- Tracing and diagnostic toolset
- Basic security

ORACLE NAMES

Oracle Names is a tool of Net8 that stores the database addresses for entire environment. This tool provides centralized configuration, simplified network administration, and client profile information.

CONNECTION MANAGER

Connection Manager is a tool of Net8 which is normally installed and configured on middle tier. It provides multiplexing of connections, cross protocol connectivity, and network access control.

ADVANCED NETWORKING OPTION

To ensure secure transmission, advanced networking option must be installed on both client and server. This offers network security options such as client side encryption, and encrypted

data plus checksumming. At the server data is decrypted along with checksumming if proper "key" is installed on the server.

SECURITY SERVER

Security Server supports authorization and authentication using cryptographic checksumming in an Oracle environment. Also provides for Single Sign-On so users who may need access to multiple databases no longer are issued different username/password pairs for each server.

OPEN GATEWAY

The Oracle open gateway extends the capability of Oracle server tier to non-Oracle environment. It provides transparent and procedural gateways.

NET8 STACK

In an Oracle client-server transaction, information passes through the following six client side layers:

- Client Application
- Oracle Call Interface (OCI)
- Two Task Common (TTS)
- Transparent Network Substrate (TNS)
- Oracle Protocol Adapters (OPA)
- Network Specific Protocols

Client Side Layer Stack

APPLICATION LAYER

It provides all user oriented activities such as character or graphical user interface (GUI) display, screen control, data presentation, application flow and other application specifies.

ORACLE CALL INTERFACE (OCI) LAYER

It defines server calls, parses SQL statements, opens cursors, binds variables, executes SQL, fetches data and closes cursors. It also initiates a SQL dialogue between the client and server.

TWO TASK COMMON (TTS) LAYER

Converts i-e data types and character sets.

TRANSPARENT NETWORK SUBSTRATE (TNS) LAYER

This layer provides a generic interface to industry standard protocols. It consists of more Net8 specific layers:

- Network Interface (NI) Layer - It provides a generic interface to access Net8 functions, resolves connect descriptors to connect strings and handles break and reset requests.
- Network Routing (NR) Layer - It routes information from client to destination.
- Network Naming (NN) Layer - It resolves network aliases to destination address.
- Network Authentication (NA) Layer - It negotiates authentication with destination.
- Network Session (NS) Layer - It handles connect handshake and negotiation, buffer management and handles connection pooling and multiplexing.

NETWORK TRANSPORT (NT) LAYER
It maps TNS generic calls to protocol-specific calls and includes NT layers that handle transport.
ORACLE PROTOCOL ADAPTER (OPA) LAYER
It consists of three sub-layers, NT Main, NT(2) and NT OS.
NETWORK SPECIFIC PROTOCOLS
All Oracle software in the client-server connection process requires an existing network protocol stack to make the machine level connection between the two machines.

Server Side Layer Stack

ORACLE PROGRAM INTERFACE (OPI) LAYER
It responds to each OCI call made by the client side.
TTS, TNS, OPA and PROTOCOL LAYERS
All perform the same functions as on the client side.
SERVER LAYER
It's responsible for receiving OCI calls from the client and resolving SQL statements on behalf of the client.

Network Program Interface (NPI)

NPI is a server specific version of OCI and it enables coordinating servers to construct SQL requests for other servers.
SERVER-SIDE CONFIGURATION
The Listener Process:
ə It can listen for more than one database.
ə Performs load balancing
ə Listens on Multiple Protocols
ə Default name of the listener is LISTENER.
The Listener Responses
When a request is made by a client to the server, the listener will perform one of the following:
Bequeath Session
When the listener spawns a dedicated server process and bequeaths (passes) the connection to a dedicated server process, the session is called a bequeath session and the sequence of events is as follows:
ə A client connects to the listener with the network address.
ə The listener receives the session request and determines if the request can be serviced.

ə If possible the listener spawns a new dedicate server process to serve the incoming session and bequeaths the session to that server process.
ə Once the session is established data flows directly between the client and dedicated server process.

Redirect Session (Prespawned Dedicated Server Process)

The connection time from client to server is faster than the bequeath method. The following occurs:
ə The listener spawns a series of dedicated server processes when it is started, so when a client connects to the listener with a network address listener determines if the request can be processed or not.
ə If the connection is possible listener issues a redirect message to the client containing the network address of one of the prespawned servers.
ə The client disconnects from the listener and connects to the prespawned address given by the listener.
ə Once the session is established, the server listener spawns another server to replace the one used by the client.

Note: The maximum number of server processes are defined by the parameter PRESPAWN_ MAX in listener.ora file.

Redirect Session (Dispatcher Server Process)

This occurs when the Oracle server is configured as a multithreaded server. In the multithreaded server the listener will redirect the incoming request to the dispatcher.
ə When a database instance is started, dispatchers are started according to the configuration parameters in the init.ora file.
ə The address of each dispatcher is then registered with the listener and the client connects to the listener with the network address.
ə If the received request can be serviced the listener issues a redirect message to client containing the network address of least used dispatcher for the multithreaded server.
ə The client disconnects from the listener and connects to the dispatcher address given by the listener and the dispatcher updates the listener with new load value.

The LISTENER.ORA File

When a listener is started using the Listener Control Utility (LSNRCTL), it creates a listener. ora file with following default settings:
ə listener name LISTENER
ə Port 1521
ə Protocols TCP/IP and IPC
ə SID name Default database
ə Hostname Default hostname

Listener Control Utility (LSNRCTL)

A tool used to control the listener. Commands from the listener Control utility can be issued from the command line or from the LSNRCTL prompt.

LSNRCTL COMMANDS:

DBSNMP_START Starts the SNMP Intelligent Agent for an Oracle database running on the same node.
DBSNMP_STOP Stops SNMP agent for an Oracle database.
RELOAD Shuts down everything except listener addresses and re-reads listener.ora. It enables you to add or change services without actually stopping the listener.
SAVE_CONFIG Creates a backup of listener configuration file listener.bak and updates listener.ora to reflect any changes.
SERVICES Provides a detailed information about the services the listener listens for.

CLIENT-SIDE CONFIGURATION

Host Naming Method

In using the host naming method you must have installed TCP/IP, Net8 and TCP/IP protocol adapter. The host-naming method requires minimal user configuration and requires the file sqlnet.ora.

Local Naming Method

Requires the configuration of a file named tnsnames.ora through Net8 Assistant.

Service Name

A service name is a short, convenient name that is mapped to a network address contained in the network descriptor which is stored in the tnsnames.ora file. Users only need to know the appropriate service name and not the full connect descriptor.

Test Service

It is used to test the simple network when it's configured using the Net8 Assistant.

Oracle Names Usage and Configuration

RESOLVING SERVICE NAMES

Connection strings are resolved using one of the methods below.

Local Naming

Uses tnsnames.ora to lookup service names.

Host Naming

The actual name of the machine on which the database resides is used to establish a connection to the service.

Centralized Naming

This method uses Oracle names, which is a names service that maintains central storage of network service addresses. The Names server has the benefit of simplified administration tasks, increased efficiency, eliminates redundancy and facilitates location transparency.

Note: Names server resolves the client request to connect to a service by translating the service name to a network address.

To use centralized naming, the Names server and the client attempting to use the Names server must be configured. The Names server can be configured using the Net8 Assistant or by manually editing the names.ora file. The names client profile can be configured using the Net8 Assistant or by manually editing the sqlnet.ora file.

NAMES SERVER

Names.ora is the configuration file for Names server and sqlnet.ora is that for client. The Names server resides in the root region. A Names server can be controlled by using Net8 Assistant, control panel and names control utility. It can be configured either manually by editing names.ora or by using Net8 Assistant.

Client Side Cache

It saves the time a client would normally take to re-query the Names server. A region database is database repository of service names stored by the Names server.

NAMES CONTROL UTILITY (NAMESCTL)

ə To test if the Names server is up and running, the PING command is used.

ə To view the location of names.ora file, trace file, etc. issue the NAMESCTL>STATUS command.

Oracle Intelligent Agent for Oracle Enterprise Manager

ORACLE INTELLIGENT AGENT (OEM)

OEM is a graphical application that combines a number of services to provide a comprehensive system management platform.

THE INTELLIGENT AGENT

An Intelligent Agent is a smart process running on a remote node (server) in the network. OEM on a client node communicates with Intelligent Agents to run jobs and monitor events on remote sites. The Intelligent Agent is responsible for accepting jobs and events, canceling jobs or events, running jobs, collecting results and queuing jobs, checking events and queuing the resulting event reports, returning jobs and event reports, handling SNMP requests and discovering services on nodes where agents reside.

LISTENER CONTROL UTILITY

Listener Control Utility is the tool used to start, stop and view the status of the Intelligent Agent.

A few commands are:

LSNRCTL> dbsnmp_start Used to start the Intelligent Agent.

LSNRCTL> dbsnmp_stop Used to stop the Intelligent Agent.

LSNRCTL> dbsnmp_status Used to view the status of Intelligent Agent.

NOTE*: The Intelligent Agent creates three configuration files. The main configuration file is snmp_ro.ora while the other two are snmp_rw.ora (second configuration file) and serv-*

ices.ora. The services.ora file contains the services discovered by the listener. A user must be created for the agents to manage jobs and events.

Repository - A set of tables in your schema that can be placed in any Oracle database in your system. A repository is created by a Repository Manager when Enterprise Manager attempts to connect for the first time to a server running an Intelligent Agent.

Multithreaded Server

SINGLE-TASK ARCHITECTURE

The user and the server processes are combined in a single user process. When one process executes both the application code and Oracle code, this configuration is called single-task Oracle.

TWO-TASK ARCHITECTURE

A dedicated server process handles requests for a single user process. If the user and server processes are separate, the term two-task is used. In two-task Oracle the servers can be shared or dedicated and the database application code and Oracle code are executed by different processes.
In Dedicate Server Processes (Shadow Server Process) the user and server processes are separate, each user has its own server process, the user and server processes can run on different machines, there is one-to-one ratio between user and server processes and even when user process is not making a database request the dedicated server exists but remains idle.

MULTITHREADED SERVER (MTS)

A multithreaded server process enables many user processes to share a number of server processes. MTS improves the server efficiency by allowing any server to process an incoming request, rather than waiting for a specific user to issue a request.
MTS reduces the number of processes against instance, increases the number of possible users per node, offers automatic load balancing, reduces the number of idle server processes and reduces memory usage, system overhead and can be set to support connection pooling..

MTS Connecting

The listener waits for any connection requests from a user process, if the user process can connect to a dispatcher the listener gives the user process address of a dispatcher process. Once the connection has been established, either through a dispatcher or a dedicated server process the connection will be maintained for the duration of session.

Processing A Request In MTS

A user sends a request to its dispatcher, the dispatcher places the request into request queue in SGA, a shared server picks up the request from the request queue and processes the request, the shared server places the response on the calling dispatcher's response queue, the response is handed off to dispatcher, and the dispatcher returns the response to the user.

Request Queue - One request queue is shared by all dispatchers, shared servers monitor the request queue for new requests and the requests are processed on a first-in, first-out basis.

Response Queue - Shared servers place all completed requests on the calling dispatchers response queue, each dispatcher has its own response queue in the SGA, each dispatcher is responsible for sending completed requests back to appropriate user process and users are connected to the same dispatcher for the duration of a session.

Configuring the MTS

LOCAL_LISTENER - Specifies the service name defined in tnsnames.ora. It specifies the service names for listeners with which dispatchers register their services.

MTS_SERVICE - Establishes the name of the service for which the dispatchers of MTS instance associate. Using this name in the CONNECT string allows users to connect to an instance through a dispatcher.

MTS_DISPATCHERS - Specifies the number of dispatchers initially started for a given protocol.

MTS_MAX_DISPATCHER - Specifies the maximum number of dispatchers that can be started and needs "alter system" command to add more dispatchers than initially started.

MTS_SERVERS - It specifies the number of shared servers initially started. If you want Oracle to use shared servers, you must set MTS_SERVERS to at least 1 and if you omit the parameter or set it to 0, Oracle does not start any shared server at all.

MTS_MAX_SERVERS - It specifies the maximum number of shared servers that can be started and will allocate shared servers dynamically if need more than initially started.

CONNECTION POOLING

It is a resource utilization feature that allows you to maximize the number of physical network connections to MTS.

- One or more connections are established to a dispatcher in a MTS environment.
- When the maximum connections are reached for available dispatchers for a given protocol and connection pooling is configured the next session trying to establish a connection will wait for an existing idle connection to be temporarily disconnected by the dispatcher and then the connection is established.
- The connection that was idle becomes temporarily disconnected.
- To enable connection pooling the init.ora parameter MTS_DISPATCHERS must be configured.

CONNECTION MANAGER

It is a multipurpose networking solution to enable greater resource utilization for increased scalability, multiprotocol connectivity and secure network access control.

NOTE: *Connection Manager is installed on a middle tier in a multi-tiered environment. It greatly reduces the resource consumption on end tier and enables highly scalable configurations. When a connection is established through Connection Manager the database SID to which connection needs to be established is given by SOURCE_ROUTE. In addi-*

tion listener address specified for Connection Manager and destination listener address needs to be listed. To use connection concentration you are required to configure the MTS on destination database.
CMAN Contains the listening address for Connection Manager.
CMAN_PROFILE Contains CMAN configuration parameters.
CMAN_RULES Contains the rules for filtering incoming connection requests.
To configure network access control this command
in cman.ora needs to be configured. **SRV** is the
SID name of targeted database.
CMCTL START It is a control command for Connection Manager and
starts the Connection Manager or one of its components.
In addition you can specify additional arguments
for CMCTL utility and can also be configured in
CMAN.ORA
CONNECTION MANAGER WITH ORACLE NAMES SERVER
ə Oracle Names server automatically updates Connection Manager listening addresses.
ə In the client sqlnet.ora file different parameters can be set.

Troubleshooting the Network Environment

ə Tracing and Logging can be enabled to help troubleshoot the environment. The output from tracing and logging are stored in ASCII files.
ə To enable tracing on the client side we can use either Oracle Net8 Assistant in which a profile is stored and implemented through sqlnet.ora or to edit the profile itself. However, trace facility uses a large amount of disk space and may have significant effect upon system performance. This feature is disabled by default. In addition to client side tracing you can also enable tracing for the listener, Names server, Connection Manager and OEM daemon.
ə The level of tracing can be specified by following commands.
ə OFF (tracing disabled), USER (trace information applicable for users), ADMIN (trace information applicable for database administrators) and SUPPORT (trace information applicable for customer support staff)
ə Logging refers to the process by which network components note and append errorspecific information to a log file. Logging can be configured using Net8 Assistant.
Sqlnet.ora contains the information you put in profile.
ə An error stack refers to information that is produced by each layer in an Oracle communication stack as a result of network error and a log file provides this information.
NOTE:
ə To use a log file to diagnose a network error starting from the bottom of the file, locate the first non-zero entry in error support. This is usually the actual cause.

ə If that error does not provide the desired information, review next error in the stack until you locate the correct error information.
ə If the cause of error is still not clear, turn on tracing and re-execute the statement that produced the error message.

TRACE ASSISTANT

It helps to diagnose and troubleshoot network problems by a better understanding of flow of packets between network nodes, which component of Net8 is failing and related set of error codes. Net8 performs its functions by sending and receiving data packets and by specifying the trace level to SUPPORT you can view the actual contents of Net8 packets in your trace file. Trace assistant is also able to provide useful statistics and data about performance of an Oracle application across the network and this information is valuable in identifying potential performance bottlenecks.

***NOTE**: To find error codes related to your problem first check the log files, if they do not give the needed information run through checklist and if the checklist does not provide the solution, enable tracing and logging and if everything is still unclear—call support.*

Security in the Network Environment

A sound network must have solid security capabilities to protect against the network intrusions that comprise the following.

Data Privacy - Ensuring that data is not disclosed or stolen during transmission.

Data Integrity - Ensuring that data is not modified or disrupted during transmission.

Authentication - Confidence that 'users', 'hosts' and 'client' identities are correctly known.

Authorization - Permitting a user, a program, or a process to access an object or set of objects.

NETWORK SECURITY SOLUTIONS

To counter network security risks, the Advanced Networking Option (ANO) can be implemented and it enables following network security solutions:

ə Data Encryption
ə Cryptographic Checksumming
ə Authentication Mechanisms

ANO ensures data integrity through cryptographic checksums using MD5 algorithm. ANO also ensures data privacy through encryption. ANO is an optional product that does not ship with Net8; the only security that Net8 provides by default is user name and password encryption.

Data Encryption
Encryption is a technique that scrambles data using a key. Without the key the data can not be unscrambled. Special hardware is used to encrypt data. The two algorithms used for encryption are Data Encryption Standard (DES) and RSARC4. The longer the length of the key the stronger is the encryption, the stronger the encryption the longer it takes to break the code.
Cryptographic Checksumming
Labels each packet before it transmits and when the packet arrives, the server checks to see if the packet is in order.
Authentication Mechanisms
The mechanisms of authentication are network authentication services, token cards and biometric authentication adapter. To configure authentication the parameter is configured in sqlnet.ora.

تم جمعها من عدة ملفات
ولا تنسونا من صالح الدعاء