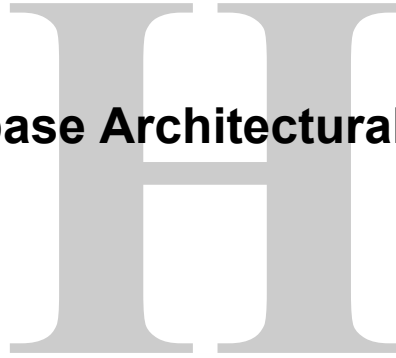


Oracle Database Architectural Components



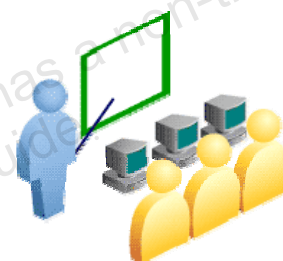
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this appendix, you should be able to:

- List the major database architectural components
- Describe the background processes
- Explain the memory structures
- Correlate the logical and physical storage structures



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

This appendix provides an overview of the Oracle Database architecture. You learn about the physical and logical structures and various components of Oracle Database and their functions.

Oracle Database Architecture: Overview

The Oracle Relational Database Management System (RDBMS) is a database management system that provides an open, comprehensive, integrated approach to information management.



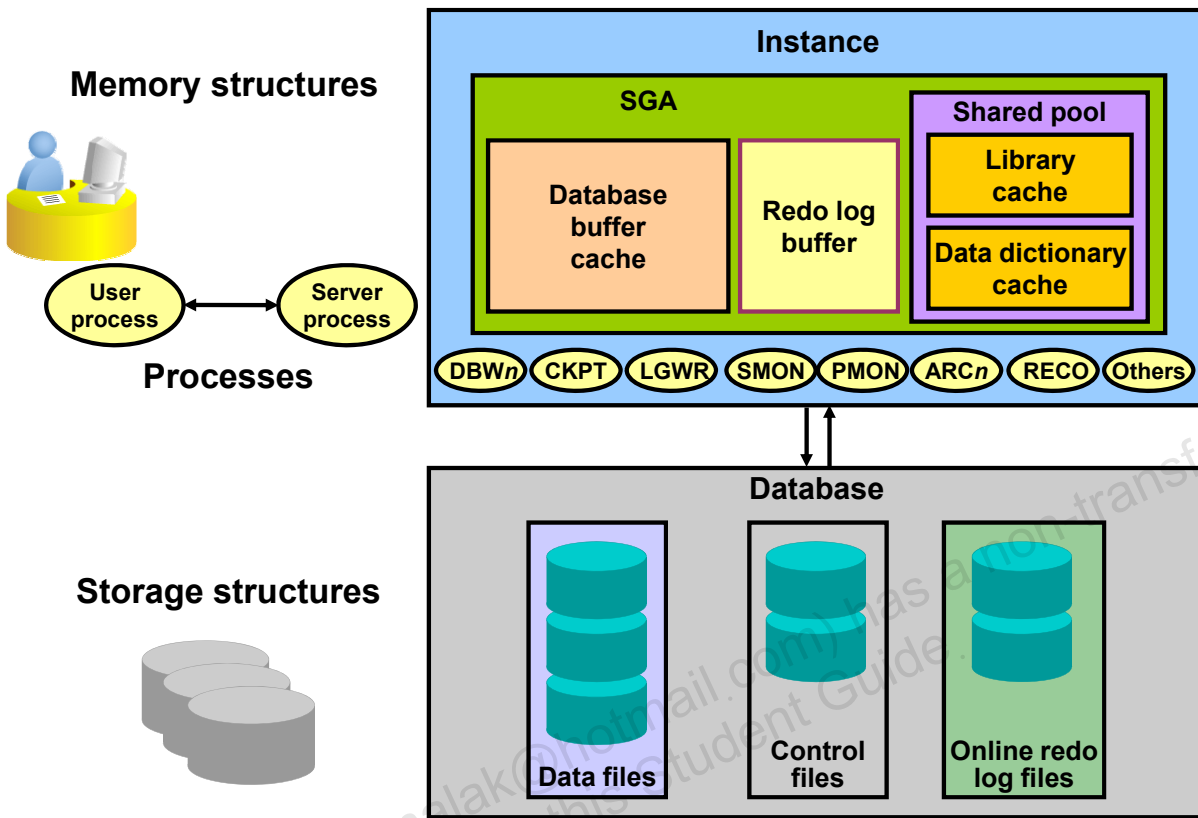
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A database is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information.

An Oracle database reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. This is accomplished while delivering high performance. At the same time, it prevents unauthorized access and provides efficient solutions for failure recovery.

Oracle Database Server Structures



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

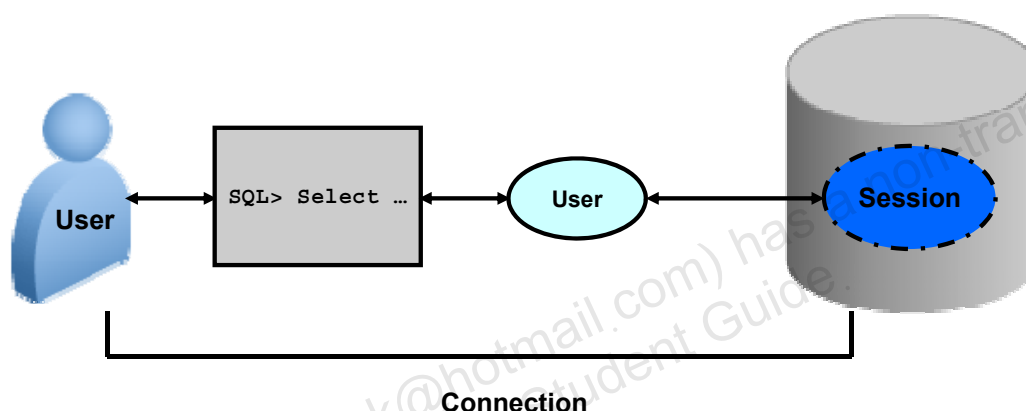
The Oracle Database consists of two main components—the instance and the database.

- The instance consists of the System Global Area (SGA), which is a collection of memory structures, and the background processes that perform tasks within the database. Every time an instance is started, the SGA is allocated and the background processes are started.
- The database consists of both physical structures and logical structures. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting access to logical storage structures. The physical storage structures include:
 - The control files where the database configuration is stored
 - The redo log files that have information required for database recovery
 - The data files where all data is stored

An Oracle instance uses memory structures and processes to manage and access the database storage structures. All memory structures exist in the main memory of the computers that constitute the database server. Processes are jobs that work in the memory of these computers. A process is defined as a “thread of control” or a mechanism in an operating system that can run a series of steps.

Connecting to the Database

- Connection: Communication pathway between a user process and a database instance
- Session: A specific connection of a user to a database instance through a user process



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To access information in the database, the user needs to connect to the database using a tool (such as SQL*Plus). After the user establishes connection, a session is created for the user. Connection and session are closely related to user process but are very different in meaning.

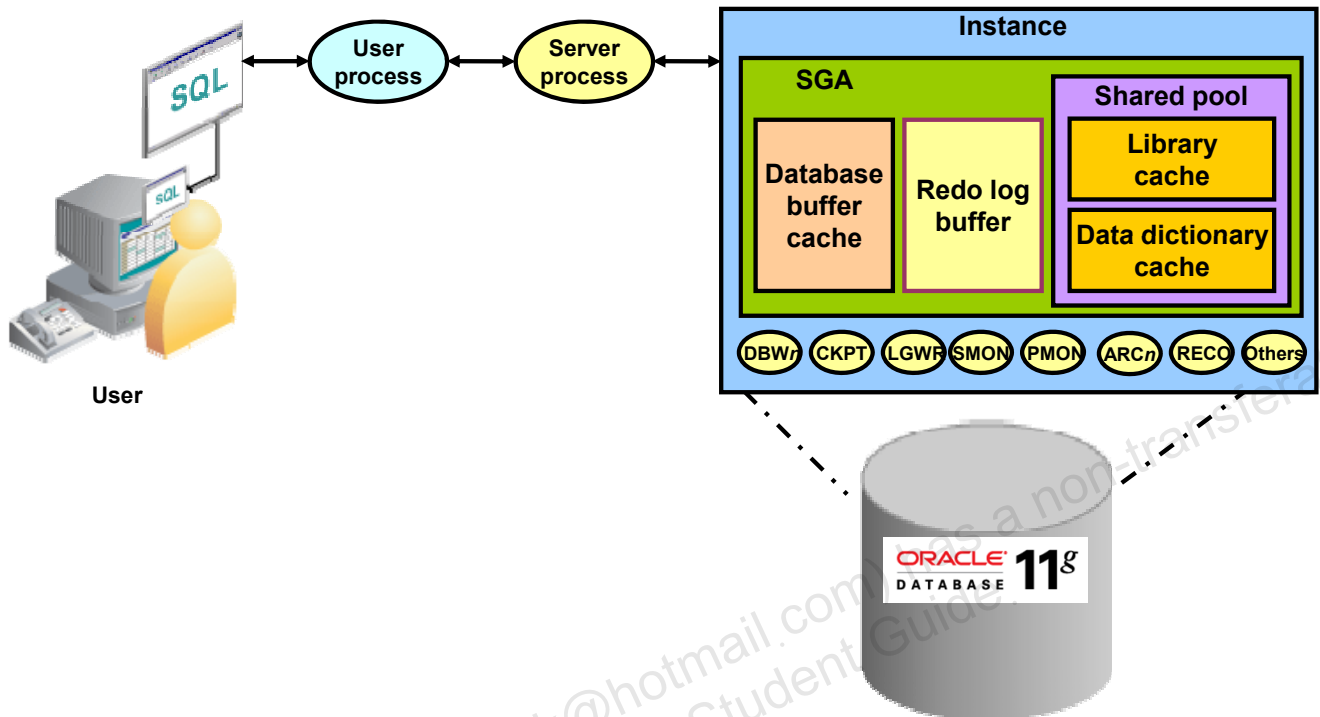
A connection is a communication pathway between a user process and an Oracle Database instance. A communication pathway is established using available interprocess communication mechanisms or network software (when different computers run the database application and Oracle Database, and communicate through a network).

A session represents the state of a current user login to the database instance. For example, when a user starts SQL*Plus, the user must provide a valid username and password, and then a session is established for that user. A session lasts from the time the user connects until the time the user disconnects or exits the database application.

In the case of a dedicated connection, the session is serviced by a permanent dedicated process. In the case of a shared connection, the session is serviced by an available server process selected from a pool, either by the middle tier or by Oracle shared server architecture.

Multiple sessions can be created and exist concurrently for a single Oracle Database user using the same username, but through different applications, or multiple invocations of the same application.

Interacting with an Oracle Database



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

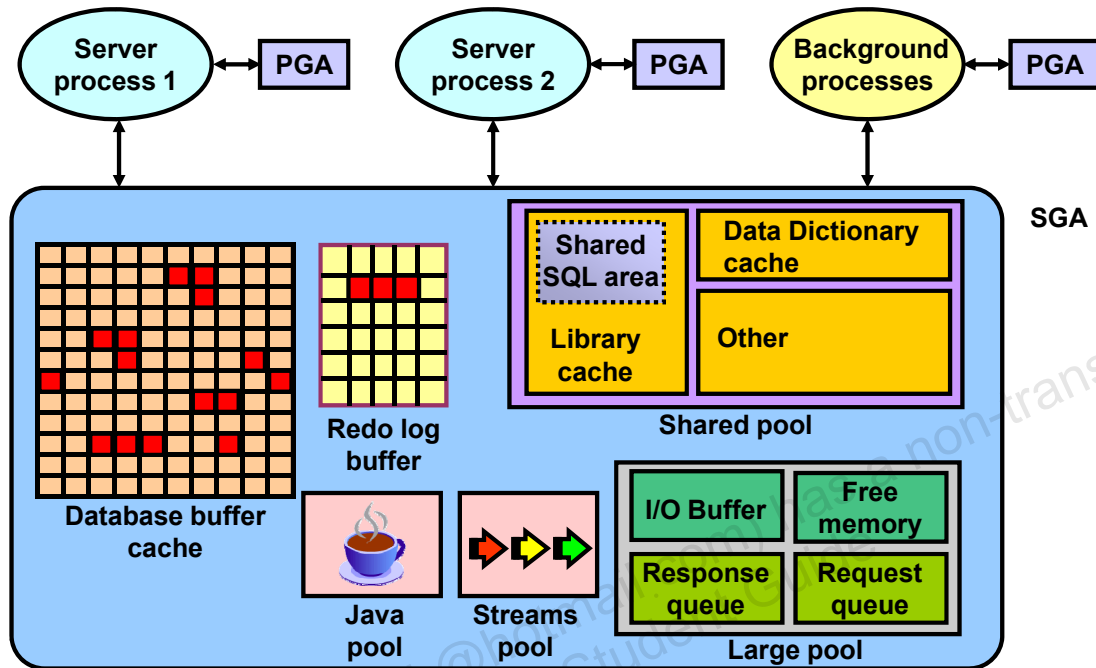
The following example describes Oracle Database operations at the most basic level. It illustrates an Oracle Database configuration where the user and associated server process are on separate computers, connected through a network.

1. An instance has started on a node where Oracle Database is installed, often called the host or database server.
2. A user starts an application spawning a user process. The application attempts to establish a connection to the server. (The connection may be local, client server, or a three-tier connection from a middle tier.)
3. The server runs a listener that has the appropriate Oracle Net Services handler. The server detects the connection request from the application and creates a dedicated server process on behalf of the user process.
4. The user runs a DML-type SQL statement and commits the transaction. For example, the user changes the address of a customer in a table and commits the change.
5. The server process receives the statement and checks the shared pool (an SGA component) for any shared SQL area that contains a similar SQL statement. If a shared SQL area is found, the server process checks the user's access privileges to the requested data, and the existing shared SQL area is used to process the statement. If not, a new shared SQL area is allocated for the statement, so it can be parsed and processed.

6. The server process retrieves any necessary data values, either from the actual data file (in which the table is stored) or those cached in the SGA.
7. The server process modifies data in the SGA. Because the transaction is committed, the log writer process (LGWR) immediately records the transaction in the redo log file. The database writer process (DBWn) writes modified blocks permanently to disk when doing so is efficient.
8. If the transaction is successful, the server process sends a message across the network to the application. If it is not successful, an error message is transmitted.
9. Throughout this entire procedure, the other background processes run, watching for conditions that require intervention. In addition, the database server manages other users' transactions and prevents contention between transactions that request the same data.

Oracle Memory Architecture

DB structures
→ **Memory**
- Process
- Storage



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Database creates and uses memory structures for various purposes. For example, memory stores program code being run, data shared among users, and private data areas for each connected user.

Two basic memory structures are associated with an instance:

- The System Global Area (SGA) is a group of shared memory structures, known as SGA components, that contain data and control information for one Oracle Database instance. The SGA is shared by all server and background processes. Examples of data stored in the SGA include cached data blocks and shared SQL areas.
- The Program Global Areas (PGA) are memory regions that contain data and control information for a server or background process. A PGA is nonshared memory created by Oracle Database when a server or background process is started. Access to the PGA is exclusive to the server process. Each server process and background process has its own PGA.

The SGA is the memory area that contains data and control information for the instance. The SGA includes the following data structures:

- **Database buffer cache:** Caches blocks of data retrieved from the database
- **Redo Log buffer:** Caches redo information (used for instance recovery) until it can be written to the physical redo log files stored on the disk
- **Shared pool:** Caches various constructs that can be shared among users
- **Large pool:** Is an optional area that provides large memory allocations for certain large processes, such as Oracle backup and recovery operations, and input/output (I/O) server processes
- **Java pool:** Is used for all session-specific Java code and data within the Java Virtual Machine (JVM)
- **Streams pool:** Is used by Oracle Streams to store information required by capture and apply

When you start the instance by using Enterprise Manager or SQL*Plus, the amount of memory allocated for the SGA is displayed.

With the dynamic SGA infrastructure, the size of the database buffer cache, the shared pool, the large pool, the Java pool, and the Streams pool changes without shutting down the instance.

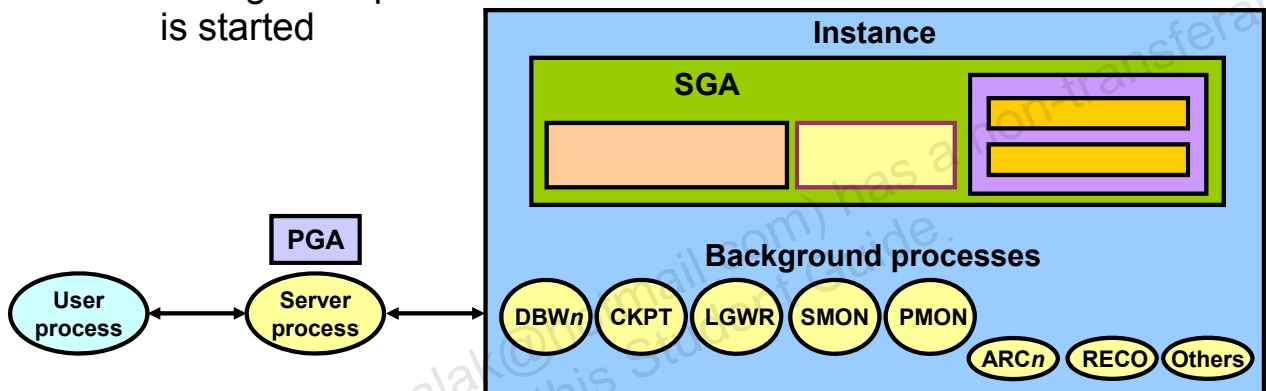
Oracle Database uses initialization parameters to create and configure memory structures. For example, the `SGA_TARGET` parameter specifies the total size of the SGA components. If you set `SGA_TARGET` to 0, Automatic Shared Memory Management is disabled.

Process Architecture

DB structures

- Memory
- **Process**
- Storage

- User process:
 - Is started when a database user or a batch process connects to the Oracle Database
- Database processes:
 - Server process: Connects to the Oracle instance and is started when a user establishes a session
 - Background processes: Are started when an Oracle instance is started



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The processes in an Oracle Database server can be categorized into two major groups:

- User processes that run the application or Oracle tool code
- Oracle Database processes that run the Oracle database server code. These include server processes and background processes.

When a user runs an application program or an Oracle tool such as SQL*Plus, Oracle Database creates a *user process* to run the user's application. The Oracle Database also creates a *server process* to execute the commands issued by the user process. In addition, the Oracle server also has a set of *background processes* for an instance that interact with each other and with the operating system to manage the memory structures and asynchronously perform I/O to write data to disk, and perform other required tasks.

The process structure varies for different Oracle Database configurations, depending on the operating system and the choice of Oracle Database options. The code for connected users can be configured as a dedicated server or a shared server.

- With a dedicated server, for each user, the database application is run by a different process (a user process) than the one that runs the Oracle server code (a dedicated server process).
- A shared server eliminates the need for a dedicated server process for each connection. A dispatcher directs multiple incoming network session requests to a pool of shared server processes. A shared server process serves any client request.

Server Processes

Oracle Database creates server processes to handle the requests of user processes connected to the instance. In some situations when the application and Oracle Database operate on the same computer, it is possible to combine the user process and the corresponding server process into a single process to reduce system overhead. However, when the application and Oracle Database operate on different computers, a user process always communicates with Oracle Database through a separate server process.

Server processes created on behalf of each user's application can perform one or more of the following:

- Parse and run SQL statements issued through the application.
- Read necessary data blocks from data files on disk into the shared database buffers of the SGA, if the blocks are not already present in the SGA.
- Return results in such a way that the application can process the information.

Background Processes

To maximize performance and accommodate many users, a multiprocess Oracle Database system uses some additional Oracle Database processes called background processes. An Oracle Database instance can have many background processes.

The following background processes are required for a successful startup of the database instance:

- Database writer (DBWn)
- Log writer (LGWR)
- Checkpoint (CKPT)
- System monitor (SMON)
- Process monitor (PMON)

The following background processes are a few examples of optional background processes that can be started if required:

- Recoverer (RECO)
- Job queue
- Archiver (ARCn)
- Queue monitor (QMNn)

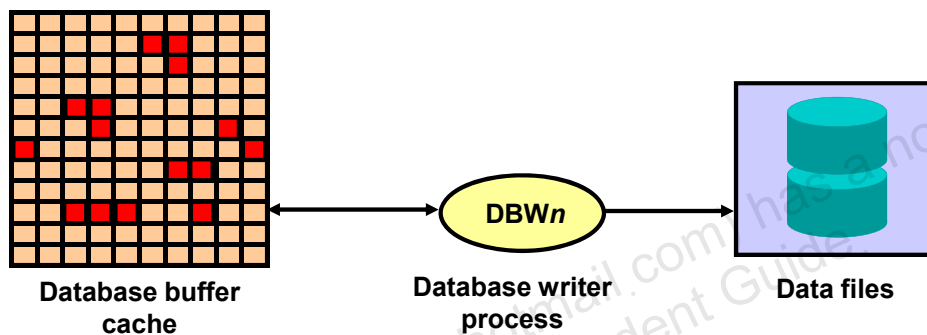
Other background processes may be found in more advanced configurations such as Real Application Clusters (RAC). See the `V$BGPROCESS` view for more information about the background processes.

On many operating systems, background processes are created automatically when an instance is started.

Database Writer Process

Writes modified (dirty) buffers in the database buffer cache to disk:

- Asynchronously while performing other processing
- Periodically to advance the checkpoint



ORACLE

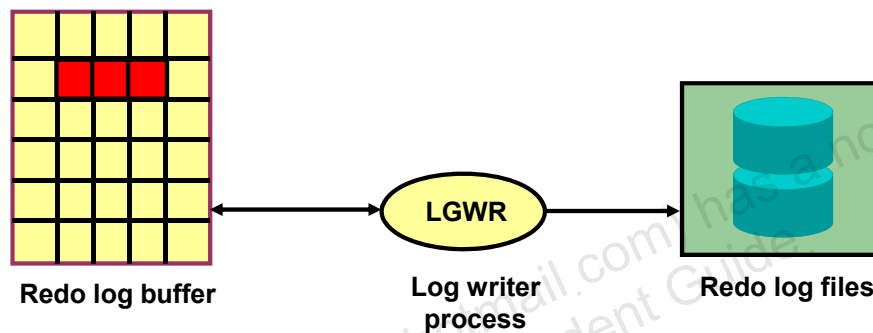
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The database writer (DBW n) process writes the contents of buffers to data files. The DBW n processes are responsible for writing modified (dirty) buffers in the database buffer cache to disk. Although one database writer process (DBW0) is adequate for most systems, you can configure additional processes (DBW1 through DBW9 and DBWa through DBWj) to improve write performance if your system modifies data heavily. These additional DBW n processes are not useful on uniprocessor systems.

When a buffer in the database buffer cache is modified, it is marked “dirty” and is added to the LRUW list of dirty buffers that is kept in system change number (SCN) order, thereby matching the order of Redo corresponding to these changed buffers that is written to the Redo logs. When the number of available buffers in the buffer cache falls below an internal threshold such that server processes find it difficult to obtain available buffers, DBW n writes dirty buffers to the data files in the order that they were modified by following the order of the LRUW list.

Log Writer Process

- Writes the redo log buffer to a redo log file on disk
- LGWR writes:
 - When a process commits a transaction
 - When the redo log buffer is one-third full
 - Before a DBWn process writes modified buffers to disk



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The log writer (LGWR) process is responsible for redo log buffer management by writing the redo log buffer entries to a redo log file on disk. LGWR writes all redo entries that have been copied into the buffer since the last time it wrote.

The redo log buffer is a circular buffer. When LGWR writes redo entries from the redo log buffer to a redo log file, server processes can then copy new entries over the entries in the redo log buffer that have been written to disk. LGWR normally writes fast enough to ensure that space is always available in the buffer for new entries, even when access to the redo log is heavy.

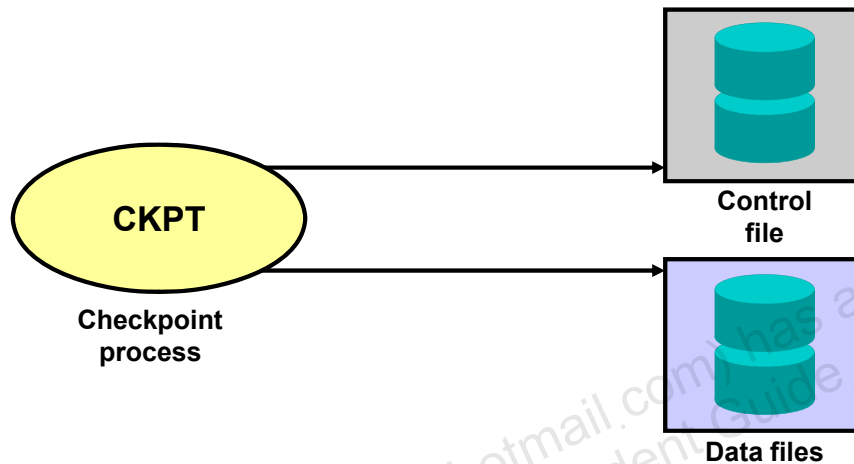
LGWR writes one contiguous portion of the buffer to disk. LGWR writes:

- When a user process commits a transaction
- When the redo log buffer is one-third full
- Before a DBWn process writes modified buffers to disk, if necessary

Checkpoint Process

Records checkpoint information in:

- The control file
- Each datafile header



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

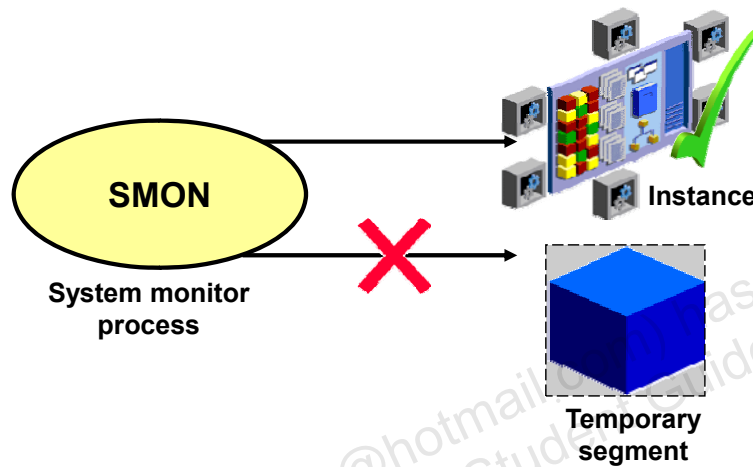
A checkpoint is a data structure that defines an SCN in the redo thread of a database. Checkpoints are recorded in the control file and each data file header, and are a crucial element of recovery.

When a checkpoint occurs, Oracle Database must update the headers of all data files to record the details of the checkpoint. This is done by the CKPT process. The CKPT process does not write blocks to disk; DBWR always performs that work. The SCNs recorded in the file headers guarantee that all the changes made to database blocks before that SCN have been written to disk.

The statistic DBWR checkpoints displayed by the `SYSTEM_STATISTICS` monitor in Oracle Enterprise Manager indicate the number of checkpoint requests completed.

System Monitor Process

- Performs recovery at instance startup
- Cleans up unused temporary segments



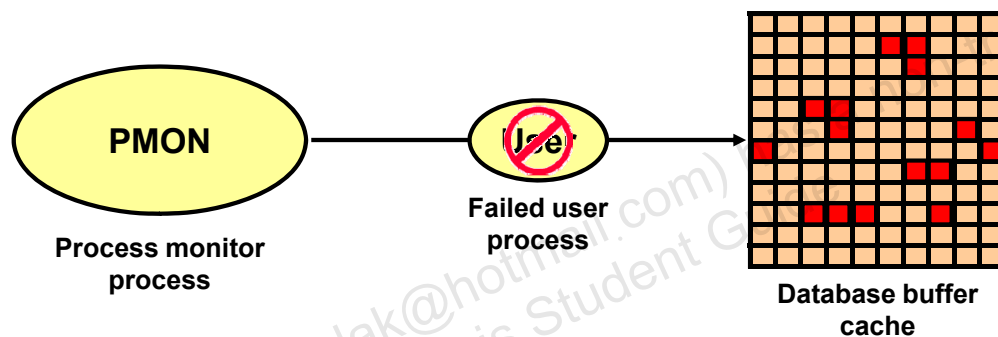
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The system monitor (SMON) process performs recovery, if necessary, at instance startup. SMON is also responsible for cleaning up temporary segments that are no longer in use. If any terminated transactions were skipped during instance recovery because of file-read or offline errors, SMON recovers them when the tablespace or file is brought back online. SMON checks regularly to see whether it is needed. Other processes can call SMON if they detect a need for it.

Process Monitor Process

- Performs process recovery when a user process fails:
 - Cleans up the database buffer cache
 - Frees resources used by the user process
- Monitors sessions for idle session timeout
- Dynamically registers database services with listeners



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The process monitor (PMON) performs process recovery when a user process fails. PMON is responsible for cleaning up the database buffer cache and freeing resources that the user process was using. For example, it resets the status of the active transaction table, releases locks, and removes the process ID from the list of active processes.

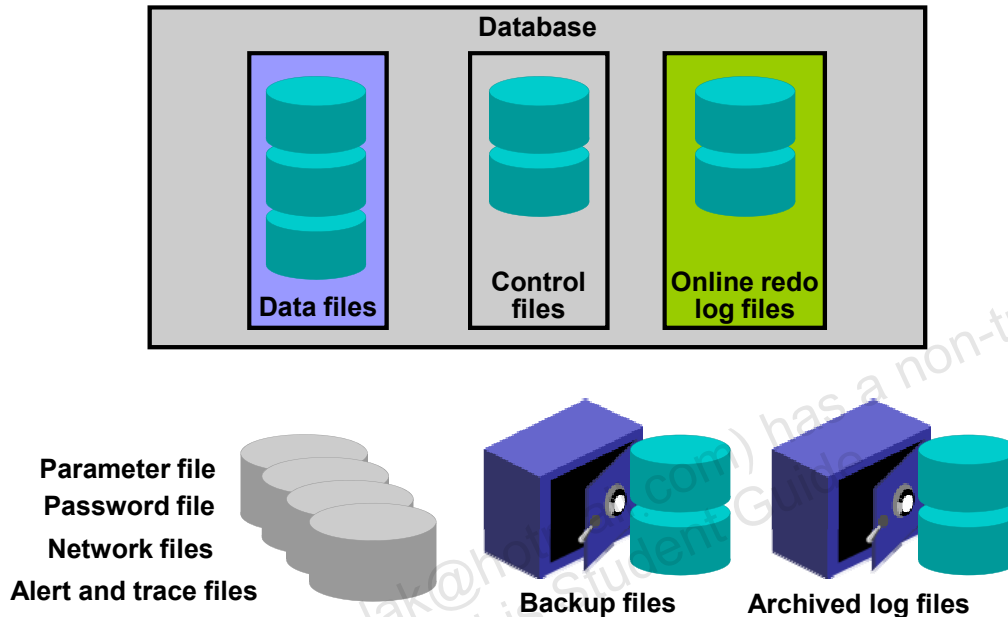
PMON periodically checks the status of dispatcher and server processes, and restarts any that have stopped running (but not any that Oracle Database has terminated intentionally). PMON also registers information about the instance and dispatcher processes with the network listener.

Like SMON, PMON checks regularly to see whether it is needed and can be called if another process detects the need for it.

Oracle Database Storage Architecture

DB structures

- Memory
- Process
- **Storage**



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The files that constitute an Oracle database are organized into the following:

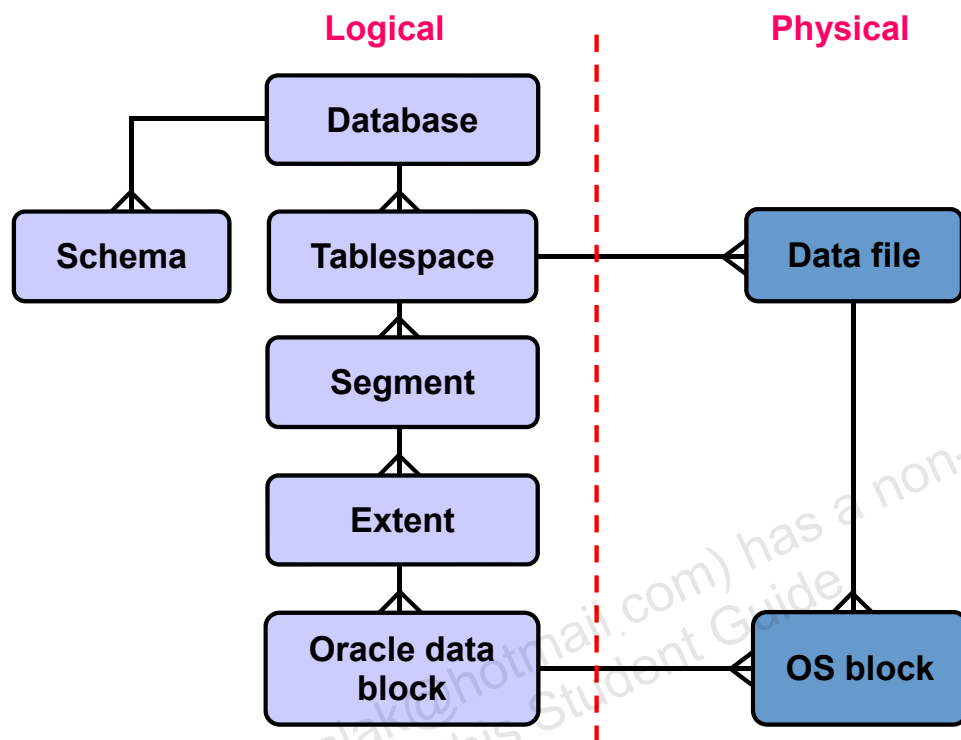
- **Control files:** Contain data about the database itself (that is, physical database structure information). These files are critical to the database. Without them, you cannot open data files to access the data within the database.
- **Data files:** Contain the user or application data of the database, as well as metadata and the data dictionary
- **Online redo log files:** Allow for instance recovery of the database. If the database server crashes and does not lose any data files, the instance can recover the database with the information in these files.

The following additional files are important to the successful running of the database:

- **Backup files:** Are used for database recovery. You typically restore a backup file when a media failure or user error has damaged or deleted the original file.
- **Archived log files:** Contain an ongoing history of the data changes (redo) that are generated by the instance. Using these files and a backup of the database, you can recover a lost data file. That is, archive logs enable the recovery of restored data files.
- **Parameter file:** Is used to define how the instance is configured when it starts up
- **Password file:** Allows `sysdba/sysoper/sysasm` to connect remotely to the database and perform administrative tasks

- **Network files:** Are used for starting the database listener and store information required for user connections
- **Trace files:** Each server and background process can write to an associated trace file. When an internal error is detected by a process, the process dumps information about the error to its trace file. Some of the information written to a trace file is intended for the database administrator, whereas other information is for Oracle Support Services.
- **Alert log files:** These are special trace entries. The alert log of a database is a chronological log of messages and errors. Each instance has one alert log file. Oracle recommends that you review this alert log periodically.

Logical and Physical Database Structures



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

An Oracle database has logical and physical storage structures.

Tablespaces

A database is divided into logical storage units called tablespaces, which group related logical structures together. For example, tablespaces commonly group all of an application's objects to simplify some administrative operations. You may have a tablespace for application data and an additional one for application indexes.

Databases, Tablespaces, and Data Files

The relationship among databases, tablespaces, and data files is illustrated in the slide. Each database is logically divided into one or more tablespaces. One or more data files are explicitly created for each tablespace to physically store the data of all logical structures in a tablespace. If it is a `TEMPORARY` tablespace, instead of a data file, the tablespace has a temporary file.

Schemas

A schema is a collection of database objects that are owned by a database user. Schema objects are the logical structures that directly refer to the database's data. Schema objects include such structures as tables, views, sequences, stored procedures, synonyms, indexes, clusters, and database links. In general, schema objects include everything that your application creates in the database.

Data Blocks

At the finest level of granularity, an Oracle database's data is stored in data blocks. One data block corresponds to a specific number of bytes of physical database space on the disk. A data block size is specified for each tablespace when it is created. A database uses and allocates free database space in Oracle data blocks.

Extents

The next level of logical database space is called an extent. An extent is a specific number of contiguous data blocks (obtained in a single allocation) that are used to store specific type of information.

Segments

The level of logical database storage above an extent is called a segment. A segment is a set of extents allocated for a certain logical structure. For example, the different types of segments include:

- **Data segments:** Each nonclustered, non-indexed-organized table has a data segment with the exception of external tables, global temporary tables, and partitioned tables, where each table has one or more segments. All of the table's data is stored in the extents of its data segment. For a partitioned table, each partition has a data segment. Each cluster has a data segment. The data of every table in the cluster is stored in the cluster's data segment.
- **Index segments:** Each index has an index segment that stores all of its data. For a partitioned index, each partition has an index segment.
- **Undo segments:** One UNDO tablespace is created per database instance that contains numerous undo segments to temporarily store *undo* information. The information in an undo segment is used to generate read-consistent database information and, during database recovery, to roll back uncommitted transactions for users.
- **Temporary segments:** Temporary segments are created by the Oracle Database when a SQL statement needs a temporary work area to complete execution. When the statement finishes execution, the temporary segment's extents are returned to the instance for future use. Specify a default temporary tablespace for every user or a default temporary tablespace, which is used database-wide.

The Oracle Database dynamically allocates space. When the existing extents of a segment are full, additional extents are added. Because extents are allocated as needed, the extents of a segment may or may not be contiguous on the disk.

Processing a SQL Statement

- Connect to an instance using:
 - The user process
 - The server process
- The Oracle server components that are used depend on the type of SQL statement:
 - Queries return rows.
 - Data manipulation language (DML) statements log changes.
 - Commit ensures transaction recovery.
- Some Oracle server components do not participate in SQL statement processing.

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Not all the components of an Oracle instance are used to process SQL statements. The user and server processes are used to connect a user to an Oracle instance. These processes are not part of the Oracle instance, but are required to process a SQL statement.

Some of the background processes, SGA structures, and database files are used to process SQL statements. Depending on the type of SQL statement, different components are used:

- Queries require additional processing to return rows to the user.
- DML statements require additional processing to log the changes made to the data.
- Commit processing ensures that the modified data in a transaction can be recovered.

Some required background processes do not directly participate in processing a SQL statement, but are used to improve performance and to recover the database. For example, the optional Archiver background process, ARCn, is used to ensure that a production database can be recovered.

Processing a Query

- Parse:
 - Search for an identical statement.
 - Check the syntax, object names, and privileges.
 - Lock the objects used during parse.
 - Create and store the execution plan.
- Execute: Identify the rows selected.
- Fetch: Return the rows to the user process.

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Queries are different from other types of SQL statements because, if successful, they return data as results. Other statements simply return success or failure, whereas a query can return one row or thousands of rows.

There are three main stages in the processing of a query:

- Parse
- Execute
- Fetch

During the *parse* stage, the SQL statement is passed from the user process to the server process, and a parsed representation of the SQL statement is loaded into a shared SQL area.

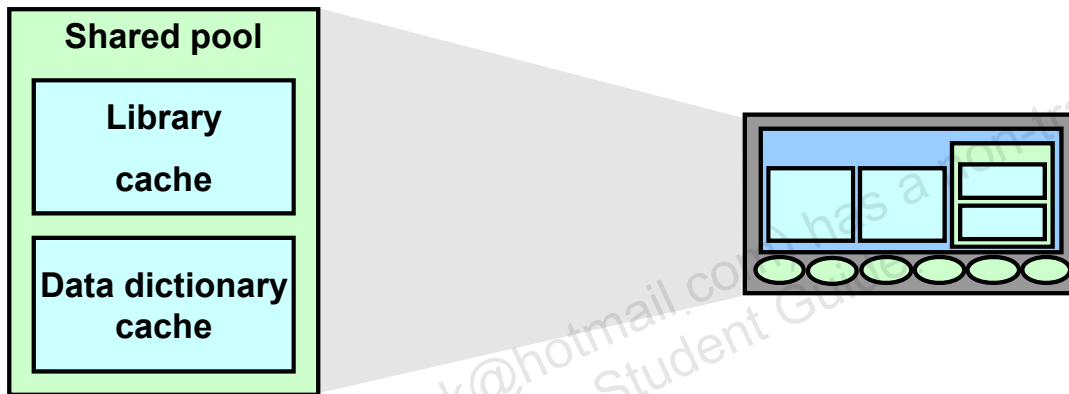
During parse, the server process performs the following functions:

- Searches for an existing copy of the SQL statement in the shared pool
- Validates the SQL statement by checking its syntax
- Performs data dictionary lookups to validate table and column definitions

The execute stage executes the statement using the best optimizer approach and the fetch retrieves the rows back to the user.

Shared Pool

- The library cache contains the SQL statement text, parsed code, and execution plan.
- The data dictionary cache contains table, column, and other object definitions and privileges.
- The shared pool is sized by `SHARED_POOL_SIZE`.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

During the parse stage, the server process uses the area in the SGA known as the shared pool to compile the SQL statement. The shared pool has two primary components:

- Library cache
- Data dictionary cache

Library Cache

The library cache stores information about the most recently used SQL statements in a memory structure called a shared SQL area. The shared SQL area contains:

- The text of the SQL statement
- The parse tree, which is a compiled version of the statement
- The execution plan, with steps to be taken when executing the statement

The optimizer is the function in the Oracle server that determines the optimal execution plan.

If a SQL statement is reexecuted and a shared SQL area already contains the execution plan for the statement, the server process does not need to parse the statement. The library cache improves the performance of applications that reuse SQL statements by reducing parse time and memory requirements. If the SQL statement is not reused, it is eventually aged out of the library cache.

Data Dictionary Cache

The data dictionary cache, also known as the dictionary cache or row cache, is a collection of the most recently used definitions in the database. It includes information about database files, tables, indexes, columns, users, privileges, and other database objects.

During the parse phase, the server process looks for the information in the dictionary cache to resolve the object names specified in the SQL statement and to validate the access privileges. If necessary, the server process initiates the loading of this information from the data files.

Sizing the Shared Pool

The size of the shared pool is specified by the `SHARED_POOL_SIZE` initialization parameter.

Database Buffer Cache

- The database buffer cache stores the most recently used blocks.
- The size of a buffer is based on `DB_BLOCK_SIZE`.
- The number of buffers is defined by `DB_BLOCK_BUFFERS`.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

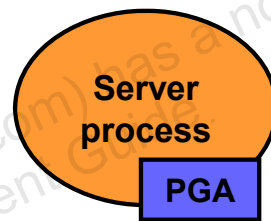
When a query is processed, the server process looks in the database buffer cache for any blocks it needs. If the block is not found in the database buffer cache, the server process reads the block from the data file and places a copy in the buffer cache. Because subsequent requests for the same block may find the block in memory, the requests may not require physical reads. The Oracle server uses a least recently used algorithm to age out buffers that have not been accessed recently to make room for new blocks in the buffer cache.

Sizing the Database Buffer Cache

The size of each buffer in the buffer cache is equal to the size of an Oracle block, and it is specified by the `DB_BLOCK_SIZE` parameter. The number of buffers is equal to the value of the `DB_BLOCK_BUFFERS` parameter.

Program Global Area (PGA)

- Is not shared
- Is writable only by the server process
- Contains:
 - Sort area
 - Session information
 - Cursor state
 - Stack space



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

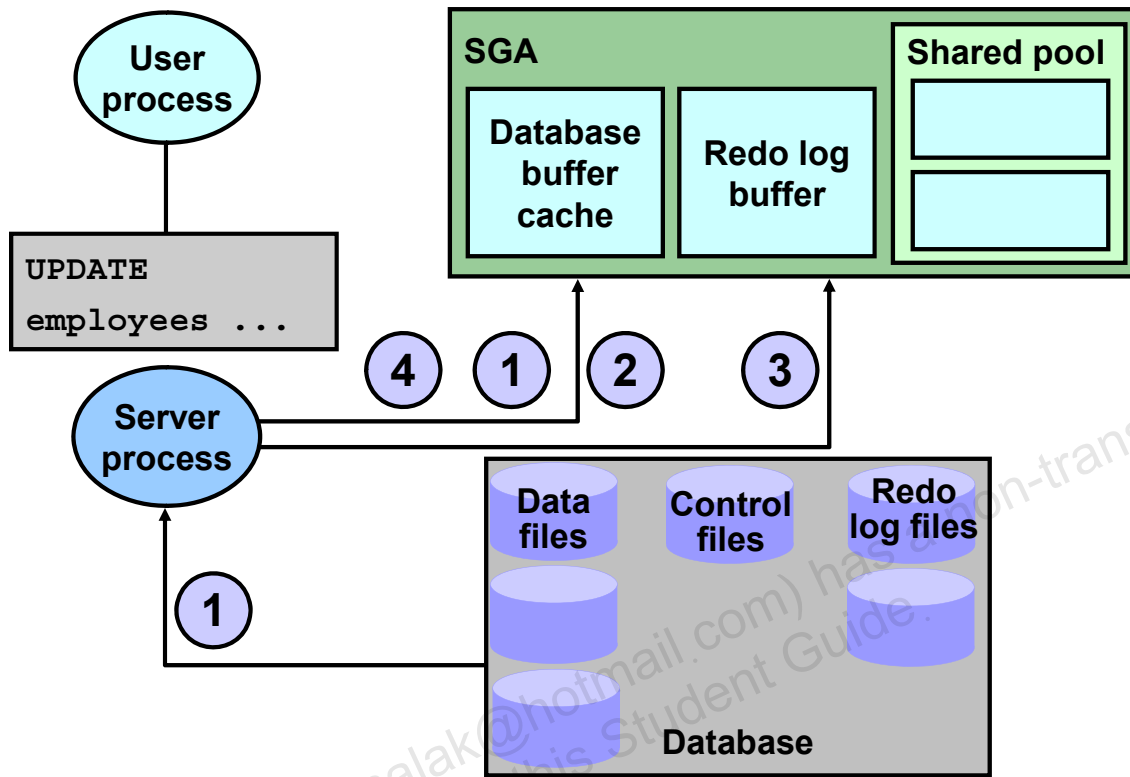
A Program Global Area (PGA) is a memory region that contains data and control information for a server process. It is a nonshared memory created by Oracle when a server process is started. Access to it is exclusive to that server process, and is read and written only by the Oracle server code acting on behalf of it. The PGA memory allocated by each server process attached to an Oracle instance is referred to as the aggregated PGA memory allocated by the instance.

In a dedicated server configuration, the PGA of the server includes the following components:

- **Sort area:** Is used for any sorts that may be required to process the SQL statement
- **Session information:** Includes user privileges and performance statistics for the session
- **Cursor state:** Indicates the stage in the processing of the SQL statements that are currently used by the session
- **Stack space:** Contains other session variables

The PGA is allocated when a process is created, and deallocated when the process is terminated.

Processing a DML Statement



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A data manipulation language (DML) statement requires only two phases of processing:

- Parse is the same as the parse phase used for processing a query.
- Execute requires additional processing to make data changes.

DML Execute Phase

To execute a DML statement:

- If the data and rollback blocks are not already in the buffer cache, the server process reads them from the data files into the buffer cache.
- The server process places locks on the rows that are to be modified.
- In the redo log buffer, the server process records the changes to be made to the rollback and data blocks.
- The rollback block changes record the values of the data before it is modified. The rollback block is used to store the “before image” of the data, so that the DML statements can be rolled back if necessary.
- The data block changes record the new values of the data.

The server process records the “before image” to the rollback block and updates the data block. Both of these changes are done in the database buffer cache. Any changed blocks in the buffer cache are marked as dirty buffers (that is, buffers that are not the same as the corresponding blocks on the disk).

The processing of a `DELETE` or `INSERT` command uses similar steps. The “before image” for a `DELETE` contains the column values in the deleted row, and the “before image” of an `INSERT` contains the row location information.

Because the changes made to the blocks are only recorded in memory structures and are not written immediately to disk, a computer failure that causes the loss of the SGA can also lose these changes.

Redo Log Buffer

- Has its size defined by `LOG_BUFFER`
- Records changes made through the instance
- Is used sequentially
- Is a circular buffer



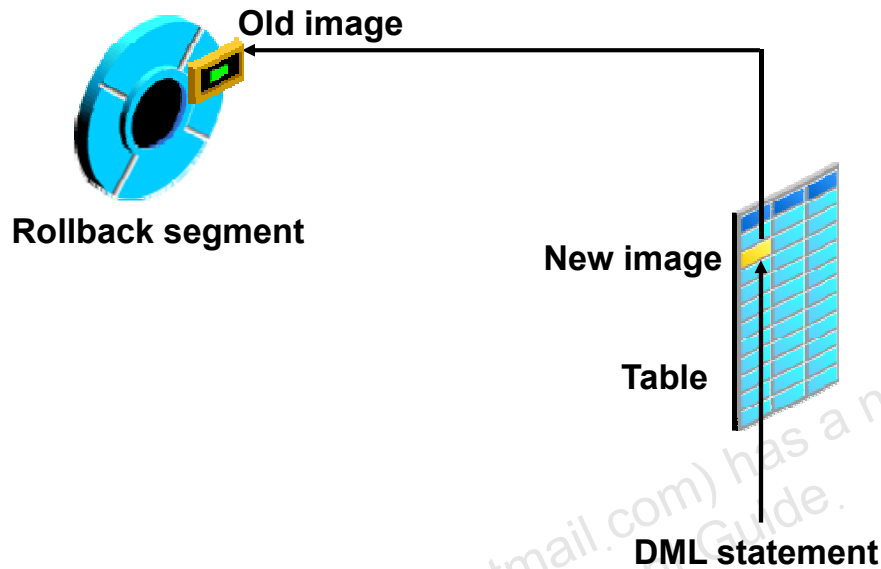
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The server process records most of the changes made to data file blocks in the redo log buffer, which is a part of the SGA. The redo log buffer has the following characteristics:

- Its size in bytes is defined by the `LOG_BUFFER` parameter.
- It records the block that is changed, the location of the change, and the new value in a redo entry. A redo entry makes no distinction between the types of block that is changed; it only records which bytes are changed in the block.
- The redo log buffer is used sequentially, and changes made by one transaction may be interleaved with changes made by other transactions.
- It is a circular buffer that is reused after it is filled, but only after all the old redo entries are recorded in the redo log files.

Rollback Segment



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

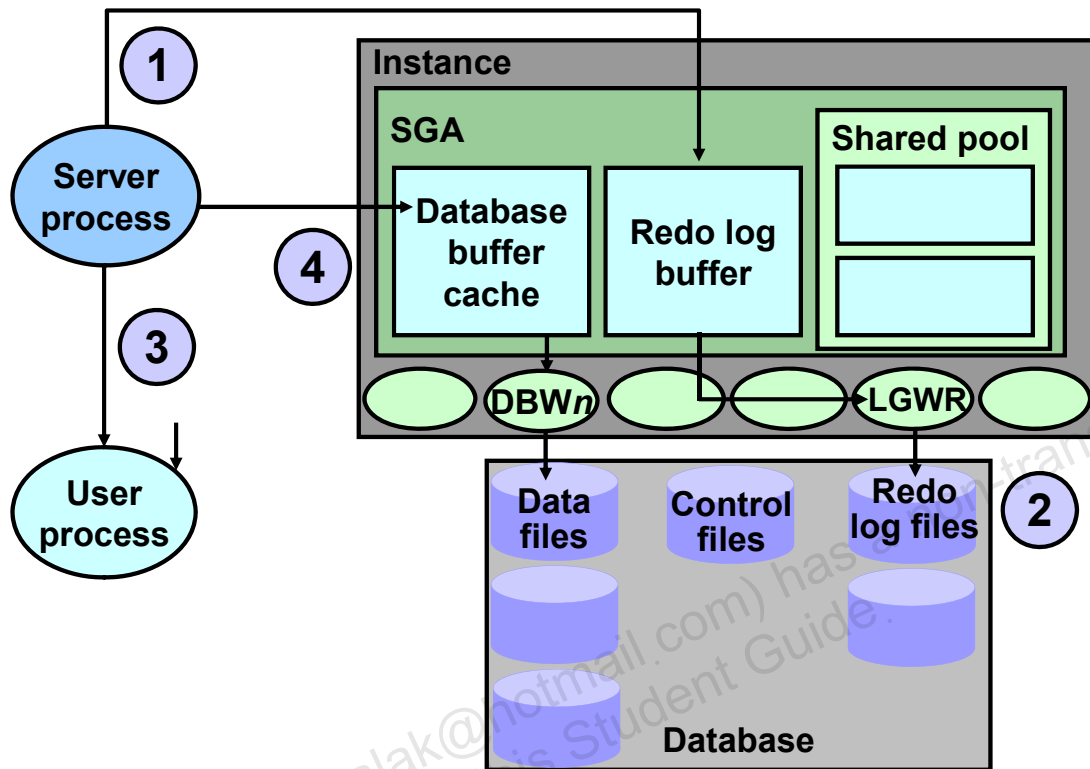
Before making a change, the server process saves the old data value in a rollback segment. This “before image” is used to:

- Undo the changes if the transaction is rolled back
- Provide read consistency by ensuring that other transactions do not see uncommitted changes made by the DML statement
- Recover the database to a consistent state in case of failures

Rollback segments, such as tables and indexes, exist in data files, and rollback blocks are brought into the database buffer cache as required. Rollback segments are created by the DBA.

Changes to rollback segments are recorded in the redo log buffer.

COMMIT Processing



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Oracle server uses a fast COMMIT mechanism that guarantees that the committed changes can be recovered in case of instance failure.

System Change Number

Whenever a transaction commits, the Oracle server assigns a commit SCN to the transaction. The SCN is monotonically incremented and is unique within the database. It is used by the Oracle server as an internal time stamp to synchronize data and to provide read consistency when data is retrieved from the data files. Using the SCN enables the Oracle server to perform consistency checks without depending on the date and time of the operating system.

Steps in Processing COMMITs

When a COMMIT is issued, the following steps are performed:

1. The server process places a commit record, along with the SCN, in the redo log buffer.
2. LGWR performs a contiguous write of all the redo log buffer entries up to and including the commit record to the redo log files. After this point, the Oracle server can guarantee that the changes will not be lost even if there is an instance failure.

3. The user is informed that the `COMMIT` is complete.
4. The server process records information to indicate that the transaction is complete and that resource locks can be released.

Flushing of the dirty buffers to the data file is performed independently by `DBW0` and can occur either before or after the commit.

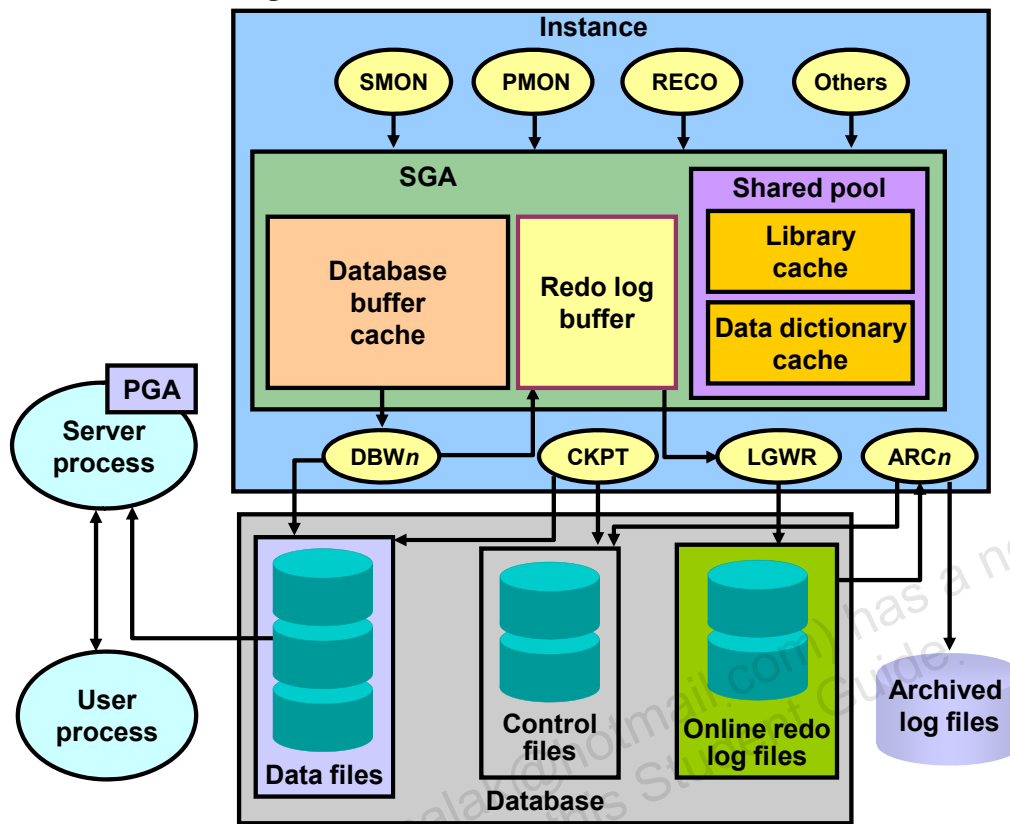
Advantages of the Fast `COMMIT`

The fast `COMMIT` mechanism ensures data recovery by writing changes to the redo log buffer instead of the data files. It has the following advantages:

- Sequential writes to the log files are faster than writing to different blocks in the data file.
- Only the minimal information that is necessary to record changes is written to the log files; writing to the data files would require whole blocks of data to be written.
- If multiple transactions request to commit at the same time, the instance piggybacks redo log records into a single write.
- Unless the redo log buffer is particularly full, only one synchronous write is required per transaction. If piggybacking occurs, there can be less than one synchronous write per transaction.
- Because the redo log buffer may be flushed before the `COMMIT`, the size of the transaction does not affect the amount of time needed for an actual `COMMIT` operation.

Note: Rolling back a transaction does not trigger LGWR to write to disk. The Oracle server always rolls back uncommitted changes when recovering from failures. If there is a failure after a rollback, before the rollback entries are recorded on disk, the absence of a commit record is sufficient to ensure that the changes made by the transaction are rolled back.

Summary of the Oracle Database Architecture



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

An Oracle database comprises an instance and its associated database:

- An instance comprises the SGA and the background processes
 - **SGA:** Database buffer cache, redo log buffer, shared pool, and so on
 - **Background processes:** SMON, PMON, DBWn, CKPT, LGWR, and so on
- A database comprises storage structures:
 - **Logical:** Tablespaces, schemas, segments, extents, and Oracle block
 - **Physical:** Data files, control files, redo log files

When a user accesses the Oracle database through an application, a server process communicates with the instance on behalf of the user process.

Summary

In this appendix, you should have learned how to:

- List the major database architectural components
- Describe the background processes
- Explain the memory structures
- Correlate the logical and physical storage structures

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.