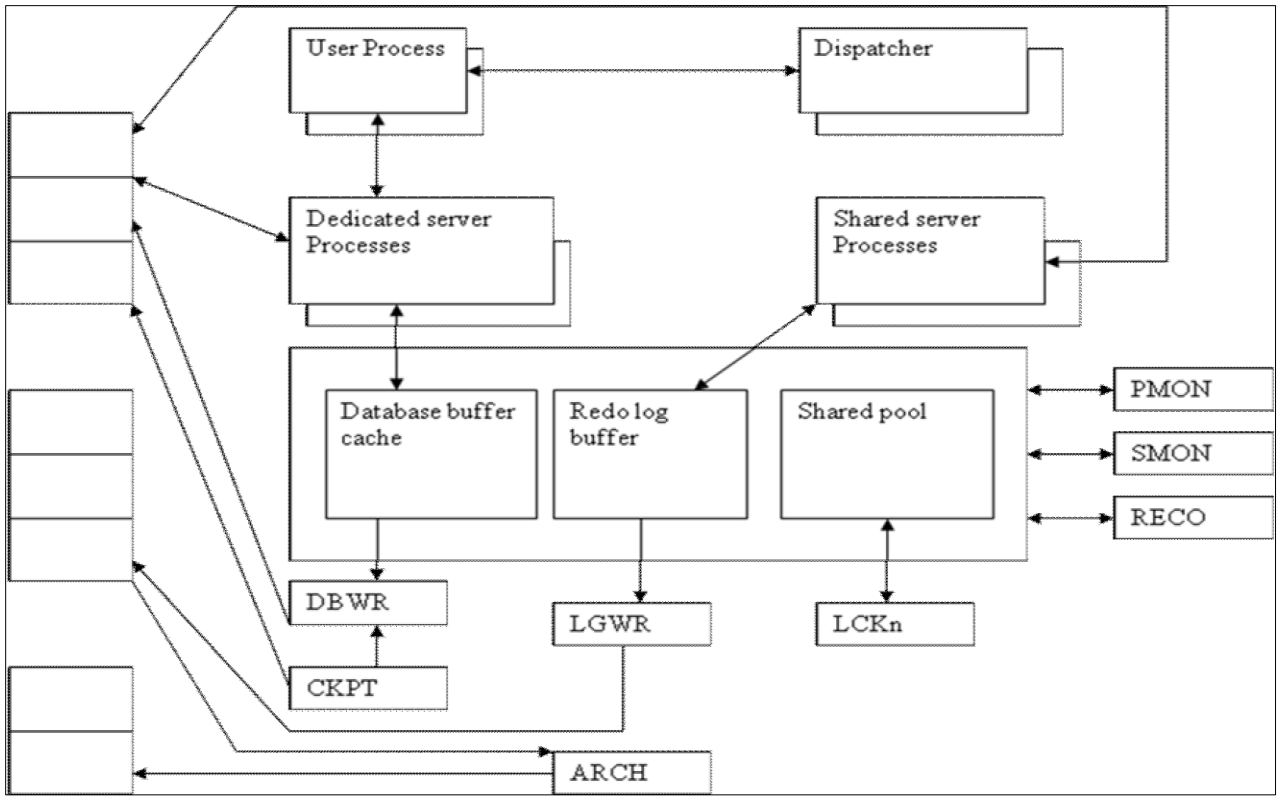
Topics Covered

- Instance Architecture (Database Processes, Memory Structure)
- Creating & Altering Database
- Opening & shutdown Database
- Data dictionary
- Control Files, Redo Logs files
- Tablespace (Create, Alter, Drop), types of tablesplace (system, read only, temporary, tool, user, data and index, rollback)
- Rollback Segment (Create, Alter) (System & Transaction RBS)
- Oracle Blocks
- Import
- Export
- SQL*Loader
- Oracle Scheduler Concepts
- Managing Resources with Oracle Database Resource Manager

5	Oracle Database Structure and Storage Database, Resource Management

Instance Architecture (Database Processes, Memory Structure)


User Process
Dispatcher
Dedicated server Processes
Shared server Processes
Database buffer cache
Redo log buffer
Shared pool
PMON
SMON
RECO
DBWR
LGWR
LCKn
CKPT
ARCH

Database Processes

The oracle uses two types of process.

1. User process
2. Background process

1. User Process:

- User or client, processes are the user's connections to the RDBMS system.
- It starts at the time when a oracle server starts.
- The user process manipulator the user's input and communicates with the oracle server.

2. Background process:

- It starts when an oracle instance is started.
- They are used to perform various tasks within the RDBMS system.
- They perform system maintenance.

1) DBWR(Database Writer):

- It is responsible for writing dirty data blocks from database block.
- When a transaction changes data in a data block need not be immediately written to disk.

2) LGWR(Log Writer):

- The LGWR process is responsible for writing data from the log buffer to the redo log files.

3) CKPT(Check Point):

- The CKPT process is responsible for signaling the DBWR process to perform a checkpoint and to update all the data files and control files.

4) PMON(Process Monitor):

- It is responsible for keeping track of database processes and cleaning up.
- It contains data and control information held by a single server process.
- It is created when a user process is created and is released when the user process is terminated.

5) SMON(System Monitor):

- SMON performs instance recovery at instance startup.
- This includes cleaning temporary segments and recovering transactions.

6) RECO(Recovery):

- It is used to clean transaction that were pending in a distributed database.
- It is responsible for committing or rolling back.

7) ARCH(Archiver):

- It is responsible for copying the online redo log files.

Memory Structure

The oracle basic memory structure consists of two memory area.

1. System Global Area (SGA)
2. Program Global Area (PGA)

➢ System Global Area:

- Oracle use SGA to store data and control information for oracle instance.
- The SGA consists of the following elements.

1) The database buffer cache:

- It is an area in memory that holds all the blocks read from disk for query or modification.
- This buffer is managed in a manner that free blocks are always made available for new blocks.
- The contents of the database buffer cache are written to the data file on disk by the database writer background process.

2) The redo log buffer:

- It is a circular buffer that stores all changes made in the database.
- It contains periodically modification.
- It done by the log writer background process.

3) The shared Pool:

- It is the area of SGA that stores shared memory structures such as shared SQL areas.
- It is used to store most recently executed SQL statements.

➢ **Program Global Area:**

- The PGA is a memory area that contains data and control information for the oracle server processes.
- PGA is allocated when a process is created and de-allocated when the process is terminated.
- It consists of the following components.

1) Stack space:

- The memory that holds the session's variables, arrays and so on.

2) Session information:

- If you are not running the multithreaded server the session information is stored in PGA.

3) Private SQL Area:

- This is an area where binding variable and runtime buffers are store.

Creating & Altering Database

Creating Database:

- The CREATE DATABASE statement is used to create a new SQL database.

Syntax

```
Create database <name>
[datafile]
[max logfiles n]
[max datafile n]
[archivelog/noarchive log]
[maxlogmemebers n]
```

Altering Database:

Syntax

```
Alter database <name>
```

Opening & shutdown Database

- A DBA can perform a startup & shutdown database.

Opening Database

- When a database started three important steps are executed.

1) Instance Creation:

- This is the first step of starting database.

- In it SGA is configured.
- The parameter file identifies the name of the database.

2) Mounting the database:

- When a database is mounted, a database administrator can perform maintenance or administrative task.

3) Opening the database:

- It is the last phase of starting a database.
- This phase has to be performed so that users of the database can access the data in database.
- In this level oracle server verifies the consistency of the database.

Syntax:

Startup [force][restrict][pfile=filename][open[recover][database] [readonly\mount\nomount]]

Parameters:

1) nomount:

- It starts database without mounting.

2) Mount:

- Starts an instance and mount a database without opening.

3) Open:

- Database is open in normal way.

4) Restrict:

- It open database but does not allow to change.

5) Force:

- It use for recovery when database does not start.

shutdown Database

- You can either perform a proper or improper shut down.
- During the proper shutdown the three phase which are used for startup and performed in reversed order.
- First the database is closed database is demounted and then memory is released.

Syntax:

Shutdown [normal | immediate | transactional | abort]

Parameters:

1) Normal:

- It is the default mode.
- The oracle server waits for all currently connected users to disconnect their session.
- New connections are not permitted.

2) Immediate:

- In this mode the oracle server automatically rolls back all currently active transaction.

3) Transactional:

- In this mode, when database is shut down, all currently active transaction will be allowed to complete.

4) Abort:

- It is one type of improper shutdown.

Data dictionary

- The Oracle data dictionary is one of the most important components of the Oracle DBMS.
- It contains all information about the structures and objects of the database such as
  - tables,
  - columns,
  - users,
  - data files etc.
- The data stored in the data dictionary are also often called metadata.
- Although it is usually the domain of database administrators (DBAs), the data dictionary is a valuable source of information for end users and developers.
- The data dictionary consists of two levels:
  1. The internal level contains all base tables that are used by the various DBMS software components and they are normally not accessible by end users.
  2. The external level provides numerous views on these base tables to access information about objects and structures at different levels of detail.

Data Dictionary Tables

An installation of an Oracle database always includes the creation of three standard Oracle users:

SYS	This is the owner of all data dictionary tables and views. This user has the highest privileges to manage objects and structures of an Oracle database such as creating new users.
SYSTEM	This is the owner of tables used by different tools such SQL*Forms, SQL*Reports etc. This user has less privileges than SYS.
PUBLIC	This is a "dummy" user in an Oracle database. All privileges assigned to this user are automatically assigned to all users known in the database.

Data Dictionary Contains...

The tables and views provided by the data dictionary contain information about

- users and their privileges,
- tables, table columns and their data types, integrity constraints, indexes,
- statistics about tables and indexes used by the optimizer,
- privileges granted on database objects,
- storage structures of the database.

Viewing the data dictionary

The SQL command

```
select * from DICT[IOARY];
        select * from DICT;
```

TABLE_NAME	COMMENTS
DBA_ROLES	All Roles which exist in the database
DBA_PROFILES	Display all profiles and their limits
USER_RESOURCE_LIMITS	Display resource limit of the user
USER_PASSWORD_LIMITS	Display password limits of the user
USER_CATALOG	Tables, Views, Synonyms and Sequences owned by the user
ALL_CATALOG	All tables, views, synonyms, sequences accessible to the user
DBA_CATALOG	All database Tables, Views, Synonyms, Sequences

The query select * from TAB;
retrieves the names of all tables owned by the user who issues this command.
select * from TAB;

TNAME	TABTYPE	CLUSTERID
SYSCATALOG	SYNONYM	-
CATALOG	SYNONYM	-
TAB	SYNONYM	-
COL	SYNONYM	-
TABQUOTAS	SYNONYM	-
SYSFILES	SYNONYM	-
PUBLICSYN	SYNONYM	-
MVIEW$_ADV_WORKLOAD	TABLE	-
MVIEW$_ADV_BASETABLE	TABLE	-
MVIEW$_ADV_SQLDEPEND	TABLE	-

Data Dictionary Views

- The views provided by the data dictionary are divided into three groups:
  USER, ALL , and DBA.
- The group name builds the prefix for each view name. For some views, there are associated synonyms as given in brackets below.
- USER : Tuples in the USER views contain information about objects owned by the account performing the SQL query (current user)

USER TABLES	all tables with their name, number of columns, storage information, statistical information etc. (TABS)
USER CATALOG	tables, views, and synonyms (CAT)
USER COL COMMENTS	comments on columns
USER CONSTRAINTS	constraint definitions for tables
USER INDEXES	all information about indexes created for tables
USER USERS	information about the current user
USER VIEWS	views defined by the user

Retreiving Data Dictionaries:
select * from USER_TABLES ;
select * from USER_COL_COMMENTS;

Control Files, Redo Logs files

Control Files

- It contains information used to start an instance such as location of data files and redo log files.
- Oracle provides a mechanism for storing multiple copies of control files.
- It resides on the operating system file system.
- A control file can also be added after the database has been created.
- It can created using create control file command.
- It also contains the physical structure of database.
- The control files will automatically update whenever needed.
- The size of the oracle control file varies according to the complexity of your database structure.

Redo Logs files

- They hold information which are used for recovery in the event of system failure.
- They stores a log of all changes made to the database.
- If redo log information is lost, no one can recover system failure.
- Each time the data is changed then the records of log file will be changed.
- When the oracle is restarted, the information in the redo log file will be reproduced.
- All previously committed transaction will be recovered.

Tablespace (Create, Alter, Drop), types of tablesplace (system, read only, temporary, tool, user, data and index, rollback)

- A database is divided into one or more logical storage units called table space.
- A table space is used to group related logical structures together.
- The table space is used in the following reason.
  - Control disk space allocation for database data.
  - Assign specific space for database users.
  - Control availability of data.
  - Perform partial database backup.

Create Tablespace

- The CREATE TABLESPACE statement allows you to create a new tablespace.

Syntax:

```
CREATE TABLESPACE tablespace_name
 DATAFILE 'filename` [SIZE integer [K | M] [REUSE]] [Autoextend_Clause]
          [,'filename` [SIZE integer [K | M] [REUSE]] [Autoextend_Clause]]
  DEFAULT STORAGE Storage_Clause
  [ONLINE | OFFLINE]
  [PERMANENT | TEMPORARY]
  [LOGGING | NOLOGGING]
  [MINIMUM EXTENT integer]
```

Alter Tablespace

- The ALTER TABLESPACE allows you to change in tablespace.

Syntax:

```
ALTER TABLESPACE tablespace_name
  [ADD DATAFILE filename [SIZE integer [K | M]] [REUSE]]
  [Autoextend_Clause]
  [RENAME 'filename1' TO 'filename2']
  [DEFAULT STORAGE Storage_Clause
  [ONLINE] |OFFLINE]
  [PERMANENT | TEMPORARY]
  [BEGIN BACKUP | END BACKUP]
  [LOGGING | NOLOGGING]
  [MAXIMUM EXTENT integer]
```

Drop Tablespace

- The DROP TABLESPACE allows you to remove a tablespace from the database.

Syntax:

```
DROP TABLESPACE tablespace_name
  [INCLUDING CONTENTS [AND | KEEP] DATAFILES]
  [CASCADE CONSTRAINTS];
```

types of tablesplace (system, read only, temporary, tool, user, data and index, rollback)

1) **System :**
   - Every Oracle database contains a tablespace named SYSTEM, which Oracle creates automatically when the database is created.
   - The SYSTEM tablespace always contains the data dictionary tables for the entire database.

2) **Read only :**
   - The primary purpose of read-only tablespaces is to eliminate the need to perform backup and recovery of large, static portions of a database.
   - Oracle never updates the files of a read-only tablespace, and therefore the files can reside on read-only media, such as CD ROMs or WORM drives.

3) **Temporary :**
   - A temporary tablespace can be used only for sort segments.
   - Temporary tablespaces provide performance improvements when you have multiple sorts that are too large to fit into memory.
   - The sort segment of a given temporary tablespace is created at the time of the first sort operation.

4) Tool :
   - Contains files usually owned by SYSTEM but that apply to the Oracle developer's toolset; these files contain base information and details of forms, reports, and menus.

5) Users :
   - Default tablespaces in Oracle.
   - Tablespace in which users can create and destroy temporary nonapplication-related tables such as those used in SQL*REPORT for intermediate queries.

6) Rollback :

- Contains the private rollback segments; its size will depend on number of rollback segments and expected transaction size

Rollback Segment (Create, Alter) (System & Transaction RBS)

- A rollback segment is an Oracle database structure that stores undo information for transactions. Undo information is the original information that was changed during a transaction. It restores the changed database information back to what it was before a transaction changed it.
- When a transaction changes data, the Oracle server assigns the transaction to the next available segment and stores the undo information there.

What are the uses of Rollback Segment?

- Use the CREATE ROLLBACK SEGMENT statement to create a rollback segment, which is an object that Oracle Database uses to store data necessary to reverse, or undo, changes made by transactions.
- If the database has a locally managed SYSTEM tablespace, then rollback segments cannot create in any dictionary-managed tablespace.
- Instead, either use the automatic undo management feature or create locally managed tablespaces to hold the rollback segments.

Create Rollback Segment

Syntax:

```
CREATE [PUBLIC] ROLLBACK SEGMENT rbs-name
[TABLESPACE tbs-name]
STORAGE (INITIAL 20K NEXT 40K MINEXTENTS 2 MAXEXTENTS 50);
```

Alter Rollback Segment

Syntax:

```
ALTER ROLLBACK SEGMENT rbs-name STORAGE storage-option
```

Oracle Blocks

- Oracle data blocks are the smallest units of storage that Oracle can use or allocate.
- It contain header information concerning the block itself as well as the data.
- They are physically store on disk.
- Oracle stores and retrieves data on disk using data blocks.
- At time of creation of database the block size can be specified.

Import

- It allows you to restore the database information that held in previously created export files.
- Import loads data in the following order.
  - •Table definitions
  - •Table data
  - •Table indexes
  - •Triggers/index/constraint
- First of all new tables are created, then data is imported and indexes are built then triggers are enabled.
- There are 3 important modes:-
  1. User: - import all objects owned by user.
  2. Table:- import all tables owned by user.
  3. Full database: - import all objects of the database.

Syntax:

IMP <username/password> file=<filename> from user=<user name>
tables=<table name>

Export

- It is a logical backup of database.
- It copies the data and database to a binary OS file in a special format.
- We can take backup database while it is open and available for use.
- Data should not be changed while export take place
- There are 3 export modes.
    1. User: - export all objects owned by user.
    2. Table:- export all tables owned by user.
    3. Full database: - export all objects of the database.

SQL*Loader

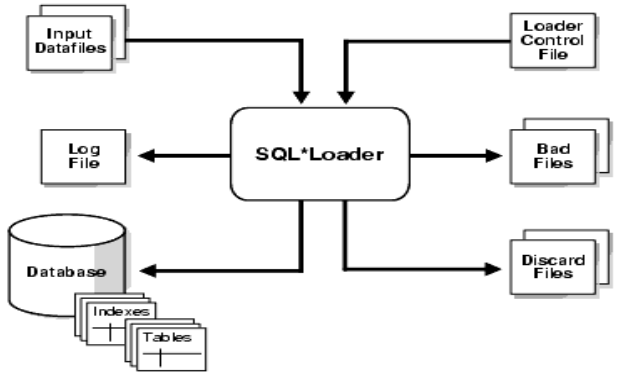
SQL*Loader loads data from external files into tables of an Oracle database. It has a powerful data parsing engine that puts little limitation on the format of the data in the datafile. You can use SQL*Loader to do the following:

- Load data across a network. This means that you can run the SQL*Loader client on a different system from the one that is running the SQL*Loader server.
- Load data from multiple datafiles during the same load session.
- Load data into multiple tables during the same load session.
- Specify the character set of the data.
- Selectively load data (you can load records based on the records' values).
- Manipulate the data before loading it, using SQL functions.
- Generate unique sequential key values in specified columns.
- Use the operating system's file system to access the datafiles.

Following are SQL *Loader file.

1. control file
2. log file
3. bad file
4. discard file
5. data file

A typical SQL*Loader session takes as input a control file, which controls the behavior of SQL*Loader, and one or more datafiles. The output of SQL*Loader is an Oracle database (where the data is loaded), a log file, a bad file, and potentially, a discard file. An example of the flow of a SQL*Loader session is shown in figure.



Oracle Scheduler Concepts

Oracle Database includes Oracle Scheduler, an enterprise job scheduler to help you simplify the scheduling of hundreds or even thousands of tasks. Oracle Scheduler (the Scheduler) is implemented by the procedures and functions in the DBMS_SCHEDULER PL/SQL package.

The Scheduler enables you to control when and where various computing tasks take place in the enterprise environment. The Scheduler helps you effectively manage and plan these tasks. By ensuring that many routine computing tasks occur without manual intervention, you can lower operating costs, implement more reliable routines, minimize human error, and shorten the time windows needed.

The Scheduler provides sophisticated, flexible enterprise scheduling functionality, which you can use to:

- Run **database program units**

  You can run program units, that is, PL/SQL anonymous blocks, PL/SQL stored procedures, and Java stored procedures on the local database or on one or more remote Oracle databases.

- Run external executables, (executables that are external to the database)

  You can run **external executables**, such as applications, shell scripts, and batch files, on the local system or on one or more remote systems. Remote systems do not require an Oracle Database installation; they require only a Scheduler agent. Scheduler agents are available for all platforms supported by Oracle Database and some additional platforms.

- Schedule job execution using the following methods:

  - **Time-based scheduling**

    You can schedule a job to run at a particular date and time, either once or on a repeating basis. You can define complex repeat intervals, such as "every Monday and Thursday at 3:00

a.m. except on public holidays" or "the last Wednesday of each business quarter." See "Creating, Running, and Managing Jobs" for more information.

- **Event-based scheduling**

  You can start jobs in response to system or business events. Your applications can detect events and then signal the Scheduler. Depending on the type of signal sent, the Scheduler starts a specific job. Examples of event-based scheduling include starting jobs when a file arrives on a system, when inventory falls below predetermined levels, or when a transaction fails. Beginning with Oracle Database 11g Release 2, a Scheduler object called a file watcher simplifies the task of configuring a job to start when a file arrives on a local or remote system. See "Using Events to Start Jobs" for more information.

- **Dependency scheduling**

  You can set the Scheduler to run tasks based on the outcome of one or more previous tasks. You can define complex dependency chains that include branching and nested chains. See "Creating and Managing Job Chains" for more information.

Basic Scheduler Concepts

The Scheduler offers a modular approach for managing tasks within the Oracle environment. Advantages of modularity include easier management of your database environment and reusability of scheduler objects when creating new tasks that are similar to existing tasks.

In the Scheduler, most components are database objects like a table, which enables you to use normal Oracle privileges.

The basic elements of the Scheduler are:

- Programs
- Schedules
- Jobs
- Events
- Chains

Advanced Scheduler Concepts

Many Scheduler capabilities enable database administrators to control more advanced aspects of scheduling. Typically, these topics are not as important for application developers.

This section discusses the following advanced topics

- Job Classes
- Windows
- Window Groups

Managing Resources with Oracle Database Resource Manager

Oracle Database Resource Manager (the Resource Manager) enables you to manage multiple workloads within a database that are contending for system and database resources.

About the Elements of Resource Manager

The elements of the Resource Manager include resource consumer groups, resource plans, and resource plan directives.

Element	Description
Resource consumer group	A group of sessions that are grouped together based on resource requirements. The Resource Manager allocates resources to resource consumer groups, not to individual sessions.
Resource plan	A container for directives that specify how resources are allocated to resource consumer groups. You specify how the database allocates resources by activating a specific resource plan.
Resource plan directive	Associates a resource consumer group with a particular plan and specifies how resources are to be allocated to that resource consumer group.

You use the DBMS_RESOURCE_MANAGER PL/SQL package to create and maintain these elements. The elements are stored in tables in the data dictionary. You can view information about them with data dictionary views.

Resource Manager Administration Privileges

- You must have the required privileges to administer the Resource Manager.
- You must have the system privilege ADMINISTER_RESOURCE_MANAGER to administer the Resource Manager. This privilege (with the ADMIN option) is granted to database administrators through the DBArole.
- Being an administrator for the Resource Manager enables you to execute all of the procedures in the DBMS_RESOURCE_MANAGERPL/SQL package.
- You may, as an administrator with the ADMIN option, choose to grant the administrative privilege to other users or roles. To do so, use the DBMS_RESOURCE_MANAGER_PRIVS PL/SQL package. The relevant package procedures are listed in the following table.

Procedure	Description
`GRANT_SYSTEM_PRIVILEGE`	Grants the `ADMINISTER_RESOURCE_MANAGER` system privilege to a user or role.
`REVOKE_SYSTEM_PRIVILEGE`	Revokes the `ADMINISTER_RESOURCE_MANAGER` system privilege from a user or role.

- The following PL/SQL block grants the administrative privilege to user HR, but does not grant HR the ADMIN option. Therefore, HR can execute all of the procedures in the DBMS_RESOURCE_MANAGER package, but HR cannot use the GRANT_SYSTEM_PRIVILEGE procedure to grant the administrative privilege to others.

```
BEGIN
  DBMS_RESOURCE_MANAGER_PRIVS.GRANT_SYSTEM_PRIVILEGE(
    GRANTEE_NAME   => 'HR',
    PRIVILEGE_NAME => 'ADMINISTER_RESOURCE_MANAGER',
    ADMIN_OPTION   => FALSE);
END;
```

- You can revoke this privilege using the REVOKE_SYSTEM_PRVILEGE procedure.

Note: The ADMINISTER_RESOURCE_MANAGER system privilege can only be granted or revoked using the DBMS_RESOURCE_MANAGER_PRIVS package. It cannot be granted or revoked through the SQL GRANT or REVOKE statements.