DATABASE LOGICAL STRUCTURE

(Cap. 4)

Every running Oracle database is associated with an Oracle instance. When a database is started on a database server (regardless of the type of computer), Oracle allocates a memory area called the System Global Area (SGA) and starts one or more Oracle processes. This combination of the SGA and the Oracle processes is called an **Oracle instance**. The memory and processes of an instance manage the associated database's data efficiently and serve the one or multiple users of the database.

A) Oracle Database Logical Structure

A logical structure hierarchy exists as follows:
- An Oracle database contains at least one tablespace.
- A tablespace contains one or more segments.
- A segment is made up of extents.
- An extent is made up of logical blocks.
- A block is the smallest unit for read and write operations.

The Oracle database architecture includes logical and physical structures database.

- The physical structure includes the control files, online redo log files, that make up the database.
- The logical structure includes tablespaces, segments, extents, and data.

The Oracle server enables fine-grained control of disk space use through logical storage structures, including segments, extents, and data blocks.

Tablespaces

The data in an Oracle database is stored in tablespaces.
- An Oracle database can be logically grouped into smaller logical areas of space known as tablespaces.
- A tablespace can belong to only one database at a time.
- Each tablespace consists of one or more operating system files, which are called data files.
- A tablespace may contain one or more segments.
- Tablespaces can be brought online while the database is running.
- Except for the `SYSTEM` tablespace or a tablespace with an active undo segment, tablespaces can be taken offline, leaving the database running.
- Tablespaces can be switched between read/write and read-only status.

Data Files (Not a logical structure)

- Each tablespace in an Oracle database consists of one or more files called data files.

These are physical structures that conform with the operating system on which the Oracle server is running.
- A data file can belong to only one tablespace.
- An Oracle server creates a data file for a tablespace by allocating the specified amount of disk space plus a small amount of overhead.
- The database administrator can change the size of a data file after its creation or can

specify that a data file should dynamically grow as objects in the tablespace grow.

Segments

- A segment is the space allocated for a specific logical storage structure within a tablespace.
- A tablespace may consist of one or more segments.
- A segment cannot span tablespaces; however, a segment can span multiple data files that belong to the same tablespace.
- Each segment is made up of one or more extents.

Extents

Space is allocated to a segment by extents.

- One or more extents make up a segment.

- When a segment is created, it consists of at least one extent.
- As the segment grows, extents are added to the segment.
- The DBA can manually add extents to a segment.

- An extent is a set of contiguous Oracle blocks.
- An extent cannot span data files, and therefore, it must exist in one datafile.

Data Blocks

The Oracle server manages the storage space in the data files in units called Oracle blocks or data blocks.

- At the finest level of granularity, the data in an Oracle database is stored in data blocks.
- Oracle data blocks are the smallest units of storage that the Oracle server can allocate, read, or write.
- One data block corresponds to one or more operating system blocks allocated from an existing data file.
- The standard data block size for an Oracle database is specified by the `DB_BLOCK_SIZE` initialization parameter when the database is created.
- The data block size should be a multiple of the operating system block size to avoid unnecessary I/O.
- The maximum data block size is dependent on the operating system.

B) Creating a Database Manually

- Create the initialization parameter file.

The initialization parameter file is created using the sample `init.ora` file installed during the installation process. Copy the sample `init.ora` and name it `initSID.ora`. Make modifications to the file specific to the needs of the database you will be creating. If an SPFILE is to be used, the PFILE must be created first. Refer to the “Managing an Oracle Instance” lesson for instructions on how to create a database specific `initSID.ora` file and an SPFILE.

- Start the instance in `NOMOUNT`.

Connect as user `SYS` with `SYSDBA` privilege. The database must be placed in the `NOMOUNT` state in order to create a database. Refer to the “Managing an Oracle Instance” lesson for directions on how to place the database in a `NOMOUNT` state.

• Create and execute the `CREATE DATABASE` command.

- Create an SQL script that contains the `CREATE DATABASE` command. Connect to SQL*Plus as the `SYS` user with the `SYSDBA` privilege. With the database in `NOMOUNT` state, execute the script.

- The `CREATE DATABASE` command will be dramatically simplified if the database being created is to use Oracle Managed Files (OMF) to manage the operating system files. Refer to the “Managing an Oracle Instance” lesson for information regarding OMF.

• Run scripts.

- Two scripts `catalog.sql` and `catproc.sql` must be run after the database is created. Both scripts must be run as the user `SYS` with `SYSDBA` privilege. Before executing the scripts the database must be placed in the `OPEN` state.

- `catalog.sql`: Creates the views on the base tables and on the dynamic performance views, and their synonyms. It starts other scripts that create objects for:

· Basic PL/SQL environment, including declarations for PL/SQL data types, predefined exceptions, built-in procedures and functions, SQL operations

· Auditing

· Import/Export

· SQL*Loader

· Installed options

Examples:

1)

```
SQL> connect sys as sysdba
SQL> startup nomount
ORACLE instance started.
Total System Global Area 21790532 bytes
Fixed Size 278340 bytes
Variable Size 16777216 bytes
Database Buffers 4194304 bytes
Redo Buffers 540672 bytes

SQL> CREATE DATABASE db01
      LOGFILE
      GROUP 1 ('$HOME/ORADATA/u03/log_01_01_db01.rdo') SIZE 1M,
      GROUP 2 ('$HOME/ORADATA/u03/log_02_01_db01.rdo') SIZE 1M
      DATAFILE '$HOME/ORADATA/u01/system_01_db01.dbf' SIZE 1M
      AUTOEXTEND ON NEXT 5M MAXSIZE 150M
      DEFAULT TEMPORARY TABLESPACE temp
      TEMPFILE '$HOME/ORADATA/u02/temp_01_db01.dbf' SIZE 1M
      AUTOEXTEND ON NEXT 5M MAXSIZE 1M
      CHARACTER SET WE8ISO8859P1
      NATIONAL CHARACTER SET AL16UTF16
      /
```

Statement processed.

2)

```
SQL>  CREATE DATABASE user01
      USER SYS IDENTIFIED BY ORACLE
      USER SYSTEM IDENTIFIED BY MANAGER
      CONTROLFILE REUSE
      LOGFILE
      GROUP 1 ('E:/student/redo01.log') SIZE 100M,
      GROUP 2 ('E:/student/redo02.log') SIZE 100M,
      GROUP 3 ('E:/student/redo03.log') SIZE 100M
      MAXLOGFILES 5
      MAXLOGMEMBERS 5
      MAXLOGHISTORY 1
      MAXDATAFILES 100
      MAXINSTANCES 1
      ARCHIVELOG
      FORCE LOGGING
      CHARACTER SET US7ASCII
      NATIONAL CHARACTER SET AL16UTF16
      /
```

3)

```
SQL>  CREATE DATABASE DBA01
      LOGFILE
      GROUP 1 (’/$HOME/ORADATA/u01/redo01.log’) SIZE 100M,
      GROUP 2 (’/$HOME/ORADATA/u02/redo02.log’) SIZE 100M,
      MAXLOGFILES 5
      MAXLOGMEMBERS 5
      MAXLOGHISTORY 1
      MAXDATAFILES 100
      MAXINSTANCES 1
      DATAFILE ’/$HOME/ORADATA/u01/system01.dbf’ SIZE 325M
      UNDO TABLESPACE undotbs
      DATAFILE ’/$HOME/ORADATA/u02/undotbs01.dbf’ SIZE 200
      DEFAULT TEMPORARY TABLESPACE temp
      TEMPFILE ’/$HOME/ORADATA/u03/temp01.dbf’ SIZE 4M
      CHARACTER SET US7ASCII
      /
```