Oracle Database Admin Guide 12cR1

# Oracle Database Architecture

## Tổng quan về các thành phần trong kiến trúc Oracle

**Oracle server :**

* + Là tập hợp những file, những tiến trình và cấu trúc bộ nhớ
  + Oracle server bao gồm Oracle Instance và Oracle Database

**Oracle Instance**

* + Là tập hợp của các tiến trình ngầm (background processes) và cấu trúc bộ nhớ (memory structure). Một Instance được khởi động để truy cập dữ liệu trong Oracle Database.
  + Định danh bởi: Oracle System Identifier **(SID)**
  + Mỗi khi một Instance được khởi động, một System Global Area (SGA) được cấp phát và các tiến trình ngầm của Oracle cũng sẽ được khởi động. Các tiến trình ngầm này sẽ thực hiện tác vụ vào/ra và quản lý các tiến trình khác của Oracle nhằm cung cấp khả năng chạy song song để thi hành tốt hơn và tin cậy hơn:
    - Pmon
    - Smon
    - DBWR
    - LogWR

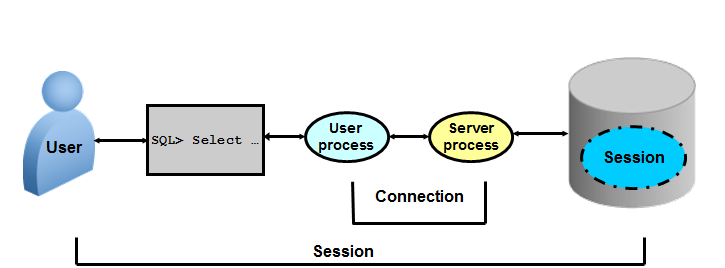
**Oracle Database**

* + Là một tập hợp các file hệ thống hay còn gọi là các file Database, cung cấp các thông tin về những thiết bị lưu trữ vật lý và thông tin và Database.
  + Định danh bởi: **DB\_NAME** và **DB\_ID**
  + Những file Database được sử dụng để đảm bảo rằng dữ liệu được lưu giữ ở trạng thái nhất quán, và có thể được khôi phục lại tại thời điểm một Instance bị lỗi:
    - Spfile : memory, vị trí control file, các setting
    - Control file : vị trí datafile,redo log thông tin backup, SCN
    - Datafile
    - Redolog file
    - Archivelog file

**Oracle Database Instance Configurations**

* + Có 2 dạng cấu hình của database instance:
    - Single instance : 1 database – 1 instance
    - Rac : 1 database – n instance

**Connecting to the Database Instance**

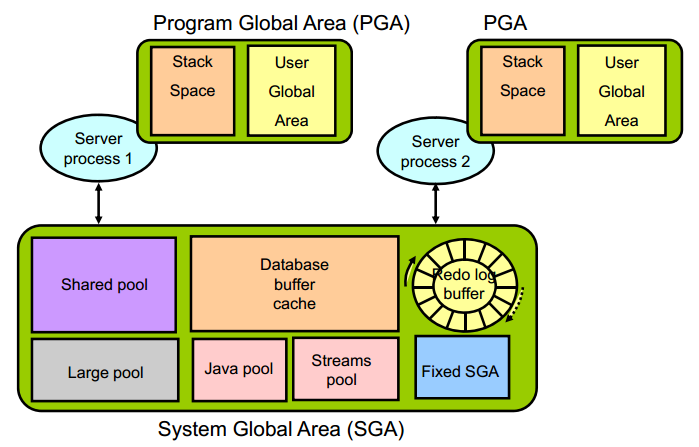


* + Connection: Communication between a user process and an instance
  + Session: Specific connection of a user to an instance through a user process
  + Khi user muốn thao tác với database thì phải tạo ra các yêu cầu ( user process), các yêu cầu này được gửi tới server và server sẽ thực hiện các yêu cầu này để thao tác với db.

## Kiến trúc bộ nhớ

Có hai cấu trúc bộ nhớ cơ bản trong một Instance:

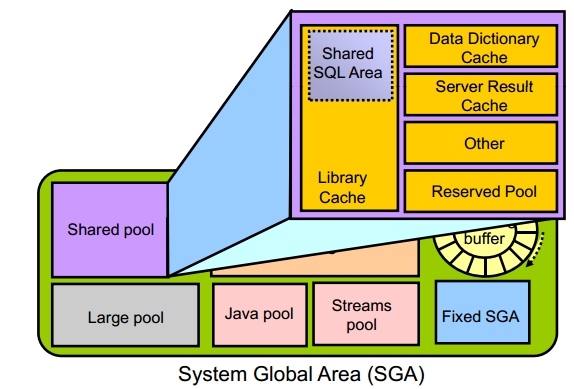
* + System Global Area (SGA):
    - Là vùng bộ nhớ chia sẻ được sử dụng để lưu trữ dữ liệu và các thông tin điều khiển của Oracle server. Được chỉ định khi một Instance được khởi động, và là thành phần cơ bản của một Oralce Instance.
    - The SGA được chia sẻ dung chung bởi server và các tiến trình nền processes
  + Program Global Area (PGA):
    - Là một phần cấu trúc bộ nhớ lưu trữ dành cho 1 User process kết nối tới 1 Instance bao gồm dữ liệu và thông tin điều khiển cho một Server hoặc một Background process.
    - Được khởi tạo Được chỉ định khi một Server Process được khởi động.



### Kiến trúc SGA

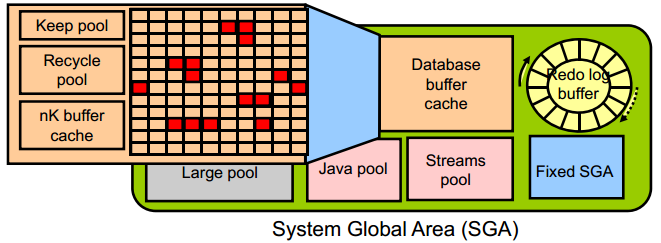
**Shared pool**

* + **Library cache**: Lưu trữ những định nghĩa về những đoạn lệnh SQL và PL/SQL vừa được thực thi gần đây nhất theo thuật giải Least Recently Used (LRU). Library cache bao gồm 2 cấu trúc là Shared SQL area và Shared PL/SQL area. Kích thước của vùng này được xác định bởi Shared pool sizing.
  + **Data dictionary cache**: Thu thập những định nghĩa được dùng gần đây nhất trên cơ sỡ dữ liệu bao gồm các thông tin về Database file, tables, indexes, columns, user, privileges,…Trong quá trình phân tích cú pháp đoạn lệnh, Server Process sẽ đọc các thông tin định nghĩa ở Data dictionary cache để lấy tên các đối tượng, xác nhận truy cập,…Kích thước của vùng này được xác định bởi Shared pool sizing.
  + **Shared SQL area:** Oracle Database represents each SQL statement that it runs with a shared SQL area (as well as a private SQL area kept in the PGA). When a new SQL statement is parsed, Oracle Database allocates memory from the shared pool to store in the shared SQL area. The size of this memory depends on the complexity of the statement
  + **The server result cache** contains the SQL query result cache and PL/SQL function result cache, which share the same infrastructure. The server result cache contains result sets, not data blocks
  + **The reserved pool** is a memory area in the shared pool that Oracle Database can use to allocate large contiguous chunks of memory



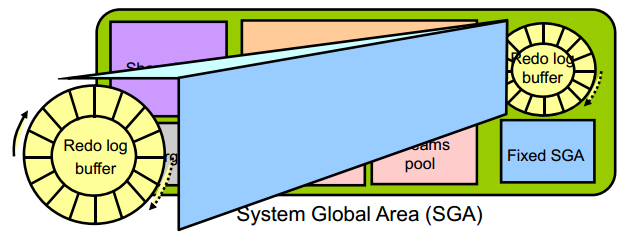
**Database Buffer Cache**

* + Lưu trữ những bản copy của **Block** dữ liệu đã được đọc từ **Data File.** Khi một đoạn SQL được thực thi, thì **Server Process** sẽ đọc các thông tin từ **Database buffer cache** để lấy các **block** dữ liệu cần thiết, điều này giúp cho tốc độ hoạt động của hệ thống sẽ cao hơn vì đọc trên cache sẽ nhanh hơn là đọc trên đĩa cứng . Nếu các block dữ liệu không có trong Database buffer cache thì Server process mới đọc dữ liệu từ data file. Database buffer cache cũng sử dụng thuật giải LRU như ở Shared Pool.
  + The **keep buffer pool** and the **recycle buffer pool** are used for specialized buffer pool tuning. The keep buffer pool is designed to retain buffers in memory longer than the LRU would normally retain them. The recycle buffer pool is designed to flush buffers from memory faster than the LRU normally would.
  + **nK buffer caches,** additional buffer can be configured to hold blocks of a size that is different from the default block size.



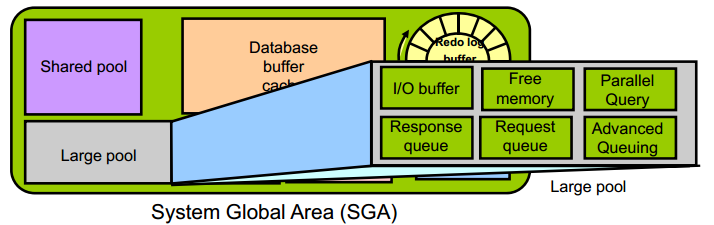
**Redo log buffer**

* + Redo entries contain the information necessary to reconstruct (or redo) changes that are made to the database by DML, DDL, or internal operations. Redo entries are used for database recovery if necessary



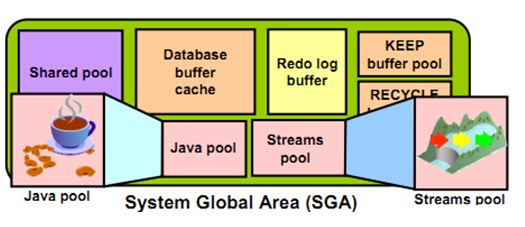
**Large Pool**

* + The database administrator can configure an optional memory area called the large pool to provide large memory allocations for:
    - Session memory for the shared server and the Oracle XA interface (used where transactions interact with multiple databases)
    - I/O server processes
    - Oracle Database backup and restore operations
    - Parallel Query operations
    - Advanced Queuing memory table storage



**Java Pool và Streams Pool**

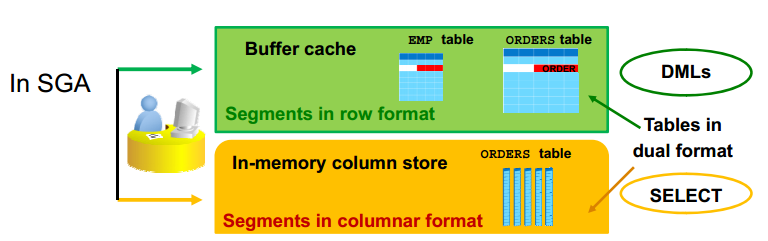
* + Java Pool là vùng dùng cho các Procedure viết bằng Java.
  + Stream Pool dùng cho Oracle Stream.



**In-Memory Column Store**

New feature

* + The In-Memory Column Store feature enables objects (tables, partitions, and other types) to be stored in memory in a new format known as the columnar format. This format enables scans, joins, and aggregates to perform much faster than the traditional on-disk format, thus providing fast reporting and DML performance for both OLTP and DW environments.
  + This is particularly useful for analytic applications that operate on **few columns returning many rows** rather than for OLTP that operates on few rows returning many columns. **The DBA must define** the segments that are to be populated into the in-memory column store (IM column store), such as hot tables, partitions, and more precisely the more frequently accessed columns.
  + The in-memory columnar format does not replace the on-disk or buffer cache format It is a consistent copy of a table or of some columns of a table converted to the new columnar format that is independent of the disk format and only available in memory. Because of this independence, applications are able to transparently use this option without any changes. For the data to be converted into the new columnar format, a new pool is requested in the SGA. The pool is the **IM column store**.
  + There are three main advantages:
    - Queries run a lot faster: All data can be populated in memory in a compressed columnar format. No index is required and used. Queries run at least 100 times faster than when fetching , p data from the buffer cache, thanks to the columnar compressed format.
    - DMLs are faster: Analytics indexes can be eliminated by being replaced by scans of the IM column store representation of the table.
    - Arbitrary ad hoc queries run with good performance, because the table behaves as if all columns are indexed.

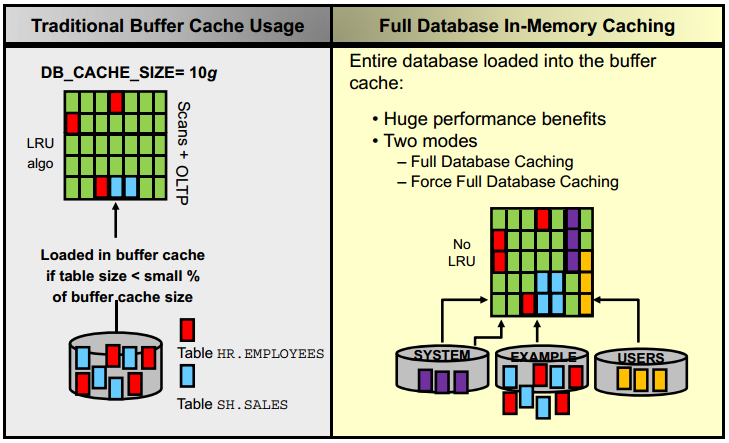


**Full Database In-Memory Caching**

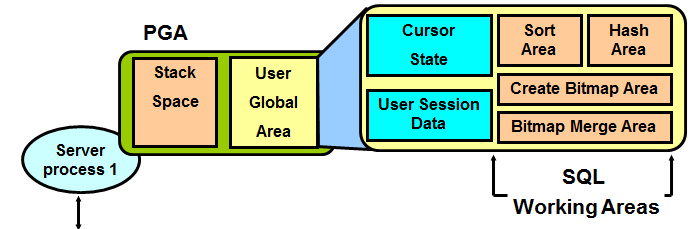
New feature

The Full Database In-memory Caching feature enables an entire database to be cached in memory when the database size (sum of all data files, SYSTEM tablespace LOB CACHE files , LOB CACHE files minus SYSAUX, TEMP) is smaller than the buffer cache size. Caching and running a database from memory leads to huge performance benefits. Two modes can be used:

* + Full Database Caching: Implicit default and automatic mode in which an internal calculation determines if the database can be fully cached for an instance. NOCACHE LOBs are not cached in Full Database Caching but in Force Full Database Caching mode even NOCACHE LOBs are cached.
  + Force Full Database Caching: Neither Full Database Caching nor Force Full Database Caching forces or prefetches data into memory. Workload must access the data first for them to be cached. It considers the entire database as eligible to be completely cached in the buffer cache. This mode requires the DBA to execute the *ALTER DATABASE FORCE FULL DATABASE CACHING* command. This mode takes precedence over Full Database Caching mode. To revert to traditional caching, use the *ALTER DATABASE NO FORCE FULL DATABASE CACHING* command



### Kiến trúc PGA



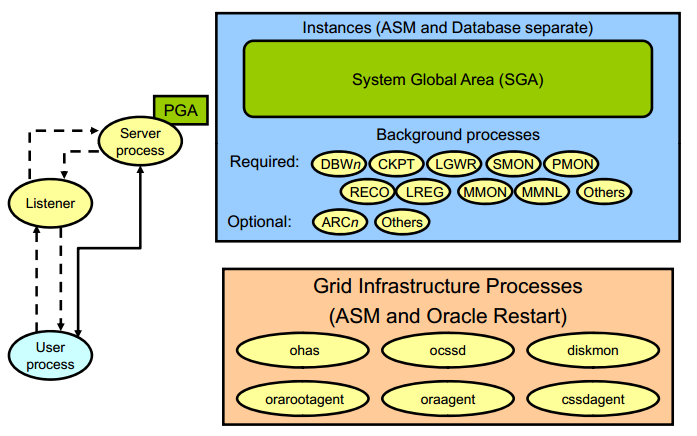
**Program Global Area (PGA)** là vùng nhớ riêng bao gồm dữ liệu và thông tin điều khiển của một Server Process. Mỗi Server Process đều có một PGA riêng biệt. PGA bao gồm:

* + Stack space : lưu trữ các biến và các mảng được xử lý trên PGA
  + User Global Area (UGA):
    - Cursor State : lưu trữ thông tin con trỏ
    - User session data : Lưu trữ thông tin điều khiển của 1 session
    - SQL working Areas : thực thi các câu lệnh truy vấn SQL :
      * Hash area: kết nối giữa các bảng bằng hash
      * Create bitmap area: sử dụng để tạo index bitmap
      * Bitmap merge area: sử dụng để tạo các kế hoạch cho index bitmap
      * Sort area: sử dụng cho các hàm gom nhóm dữ liệu như: ORDER BY and GROUP BY

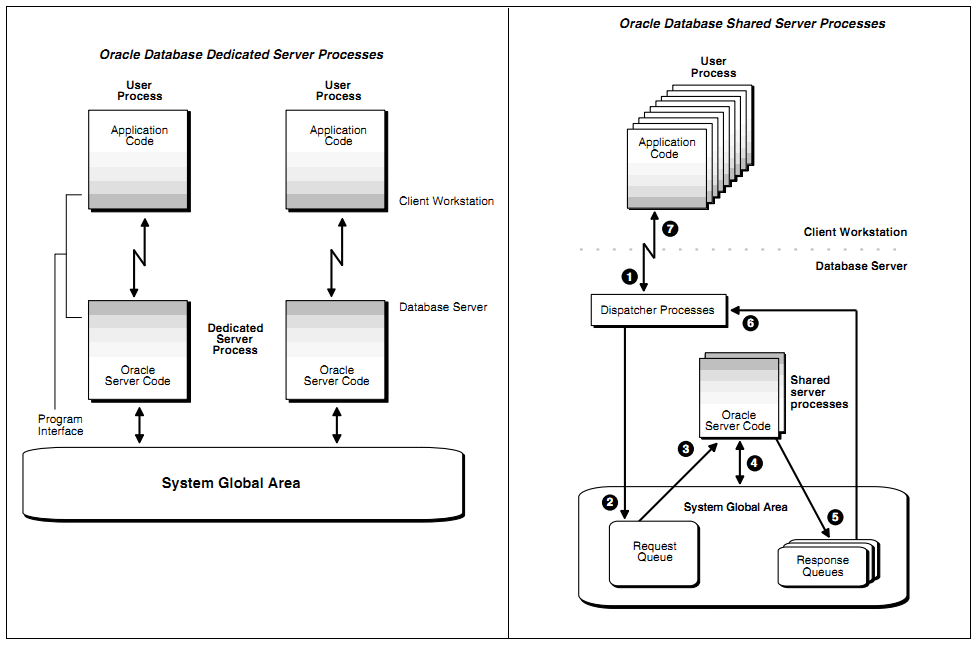
## Kiến trúc Processes

The processes in an Oracle Database system can be divided into three major groups:

* + User processes that run the application or Oracle tool code
  + Oracle Database processes that run the Oracle Database server code (including server processes and background processes)
    - 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
    - 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.
  + Oracle daemons and application processes not specific to a single database
    - Networking listeners
    - Grid Infrastructure daemons



### Server process



**Dedicated Server Processes**

Khái niệm:

* Dedicated server process: một server process chỉ phục vụ một user process

Sử dụng khi:

* Thực thi các job lớn ( các job rất ít, hoặc không có thời gian chờ đợi...)
* Sử dụng (RMAN) để back up, restore, hoặc recover database

**Shared Server Processes**

Khái niệm:

* Shared server process: một server process có thê phục vụ nhiều user process
* Sử dụng 1 bộ điều phối để quản lý tài nguyên ( **DISPATCHERS** )

Sử dụng khi:

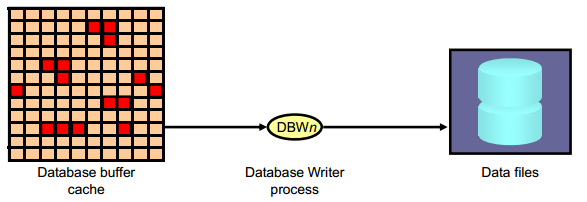
* Có rất nhiều user process kết nối tới DB và các server này tương tự nhau và có thể chờ đợi được ( đăng kí vé ....)

### Background process

Background process (các tiến trình nền) thực hiện các chức năng thay cho lời gọi tiến trình xử lý tương ứng. Nó điều khiển vào ra, cung cấp các cơ chế xử lý song song nâng cao hiệu quả và độ tin cậy. Tùy theo từng cấu hình mà Oracle instance có các Background process như:

* + Database Writer process (DBWn)
  + Log Writer process (LGWR)
  + Checkpoint process (CKPT)
  + System monitor process (SMON)
  + Process monitor process (PMON)
  + Recoverer process (RECO)
  + Listener registration process (LREG)
  + Manageability monitor process (MMON)
  + Manageability monitor lite process (MMNL)
  + Job queue coordinator (CJQ0)
  + Job slave processes (Jnnn)
  + Archiver processes (ARCn)
  + Queue monitor processes (QMNn)

**Database Writer (DBWn)**



The Database Writer process (DBWn) writes the contents of buffers to data files.

* + The DBWn processes are responsible for writing modified (dirty) buffers in the database buffer cache to disk.
  + The *DB\_WRITER\_PROCESSES* initialization parameter specifies the number of DBWn processes. The maximum number of Database Writer processes is 100. If it is not specified by the user during startup, Oracle Database determines how to set *DB\_WRITER\_PROCESSES* based on the number of CPUs and processor groups.

The DBWn process writes dirty buffers to disk under the following conditions:

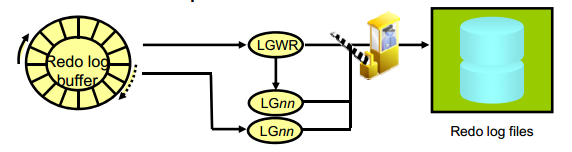
* + When a server process cannot find a clean reusable buffer after scanning a threshold

number of buffers, it signals DBWn to write. DBWn writes dirty buffers to disk asynchronously while performing other processing.

* + DBWn writesbuffers to advance the checkpoint, which is the position in the redo thread (log) from which instance recovery begins. This log position is determined by the oldest dirty buffer in the buffer cache.

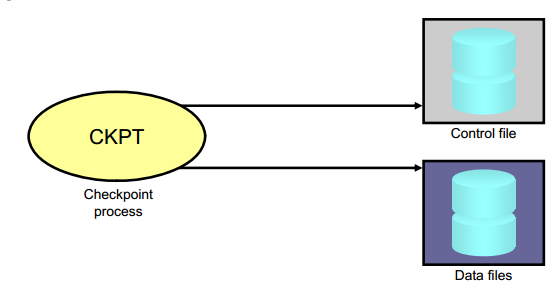
In all cases, DBWn performs batched (multiblock) writes to improve efficiency. The number of blocks written in a multiblock write varies by operating system

**Log Writer Process (LGWR)**



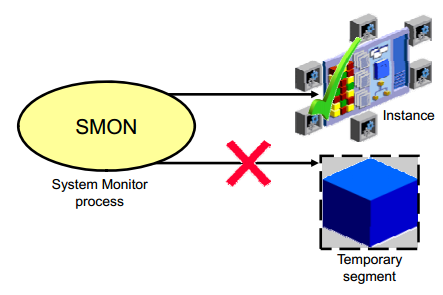
* + The Log Writer process (LGWR) is responsible for redo log buffer management by writing the redo log buffer entries to a redo log file on disk. LGWR writes all redo entries that have been copied into the buffer since the last time it wrote.
  + LGWR starts and coordinates multiple helper processes that concurrently perform some of the work. LGWR handles the operations that are very fast, or must be coordinated, and delegates operations to the LGnn that could benefit from concurrent operations, primarily writing the redo from the log buffer to the redo log file and posting the completed write to the foreground process that is waiting.
  + Writes the redo log buffer to a redo log file on disk
    - When a user process commits a transaction
    - When an online redo log switch occurs
    - When the redo log buffer is one-third full or contains 1 MB of buffered data
    - Before a DBWn process writes modified buffers to disk
    - When three seconds have passed since the last write
  + Serves as coordinator of LGnn processes and ensures correct order for operations that must be ordered

**Checkpoint Process (CKPT)**



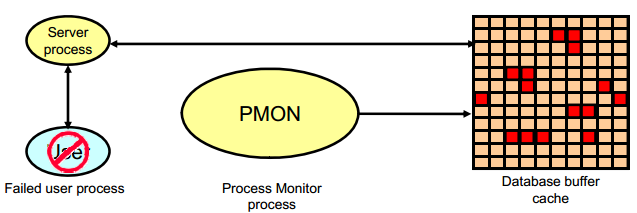
* + A checkpoint is a data structure that defines a system change number (SCN) in the redo thread of a database. Checkpoints are recorded in the control file and in each data file header. They are a crucial element of recovery.
  + When a checkpoint occurs, Oracle Database must update the headers of all data files to record the details of the checkpoint. This is done by the CKPT process. And Signals DBWn to write blocks to disk

**System Monitor Process (SMON)**



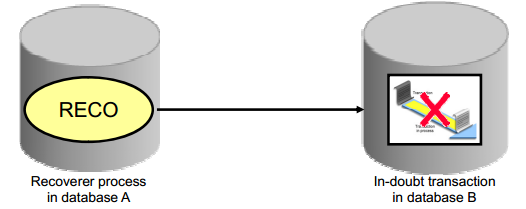
* + The System Monitor process (SMON) performs recovery at instance startup if necessary. SMON is also responsible for cleaning up temporary segments that are no longer in use. If any terminated transactions were skipped during instance recovery because of file-read or offline errors, SMON recovers them when the tablespace or file is brought back online

**Process Monitor Process (PMON)**



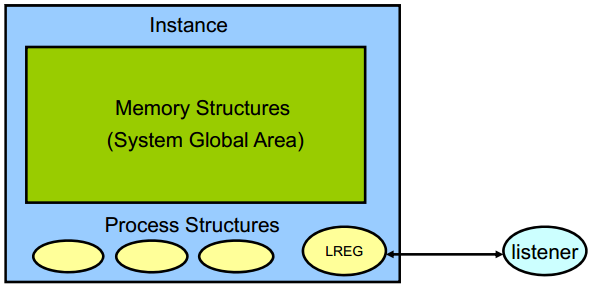
* + The Process Monitor process (PMON) performs process recovery when a user process fails. PMON is responsible for cleaning up the database buffer cache and freeing resources that the user process was using
  + PMON periodically checks the status of dispatcher and server processes, and restarts any that have stopped running (but not any that Oracle Database has terminated intentionally).

**Recoverer Process (RECO)**



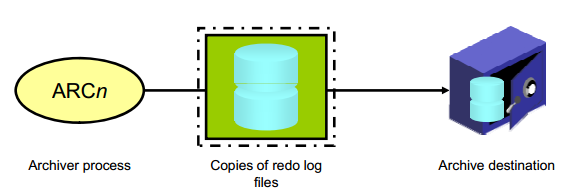
* + The Recoverer process (RECO) is a background process that is used with the distribute database configuration that automatically resolves failures involving distributed transactions
  + Automatically connects to other databases involved in indoubt distributed transactions
  + Automatically resolves all in-doubt transactions
  + Removes any rows that correspond to in-doubt transactions

**Listener Registration Process (LREG)**



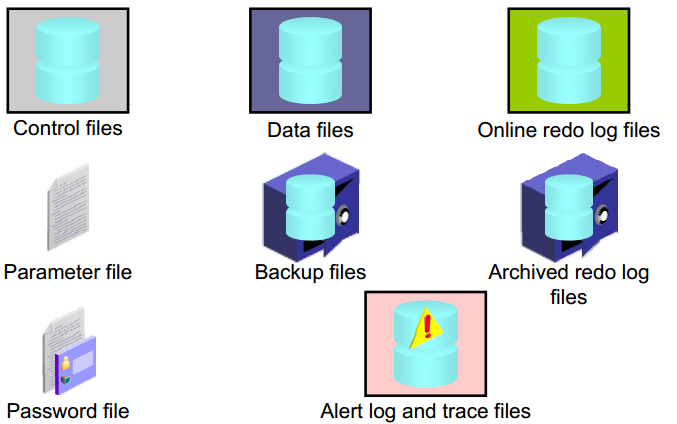
* + The Listener Registration process, LREG, registers information about the database instance and dispatcher processes with the Oracle Net Listener. LREG provides the listener with the following information:
    - Names of the database services
    - Name of the database instance associated with the services and its current and maximum load
    - Service handlers available for the instance, including their type, protocol addresses, and current and maximum load

**Archiver Processes (ARC*n*)**



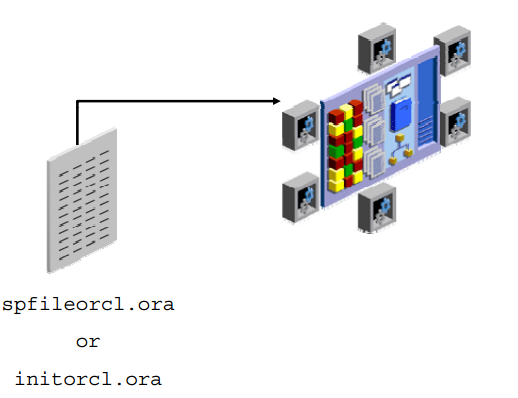
* + The Archiver processes (ARCn) copy redo log files to a designated storage device after a log switch has occurred. ARCn processes are present only when the database is in ARCHIVELOG mode and automatic archiving is enabled.
  + Can collect transaction redo data and transmit that data to standby destinations

## **Kiến trúc lưu trữ của Oracle**



### Parameter file

* + File chứa những thông tin cấu hình của 1 instance và địa chỉ của controlfile.
  + Ví dụ như: Các vùng nhớ khởi tạo được cấp phát bao nhiêu GB, các control\_file nằm ở đâu, bao nhiêu process, session v.v…
  + Parameter có 2 loại là SPFile (file Binary) và Pfile (file text), nhiệm vụ của 2 file này là giống nhau. Mặc định, lúc khởi động DB, Oracle sẽ ưu tiên sử dụng spfile trước, nếu không có thì sẽ chuyển sang dùng pfile ( pfile có thể lưu trữ địa chỉ của spfile)



When you start the instance, an initialization parameter file is read. There are two types of parameter files.

* Server parameter file (SPFILE): This is the preferred type of initialization parameter file. It is a binary file that can be written to and read by the database server and must not be edited manually. It resides on the server on which the Oracle instance is executing; it is persistent across shutdown and startup. The default name of this file, which is automatically sought at startup, is spfile<SID>.ora.
* Text initialization parameter file: This type of initialization parameter file can be read by the database server, but it is not written to by the server. The initialization parameter settings must be set and changed manually by using a text editor so that they are persistent across shutdown and startup. The default name of this file (which is automatically sought at startup if an SPFILE is not found) is init<SID>.ora.

In the majority of cases, it is necessary to set and tune only the 30 or so basic parameters to get reasonable performance from the database. In rare situations, modification of the advanced parameters may be needed to achieve optimal performance. There are more than 300 advanced parameters.

* **CONTROL\_FILES** **parameter**: Specifies one or more control file names. Oracle strongly recommends that you multiplex and mirror control files. Range of values: from one to eight file names (with path names). Default value: OS dependent.
* **DB\_FILES** **parameter**: Specifies the maximum number of database files that can be opened for this database. Range of values: OS dependent. Default value: 200.
* **PROCESSES parameter**: Specifies the maximum number of OS user processes that can simultaneously connect to an Oracle server. This value should allow for all background processes and user processes. Range of values: from 6 to an OS-dependent value. Default value: Dynamic and dependent on the number of CPUs.
* **DB BLOCK SIZE \_BLOCK\_SIZE parameter**: Specifies the size (in bytes) of an Oracle database block. This value is set at database creation and cannot be subsequently changed. This specifies the standard block size for the database. All tablespaces will use this size by default. Range of values: 2048 to 32768 (OS-dependent). Default value: 8192.
* **DB\_CACHE\_SIZE parameter**: Specifies the size of the default buffer pool. Range of values: At least 4 MB times the number of CPUs (smaller values are automatically rounded up to this value). Default value: 0 if SGA\_TARGET is set, otherwise the larger of 48 MB or (4 MB\*CPU\_COUNT)
* **PGA\_AGGREGATE\_TARGET parameter**: Specifies the amount of Program Global Area (PGA) memory available to all server processes attached to the instance. This memory does not reside in the System Global Area (SGA). The database uses this parameter as a target amount of PGA memory to use. When setting this parameter, subtract the SGA from the total memory on the system that is available to the Oracle instance. The minimum value is 10 MB and the maximum value is (4096 GB – 1). The default is 10 MB or 20% of the size of the SGA, whichever is greater.
* **SHARED\_POOL\_SIZE parameter**: Specifies the size of the shared pool in bytes. The shared pool contains objects such as shared cursors, stored procedures, control structures, and parallel execution message buffers. Range of values: OS-dependent. Default value: 0 if SGA\_TARGET is set, otherwise 128 MB if 64-bit; 48 MB if 32-bit.
* **UNDO\_MANAGEMENT parameter**: Specifies the undo space management mode that the system should use. When set to AUTO, the instance is started in automatic undo management mode. Otherwise, it is started in rollback undo mode. In rollback undo mode, undo space is allocated as rollback segments. In automatic undo mode, undo space is allocated as undo tablespaces. Range of values: AUTO or MANUAL. If the UNDO MANAGEMENT parameter is omitted when the instance is started the default value AUTO is used.

### Control Files

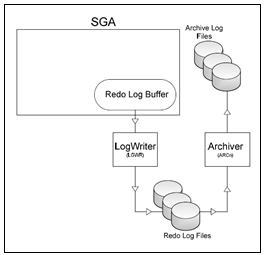
* + Mỗi Oracle database đều có ít nhất một control file. Control file chứa các mục thông tin quy định cấu trúc vật lý của database như:
    - Tên của database.
    - Tên và nơi lưu trữ các datafiles, redo log files.
    - Các thông tin về backup của database
  + Control file cũng được sử dụng đến khi thực hiện khôi phục lại dữ liệu.
  + The control file is critical to the database. Without the control file, the database cannot be opened

### Datafiles

* + Mỗi một Oracle database đều có thể có một hay nhiều datafiles. Các database datafiles chứa toàn bộ dữ liệu trong database. Các dữ liệu thuộc cấu trúc logic của database như tables hay indexes đều được lưu trữ dưới dạng vật lý trong các datafiles của database.
  + Một số tính chất của datafiles:
    - Mỗi datafile chỉ có thể được sử dụng trong một database.
    - Datafiles cũng còn có một số tính chất cho phép tự động mở rộng kích thước mỗi khi database hết chỗ lưu trữ dữ liệu.
    - Một hay nhiều datafiles tạo nên một đơn vị lưu trữ logic của database gọi là tablespace. Một datafile chỉ thuộc về một tablespace.

### Redo Log Files - Archivelog

* + REDO LOG file là một phần quan trọng của quá trình Oracle recovery. Được sử dụng để recover database khi bị crash.
  + Chức năng chính của redo log là ghi lại tất cả các thay đổi đối với dữ liệu trong database. Redo log files được sử dụng để bảo vệ database khỏi những sự cố. Oracle cho phép sử dụng cùng một lúc nhiều redo log gọi là multiplexed redo log để cùng lưu trữ các bản sao của redo log trên các ổ đĩa khác nhau
  + Khi mỗi Oracle block thay đổi, Oracle tạo ra các vector thay đổi cho nó. Mỗi vetor này được cọi là REDO entry hoặc REDO records. Những sự thay đổi này sẽ được server process ghi vào redo log buffer trên PGA. Redo log buffer ghi xuống online redo log vào 1 thời điểm nào đó bởi LGWR
  + Redolog thường gồm ít nhất 2 đơn vị, ghi lại những thao tác tác động lên DB, khi 1 redolog đầy sẽ switch sang redo log bên cạnh. Khi switch sang 1 redolog đã có dữ liệu, dữ liệu này sẽ được ghi xuống archivelog để lưu trữ



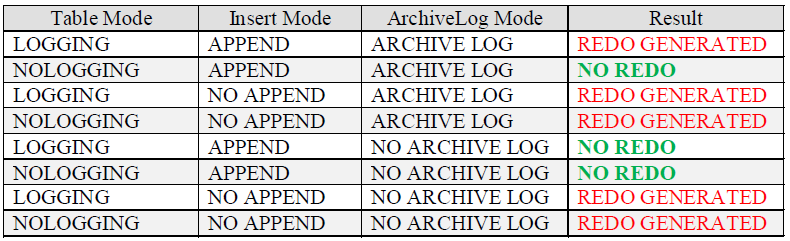
**Logging và nologging**

Tuy việc tạo nên REDO LOG file là một phần quan trọng của quá trình Oracle recovery nhưng vẫn có khả năng có những redo log file được tạo nên từ những thao tác không mong muốn

Luật quan trọng nhất với dữ liệu là không được phép đặt nó trong trình trạng không thể recovery. Nhưng điều luật này đồng nghĩa với việc không thể sử dụng các option để rút ngắn thời gian hoặc gia tăng hiệu năng của hệ thống

Oracle cung cấp cho người sử dụng khả năng giới hạn ghi log vào bảng và indexes bằng NOLOGGING mode.

* + NOLOGGING có tác động mạnh đến việc recovery.
  + Đặc điểm chính cần chú ý:
    - NOLOGGING được thiết kế để insert 1 lượng dữ liệu lớn(bulk data)
    - Nếu dữ liệu không được ghi log thì nó không thể được recover, dữ liệu nên được backup sau khi chỉnh sửa.
  + Các trường hợp:



Trong một số trường hợp khi ở chế độ NOLOGGING những câu lệnh vẫn ghi log:

* + CREATE TABLE ... AS SELECT
  + CREATE INDEX.
  + UPDATE/INSERRT/DELETE

**Password file**:

* + Allows users using the SYSDBA, SYSOPER, SYSBACKUP, SYSDG, SYSKM, and SYSASM roles to connect remotely to the instance and perform administrative tasks

**Backup files** :

* + Are use for database recovery. You typically restore a backup file when a media failure or user error has damaged or deleted the original file

**Archived redo 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.

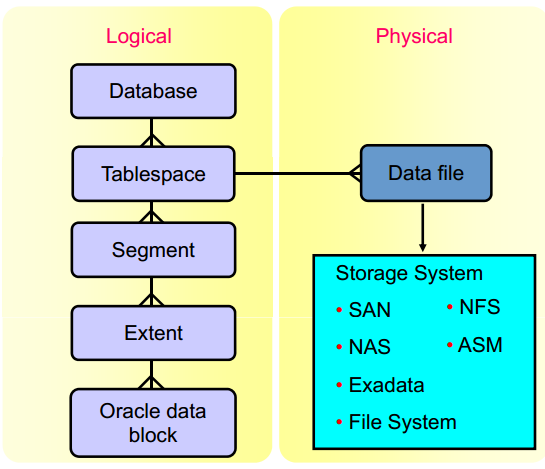
**Trace files**:

* + Each server and backg p round 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 file**:

* + These are special trace entries. The alert log of a database is a chronological log of messages and errors. Oracle recommends that you review the alert log periodically.

## Cấu trúc logic databse và các vấn đề liên quan



Each database is logically divided into two or more tablespaces. One or more data files are explicitly created for each tablespace to physically store the data of all segments in a tablespace. If it is a TEMPORARY tablespace, it has a temporary file instead of a data file. A tablespace’s data file can be physically stored on any supported storage technology

### Tablespaces

* + Một database có thể được phân chia về mặt logic thành các đơn vị gọi là các tablespaces. Tablespaces thường bao gồm một nhóm các thành phần( logic )có quan hệ logic với nhau.
  + Mỗi tablespace có thể được tạo nên, bởi một hoặc nhiều datafiles( physic). Kích thước của database cũng có thể xác định được bằng tổng kích thước của các tablespaces của nó.
  + **Phân loại:**
    - Tablespace SYSTEM
      * Sẽ tự động được tạo khi database tạo.
      * Có trong tất cả các database dùng cho hoạt động của database
      * Chứa thông tin về các data dictionary view,các định nghĩa của store procedure,pakage và các database trigger.
      * Chứa SYSTEM **Undo segment**.
      * Chứa SYSTEM **rollback segment**
  + Non – System Tablespace
    - Có thể lưu trữ rollback segment, temporary segment, data segment, index segment
    - Giúp cho quản trị database linh hoạt hơn.

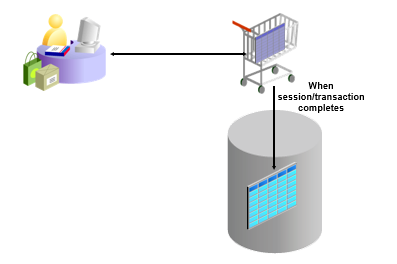
### Temporary tablespace & tempory table

**Temporary tablespace**

* + Tempory Tablespace được sử dụng để chứa các tempory table trong thời gian của các phiên giao dịch.
  + Temporary tablespaces sử dụng lưu trữ:
    - Intermediate sort results: nếu dữ liệu truy vấn trong PGA quá nhiều thì nó có thể được lưu trữ tại đây và lấy dần ra
    - Temporary tables and temporary indexes
    - Temporary LOBs
    - Temporary B-trees
  + Mặc định, chỉ có Temp tablespace duy nhất được tạo ra khi cài đặt một Oracle mới. Nhưng có thể tạo thêm tablespace bổ sung

**Temporary table**

* + Temporary Table là một bảng chứa dữ liệu mà chỉ tồn tại trong thời gian của một giao dịch hoặc phiên . Dữ liệu trong một bảng tạm thời là trong phiên giao dịch, có nghĩa là mỗi phiên chỉ có thể xem và sửa đổi dữ liệu riêng của mình .
  + Temporary table rất hữu ích trong các ứng dụng một tập hợp kết quả được đếm.
  + Ví dụ : một giỏ mua hàng trong một ứng dụng trực tuyến có thể là một bảng tạm thời. Mỗi mục được đại diện bởi một hàng trong bảng tạm thời. Bởi vì bảng tạm thời được định nghĩa cố định , nên có thể tạo index cho Chỉ số tạo ra trên bảng tạm thời cũng chỉ là tạm thời . Các dữ liệu trong các chỉ số có cùng một phiên , phạm vi giao dịch như các dữ liệu trong bảng tạm thời .



### Segments, Extents, and Blocks

**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 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**

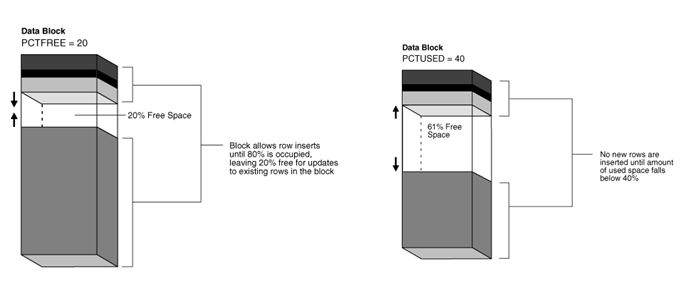
* The next level of logical database space is an extent. An extent is a specific number of contiguous Oracle data blocks (obtained in a single allocation) that are used to store a specific type of information. Oracle data blocks in an extent are logically contiguous but can be physically spread out on disk because of RAID striping and file system implementations.

**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. Example:
  + Data segments: Each nonclustered, non-index-organized table has a data segment, with the exception of external tables, global temporary tables, and partitioned tables (in which 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 for each database instance This tablespace 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 database for future use. Specify either a default temporary tablespace for every user, or a default temporary tablespace that is used database-wide

### Pct used, pct free

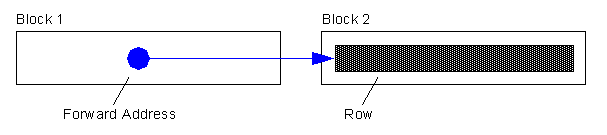
* Freespace: Không gian cấp phát cho việc insert/update trong tương lai, ảnh hưởng bởi giá trị của 2 tham số là PCTUSED và PCTFREE
* Data Row chứa dữ liệu
* Khi create/alter bất kỳ table/index nào, Oracle sẽ sử dụng 2 tham số đề điều khiển không gian
  + PCTFREE: Số % dành riêng cho việc update các dữ liệu đã có trong tương lai
  + PCTUSED: Số % của không gian nhỏ nhất đã được sử dụng cho việc insert data mới, giá trị này xác định khi nào thì các blocks sẽ được đưa trở lại vào trong FREELIST
  + FREELIST: Cấu trúc xác định mà Oracle sử dụng để maitains 1 danh sách các block free hiện có.



* Tham số PCTFREE xác định số % nhỏ nhất (không gian) của 1 data block được dành riêng cho việc update những row đã có trong block đó.
* Ví dụ:
  + Ta xác định 20% là giá trị của PCTFREE trong câu lệnh CREATE TABLE, thì có nghĩa, 20% của từng data block trong table segment sẽ được dành riêng cho việc update các row đã có bên trong từng block.
  + Block chỉ được phép insert dữ liệu khi dữ liệu khi % block giảm xuống nhỏ hơn PCTUSED

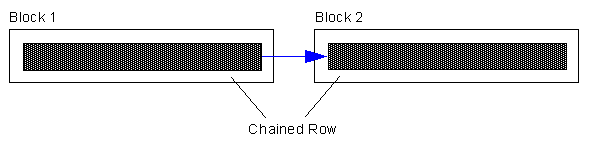
### Row Migration

* Trong trường hợp : 1 row đã được lưu overhead vào trong 1 data block, tuy nhiên, khi insert dữ liệu vào thì free space đã đầy. Vì thế, Oracle sẽ (dịch chuyển) data của toàn bộ row này sang 1 data block mới, với data block cũ, Oracle sẽ giữ lại 1 row piece – cũng chính là rowID để trỏ tới block mới.
* Đối với trường hợp Row Migration, việc Full Table Scan không bị ảnh hưởng (I/O increase), là bởi vì Forward address sẽ bị bỏ qua, tuy nhiên, Row Migration lại làm ảnh hưởng đến việc đọc Index. Bởi vì, Index sẽ nói rằng : “Đến file X, block Y, slot Z…để tìm row này”, và vì thế, khi nhận được thông điệp trên, ta lại mất thêm 1 I/O physical hoặc logical để tìm row này. (Physical I/O, Logical I/O đề cập ở phần khác).

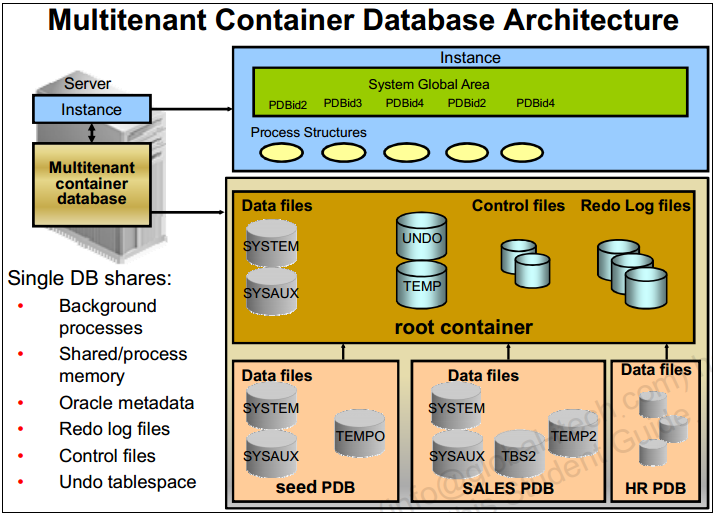


### Row Chaining

* Trong nhiều trường hợp, data cho 1 row quá lớn để lưu trong 1 single data block, do vậy, Oracle sẽ lưu data của row này vào trong 1 hoặc nhiều chained data block (các data block móc nối). Lấy ví dụ, nếu ta sử dụng 1 data block size 4KB cho Database, và cần insert 1 row với size 8KB, Oracle sẽ sử dụng 3 blocks để lưu lại data trong row. Row chaining xảy ra với các trường hợp:
  + Rowsize vượt quá data block size
  + Table có LONG và LONGRAW columns
  + Table có trên 255 column



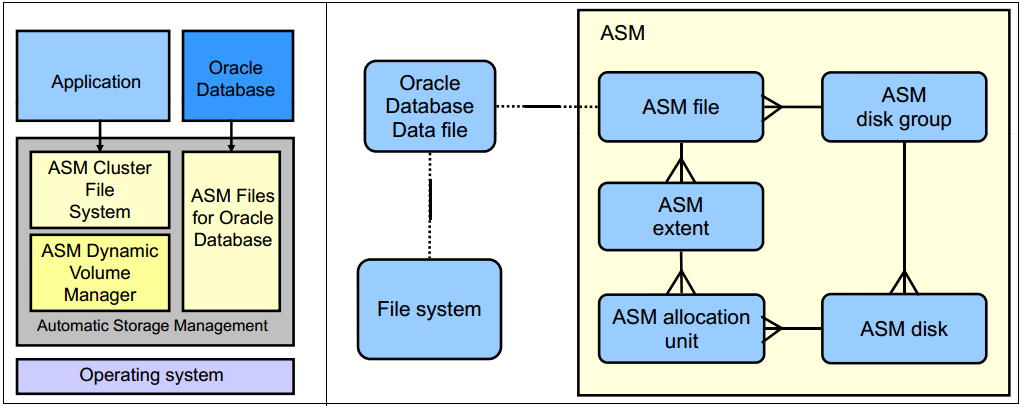
## Multitenant Container Database



At the physical level, the CDB has a database instance and database files, just as a non-CDB does.

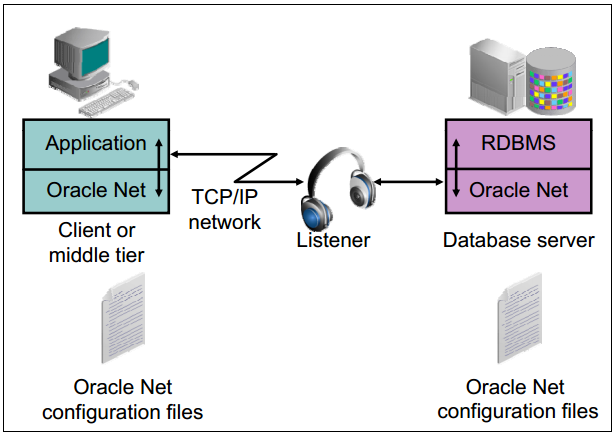
* The redo log files are common for the whole CDB. The information it contains is annotated with the identity of the PDB where a change occurs. Oracle GoldenGate is enhanced to understand the format of the redo log for a CDB. All PDBs in a CDB share the ARCHIVELOG mode of the CDB.
* The control files are common for the whole CDB. The control files are updated to reflect any additional tablespace and data files of plugged PDBs.
* The UNDO tablespace is common for all containers.
* A temporary tablespace common to all containers is required. But each PDB can hold its own temporary tablespace for its own local users.
* Each container has its own data dictionary stored in its proper SYSTEM tablespace, containing its own metadata, and a SYSAUX tablespace.
* Th PDBs can create tablespaces within the PDB according to application needs.
* Each data file is associated with a specific container, named CON\_ID.

## Automatic Storage Management



* Is a portable and high-performance cluster file system
* Manages Oracle database files
* Manages application files with ASM Cluster File System (ACFS)
* Spreads data across disks to balance load
* Mirrors data in case of failures
* Solves storage management challenges

## Oracle Networking



Oracle Net Services enables network connections from a client or middle-tier application to the Oracle server. After a network session is established, Oracle Net acts as the data courier for both the client application and the database server. It is responsible for establishing and maintaining the connection between the client application and database server, as well as exchanging messages between them. Oracle Net (or something that simulates Oracle Net, such as Java Database Connectivity) is located on each computer that needs to talk to the database server.

On the client computer, Oracle Net is a background component for application connections to the database.

On the database server, Oracle Net includes an active process called Oracle Net Listener, which is responsible for coordinating connections between the database and external applications.

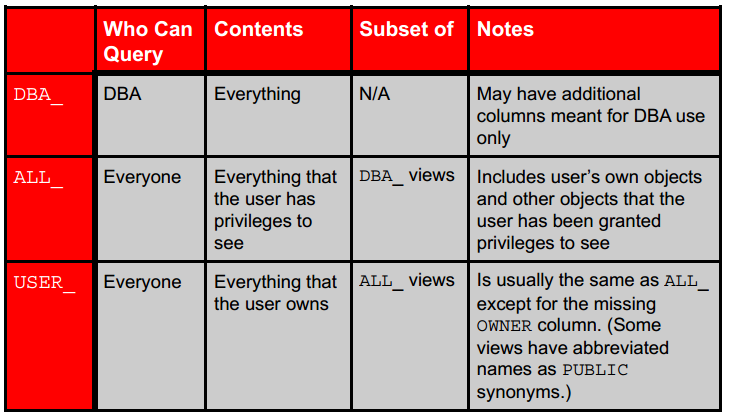
To make a client or middle-tier connection, Oracle Net requires the client to know the:

* Host where the listener is running
* Port that the listener is monitoring
* Protocol that the listener is using
* Name of the service that the listener is handling

# Hoạt động của Database

## Các view cơ bản của Oracle

**Data dictionary views**



**Instance/Database**

* V$DATABASE:  hiển thị thông tin về cơ sở dữ liệu từ các tập tin control file.
* V$INSTANCE:  hiển thị trạng thái của instance hiện tại.
* V$PARAMETER:  hiển thị thông tin về các tham số khởi tạo hiện đang có hiệu lực trong phiên làm việc. Một phiên làm việc mới được thừa hưởng giá trị tham số từ các giá trị sử dụng hoàn toàn hiển thị bởi V $ SYSTEM\_PARAMETER.
* V$SPPARAMETER: hiển thị thông tin nội dung của các tập tin tham số máy chủ.
* V$SYSTEM\_PARAMETER:  hiển thị thông tin về các tham số parameter hiện đang có hiệu lực. Một session mới được thừa hưởng giá trị tham số từ các giá trị instance mở rộng.
* V$PROCESS: chứa thông tin về các tiến trình đang hoạt động
* V$BGPROCESS: hiển thị thông tin về các tiến trình nền(background processes)
* V$PX\_PROCESS\_SYSSTAT: chứa thông tin về các session chạy thực hiện song song
* V$SYSTEM\_EVENT: chứa thông tin về tổng số chờ đợi cho một sự kiện.

**Disk**

* + V$DATAFILE:  chứa thông tin datafile từ các tập tin control file.
  + V$FILESTAT: hiển thị số lượng vật lý đọc và ghi được thực hiện và tổng số single-block và multi-block I / O được thực hiện.
  + V$LOG\_FILE: chứa thông tin về redo log file.
  + V$LOG\_HISTORY: chứa log thông tin đăng nhập lịch sử từ control file.
  + V$DBFILE: chứa thông tin datafile từ các tập tin control file.
  + V$TEMPFILE: hiển thị thông tin tempfile.
  + V$TEMPSEG\_USAGE: mô tả chi tiết sử dụng temporary segment (Bảng chứa dữ liệu tạm thời)
  + V$SEGMENT\_STATISTICS:  hiển thị thông tin về số liệu thống kê segment-level statistics. Tham số tĩnh.

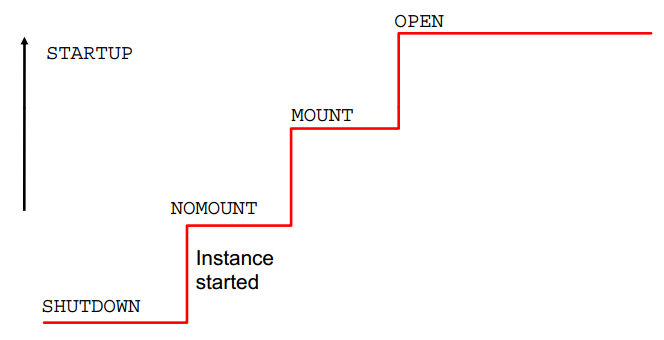
**Memory**

* + V$BUFFER\_POOL\_STATISTICS:  hiển thị số liệu thống kê về tất cả tham số về vùng nhớ đệm buffer pools.
  + V$LIBRARYCACHE: chứa số liệu thống kê tham số về hiệu suất bộ nhớ cache thư viện và hoạt động.
  + V$SGAINFO: hiển thị thông tin về kích thước SGA, bao gồm các kích thước của các thành phần khác nhau SGA, kích thước hạt, và bộ nhớ miễn phí.
  + V$PGASTAT:  cung cấp thông tin về PGA thống kê sử dụng bộ nhớ cũng như thống kê về quản lý bộ nhớ PGA tự động khi nó được kích hoạt.

**Contention**

* + V$LOCK: liệt kê các lock hiện đang nắm giữ cơ sở dữ liệu Oracle và yêu cầu requests lock
  + V$UNDOSTAT: hiển thị một biểu đồ số liệu thống kê cho thấy hệ thống làm việc như thế nào. Số liệu thống kê bao gồm tiêu thụ undo space, các transaction đồng thời , và chiều dài của các truy vấn được thực hiện trong mọi trường hợp.
  + V$WAITSTAT: thông số block contention. Bảng này chỉ được cập nhật khi time statistic được kích hoạt.

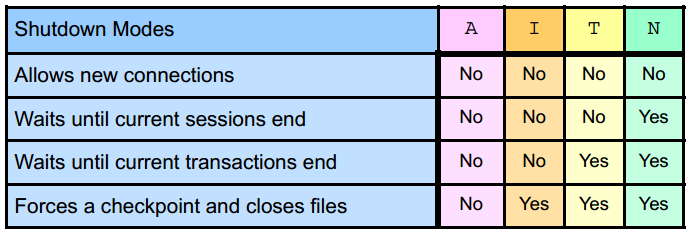
## Startup và Shutdown Database



Database chỉ có thể được khởi động bởi SYSDBA hoặc SYSOPER

Bao gồm 3 bước :

* + Nomount
  + Instance đã được khởi động nhưng chưa được kết nối vs db
    - Oracle sẽ tìm đến spfile ( hoặc pfile nếu được chỉ định)
    - Đọc parameter file để cài đặt các tham số
    - Khởi động các background process cần thiết
    - Mở alertlog và trace file để ghi lại log về các tham số đã khởi động
  + Mount
    - Instance đã được kết nối với db bằng cách đọc control file
    - DB chưa được mở, chỉ có DBA mới có quyền truy cập
    - Oracle sau khi đọc spfile ( hoặc pfile) xác định được vị trí control file và đọc nó để có tên và vị trí của các datafile và redolog file
  + Open
  + DB đã mở, người dùng hợp lệ có thể thao tác vs dữ liệu trên db thông qua instance
  + Oracle open datafile và online redo logfile



Shutdown modes:

* + A = ABORT
  + I = IMMEDIATE
  + T = TRANSACTIONAL
  + N = NORMAL

Shutdown modes are progressively more accommodating of current activity in this order:

* + **ABORT**: Performs the least amount of work before shutting down. Because this mode requires recovery before startup, use it only when necessary. It is typically used when no other form of shutdown works, when there are problems with starting the instance, or when you need to shut down immediately because of an impending situation (such as notice of a power outage within seconds).
  + A shutdown in IMMEDIATE mode proceeds with the following conditions:
    - Current SQL statements being processed by the Oracle database are not completed.
    - The Oracle server does not wait for the users who are currently connected to the database to disconnect.
    - The Oracle server rolls back active transactions and disconnects all connected users.
    - The Oracle server closes and dismounts the database before shutting down the instance.
    - The next startup does not require an instance recovery. TRANSACTIONAL: Allows existing transactions to finish, but does not allow new transactions to start
  + **TRANSACTIONAL**: Allows existing transactions to finish, but does not allow new transaction to start
  + **NORMAL**: Waits for sessions to disconnect

## Loging: Admin log in database

**Alert log file**

Location*: $ORACLE\_BASE/diag/rdbms/<db\_name>/<SID>/trace*

The alert file of a database is a chronological log of messages such as the following:

* + Any nondefault initialization parameters used at startup
  + All internal errors (ORA-600), block corruption errors (ORA-1578), and deadlock errors (ORA-60) that occurred
  + Administrative operations, such as the SQL statements *CREATE, ALTER*, *DROP*

*DATABASE*, and *TABLESPACE*; and the Enterprise Manager or SQL\*Plus statements *STARTUP, SHUTDOWN, ARCHIVE LOG,* and *RECOVER*

* + Several messages and errors relating to the functions of shared server and dispatcherprocesses
  + Errors during the automatic refresh of a materialized view

**DDL Log File**

The DDL log is created only if the ENABLE\_DDL\_LOGGING initialization parameter is set to TRUE

* + DDL log contains one log record for each DDL statement.
  + Two DDL logs containing the same information:
    - XML DDL log: *log.xml* written to

*$ORACLE\_BASE/diag/rdbms/<dbname>/<SID>/log/ddl*

* + - Text DDL: *ddlsid.log* written to

*$ORACLE\_BASE/diag/rdbms/<dbname>/<SID>/log*

**Debug Log File**

Debug log contains warnings about conditions, states, or events that do not inhibit correct operation of an Oracle Database component.

* + The log is intended for use by Oracle Support when diagnosing a problem.
  + It is included in incident packaging service (IPS) incident packages.
  + It is written to

*$ORACLE BASE/diag/rdbms/<db name \_name>/<SID>/debug.*

## **Oracle User Security**

### Database User Accounts

Each database user account has:

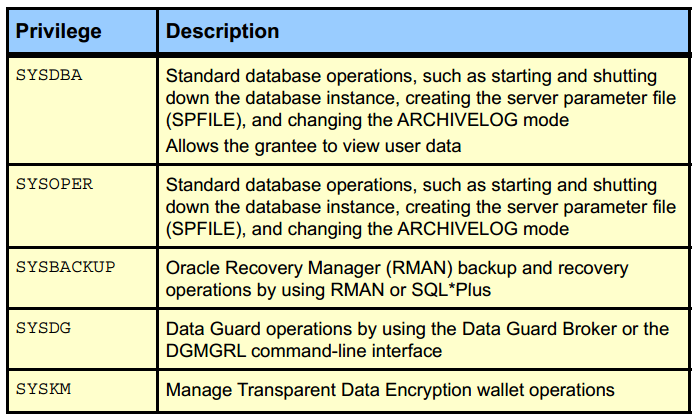
* A unique username
* An authentication method
* A default tablespace
* A temporary tablespace
* A user profile
* An initial consumer group (quota)
* An account status

A schema:

* Is a collection of database objects that are owned by database user
* Has the same name as the user account

### Admin Account

* SYS:
  + Owns the data dictionary and the Automatic Workload Repository (AWR)
  + Used for startup and shutdown of the database instance
* SYSTEM: Owns additional administrative tables and views
* SYSBACKUP: Facilitates Oracle Recovery Manager (RMAN) backup and recovery operations
* SYSDG: Facilitates Oracle Data Guard operations
* SYSKM: Facilitates Transparent Data Encryption walletoperations



### Privileges

A privilege is a right to execute a particular type of SQL statement or to access another user’s object. Privileges are divided into two categories:

* **System privileges**: Each system privilege allows a user to perform a particular database operation or class of database operations. For example, the privilege to create tablespaces is a system privilege. System privileges can be granted by the administrator or by someone who has been given explicit permission to administer the privilege. There are more than 170 distinct system privileges. Many system privileges contain the ANY clause.
* **Object privileges**: O f bject privileges allow a user to perform a particular action on a specific object, such as a table, view, sequence, procedure, function, or package. Without specific permission, users can access only their own objects. Object privileges can be granted by the owner of an object, by the administrator, or by someone who has been explicitly given permission to grant privileges on the object.

### Role

A role is a named group of related privileges that are granted to users or to other roles. You can use roles to administer database privileges. You can add privileges to a role and grant the role to a user. The user can then enable the role and exercise the privileges granted by the role. A role contains all privileges that are granted to that role and all privileges of other roles that are granted to it.

Roles provide the following benefits with respect to managing privileges:

* Easier privilege management: Use roles to simplify privilege management. Rather than granting the same set of privileges to several users, you can grant the privileges to a role and then grant that role to each user.
* Dynamic privilege management: If the privileges associated with a role are modified, all users who are granted the role acquire the modified privileges automatically and immediately.
* Selective availability of privileges: Roles can be enabled and disabled to turn privileges on and off temporarily. This allows the privileges of the user to be controlled in a given situation.

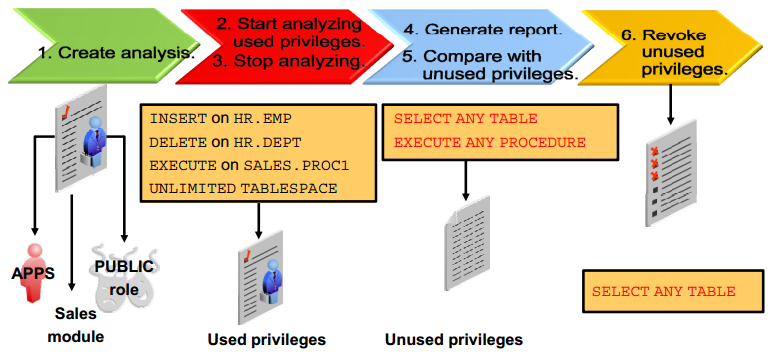
### Privilege Analysis

* Analyze used privileges to revoke unnecessary privileges.
* Use DBMS\_PRIVILEGE\_CAPTURE package.

A major concern in many databases is that existing database and application users have excessive privileges. Excessive privileges violate the principle of least privilege. To achieve the least privilege principle, unused privileges need to be identified.

Oracle Database 12c provides a package named DBMS\_PRIVILEGE\_CAPTURE to analyze used privileges.

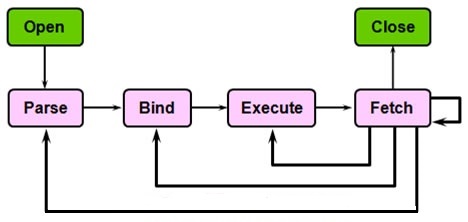
You can use a privilege analysis policy to identify object and system privileges used to run an application module or to execute certain SQL statements or privileges used by defined roles. You can generate reports of used and unused privileges during the analysis period. The report helps the security officer revoke unnecessary privileges by comparing the used and unused granted privilege lists.



## **Query SQL**

### **Kế hoạch thực thi 1 câu lệnh trong Oracle**

Oracle Server xử lý câu lệnh SQL theo trình tự chính sách thực thi các bước sau:



**Open**

* + Ngầm định khai báo và khởi tạo Cursor cho câu lệnh SQL

**Parse**

* + Syntatic: Kiểm tra cú pháp
  + Semantic : Kiểm tra đối tượng (object)
  + View merging: Rewrite lại câu lệnh dựa vào các based table thay vì sử dụng view
  + Statement Transformation : Rewirte lại sự biến đổi của câu lệnh để phân tích thành những câu đơn giản hơn.
  + Optmization : Tối ưu hóa câu lệnh
  + QEP Generation : Query Evulation Plan : đánh giá kế hoạch cho câu lệnh

**Bind**

* + Tìm và gán giá trị cho các [**bind-variable**](http://www.vietpace.com/kienthuc/VietPace_toiuu_caulenh_Oracle_SQL_Phan1.html#_Giải_nghĩa_từ) nếu có

**Execute**

* + Thực thi các bước mô tả trong “sơ đồ thực thi câu lệnh SQL”

**Fetch**

* + Chuyển kết quả về nơi gọi thực thi lệnh
  + Fetch ở đây có thể lặp lại nhiều lần do tham số limit của nó ( giới hạn xử lý mỗi lần)

**Close**

* + Ngầm định đóng Cursor cho câu lệnh

### Excution plan

EXPLAIN PLAN

SET statement\_id = 'ex\_plan2' FOR

SELECT last\_name FROM employees

WHERE last\_name LIKE 'Pe%';

SELECT PLAN\_TABLE\_OUTPUT

FROM TABLE(DBMS\_XPLAN.DISPLAY(NULL, 'ex\_plan2','BASIC'));

----------------------------------------

| Id | Operation | Name |

----------------------------------------

| 0 | SELECT STATEMENT | |

| 1 | INDEX RANGE SCAN| EMP\_NAME\_IX |

----------------------------------------

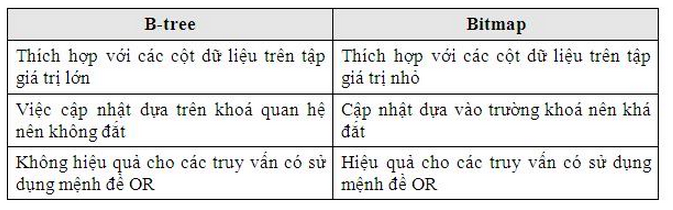
### Cơ chế index của Oracle.

Index là những cấu trúc tùy chọn liên quan đến bảng và cột.

* + Có thể tạo index trên một hoặc nhiều cột của một bảng để tăng tốc độ thực hiện câu lệnh SQL trên bảng đó.
  + Các index này sẽ giúp bạn xác định vị trí thông tin nhanh hơn. Index là cách thức chính của việc giảm đĩa I / O khi được sử dụng đúng cách.
  + Quyết định index đối với 1 bảng dựa vào yêu tố
    - Tạo Index nếu kết quả nhỏ hơn 15% tổng số row của 1 bảng lớn
    - Tăng hiệu năng cho phép join bảng ( bằng cách index cột dùng để join)
    - Không index cho bảng nhỏ
  + Không index tại cột kiểu LONG và LONGRAW

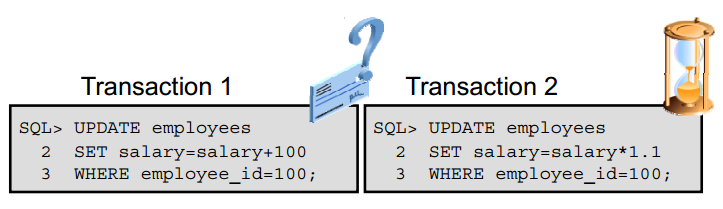
Các loại index chính:

* + B-Tree indexes
    - Btree Index được dùng để giúp truy vấn các câu truy vấn dạng Insert, Update, delete.
    - B-tree là loại index mặc định - nếu tạo ra một index mà không xác định bất cứ điều gì, thì đó là một index B-tree.
  + Bitmap indexes
    - Là 1 index được dùng để làm việc với những trường có dữ liệu rời rạc, với số lượng ít các giá trị khác nhau (tức mức độ lặp lại ở trong trường này thường là lớn)
    - Được sử dụng khI Mức độ dữ liệu trùng lặp lớn: trong oracle thì quyết định là: distinct val/ total val < 1% thì dùng bitmap index : Tức nếu số giá trị rời rạc của 1 cột trong bảng trên tổng số dòng của bảng mà nhỏ hơn 1% thì ta dùngBit map index.
    - Ví dụ trường giới tính: distinct val = 2 <số giá trị riêng biệt> và Total val = 1000 dòng. 2/1000 < 1% dùng bitmap.
    - Không hoặc là ít thao tác update hoặc insert lên bảng dữ liệu.
    - Bảng có rất nhiều cột



* + Bitmap join indexes
    - Bitmap join index cho tham gia giữa các bảng (2+). Tạo Bitmap join index được xác định trên một bảng duy nhất. Nó lưu trữ các kết quả của sự join.
  + Function-based indexes
    - Có thể tạo fuction-based indexes cho những truy vấn sử dụng các function đối với các trường:
      * Upper(abc)
      * 12 \*salary

### Locks



Before the database server allows a session to modify data, the session must first lock the data that is being modified. A lock gives the session exclusive control over the data so that no other transaction can modify the locked data until the lock is released

* + Prevent multiple sessions from changing the same data at the same time
  + Are automatically obtained at the lowest possible level for a given statement
  + Do not escalate

**Locking Mechanism**

* + High level of data concurrency:
    - Row-level locks for inserts, updates, and deletes
    - No locks required for queries
  + Automatic queue management
  + Locks held until the transaction ends (with a commit or rollback operation)

**Lock modes**

* + ROW SHARE: Permits concurrent access to the locked table but prohibits sessions from locking the entire table for exclusive access
  + ROW EXCLUSIVE: Is the same as ROW SHARE, but also prohibits locking in SHARE mode. The ROW EXCLUSIVE locks are automatically obtained when updating, inserting, or deleting data. ROW EXCLUSIVE locks allow multiple readers and one writer.
  + SHARE: Permits concurrent queries but prohibits updates to the locked table. Share locks allow multiple readers and no writers.
    - SHARE ROW EXCLUSIVE: Is used to query y a whole table and to allow others to query rows in the table, but prohibits others from locking the table in SHARE mode or updating rows
    - EXCLUSIVE: Permits queries on the locked table but prohibits any other activity on it. An EXCLUSIVE lock is required to drop a table

**DML Locks**

Each DML transaction obtains two locks:

* + An EXCLUSIVE row lock on the row or rows being updated
  + A table lock (TM) in ROW EXCLUSIVE (RX) mode on the table being updated This prevents another session from locking the whole table (possibly to drop or truncate it) while the change is being made. This mode is also called a subexclusive table lock (SX)

**Lock Conflicts**

Cause

* + Uncommitted changes
  + Long-running transactions
  + Unnecessarily high locking levels

To resolve a lock conflict:

* + Have the session holding the lock commit or roll back
  + Terminate the session holding the lock (in an emergency)

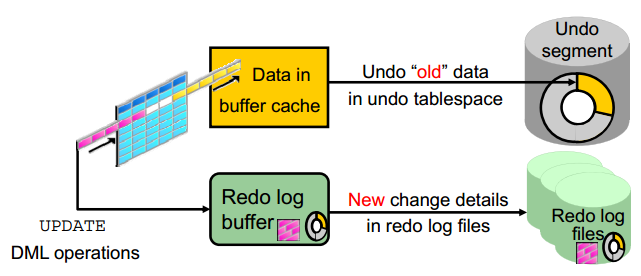
**Deadlock**

A deadlock is a special example of a lock conflict. Deadlocks arise when two or more sessions wait for data that has been locked by the other. Because each is waiting for the other, neither can complete their transaction to resolve the conflict.

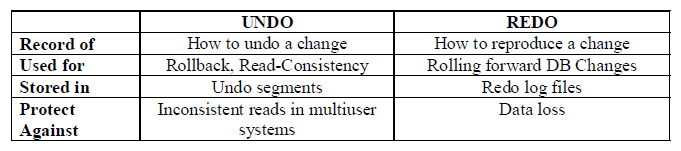
Oracle Database automatically detects deadlocks and terminates the statement with an error. The proper response to that error is either commit or roll back, which releases any other locks in that session so that the other session can continue its transaction

## Backup & Recover Database

### Undo and Redo

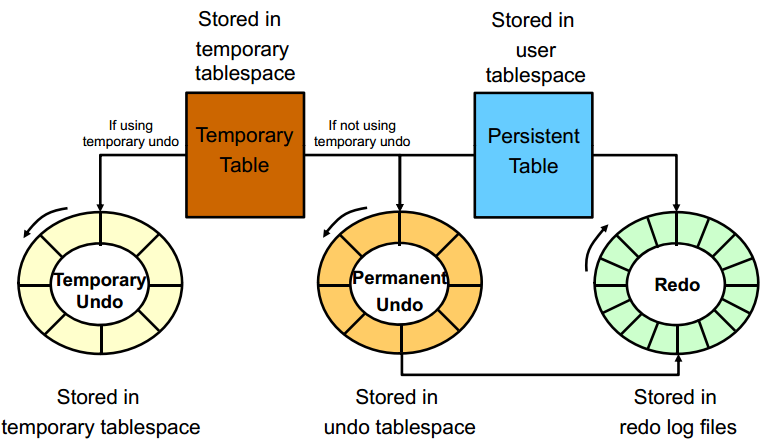


* + Khi các thao tác DML như: insert/ update / delete thực sự tạo thay đổi cho data block, mặc dù chưa commit thao tác. Để chắc chắn tính nguyên vẹn của database Oracle ghi dữ liệu cũ vào UNDO để có thể rollback lại. Còn các thao tác với dữ liệu ghi vào REDO log để re-play lại thao tác đó nếu như nó thất bại
  + Undo data is:
    - A record of the action of a transaction
    - Captured for every transaction that changes data
    - Retained at least until the transaction is ended
    - Used to support:
      * Rollback operations
      * Read-consistent queries
      * Oracle Flashback Query, Oracle Flashback Transaction, and Oracle Flashback Table
      * Recovery from failed transactions



### Temporary Undo

New Feature



Starting with Oracle Database 12c it is possible for undo generated by temporary tables’ transactions to be stored in a separate undo stream directly in the temporary tablespace to avoid for that undo to be logged in the redo stream. This mode is called temporary undo

Benefits

* + Temporary undo reduces the amount of undo stored in the undo tablespaces.
  + Temporary undo reduces the size of the redo log.
  + Temporary undo enables DML operations on temporary tables in a physical standby database with the Oracle Active Data Guard option.

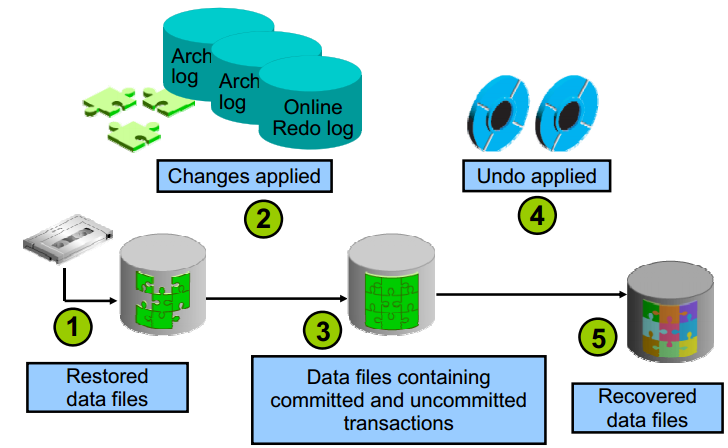
### Recovery

**Automatic Recovery**

Nếu database bị crash hoặc shutdown đột ngột thì khi khởi động lại thì Oracle thực hiện công việc recovery. Công việc này diễn ra như sau

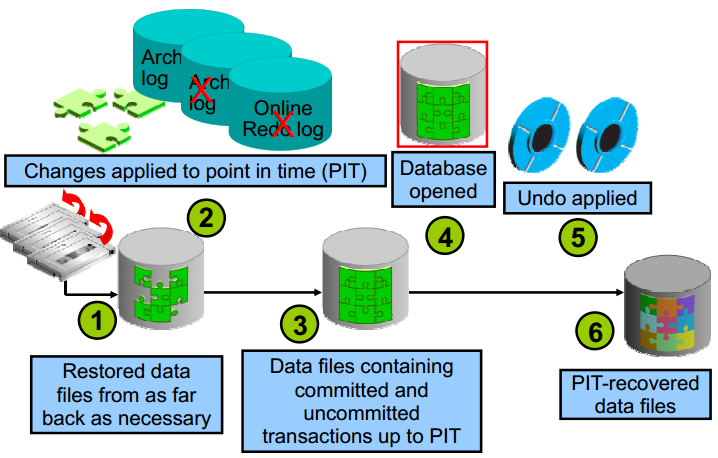
* + Instance startup (data file are out of sync)
  + Roll forward (redo)
  + Committed and uncommitted data in files
  + Database opened
  + Roll back (undo)
  + Committed data in files

**Complete Recovery**



* + Damaged or missing files are restored from a backup.
  + Changes from incremental backups archived redo log files and online redo log files are applied as necessary. The redo log changes are applied to the data files until the current online log is reached and the most recent transactions have been re-entered. Undo blocks are generated during this entire process. This is referred to as rolling forward or cache recovery.
  + The restored data files may now contain committed and uncommitted changes.
  + The undo blocks are used to roll back any uncommitted changes. This is sometimes referred to as transaction recovery.
  + The data files are now in a recovered state and are consistent with the other data files in the database.

**Point-in-Time Recovery**



Incomplete recovery, or database point-in-time recovery (DBPITR), uses a backup to produce a noncurrent version of the database. That is, you do not apply all of the redo records generated after the most recent backup. You need:

* + A valid offline or online backup of all the data files made before the recovery point
  + All archived logs from the time of the backup until the specified time of recovery

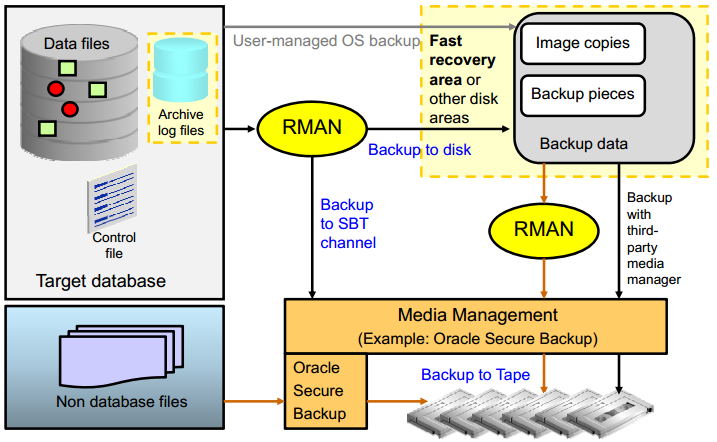
The steps to perform a point-in-time recovery are as follows:

* + Restore the data files from backup: The backup that is used must be from before your target recovery point, using the RMAN RESTORE command.
  + Use the RECOVER command: Apply redo from the archived redo log files, including as many as necessary to reach the restore point destination.
  + State of over-recovery: Now the data files contain some committed and some

uncommitted transactions because the redo can contain uncommitted data.

* + Use the ALTER DATABASE OPEN command: The database is opened before undo is applied This is to provide higher availability.
  + Apply undo data: While the redo was being applied, redo supporting the undo data files was also applied.
  + Process complete: The data files are now recovered to the point in time that you chose.

### Backup



Data backup là việc tạo ra các bản sao của dữ liệu gốc, cất giữ ở một nơi an toàn. Và lấy ra sử dụng (restore) khi hệ thống gặp sự cố. Sao lưu (backup) dữ liệu là cách tốt nhất hiện nay để bảo vệ dữ liệu

Recovery Manager (RMAN) is the recommended method of backing up your Oracle database. You can use it to back up to disk or to a system backup to tape (SBT) channel. Oracle recommends that disk backups be stored in the fast recovery area (FRA).

Mục địch của việc backup-restore dữ liệu này là để đưa hệ thống trở lại trạng thái trước khi gặp sự cố. Nguyên nhân của sự cố gây ảnh hưởng đến dữ liệu có thể thuộc một trong 2 dạng chính sau:

* + Nguyên nhân khách quan: Sự cố xảy ra ngoài ý muốn, con người không thể biết trước được, thường là các thảm họa (VD: thiên tai, cháy nổ,…).
  + Nguyên nhân chủ quan: Sự cố xảy ra do những thao tác không chính xác của con người (ví dụ: lỗi phần cứng, lỗi phần mềm, thao tác nhầm…).

**RMAN Backup Types**

**Full Backups**

* + A full backup is different from a whole database backup. A full data file backup is a backup that includes every used data block in the file. RMAN copies all blocks into the backup set or image copy, skipping only those data file blocks that are not part of an existing segment. For a full image copy, the entire file contents are reproduced exactly.
  + A full backup cannot be part of an incremental backup strategy; it cannot be the parent for a subsequent incremental backup.

**Incremental Backups**

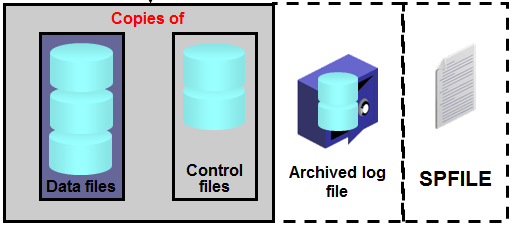
* + An incremental backup is either a level 0 backup, which includes every block in the data files except blocks that have never been used, or a level 1 backup, which includes only those blocks that have been changed since a previous backup was taken. A level 0 incremental backup is physically identical to a full backup.
  + The only difference is that the level 0 backup (as well as an image copy) can be used as the base for a level 1 backup, but a full backup can never be used as the base for a level 1 backup. Incremental backups are specified using the INCREMENTAL keyword of the BACKUP command. You specify INCREMENTAL LEVEL [0 | 1].
    - **Differential**: (default) backs up all blocks changed after the most recent incremental backup at either level 1 or level 0
    - **Cumulative**: Backs up all blocks changed after the most recent backup at level 0
  + Mục đích chính của incremental backup:
    - Tiết kiệm thời gian và dung lượng backup.
    - Restore nhanh hơn so với Full backup. Tuy nhiên khi Restore cần đủ file:1 File Full backup lần gần nhất và tất cả các File Incremental backup từ thời điểm Full backup đến thời điểm cần restore.

Thực hiện incremental backup:

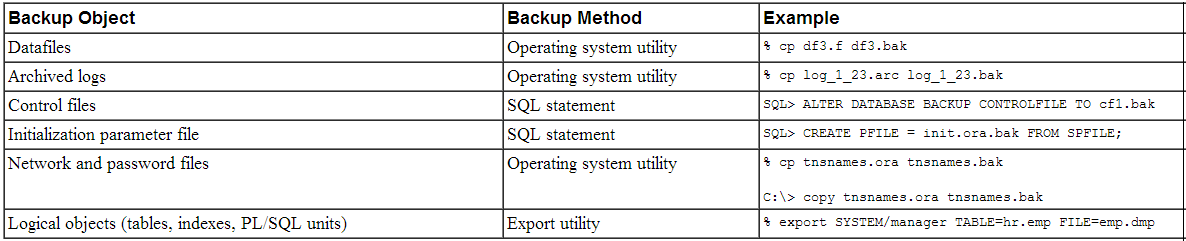
* + - Đầu tiên ta phải có một bản backup full level 0.
    - Ngày tiếp theo tạo bản backup incremental backup tất cả các block thay đổi so với bản backup level 0 ban đầu.
    - Ngày thứ 2 tạo bản backup incremental backup tất cả block thay đổi so với bản backup level 1 của ngày thứ 1.

**Các cách backup DB:**

* + Export và import : data pump
    - Oracle export dữ liệu ra file .dmp
    - File .dmp có thể được lưu trữ như 1 file backup cho db
  + Backup bằng cách sử dụng RMAN
    - * Oracle copy toàn bộ các data file và control file, có thể bao gồm archivelog file và spfile thành 1 hoặc nhiều file, lưu trữ ở vùng FRA. Khi cần recover thì mang ra sử dụng



* + User-manged backup
    - Phương thức này tương đối thủ công, thực hiện do DBA là chính.
    - Chủ yếu thực hiện cho những DB có dung lượng rất lớn



* + - Phương án hay sử dụng nhất : đóng băng DB ( suspended mode)
      * Nguyên lý hoạt động : đóng băng DB ở suspended mode, sau đó, di chuyển con trỏ dữ liệu sang 1 vùng mới và giải băng cho DB. DB hiện trạng = bản đóng băng + vùng mới.

## Moving Data

### Moving or Renaming an Online Data File

New feauture

* + Relocating an online data file:

ALTER DATABASE MOVE DATAFILE '/disk1/myexample01.dbf'

TO '/disk2/myexample01.dbf';

* + Copying a data file from a file system to ASM:

ALTER DATABASE MOVE DATAFILE '/disk1/myexample01.dbf'

TO '+DiskGroup2' KEEP;

* + Renaming an online data file

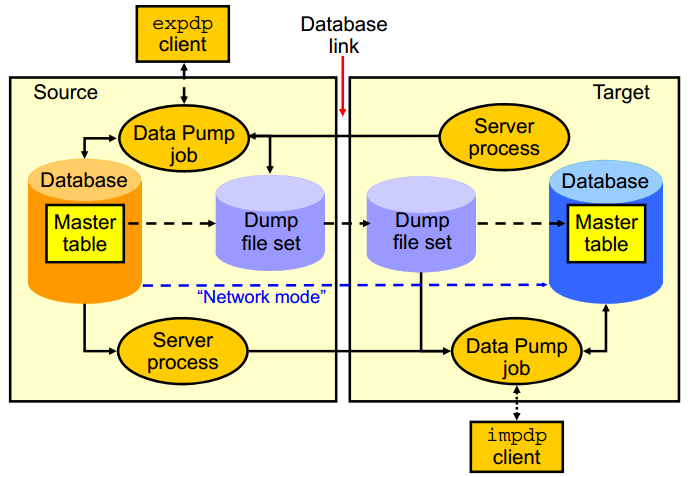
ALTER DATABASE MOVE DATAFILE '/disk1/myexample1.dbf'

TO '/disk1/myexample01.dbf';

Real action:

* + Back up as copy ( use rman)
  + Offline datafile
  + Switch to copy
  + Recover datafile
  + Online datafile

### Oracle Data Pump



Oracle Data Pump enables very high-speed data and metadata loading and unloading of Oracle databases. The Data Pump infrastructure is callable via the DBMS\_DATAPUMP PL/SQL package. Thus, custom data movement utilities can be built by using Data Pump.

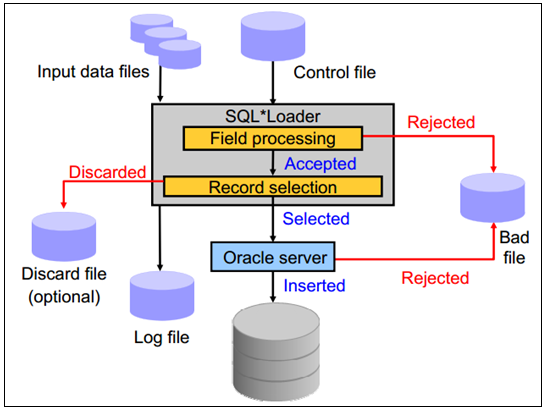
Oracle Database provides the following tools:

* + Command-line export and import clients called expdp and impdp, respectively
  + Export and import interface in Enterprise Manager Cloud Control

Data Pump automatically decides the data access methods to use; these can be either direct path or external tables. Data Pump uses direct path load and unload when a table’s structure allows it and when maximum single-stream, performance is desired. However, if there are clustered tables, referential integrity constraints, encrypted columns, or several other items, Data Pump uses external tables rather than direct path to move the data.

The ability to detach from and re-attach to long-running jobs without affecting the job itself enables you to monitor jobs from multiple locations while they are running. All stopped Data Pump jobs can be restarted without loss of data as long as the metainformation remains undisturbed. It does not matter whether the job is stopped voluntarily or involuntarily due to a crash.

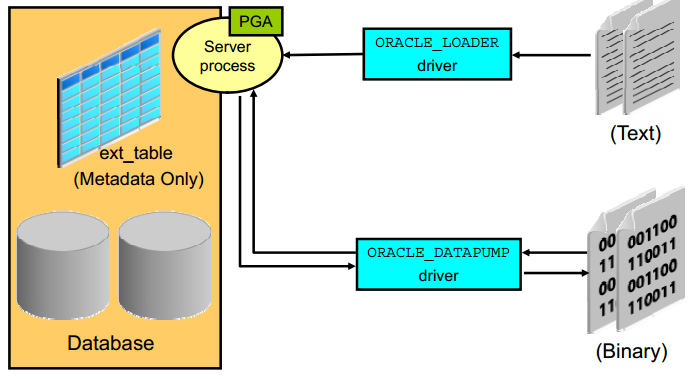
### SQL\*Loader



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 data file. SQL\*Loader uses the following files:

* + **Input data files**: SQL\*Loader reads data from one or more files (or operating system equivalents of files) that are specified in the control file.
  + **Control file**: The control file is a text file that is written in a language that SQL\*Loader understands. The control file indicates to SQL\*Loader where to find the data, how to parse and interpret the data, where to insert the data, and so on. • The first section contains such session-wide information as the following:
  + **Log file**: When SQL\*Loader begins execution, it creates a log file. If it cannot create a log file, execution terminates. The log file contains a detailed summary of the load, including a description of any errors that occurred during the load.
  + **Bad file**: The bad file contains records that are rejected, either by SQL\*Loader or by the Oracle database. Data file records are rejected by SQL\*Loader when the input format is invalid.
  + **Discard file**: This file is created only when it is needed and only if you have specified that a discard file should be enabled. The discard file contains records that are filtered out of the load because they do not match any record-selection criteria specified in the control file.

### External Tables



External tables access data in external sources as if it were in a table in the database. You can connect to the database and create metadata for the external table using DDL. The DDL for an external table consist of two parts: one part that describes the Oracle Database column types, and another part that describes the mapping of the external data to the Oracle Database data columns.

An external table does not describe any data that is stored in the database. Nor does it describe how data is stored in the external source. Instead, it describes how the external table layer must present the data to the server. It is the responsibility of the access driver and the external table layer to do the necessary transformations required on the data in the external file so that it matches the external table definition External tables are read only; therefore no . External tables are read only; therefore, no DML operations are possible, and no index can be created on them.

There are two access drivers used with external tables. The ORACLE\_LOADER access driver can be used only to read table data from an external table and load it into the database. It uses text files as the data source. The ORACLE\_DATAPUMP access driver can both load table data from an external file into the database and also unload data from the database into an external file. It uses binary files as the external files. The binary files have the same format as the file used by the Data pump Import and Export utilities and can be interchanged with

## Oracle Scheduler

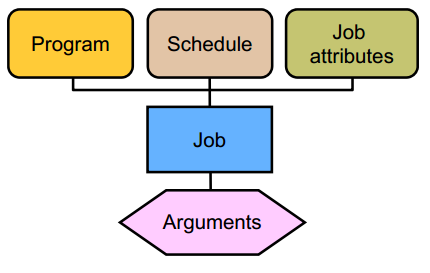
### Concept

Oracle Database provides advanced scheduling capabilities through the Oracle Scheduler, which is a collection of functions and procedures in the DBMS\_SCHEDULER package. The Scheduler can be invoked in any SQL environment, or through Enterprise Manager Cloud Control.

A job has two key components:

* + Action: “What” needs to be done
  + Schedule: “When” the action occurs

### Components



The Scheduler uses the following basic components:

* + **A job** specifies what needs to be executed. It could be as an example a PL/SQL procedure, a native binary executable, a Java application, or a shell script.
  + **A schedule** specifies when and how many times a job is executed A schedule can be . A schedule can be based on time or an event. You can define a schedule for a job by using a series of dates, an event, or a combination of the two, along with additional specifications to denote repeating intervals. You can store the schedule for a job separately and then use the same schedule for multiple jobs.
  + **A program** is a collection of metadata about a particular executable, script, or procedure. An automated job executes some task. Using a program enables you to modify the job task, or the “what,” without modifying the job itself. You can define arguments for a program, enabling users to modify the runtime behavior of the task.

Using Oracle Scheduler

* + Create a program (enabled or disabled)—optional
    - To reuse this action within multiple jobs
    - To change the schedule for a job without having to re-create the PL/SQL block
  + Create and use a schedule ( event base, time base, fix base )
  + Create and submit a job.

### Make a schedule

* + Time based

BEGIN  
DBMS\_SCHEDULER.CREATE\_JOB(  
job\_name=>'HR.DO\_BACKUP',  
job\_type => 'EXECUTABLE',  
job\_action =>  
'/home/usr/dba/rman/nightly\_incr.sh',  
start date \_date=> SYSDATE,  
repeat\_interval=>'FREQ=DAILY;BYHOUR=23',  
/\* next night at 11:00 PM \*/  
comments => 'Nightly incremental backups');  
END;  
/

* + Event-Based

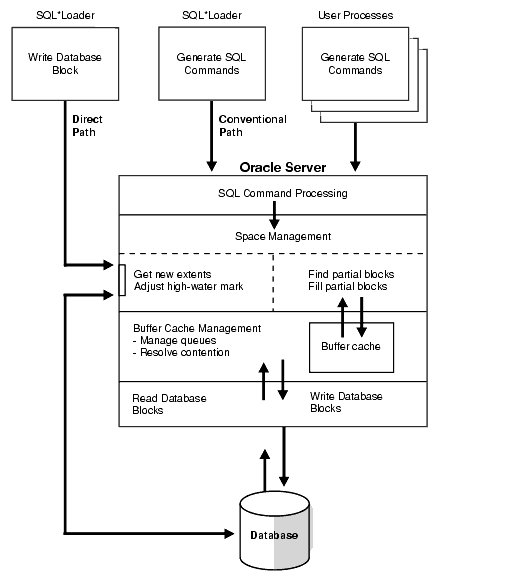
BEGIN  
DBMS SCHEDULER.CREATE JOB( \_ \_  
job\_name=>'ADMIN.PERFORM\_DATA\_LOAD',  
job\_type => 'EXECUTABLE',  
job\_action => '/loaddir/start\_my\_load.sh',  
start\_date => SYSTIMESTAMP,  
event\_condition => 'tab.user\_data.object\_owner =  
''HR'' and tab.user\_data.object\_name = ''DATA.TXT''  
and t b d t t t ''FILE ARRIVAL'' ab.user\_data.event\_type = ''FILE\_ARRIVAL''  
and tab.user\_data.event\_timestamp < 9 ',  
queue\_spec => 'HR.LOAD\_JOB\_EVENT\_Q');  
END;

/

## Managing Database

### Direct path và conventional path

SQL loader cung cấp 2 phương thức để cập nhật dữ liệu

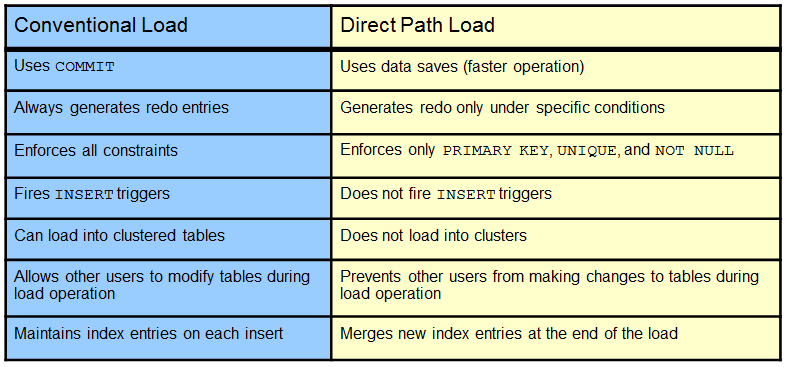


* + Conventional path thực thi câu lệnh insert tới các bảng trong 1 Oracle DB:

INSERT INTO TABLE T PARTITION (P) VALUES ...

* + Direct path ghi trực tiếp lên các datablock trên datafile

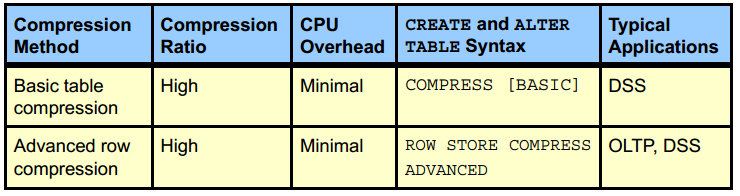
LOAD INTO TABLE T PARTITION (P) VALUES ...



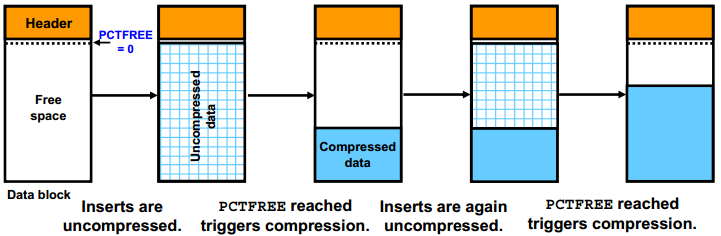
### Table Compression

Reducing storage costs by compressing all data:

* + Basic compression for direct-path insert operations: 10x
  + Advanced row compression for all DML operations: 2–4x



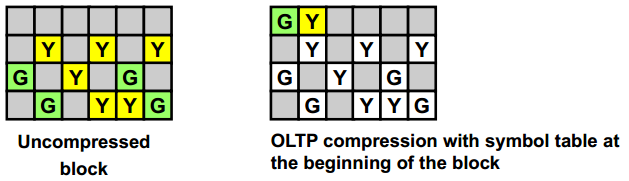
**Basic table compression**



With *COMPRESS* or *COMPRESS BASIC*, you enable basic table compression.

* + The Oracle Database server attempts to compress data during the following direct-pathinsert operations when it is productive to do so:
    - Direct-path SQL\*Loader
    - CREATE TABLE AS SELECT statements
    - Parallel INSERT statements
    - INSERT statements with an APPEND hint
  + **How**: Compression eliminates holes created due to deletions and maximizes contiguous free space in blocks.

**Advanced Row Compression**



With *ROW STORE COMPRESS ADVANCED*, you enable advanced row compression.

* + The Oracle database compresses data during all DML operations on the table. This form of compression is recommended for active OLTP environments.
  + In earlier releases, OLTP table compression was enabled with COMPRESS FOR ALL OPERATIONS and COMPRESS FOR OLTP. This syntax has been deprecated.

**How**:With advanced row compression, duplicate values in the rows and columns in a data block are stored once at the beginning of the block in a symbol table. Duplicate values are replaced with a short reference to the symbol table (as shown in the slide). Thus, information needed to re-create the uncompressed data is stored in the block.

**Specifying Table Compression**

You can specify table compression:

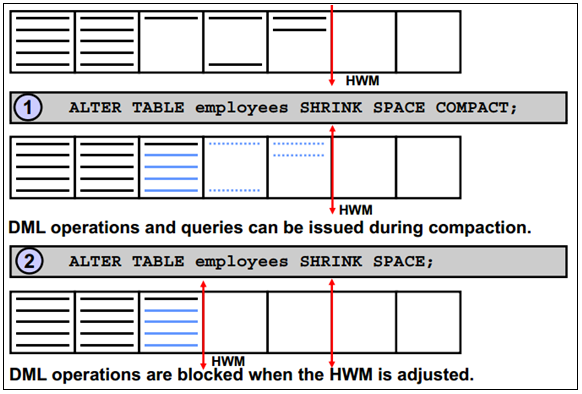
* + For an entire heap-organized table (in the *physical\_properties* clause of *relational\_table* or *object table*)
  + For partitioned tables (Each partition can have a different type or level of compression.)
  + For the storage of a nested table (in the nested\_table\_col\_properties clause)

Table compression has the following restrictions:

* + *ROW STORE COMPRESS ADVANCED* and *COMPRESS BASIC* are not supported for tables with more than 255 columns.
  + You cannot drop a column from a table that is compressed for direct-load operations, although you can set such a column as unused.

### Shrinking Segments

\*\*\*One way to Improved performance and space utilization



The diagram in the slide describes the two phases of a table shrink operation.

* + Compaction is performed in the first phase. During this phase, rows are moved to the left part of the segment as much as possible. Internally, rows are moved by packets to avoid locking issues.
  + After the rows have been moved, the second phase of the shrink operation is started. During this phase, the high-water mark (HWM) is adjusted and the unused space is released.

The *COMPACT* clause is useful if you have long-running queries that might span the shrink operation and attempt to read from blocks that have been reclaimed. When you specify the *SHRINK SPACE COMPACT* clause, the progress of the shrink operation is saved in the bitmap blocks of the corresponding segment. This means that the next time a shrink operation is executed on the same segment, the Oracle Database server remembers what has been done already. You can then reissue the *SHRINK SPACE* clause without the *COMPACT* clause during off-peak hours to complete the second phase.

### Managing Resumable Space Allocation

A resumable statement:

* + Enables you to suspend large operations instead of receiving an error
  + Gives you a chance to fix the problem while the operation is suspended, rather than starting over
  + Is suspended for the following conditions:
    - Out of space
    - Maximum extents reached
    - Space quota exceeded
  + A resumable statement can be suspended and resumed multiple times.

ALTER SESSION ENABLE RESUMABLE;  
INSERT INTO sales\_new SELECT \* FROM sh.sales;  
ALTER SESSION DISABLE RESUMABLE;

The following operations are resumable:

* + **Queries**: *SELECT* statements that run out of temporary space (for sort areas) are candidates for resumable execution. When using OCI, the *OCIStmtExecute()* and *OCIStmtFetch()* calls are candidates.
  + **DML**: *INSERT, UPDATE*, and *DELETE* statements are candidates. The interface used to execute them does not matter; it can be OCI, SQLJ, PL/SQL, or another interface. Also, *INSERT INTO...SELECT* from external tables can be resumable.
  + **DDL**: The following statements are candidates for resumable execution:
    - *CREATE TABLE ... AS SELECT*
    - *CREATE INDEX*
    - *ALTER INDEX ... REBUILD*
    - *ALTER TABLE ... MOVE PARTITION*
    - *ALTER TABLE ... SPLIT PARTITION*
    - *ALTER INDEX ... REBUILD PARTITION*
    - *ALTER INDEX SPLIT PARTITION ... SPLIT PARTITION*
    - *CREATE MATERIALIZED VIEW*

### Auditing

Auditing, which means capturing and storing information about what is happening in the system, increases the amount of work the system must do. Auditing must be focused so that only events that are of interest are captured. Properly focused auditing has minimal impact on system performance. Improperly focused auditing can significantly affect performance.

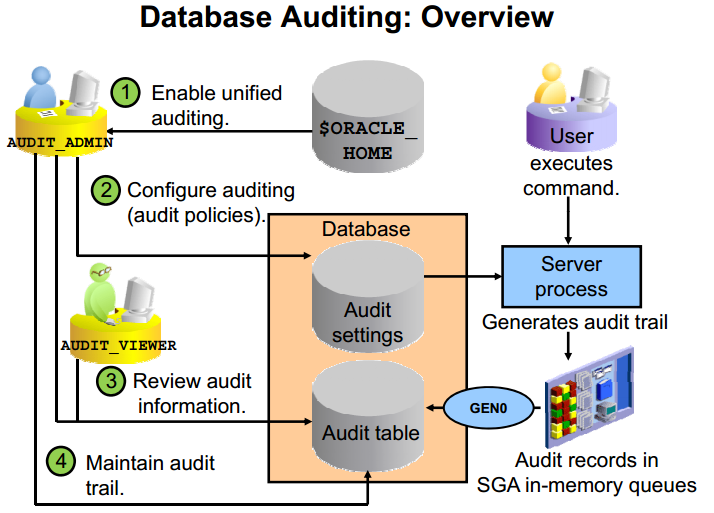
* **Mandatory auditing**: All Oracle databases audit certain actions regardless of other audit options or parameters. The reason for mandatory audit logs is that the database needs to record some database activities, such as connections by privileged users.
* **Standard database auditing**: Select the objects and privileges that you want to audit and create the appropriate audit policies.
* **Value-based auditing**: Extends standard database auditing, capturing not only the audited event that occurred but also the actual values that were inserted, updated, or deleted. Value-based auditing is implemented through database triggers.
* **Fine-grained auditing (FGA)**: Extends standard database auditing, capturing the actual SQL statement that was issued rather than only the fact that the event occurred.

Through the use of auditing policies, you can configure audit settings for the following activities:

* Logging on to the database and the use of privileges and roles
* Executing SQL statements against specific database objects
* Application context values
  + Utilities and features:
  + Oracle Data Pump
  + Oracle Database Real Application Security
  + Oracle Database Vault
  + Oracle Label Security
  + Oracle Recovery Manager
  + Oracle SQL\*Loader Direct Load

**Unified auditing**

**New feature**



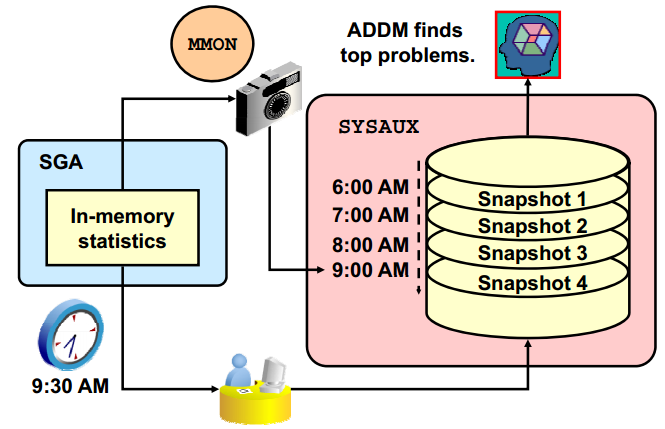
Prior to Oracle Database 12c, audit records from various sources were stored in different locations. Oracle Database 12c supports unified auditing, in which all audit records are stored in a single audit table.

When you create a new Oracle Database 12c database, mixed mode auditing is enabled. This mode enables you to use the auditing features available before Oracle Database 12c and also the unified auditing features. Mixed mode auditing is enabled by default through the ORA\_SECURECONFIG predefined auditing policy for newly created databases.

A user must be granted one of the following roles to perform auditing:

* AUDIT\_ADMIN enables the user to:
  + Create unified and fine-grained audit policies
  + Execute the AUDIT and NOAUDIT SQL statements
  + View audit data
  + Manage the audit trail (table in the AUDSYS schema)
* AUDIT\_VIEWER enables the user to:
  + View and analyze audit data

### AWR



The Automatic Workload Repository (AWR) is a collection of persistent system performance statistics owned by SYS. The AWR resides in the SYSAUX tablespace.

A snapshot is a set of performance statistics captured at a certain time and stored in the AWR. Each snapshot is identified by a snapshot sequence number (SNAP\_ID) that is unique in the AWR. By default, snapshots are generated every 60 minutes. You can adjust this frequency by changing the snapshot INTERVAL parameter. Because the database advisors rely on these snapshots, be aware that adjustment of the interval setting can affect diagnostic precision. For example, if the INTERVAL is set to four hours, you may miss transient events that would be noticeable in 60-minute intervals.

# III. Oracle Database New feature

## Online rename and relocation of an active data file

Unlike in the previous releases, a data file migration or renaming in Oracle database 12c R1 no longer requires a number of steps i.e. putting the tablespace in READ ONLY mode, followed by data file offline action. In 12c R1, a data file can be renamed or moved online simply using the ALTER DATABASE MOVE DATAFILE SQL statement. While the data file is being transferred, the end user can perform queries, DML and DDL tasks. Additionally, data files can be migrated between storages e.g. from non-ASM to ASM and vice versa.

* Rename a data file:

SQL> ALTER DATABASE MOVE DATAFILE '/u00/data/users01.dbf' TO '/u00/data/users\_01.dbf';

* Migrate a data file from non-ASM to ASM:

SQL> ALTER DATABASE MOVE DATAFILE '/u00/data/users\_01.dbf' TO '+DG\_DATA';

* Migrate a data file from one ASM disk group to another:

SQL> ALTER DATABASE MOVE DATAFILE '+DG\_DATA/DBNAME/DATAFILE/users\_01.dbf' TO '+DG\_DATA\_02';

* Overwrite the data file with the same name, if it exists at the new location:

SQL> ALTER DATABASE MOVE DATAFILE '/u00/data/users\_01.dbf' TO '/u00/data\_new/users\_01.dbf' REUSE;

* Copy the file to a new location whilst retaining the old copy in the old location:

SQL> ALTER DATABASE MOVE DATAFILE '/u00/data/users\_01.dbf' TO '/u00/data\_new/users\_01.dbf' KEEP;

You can monitor the progress while a data file being moved by querying the **v$session\_longops** dynamic view. Additionally, you can also refer the alert.log of the database where Oracle writes the details about action being taken place.

## Online migration of table partition or sub-partition

Migration of a table partition or sub-partition to a different tablespace no longer requires a complex procedure in Oracle 12c R1. In a similar way to how a heap (non-partition) table online migration was achieved in the previous releases, a table partition or sub-partition can be moved to a different tablespace online or offline. When an ONLINE clause is specified, all DML operations can be performed without any interruption on the partition|sub-partition which is involved in the procedure. In contrast, no DML operations are allowed if the partition|sub-partition is moved offline.

Here are some working examples:

SQL> ALTER TABLE table\_name MOVE PARTITION|SUBPARTITION partition\_name TO tablespace tablespace\_name;

SQL> ALTER TABLE table\_name MOVE PARTITION|SUBPARTITION partition\_name TO tablespace tablespace\_name UPDATE INDEXES ONLINE;

The first example is used to move a table partition|sub-partition to a new tablespace offline. The second example moves a table partition/sub-partitioning online maintaining any local/global indexes on the table. Additionally, no DML operation will get interrupted when ONLINE clause is mentioned.

Important notes:

* The UPDATE INDEXES clause will avoid any local/global indexes going unusable on the table.
* Table online migration restriction applies here too.
* There will be locking mechanism involved to complete the procedure, also it might leads to performance degradation and can generate huge redo, depending upon the size of the partition, sub-partition.

## Invisible columns

In Oracle 11g R1, Oracle introduced a couple of good enhancements in the form of invisible indexes and virtual columns. Taking the legacy forward, invisible column concepts has been introduced in Oracle 12c R1. I still remember, in the previous releases, to hide important data –columns from being displayed in the generic queries– we used to create a view hiding the required information or apply some sort of security conditions.

In 12c R1, you can now have an invisible column in a table. When a column is defined as invisible, the column won’t appear in generic queries, unless the column is explicitly referred to in the SQL statement or condition, or DESCRIBED in the table definition. It is pretty easy to add or modify a column to be invisible and vice versa:

SQL> CREATE TABLE emp (eno number(6), ename name varchar2(40), sal number(9) INVISIBLE);

SQL> ALTER TABLE emp MODIFY (sal visible);

You must explicitly refer to the invisible column name with the INSERT statement to insert the database into invisible columns. A virtual column or partition column can be defined as invisible too. However, temporary tables, external tables and cluster tables won’t support invisible columns.

## Multiple indexes on the same column

Pre Oracle 12c, you can’t create multiple indexes either on the same column or set of columns in any form. For example, if you have an index on column {a} or columns {a,b}, you can’t create another index on the same column or set of columns in the same order. In 12c, you can have multiple indexes on the same column or set of columns as long as the index type is different. However, only one type of index is usable/visible at a given time. In order to test the invisible indexes, you need to set the optimizer\_use\_use\_invisible\_indexes=true.

Here’s an the example:

SQL> CREATE INDEX emp\_ind1 ON EMP(ENO,ENAME);

SQL> CREATE BITMAP INDEX emp\_ind2 ON EMP(ENO,ENAME) INVISIBLE;

## DDL logging

There was no direction option available to log the DDL action in the previous releases. In 12cR1, you can now log the DDL action into xml and log files. This will be very useful to know when the drop or create command was executed and by who. The ENABLE\_DDL\_LOGGING initiation parameter must be configured in order to turn on this feature. The parameter can be set at the database or session levels. When this parameter is enabled, all DDL commands are logged in an xml and a log file under the $ORACLE\_BASE/diag/rdbms/DBNAME/log|ddl location. An xml file contains information, such as DDL command, IP address, timestamp etc. This helps to identify when a user or table dropped or when a DDL statement is triggered.

To enable DDL logging

SQL> ALTER SYSTEM|SESSION SET ENABLE\_DDL\_LOGGING=TRUE;

The following DDL statements are likely to be recorded in the xml/log file:

CREATE|ALTER|DROP|TRUNCATE TABLE

DROP USER

CREATE|ALTER|DROP PACKAGE|FUNCTION|VIEW|SYNONYM|SEQUENCE

## Temporary Undo

Each Oracle database contains a set of system related tablespaces, such as, SYSTEM, SYSAUX, UNDO & TEMP, and each are used for different purposes within the Oracle database. Pre Oracle 12c R1, undo records generated by the temporary tables used to be stored in undo tablespace, much similar to a general/persistent table undo records. However, with the temporary undo feature in 12c R1, the temporary undo records can now be stored in a temporary table instead of stored in undo tablespace. The prime benefits of temporary undo includes: reduction in undo tablespace and less redo data generation as the information won’t be logged in redo logs. You have the flexibility to enable the temporary undo option either at session level or database level.

Enabling temporary undo

To be able to use the new feature, the following needs to be set:

* Compatibility parameter must be set to 12.0.0 or higher
* Enable TEMP\_UNDO\_ENABLED initialization parameter
* Since the temporary undo records now stored in a temp tablespace, you need to create the temporary tablespace with sufficient spaceFor session level, you can use: ALTER SESSION SET TEMP\_UNDO\_ENABLE=TRUE;

Query temporary undo information

The dictionary views listed below are used to view/query the information/statistics about the temporary undo data:

* V$TEMPUNDOSTAT
* DBA\_HIST\_UNDOSTAT
* V$UNDOSTAT

To disable the feature, you simply need to set the following:

SQL> ALTER SYSTEM|SESSION SET TEMP\_UNDO\_ENABLED=FALSE;

## Backup specific user privilege

In 11g R2, SYSASM privilege was introduced to perform ASM specific operations. Similarly, backup and recovery tasks specific privilege SYSBACKUP has been introduced in 12c to execute backup and recovery commands in Recovery Manager (RMAN). Therefore, you can create a local user in the database and grant the SYSBACKUP privilege to perform any backup and recovery related tasks in RMAN without being granting the SYSDBA privilege.

$ ./rman target "username/password as SYSBACKUP"

## Restricting PGA size

Pre Oracle 12c R1, there was no option to limit and control the PGA size. Although, you set a certain size to PGA\_AGGREGATE\_TARGET initialization parameter, Oracle could increase/reduce the size of the PGA dynamically based on the workload and requirements. In 12c, you can set a hard limit on PGA by enabling the automatic PGA management, which requires PGA\_AGGREGATE\_LIMIT parameter settings. Therefore, you can now set the hard limit on PGA by setting the new parameter to avoid excessive PGA usage.

SQL> ALTER SYSTEM SET PGA\_AGGREGATE\_LIMIT=2G;

SQL> ALTER SYSTEM SET PGA\_AGGREGATE\_LIMIT=0; --disables the hard limit

## Table partition maintenance enhancements

In this section, you will learn other enhancements relating to table partitioning.

### Adding multiple new partitions

Before Oracle 12c R1, it was only possible to add one new partition at a time to an existing partitioned table. To add more than one new partition, you had to execute an individual ALTER TABLE ADD PARTITION statement to every new partition. Oracle 12c provides the flexibility to add multiple new partitions using a single ALTER TABLE ADD PARTITION command. The following example explains how to add multiple new partitions to an existing partitioned table:

SQL> CREATE TABLE emp\_part

(eno number(8), ename varchar2(40), sal number (6))

PARTITION BY RANGE (sal)

(PARTITION p1 VALUES LESS THAN (10000),

PARTITION p2 VALUES LESS THAN (20000),

PARTITION p3 VALUES LESS THAN (30000)

);

* Now lets add a couple of new partitions:

SQL> ALTER TABLE emp\_part ADD PARTITION

PARTITION p4 VALUES LESS THAN (35000),

PARTITION p5 VALUES LESS THAN (40000);

In the same way, you can add multiple new partitions to a list and system partitioned table, provided that the MAXVALUE partition doesn’t exist.

### How to drop and truncate multiple partitions/sub-partitions

As part of data maintenance, you typically either use drop or truncate partition maintenance task on a partitioned table. Pre 12c R1, it was only possible to drop or truncate one partition at a time on an existing partitioned table. With Oracle 12c, multiple partitions or sub-partitions can be dropped or merged using a single ALTER TABLE table\_name {DROP|TRUNCATE} PARTITIONS command.

* The following example explains how to drop or truncate multiple partitions on an existing partitioned table:

SQL> ALTER TABLE emp\_part DROP PARTITIONS p4,p5;

SQL> ALTER TABLE emp\_part TRUNCATE PARTITONS p4,p5;

* To keep indexes up-to-date, use the UPDATE INDEXES or UPDATE GLOBAL INDEXES clause, shown below:

SQL> ALTER TABLE emp\_part DROP PARTITIONS p4,p5 UPDATE GLOBAL INDEXES;

SQL> ALTER TABLE emp\_part TRUNCATE PARTITIONS p4,p5 UPDATE GLOBAL INDEXES;

If you truncate or drop a partition without the UPDATE GLOBAL INDEXES clause, you can query the column ORPHANED\_ENTRIES in the USER\_INDEXES or USER\_IND\_PARTITIONS dictionary views to find out whether the index contains any stale entries.

### Splitting a single partition into multiple new partitions

The new enhanced SPLIT PARTITION clause in 12c will let you split a particular partition or sub-partition into multiple new partitions using a single command. The following example explains how to split a partition into multiple new partitions:

SQL> CREATE TABLE emp\_part

(eno number(8), ename varchar2(40), sal number (6))

PARTITION BY RANGE (sal)

(PARTITION p1 VALUES LESS THAN (10000),

PARTITION p2 VALUES LESS THAN (20000),

PARTITION p\_max VALUES LESS THAN (MAXVALUE)

);

SQL> ALTER TABLE emp\_part SPLIT PARTITION p\_max INTO

(PARTITION p3 VALUES LESS THAN (25000),

PARTITION p4 VALUES LESS THAN (30000), PARTITION p\_max);

### Merge multiple partitions into one partition

You can merge multiple partitions to a single partition using a single ALTER TBALE MERGE PARTITIONS statement:

SQL> CREATE TABLE emp\_part

(eno number(8), ename varchar2(40), sal number (6))

PARTITION BY RANGE (sal)

(PARTITION p1 VALUES LESS THAN (10000),

PARTITION p2 VALUES LESS THAN (20000),

PARTITION p3 VALUES LESS THAN (30000),

PARTITION p4 VALUES LESS THAN (40000),

PARTITION p5 VALUES LESS THAN (50000),

PARTITION p\_max (MAXVALUE)

);

SQL> ALTER TABLE emp\_part MERGE PARTITIONS p3,p4,p5 INTO PARTITION p\_merge;

If the range falls in the sequence, you can use the following example:

SQL> ALTER TABLE emp\_part MERGE PARTITIONS p3 TO p5 INTO PARTITION p\_merge;

## Database upgrade improvements

Whenever a new Oracle version is announced, the immediate challenge that every DBA confronts is the upgrade process. In this section, I will explain the two new improvements introduced for upgrading to 12c.

**Pre-upgrade script**

SQL> @$ORACLE\_12GHOME/rdbms/admin/preupgrd.sql

The above script generates a log file and a [pre/post]upgrade\_fixup.sql script. All these files are located under the $ORACLE\_BASE/cfgtoollogs directory. Before you continue with the real upgrade procedure, you should run through the recommendations mentioned in the log file and execute the scripts to fix any issues.

Note: Ensure you copy the preupgrd.sql and utluppkg.sql scripts from the 12c Oracle home/rdbms/admin directory to the current Oracle database/rdbms/admin location.

**Parallel-upgrade utility**

The database upgrade duration is directly proportional to the number of components that are configured on the database, rather than the database size. In previous releases, there was no direct option or workaround available to run the upgrade process in parallel to quickly complete the overall upgrade procedure.

The catctl.pl (parallel-upgrade utility) that replaces the legacy catupgrd.sql script in 12c R1 comes with an option to run the upgrade procedure in parallel mode to improve the overall duration required to complete the procedure.

The following procedure explains how to initiate the parallel (with 3 processes) upgrade utility; you need to run this after you STARTUP the database in UPGRADE mode:

cd $ORACLE\_12\_HOME/perl/bin

$ ./perl catctl.pl –n 3 -catupgrd.sql

The above two steps need to be run explicitly when a database is upgraded manually. However, the **DBUA** inherits the both new changes.

## Restore/Recover data files over the network

Yet another great enhancement in 12c R1. You can now restore or recover a data file, control file, spfile, tablespace or entire database between primary and standby databases using a SERVICE name. This is particularly useful to synchronize the primary and standby databases.

When there is a pretty long gap found between the primary and standby database, you no longer require the complex roll-forward procedure to fill the gap between the primary and standby. RMAN is able to perform standby recovery getting the incremental backups through the network and applying them to the physical standby database. Having said that, you can directly copy the required data files from the standby location to the primary site using the SERVICE name e.g. in the case of a data file, tablespace lost on the primary database, or without actually restoring the data files from a backup set.

The following procedure demonstrates how to perform a roll forward using the new features to synchronize the standby database with its primary database:

On the physical standby database:

./rman target "username/password@standby\_db\_tns as SYSBACKUP"

RMAN> RECOVER DATABASE FROM SERVICE primary\_db\_tns USING COMPRESSED BACKUPSET;

The above example uses the primary\_db\_tns connect string defined on the standby database, connects to the primary database, performs an incremental backup, transfers these incremental backups over standby destination, and then applies these files to the standby database to synchronize the standby. However, you need to ensure you have configured primary\_db\_tns to point to the primary database on the standby database side.

In the following example, I will demonstrate a scenario to restore a lost data file on the primary database by fetching the data file from the standby database:

On the primary database:

./rman target "username/password@primary\_db\_tns as SYSBACKUP"

RMAN> RESTORE DATAFILE ‘+DG\_DISKGROUP/DBANME/DATAFILE/filename’ FROM SERVICE standby\_db\_tns;

## Data Pump enhancements

This part of the section will focus on the important enhancements introduced in data pumps. There are quite a few useful additions, such as converting view into a table while exporting and turning off logging while import.

### Turn off redo log generation

The new TRANSFORM option introduced in data pumps import provides the flexibility to turn off the redo generation for the objects during the course of import. When DISABLE\_ARCHIVE\_LOGGING values is specified with the TRANSFORM option, redo generation for the objects in the context will be turned off during the entire import duration. This feature provides a great relief when importing large tables, and reduces the excessive redo generation, which results in quicker imports. This attribute applies to tables and indexes.

This example demonstrates this feature:

$ ./impdp directory=dpump dumpfile=abcd.dmp logfile=abcd.log TRANSFORM=DISABLE\_ARCHIVE\_LOGGING:Y

### Transport view as table

This is another improvement in the data pumps. With the new VIEWS\_AS\_TABLES option, you can unload the view data into a table. The following example describes how to unload views data into a table during export:

$ ./expdp directory=dpump dumpfile=abcd.dmp logfile=abcd.log views\_as\_tables=my\_view:my\_table

## Real-time ADDM analysis

Analyzing past and current database health statuses through a set of automatic diagnostic tools such as AWR, ASH and ADDM is part of every DBAs life. Though each individual tool can be used at various levels to measure the database’s overall heath and performance, no tool can be used when the database is unresponsive or totally hung.

When you encounter an unresponsive database or hung state, and if you have configured Oracle Enterprise Manager 12c Cloud Control, you can diagnose serious performance issues. This would give you a good picture about what’s currently going on in the database, and might also provide a remedy to resolve the issue.

The following step-by-step procedure demonstrates how to analyze the situation on the Oracle EM 12c Cloud Control :

* Select the Emergency Monitoring option from the Performance menu on the Access the Database Home page.This will show the top blocking sessions in the Hang Analysis table.
* Select the Real-Time ADDM option from the Performance to perform Real-time ADDM analysis.
* After collecting the performance data, click on the Findings tab to get the interactive summary of all the findings.

## Gathering statistics concurrently on multiple tables

In previous Oracle database editions, whenever you execute a DBMS\_STATS procedure to gather table, index, schema or database level statistics, Oracle used to collect stats one table at a time. If the table is big enough, then increasing the parallelism was recommended. With 12c R1, you can now collect stats on multiple tables, partitions and sub partitions concurrently. Before you start using it, you must set the following at the database level to enable the feature:

SQL> ALTER SYSTEM SET RESOURCE\_MANAGER\_PLAN='DEFAULT\_MAIN';

SQL> ALTER SYSTEM SET JOB\_QUEUE\_PROCESSES=4;

SQL> EXEC DBMS\_STATS.SET\_GLOBAL\_PREFS('CONCURRENT', 'ALL');

SQL> EXEC DBMS\_STATS.GATHER\_SCHEMA\_STATS('SCOTT');

## Useful Enchanment to developers

### Truncate table CASCADE

In the previous releases, there wasn’t a direct option provided to truncate a master table while it is referred to by the child tables and child records exist. The TRUNCATE TABLE with CASCADE option in 12c truncates the records in the master table and automatically initiates recursive truncate on child tables too, subject to foreign key reference as DELETE ON CASCADE. There is no CAP on the number of recursive levels as it will apply on all child, grand child and great grandchild etc.

This enhancement gets rid of the prerequisite to truncate all child records before truncating a master table. The new CASCADE clause can also be applied on table partitions and sub-partitions etc.

SQL> TRUNCATE TABLE <table\_name> CASCADE;

SQL> TRUNCATE TABLE <table\_name> PARTITION <partition\_name> CASCADE;

ORA-14705 error will be thrown if no ON DELETE CASCADE option is defined with the foreign keys of the child tables.

### Session level sequences

A new SESSION level database sequence can be created now in 12c to support the session level sequence values. These types of sequences are most useful and suitable on global temporary tables that have session level existence.

Session level sequences produce a unique range of values that are limited within the session, not across the sessions. Once the session ends, the state of the session sequences also goes away. The following example explains creating a session level sequence:

SQL> CREATE SEQUENCE my\_seq START WITH 1 INCREMENT BY 1 SESSION;

SQL> ALTER SEQUENCE my\_seq GLOBAL|SESSION;

The CACHE, NOCACHE, ORDER or NOORDER clauses are ignored for SESSION level sequences.

### ROW limiting for Top-N result queries

There are various indirect approaches/methods exist to fetch Top-N query results for top/bottom rows in the previous releases. In 12c, retrieving Top-N query results for top/bottom rows simplified and become straight forward with the new FETCH FIRST|NEXT|PERCENT clauses.

* In order to retrieve top 10 salaries from EMP table, use the following new SQL statement:

SQL> SELECT eno,ename,sal FROM emp ORDER BY SAL DESC

FETCH FIRST 10 ROWS ONLY;

* The following example fetches all similar records of Nth row. For example, if the 10th row has salary of 5000 value, and there are other employees whose salary matches with the Nth value, the will also be fetched upon mentioning WITH TIES clause.

SQL> SELECT eno,ename,sal FROM emp ORDER BY SAL DESC

FETCH FIRST 10 ROWS ONLY WITH TIES;

* The following example limits the fetch to 10 per cent from the top salaries in the EMP table:

SQL> SELECT eno,ename,sal FROM emp ORDER BY SAL DESC

FETCH FIRST 10 PERCENT ROWS ONLY;

* The following example offsets the first 5 rows and will display the next 5 rows from the table:

SQL> SELECT eno,ename,sal FROM emp ORDER BY SAL DESC

OFFSET 5 ROWS FETCH NEXT 5 ROWS ONLY;

* All these limits can be very well used within the PL/SQL block too.

BEGIN

SELECT sal BULK COLLECT INTO sal\_v FROM EMP

FETCH FIRST 100 ROWS ONLY;

END;

### Miscellaneous SQL\*Plus enhancements

Implicit Results on SQL\*Plus: SQL\*Plus in 12c returns results from an implicit cursor of a PL/SQL block without actually binding it to a RefCursor. The new dbms\_sql.return\_result procedure will return and formats the results of SELECT statement query specified within PL/SQL block. The following code descries the usage:

SQL> CREATE PROCEDURE mp1 as res1 sys\_refcursor;

BEGIN

open res1 for SELECT eno,ename,sal FROM emp;

dbms\_sql.return\_result(res1);

END;

SQL> execute mp1;

When the procedure is executed, it return the formatted rows on the SQL\*Plus.

Display invisible columns: setting the following on the SQL\*Plus prompt:

SQL> SET COLINVISIBLE ON|OFF

The above setting is only valid for DESCRIBE command. It has not effect on the SELECT statement results on the invisible columns.

### In Line PL/SQL Functions in SQL - WITH clause improvements

In 12c, you can have faster running PL/SQL function/procedure in SQL, that are defined and declared within the WITH clause of SQL statements. The following examples demonstrate how to define and declare a procedure or function within the WITH clause:

WITH

PROCEDURE|FUNCTION test1 (…)

BEGIN

<logic>

END;

SELECT <referece\_your\_function|procedure\_here> FROM table\_name;

/

Although you can’t use the WITH clause directly in the PL/SQL unit, it can be referred through a dynamic SQL within that PL/SQL unit.

### Extended data types

In 12c, the data type VARCHAR2, NAVARCHAR2, and RAW size will support up to 32,767 bytes in contrast to 4,000 and 2,000 in the earlier releases. The extended character size will reduce the use of going for LOB data types, whenever possible. In order to enable the extended character size, you will have to set the MAX\_STRING\_SIZE initialization database parameter to EXTENDED.

Note: Once modified, you can’t change the settings back to STANDARD

### Generated as identity / Sequence Replacement

In old version of oracle database if you want to create automatic generated number you have to create sequence and use attribute nextval.

But with oracle database 12c this concept is changed new features add when you create table called generated as identity.

SQL> create table test (test\_id number generated as identity , test\_name varchar2(20));

SQL> desc test ;

Name Null? Type

----------------------------------------- -------- --------------

TEST\_ID NOT NULL NUMBER

TEST\_NAME VARCHAR2(20)

SQL> insert into test values (1,'osama');

insert into test values (1,'osama')

\*

ERROR at line 1:

ORA-32795: cannot insert into a generated always identity column

The TEST\_ID Column will be inserting automatically no need to use in insert command.

SQL> insert into test (TEST\_NAME) values ( 'Jennifer' );

1 row created.

SQL> select \* from test ;

TEST\_ID TEST\_NAME

---------- --------------------

1 Jennifer

## Additions/Enhancements in Automatic Storage Management (ASM)

### Flex ASM

In a typical Grid Infrastructure installation, each node will have its own ASM instance running and act the as the storage container for the databases running on the node. There is a single point-of-failure threat with this setup. For instance, if the ASM instance on the node suffers or fails all the databases and instances running on the node will be impacted. To avoid ASM instance single-point-failure, Oracle 12c provides a Flex ASM feature. The Flex ASM is a different concept and architecture all together. Only a fewer number of ASM Instances need to run on a group of servers in the cluster. When an ASM instance fails on a node, Oracle Clusterware automatically starts surviving (replacement) ASM instance on a different node to maintain availability. In addition, this setup also provides ASM instance load balancing capabilities for the instances running on the node. Another advantage of Flex ASM is that it can be configured on a separate node.

When you choose Flex Cluster option as part of the cluster installation, Flex ASM configuration will be automatically selected as it is required by the Flex Cluster. You can also have traditional cluster over Flex ASM. When you decide to use Flex ASM, you must ensure the required networks are available. You can choose the Flex ASM storage option as part of Cluster installation, or use ASMCA to enable Flex ASM in a standard cluster environment.

The following command shows the current ASM mode:

$ ./asmcmd showclustermode

$ ./srvctl config asm

Or connect to the ASM instances and query the INSTANCE\_TYPE parameter. If the output value is ASMPROX, then, the Flex ASM is configured.

### Increased ASM storage limits

The ASM storage hard limits on maximum ASM disk groups and disk size has been drastically increased. In 12cR1, ASM support 511 ASM disk groups against 63 ASM disk groups in 11gR2. Also, an ASM disk can be now 32PB size against 20PB in 11gR2.

### Tuning ASM rebalance operations

The new EXPLAIN WORK FOR statement in 12c measures the amount of work required for a given ASM rebalance operation and inputs the result in V$ASM\_ESTIMATE dynamic view. Using the dynamic view, you can adjust the POWER LIMIT clause to improve the rebalancing operation work. For example, if you want to measure the amount of work required for adding a new ASM disk, before actually running the manual rebalance operation, you can use the following:

SQL> EXPLAIN WORK FOR ALTER DISKGROUP DG\_DATA ADD DISK data\_005;

SQL> SELECT est\_work FROM V$ASM\_ESTIMATE;

SQL> EXPLAIN WORK SET STATEMENT\_ID='ADD\_DISK' FOR ALTER DISKGROUP DG\_DATA AD DISK data\_005;

SQL> SELECT est\_work FROM V$ASM\_ESTIMATE WHERE STATEMENT\_ID = 'ADD\_DISK’;

You can adjust the POWER limit based on the output you get from the dynamic view to improve the rebalancing operations.

### ASM Disk Scrubbing

The new ASM Disk Scrubbing operation on a ASM diskgroup with normal or high redundancy level, verifies the logical data corruption on all ASM disks of that ASM diskgroup, and repairs the logical corruption automatically, if detected, using the ASM mirror disks. The disk scrubbing can be performed at disk group, specified disk or on a file and the impact is very minimal. The following examples demonstrate the disk scrubbing scenario:

SQL> ALTER DISKGROUP dg\_data SCRUB POWER LOW:HIGH:AUTO:MAX;

SQL> ALTER DISKGROUP dg\_data SCRUB FILE '+DG\_DATA/MYDB/DATAFILE/filename.xxxx.xxxx'

REPAIR POWER AUTO;

### Active Session History (ASH) for ASM

The V$ACTIVE\_SESSION\_HISOTRY dynamic view now provides the active session sampling on ASM instance too. However, the use of diagnostic pack is subject to the license.

## Additions/Enhancements in Grid Infrastructure

### Flex Clusters

Oracle 12c support two types of cluster configuration at the time of Clusterware installation: Traditional Standard Cluster and Flex cluster. In a traditional standard cluster, all nodes in a cluster are tightly integrated to each other and interact through a private network and can access the storage directly. On the other hand, the Flex Cluster introduced two types of nodes arranged in Hub and Leaf nodes architecture. The nodes arranged in Hub nodes category are similar to the traditional standard cluster, i.e. they are interconnected to each other through a private network and have the directly storage read/write access. The Leaf nodes are different from the Hub nodes. They don’t need to have direct access to the underlying storage; rather they access the storage/data through Hub nodes.

You can configure Hub nodes up to 64, and Leaf nodes can be many. In an Oracle Flex Cluster, you can have Hub nodes without having Leaf nodes configured, but no Leaf nodes exist without Hub nodes. You can configure multiple Leaf nodes to a single Hub node. In Oracle Flex Cluster, only Hub nodes will have direct access to the OCR/Voting disks. When you plan large scale Cluster environments, this would be a great feature to use. This sort of setup greatly reduces interconnect traffic, provides room to scale up the cluster to the traditional standard cluster.

There are two ways to deploy the Flex Cluster:

* While configuring a brand new cluster
* Upgrade a standard cluster mode to Flex Cluster

If you are configuring a brand new cluster, you need to choose the type of cluster configuration during step 3, select Configure a Flex Cluster option and you will have to categorize the Hub and Leaf nodes on Step 6. Against each node, select the Role, Hub or Leaf, and optionally Virtual Hostname too.

The following steps are required to convert a standard cluster mode to Flex Cluster mode:

* Get the current status of the cluster using the following command:

$ ./crsctl get cluster mode status

* Run the following command as the root user:

$ ./crsctl set cluster mode flex

$ ./crsctl stop crs

$ ./crsctl start crs –wait

* Change the node role as per your design

$ ./crsctl get node role config

$ ./crsctl set node role hub|leaf

$ ./crsctl stop crs

$ ./crsctl start crs -wait

Note the following:

* You can’t revert back from Flex to Standard cluster mode
* Cluster node mode change requires cluster stack stop/start
* Ensure GNS is configured with a fixed VIP

### OCR backup in ASM disk group

With 12c, OCR can be now be backed-up in ASM disk group. This simplifies the access to the OCR backup files across all nodes. In case of OCR restore, you don’t need to worry about which node the OCR latest backup is on. One can simply identify the latest backup stored in the ASM from any node and can perform the restore easily.

The following example demonstrates how to set the ASM disk group as OCR backup location:

$ ./ocrconfig -backuploc +DG\_OCR

### IPv6 support

With Oracle 12c, Oracle now supports IPv4 and IPv6 network protocol configuration on the same network. You can now configure public network (Public/VIP) either on IPv4, IPv6 or combination protocol configuration. However, ensure you use the same set of IP protocol configuration across all nodes in a cluster.

## Additions/Enhancements in RAC (database)

### What-If command evaluation

Using the new What-if command evaluation (-eval) option with srvctl, one can now determine the impact of running the command. This new addition to the srvctl command, will let you simulate the command without it actually being executed or making any changes to the current system. This is particularly useful in a situation when you want to make a change to an existing system and you’re not sure of the outcome. Therefore, the command will provide the effect of making the change. The –eval option also can be used with crsctl command.

For example, if you want to know what will happen if you stop a particular database, you can use the following example:

$ ./srvctl stop database –d MYDB –eval

$ ./crsctl eval modify resource <resource\_name> -attr “value”

### Miscellaneous srvctl improvements

There are a few new additions to the srvctl command. The following demonstrates the new addition to stop/start database/instance resources on the cluster:

srvctl start database|instance –startoption NOMOUNT|MOUNT|OPEN

srvctl stop database|instance –stopoption NOMOUNT|MOUNT|OPEN

# IV. RMAN new feauture

## SQL Interface Improvements

In Oracle 12c, you can run SQL commands in RMAN without preceding the command with the SQL keyword.  You also no longer need to enclose the SQL command in quotes.

* The RMAN DESCRIBE provides the same functionality of SQL\*Plus DESCRIBE:

RMAN> desc dba\_profiles;

Name                              Null?           Type

---------------------------- --------         ----------------------

PROFILE                           NOT NULL     VARCHAR2(128)

RESOURCE\_NAME                 NOT NULL     VARCHAR2(32)

RESOURCE\_TYPE                                      VARCHAR2(8)

LIMIT                                              VARCHAR2(128)

COMMON                                             VARCHAR2(3)

* You can run SQL statements from RMAN command prompt:

RMAN> select sysdate from dual;

SYSDATE

---------

10-SEP-14

* You can run DDL/DML Commands from RMAN Command prompt, but note that in order to insert you need to use:

RMAN> create table ora\_table(col1 number, col2 varchar2(20));

Statement processed

RMAN> insert into ora\_table values (1,'Test');

Statement processed

RMAN> update ora\_table set col1=2;

Statement processed

RMAN> drop table ora\_table;

Statement processed

* The user can SHUTDOWN/STARTUP the database and also can use ALTER commands:

RMAN> shutdown immediate

database closed

database dismounted

Oracle instance shut down

RMAN> startup mount

connected to target database (not started)

Oracle instance started

database mounted

Total System Global Area    1610612736 bytes

Fixed Size                 2924928 bytes

Variable Size               520097408 bytes

Database Buffers         1073741824 bytes

Redo Buffers                13848576 bytes

RMAN> alter database open;

Statement processed

## SYSBACKUP Privilege

Prior to 12c, users needed SYSDBA privilege to backup the database. The new SYSBACKUP privilege allows the user the permissions to perform only backup operations.

The SYSBACKUP privilege allows the DBA to perform RMAN backup commands without additional privileges. Using this new role in 12c, you can segregate Administration and Backup operations.

With RMAN you have same authentication options that are available with SQL\*Plus, which are Operating system authentication and password file authentication.

* To connect to RMAN using Operating system Authentication Authentication with the SYSBACKUP Privilege use:

$ rman target ' "/ as sysbackup" '

* Authentication with the SYSDBA Privilege use:

$ rman target ' "/ as sysdba" '

* You can also implicitly connect using below command

$ rman target /

* To Connect to RMAN using Password file Authentication Authentication with the SYSBACKUP Privilege use:

$ rman target1 ‘ “bkpadm@DB1 as sysbackup” ‘

**Where bkpadm is the user and should have SYSDBA privilege.**

* Authentication with the SYSDBA Privilege

$ rman target ‘ “sysadm@DB1 as sysdba” ‘

* You can implicitly connect using below command. Where sysadm is the user and should have SYSDBA privilege.

$ rman target sysadm@DB1

Note that SYSBACKUP does not include data access privilege, such as SELECT ANY TABLE. When you don’t specify the role explicitly then the default used is AS SYSDBA.

## Support for multitenant container and pluggable databases

The multitenant container database (CDB) and pluggable databases (PDB) are introduced in Oracle 12c, and RMAN provides full support for backup and recovery. Using RMAN you can back up an entire container database or individual pluggable databases and also can perform point-in-time recovery.

The multitenant architecture manages many databases as one and retains the isolation, resource control of each database. This will help to manage both infrastructure and human resources effectively.



Backing up a container database is similar to backing up a non-container database. When you back up a container database, RMAN backs up the root, pluggable databases in the container, and archive logs.  When you need to restore you can choose the whole container, one or more pluggable databases or the root only.

### Backup the CDB, PDB, and root

You should have SYSBACKUP or SYSDBA privilege to backup any of the databases.

* You can backup the **Container Database (CDB)** as same as non-container database using below command:

RMAN> BACKUP DATABASE plus ARCHIVELOG;

* You can backup the **Pluggable Database (PDB)** using below command:

RMAN> BACKUP PLUGGABLE DATABASE PDB1, PDB2;

* Or connect to pluggable Database in RMAN :

% rman target sys@PDB1  
RMAN> BACKUP DATABASE;

* You can backup the **root** using below command:

RMAN> BACKUP DATABASE ROOT;

### Complete recovery of CDB, PDB and root

You should have SYSBACKUP or SYSDBA privilege to restore any of the databases.  
**Restoring Container Database (CDB)** is similar to non-container database.  
You can restore the whole CDB using below script:

RMAN> RUN {

STARTUP MOUNT;

RESTORE DATABASE;

RECOVER DATABASE;

ALTER DATABASE OPEN;

}

Note that restoring CDB database will restore all the pluggable databases.  
You can **restore only ROOT**Database using below script:

RMAN> RUN {

STARTUP MOUNT;

RESTORE DATABASE ROOT;

RECOVER DATABASE ROOT;

ALTER DATABASE OPEN;

}

You can **restore Pluggable Databases** in two ways. Either you can restore from root container and connect directly to PDB to restore.

Use below script to restore from root. Using this approach you can able to restore and recover multiple PDB’s with a single command.

RMAN > RUN {

RESTORE PLUGGABLE DATABASE PDB1, PDB2;

RECOVER PLUGGABLE DATABASE PDB1, PDB2;

ALTER PLUGGABLE DATABASE PDB1, PDB2 OPEN;

}

Use below script to connect PDB, restore and recover the database. Using this approach you will be able to restore and recover **only** one PDB.

$ rman target=bkpadm@PDB1

RMAN> run{

RESTORE DATABASE;

RECOVER DATABASE;

}

The steps for performing a point-in-time recovery of the CDB or PDB are the same as a normal database. But note that when you perform Point-in-time recovery on the CDB, it will effect on all the PDBs as well.

When you perform point-in-time recovery on a PDB, it will affect that single database.  
The command to perform a point-in-time recovery is:

SET UNTIL TIME "TO\_DATE(’01-Jan-2014 01:00:00’,’DD-MON-YYYY HH24:MI:SS’)";

SET UNTIL SCN 1999945; # alternatively, specify SCN

SET UNTIL SEQUENCE 100; # alternatively, specify log seq

Below are the few examples to ALTER PLUGGABLE DATABASE.

* Use this command to open all PDBs in one command:

ALTER PLUGGABLE DATABASE ALL OPEN;

* Use this command to open all PDBs except PDB3:

ALTER PLUGGABLE DATABASE ALL EXCEPT PDB3 OPEN;

* Use this command to open PDB4,PDB5 in read only mode:

ALTER PLUGGABLE DATABASE PDB4, PDB5 OPEN READ ONLY;

* Use below command to shut down all PDBs in single command:

ALTER PLUGGABLE DATABASE ALL CLOSE IMMEDIATE;

### Backup of Archived redo logs

You can back up archive logs when they connect to root as a *common* user with SYSDBA or SYSBACKUP privilege, but you cannot back up or delete archive logs when you connect to PDB as a *local* user with SYSDBA or SYSBACKUP privilege.

You are only able to switch the archived logs when you connect to the root of a CDB, but you cannot switch archived redo logs when connected to a PDB.

If you have more than one archive log destination, when you use RMAN to backup the archive redo logs it backs up only one copy of the archived redo logs. RMAN does not include more than one copy because multiple destinations will have same log sequence number.

You can use any of the below commands to backup the archived redo logs  
The command below backs up the database and all the archived redo logs:

RMAN > BACKUP DATABASE PLUS ARCHIVELOG;

The command below only backs up one copy of the sequence number for all archived redo logs.

RMAN> BACKUP ARCHIVELOG ALL;

## DUPLICATE enhancements:

When you duplicate a database using RMAN DUPLICATE, the database is created and opened with RESETLOGS mode. With Oracle database 12c, you can specify that the database must not be opened with “NOOPEN” clause.

This NOOPEN clause useful under following situations:

* If you need to make changes to initialization parameters such as block change tracking, flashback database settings
* Opening the database conflict with other source database
* If you plan to create database for upgrade and want to open in upgrade mode

The command below creates duplicate database, but it will not open.

RMAN> DUPLICATE TARGET DATABASE TO DB1

FROM ACTIVE DATABASE NOOPEN;

## Multisection Backup Improvements

The multisection backup functionality was introduced in Oracle 11g to handle large data file backups. Using this functionality RMAN can break up a large file into sections during the backup and recovery, which can improve the performance of large datafiles backup. You can select the size using the SECTION SIZE keyword and each channel will create separate files within the backup set, and backup the database in parallel. This functionality supports only backup sets in 11g.

In Oracle 12c, the multisection backup supports incremental backups and image copies, including backup sets (introduced in 11g). This functionality can only be used for data files, you cannot use this to backup control files.

If the SECTION SIZE that you selected is larger than the actual file then RMAN does not use multisection backup. If you specify a small SECTION SIZE that produces more than 256 sections then RMAN increases the SECTION SIZE to a value that results 256 sections.

* The following example creates a multisection backup of the database using image copies.

RMAN> BACKUP AS COPY SECTION SIZE 1024M DATABASE;

RMAN> BACKUP INCREMENTAL LEVEL 1 SECTION SIZE 1024M DATABASE;

* The following example creates multisection incremental level1 backup

RMAN> BACKUP INCREMENTAL LEVEL 1 SECTION SIZE 1024M DATABASE;

To improve the backup performance, use unused block compression and block change tracking in conjunction with multisection incremental backups.

## Restoring and Recovering Files over Network

Using RMAN you can restore and recover a database, datafile, controlfile, tablespace or spfile over the network from a physical standby database. To restore the database over the network, use the RESTORE… FROM SERVICE command and use the RECOVER…FROM SERVICE command to recover the database over the network. The FROM SERVICE clause specifies the service name of the physical standby.

You can also use multisection, compression and encryption to improve backup and restore performance.

* Use SECTION SIZE with RESTORE command to perform multisection restore
* Use SET ENCRYPTION clause before the RESTORE command to specify the encryption
* Use USING COMPRESSED BACKUPSET clause to compress backup sets

This feature is useful to synchronize primary and standby database. Here are the few scenarios

* Roll-forward a physical standby database to sync with the primary database
* Restore the primary database using physical standby database.
* Restore physical standby database using the primary database.

In the following example restoring data file over the network from physical standby to primary database:

* Connected to primary database implicitly

RMAN> CONNECT TARGET /

* Backup sets encrypted using AES128 encryption algorithm

RMAN> SET ENCRYPTION ALGORITHM 'AES128';

* Restoring the datafile on the primary using datafile on physical database with service “standby\_db”

RMAN> RESTORE DATAFILE '/db1/oradata/users.dbf'

FROM SERVICE standby\_db SECTION SIZE 1024M;

## Storage Snapshot Optimization

This new feature enables you to take a storage snapshot of your database using third-party technologies without keeping the database in BACKUP mode. When you need to recover, you can use point in time of the snapshot. You can roll forward by using the database archive logs, and use this snapshot feature to recover part or all of the database.

The best practice is to use different storage location to keep snapshot than the one where the database currently running.

In order to backup your Oracle database using storage snapshot optimization, the third-party snapshot technologies must meet the following requirements:

* The snapshot preserves the write order for each file.
* The database is crash consistent during the snapshot.
* The snapshot technology stores the time at which the snapshot is completed.

If third-party snapshot technology vendor cannot guarantee compliance with above requirements then you must keep the database in BACKUP mode to take the snapshot.

Follow these steps to keep the database in BACKUP mode and take snapshot

* SQL> ALTER DATABASE BEGIN BACKUP
* Take Snapshot using third-part technologies
* SQL> ALTER DATABASE END BACKUP

Use RECOVER…SNAPSHOT TIME command to recover the database in one step from RMAN or SQL\*Plus. You can recover the database to a point-in-time or current time after the snapshot was taken. When performing point-in-time make sure that the recovery time cannot be earlier than the snapshot time.

To recover database completely use below command

RECOVER DATABASE;

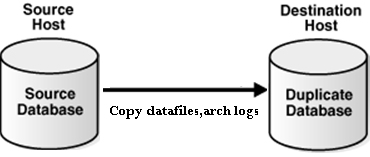
To recover database using particular snapshot use below command

RECOVER DATABASE UNTIL TIME ‘10/10/2014 10:00:00’ SNAPSHOT TIME ‘10/10/2014 09:00:00’

If there are any structural changes during the snapshot then the snapshots are unusable. Do not perform the operations such as ONLINE, OFFLINE, DROP, RENAME, ADD, READONLY and SHRINK on data files and table spaces.

## Active Database Duplication Improvements

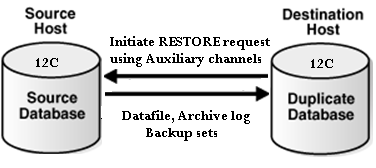
Active Database duplication was introduced in Oracle Database 11g. Using this feature you can create clone or standby database by copying the data files and archive logs using the TARGET (Source) database channels over the network to clone AUXILIARY database. As you are using the TARGET database channels you will see processing load on the TARGET instance. In this method no backups of the TARGET database are required.



In Oracle 11g, Performance typically gated by network bandwidth

In Oracle 12c, you can perform Active Database duplication using the backup sets. You can allocate sufficient AUXILIARY channels to connect TARGET database and retrieve the backup’s sets over the network, this reduced the load on the TARGET (source) database. You can use unused block compression to reduce the size of the backups transported over the network.

While performing the Active database duplication you can also encrypt backups and multisection backups.



In Oracle 12c, it helps to reduce network consumption and improve performance

As you see in below duplication example, it is using AUXILIARY channel.

oracle@gc12c ~]$ rman target sys/password@TESTDB auxiliary sys/password@DUPDB

Recovery Manager: Release 12.1.0.1.0 - Production on Wed Oct 15 10:26:58 2014

Copyright (c) 1982, 2013, Oracle and/or its affiliates. All rights reserved.

connected to target database: TESTDB (DBID=488428308732)

connected to auxiliary database: DUPDB (not mounted)

RMAN> run {

2> duplicate target database to DUPDB from active database

2> using backupset

4> db\_file\_name\_convert('TESTDB','DUPDB');

5> }

Starting Duplicate Db at 15-OCT-2014 10:27:11

using target database control file instead of recovery catalog

allocated channel: ORA\_AUX\_DISK\_1

channel ORA\_AUX\_DISK\_1: SID=20 device type=DISK

current log archived ……………….

Note that if the numbers of AUXILIARY channels allocated are less than target channels, or no auxiliary channels are allocated, then RMAN uses image copies to perform active database duplication. In order to RMAN use backup sets you need to establish connection to target database using a net service name and one of the following conditions must be satisfied.

* The number of AUXILIARY channels should be equal or greater than TARGET channels
* The DUPLICATION …FROM ACTIVE DATABASE should contain either USING BACKUPSET, USING COMPRESSED BACKUPSET or SECTION SIZE clause

## Cross-Platform Backup and Restore Improvements

Cross-platform transportable tablespace and database were introduced in Oracle 10g. In this method the user must use EXPORT or DATAPUMP to export metadata. The tablespace should be in read-only mode during the process and only image copies are used. This method is more suitable for smaller or low transaction rate applications.

In Oracle 12c, you can transport data across platforms using either full or incremental backups, using image copies or backup sets and added platform support for incremental backup. To perform cross-platform backups using backup sets, the destination database must be Oracle 12c or later. These features simplify the platform migration and minimize read-only down time on the source database using incremental based approach. By using this improvement users can reduce downtime by 8 times versus traditional migration approaches.

If the user performs the conversion at source, CONVERT TABLESPACE command should be used. If the user performs conversion at destination, CONVERT DATAFILE command should be used.

### Cross platform transportable tablespace

You can use RMAN to transport tablespace across platforms with the same endian or different endian formats. If transporting tablespace between different endian formats then the user must use CONVERT command to perform conversion. If transporting tablespace between same endian formats then the CONVERT command isn’t needed.

Transportable tablespace is useful under following situations:

* When migrating tablespace across the platforms
* When moving data from large data warehouse to data marts on small servers
* Sharing read-only tablespace across heterogeneous cluster where hosts have same endian format

User can use below query to get the platform name of the connected database

SQL> SELECT PLATFORM\_NAME, ENDIAN\_FORMAT FROM V$TRANSPORTABLE\_PLATFORM

WHERE PLATFORM\_ID = (SELECT PLATFORM\_ID FROM V$DATABASE);

Before you convert the tablespace make sure you run DBMS\_TTS.TRANSPORT\_SET\_CHECK procedure to check if tablespace is self-contained or not.

For ex: – EXECUTE DBMS\_TTS.TRANSPORT\_SET\_CHECK(‘USER\_TBS’, TRUE);

After executing these commands, you can check for violations in TRANSPORT\_SET\_VIOLATIONS view. User must resolve before proceeding with conversion.

### Cross-platform transportable database

You can use RMAN to transport entire database across platforms with same endian format only. When transporting database the user doesn’t need to perform EXPORT/IMPORT as SYSTEM tablespace is part of the database that being copied. You can use CONVERT DATABASE command to convert and automatically transport the database to its destination. You can convert the data files either on source or destination platforms.

When you run the CONVERT DATABASE on the destination platforms, it does not convert the format of the files, rather it generates the two scripts to perform conversion manually.

* CONVERT SCRIPT – This script used to convert data files copies in batch mode
* TRANSPORT SCRIPT – This script contains SQL Statements to create new databases on the destination platform

Transportable Database is useful under following situations:

* When Evaluation migration path for new platforms
* When you need database on less expensive servers that use different platform
* Distribute data from one source system to multiple targets with different platforms

Before converting the database make sure you run DBMS\_TDB.CHECK\_DB function. This function checks for incompatible endian formats, active transactions or incorrect compatibility settings between source and destination.

In Oracle 12c, Oracle 10.2.x and 11g database backups can be restored and recovered cross-platform to Oracle 12c. Note that cross-platform transportable database is not the same thing as transportable tablespace

## Recovering Tables and Table Partitions using RMAN Backups

RMAN enables you to recover tables and table partitions at a point-in-time without affecting the other objects in the database. Use RECOVER TABLE command to recover tables or table partitions from an RMAN backup.

This feature is useful in the following scenarios:

* When the object has Logical corruption or dropped
* When there is no sufficient undo to perform Flashback table
* When DDL operation modified the structure and you want to recover the data (Flashback cannot rewind the structural changes)
* If you need to recover a small number of tables to a point-in-time

Before you prepare to recover the tables and table partitions, make sure you verify the pre-requisites and determine the point-in-time. RMAN enables you to specify the point-in-time either using SCN, Time or sequence number. In order to perform table/ table Partition recovery these conditions must be met:

* Database must be in ARCHIVELOG mode and read-write mode
* At least one full backup is available along with archived logs
* Enough disk space is available on the database server for auxiliary instance
* If present, any dependent objects to include in recovery
* COMPATIBLE parameter must be set to 11.1.0 or higher to recover table partition

RMAN enables recovery of selected tables without affecting remaining database objects. During the recovery process RMAN creates an auxiliary database, which is used to recover the tables or table partitions to a specified point-in-time. User need to specify the auxiliary database location using AUXILIARY DESTINATION clause in the RECOVERY command or SET NEWNAME command. Please find the steps performed by RMAN during the recovery process:

* Determine the backup which has the tables or table partitions that needs to recover to specified point-in-time
* Create auxiliary database and recovery the tables or table partitions until specified point-in-time
* Take a data dump export with recovered tables or table partitions
* Import the dump into target database
* Rename the recovered tables or table partitions in the target database

Please find an example to recovery TBL1 table.

RECOVER TABLE TESTUSER.TBL1

UNTIL SCN 384840289

AUXILIARY DESTINATION '/tmp/TESTDB/recover'

DATAPUMP DESTINATION '/tmp/TESTDB/dumpfiles'

DUMP FILE 'testdump.dat';

If source table exists then user can specify NOTABLEIMPORT or REMAP TABLE. Also user can use UNTIL TIME or UNTIL SEQUENCE clause to specify point-in-time recovery.

Please find an example to recovery TBL1 table as TBL1\_REC to the state that it was 2 days before the current date.

RECOVER TABLE TESTUSER.TBL1

UNTIL TIME 'SYSDATE-2'

AUXILIARY DESTINATION '/tmp/TESTDB/recover'

REMAP TABLE 'TESTUSER'.'TBL1':'TBL1\_REC';

There are some limitations recovering tables and table partitions:

* We cannot recover table and table partitions belonging to SYS schema
* We cannot recover table or table partitions from SYSAUX,SYSTEM tablespace Tables
* We cannot recover tables with named NOT NULL constraint using REMAP option.
* We cannot recover Table/Table partitions ON STANDBY database
* We cannot recovery table partitions if version is prior Oracle Database 11g R1

Note that there are other methods available to recover tables to a point-in-time such as Oracle Flashback and Tablespace Point-in-Time Recovery.

## Unified auditing and RMAN

In Unified auditing you can consolidate all audit records into single audit trail. User can view the audit trail by querying UNIFIED\_AUDIT\_TRAIL data dictionary view for single-instance and GV$UNIFIED\_AUDIT\_TRAIL for Oracle RAC. In order to query UNIFIED\_AUDIT\_TRAIL view user must have AUDIT\_ADMIN OR audit viewer ROLE.

The UNIFIED\_AUDIT\_TRAIL view has fields begin with RMAN\_ and these fields automatically record RMAN related events.

Please find the Oracle recovery Manager specific columns in UNIFIED\_AUDIT\_TRAIL data dictionary view.

* RMAN\_SESSION\_RECID – It contains the RMAN session identifier
* RMAN\_SESSION\_STAMP – It contains the timestamp for the RMAN session.
* RMAN\_OPERATION – It contains the operation executed by RMAN job
* RMAN\_OBJECT\_TYPE – It contains the type of object involved in a RMAN session.
* RMAN\_DEVICE\_TYPE – it contains device type associated with RMAN session. This can be SBT or DISK.

# V. Oracle Database Concept

## Oracle SCN

SCN (System Change Number): đây là một số gia tăng duy nhất trong cơ sở dữ liệu (như đồng hồ thời gian của bạn). Số SCN được tăng lên mỗi 3 giây. Con số này là rất hữu ích trong khi phục hồi cơ sở dữ liệu.

* + SCN xác định một phiên bản đã commit của cơ sở dữ liệu tại một thời điểm. Oracle gán mọi transaction đã commit một SCN.
  + SCN là một dấu thời gian nội bộ cho một commited version của cơ sở dữ liệu . Các máy chủ cơ sở dữ liệu Oracle sử dụng SCN clock để đảm bảo tính nhất quán SCN transaction. Ví dụ, khi một người sử dụng commit một transaction, cơ sở dữ liệu ghi lại một SCN này và commit ghi trên redo log.
  + SCNs là quan trọng đối với transaction bởi vì nó có chức năng như một dấu thời gian đồng bộ commit transaction , ngay cả khi transaction không thành công. Nếu một transaction thay đổi dữ liệu sai hoặc không phù hợp, một quản trị viên có thể sử dụng SCN này để phối hợp thay đổi trên cơ sở dữ liệu. SCN cho transaction commit cũng có thể được sử dụng để xác định các transaction sau đó.
  + Nếu SCN không đồng nhất giữa các thành phần CSDL sẽ nảy sinh vấn đề. Database sẽ lấy SCN của control file và apply cho tất cả các thành phần khác
  + Control file lưu 1 thông tin rất quan trọng là SCN. Con số này được phát sinh và tăng liên tục theo thời gian. Oracle dựa vào con số này đề đồng bộ tất cả các file trong database như: datafile, control file, redo log file v.v...

## Oracle Service

An automatic workload management facility, called database services. Database services (services) are logical abstractions for managing workloads in Oracle Database. Services divide workloads into mutually disjoint groupings. Each service represents a workload with common attributes, service-level thresholds, and priorities.

A single service can represent an application, multiple applications or a subset of a single application. For example, the Oracle E-Business suite defines a service for each responsibility, such as general ledger, accounts receivable, order entry, and so on. A single service can be associated with one or more instances of an Oracle RAC database, and a single instance can support multiple services.

## AMM vs ASMM

* + **AMM in Oracle 11g:** The 11g release uses AMM and manages all of the SGA AND PGA via the memory\_targetparameter.
  + **ASMM in Oracle10g:** Oracle ASMM was with Oracle 10g and uses two parameters  sga\_max\_size for the SGA and pga\_aggregate\_targetfor the PGA.
* **Use ASMM better**

## Cluster table

* + Ý tưởng: ghép các table thường xuyên được sử dụng để truy vấn cùng nhau thành 1 cluster
  + Sử dụng cột dùng để join các bảng thành cluster key
  + Đọc thêm dữ trong Tài liệu của hãng

## Quản lý Oracle Resources

* + Ý tưởng chính : cấp quota cho các resource consumer group
  + Đọc thêm trong tài liệu

## Database Link

Sending data or messages between sites requires network configuration on both sites.

You must configure the following:

* + Network connectivity (for example, tnsnames.ora)
  + Database links

SQL> CREATE [PUBLIC] DATABASE LINK <link\_name>

CONNECT TO <user\_name>

IDENTIFIED BY <password>

USING '<service\_name>';

## Database Smart Flash Cache:

* Tính được chỉ có thể được sử dụng trên solaris và Oracle Linux
* Cung cấp cache lv 2, tăng tốc độ xử lý dữ liệu

Sử dụng khi:

* Database chạy trên OS Solaris và Oracle Linux
* The Buffer Pool Advisory trong AWR yêu cầu tăng buffer cache lên gấp đôi
* db file sequential read thuộc top wait event
* Có 1 Cpu đang nhàn rỗi

Sizing Flash Cache:

* Gấp 2-10 lần dung lượng của bufer cache
* Gấp 2-10 lần dung lượng của SGA\_TARGET ( ở chế độ ASMM)

## Oracle Restart

Tên thành phần: Oracle restart

Tool quản trị : srvctl

Khái niệm:Một thành phần của **clusterware**

Chức năng chính:

* Được sử dụng để quản lý tài nguyên trên máy cục bộ
* Tự động **start và restart** các tài nguyên khi cần thiết
* **DB instance, listener, services** ..... đăng kí với **Oracle restart, Oracle restart** đăng kí với **Clusterware (** trong môi trường **Rac)**