

# Administración Básica de Oracle 10g

Este documento electrónico puede ser descargado libre y gratuitamente desde Internet para su ejecución e impresión, sólo para fines educativos y/o personales, respetando su integridad y manteniendo los créditos de los autores en el pie de página.

**Francisco Fernández Martínez (pacof@um.es)**

**Juan Luis Serradilla Amarilla (juanlu@um.es)**

**Universidad de Murcia**

# TEMARIO

- Arquitectura de la Base de Datos
- Arranque y parada
- Fichero de control
- Redo log
- Tablespaces
- Segmentos de rollback
- Usuarios, roles, privilegios y perfiles
- Jobs
- Auditoría
- Copias de seguridad y recuperación

# OBJETIVOS

- Conocer la Arquitectura Oracle.
- Saber arrancar y parar una base de datos Oracle.
- Gestionar los ficheros Redo log.
- Gestionar el fichero de control.
- Gestionar tablespaces, incluyendo temporales y undo.
- Gestionar segmentos de rollback.
- Gestionar usuarios, roles, privilegios y perfiles.
- Gestionar jobs.
- Gestionar la auditoría del sistema gestor de base de datos.
- Realizar copias de seguridad y recuperación de la base de datos.

# TEMA 1

## ARQUITECTURA DE LA BASE DE DATOS

# TEMA 1.

## ARQUITECTURA DE LA BD

- Servidor oracle
- Instancia
- Conexión a la Base de datos
- Memoria: SGA (Automatic Shared Memory Management  $\geq 10g$ ) y PGA
- Procesos: procesos de usuario, servidores y background
- Estructura lógica: tablespaces, segmentos, extensiones, bloques
- Arquitectura OFA
- Usuarios administradores de la BD: sys y system
- Fichero de autenticación (orapw)

# SERVIDOR ORACLE

Servidor Oracle:

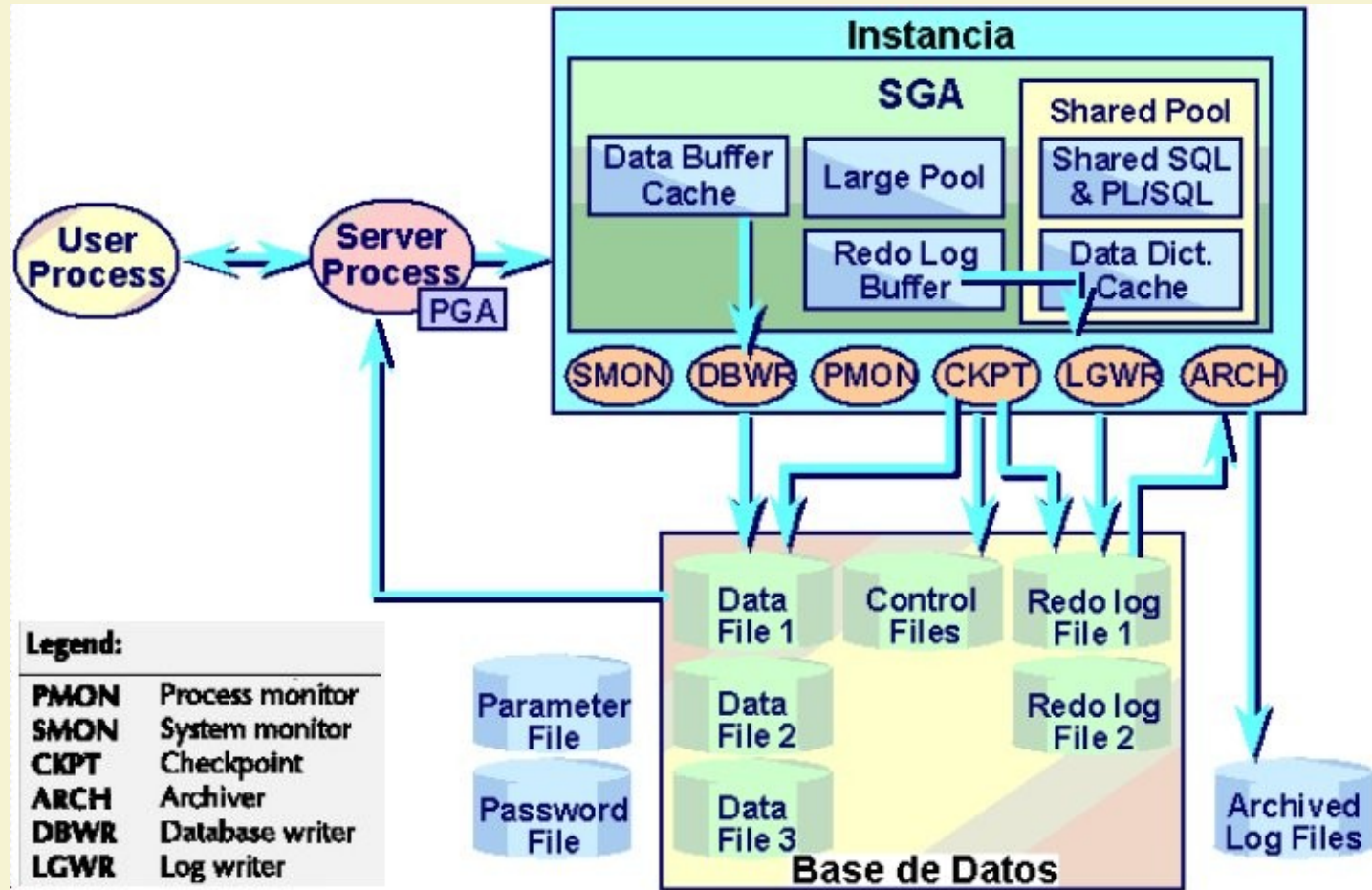
- Instancia Oracle
- Base de datos Oracle

Instancia Oracle:

- System Global Area (SGA)
- Procesos Background

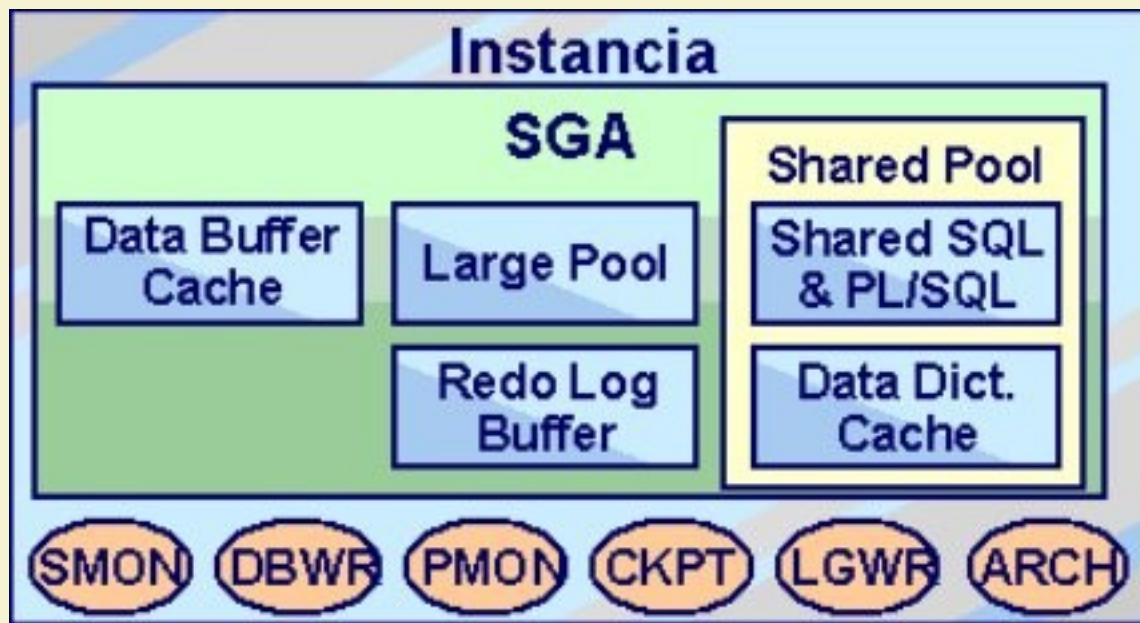
Base de datos Oracle:

- Ficheros de Datos
- Fichero(s) de Control
- Ficheros Redo Log.



# INSTANCIA

- La integran los procesos “background” y la SGA
- Abre una y sólo una BDO, y permite acceder a ella.  
Nota: con Oracle Real Application Cluster (RAC), más de una instancia usarán la misma BD.
- En la máquina donde reside el servidor Oracle, la variable ORACLE\_SID identifica a la instancia con la que estamos trabajando.

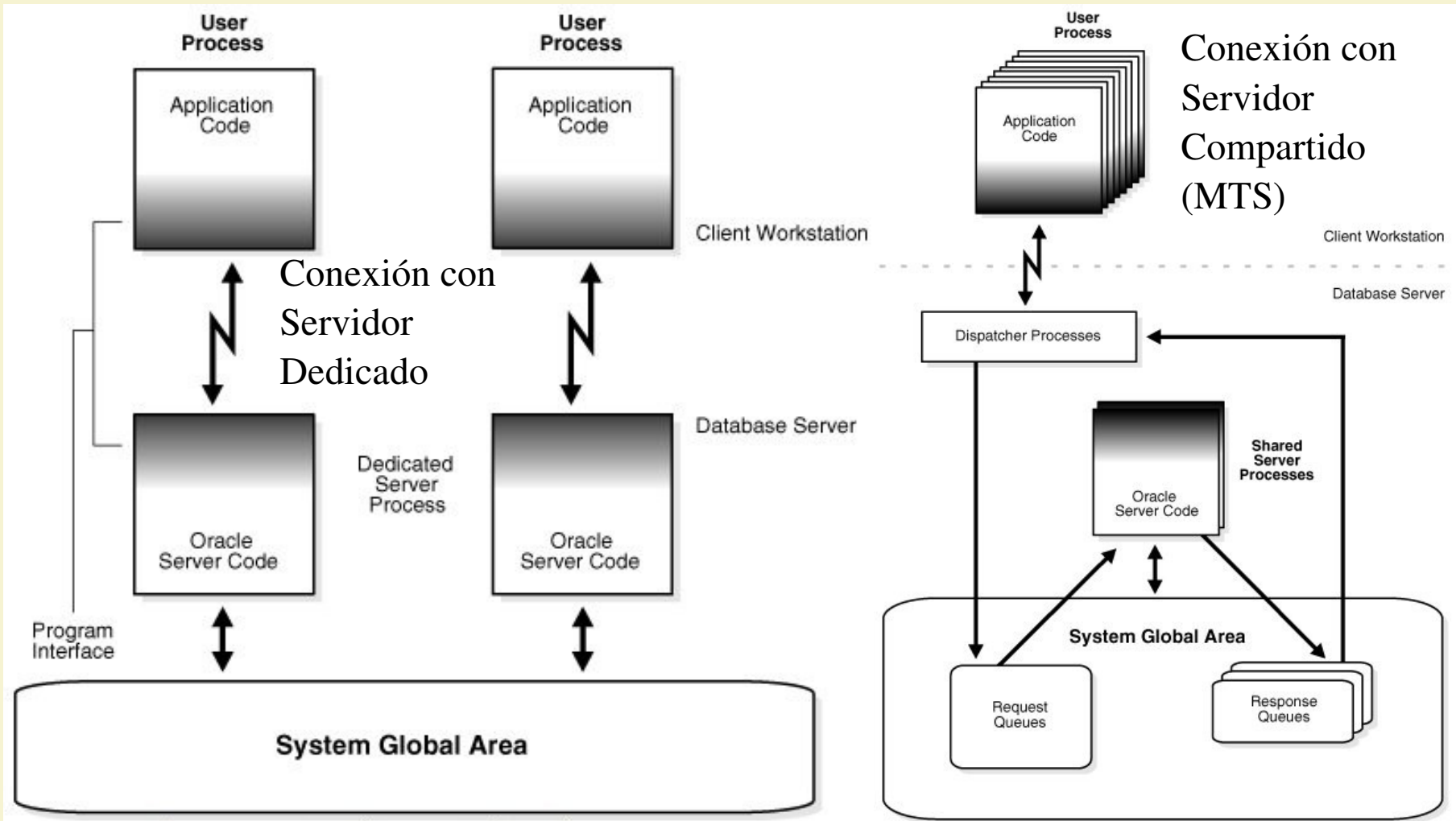


# CONEXIÓN A LA BD

- Para poder conectarnos a una BDO, necesitamos una serie de variables en el entorno del usuario del S.O. desde el que realizaremos la conexión. En el caso de unix/linux:
  - ORACLE\_HOME. Localización del sw Oracle a utilizar.
  - ORACLE\_SID. BD, asociada al ORACLE\_HOME, a la que vamos a conectarnos. Tiene sentido sólo en un SBD.
  - PATH=\$PATH:\$ORACLE\_HOME/bin. Programas Oracle
  - LD\_LIBRARY\_PATH=\$ORACLE\_HOME/lib. Localización de las librerías compartidas (Linux/Unix).
  - NLS\_LANG=spanish\_spain. Idioma del cliente (opcional).
- Proceso de usuario: la ejecución de la aplicación que permite al usuario iniciar la conexión; por ejemplo, sql\*plus.
- Proceso servidor: se crea en el SBD cuando el usuario se conecta a la BD, y es el que realmente interactúa con la BD.
- Una conexión de un proceso de usuario al SBD es una sesión en la BD (puede haber varias del mismo usuario). Se inicia cuando el usuario se valida contra la BD y termina cuando el usuario se desconecta.
- Podemos desconectar una sesión con “ALTER SYSTEM DISCONNECT SESSION 'sid, serial#' [POST\_TRANSACTION] [IMMEDIATE];”. Destruye el servidor dedicado (o el circuito virtual si MTS).

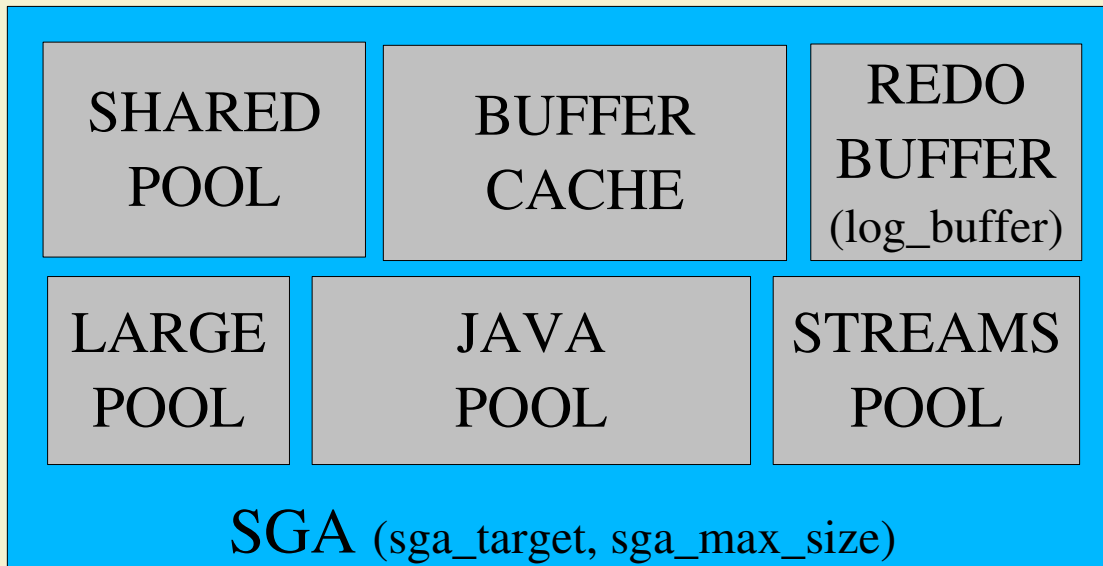


# CONEXIÓN A LA BD: procesos



# SGA (System Global Area)

## y Automatic Shared Memory Management



- Es una zona de memoria compartida, que se reserva en el momento de arrancar la instancia. Su tamaño es dinámico ( $\geq 9i$ ) y limitado por el parámetro de inicialización `SGA_MAX_SIZE`.
- `SGA_TARGET` ( $\geq 10g$ ) fija el tamaño de la SGA y activa el reparto automático de su espacio entre: sga fija, shared pool, large pool, java pool, buffer caché y streams pool. El resto se ajustan manualmente.
- log buffer, buffer cachés keep y recycle, y buffer cachés con tamaño de bloque especial; aunque consumen espacio de `SGA_TARGET`, se fijan manualmente (`LOG_BUFFER`, `DB_KEEP_CACHE_SIZE`, `DB_RECYCLE_CACHE_SIZE`, `DB_nK_CACHE_SIZE`). El resto (punto anterior) los dejaremos a cero (tb podemos darles valores, q serán tomados como mínimos).
- `V$SGAINFO`: tamaño componentes SGA (tb gránulo y libre). `V$SGA_DYNAMIC_COMPONENTS`.
- `V$SGA_TARGET_ADVICE`: recomendaciones sobre `SGA_TARGET`.

# SGA (System Global Area) y ASMM

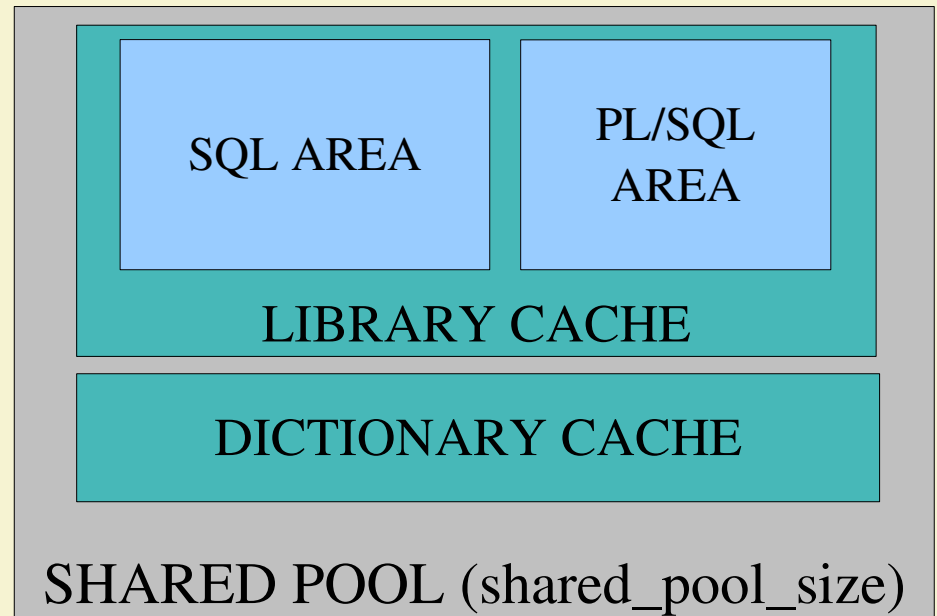
- La SGA está formada por gránulos (espacio contiguo de memoria virtual), que serán de 4M para SGAs  $\leq$  1Gb (en 9i  $\leq$  128M), y de 16M en caso contrario (ver V\$SGAINFO). Los componentes de la SGA (buffer cache, sharedpool, largepool, javapool, etc) variarán su tamaño usando en base a gránulos. Al arrancar, se asignan al menos tres gránulos (uno para la SGA fija que incluye los redo buffers, otro para la buffer cache y uno más de sharedpool); y cada componente seguirá reservando tantos gránulos como necesite.
- La SGA está compuesta, fundamentalmente, por tres estructuras de memoria: shared pool, database buffer cache y redo log buffer. Además, existen tres estructuras de memoria que, opcionalmente, pueden estar presentes en la SGA: large pool, streams pool y java pool. Los parámetros del fichero de inicialización que más afectan al tamaño de la SGA son: DB\_CACHE\_SIZE ( $\geq$ 9i, antiguo db\_block\_buffer en  $\leq$ 8i), LOG\_BUFFER, SHARED\_POOL\_SIZE.
- Ejemplo de parámetros de inicialización para usar gestión automática de SGA (SGA\_TARGET):

```
sga_max_size = 80M
sga_target = 70M
db_cache_size = 0
shared_pool_size = 0
large_pool_size = 0
java_pool_size = 0
streams_pool_size = 0
log_buffer = 1048576
```

# SGA: Shared Pool

- Está formada por dos estructuras de memoria gestionadas por algoritmos LRU:
  - library cache
  - dictionary cache
- Su tamaño viene determinado por el parámetro `shared_pool_size`, del `init.ora` (sin que la SGA supere `sga_max_size`). Desde 10g se recomienda usar `sga_target`, dejando `shared_pool_size` a cero, o indicando un valor mínimo). Se puede cambiar dinámicamente con:  

```
ALTER SYSTEM SET  
  SHARED_POOL_SIZE = 64M;
```
- Se puede “vaciar” con `ALTER SYSTEM FLUSH SHARED_POOL;` (menos lo q está en uso por sesiones o q esté “fijado” con `dbms_shared_pool.keep`).



# SGA: Shared Pool

- En la library cache se almacena información sobre las sentencias SQL y PL/SQL, usadas recientemente. Está formada por dos estructuras:
  - Shared SQL área; se almacenan los planes de ejecución y los árboles sintácticos (parse tree) de las sentencias SQL.
  - Shared PL/SQL área; contiene las unidades de programa compiladas y analizadas sintácticamente (parsed): procedures, functions, packages y triggers.
- En la dictionary cache se guardan las definiciones de datos usadas más recientemente: database files, tablas, índices, columnas, usuarios, privilegios, etc. Esta información se genera y utiliza en la fase de análisis sintáctico (parse); y se obtiene de las tablas del diccionario de datos. Es como una caché de datos para el DD.

# SGA: Database Buffer Cache

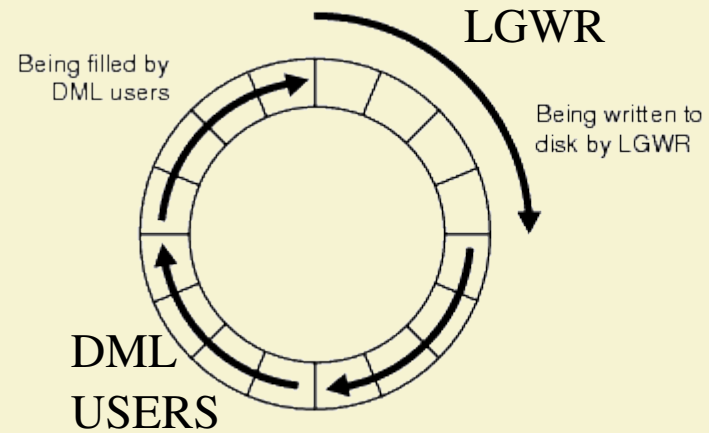
- Almacena copias de los bloques de datos, extraídos de los ficheros de datos (data files); y está gestionado por un algoritmo LRU.
- Cuando se procesa una query, el proceso servidor busca los bloques de datos en la Database Buffer Cache; si no los encuentra, los lee de los ficheros de datos y guarda una copia en la Database Buffer Cache.
- Su tamaño depende principalmente del parámetro DB\_CACHE\_SIZE (desde 10g mejor usar sga\_target, dejando db\_cache\_size a cero o un valor mínimo). Puede modificarse dinámicamente (sin sobrepasar SGA\_MAX\_SIZE):

```
ALTER SYSTEM SET DB_CACHE_SIZE = 96M;
```

- Se pueden definir varias cachés de datos independientes:
    - DB\_CACHE\_SIZE. Dimensiona la caché por defecto, que siempre existe y cuyo tamaño no puede valer cero.
    - DB\_KEEP\_CACHE\_SIZE. Dimensiona la caché donde se guardarán los bloques de tablas que se usan frecuentemente.
    - DB\_RECYCLE\_CACHE\_SIZE. Dimensiona la caché que almacena los bloques de las tablas que se usan poco.
- El uso de una u otra caché lo indicaremos con el parámetro BUFFER\_POOL, de la clausula STORAGE de la tabla: keep, recycle o default:
- ```
ALTER TABLE mitabla STORAGE (BUFFER POOL KEEP);
```
- Se pueden definir cachés adicionales para tablas que no usan el tamaño de bloque por defecto de la BD; con los parámetros DB\_nK\_CACHE\_SIZE, por ejemplo, DB\_16K\_CACHE\_SIZE. Después se crea un tablespace que use el nuevo tamaño de bloque (create tablespace ... BLOCKSIZE 16384;). Util al importar un tablespace de otra BD con otro db\_block\_size.
  - Desde 10g, se puede vaciar con “ALTER SYSTEM FLUSH BUFFER\_CACHE; “. Vacía completamente la caché de datos de la SGA. ¡¡¡OJO!!!, no usar en producción. Util si se quiere medir el rendimiento de sentencias sql como si se ejecutasen por primera vez.

# SGA: Redo Log Buffer Cache

- Es un **buffer circular** que **registra todos los cambios** hechos en los bloques de la caché de datos (incluidos datos en sí, índices y rollback), en lo que llamaremos “redo entries”. Su **propósito principal es la recuperación de la instancia** (no confundir con el “rollback”).
- El tamaño viene determinado por el parámetro log\_buffer (en bytes).
- Las “redo entries” contienen la información necesaria (índices y rollback incluidos) para **repetir los cambios** hechos mediante insert, update, delete, create, alter o drop.
- Los procesos servidores copian las entradas de redo en la Redo Log Buffer Cache (después de modificar los bloques en la caché de datos) ; y el proceso LGWR es el encargado de volcar dichos buffers al fichero redo log activo (en disco).



# SGA: Large Pool

- Es un área de memoria de la SGA, a configurar sólo si se usa:
  - un entorno de servidores compartidos (shared server o MTS)
  - recovery manager (RMAN)
  - parallel query
- Con MTS, almacena información sobre las sesiones conectadas a través de servidores compartidos: UGA, I/O y operaciones de backup y recuperación.
- No hace uso de algoritmo LRU para su gestión.
- Su tamaño depende del parámetro `large_pool_size` (en bytes), del fichero de inicialización, que se puede modificar dinámicamente (sin que el tamaño total de la SGA sobrepase el parámetro `SGA_MAX_SIZE`):

```
ALTER SYSTEM SET LARGE_POOL_SIZE = 64M;
```
- Desde 10g mejor usar `sga_target`, dejando `large_pool_size` a cero (o con un valor mínimo).



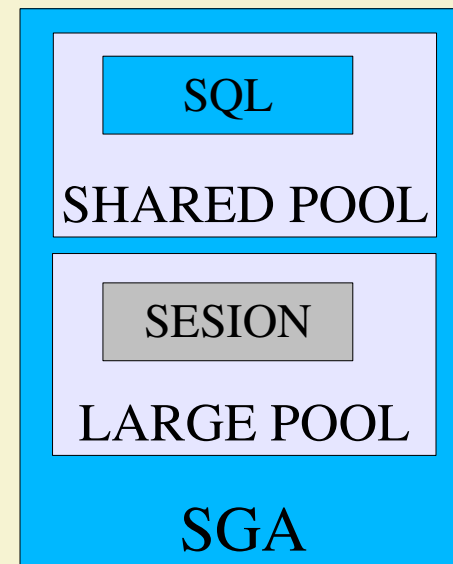
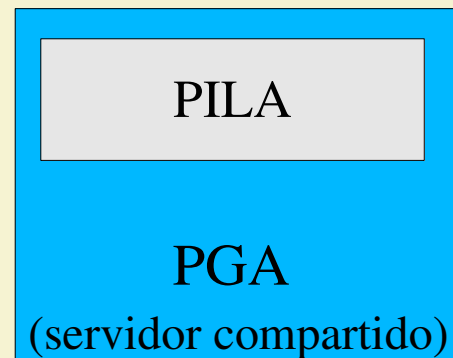
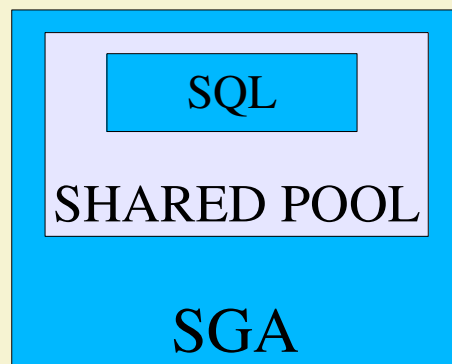
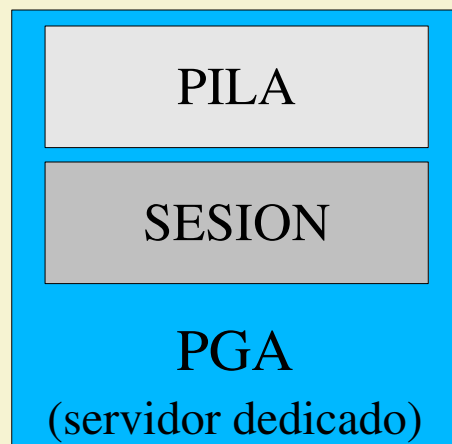
# SGA: Java Pool

- Sólo es necesaria si se instala y se usa Java en la BD. Se utiliza para compilar (parsing) el código Java (de forma similar a la Shared Pool para el código PL/SQL).
- Su tamaño depende del parámetro `java_pool_size` (redondeado por encima a un múltiplo del valor del gránulo), del fichero de inicialización. En Oracle9i, su tamaño por defecto es de 24M (si el tamaño del gránulo es de 4M, y de 32M si es de 16M).
- Desde 10g mejor usar `sga_target`, dejando `java_pool_size` a cero (o con un valor mínimo), de modo q sea Oracle el que se encargue de ajustar su tamaño automáticamente.

# Program Global Area (PGA)

- Es una zona de memoria, fuera de la SGA, reservada para cada proceso de usuario que se conecta a la BD.
- Se crea cada vez que se arranca un nuevo proceso servidor (o un proceso background); y se libera cuando el proceso termina.
- En un entorno de servidores dedicados (dedicated server) contiene: sort area, información de sesión (privilegios de usuario y estadísticas de sesión), estado de los cursores (etapa del procesamiento de cada sentencia SQL que está usando actualmente la sesión), pila (stack space).
- Con servidores compartidos (MTS), parte de estas estructuras se guardan en la SGA. Si se activa la Large Pool se almacenan en ella, si no se quedan en la Shared Pool.
- PGA\_AGGREGATE\_TARGET ( $\geq 9i$ ). Valor mínimo 10M, y default=20%SGA.. Vista V\$PGASTAT. Tamaño = en OLTP  $RAM * 0.80 * 0.20$  (en DSS  $RAM * 0.80 * 0.50$ ). Activar: si es distinto de cero (junto a WORKAREA\_SIZE\_POLICY=AUTO). Habilita el uso de una zona de memoria compartida para las PGA, evitando la necesidad de asignar parámetros como SORT\_AREA\_SIZE o HASH\_AREA\_SIZE.

# Program Global Area (PGA)



# Estructura de procesos

- Procesos de usuario: se arranca uno cuando un usuario solicita una conexión a la BD. Establece la conexión con la BD pero no interactúa directamente con ella.
- Procesos servidores: creado cuando se establece la conexión a la BD. Es el proceso que interactúa con la BD, para cada sesión. Un proceso servidor puede ser dedicado o compartido. Uno dedicado sólo gestiona las peticiones de la sesión que lo inicia; sin embargo, uno compartido gestiona las peticiones de varios procesos de usuario.
- Procesos background: disponibles cuando se arranca una instancia Oracle. Son los siguientes: DBWR, PMON, SMON, LGWR y CKPT; y nuevos en 10g: PSP0 (Process Spawner), MMAN (Memory Manager), MMON (Memory Monitor), MMNL (Memory Monitor Light). Opcionalmente podemos tener: ARCH, RECO, Dispatchers (Dnnn), Shared Servers (Snnn), etc. El parámetro BACKGROUND\_DUMP\_DEST, del fichero de inicialización, define el directorio donde se guardan los ficheros de traza de los procesos background.

# Procesos background (DBWR)

- DBWR. Escribe los bloques de datos (y rollback) de la SGA (data buffer cache) en los ficheros de datos. Esto lo hace de forma asíncrona, cuando:
    - Sucede un checkpoint.
    - El número de buffers modificados alcanza un umbral.
    - No quedan buffers libres.
    - Ocurre un timeout.
    - Ponemos un tablespace offline.
    - Dejamos un tablespace en modo readonly.
    - Borrarnos o “truncamos” una tabla.
    - ALTER TABLESPACE nombretsp BEGIN BACKUP.
- Nota. Un checkpoint sucede cuando:
- El fichero redo log se llena al 90%.
  - Se alcanza log\_checkpoint\_interval (bloques del SO).
  - Se llega a log\_checkpoint\_timeout (en segundos).
- Nombre del proceso: DBW0 a DBW9 y DBWa DBWj (máximo 20).
  - DB\_WRITER\_PROCESSES. Nº de procesos arrancados.

# Procesos background (LGWR, SMON)

- LGWR. Realiza escrituras secuenciales del contenido de la redo log buffer cache en los ficheros redo log. ¿Cuándo?
  - Se hace commit.
  - La redo log buffer cache se llena 1/3.
  - Hay 1Mb de cambios en la redo log buffer cache.
  - Como mucho, cada 3 segundos.
  - Siempre antes que escriba el DBWR.
- SMON. Recupera la instancia, si es necesario, cuando ésta arranca: aplica los cambios registrados en los redo log (roll forward), abre la base de datos dejándola accesible a los usuarios, y hace rollback de las transacciones que no terminaron. También se activa periódicamente, agrupando extensiones libres contiguas en extensiones de mayor tamaño (sólo para tablespaces con “default storage” cuyo pctincrease > 0). Además libera el espacio ocupado por segmentos temporales durante el procesamiento de sentencias SQL.

## Procesos background (PMON, CKPT, ARCH)

- PMON. Se activa periódicamente, recuperando los recursos después de que un proceso falle: hace rollback de las transacciones que el usuario tenía en curso, libera bloqueos a nivel de tabla/fila y otros recursos reservados por el usuario, y vuelve a arrancar dispatchers “muertos” (dead dispatchers).
- CKPT. Avisa al DBWR cuando sucede un checkpoint y actualiza las cabeceras de los ficheros de datos y de control (el DBWR volcará los buffers actualizados a los ficheros de datos). Si los checkpoints suceden muy frecuentemente puede haber contención en disco. Si tardan mucho se alargará el proceso de recovery. Como mucho sucederá un checkpoint al llenarse el redolog.
- ARCH. Proceso opcional. Archiva automáticamente los redo log online si se activa el modo ARCHIVELOG; asegurando que se registran todos los cambios hechos en la base de datos. Archiva el redo log que se ha llenado, cuando sucede un “log switch”.
- MMAN (>=10g). Memory Manager. Gestiona Automatic Shared Memory Managment.
- MMON (>=10g). Memory monitor. Genera snapshots del AWR (abre procesos esclavos M000).
- MMNL (>=10g). Memory Monitor Light. Captura frecuentemente “session history” (V\$ACTIVE\_SESSION\_HISTORY) y calcula métricas.
- PSP0 (>=10G). Process spawner. Crea y gestiona otros procesos Oracle.

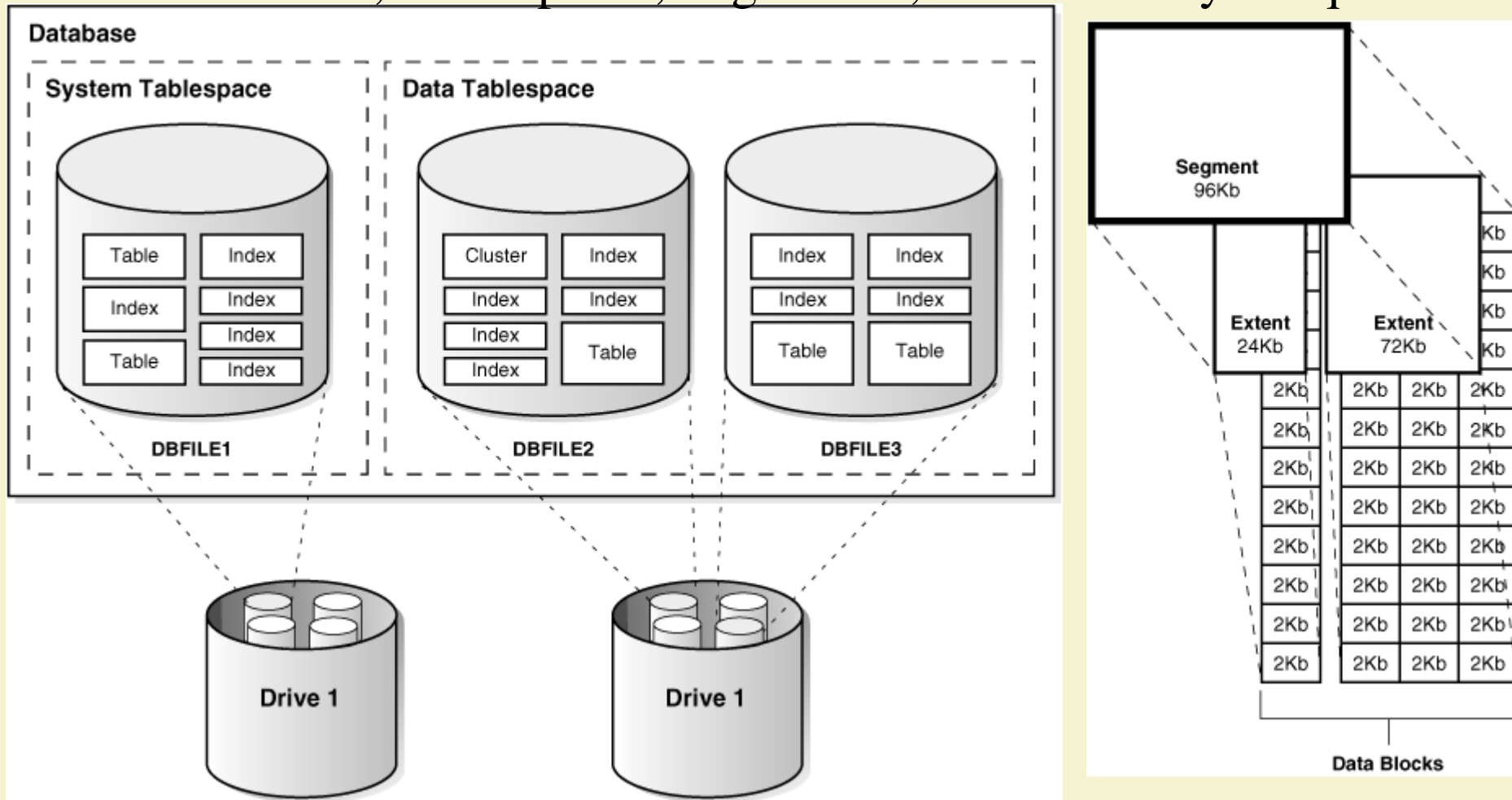
# ESTRUCTURA LÓGICA

- La estructura lógica de la base de datos determina el uso que se hace del espacio físico que la sustenta. Existe una jerarquía top-down en esta estructura, consistente en tablespaces, segmentos, extensiones y bloques.
- Una BDO la forman un grupo de tablespaces. Un tablespace puede contener uno o más segmentos. Un segmento lo integran una o más extensiones. Una extensión tendrá al menos un bloque. El bloque es la unidad mínima de almacenamiento.
- El tamaño del bloque será múltiplo del que tenga el SO, y lo determina la variable `db_block_size` (2K, 4K, 8K, 16K y 32K).
- Cuando un segmento (tabla, índice, rollback o temporal) crece, el espacio que se añade es de una extensión.
- ASM ( $\geq 10g$ ). Gestor de volúmenes para bases de datos Oracle. Gestiona directamente los discos. Además distribuye automáticamente los datos entre los discos, manteniendo el reparto uniforme cuando se añaden o quitan discos (incluso en caliente). También se encarga de borrar los ficheros que ya no forman parte de la BD.



# ESTRUCTURA LÓGICA

## Base de Datos, Tablespaces, Segmentos, Extensiones y Bloques



# ARQUITECTURA OFA

- OFA (Oracle Flexible Architecture) propone una estructura de directorios que permite ubicar fácilmente cualquier fichero del servidor de base de datos; además de agrupar dichos ficheros por componentes.
- Además, facilita el reparto de los ficheros entre diferentes discos, optimizando la E/S. Oracle recomienda separar el software de los datos; y estos últimos repartirlos entre varios discos (por ejemplo, separando datos e índices, incluso también temp y rollback):

```
/u01/app/oracle (ORACLE_BASE)
  $ORACLE_BASE/product/10.2.0.1 (ORACLE_HOME)
    $ORACLE_HOME/bin (Ejecutables)
    $ORACLE_HOME/dbs (initSID.ora, orapwSID)
  $ORACLE_BASE/admin/SID (ADMIN)
  $ORACLE_BASE/admin/SID/pfile (PFILE)
    initSID.ora (crear enlace en $ORACLE_HOME/dbs)
  $ORACLE_BASE/admin/SID/bdump (BDUMP)
    alertSID.ora y ficheros de traza de procesos background
  $ORACLE_BASE/admin/SID/udump (UDUMP)
    alertSID.ora y ficheros de traza de procesos background
/u02/oradata/SID (Ficheros de la BD: *.dbf, *.ctl, *.log)
/u03/oradata/SID (Ficheros de la BD: *.dbf, *.ctl, *.log)
```

# USUARIOS ADMINISTRADORES DE LA BD

- Cuando creamos una BDO se crean automáticamente los usuarios SYS y SYSTEM, ambos con el rol DBA.
- El SYS, cuya clave inicial es change\_on\_install, es el propietario del DD y habitualmente se usa para arrancar y parar la base de datos, así como para modificar los componentes de la misma (como instalar nuevas opciones).

Para conectar como SYS:

```
CONNECT SYS AS SYSDBA
```

```
CONNECT / AS SYSDBA
```

Nota: hay que pertenecer al grupo dba (Unix/Linux) o crear un "fichero de autenticación" en el SBD.

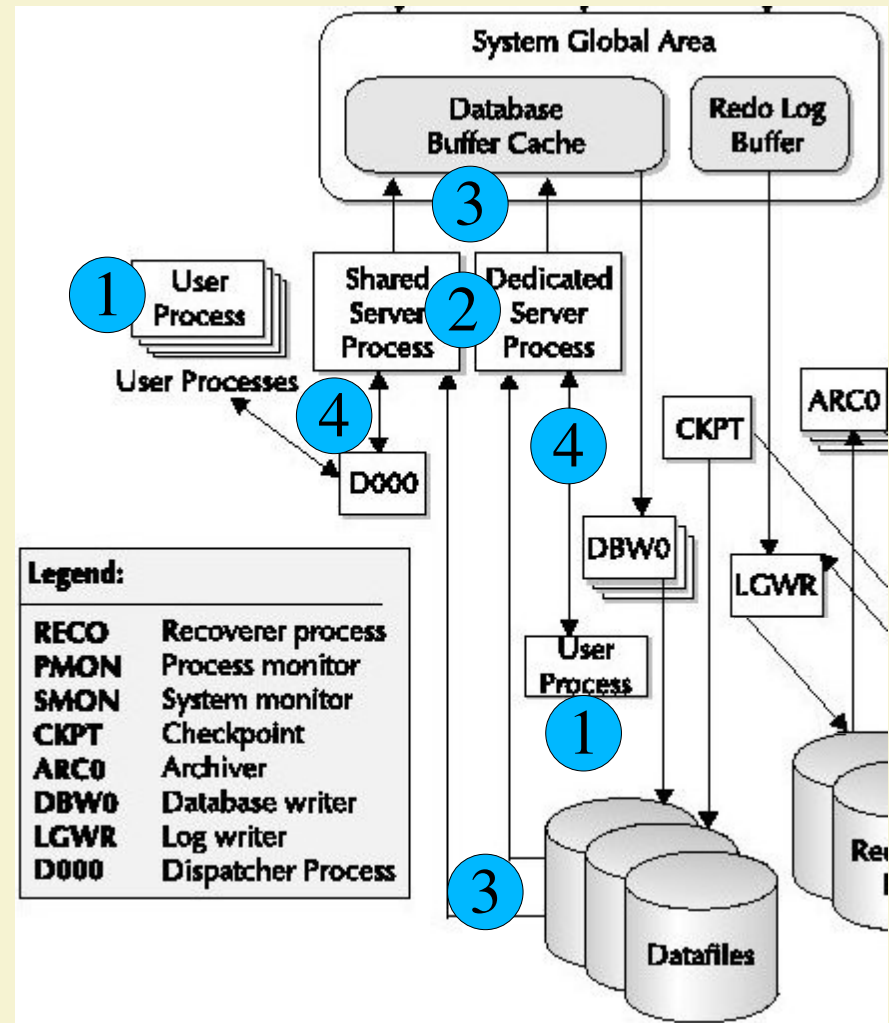
- El SYSTEM, con clave inicial manager, es el DBA por excelencia. Se usara para las tareas administrativas habituales: alta de usuarios, creación de tablespaces, etc.

# FICHERO DE AUTENTICACIÓN

- Un “fichero de autenticación” nos permite conectar a la BD como SYS AS SYSDBA, sin pertenecer al grupo dba o desde un puesto remoto al SBD, realizando dicha autenticación contra el mencionado fichero. Lo usaremos cuando no dispongamos de una conexión desde el propio SBD como grupo dba.
- Para usar un fichero de autenticación:
  - Crearemos el fichero con la utilidad orapwd:  
`orapwd file=nombre_fichero password=clave  
entries=máximo_de_usuarios`  
Nota: el fichero se llamará orapw\$ORACLE\_SID y estará en \$ORACLE\_HOME/dbs.
  - Activaremos el parámetro REMOTE\_LOGIN\_PASSWORDFILE del init:
    - EXCLUSIVE. Permite dar el privilegio SYSDBA a otros usuarios (además del SYS). Sólo una instancia usa el fichero.
    - SHARED. El único usuario reconocido por el fichero es el SYS. El fichero puede ser compartido por varias instancias.
  - Incluiremos el usuario en el fichero de claves (para el SYS no hay que hacerlo):  
`GRANT SYSDBA TO usuario;`  
Nota: en V\$PWFILERS están los usuarios con SYSDBA y/o SYSOPER.
  - Conectaremos a la BD (el usuario Oracle que se conecta siempre es el SYS):  
`CONNECT usuario/clave AS SYSDBA`

# PROCESO DE CONSULTA

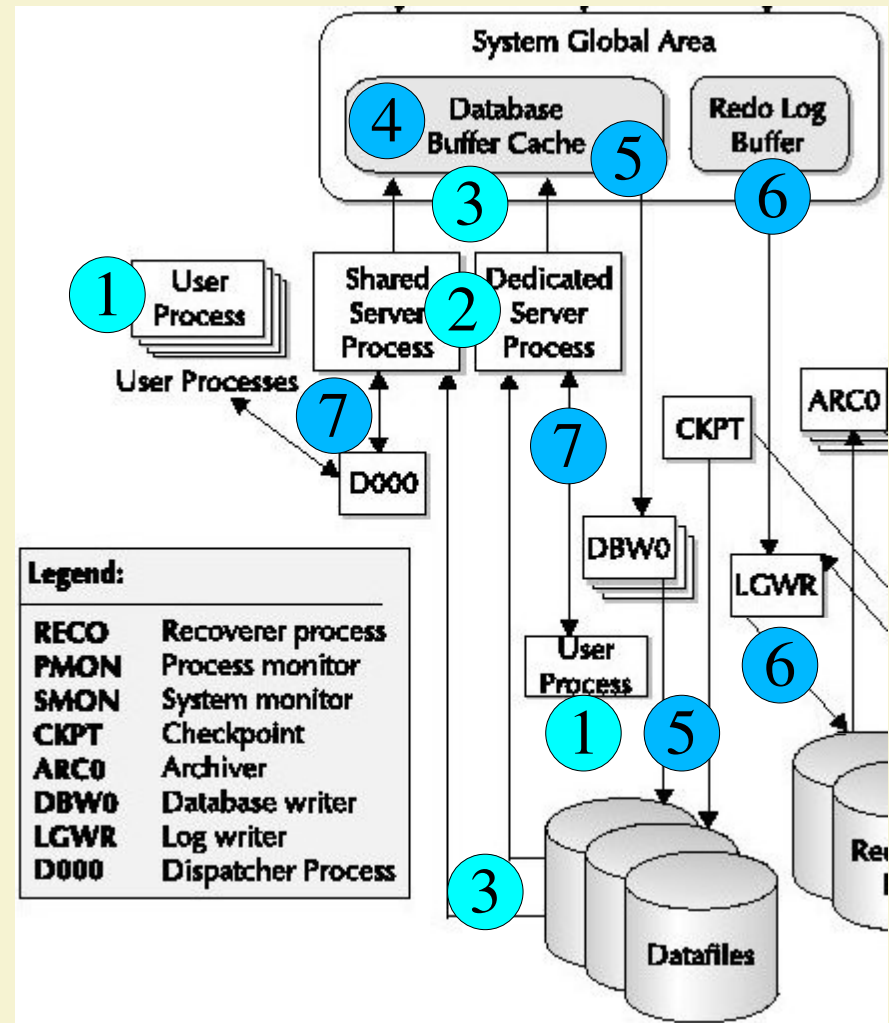
1. El proceso de usuario envía la sentencia SQL al proceso servidor.
2. El proceso servidor busca la sentencia SQL en la Shared Pool. Si no la encuentra, la compila y la guarda en la Shared Pool.
3. El proceso servidor accede a los datos en la Database Buffer Cache. Si no los encuentra, accede directamente a los ficheros de datos, llevando los datos a la Database Buffer Cache.
4. El proceso servidor devuelve los datos al proceso de usuario q inició la conexión.



# PROCESO DE ACTUALIZACIÓN

En primer lugar se repiten las fases 1 (envío de la sentencia), 2 (compilación) y 3 (acceso a los datos) vistas en el proceso de consulta.

4. Se guarda una copia del dato (antes del cambio) en un segmento de Rollback (por si se deshace la transacción).
5. Se modifican los bloques de datos en la Database Buffer Caché. El DBWR (de forma asíncrona) los llevara a los ficheros de datos cuando suceda un checkpoint.
6. Se guardan en la caché de Redo las "redo entries" (vector de cambios de cada bloque modificado) necesarias para registrar el cambio q se va a hacer (el LGWR vuelca el buffer al fichero redo log activo, cuando se hace commit o cada 3 segundos).
7. El proceso servidor devuelve el número de filas actualizadas al proceso de usuario.



# VISTAS DEL DD

- V\$DATABASE (Base de datos).
- V\$INSTANCE (Instancia).
- V\$SGA (SGA).
- V\$SGAINFO (Gestión dinámica de la SGA).
- V\$SGASTAT (SGA detallada).
- V\$BUFFER\_POOL (Buffers en la caché de datos)
- V\$SQLAREA (Sentencias SQL).
- V\$PROCESS (Procesos).
- V\$BGPROCESS (Procesos background).
- V\$DATAFILE (Ficheros de datos de la BD).
- V\$CONTROLFILE (Ficheros de control de la BD).
- V\$LOGFILE (Ficheros redo log de la BD).
- DBA\_TABLESPACES (Tablespaces de la BD).
- DBA\_SEGMENTS (Segmentos que hay en los tablespaces).
- DBA\_EXTENTS (Extensiones que componen los segmentos).
- DBA\_USERS (Usuarios de la BD).

# PRACTICAS TEMA 1

- 1.1. Comprobar la asignación de variables de entorno necesarias para conectarnos a la BD:
  - echo \$ORACLE\_HOME
  - echo \$ORACLE\_SID
  - echo \$LD\_LIBRARY\_PATH
  - echo \$PATH
- 1.2. Identificar los procesos que componen instancia:
  - ps -ef | grep \$ORACLE\_SID
  - select username, program from v\$process where background is not null;
  - select name, description from v\$bgprocess where PADDR != '00';
- 1.3. Ver el tamaño de la SGA de la BD:
  - select \* from v\$sgainfo;
  - select \* from v\$sgastat;
  - select \* from v\$sgastat where name in ('library cache', 'row cache', 'sql area', 'buffer\_cache', 'log\_buffer');
- 1.4. Comprobar valores de parámetros del init relacionados con el tamaño de la SGA:
  - show parameter sga\_target
  - show parameter sga\_max\_size
  - show parameter shared\_pool\_size
  - show parameter db\_cache\_size
  - show parameter db\_block\_size
  - show parameter log\_buffer
  - show parameter large\_pool\_size
  - show parameter java\_pool\_size



# PRACTICAS TEMA 1.

- 1.5. Comprobar ficheros que componen la BD y ubicarlos en la estructura OFA:
  - `ls -l /u0?/oradata/$ORACLE_SID`
  - `select name from v$datafile;`
  - `select name from v$tempfile;`
  - `select member from v$logfile;`
  - `select name from v$controlfile;`
- 1.6. Identificar la estructura lógica de la BD: tablespaces, segmentos, extensiones.
  - `Select tablespace_name from dba_tablespaces;`
  - `select tablespace_name, file_name from dba_data_files order by tablespace_name, file_name;`
  - `select tablespace_name, segment_type, count(*) segmentos from dba_segments group by tablespace_name,segment_type;`
  - `select tablespace_name,segment_type, count(*) extensiones from dba_extents group by tablespace_name,segment_type;`

# PRACTICAS TEMA 1.

- 1.7. Consultar información sobre la base de datos (v\$database) y la instancia (v\$instance).
  - select name, created, log\_mode, checkpoint\_change#, open\_mode, platform\_name, current\_scn from v\$database;
  - select instance\_name, host\_name, version, startup\_time, status, archiver, logins, database\_status from v\$instance;
- 1.8. Localizar el proceso “servidor” asociado a mi sesión (v\$process y v\$session). ¿Es un servidor dedicado o compartido?  

```
select a.server, a.username dbuser, a.program user_program, b.spid server_process, b.program
server_program
from v$session a, v$process b
where a.username=USER and a.PADDR=b.ADDR;
```

# PRACTICAS TEMA 1.

- 1.9. ¿Cuanto ocupa la Dictionary cache y la Library cache en tu BD? (v\$sgastat)  
`select SQL_TEXT, PERSISTENT_MEM, EXECUTIONS, LOADS, DISK_READS, CPU_TIME, ELAPSED_TIME from v$sqlarea order by DISK_READS desc;`
- 1.10. Ver la actividad de la Library Cache (v\$librarycache).  
`select namespace, gethitratio, gethitratio from v$librarycache;`
- 1.11. Ver las sentencias SQL que guarda la Shared-Pool (v\$sqlarea).  
`select sql_text from v$sqlarea;`
- 1.12. Crear el fichero de autenticación y activarlo (orapwd).  
`orapwd file=$ORACLE_HOME/dbs/orapwCURSOxy password=miclave entries=5  
remote_login_passwordfile=EXCLUSIVE"`

# TEMA 2

## ARRANQUE Y PARADA DE LA BASE DE DATOS

# TEMA 2.

## ARRANQUE Y PARADA DE LA BD

- Ficheros de inicialización: init.ora y spfile.ora
- Creación de la BD
- OMF (Oracle Managed Files)
- Arranque de la base de datos
- Comando startup
- Comando alter database
- Parar la base de datos
- Fichero alertSID.Log
- Trazas de los procesos background
- Trazas de los procesos de usuario
- Diccionario de datos
- Automatic Storage Managment (ASM)

# FICHEROS DE PARÁMETROS DE INICIALIZACIÓN – I

- Para arrancar la instancia, el servidor Oracle tiene que leer el fichero de parámetros de inicialización (spfile o init), cuya ubicación predeterminada es \$ORACLE\_HOME/dbs.
- El fichero de parámetros de inicialización puede ser de dos tipos:
  - Init: se trata de un fichero de texto, editable, cuyo nombre sigue el patrón init\$ORACLE\_SID.ora.
  - Spfile: es un fichero binario, no editable pero visualizable, cuyo nombre sigue el patrón spfile\$ORACLE\_SID.ora.

- Se crea, a partir de un init, con:

```
CREATE SPFILE [= 'nombre'] FROM PFILE [= 'nombre'];
```

Nota1. Si se omiten los nombres, toma los valores por defecto.

Nota2. La BD no podrá abrir el nuevo spfile hasta el siguiente arranque.

Nota3. Se puede crear un init a partir de un spfile, invirtiendo la sintaxis.

- Los parámetros del spfile se modifican con:

```
ALTER SYSTEM SET parametro = valor [SCOPE = MEMORY  
| SPFILE | BOTH]
```

Nota. Si sólo queremos modificar el parámetro en el spfile, indicaremos SPFILE. Para hacer el cambio solo en memoria, especificar MEMORY.

# FICHEROS DE PARÁMETROS DE INICIALIZACIÓN – II

- Hay dos tipos de parámetros:
  - Explícitos: los que se indican en el fichero de parámetros.
  - Implícitos: el resto, que tomarán un valor por defecto.
- La forma de indicar valor a los parámetros es parametro=valor.
- El símbolo # indica el comienzo de un comentario, pudiendo estar al principio o en medio de la línea.
- En el init.ora, el parámetro ifile permite incluir otros ficheros con parámetros.
- Una lista de valores se indicará entre paréntesis, separando los valores por comas.
- Para indicar un valor de tipo cadena de caracteres hay que encerrarlo entre comillas simples.
- Si usamos OFA, la ubicación típica para el init.ora es \$ORACLE\_BASE/admin/\$ORACLE\_SID/pfile. Después creamos un enlace en \$ORACLE\_HOME/dbs (ubicación por defecto).

# EJEMPLO DE INIT.ORA

```
db_name=CURSOxy
db_block_size=2048
compatible = 10.2.0
control_files = (/u02/oradata/CURSOxy/control1.ctl,
/u03/oradata/CURSOxy/control2.ctl)
undo_management = auto
undo_tablespace = undo_rbs
background_dump_dest = /u01/app/oracle/admin/CURSOxy/bdump
core_dump_dest = /u01/app/oracle/admin/CURSOxy/cdump
user_dump_dest = /u01/app/oracle/admin/CURSOxy/udump
max_dump_file_size = 10240
sga_max_size = 80M
sga_target = 72M
db_cache_size = 0
shared_pool_size = 0
large_pool_size = 0
java_pool_size = 0
log_buffer = 2886656
log_checkpoint_interval = 0
log_checkpoint_timeout = 1800
pga_aggregate_target = 10M
processes = 30
remote_login_passwordfile=EXCLUSIVE
nls_territory=spain
nls_language=spanish
```

Nota. Es un fichero de texto que hay que mantener manualmente con un editor ASCII (vi, notepad, etc).



# PARÁMETROS BÁSICOS DE ORACLE 10g

Como curiosidad, son aquellos parámetros que sería obligatorio fijar para una instancia (los demás se podrían dejar por defecto). Realmente ésto no debe hacerse (dejar el resto de parámetros sin asignar para q tomen sus valores por defecto).

**COMPATIBLE**

**CONTROL\_FILES**

**DB\_BLOCK\_SIZE**

**DB\_CREATE\_FILE\_DEST**

**DB\_CREATE\_ONLINE\_LOG\_DEST**

**DB\_DOMAIN**

**DB\_NAME**

**DB\_RECOVERY\_FILE\_DEST**

**DB\_RECOVERY\_FILE\_DEST\_SIZE**

**INSTANCE\_NUMBER**

**JOB\_QUEUE\_PROCESSES**

**LOG\_ARCHIVE\_DEST\_n**

**LOG\_ARCHIVE\_DEST\_STATE\_n**

**NLS\_LANGUAGE**

**NLS\_TERRITORY**

**OPEN\_CURSORS**

**PROCESSES**

**REMOTE\_LISTENER**

**REMOTE\_LOGIN\_PASSWORDFILE**

**ROLLBACK\_SEGMENTS**

**SESSIONS**

**SHARED\_SERVERS**

**STAR\_TRANSFORMATION\_ENABLED**

**UNDO\_MANAGEMENT**

**UNDO\_TABLESPACE**

# CREAR LA BASE DE DATOS y borrarla

- Para crear una BD necesitamos:
  - Conectarnos al servidor Oracle como SYS AS SYSDBA, autenticándonos contra el S.O. o usando un fichero de claves.
  - Suficiente memoria para arrancar la instancia y espacio en disco para crear la BD.
- Para ubicar los ficheros que componen la BD:
  - Guardaremos, al menos, dos copias del fichero de control, en discos separados.
  - Multiplexaremos los redolog en discos diferentes (separados del resto de la BD).
  - Separaremos los ficheros de datos que provoquen contención en disco; por ejemplo: datos, índices, system (DD), temp y rollback.
- La BD la podemos crear con el asistente gráfico (en Linux “dbca”) o con el comando CREATE DATABASE:
  - Crearemos un fichero init.ora, y si queremos, un spfile.ora.
  - Arrancaremos la instancia con STARTUP NOMOUNT.
  - Crearemos la BD con el comando CREATE DATABASE.
  - Ejecutaremos los scripts catalog.sql y catproc.sql que están en \$ORACLE\_HOME/rdbms/admin.
- Desde 10g se puede borrar con “DROP DATABASE;” (sólo montada).

# EJEMPLO DE CREACION DE BASE DE DATOS

```
connect / as sysdba
startup nomount
CREATE DATABASE "CURSOxy"
    maxdatafiles 254
    maxinstances 1
    maxlogfiles 32
    character set WE8ISO8859P15
DATAFILE '/u02/oradata/CURSOxy/system01.dbf' SIZE 260M
    AUTOEXTEND ON NEXT 10M
    EXTENT MANAGEMENT LOCAL
SYSAux DATAFILE '/u02/oradata/CURSOxy/sysaux01.dbf' size 100M
    EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT AUTO
UNDO TABLESPACE UNDO_RBS1
    DATAFILE '/u03/oradata/CURSOxy/rbs01.dbf' SIZE 10M
DEFAULT TABLESPACE USERS
    DATAFILE '/u02/oradata/CURSOxy/users01.dbf' SIZE 10M
DEFAULT TEMPORARY TABLESPACE TEMP
    TEMPFILE '/u03/oradata/CURSOxy/temp01.dbf' SIZE 10M
    EXTENT MANAGEMENT LOCAL UNIFORM SIZE 64K
logfile '/u04/oradata/CURSOxy/redo01.log' SIZE 3M,
    '/u04/oradata/CURSOxy/redo02.log' SIZE 3M,
    '/u04/oradata/CURSOxy/redo03.log' SIZE 3M;
rem *** CREACION DE LAS VISTAS DEL DD ***
@$ORACLE_HOME/rdbms/admin/catalog.sql
@$ORACLE_HOME/rdbms/admin/catproc.sql
```

# ORACLE MANAGED FILES (OMF)

- OMF permite que Oracle se encargue de la creación de los ficheros que componen la BD, simplificando la administración de la misma.
- OMF se activa mediante dos parámetros de inicialización:
  - DB\_CREATE\_FILE\_DEST. Define el directorio donde se ubicarán los ficheros.
  - DB\_CREATE\_ONLINE\_LOG\_DEST\_N. Establece los directorios donde se guardarán los ficheros de control y redolog; donde N puede valer de 1 a 5.

Nota. Se pueden activar ambos parámetros o sólo uno ellos.

- Ejemplo para crear una BD, usando OMF, separando los ficheros redolog y de control del resto:
  - Parámetros de inicialización:
    - DB\_CREATE\_FILE\_DEST='/u02/oradata/CURSOxy'
    - DB\_CREATE\_ONLINE\_LOG\_DEST\_1='/u03/oradata/CURSOxy'
    - DB\_CREATE\_ONLINE\_LOG\_DEST\_2='/u04/oradata/CURSOxy'
  - Creación de la BD:

```
CREATE DATABASE "CURSOxy"  
  character set WE8ISO8859P15  
  UNDO TABLESPACE UNDO_RBS  
  DEFAULT TABLESPACE USERS  
  DEFAULT TEMPORARY TABLESPACE TEMP;
```

# ARRANCAR LA BASE DE DATOS

- Cuando arrancamos una BDO, pasa por varios estados hasta que finalmente queda accesible a los usuarios: nomount, mount y open.
- En el primer estado (**nomount**) se arranca la instancia: lectura del fichero de parámetros, creación de la SGA, arranque de los procesos background y apertura del fichero alert\$ORACLE\_SID.log.  
Nota: el fichero de parámetros se busca en \$ORACLE\_HOME/dbs, comenzando por spfile\$ORACLE\_SID.ora. Si no lo encuentra, sigue con spfile.ora, y finalmente init\$ORACLE\_SID.ora.
- Seguidamente la BD se monta (**mount**) abriendo el fichero de control y obteniendo de él los nombres de los ficheros que la componen: datafiles y redo log.
- Finalmente se abre la BD (**open**), procediendo a la apertura de los ficheros de datos (datafiles) y los ficheros redo log. El servidor oracle comprueba la consistencia de la base de datos, y si es necesario el proceso SMON inicia la recuperación de la instancia.

# COMANDO STARTUP

- Arranca la instancia y abre la BD. Permite parar el proceso de arranque de la BD en cualquiera de sus fases (NOMOUNT, MOUNT).
- STARTUP (abre la base de datos con el fichero de parámetros por defecto).
- STARTUP PFILE=/home/CURSO/cursoXY/miinit.ora
- STARTUP NOMOUNT (para crear la base de datos).
- STARTUP MOUNT (para renombrar datafiles, activar ARCHIVELOG o hacer una recuperación completa de la BD).
- STARTUP RESTRICT (sólo permite la conexión de usuarios con el privilegio RESTRICTED SESSION).
- STARTUP FORCE (hace SHUTDOWN ABORT y arranca la BD).

# COMANDO ALTER DATABASE Y “ENCOLAR Y SUSPENDER” LA BD

- Permite cambiar el estado de la base de datos, p.e. de NOMOUNT a MOUNT, o de MOUNT a OPEN. También se utiliza para dejar la BD en modo READ ONLY.
  - ALTER DATABASE {MOUNT | OPEN}
  - ALTER DATABASE OPEN [READ WRITE | READ ONLY]
- “Encolar” la BD. Util si el DBA necesita q no haya transacciones ni consultas concurrentes a la suya. Espera a q terminen transacciones/consultas (se pueden ver en V\$BLOCKING\_QUESCE) e impide nuevas (excepto de SYS/SYSTEM). Sólo desde SYS/SYSTEM. V\$INSTANCE.ACTIVE\_STATE:
  - ALTER SYSTEM QUIESCE RESTRICTED;
  - ALTER SYSTEM UNQUIESCE;
- “Suspende” la BD. Util para copias de seguridad en caliente. Suspende consultas y la E/S a ficheros de datos y control (deja terminar operaciones de E/S preexistentes y encola las nuevas). Sólo SYS/SYSTEM. V\$INSTANCE.DATABASE\_STATUS. ¡¡¡ Ojo, no cerrar la sesión q hace el SUSPEND pues es la única q puede hacer RESUME:
  - ALTER SYSTEM SUSPEND;
  - ALTER SYSTEM RESUME;

# PARAR LA BASE DE DATOS

- Hay determinadas operaciones que requieren parar la BD; como la actualización de algunos parámetros del init.ora; o hacer una copia física de la BD (copia en frio). La BD se para con el comando SHUTDOWN, impidiendo cualquier conexión posterior.
- SHUTDOWN [**NORMAL**], espera a que terminen todas las transacciones en curso y todas las sesiones, fuerza un checkpoint, además de cerrar todos los ficheros y destruir (parar) la instancia.
- SHUTDOWN **TRANSACTIONAL**, sólo espera a que terminen las transacciones en curso, fuerza un checkpoint, cierra los ficheros y destruye (para) la instancia.
- SHUTDOWN **IMMEDIATE**, hace rollback de todas las transacciones en curso y cierra todas las sesiones; cierra y desmonta la BD, además de forzar un checkpoint, cerrar ficheros y parar la instancia (como los anteriores).
- SHUTDOWN **ABORT**, cierra la instancia (destruye procesos background y SGA) sin esperar a desmontar ni cerrar la BD (como en una “caída”, ni hace checkpoint ni cierra ficheros)). Requiere recovery de la instancia al arrancar (lo hace automáticamente el proceso SMON).



# COMANDO SHUTDOWN

- Sintaxis:  
SHUTDOWN [NORMAL | TRANSACTIONAL | IMMEDIATE | ABORT]
- Tipos de parada. Cada una de las opciones de parada se comporta de forma diferente en cuanto a esperar a que terminen transacciones y sesiones, hacer checkpoint, o desmontar y cerrar la BD:

|                       | NORMAL | TRANSACTIONAL | IMMEDIATE | ABORT |
|-----------------------|--------|---------------|-----------|-------|
| Esperar Transacciones | S      | S             | N         | N     |
| Esperar Sesiones      | S      | N             | N         | N     |
| Checkpoint            | S      | S             | S         | N     |
| Cerrar BD             | S      | S             | S         | N     |
| Desmontar BD          | S      | S             | S         | N     |
| Parar instancia       | S      | S             | S         | S     |

# FICHERO alert.log

- Es el fichero de log de la BD y la primera referencia para el DBA en el “día a día” de la administración de la misma.
- Por defecto está en \$ORACLE\_HOME/rdbms/log; o en el directorio que indique el parámetro BACKGROUND\_DUMP\_DEST del init. Si usamos OFA, una ubicación típica es \$ORACLE\_BASE/admin/\$ORACLE\_SID/bdump.
- Recoge tanto información de estado como errores:
  - arranque y parada,
  - parámetros del init sin valores por defecto,
  - arranque de los procesos background,
  - cambio de fichero redolog (log switch),
  - creación de tablespaces y segmentos de rollback,
  - comandos alter (alter database, alter tablespace, etc),
  - errores ORA-600 y los que indican falta de espacio (llenado de tablas, índices, tablespaces, etc).

# EJEMPLO DE alert.log

```
Tue Oct 26 13:11:08 2006
Starting ORACLE instance (normal)
...
Starting up ORACLE RDBMS Version:
    10.2.0.2.0.
System parameters with non-default values:
  processes                = 30
  sga_max_size              = 83886080
  __shared_pool_size        = 50331648
  shared_pool_size          = 0
  __large_pool_size         = 4194304
  large_pool_size           = 0
  __java_pool_size          = 4194304
  java_pool_size            = 0
...
  db_cache_size             = 0
  compatible                = 10.2.0
  log_buffer                 = 2886656
...
  db_name                   = CURSOxy
  pga_aggregate_target       = 10485760
PMON started with pid=2, OS id=18002
PSP0 started with pid=3, OS id=18004
MMAN started with pid=4, OS id=18006
DBW0 started with pid=5, OS id=18008
LGWR started with pid=6, OS id=18010

CKPT started with pid=7, OS id=18012
SMON started with pid=8, OS id=18014
RECO started with pid=9, OS id=18016
MMON started with pid=10, OS id=18018
MMNL started with pid=11, OS id=18020
Tue Oct 26 13:11:08 2006
ALTER DATABASE      MOUNT
...
SMON: enabling cache recovery
Mon Oct 26 13:11:13 2006
Successfully onlined Undo Tablespace 1.
Mon Oct 26 13:11:13 2006
SMON: enabling tx recovery
Mon Oct 26 13:11:13 2006
Database Characterset is WE8ISO8859P15
...
Tue Oct 26 13:11:16 2006
Completed: ALTER DATABASE OPEN
Wed Oct 26 13:52:06 2006
Thread 1 advanced to log sequence 552
  Current log# 3 seq# 4 mem# 0:
    /home/u04/oradata/CURSOxy/redo03.log
```

Nota. En el ejemplo de arriba se puede ver el arranque de la instancia, los parámetros asignados, los procesos arrancados, como se monta la BD, y cómo finalmente se abre la BD y se empiezan a usar los fichero redolog.

# FICHEROS DE TRAZA DE LOS PROCESOS BACKGROUND

- Registran errores producidos en los procesos background de la instancia: LGWR, DBWR, SMON, PMON, etc.
- Se generan en el directorio indicado por el parámetro de inicialización `BACKGROUND_DUMP_DEST`, que por defecto es `$ORACLE_HOME/rdbms/log`. Si se utiliza la arquitectura OFA para ubicar los ficheros Oracle, una ubicación típica para estos ficheros de traza es `$ORACLE_BASE/admin/$ORACLE_SID/bdump`.
- Su nombre sigue el patrón `${ORACLE_SID}_nombreproceso_pid.trc`; por ejemplo, `cursoXY_smon_16432.trc` (los nombres de los ficheros de traza en Unix/Linux están siempre en minúsculas).

# FICHEROS DE TRAZA DE LOS PROCESOS DE USUARIO

- Recogen estadísticas de seguimiento de sentencias SQL o errores en las sesiones de usuario.
- Las trazas de usuario se generan en el directorio que indique el parámetro `USER_DUMP_DEST` (por defecto, `$ORACLE_HOME/rdbms/log`). Si usamos OFA, una ubicación típica es `$ORACLE_BASE/admin/$ORACLE_SID/udump`.
- Su tamaño está limitado por el parámetro `MAX_DUMP_FILE_SIZE`.
- Sus nombres siguen el patrón `${ORACLE_SID}_ora_pid.trc`; por ejemplo, `cursoXY_ora_23654.trc` (siempre en minúsculas).
- Pueden ser muy útiles para el ajuste de sentencias SQL. En este caso se pueden generar voluntariamente “activando la traza”:
  - A nivel de sesión con `“ALTER SESSION SET SQL_TRACE=TRUE;”`,
  - Desde una sesión del DBA con `“dbms_system.set_sql_trace_in_session(sid,true)”`, donde SID es el nº de sesión oracle (`V$SESSION`, `V$PROCESS`). Se desactiva de igual forma (indicando false en lugar de true).
  - A nivel de instancia, con el parámetro `“SQL_TRACE=TRUE”` del init.
- Utilidad `TKPROF`. Permite generar un informe “legible”, a partir de un fichero de traza generado explícitamente para una sesión.
  - `tkprof cursoXY_ora_23654.trc salida_traza.txt explain=scott/tiger sys=no`

# DICCIONARIO DE DATOS (DD)

- El DD está compuesto por un conjunto de tablas y vistas asociadas donde se almacena toda la información sobre los objetos que componen la BD, así como la estructura lógica y física de la misma.
- El DD incluye dos tipos de objetos: tablas base y vistas.
  - Las tablas base se crean automáticamente cuando creamos la BD con el comando `CREATE DATABASE`; y son las que realmente contienen la información del DD.
  - Las vistas se crean al lanzar el script `catalog.sql`; y permiten acceder a la información de las tablas del DD (que está codificada).
- El DD contiene información sobre: la definición de todos los objetos de la BD (tablas, vistas, índices, sinónimos, secuencias, procedimientos, funciones, paquetes, triggers, etc), el espacio ocupado por cada objeto, condiciones de integridad, usuarios, privilegios, roles, así como auditoría del sistema.

# VISTAS DEL DICCIONARIO DE DATOS

- El DD se modifica cada vez que lanzamos una sentencia DDL.
- Las vistas estáticas que forman parte del DD son de tres tipos: dba, all y user. Cada una de ellas tendrá un prefijo asociado que la ubica en uno de dichos tipos.
  - DBA: todos los objetos de la BD.
  - ALL: todos los objetos accesibles por el usuario actual.
  - USER: todos los objetos propiedad del usuario actual.
- La vista `DICTIONARY` contiene una lista de todas las vistas del DD; y en `DICT_COLUMNS` tenemos el detalle de las columnas de cada una de ellas.
- Ejemplos de vistas del DD:
  - Objetos de la BD: `dba_objects`, `dba_tables`, `dba_indexes`, `dba_tab_columns`, `dba_ind_columns`, `dba_constraints`, `dba_views`.
  - Espacio ocupado: `dba_data_files`, `dba_segments`, `dba_extents`.
  - Estructura de la BD: `dba_tablespace`s, `dba_data_files`.
- El DD también tiene las llamadas tablas dinámicas, cuyas vistas tienen el prefijo V\$ (como V\$SESSION). Se crean al arrancar la instancia y residen en memoria. Cuando cerramos la BD (y por tanto la instancia), desaparecen y con ellas su contenido.

# AUTOMATIC STORAGE MANAGEMENT (ASM) - I

- ASM ( $\geq 10g$ ). Gestor de volúmenes para bases de datos Oracle. Mejor rendimiento de E/S y fácil de gestionar. Maneja el espacio en forma de grupos de discos. Divide cada fichero en extensiones (de 128K o 1M) y las reparte entre los discos de un grupo (striping). Tb permite mirror, y lo hace a nivel de fichero (más granular q a nivel de disco), gestionando el mirror a nivel de extensión. Varias opciones de mirror (a nivel de grupos de discos): 2-way mirroring (1 copia por extensión), 3-way (2 copias) y unprotected (sin mirror). Opciones de striping: fine (128Kb) y coarse (1M). Se puede desactivar mirror y/o striping.
- Al añadir nuevo disco a un grupo, rebalancea los datos online. Si esto genera mucha E/S, se puede “frenar” con `ASM_POWER_LIMIT` (en el init de la propia instancia ASM).
- ASM necesita un tipo especial de instancia:
  - Tiene init y orapw, pero no DD. Usuarios SYS y SYSTEM con autenticación SO (no más usuarios).
  - Instancia: mount o no mount (nunca open). Memoria: de 60M a 120M.
  - Comandos de gestión propios: “create | alter | drop diskgroup”.
  - Nuevos procesos background:
    - RBAL: coordina la actividad de rebalanceo de los discos en cada grupo.
    - ORBn (n=0..9): ejecuta el rebalanceo, moviendo extensiones entre discos.



# AUTOMATIC STORAGE MANAGEMENT (ASM) - II

- Cada BD q usa ASM, tiene dos nuevos procesos background:
  - OSMB: comunicación entre BD y la instancia ASM.
  - RBAL: abre y cierra los discos en los grupos, en la parte de la BD.
- Parámetros del init (para instancia ASM y/o BD q la usa):
  - instance\_type = ASM (para una BD es RDBMS)
  - db\_unique\_name = +ASM (valor por defecto)
  - asm\_power\_limit = 1 (máximo 11; velocidad rebalanceo; 1 es el más lento)
  - asm\_diskstring (limita los dispositivos de disco usables para grupos de discos; ejemplo: ' /dev/hd\* ').
  - asm\_diskgroups (nombres de grupos de discos q se mostrarán automáticamente; por defecto vale NULL y los monta todos).
  - large\_pool\_size (al menos 8M, para ejecutar los paquetes internos de uso de ASM)
- Ejemplo de init de una instancia ASM

```
instance_type=ASM
db_unique_name=+ASM
asm_power_limit=1
asm_disk_string='/dev/vgora01/rdisk/*', '/dev/vgora02/rdisk/*'
asm_disk_groups=diskgrp1, diskgrp2
large_pool_size=16M
```

# VISTAS DEL DD

- V\$INSTANCE
- V\$DATABASE
- V\$SESSION
- V\$PROCESS
- V\$PARAMETER
- V\$PARAMETER2
- V\$SYSTEM\_PARAMETER
- V\$SYSTEM\_PARAMETER2
- DICTIONARY
- DICT\_COLUMNS
- V\$ASM\_DISKGROUP
- V\$ASM\_CLIENT
- V\$ASM\_DISK
- V\$ASM\_FILE
- V\$ASM\_TEMPLATE
- V\$ASM\_OPERATION

# PRACTICAS TEMA 2.

## 2.1. Fichero init.ora. Arranque y parada de la BD.

- Localizar el fichero init.ora (y spfile si existe) de nuestra BD:
  - `ls -l $ORACLE_HOME/dbs/init$ORACLE_SID.ora`
  - `ls -l $ORACLE_HOME/dbs/spfile$ORACLE_SID.ora`
- Anotar el valor de los parámetros: `db_block_size`, `sga_target`, `sga_max_size`, `shared_pool_size`, `db_cache_size`, `log_buffer`, `processes`.
  - `more $ORACLE_HOME/dbs/init$ORACLE_SID.ora`
  - `grep processes $ORACLE_HOME/dbs/init$ORACLE_SID.ora`
- Crear fichero `/home/CURSO/cursoxy/init01xy.ora`, copia del `initCURSOxy.ora`, y modificar `processes=17`.
  - `cp $ORACLE_HOME/dbs/init$ORACLE_SID.ora init01xy.ora`
  - `echo processes=17 >> init01xy.ora`
  - `tail init01xy.ora`
- Arrancar la BD y comprobar valor de parámetros, ¿por qué no ha tomado el nuevo valor?.
  - `STARTUP`
  - `show parameter processes`
- Parar la BD y arrancar con el `init01xy.ora`. Comprobar parámetro `processes`. Abrir otra conexión de `sqlplus` y ver qué ocurre.
  - `SHUTDOWN immediate`
  - `STARTUP pfile=init01xy.ora`
  - `show parameter processes`
- Parar la BD de forma normal, estando conectado algún usuario y ver qué ocurre. Y qué sucede cuando todos los usuarios se desconectan.
  - `SHUTDOWN`
- Repetir la parada de la BD, estando conectado algún usuario, de forma que no espere:
  - `SHUTDOWN IMMEDIATE`

# PRACTICAS TEMA 2.

## 2.2. Arranque de la BD (STARTUP). Comprobar las diferentes fases en el arranque de la BD.

- Arrancar sólo la instancia (NOMOUNT) y consultar algún parámetro. Qué ocurre al acceder a V\$CONTROLFILE.
  - `STARTUP NOMOUNT`
  - `SHOW PARAMETER processes`
  - `SELECT * FROM V$CONTROLFILE;`
- Ahora montar la BD y volver a consultar V\$CONTROLFILE. Que sucede al leer DBA\_USERS.
  - `ALTER DATABASE MOUNT`
  - `SELECT * FROM V$CONTROLFILE;`
  - `SELECT * FROM DBA_USERS;`
- Abrir la BD en modo READ ONLY y crear una tabla. Activar modo READ WRITE y volver a crear la tabla.
  - `ALTER DATABASE OPEN READ ONLY`
  - `CREATE TABLE MITABLA (C1 VARCHAR2(2));`
  - `ALTER DATABASE OPEN READ WRITE`
  - `CREATE TABLE MITABLA (C1 VARCHAR2(2));`

## 2.3. Fichero alert.log y ficheros de traza. Diccionario de datos.

- Buscar y consultar el fichero de alert de la BD.
  - `ls -l $ORACLE_HOME/rdbms/log/alert_$ORACLE_SID.log`
  - `ls -l $ORACLE_BASE/admin/$ORACLE_SID/bdump/alert_$ORACLE_SID.log`
  - `cat alert_$ORACLE_SID.log`
  - `tail -26f alert_$ORACLE_SID.log`
- Buscar si hay ficheros de traza.
  - `ls -l $ORACLE_HOME/rdbms/log/*.trc`
  - `ls -l $ORACLE_BASE/admin/$ORACLE_SID/bdump/*.trc`
  - `ls -l $ORACLE_BASE/admin/$ORACLE_SID/udump/*.trc`
- Sacar la lista de vistas del DD. Consultar las columnas de dichas vistas.
  - `Select * from DICTIONARY;`
  - `Select * from DICT_COLUMNS;`

# PRACTICAS TEMA 2.

**2.4. Impedir las conexiones de usuarios, de modo que el DBA sí pueda conectarse. Intenta conectarte como scott/tiger. Volver a permitir conexiones de usuarios.**

- STARTUP RESTRICT (si la BD está parada)
- ALTER SYSTEM ENABLE RESTRICTED SESSION; (si la BD estaba arrancada)
- ALTER SYSTEM DISABLE RESTRICTED SESSION;

**2.5. Forzar un checkpoint y hacer un insert en la tabla SCOTT.DEPT. Inmediatamente después hacer SHUTDOWN ABORT. Arrancar y comprobar SCOTT.DEPT. Volver a repetir el insert, haciendo commit antes del SHUTDOWN ABORT; y comprueba el contenido de SCOTT.DEPT.**

- ALTER SYSTEM CHECKPOINT;
- insert into scott.dept values (99,'FORMACION','MURCIA');
- SHUTDOWN ABORT
- STARTUP
- SELECT \* FROM SCOTT.DEPT;
- insert into scott.dept values (99,'FORMACION','MURCIA');
- COMMIT;
- SHUTDOWN ABORT
- STARTUP
- SELECT \* FROM SCOTT.DEPT;

# PRACTICAS TEMA 2.

**2.6. Conéctate como SCOTT y haz update sobre DEPT (sin hacer commit). Desde otra sesión, cierra la BD con SHUTDOWN TRANSACTIONAL. ¿Qué pasa al hacer commit en la sesión de SCOTT?**

- CONNECT SCOTT/TIGER
- SHUTDOWN TRANSACTIONAL (sesión del sys)
- update dept set deptno=88 where deptno=99; (sesión de scott)
- COMMIT; (sesión de scott)

**2.7. Conéctate como usuario scott/tiger. Activa la traza y haz una query con una join entre EMP y DEPT (select a.ename, b.dname from emp a, dept b where a.deptno=b.deptno;). Desactiva la traza y analiza el fichero que se ha generado con el comando tkprof (es un comando unix, no de SQL).**

- CONNECT SCOTT/TIGER
- ALTER SESSION SET SQL\_TRACE=TRUE;
- tkprof nombre\_fichero\_traza.trc salida\_traza.txt explain=scott/tiger sys=no (desde fuera SQL)
- cat salida\_traza.txt

# TEMA 3

## FICHERO DE CONTROL

# TEMA 3.

## FICHERO DE CONTROL

- Fichero de control
- Contenido del fichero de control
- Multiplexar fichero de control
- Backup del fichero de control



# FICHERO DE CONTROL

- Se trata de un fichero binario, sin el cual no es posible arrancar la BD. Por ello es conveniente mantener varias copias del mismo, en diferentes discos.
- Se lee al montar la BD.
- Su tamaño es fijo, y depende de los parámetros indicados al crear la BD con CREATE DATABASE; como por ejemplo MAXLOGFILES y MAXDATAFILES.
- El fichero de control contiene información como: nombre de la BD, fecha de creación de la BD, nombres de los tablespaces, nombre y localización de los ficheros de datos y de redo, número de secuencia del redo log en curso, información de checkpoint, información del archivado de los redo log, información de backup.

# AÑADIR COPIAS Y BACKUP DEL FICHERO DE CONTROL

- Para añadir una copia del fichero de control:
  - Se para la BD con SHUTDOWN NORMAL.
  - Se hace una copia física del fichero de control, a nivel del sistema operativo. En Unix con el comando cp.
  - Se incluye la nueva copia del fichero de control en el init.ora (o spfile); en el parámetro CONTROL\_FILES.
  - Arrancar la BD con STARTUP.
- Se recomienda sacar una copia de seguridad del fichero de control cada vez que cambie la estructura física de la BD:
  - ALTER DATABASE BACKUP CONTROLFILE TO TRACE;  
De esta forma se generan, en un fichero de traza, las sentencias sql necesarias para volver a crear el fichero de control.
  - ALTER DATABASE BACKUP CONTROLFILE TO  
'/u02/oradata/CURSOxy/ora\_control01.bak';  
Hace una copia binaria y aislada del fichero.
- En la vista V\$CONTROLFILE tenemos la lista de todos los ficheros de control de la BD. En V\$CONTROLFILE\_RECORD\_SECTION veremos las diferentes secciones y su estado de uso.

# ALTER DATABASE BACKUP CONTROLFILE TO TRACE

```
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE DATABASE "CURSOxy" NORESETLOGS
    NOARCHIVELOG
    MAXLOGFILES 32
    MAXLOGMEMBERS 2
    MAXDATAFILES 1000
    MAXINSTANCES 1
    MAXLOGHISTORY 292
LOGFILE
    GROUP 1 '/u04/oradata/CURSOxy/redo01.log' SIZE 10M,
    GROUP 2 '/u04/oradata/CURSOxy/redo02.log' SIZE 10M,
    GROUP 3 '/u04/oradata/CURSOxy/redo03.log' SIZE 10M
DATAFILE
    '/u02/oradata/CURSOxy/system01.dbf' ,
    '/u03/oradata/CURSOxy/undo_rbs01.dbf' ,
    '/u03/oradata/CURSOxy/sysaux01.dbf' ,
    '/u02/oradata/CURSOxy/users01.dbf'
CHARACTER SET WE8ISO8859P15;
ALTER DATABASE OPEN;
```

Nota: si hubiese que indicar la opción RESETLOGS, se perderá el contenido de los ficheros redolog.

# VISTAS DEL DD

- V\$CONTROLFILE
- V\$CONTROLFILE\_RECORD\_SECTION
- V\$PARAMETER

# PRACTICAS TEMA 3

3.1. Localizar el fichero de control desde el SO y desde la BD.

- `ls -lt /u0?/oradata/$ORACLE_SID/*.ctl`
- `ls -l /u0?/oradata/$ORACLE_SID/*control*`
- `select * from v$controlfile;`

3.2. Consultar la información de la BD relativa al contenido del fichero de control.

Forzar un checkpoint y volver a consultar. Consultar las secciones que contiene el fichero de control y su estado de uso.

- `select * from v$database;`
- `alter system checkpoint;`
- `select * from v$controlfile_record_section;`

# PRACTICAS TEMA 3

3.3. Añade una copia al fichero de control de la BD en "/u04/oradata/\$ORACLE\_SID". Crea un initxy02.ora e incluye el nuevo fichero. Arranca la BD con el nuevo init y comprueba que ha tomado la nueva copia del fichero de control.

- shutdown immediate
- cp /u02/oradata/\$ORACLE\_SID/control11.ctl /u04/oradata/\$ORACLE\_SID/control13.ctl
- cp \$ORACLE\_HOME/dbs/init\$ORACLE\_SID.ora init02xy.ora
- vi init02xy.ora
- startup pfile=init02xy.ora
- select \* from v\$controlfile;

3.4. Sacar una copia de seguridad del fichero de control, tanto en un fichero de traza, como un nuevo fichero de control.

- alter database backup controlfile to trace;
- ls -lt \$ORACLE\_BASE/admin/\$ORACLE\_SID/udump|head -2
- alter database backup controlfile to '/u02/oradata/CURSOxy/ora\_control11.bak';
- ls -lt /u02/oradata/\$ORACLE\_SID/control\_copia1.ctl
- select \* from v\$controlfile;

# TEMA 4

## GESTION DEL REDO LOG

# TEMA 4.

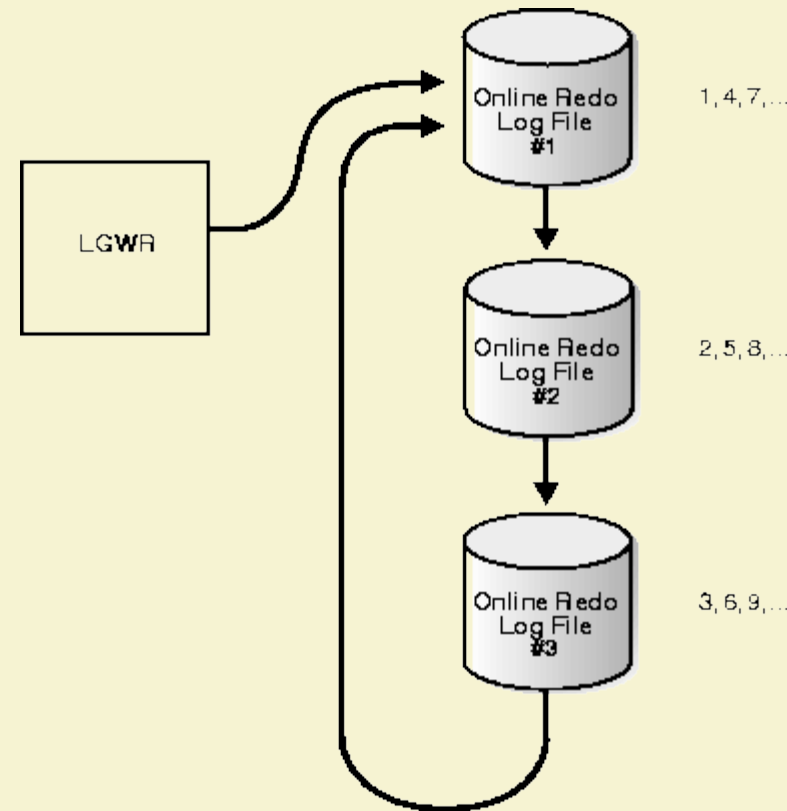
## GESTION DEL REDO LOG

- Ficheros redo log
- Funcionamiento del redo log
- Añadir/quitar grupos/miembros de redo
- Configuración de los ficheros redo log
- Modo archivelog



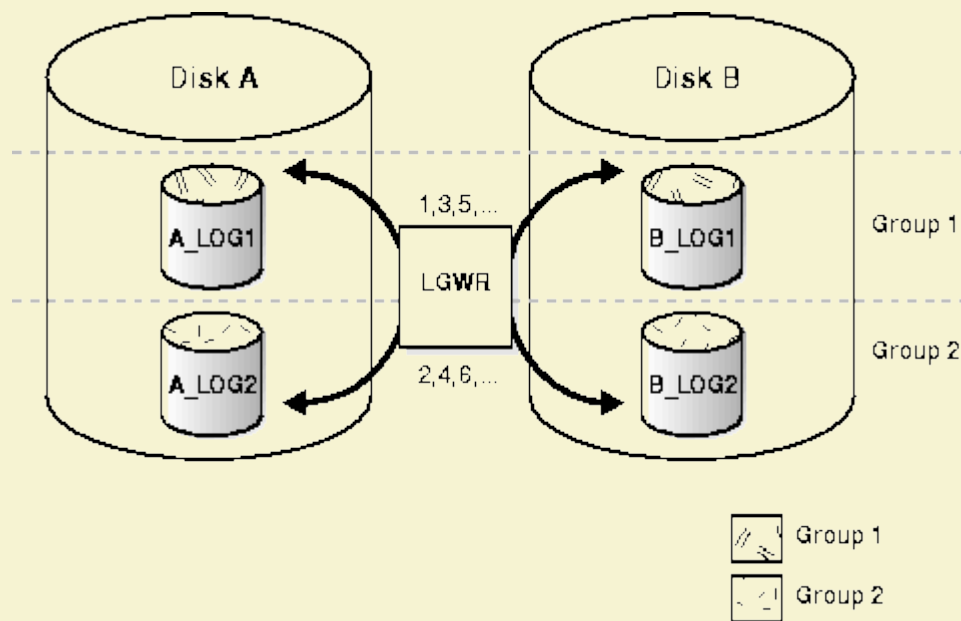
# FICHEROS REDO LOG

- Los ficheros redo log guardan todos los cambios hechos en los datos y permiten volver a aplicarlos en caso de caída de la BD.
- Los ficheros redo log se organizan en grupos. Una BD requiere al menos dos grupos. Cada fichero redo log dentro de un grupo se llama miembro.
- La idea es que cada grupo tenga tantos miembros como discos disponemos para mantener las copias de los redo. Lo usual es tener 3 grupos de redo con 2 miembros cada uno.



# FUNCIONAMIENTO DEL REDO LOG

- Los ficheros redo log se usan de manera circular: cuando uno se llena, el LGWR comienza a escribir en el siguiente grupo (“log switch”), hasta volver al primero. Cuando ocurre un “log switch”, también sucede un “checkpoint”; y se actualiza el fichero de control. Podemos forzar un log switch o un checkpoint explícitamente con:
  - ALTER SYSTEM SWITCH LOGFILE;
  - ALTER SYSTEM CHECKPOINT;
- El LGWR escribe al hacer commit, o cada 3 segundos, o si el buffer redolog se llena 1/3, y antes de que el DBWR vuelque los cambios de los buffers de datos a los ficheros de la BD.



# AÑADIR GRUPOS Y MIEMBROS DE REDO

- Añadir grupos al Redo Log Online:

```
ALTER DATABASE ADD LOGFILE GROUP 3  
  ('/u04/oradata/CURSOxy/redo03a.log',  
   '/u03/oradata/CURSOxy/redo03b.log')  
SIZE 1M;
```

- Añadir miembros Redo Log Online :

```
ALTER DATABASE ADD LOGFILE MEMBER  
  '/u03/oradata/CURSOxy/redo01b.log' TO GROUP 1,  
  '/u03/oradata/CURSOxy/redo02b.log' TO GROUP 2;
```

# ELIMINAR GRUPOS Y MIEMBROS DE REDO

- Eliminar grupos del Redo Log Online (por ejemplo si he creado otros más grandes):

```
ALTER DATABASE DROP LOGFILE GROUP 3;
```

No puede haber menos de 2 grupos; no se puede borrar el grupo activo; al borrar un grupo no se eliminan los ficheros del sistema operativo (a no ser que se use OMF).

- Eliminar miembros Redo Log Online:

```
ALTER DATABASE DROP LOGFILE MEMBER  
  '/u03/oradata/CURSOxy/redo03b.log';
```

No se puede borrar el último miembro q quede de un grupo (y dejarlo vacío); tampoco un miembro del grupo en curso; si la BD está en modo ARCHIVELOG no se puede borrar un miembro cuyo grupo no ha sido archivado; cuando borramos un miembro no se elimina el fichero correspondiente del sistema operativo.

# CONFIGURACIÓN DE LOS FICHEROS REDO LOG

- Vaciado (por ejemplo si todos los miembros de un grupo están corruptos):

```
ALTER DATABASE CLEAR [UNARCHIVED] LOGFILE  
    '/u02/oradata/CURSOxy/redo02a.log';
```

Es equivalente a añadir y borrar un fichero redolog.

- Mover o renombrar (¡ojo! la BD debe estar sólo montada):

```
!mv /u04/oradata/CURSOxy/redo03a.log  
    /u03/oradata/CURSOxy/redo3a.log  
ALTER DATABASE RENAME FILE '/u04/oradata/CURSOxy/redo03a.log' TO  
    '/u03/oradata/CURSOxy/redo3a.log';
```

- Configuración del Redo Log Online:

- El número de grupos Redo Log Online necesarios, como mínimo será dos. Es probable que se necesite alguno más debido a que, al llenarse circularmente, los checkpoints no completen. La configuración debe ser simétrica: mismo número de miembros para todos los grupos.
- Cada miembro de un grupo debe estar en un disco diferente. Además hay que separar en diferentes discos los Redo Log Archivados de los Redo Log Online, para reducir la contención entre el LGWR y el ARCH. El Redo Log Online también debería estar en un disco distinto a los ficheros de datos, para reducir la contención entre LGWR y DBWR.

- En las vistas V\$LOG, V\$LOG\_HISTORY y V\$LOGFILE están los detalles del redo.

# MODO ARCHIVELOG

- Por defecto, la BD se crea en modo NOARCHIVELOG (con CREATE DATABASE). Si activamos el modo ARCHIVELOG se irán archivando los ficheros redo log conforme se llenan (cada vez que ocurre un “log switch”).
- LOG\_ARCHIVE\_START=TRUE, activa archivado automático (en 10g no hace falta). El proceso ARCH irá archivando el grupo redo log lleno, después de cada “log switch”, en el directorio indicado por el parámetro LOG\_ARCHIVE\_DEST (por defecto \$ORACLE\_HOME/dbs/arch).  
Nota. Se puede ver el estado del archivado con el comando “archive log list” del sqlplus.
- Cada vez que se archiva un redo log, en el fichero de control se guarda el nombre del redolog archivado, número de secuencia, y números SCN más alto y más bajo.
- El redolog que se ha llenado no puede reutilizarse hasta que ocurra un checkpoint y haya sido copiado por el proceso ARCH.
- Poner BD en modo ARCHIVELOG: SHUTDOWN, backup (por seguridad), configurar log\_archive\_dest en el INIT, STARTUP MOUNT, activar archivado (ALTER DATABASE ARCHIVELOG;), abrir BD (ALTER DATABASE OPEN;), parar BD, y hacer backup (pues ha cambiado el fichero de control y la copia anterior ya no nos sirve).

Más información en “<http://cursos.atica.um.es/oradoc102/server.102/b14231/archredo.htm#i1006246>”

# VISTAS DEL DD

- V\$LOG
- V\$LOG\_HISTORY
- V\$LOGFILE
- V\$DATABASE

# PRACTICAS TEMA 4

4.1. Localizar los ficheros redolog de la BD. ¿Cuántos grupos hay y cuántos miembros tiene cada grupo? ¿están correctamente distribuidos?

- `ls -lt /u03/oradata/$ORACLE_SID/*.log`
- `ls -lt /u03/oradata/$ORACLE_SID/*redo*`
- `select * from v$logfile;`

4.2. Comprobar el fichero redo log activo. ¿Qué ocurre al forzar un "log switch"? ¿y al forzar un checkpoint?

- `select * from v$log;`
- `alter system switch logfile;`
- `alter system checkpoint;`

4.3. Añade un miembro más a cada grupo: `/u03/oradata/$ORACLE_SID/redo11.log`, `/u03/oradata/$ORACLE_SID/redo12.log`, `/u03/oradata/$ORACLE_SID/redo13.log`.

```
alter database add logfile member  
  '/u03/oradata/CURSOxy/redo11.log' to group 1,  
  '/u03/oradata/CURSOxy/redo12.log' to group 2,  
  '/u03/oradata/CURSOxy/redo13.log' to group 3;
```



# PRACTICAS TEMA 4

4.4. Añade un grupo más (grupo 4), con dos miembros de 1M: /u03/oradata/\$ORACLE\_SID/redo04.log y /u04/oradata/\$ORACLE\_SID/redo14.log. Añade 2 grupos más (grupo 5 y 6), con las mismas características.

- alter database add logfile group 4 ('/u03/oradata/CURSOxy/redo04.log', '/u04/oradata/CURSOxy/redo14.log') size 1M;
- alter database add logfile group 5 ('/u03/oradata/CURSOxy/redo05.log', '/u04/oradata/CURSOxy/redo15.log') size 1M;
- alter database add logfile group 6 ('/u03/oradata/CURSOxy/redo06.log', '/u04/oradata/CURSOxy/redo16.log') size 1M;

4.5. Elimina los miembros del grupo 1, de uno en uno. ¿Qué ocurre al eliminar el último?. Borrar los grupos 1, 2 y 3. ¡¡¡ Ojo y no borrar el redo log activo !!!

- Alter database drop logfile member '/u04/oradata/CURSOxy/redo01.log';
- alter database drop logfile member '/u03/oradata/CURSOxy/redo11.log';
- alter database drop logfile group 1;
- alter database drop logfile group 2;
- alter database drop logfile group 3;

# PRACTICAS TEMA 4

4.6. Cambiar el nombre de los miembros de redo de los grupos 4, 5 y 6; a redo1a.log, redo1b.log, redo2a.log, redo2b.log, redo3a.log, redo3b.log.

- shutdown immediate

iii OJO, la BD debe estar sólo montada!!!

- startup mount
- mv /u03/oradata/CURSOxy/redo01.log /u03/oradata/CURSOxy/redo1a.log
- mv /u04/oradata/CURSOxy/redo11.log /u04/oradata/CURSOxy/redo1b.log
- mv /u03/oradata/CURSOxy/redo02.log /u03/oradata/CURSOxy/redo2a.log
- mv /u04/oradata/CURSOxy/redo12.log /u04/oradata/CURSOxy/redo2b.log
- mv /u03/oradata/CURSOxy/redo03.log /u03/oradata/CURSOxy/redo3a.log
- mv /u04/oradata/CURSOxy/redo13.log /u04/oradata/CURSOxy/redo3b.log
- alter database rename file '/u03/oradata/CURSOxy/redo01.log' to '/u03/oradata/CURSOxy/redo1a.log';
- alter database rename file '/u04/oradata/CURSOxy/redo11.log' to '/u04/oradata/CURSOxy/redo1b.log';
- alter database rename file '/u03/oradata/CURSOxy/redo02.log' to '/u03/oradata/CURSOxy/redo2a.log';
- alter database rename file '/u04/oradata/CURSOxy/redo12.log' to '/u04/oradata/CURSOxy/redo2b.log';
- alter database rename file '/u03/oradata/CURSOxy/redo03.log' to '/u03/oradata/CURSOxy/redo3a.log';
- alter database rename file '/u04/oradata/CURSOxy/redo13.log' to '/u04/oradata/CURSOxy/redo3b.log';
- alter database open;
- select \* from v\$logfile;

# TEMA 5

## TABLESPACES

# TEMA 5.

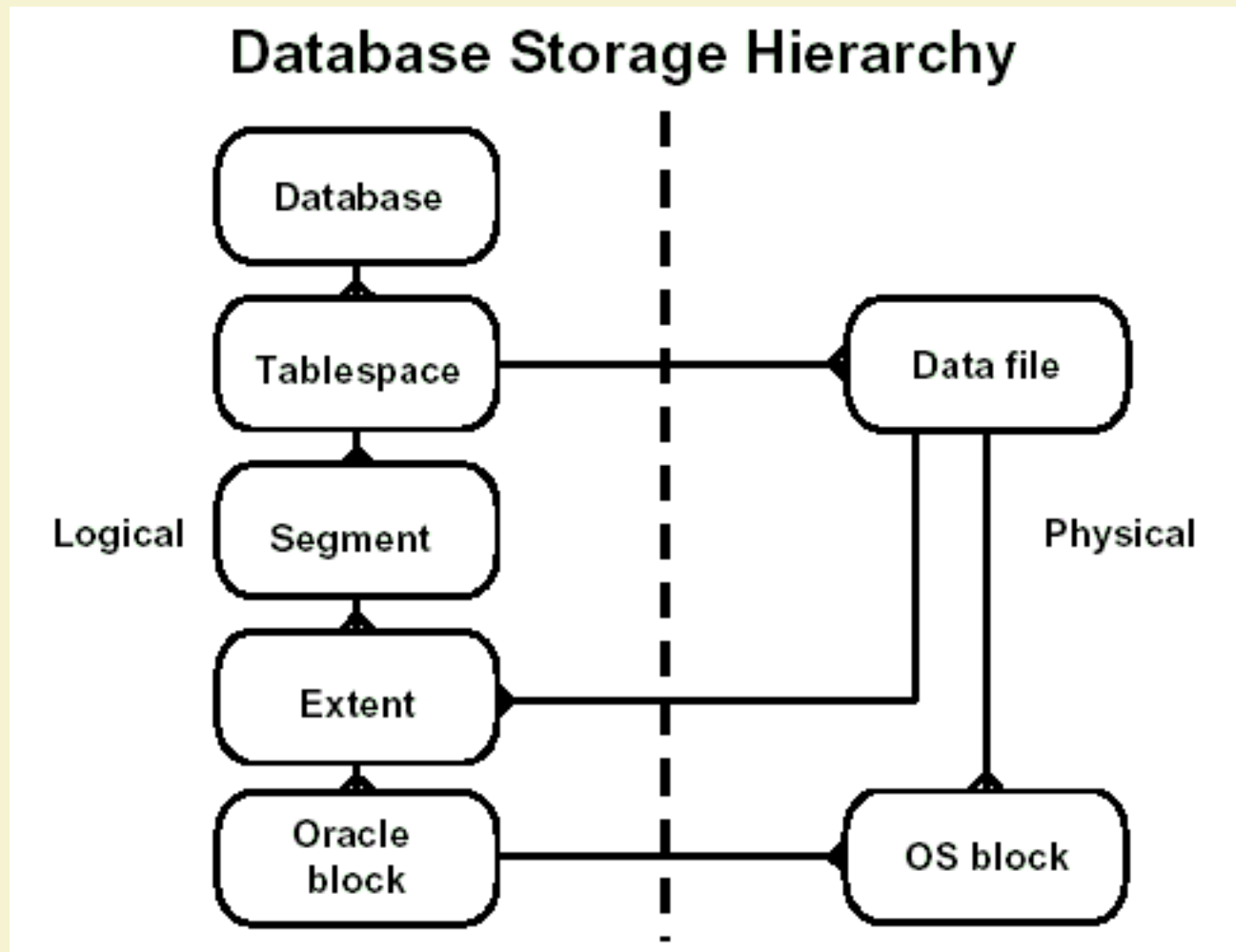
## TABLESPACES

- Qué es un tablespace
- Tablespaces system y sysaux
- Create tablespace
- Formas de gestionar el espacio de un tablespace
- Tablespaces “undo”
- Tablespaces temporales
- Tablespace temporal por defecto de la BD y tablespace por defecto de la BD
- Tablespace offline y rename
- Tablespace read-only y bigfile
- Borrar un tablespace y grupos
- Redimensionar un tablespace
- Mover ficheros

# QUE ES UN TABLESPACE

- Una BD 10g podría tener sólo los tablespaces SYSTEM y SYSAUX ( $\geq 10g$ ). Oracle recomienda crear tablespaces adicionales para los datos, índices, rollback y segmentos temporales.
- Según la estructura física, una BD está compuesta por: el fichero de control, ficheros redo log y ficheros de datos. La estructura lógica nos dice que está compuesta de tablespaces, segmentos, extensiones y bloques.
- Cada tablespace consiste en uno o más ficheros del s.o. llamados ficheros de datos (un fichero pertenece a un solo tablespace):
  - Un tablespace sólo puede pertenecer a una BD a la vez.
  - Puede tener cero o más segmentos (un segmento sólo pertenece a un tablespace).
  - Exceptuando el tablespace SYSTEM, o aquellos que contengan segmentos de rollback activos, un tablespace se puede poner offline, estando la BD funcionando.
  - Un tablespace se puede poner en modo read-only o read-write, y viceversa.
- Tipos de tablespaces: permanent (datos: system, sysaux, aplicaciones), undo (rollback) y temporary (sort).
- Oracle 10g permite crear “bigfile tablespaces”, de hasta 8EB (millones de terabytes).

# TABLESPACES



# TABLESPACES SYSTEM Y SYSAUX

- Los tablespaces SYSTEM y SYSAUX son los únicos que, cómo mínimo, se crean con la BD (create database).
- El tablespace SYSTEM (No debe contener datos de aplicaciones):
  - Contiene el DD, incluidos procedimientos almacenados, funciones, triggers y paquetes.
  - También alberga al segmento de rollback system
- El tablespace SYSAUX (>=10g) permite que en el tablespace SYSTEM sólo esté el DD, aglutinando las utilidades del sistema (Repositorio OEM, Intermedia, Spatial, OLAP, RMAN, XML DB, etc).
  - ¿Qué hay en el tablespace SYSAUX y cuánto ocupa?  

```
select occupant_name, space_usage_kbytes from v$sysaux_occupants;
```
  - ¿Se puede mover el contenido de SYSAUX a otro tablespace?  

```
select occupant_name, move_procedure, move_procedure_desc from  
v$sysaux_occupants;
```
- Respecto al resto de tablespaces (no SYSTEM), se recomienda separar los ficheros de redo, luego datos de índices, después rollback y segmentos temporales. También es bueno separar datos estáticos y dinámicos.

# CREAR UN TABLESPACE

## CREATE [BIGFILE] TABLESPACE

nombre

[DATAFILE cláusula\_fichero]

[MINIMUM EXTENT n[K|M]]

[BLOCKSIZE n[K]]

[LOGGING|NOLOGGING]

[cláusula\_extensiones]

[DEFAULT cláusula\_storage]

[ONLINE|OFFLINE]

[PERMANENT|TEMPORARY];

cláusula\_fichero ::= nombre\_fichero

[SIZE n[K|M] [REUSE] | REUSE]

[AUTOEXTEND ON|OFF [NEXT n[K|M]]

[MAXSIZE n[K|M]]]

cláusula\_extensiones ::= EXTENT MANAGEMENT

[DICTIONARY | LOCAL [AUTOALLOCATE |

UNIFORM [SIZE n[K|M]]

[SEGMENT SPACE MANAGEMENT AUTO | MANUAL]]]

cláusula\_storage ::= STORAGE (INITIAL

n[K|M] [NEXT n[K|M]] [MINEXTENTS n]

[MAXEXTENTS n] [PCTINCREASE n])

*CREATE TABLESPACE DATOS\_USUARIOS*

*DATAFILE '/u02/oradata/CURSOxy/datos\_usuarios01.dbf' SIZE 16M*

*AUTOEXTEND ON NEXT 1M MAXSIZE 32M*

*EXTENT MANAGEMENT LOCAL UNIFORM SIZE 128K*

*SEGMENT SPACE MANAGEMENT AUTO;*

*CREATE TABLESPACE DATOS\_USUARIOS*

*DATAFILE '/u02/oradata/CURSOxy/datos\_usuarios01.dbf' SIZE 16M*

*AUTOEXTEND ON NEXT 1M MAXSIZE 32M*

*EXTENT MANAGEMENT DICTIONARY*

*DEFAULT STORAGE (INITIAL 16K NEXT 32K MAXEXTENTS 10 PCTINCREASE 50);*



# FORMAS DE GESTIONAR EL ESPACIO DE UN TABLESPACE

- Tablespaces manejados localmente (Oracle los recomienda  $\geq 8i$ ):
  - Las extensiones libres se registran en un bitmap, de forma que cada bit corresponde a un bloque. El valor de cada bit indica si el bloque correspondiente está libre o usado. Existe un bitmap de este tipo en cada fichero del tablespace. Cada vez que una extensión se reserva o se libera, se modifica el bitmap correspondiente.
  - Ventajas:
    - Reducción de la contención en las tablas del DD.
    - No se genera rollback al reservar/liberar espacio (pues no se actualiza el DD).
    - No es necesario hacer “coalesce”.
  - No se puede usar INITIAL\_EXTENT, NEXT\_EXTENT, MIN\_EXTENTS, MAXEXTENTS y PCTINCREASE del STORAGE al crear una tabla. Tampoco tiene sentido DEFAULT STORAGE del tablespace.
  - El tablespace system se puede “manejar localmente”, desde Oracle9i (en 8i no). Si el SYSTEM es “local”, NO se pueden crear tablespaces “por diccionario”. Desde Oracle9i, por defecto, los tablespaces se crean como “locales” (si el parámetro compatible  $\geq 9.0$ ), excepto el SYSTEM.
- Tablespaces gestionados a través del DD (a extinguir):
  - Es el método por defecto en Oracle8i. Las extensiones libres quedan registradas en tablas del DD. Cada vez que una extensión se libera o se reserva, las tablas correspondientes del DD deben ser actualizadas.
  - Permite definir STORAGE flexible a los segmentos (los “locales” NO).

# TABLESPACES “UNDO”

- En Oracle 9i se introducen los segmentos de undo automáticos, que permiten sustituir la gestión manual de segmentos de rollback. Con Oracle 10g todavía se puede optar por la gestión manual, pero se avisa de que ya no estará disponible en futuras versiones.
- Los tablespaces “undo” sólo pueden contener segmentos de rollback (ningún otro tipo de objeto). Los llamaremos tablespaces de rollback.
- Son del tipo “manejados localmente” (de forma automática).

```
CREATE UNDO TABLESPACE undo01  
DATAFILE '/u03/oradata/CURSOxy/undo01.dbf' SIZE 100M;
```

- Un tablespace de rollback sólo se usa cuando se activa la gestión automática de rollback en la BD (undo\_management=auto y undo\_tablespace=nombre\_tablespace). Sólo puede haber un tablespace de rollback activo en un momento dado.

# TABLESPACES TEMPORALES Y GRUPOS

- Los segmentos temporales (o de sort) se crean en tablespaces temporales, automáticamente, para operaciones de ordenación (order by, joins, create index, etc) que no caben en memoria. Existen sólo durante la ejecución de la sentencia SQL.
- Un tablespace temporal no puede contener objetos permanentes.
- Es recomendable que sean “locally managed” (no pueden usar AUTOALLOCATE ni SEGMENT SPACE MANAGEMENT AUTO) y usen ficheros temporales:

```
CREATE TEMPORARY TABLESPACE temp
TEMPFILE '/u03/oradata/CURSOxy/temp01.dbf' SIZE 100M
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 2M;
```

- Para optimizar el rendimiento es recomendable que UNIFORM SIZE sea múltiplo de SORT\_AREA\_SIZE.
- No se puede renombrar un fichero temporal (tempfile). Tampoco se pueden poner en modo read-only. Siempre tienen el modo NOLOGGING (no producen entradas de redo log).
- Desde Oracle10g se pueden agrupar tablespaces temporales.

```
ALTER TABLESPACE temp1 TABLESPACE GROUP gtemp;
ALTER TABLESPACE temp2 TABLESPACE GROUP gtemp;
ALTER TABLESPACE temp3 TABLESPACE GROUP gtemp;
ALTER TABLESPACE temp3 TABLESPACE GROUP '';
ALTER USER nombre_usuario TEMPORARY TABLESPACE gtemp;
```

# TABLESPACE TEMPORAL POR DEFECTO DE LA BD Y TABLESPACE POR DEFECTO

- Desde Oracle9i, al crear la BD se puede (y se debe) indicar un **tablespace temporal por defecto** para aquellos usuarios a los que no se le asigne uno explícitamente. Si no se hace así, por defecto, se asignará el tablespace SYSTEM (ésto hay que evitarlo a toda costa).
- EL tablespace temporal por defecto de la BD se puede cambiar:  

```
ALTER DATABASE DEFAULT TEMPORARY TABLESPACE nombre_tablespace;
```

Nota: nombre\_tablespace podría ser un nombre de grupo.
- También se puede crear con la BD (CREATE DATABASE), en cuyo caso, será del tipo “local”.
- Al asignar a la BD un tablespace temporal por defecto, todos los usuarios que no tengan uno asignado explícitamente, pasarán a tenerlo. Y cuando cambiemos el tablespace temporal por defecto de la BD, cambiará para todos los usuarios que no lo tengan asignado de forma explícita.
- No podemos borrarlo hasta que hayamos asignado otro. No podemos ponerlo offline.
- Desde Oracle10, además, se puede definir un **tablespace por defecto** para los usuarios a los que no se le asigna uno explícitamente, en el momento de la creación de la BD con la cláusula `DEFAULT TABLESPACE nombre`. Además se puede cambiar en cualquier momento con (¡¡¡jojo!!!, se cambiará para todos los usuarios, incluso los q tuviesen asignado uno concreto, excepto especiales como SYS, SYSTEM, DBSNMP, OUTLN, etc):

```
ALTER DATABASE DEFAULT TABLESPACE nombre;
```

# TABLESPACE OFFLINE Y RENAME

- Cuando un tablespace está OFFLINE no se puede acceder a los datos que contiene:

```
ALTER TABLESPACE DBA01USER OFFLINE;
```

- Para ponerlo de nuevo ONLINE:

```
ALTER TABLESPACE DBA01USER ONLINE;
```

- Algunos tablespaces no se pueden poner OFFLINE: SYSTEM, tablespaces con segmentos de rollback o temporales activos.

- Sintaxis

```
ALTER TABLESPACE nombre ONLINE | OFFLINE;
```

- Los segmentos que contiene pueden ser borrados (por ejemplo “drop table”, porque sólo afectan al DD). En tablespaces “locales”, el segmento borrado pasa a ser del tipo temporal.
- Desde Oracle 10g, se puede renombrar un tablespace (excepto SYSTEM y SYSAUX), incluso estando READ ONLY:

```
ALTER TABLESPACE nombre1 RENAME TO nombre2;
```

# TABLESPACE READ-ONLY Y BIGFILE

- Al poner un tablespace en modo READ-ONLY, sólo se permiten operaciones de lectura sobre sus datos; sin embargo los segmentos que contiene pueden ser borrados (por ejemplo “drop table”, porque sólo afectan al DD). En tablespaces “locales”, el segmento borrado pasa a ser del tipo temporal.
- Un tablespace READ-ONLY podría residir en un CD-ROM o un DVD-ROM.:
  - `ALTER TABLESPACE tsp_lectura READ ONLY;`
  - Moveríamos el fichero correspondiente al dispositivo de sólo lectura.
  - `ALTER TABLESPACE tsp_lectura RENAME DATAFILE  
'/u02/oradata/CURSOxy/tsp_lectura.dbf' to  
'/mnt/cdrom/CURSOxy/tsp_lectura.dbf';`
- Sintaxis:  
`ALTER TABLESPACE nombre READ [ONLY | WRITE];`
- Un tablespace BIGFILE puede tener hasta 8Exabytes (millones de Tb). No pueden serlo ni SYSTEM ni SYSAUX.

```
CREATE BIGFILE TABLESPACE nombre DATAFILE  
'/u02/oradata/CURSOxy/nombre01.dbf' size 50G EXTENT  
MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT AUTO;
```

# BORRAR UN TABLESPACE

- Al borrar un tablespace, se elimina del DD. No podemos borrar SYSTEM/SYSAUX. Los ficheros asociados no se borran (hacerlo desde el SO después de eliminar el tablespace), a no ser que usemos INCLUDING CONTENTS AND DATAFILES (>= Oracle9i).
- No podremos borrarlo si contiene objetos, a menos que indiquemos INCLUDING CONTENTS. Tampoco podemos hacerlo si existen “foreign keys” apuntando a sus objetos, a menos que además indiquemos CASCADE CONSTRAINTS (se borrarán las citadas constraints FK).
- Se recomienda poner el tablespace OFFLINE antes de borrarlo, para asegurarnos que nadie está usando su contenido.

```
DROP TABLESPACE nombre_tablespace  
[INCLUDING CONTENTS [AND DATAFILES] [CASCADE CONSTRAINTS]];
```

# REDIMENSIONAR UN TABLESPACE

- Podemos cambiar el tamaño de un tablespace añadiendo un fichero al mismo, o bien cambiando el tamaño del fichero que lo compone.

- `ALTER TABLESPACE users ADD DATAFILE  
'/u02/oradata/CURSOxy/users02.dbf' size 1M;`
  - `ALTER DATABASE DATAFILE  
'/u02/oradata/CURSOxy/users01.dbf' resize 2M;`

Podemos reducir el tamaño de un fichero, pero no podremos hacerlo si hay espacio ocupado al final del mismo.

- Una opción interesante es programar el crecimiento del fichero que compone el tablespace:

- `ALTER DATABASE DATAFILE  
'/u02/oradata/CURSOxy/users01.dbf' SIZE 1M AUTOEXTEND  
ON NEXT 1M MAXSIZE 4M;`

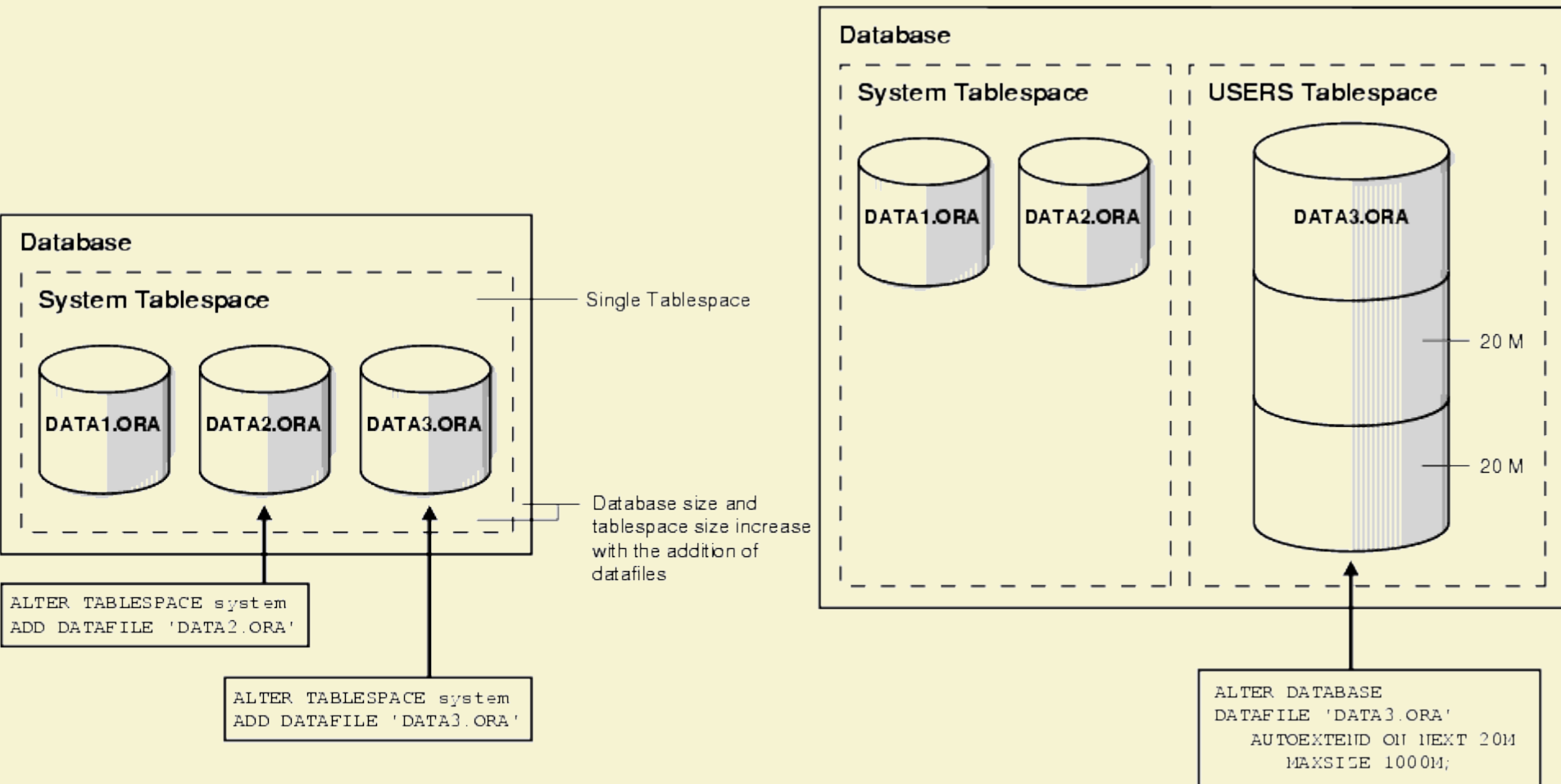
Podemos indicar UNLIMITED como MAXSIZE. Y las unidades también pueden ser K (p.e. 512K) o bytes (p.e. 100000).

- Oracle 10g introduce “BIGFILE tablespaces” (SYSTEM/SYSAUX no), de hasta 8EB. Sólo éstos se pueden redimensionar con ALTER TABLESPACE (sin indicar DATAFILE):

- `ALTER TABLESPACE bigtbs RESIZE 60G;`



# REDIMENSIONAR UN TABLESPACE



# MOVER FICHEROS

- Existen dos métodos para mover ficheros: con ALTER TABLESPACE y con ALTER DATABASE.
- El primero sólo es aplicable a tablespaces que no son el SYSTEM, y que no contienen segmentos de rollback o temporales activos:
  - Poner el tablespace offline
  - Mover el fichero a nivel del S.O.
  - `ALTER TABLESPACE RENAME DATAFILE '/path1/fichero1' TO '/path2/fichero2';`
  - Poner el tablespace online.
- El segundo requiere que la BD esté sólo montada, y es la única forma de mover el tablespace SYSTEM:
  - Parar la BD.
  - Mover el fichero desde el S.O.
  - Montar la BD.
  - `ALTER DATABASE RENAME FILE '/path1/fichero1' TO '/path2/fichero2';`
  - Abrir la BD.

# RECYCLE BIN Y DROP TABLE

- RECYCLE BIN (>=10g). Contenedor donde Oracle guarda las tablas borradas (a no ser q se borre con la opción PURGE).
- DBA\_RECYCLEBIN. Ver todas las tablas borradas. En USER\_RECYCLEBIN o RECYCLEBIN, sólo las mías. Tb con SHOW RECYCLEBIN (desde sql\*plus).
- Se puede desactivar con parámetro del init “recyclebin = off” (por defecto “on”). Tb con “ALTER SYSTEM | SESSION ...”.
- Recuperar tabla borrada:
  - FLASHBACK TABLE tablaBorrada TO BEFORE DROP;
- Borrar definitivamente:
  - PURGE TABLE tabla; (tb PURGE INDEX)

Sólo van a RECYCLEBIN los índices de las tablas borradas (no con DROP INDEX)

  - PURGE RECYCLEBIN | DBA\_RECYCLEBIN;
  - PURGE TABLESPACE nomtsp [USER usuario];
- Borrar directamente sin pasar por RECYCLEBIN:
  - DROP TABLE tabla PURGE;

# ADDM (AUTOMATIC DATABASE DIAGNOSTIC MONITOR)

- Se activa por defecto (statistics\_level = TYPICAL u ALL; se desactiva con BASIC).
- Informes. Entre dos snapshots del AWR. Su objetivo es reducir la estadística “DB time” (tiempo acumulado invertido por la BD para atender las peticiones de usuarios, ver V\$SYS\_TIME\_MODEL).
  - \$ORACLE\_HOME/rdbms/admin/addmrpt.sql. Tb usando el paquete DBMS\_ADVISOR.
- Para el análisis de E/S, por defecto, se toma un valor de 10000 microsegundos para la lectura de un bloque de la BD. Si calculamos el valor real para nuestra BD, podemos cambiarlo con:  
EXECUTE DBMS\_ADVISOR.SET\_DEFAULT\_TASK\_PARAMETER(  
                    'ADDM', 'DBIO\_EXPECTED', 8000);
- Vistas del DD:
  - DBA\_ADVISOR\_TASKS.
  - DBA\_ADVISOR\_LOG.
  - DBA\_ADVISOR\_RECOMMENDATIONS. Ordenar por RANK (importancia) y ver BENEFIT.
  - DBA\_ADVISOR\_FINDINGS.
- Más información sobre ADDM:
  - [“http://cursos.atica.um.es/oradoc102/server.102/b14211/diagnsis.htm#sthref433”](http://cursos.atica.um.es/oradoc102/server.102/b14211/diagnsis.htm#sthref433)

# AWR (AUTOMATIC WORKLOAD REPOSITORY)

- Genera y procesa estadísticas de rendimiento para que el propio servidor Oracle pueda detectar problemas y dar recomendaciones de ajuste (usando ADDM).
- Se activa por defecto (statistics\_level = TYPICAL u ALL; se desactiva con BASIC).
- Genera “snapshots” automáticamente. Tb manualmente con paquete DBMS\_WORKLOAD\_REPOSITORY.
- Informes:
  - AWR: \$ORACLE\_HOME/rdbms/admin/awrrpt.sql. Permite generar HTML.
  - ASH (V\$ACTIVE\_SESSION\_HISTORY): \$ORACLE\_HOME/rdbms/admin/ashrpt.sql.
- Vistas del DD:
  - V\$ACTIVE\_SESSION\_HISTORY (ASH, añade sesiones activas cada segundo a buffer circular en SGA)
  - V\$METRIC\*: V\$METRIC, V\$METRICGROUP, V\$METRICNAME, V\$METRIC\_HISTORY.
  - DBA\_HIST\*: DBA\_HIST\_ACTIVE\_SESS\_HISTORY, DBA\_HIST\_BASELINE, DBA\_HIST\_DATABASE\_INSTANCE, DBA\_HIST\_SNAPSHOT, DBA\_HIST\_SQL\_PLAN, DBA\_HIST\_WR\_CONTROL.
- Más información sobre AWR:
  - “<http://cursos.atica.um.es/oradoc102/server.102/b14211/autostat.htm#i27008>”

# VISTAS DEL DD

- DBA\_TABLESPACES
- DBA\_DATA\_FILES
- DBA\_TEMP\_FILES
- V\$TABLESPACE
- V\$DATAFILE
- V\$TEMPFILE
- V\$UNDOSTAT
- DBA\_TABLESPACE\_GROUPS

## 5.1. Consultar los tablespaces que componen la BD. Comprobar los ficheros que tienen cada uno de ellos.

- `Select * from dba_tablespaces;`
- `select * from v$tablespace`
- `select * from dba_data_files;`
- `select * from v$datafile;`
- `select * from dba_temp_files;`
- `select * from v$tempfile;`

# PRACTICAS TEMA 5

5.2. Crea el tablespace DATACURSOxy, NO manejado localmente, con el fichero /u02/oradata/datacursoxy01.dbf, con un tamaño de 1M. Crea 4 tablas (TABLA01, TABLA02, TABLA03, TABLA04) de 256K sobre dicho tablespace. Borra las tablas TABLA02 y TABLA04, y crea una tabla TABLA05 de 512K. ¿Qué ocurre y por qué?. Borra el tablespace DATACURSOxy y créalo de nuevo, manejado localmente. Vuelve a crear las tablas y repite el borrado de TABLA02 y TABLA04; y la creación de TABLA05 de 512K. ¿Qué ocurre esta vez y por qué?

- Create tablespace DATACURSOxy datafile '/u02/oradata/CURSOxy/datacursoxy01.dbf' size 1M extent management dictionary;
- create table TABLA01 (C1 VARCHAR2(4000)) tablespace DATACURSOxy storage (initial 256K minextents 1);
- drop table TABLA02;
- create table TABLA05 (C1 VARCHAR2(4000)) tablespace DATACURSOxy storage (initial 512K minextents 1);
- alter tablespace DATACURSOxy offline;
- drop tablespace DATACURSOxy including contents and datafiles;
- create tablespace DATACURSOxy datafile '/u02/oradata/CURSOxy/datacursoxy01.dbf' size 1M extent management local uniform size 256K;



# PRACTICAS TEMA 5

5.3. Pon el tablespace DATACURSOxy en modo READ-ONLY. Inserta una fila en alguna de sus tablas, ¿qué ocurre?. Borra la tabla TABLA01, ¿por qué se puede borrar?. Deja el tablespace DATACURSOxy en modo READ-WRITE. Repite el insert sobre TABLA01.

- alter tablespace DATACURSOxy read only;
- insert into TABLA01 values ('PRIMERA FILA');
- drop table TABLA01;
- alter tablespace DATACURSOxy read write;
- insert into TABLA01 values ('PRIMERA FILA');

5.4. Crea una tabla TABLA06 en el tablespace DATACURSOxy, ¿qué ocurre y por qué?. Activa el autoextend de su fichero, ajustando next 256K y maxsize 2M. Vuelve a crear la tabla TABLA06.

- Create table TABLA06 (C1 varchar2(4000)) tablespace DATACURSOxy storage (initial 256K minextents 1);
- alter database datafile '/u02/oradata/CURSOxy/datacursoxy01.dbf' autoextend on next 256K maxsize 2M;
- Create table TABLA06 (C1 varchar2(4000)) tablespace DATACURSOxy storage (initial 256K minextents 1);

# PRACTICAS TEMA 5

**5.5. Crea el tablespace INDCURSOxy de 1M con el fichero /u02/oradata/CURSOxy/indcursoxy01.dbf. Muévelo al directorio /u03/oradata/CURSOxy.**

- Create tablespace INDCURSOxy datafile  
‘/u02/oradata/CURSOxy/indcursoxy01.dbf’ size 1M;
- alter tablespace CURSOxy offline;
- mv /u02/oradata/CURSOxy/indcursoxy01.dbf  
/u03/oradata/CURSOxy/indcursoxy01.dbf
- alter tablespace rename datafile ‘/u02/oradata/CURSOxy/indcursoxy01.dbf’ to  
‘/u03/oradata/CURSOxy/indcursoxy01.dbf’;
- alter tablespace INDCURSOxy online;
- select \* from dba\_data\_files;

# APENDICE A.

## Recursos Oracle en Internet.

- [www.orafaq.org](http://www.orafaq.org) (Underground Oracle FAQs)
  - Sitio no oficial sobre Oracle (FAQs, foros, artículos, scripts, etc).
- [otn.oracle.com](http://otn.oracle.com) (Oracle Technology Network)
  - Descargas de sw, documentación, foros, artículos, scripts, etc. Interesante registrarse (es gratuito).
- [otn.oracle.com/oramag](http://otn.oracle.com/oramag) (Oracle Magazine)
  - Revista Oracle Magazine.
- [www.oracle.com](http://www.oracle.com)
  - Portal oficial de Oracle.
- [metalink.oracle.com](http://metalink.oracle.com)
  - Soporte técnico para usuarios con contrato de mantenimiento.

# APENDICE B.

## Novedades Oracle 9i

- undo tablespace y gestión automática de undo
- database default temporary tablespace
- drop tablespace INCLUDING CONTENTS AND DATAFILES;
- SGA dinámica: sga\_max\_size, db\_cache\_size
- db\_nk\_cache\_size (cachés con tamaño de bloque no estándar)
- desaparece “connect internal”, ahora es “connect / as sysdba”
- spfile (fichero de parámetros binario, mantenido con “alter system set ...”)
- OMF
- tablespaces con gestión automática de segmentos

# APENDICE B.

## Novedades Oracle 10g

- Tablespace SYSAUX (permite dejar en el SYSTEM sólo el DD)
- alter tablespace nombre1 RENAME TO nombre2;
- create BIGFILE tablespace tsbig1 ... size 50G; (hasta 8Eb)
- SGA\_TARGET (gestión automática del tamaño de las partes de la SGA)
- alter table t1 SHRINK SPACE CASCADE; (antes "alter table t1 enable row movement;")
- "create temporary tablespace ts1 ... TABLESPACE GROUP g1;" y "alter tablespace t2 TABLESPACE GROUP g1;.
- Data Pump (exp/imp todavía existen): expdp/impdp muy eficiente para grandes cantidades de datos
- Flashback Database (db\_recovery\_file\_dest, db\_recovery\_file\_dest\_size, db\_flashback\_retention\_size). Muy útil para auditoría: ALTER DATABASE FLASHBACK on; ALTER TABLESPACE nombre FLASHBACK ON; FLASHBACK TABLE nombre TO SCN numero; FLASHBACK TABLE nombre TO TIMESTAMP '2006-03-03 12:05:00';;
- DROP DATABASE; (sólo montada)
- ALTER DATABASE DEFAULT TABLESPACE nombre;
- ALTER SYSTEM FLUSH BUFFER\_CACHE;
- Automatic Storage Management
- ALTER SYSTEM QUIESCE RESTRICTED | UNQUIESCE (tb SUSPEND y RESUME).
- DROP TABLE nombre PURGE; (DBA\_RECYCLEBIN, recyclebin=on, SHOW RECYCLEBIN, FLASHBACK TABLE nombreTablaBorrada TO BEFORE DROP; PURGE TABLE NombreTabla;)
- ADDM. DBA\_OUTSTANDING\_ALERTS, DBA\_ALERT\_HISTORY, V\$ALERT\_TYPES. Script \$ORACLE\_HOME/rdbms/admin/addmrpt.sql.