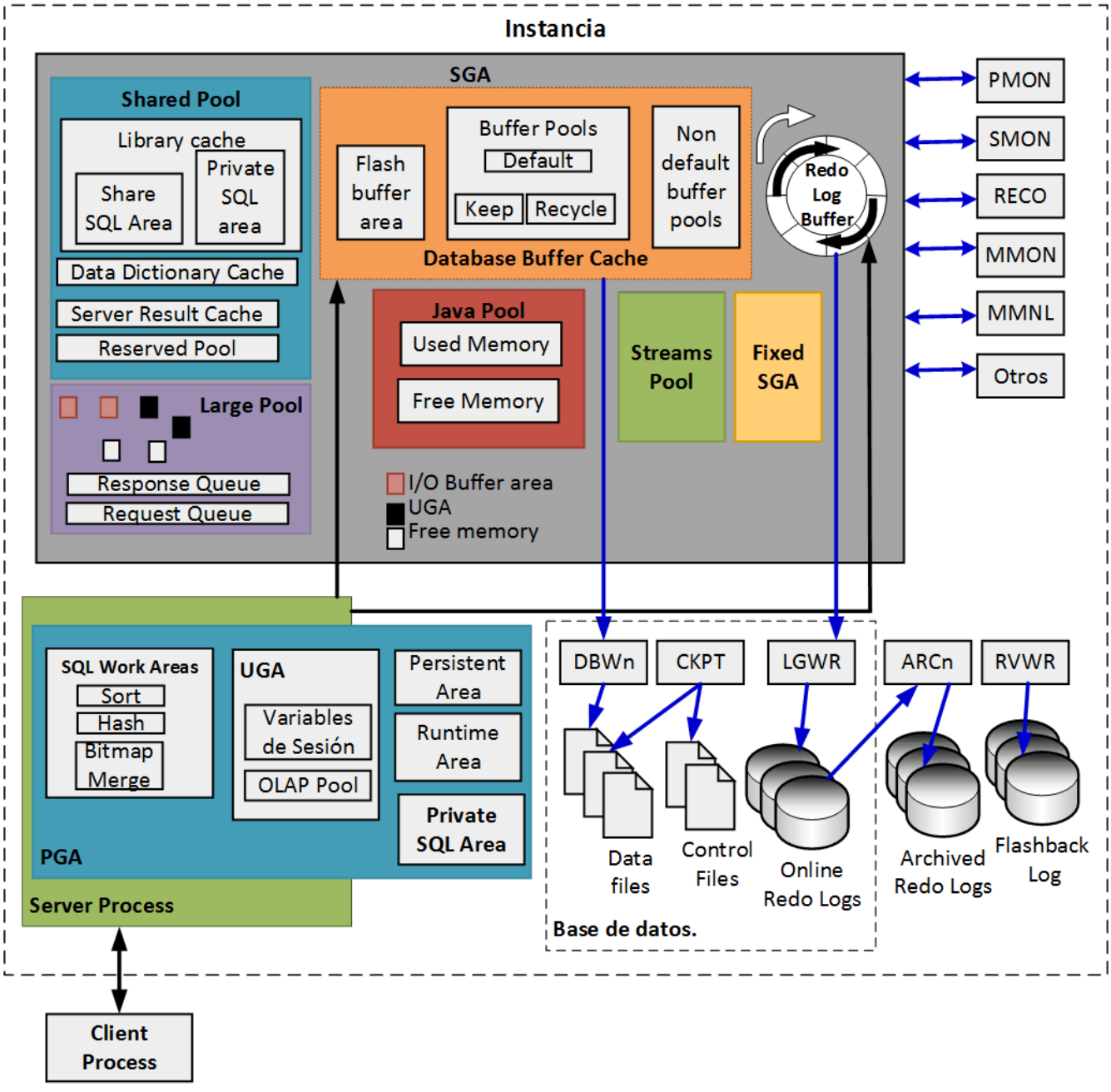
**TEMA 7 - Parte 1**
**Administración de las estructuras físicas de almacenamiento**

## 7.1. INTRODUCCIÓN.

- Las estructuras físicas de almacenamiento a diferencia de las lógicas, son visibles para el sistema operativo.
- Estas estructuras son completamente independientes a las estructuras físicas. Si se aplica un cambio, las estructuras lógicas no se afectan.

*Ejemplo:*
Si se renombra un data file, las tablas cuyos datos se guardan ahí, no serán afectadas.



- La independencia anterior permite administrarlas de forma separada.
- Como se mencionó anteriormente, una base de datos está formada por un conjunto de archivos (estructuras físicas) que son generadas cuando se crea una BD a través de la instrucción `create database`:

    - Data files y temp files
    - Control files
    - Online redo logs.

### *7.1.1. Mecanismos empleados para almacenar archivos.*

#### *7.1.1.1. Automatic Storage Management (ASM)*

Esta solución incluye la definición de un sistema de archivos diseñado exclusivamente para uso de la base de datos. Las operaciones que se realizan en estos archivos son administradas por este sistema.

#### *7.1.1.2. Sistema de archivos del sistema operativo.*

- Las operaciones que se realizan en estos archivos son administradas por el sistema operativo.
- Recordando, un sistema de archivos se construye a partir de un volumen lógico que es construido por un software llamado "Logical volume Manager" (LVM).
- Un LVM permite habilitar uno o más dispositivos de almacenamiento (discos) para ser combinados en un espacio secuencial de direcciones de tal forma que un programa o software ubicado en capas superiores lo visualice como si se tratara de un solo disco.

#### *7.1.1.3. Sistema de archivos clusterizado*

- Representado por un sistema de archivos distribuido formado por un conjunto de servidores que colaboran entre sí para implementar requerimientos como mayor rendimiento, tolerancia a fallas (si falla un nodo el sistema de archivos sigue funcionando). De forma similar, para usuarios, el sistema de archivos es visualizado como un solo disco, en este caso, compartido.

Una misma base de datos puede combinar estas 3 técnicas.

### *7.1.2. DB Managed Files Vs User Managed Files.*

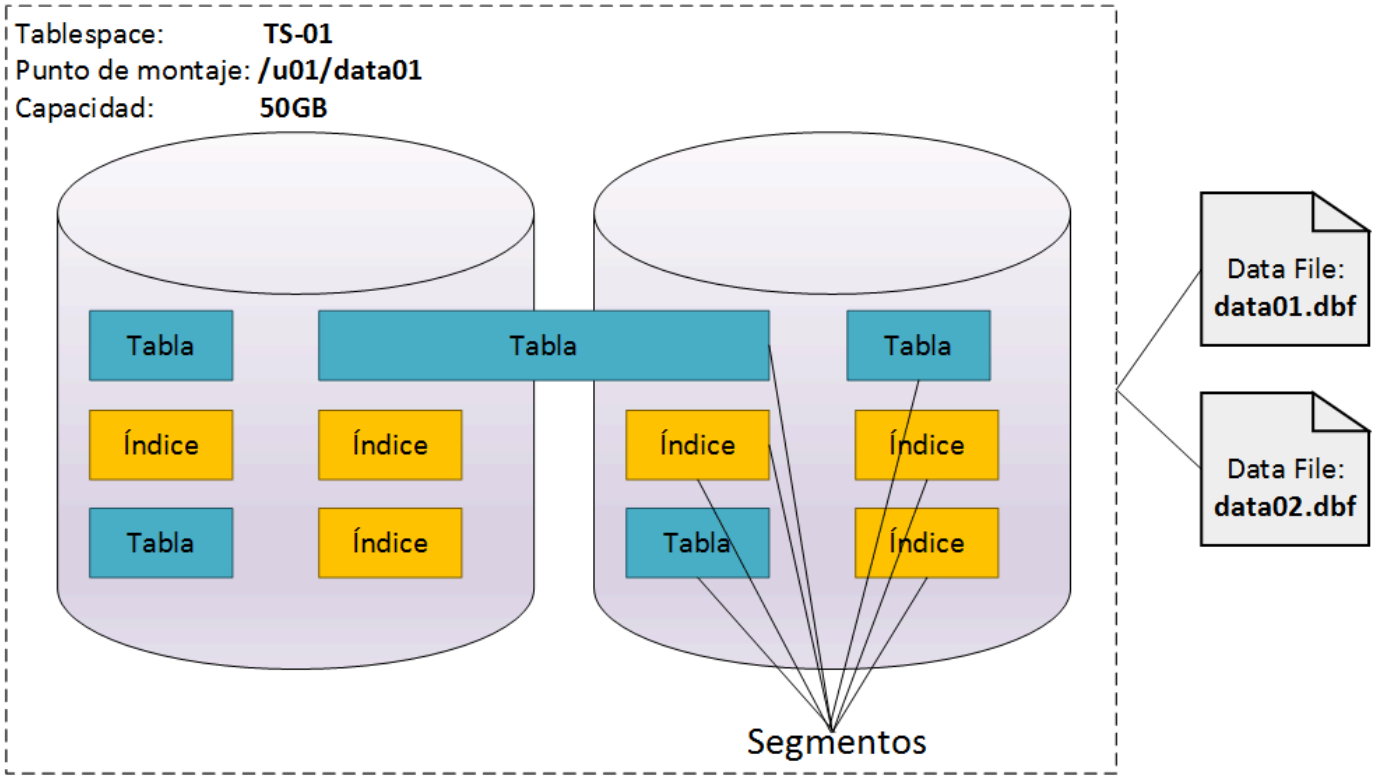
- DB (Oracle) Managed Files representa una estrategia de nombrado de archivos que permite realizar operaciones sobre ellos sin tener que especificar rutas o nombres de archivos.
- En su lugar, el nombrado se realiza en términos de objetos de la base de datos.
- Por ejemplo, es posible crear un tablespace sin especificar el nombre real y la ubicación de su data file.

```
alter system set db_create_file_dest = '$ORACLE_HOME/rdbms/dbs';
create tablespace omf_ts1;
create tablespace omf_ts2 datafile autoextend off;
```

### 7.2. DATA FILES.

A nivel del sistema operativo, los datos se almacenan en archivos con extensión `dbf` llamados data files. Cada base de datos debe contar con al menos un data file.

- Los datos de cada objeto no particionado (p.e. una tabla) son almacenados en un ***segmento.*** Cada segmento pertenece a un solo ***tablespace.***
- El concepto de *tablespace* y *data file* son similares pero con las siguientes diferencias:
  - Cada tablespace está formado por al menos un data file. Agrupa a un conjunto de data files.
  - Un segmento puede expandirse en varios data files, pero dentro de un único tablespace.
  - Toda base de datos debe contar mínimo con 2 tablespaces: `system` Y `sysaux`. Típicamente se crean de forma adicional los tablespaces `undo` y `temp`.




#### *7.2.1. Data files temporales y permanentes.*

- Data file permanente contiene datos de objetos permanentes asociados a un esquema
- Un archivo temporal también contiene datos de objetos, pero estos existen mientras la correspondiente sesión exista.
- Ambos archivos son similares excepto por las siguientes diferencias:
  - Los datos de objetos permanentes nunca son almacenados en archivos temporales.
  - Archivos temporales siempre son marcados con la propiedad `nologging`, es decir, no producen datos REDO.
  - Lo anterior implica que nunca existirá `recovery` sobre un archivo temporal.
  - Un archivo temporal no puede ser read only.
  - No es posible crear un archivo temporal con la instrucción `create database`.
  - Al crear un temp file, el espacio no es reservado en ese momento, se reserva al ser usado por primera vez (*sparce files*). Esto permite agilizar operaciones de creación y crecimiento.

- Las vistas del diccionario de datos asociadas con data files permanentes son:
  - `dba_datafiles`
  - `v$datafile`

- Las vistas del diccionario de datos asociadas con archivos temporales son
  - `dba_temp_files`
  - `v$tempfile`

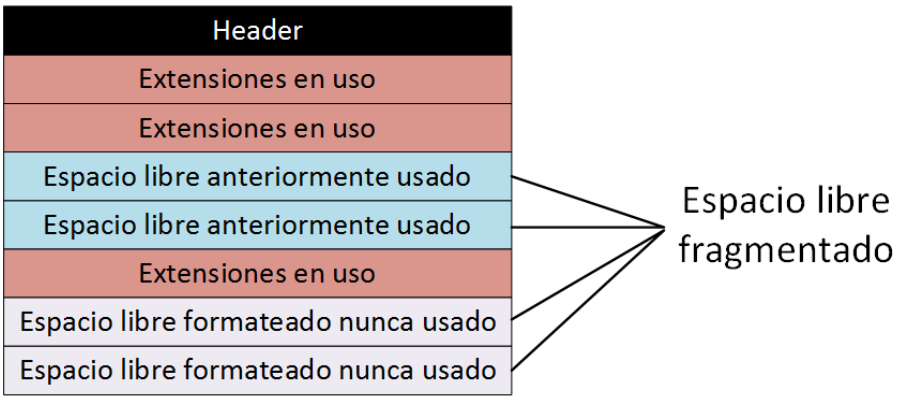
#### *7.2.2. Data files Online, Offline*

Un data file puede ser Online u Offline

- Data file Online: Se considera como un archivo disponible en que se pueden realizar lecturas y escrituras.
- Data file Offline: Se considera como un archivo no disponible. La instancia no puede hacer uso del archivo. Este estado es adecuado cuando se desea realizar algún backup tipo offline, o cuando el archivo se vuelve corrupto.
- Un tablespace también puede ser marcado como offline/Online
- Si un data file se marca como offline, el tablespace que lo contiene sigue considerando online.
- Si un tablespace se marca como offine, todos sus data files son considerados offline de forma temporal.
- La instrucción alter `database move datafile` se emplea para mover un data file de una ubicación física a otra. Útil en los siguientes escenarios:
  - Mover un tablespace a otro disco físico o a otro tipo de almacenamiento.
  - Mover data files con poca frecuencia de acceso a algún sistema de archivos que no es usado frecuentemente.
  - Marcar el tablespace como Read-Only y mover sus data files a nuevas ubicaciones.
  - Mover la base de datos a la arquitectura ASM.

#### *7.2.3. Estructura de un data file.*

- Cuando se crea un tablespace con una capacidad solicitada, se creará un nuevo data file con la capacidad solicitada y un espacio adicional para almacenar su ***header***.
- El header contiene:
  - Información del archivo y un dato muy importante llamado ***checkpoint SCN***
  - Número absoluto de data file que lo identifica de forma única a nivel de la BD.
  - Número relativo de data file que lo identifica de forma única a nivel de tablespace.
- Cuando el data file es creado, se realiza un *formateo* del espacio.
- Se reserva espacio para almacenar ***segmentos*** (datos de los objetos).
- Conforme los datos se almacenan, el espacio reservado se ocupa creando nuevas ***extensiones*** para ser asignadas a un segmento.
- Al realizar actualizaciones y eliminaciones de datos, se pueden crear espacios vacíos. Estos espacios pueden ser insuficientes para almacenar nuevos datos provocando la existencia de *espacio libre fragmentado.*




*Ejercicio 1.*

A. ¿Qué diferencia existe entre las vistas `dba_data_files` y `v$datafile`?
B. Generar una consulta que muestre los siguientes datos de los data files existentes en la base de datos. Ubicación y nombre del archivo, identificador único a nivel de la base de datos, identificador único a nivel de tablespace, nombre del tablespace al que pertenece, tamaño del archivo en MB, status del data file (indicar qué otros posibles status puede tener el archivo), indicar si tiene capacidades de auto crecimiento, el número de bloques que se reservan en caso de auto crecimiento habilitado, tamaño en MB que se destina para guardar datos del usuario, tamaño en KBs que se destina para almacenar el header del data file, mostrar el status de disponibiliad (indicar qué otros status puede adquirir).
C. Generar una sentencia SQL que muestre los siguientes datos de los data files existentes en la base de datos. Nombre de data file, identificador del archivo, número de cambio asignado cuando el data file fue creado, fecha de creación de data file, el valor del SCN (System change Number) que fue asignado en el último checkpoint realizado, fecha del último checkpoint, último número de cambio de configuración realizado al data file, fecha del último cambio realizado al data file. Las fechas de esta consulta deben mostrarse hasta nivel de segundos.
D. Realizar una inserción de un dato cuyo segmento pertenezca al tablespace users, provocar un full checkpoint manual. Finalmente, ejecutar nuevamente la sentencia del inciso anterior, Incluir el resultado e indicar las diferencias.
E. Generar una consulta SQL que muestre los siguientes datos del header de un data file. Id del archivo, Nombre del data file, indicador que permite saber si el header del data file pudo ser leído correctamente y si este es válido, bandera que indica si el data file requiere una operación de recovery, el valor del SCN (System change Number) que fue asignado en el último checkpoint realizado, fecha del último checkpoint.
F. Mostrar los siguientes datos referentes a los archivos temporales de la base de datos. Id del archivo, nombre, tablespace al que pertenece, status, bandera que indica si es autoextensible, tamaño en MB.

### *7.2.4. Aspectos importantes de configuración*

#### *7.2.4.1. Determinar el número de data files a crear.*

- El parámetro `db_files` indica el número máximo de data files que pueden crearse. Por default tiene el valor 200.
- Este valor también influye en la cantidad de espacio que se va reservar en la SGA para guardar metadatos referentes a los data files existentes en la BD.
    - Si su valor es muy grande puede traer impacto en la cantidad de memoria reservada.
    - Si su valor es muy bajo, no se podrán crear más data files de los especificados en el parámetro.
- El parámetro puede modificarse, pero requiere reinicio.
- Sistemas operativos pueden imponer límites en cuanto al número de archivos que un mismo proceso puede abrir de forma simultánea.
- Sistemas operativos pueden imponer límites en cuanto al número de archivos y a su tamaño
- El parámetro `maxdatafiles` representa el límite superior para crear data files.
- Los procesos DBWn pueden abrir todos los data files existentes y mantener en cache sus metadatos (file descriptors, etc.). Si el número de archivos es grande, y si el sistema operativo impone límites a la cantidad de archivos abiertos, la instancia cerrará los archivos provocando penalización en desempeño.
    - El límite de archivos abiertos que impone el s.o. debe ser mayor al #de data files.

*7.2.4.2. Determinar el tamaño adecuado de un data file.*

- Al crear un tablespace se debe realizar una estimación del tamaño que ocuparán, los objetos cuyos datos serán almacenados en él, y con base a ello, crear suficientes data files.
- Se recomienda crear archivos en diferentes dispositivos para realizar una distribución homogénea.
- La ubicación del tablespace se determina por la ubicación de sus data files.
- Data files no deberían almacenarse en el mismo disco donde se encuentran los Online Redo Logs.

*Ejemplo:*

- Suponer que se cuentan con varios drives, cada uno con su disco.
- En este escenario se recomienda ubicar en discos diferentes a los data files que pudieran competir entre ellos, o ser leídos al mismo tiempo, y provocar así lecturas paralelas.

*7.2.4.3. Determinar la ubicación de un data file.*

- Recordar que la ubicación de un tablespace está determinada con base a la ubicación de sus tablespaces.
- Lo más recomendable es usar los recursos físicos de almacenamiento de la mejor forma: Si se tienen diversos discos, los data files deberían estar distribuidos en estos dispositivos para incrementar paralelismo de operaciones I/O.

*7.2.4.4. Ubicación de REDO Log files vs Data files*

- Data files no deberían estar almacenados en el mismo disco respecto a los Online Redo Logs.
- Si ambos se guardan en el mismo disco y si este llega a fallar, no podría aplicarse una operación de recovery para recuperar un data file dañado.
- Si los redo log files están multiplexados, la probabilidad de perderlos baja por lo que podrían compartir disco con los data files.

*Ejercicio 2*

Considerar para este ejercicio el tablespace formado por 3 data files creado anteriormente `store_tbs_multiple`.

A. Generar una consulta que muestre el nombre del data file, su id, y el tamaño en MB de cada uno de sus data files.
B. Para comprobar los resultados anteriores, generar una consulta que muestre el nombre del tablespace `store_tbs_multiple`, el total de MB libres, y el total de bloques de datos disponibles.
C. Crear a un usuario `<nombre>_tbs_multiple` con cuota ilimitada en el tablespace `store_tbs_multiple` y asignarlo como su tablespace por default.
D. Crear una tabla llamada `<nombre>_tbs_multiple` en el esquema del usuario creado anteriormente. La tabla deberá definir una columna llamada `str` con una longitud de 1K. La creación de su segmento se debe realizar al momento de crear la tabla.
E. Generar una consulta que muestre el nombre del data file, el id del data file, el total de extensiones, total de MBs y total de bloques que se han reservado para esta tabla justo después de su creación. ¿Qué data files se usaron para inicializar su segmento?
F. Generar un programa PL/SQL que inserte un total de 512K de datos en la tabla anterior. Hacer `commit` al final.
G. Ejecutar nuevamente la consulta del punto E. ¿Qué diferencias se observan?
H. Ejecutar nuevamente el programa PL/SQL pero ahora agregar 2.5 MB de datos. Hacer `commit` al final. Ejecutar nuevamente la consulta del punto E. ¿Qué diferencias se observan?, ¿Cuántas extensiones hasta el momento se han reservado?, ¿Cuál es el espacio total reservado hasta el momento?, ¿Qué sucede con los demás data files que integran a este tablespace?
I. Ejecutar nuevamente la consulta del punto B. ¿Qué diferencias se observan?

- *Incluir sentencias SQL y resultados.*

### *7.2.5. Creación de un data file.*

Para realizar esta tarea se pueden emplear diferentes instrucciones SQL.

| | |
|---|---|
| `create tablespace`<br>`create temporary tablespace` | Se crean los data files al momento de crear un TS (vista en el tema anterior) |
| `alter tablespace . . . add datafile`<br>`alter tablespace . . . add tempfile` | La capacidad de almacenamiento de un tablespace se puede aumentar al agregar un Nuevo data file al TS (vista en el tema anterior). |
| `create database` | Los data files se crean al momento de crear la base de datos (visto en temas anteriores) |
| `alter database . . . create datafile` | Permite crear un nuevo data file vacío para ser sustituido por otro |

*Ejemplo:*

```
alter tablespace myappts1
add datafile '/u01/app/oracle/oradata/JRCBD2/myappts2.dbf' size 50m
autoextend on
next 512k
maxsize 512m;
```

### *7.2.6. Modificación del tamaño de un data file.*

Existen 2 mecanismos:

- Habilitar o deshabilitar el crecimiento automático de un data file.
- Modificar el tamaño del data file de forma manual.

#### *7.2.6.1. Habilitar y deshabilitar el crecimiento automático de un data file*

- El data file puede crecer en incrementos configurados hasta un máximo permitido.

*Ventajas:*

- Esta funcionalidad reduce la necesidad de atención inmediata cuando un data file se llena.
- Las aplicaciones no serán suspendidas cuando debido a la imposibilidad para crear nuevas extensiones.
- Se emplea la instrucción `autoextend on|off` en alguna de las siguientes instrucciones
  - `create database`
  - `alter database`
  - `create tablespace`
  - `alter tablespace`
- La columna `autoextensible` de la vista `dba_data_files` permite identificar si el data file tiene habilitada la propiedad `autoextend`.

*Ejemplo:*

```
alter database
datafile '/u01/app/oracle/oradata/JRCBD2/myappts2.dbf'
autoextend off;
```

#### *7.2.6.2. Modificar el tamaño de un data file de forma manual.*

- Se emplea la instrucción `alter database`.
- Para el caso de un tablespace tipo `bigfile` se emplea la instrucción `alter tablespace`. No es posible agregar un nuevo data file ya que este tipo de tablespace solo permite la existencia de un solo data file.

```
alter database
datafile '/u01/app/oracle/oradata/JRCBD2/myappts2.dbf'
resize 512m;
```

### *7.2.7. Modificar la disponibilidad de un data file.*

- Se refiere a la posibilidad de pasar a un data file a modo online / offiline de forma individual.
- Algunas razones para marcarlo en modo offline:
  - Backups tipo offline
  - Para renombrar, o mover su ubicación

- La BD detecta problemas para escribir en el data file. De forma automática, la BD lo marcará como offline hasta que sea reparado. El DBA puede regresarlo a modo online una vez que el error sea corregido.
- El data file se corrompe durante la operación de la BD. El DBA puede marcarlo como offline para que la BD pueda levantarse a pesar de estar dañado (siempre y cuando no sea de los tablespaces vitales, system, sysaux).

*Ejemplo:*

- Modo online

```
alter database datafile '/u02/oracle/rbdb1/stuff01.dbf' online;
```

- Modo offiline

```
alter database datafile '/u02/oracle/rbdb1/stuff01.dbf' offline;
```

Importante: Si la base de datos está en modo NOARCHIVELOG, el archivo no podrá ponerse en modo offline, básicamente para prevenir la pérdida del archivo al ponerlo offline. El modo ARCHIVELOG permite recuperar al data file en caso de pérdida ya sea por corrupción o alguna otra falla.

Las alternativas son:

- Poner al tablespace que contiene al data file en modo offline (todos los data files del tablespace y el propio tablespace serán marcados offline).

```
alter tablespace store_tbs_custom offline;
```

- Usar la cláusula `for drop`. Esto marcará al archivo como offline pero ***ya no podrá*** ser puesto online nuevamente.

```
alter database datafile '/u02/oracle/rbdb1/users03.dbf' offline for drop;
```

Posterior a estas instrucciones, lo único que restaría es eliminar el data file del tablespace, o en su defecto eliminar el tablespace empleando las siguientes cláusulas.

```
alter tablespace ... drop datafile
drop tablespace ... including contents and datafiles
```

Por lo anterior, el modo ARCHIVELOG es fundamental en bases de datos productivas (se revisará más adelante).

#### *7.2.8. Renombrar y reubicar online data files.*

- Se emplea la instrucción `alter database move datafile`
- La instrucción puede emplearse inclusive cuando la base de datos está abierta y los usuarios están haciendo uso del data file.
- Si el archivo se encuentra en modo `offline`, el comando anterior genera error. El archivo se debe encontrar en modo `online`.
  - Al realizar esta operación los datos almacenados en el control file son modificados.
  - Los archivos son físicamente renombrados y/o reubicados a la ubicación indicada.
- Algunas razones para mover los archivos son:
  - Mover un archivo a un tipo de almacenamiento diferente.

- Mover archivos que son accedidos de forma infrecuente a un tipo de almacenamiento de bajo costo.
- Hacer un tablespace read-only, mover sus archivos a un sistema de archivos "write-once".
- Mover la base de datos a una arquitectura ASM (Automatic Storage Management).

- La instrucción `reuse` puede emplearse para sobrescribir el archivo en caso de existir en la ubicación destino.
- La instrucción `keep` permite conservar el archivo en su ubicación original. La BD no hará uso del archivo original, realiza la copia del archivo y únicamente hará uso de ella al terminar de ejecutar la instrucción `alter database move datafile`.

*Ejemplo:*

```
alter database move
datafile '/u01/app/oracle/oradata/JRCBD2/myappts2.dbf'
to '/u02/app/oracle/oradata/JRCBD2/myappts22.dbf' keep;
```

#### *7.2.1. Renombrar y reubicar offline data files.*

El siguiente procedimiento ilustra el proceso para renombrar y reubicar data files offline de un tablespace.

1. Poner al tablespace en modo offline.

```
alter tablespace users offline normal;
```

2. Renombrar los data files de forma manual empleando comandos del sistema operativo
3. Utilizar `alter tablespace … rename datafile` para actualizar los nombres de los data files. Utilizar siempre las rutas absolutas.

```
alter tablespace users
  rename datafile '/u02/oracle/rbdb1/user1.dbf',
                  '/u02/oracle/rbdb1/user2.dbf'
               to '/u02/oracle/rbdb1/users01.dbf',
                  '/u02/oracle/rbdb1/users02.dbf';
```

4. En bases de datos productivas es importante realizar un full backup. Cualquier cambio físico a las estructuras de la base de datos requieren realizar un backup completo antes de continuar.
5. Poner al tablespace en modo online nuevamente.

```
alter tablespace users online;
```

#### *7.2.2. Eliminar data files.*

- Se emplea la instrucción `drop datafile` y `drop tempfile` como parte de la instrucción `alter tablespace`.
- El data file debe estar vacío. Un data file se considera vacío cuando no contiene extensiones reservadas para él. Una forma de cumplir con este punto es eliminar el tablespace que contiene al data file.
- Si el data file a eliminar es el único asociado al tablespace, este no se podrá eliminar
- La base de datos debe encontrarse en modo open.

- Al eliminar un data file, las referencias en el control file y en el diccionario de datos son eliminadas.
- El archivo es físicamente eliminado.

*Ejemplo:*

```
alter tablespace example drop datafile '/u02/oracle/rbdb1/user1.dbf ';
```

*Ejercicio 3.*

Considerar nuevamente el tablespace y la tabla creada en el ejercicio anterior. Suponer que ahora se decide realizar algunas tareas administrativas.

A. Como primer paso el DBA ha decidido poner a los data files del tablespace en modo offline para iniciar con las tareas administrativas. ¿Qué sucede intentar poner al data file `store_tbs_multiple_01.dbf` en modo offline? Asumir que la instancia no puede detenerse para no afectar a otros tablespaces. Escribir la sentencia SQL correspondiente.
B. Para evitar que los usuarios accedan a los data files del tablespace durante las labores administrativas, ¿Cuál será la forma correcta de proceder?, Indicar la sentencia SQL correspondiente y ejecutarla.
C. Para comprobar lo anterior, ¿Qué sucede si el usuario creado anterior intenta hacer un conteo de los registros de la tabla `long_strings` ?
D. Generar una consulta que muestre el nombre del data file, su id, y la columna correspondiente para confirmar que los data files están offline.
E. Suponer que la tarea de mantenimiento consiste en reasignar las ubicaciones y nombres de los data files con base a la siguiente tabla:

| Ubicación y nombre anterior | Nueva ubicación |
|---|---|
| /u01/app/oracle/oradata/JRCBDA2/store_tbs_multiple_01.dbf | /u0**3**/app/oracle/oradata/JRCBDA2/store_tbs_multiple_01**3**.dbf |
| /u02/app/oracle/oradata/JRCBDA2/store_tbs_multiple_02.dbf | /u02/app/oracle/oradata/JRCBDA2/store_tbs_multiple_02**3**.dbf |
| /u03/app/oracle/oradata/JRCBDA2/store_tbs_multiple_03.dbf | /u0**1**/app/oracle/oradata/JRCBDA2/store_tbs_multiple_03**1**.dbf |

Aplicar las acciones necesarias para realizar los cambios solicitados. Al concluir poner nuevamente el tablespace online y realizar el conteo de los registros de la tabla para verificar que todo funcione correctamente.

F. Posteriormente el DBA se percató que el ejercicio anterior se pudo haber realizado sin haber usado el modo offline. Aplicar las acciones necesarias ahora para regresar a los data files a su ubicación original, es decir:

| Ubicación anterior | Ubicación nueva |
|---|---|
| /u0**3**/app/oracle/oradata/JRCBDA2/store_tbs_multiple_01**3**.dbf | /u01/app/oracle/oradata/JRCBDA2/store_tbs_multiple_01.dbf |
| /u02/app/oracle/oradata/JRCBDA2/store_tbs_multiple_02**3**.dbf | /u02/app/oracle/oradata/JRCBDA2/store_tbs_multiple_02.dbf |
| /u0**1**/app/oracle/oradata/JRCBDA2/store_tbs_multiple_03**1**.dbf | /u03/app/oracle/oradata/JRCBDA2/store_tbs_multiple_03.dbf |

Comprobar a nivel de sistema operativo que las nuevas ubicaciones se actualizaron correctamente.

G. Considerar el tablespace `store_tbs1`. Asumiendo que la base de datos está en modo open, empleando comandos del sistema operativo eliminar el data file `store_tbs01.dbf`. Contestar

las siguientes preguntas una vez que se ha realizado la eliminación. Se pueden ejecutar las instrucciones correspondientes para comprobar resultados.

- ¿Qué le pasa a la instancia una vez que se ha eliminado el data file?
- Suponer que el DBA se percata del problema y decide poner offline al otro data file que integra al TS. ¿Qué sucederá?
- Suponer que el DBA intenta marcar al tablespace en modo offiline, ¿Qué sucederá?
- ¿Qué sucede si el DBA intenta detener la instancia?
- ¿Qué sucede si el DBA intenta iniciar la instancia con el data file perdido?
- El DBA se percata que no se cuenta con respaldo físico del data file. Sin embargo cuenta con scripts SQL para recuperar los datos. Aplicar las sentencias SQL necesarias para que el tablespace vuelva a estar en modo online y poder así iniciar la ejecución de las sentencias SQL.
- *Incluir sentencias SQL y resultados.*

### 7.3. CONTROL FILES.

- Cada base de datos cuenta con un archivo de control.
- Es un archivo pequeño y binario asociado a una sola base de datos. Múltiples copias son permitidas para evitar pérdidas.
- Su objetivo principal es almacenar la configuración de la estructura física de una base de datos, indica la ubicación de los archivos; indica el estado de la base de datos.
- Contiene metadatos que deben ser accedidos cuando la base de datos no está abierta.
- El archivo incluye la siguiente información:
    - Nombre de la BD.
    - Nombres y ubicaciones de los data files
    - Fecha exacta de creación de la BD.
    - El valor actual del SCN (System Change Number)
    - Información acerca de un checkpoint.
    - Información de los data files, online Redo Log, Archived Redo Log files.
    - Información de RMAN
    - Información de tablespaces.

*Importante*:

- El archivo debe estar disponible y con permisos de escritura al momento de montar una BD.
- Sin este archivo la BD no podrá ser montada.
- La recuperación o restauración del archivo de control *es muy complicada*.
- Se requiere al menos una copia de este archivo la cual es creada al momento de crear la base de datos.
- Por lo anterior, lo más adecuado es crear al menos 2 copias, cada uno almacenado en discos diferentes.
- Una forma de multiplexar estos archivos, es agregando una archivo de control en cada disco donde se encuentre un grupo de archivos Redo Log.
- Si alguna de las copias se daña o no está disponible, la BD se vuelve inoperable y se deberá abortar.

El parámetro `control_files` muestra la lista de archivos de control de la base de datos:

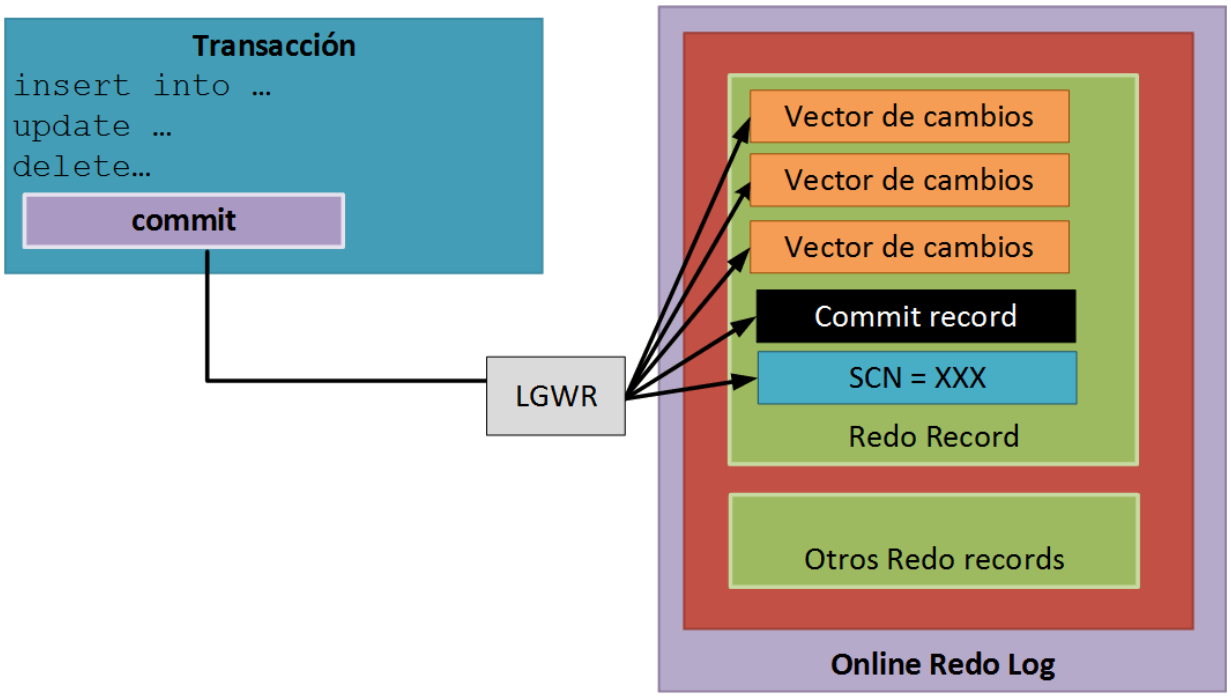
```
show parameters control_files
```

```
NAME            TYPE    VALUE
--------------- ------- ------------------------------------------
control_files   string  /u01/app/oracle/oradata/JRCBD2/control01.ctl,
                        /u01/app/oracle/oradata/JRCBD2/control02.ctl
```

### *7.3.1. Importancia de un control file para realizar recovery.*

Para entender adecuadamente esta sección, se revisan/repasan los siguientes conceptos:

- Recordando, el proceso de background LGWR (Log Writer) escribe en los Online Redo Logs cuando ocurre alguno de los siguientes eventos:
  - Cuando un user process hace commit de una transacción.
  - Cuando ocurre un log switch
  - Cuando el Redo Buffer alcanza 1/3 de su capacidad o contiene 1MB de datos en el buffer.
  - Han pasado 3 segundos desde la última escritura a los Online Redo Logs.
  - Justo antes de que e DBW (DB writer) sincronice los datos del db buffer cache con data files (Write-ahead protocol). DBW ***debe*** esperar a que LGWR termine de escribir a los Redo Logs para poder sincronizar con data files.
- ¿Qué escribe LGWR en el Online Redo Log cuando un usuario hace commit?
  - LGWR escribe varias entradas al Online Redo Log llamados ***Vectores de cambios*** que describen los cambios realizados por la transacción (su contenido se estudia más adelante).
  - LGWR agrega un ***commit record*** al Online redo Log que sirve para indicar que los cambios contenidos en los Online Redo Logs fueron confirmados por el usuario.
  - Al hacer commit, se le asigna un SCN (System Change Number) a la transacción. Este número es también almacenado en los Online Redo Logs.
- ¿Qué escribe LGWR en el Online Redo Log cuando se produce un evento diferente a un commit?
  - No perder de vista que LGWR puede escribir cambios de transacciones que aún no hacen commit. Por lo tanto, se almacenan los vectores de cambios pero no el ***commit record*** ni el ***SCN.***



- Recordando el concepto de checkpoint:
    - Estructura de datos que indica la posición dentro del Redo Log a partir del cual debe iniciar una posible recuperación de datos en caso de falla en la instancia.
    - Visto de otra forma, esta posición apunta directamente a un SCN contenido en el Redo Log.
    - Todos los SCNs anteriores a esta marca corresponden a cambios que ya fueron sincronizados con los data files, y por lo tanto ya no requieren recuperación en caso de falla, pero, los SCNs que estén por debajo de esta posición, si lo requieren.
    - El proceso de background ckpt (Checkpoint process) es el que se encarga de disparar la ejecución de un checkpoint:
        - A partir de la última posición (o desde el inicio del Redo Log en caso de no existir) determina al conjunto de N Redo Records que serán sincronizados con data files.
        - La marca de checkpoint se mueve N posiciones hacia abajo dentro del Redo Log para indicar que este nuevo conjunto ya fue sincronizado en data files.
        - Registra esta nueva posición en el ***control file*** y en los ***headers de los data files.***
        - Ordena a DBW que sincronice este conjunto de vectores en data files.
- La importancia de almacenar el SCN tanto en el control file como en los headers de los data files, es que le permite a la instancia conocer que todos los SCNs previos a este nuevo valor almacenado han sido ya sincronizados con data files y por lo tanto ya no requieren recovery.
- Cuando ocurre una falla, se lee el valor de este SCN y se ubica dentro del Redo Log. Todos los cambios encontrados a partir de esta posición requieren recovery.

### *7.3.2. Estructura de un Control File.*

- La información de la base de datos es almacenada en diferentes secciones del control file.
- Cada sección contiene un junto de registros que describe un aspecto específico de la BD.

#### *7.3.2.1. Tipos de registros*

El archivo de control contiene 2 tipos de registros:

- *Registros circulares*: Registros reutilizables que pueden ser sobrescritos que contienen información no crítica y lo tanto puede ser sobrescrita. En este tipo de registros se almacena:
    - Datos referentes a los Archive Redo Log Files
    - Datos acerca de RMAN Backups.
- *Registros no circularles*: Contiene información crítica que no sufre cambios y por lo tanto no debe ser sobrescrita. En este tipo de registros se almacena:
    - Información de los Data files, tablespaces, Online Redo Log.

### *7.3.3. Respaldar un Control file.*

Cada vez que se realice alguna de las siguientes actividades, el archivo de control debería ser respaldado:

- Agregar, modificar o eliminar data files.
- Agregar o eliminar tablespaces, o modificar su estado de lectura/escritura.
- Agregar o eliminar archivos Online Redo Log o grupos de archivos.

Para respaldar un archivo de control se puede emplear alguna de las siguientes estrategias:

- Crear una copia (archivo binario) del control file:

```
alter database backup controlfile to
'/home/oracle/backups/controlfile.bkp';
```

- Crear un script SQL que al ser ejecutado permita la reconstrucción del archivo de control. El script hace uso de la instrucción `create controlfile`

```
alter database backup controlfile to trace as
'/home/oracle/backups/controlfile.bkp.sql';
```

### *7.3.4. Agregar o eliminar una copia del archivo de control.*

*Ejercicio 4:*

A. Respaldar un archivo de control.

- Crear la carpeta `/home/oracle/backups` en caso de no existir
- Considerando que la BD está abierta, ejecutar las 2 instrucciones anteriores para crear un backup del archivo de control, archivo binario y archivo SQL.

B. Realizar las siguientes acciones para eliminar y restaurar una nueva copia a la lista de archivos de control. Hasta ahora, la instancia tiene 3 copias del archivo de control ubicadas en `/u01`, `/u02` y `/u03`. En este ejercicio primero se eliminará la copia de `/u03` y posteriormente se recuperará.

1. Detener la instancia para evitar que los archivos actuales se modifiquen mientras se realiza la copia.
2. A nivel de sistema operativo eliminar la copia del archivo de control que se ubica en `/u03`
3. Intentar levantar la instancia sin este archivo ¿Qué sucede?
4. Iniciar la instancia en modo `nomount`, crear una nueva copia del `pfile` a partir del `spfile`, detener nuevamente la instancia.
5. Editar el `pfile` y modificar el parámetro `control_files` para eliminar la tercera copia.

```
*.control_files='/u01/app/oracle/oradata/JRCBDA2/control01.ctl','/u02/app/o
racle/oradata/JRCBDA2/control02.ctl'
```

6. Iniciar la instancia en modo `nomount` empleando el `pfile`
7. Crear un nuevo `spfile` a partir del `pfile`. Esto con la finalidad de guardar el nuevo valor del parámetro `control_files`.
8. Comprobar el cambio con el comando `show parameters control_files`
9. Deterner la instancia
10. Iniciar nuevamente de forma normal comprobar el valor del parámetro. Incluir la salida
11. Realizar el procedimiento inverso, pero ahora para restaurar la tercera copia. Ahora el archivo de control se debe llamar `control33.ctl`. Ojo: Este archivo deberá ser creado a partir de una copia de cualquiera de los otros 2. Dicha copia se deberá realizar una vez que se haya detenido la instancia. Al final del proceso mostrar nuevamente la salida.

- *Incluir las sentencias SQL, instrucciones y resultado de las operaciones.*

### *7.3.5. Crear un nuevo control file.*

En algunas situaciones puede requerirse la creación de un control file completamente desde cero. Algunas causas pueden ser:

- Cuando todas las copias del archivo de control se han perdido, todos los archivos están corruptos o no disponibles.
- No se cuenta con respaldos de los archivos de control.
- Se desea cambiar el nombre a la base de datos.

El procedimiento para realizar la creación se describe a continuación. Se asume que no existen copias del archivo de control y tampoco respaldos.

*Ejercicio 5:*

Para comprobar este procedimiento, realizar las siguientes acciones antes de iniciar con la creación del archivo:

1. Iniciar la instancia en modo open en caso de no estar abierta.
2. Forzar un full checkpoint manual.
3. Considerar la vista `v$database`. Recordando, esta vista muestra información de la base de datos contenida en el archivo de control. Generar una consulta que muestre: el nombre de la BD, el SCN obtenido al hacer reset de los Online redo lods `resetlogs_change#`, fecha en la que se realizó un reset de logs `resetlogs_time`, SCN que se tenía antes de realizar un reset de logs. `prior_resetlogs_change#`, la fecha de un reset de logs anterior `prior_resetlogs_time`, el último SCN que se sincronizó con data files durante un checkpoint, `checkpoint_change#`, la fecha de creación del control file `controlfile_created`, el valor de SCN actual `current_scn` Mostrar las fechas hasta nivel de segundos.
4. Conectarse con cualquier usuario, crear una tabla `<iniciales>_numeros` en caso de no existir con la siguiente estructura:

```
create table <iniciales>_nombres(
  id number(10,0),
  nombre varchar2(100)
);
```

5. Insertar 3 registros, hacer `commit`.
6. Mostrar el SCN y la fecha de creación asignados a la transacción anterior así como los valores insertados. Tomar la siguiente consulta como ejemplo.

```
select ora_rowscn, scn_to_timestamp(ora_rowscn) ora_rowscn_ts,id,nombre
from jorge.jrc_nombres;
```

| | ORA_ROWSCN | ORA_ROWSCN_TS | ID | NOMBRE |
|---|---|---|---|---|
| 1 | 2064191 | 05-MAY-20 04.53.47.000000000 PM | 1 | Juan |
| 2 | 2064191 | 05-MAY-20 04.53.47.000000000 PM | 2 | Euduardo |
| 3 | 2064191 | 05-MAY-20 04.53.47.000000000 PM | 3 | Maribel |

7. Insertar 3 nuevos registros, hacer `commit`, repetir nuevamente la consulta anterior e incluir el resultado. ¿Qué se observa en los valores del SCN?
8. Mover los archivos de control de la base de datos para simular su pérdida, por ejemplo, moverlos a `/home/oracle/backups`.
9. Conectarse como usuario SYS, e intentar detener la instancia con `shutdown immediate`. Debido a que los archivos de control se eliminaron, se obtendrá un error. Incluirlo.
10. En este caso la instancia se tendrá que detener con la opción `shutdown abort`. Detenerla.

Una vez que se ha provocado el error, el archivo de control será regenerado. Notar que al hacer `shutdown abort`, se va a requerir realizar la recuperación de la base de datos. El registro que se insertó en pasos anteriores seguramente no está sincronizado con los data files, y por lo tanto la BD debe recuperarse. Realizar las siguientes acciones:

11. Realizar una lista de todos los data files y de todos los archivos Redo Logs que integran a la base de datos.

- Si la instancia está iniciada, es posible realizar las siguientes consultas para determinar esta lista:

```
select member from v$logfile;
select name from  v$datafile;
select value from v$parameter where name = 'control_files';
```

- En caso de no poder levantar la instancia, tratar de ubicar manualmente los archivos que pertenecen a la base de datos, por ejemplo, realizando búsquedas en el sistema de archivos. Para efectos del curso, los archivos de esta BD se encuentran dentro de los directorios /u01, /u02 y /u03. Incluir la lista de data files y redo log files encontrados.

12. Realizar un respaldo de todos los data files y Online Redo Logs de la BD (para efectos del curso no es necesario realizar este paso).
13. Iniciar la instancia en modo `nomount`.
14. Crear y ejecutar un script SQL con la instrucción `create controlfile`. Emplear como base el ejemplo que se muestra a continuación, completarlo con la lista de archivos correspondientes.

- Los valores de los parámetros MAX* se obtienen del script que se empleó para crear la BD, o simplemente se pueden redefinir.
- Hacer uso de la cláusula `resetlogs` en caso de haber perdido grupos de Online Redo Logs, o en caso de requerir realizar una operación de recuperación. Al especificar esta cláusula, se deberá especificar posterior a la ejecución de la sentencia: `recover database`.

```
create controlfile reuse database jrcbd2 resetlogs  noarchivelog
    maxlogfiles 16
    maxlogmembers 3
    maxdatafiles 1024
    maxinstances 1
    maxloghistory 292
logfile
  group 1 (
    '/u01/app/oracle/oradata/JRCBD2/redo01a.log',
    --completar
  ) size 100m blocksize 512,
  group 2 (
    '/u01/app/oracle/oradata/JRCBD2/redo02a.log',
    --completar
  ) size 100m blocksize 512,
  group 3 (
    '/u01/app/oracle/oradata/JRCBD2/redo03a.log',
    --completar
  ) size 100m blocksize 512
datafile
  '/u01/app/oracle/oradata/JRCBD2/system01.dbf',
  --completar
character set al32utf8
;
```

***Ojo: Modificar las rutas con base al ambiente correspondiente y agregar las rutas de los archivos que sean necesarias.***

15. Al ejecutar la instrucción puede mostrarse el siguiente mensaje:

```
ERROR at line 1:
ORA-01503: CREATE CONTROLFILE failed
ORA-01210: data file header is media corrupt
ORA-01110: data file : '/u01/app/oracle/oradata/JRCBDA2/temp01.dbf'
ORA-01160: file is not a data file
ORA-01110: data file : '/u01/app/oracle/oradata/JRCBDA2/temp01.dbf'
```

Este error hace referencia al data file que se está empleando en el tablespace temporal de la base de datos. Debido a su naturaleza, este data file no requiere de recovery, y por lo tanto puede regenerarse sin mayor dificultad. Para corregir el error, excluir al data file de la lista. Al final del ejercicio se realizarán las acciones para recuperarlo.

16. Reintentar la instrucción, la creación de los archivos de control debe ser exitosa. Comprobar a nivel de sistema operativo que los 3 archivos fueron generados.
17. Debido a que se especificó **resetlogs** y debido a que la instancia no se detuvo de forma normal, se debe ejecutar la siguiente instrucción:

```
recover database using backup controlfile
```

Al ejecutarla se obtiene un error similar al siguiente:

```
sys@jrcbda2> recover database using backup controlfile
ORA-00279: change 2063078 generated at 05/05/2020 16:51:08 needed for thread 1
ORA-00289: suggestion :
/u01/app/oracle/product/18.0.0/dbhome_1/dbs/arch1_4_1039543022.dbf
ORA-00280: change 2063078 for thread 1 is in sequence #4
```

El significado de este mensaje es el siguiente:

A) Se está intentando realizar el proceso de recuperación. Para ello es necesario especificar la ruta absoluta del archivo log que contiene los cambios a recuperar.
B) Observar que esta base de datos contiene 3 grupos de Online Redo Log Files. Esto significa que uno de los 3 grupos se estaba utilizando al momento de detener la instancia. En este momento, no se sabe con certeza cuál de los 3.
C) Para resolver lo anterior, es posible ejecutar la instrucción y probar con cualquiera de los Online Redo Logs de cada grupo. Por ejemplo, al capturar la ruta de un Online Redo Log del primer grupo se tiene lo siguiente:

```
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
/u01/app/oracle/oradata/JRCBD2/redo01a.log
ORA-00310: archived log contains sequence 31; sequence 33 required
ORA-00334: archived log: '/u01/app/oracle/oradata/JRCBD2/redo01a.log'
```

El mensaje de error indica una diferencia de secuencia. Se esperaba la secuencia 33 y se obtuvo la 31. Esto significa que este grupo de archivos no es el que se estaba empleando. Por lo tanto, se intenta con el archivo siguiente, el cual pertenece al grupo 3.

```
Specify log: {<RET>=suggested | filename | AUTO | CANCEL}
/u01/app/oracle/oradata/JRCBD2/redo03a.log
Log applied.
Media recovery complete.
```

Notar que en este caso la recuperación fue exitosa.

Hay un último escenario que puede presentare y a pesar de especificar todos los posibles Online Redo Logs, la recuperación puede fallar.

- Recordando de ejercicios anteriores, el tablespace `store_tbs_custom` fue puesto en modo offline. Al perder el archivo de control, la BD intentará realizar la recuperación del data file correspondiente `store_tbs_custom_01.dbf`.
- Al leer el SCN en el header de dicho data file, la BD intentará aplicar Redo Logs sobre este archivo. Debido a que el tablespace ha estado cierto tiempo fuera le línea, estos SCNs seguramente ya no serán encontrados en los Online Redo logs.
- Para solucionar el problema, este data file se puede remover de la lista. Al haberse puesto en modo offline de forma normal, el data file podrá recuperarse una vez que la BD sea abierta. Por lo tanto, si este error ocurre eliminar al data file de la lista y reintentar.

18. Abrir la base de datos e indicar que se haga reset de los Online Redo Logs.

```
alter database open resetlogs
```

19. El siguiente paso una vez que la base de datos fue abierta es recuperar o restaurar las 2 data files pendientes: El data file temporal y el data file `store_tbs_custom_01.dbf`. Para el caso del segundo, al intentar poner a su correspondiente tablespace online se producirá el siguiente error:

```
sys@jrcbda2> alter tablespace STORE_TBS_CUSTOM online;
alter tablespace STORE_TBS_CUSTOM online
*
ERROR at line 1:
ORA-01157: cannot identify/lock data file 12 - see DBWR trace file
ORA-01111: name for data file 12 is unknown - rename to correct file
ORA-01110: data file 12: '/u01/app/oracle/product/18.0.0/dbhome_1/dbs/MISSING00012'
```

- Observar que al no existir información en el nuevo control file de este data file, la bd emplea una ruta inexistente.
- Derivado a que este TS estaba offline, no requiere recovery. Por otro lado, al hacer reset de los logs no se hará recovery.
- Para corregir el error se deberá renombrar el data file por su nombre real.

```
alter tablespace store_tbs_custom
  rename datafile
'/u01/app/oracle/product/18.0.0/dbhome_1/dbs/MISSING00012'
  to '/u01/app/oracle/oradata/JRCBDA2/store_tbs_custom_01.dbf';
```

- Posterior a esta instrucción, volver a intentar poner online al tablespace. Ejecutar la siguiente sentencia e incluir los resultados para comprobar que todos los data files son correctos.

```
select file_name,file_id,tablespace_name,status,online_status
from dba_data_files;
```

20. Para el data file temporal al consultar los archivos temporales que existen en la base de datos, se observa que no se obtienen registros:

```
select * from v$tempfile;
select * from dba_temp_files;
```

Para corregirlo anterior, se agrega nuevamente el data file al tablespace temporal.

```
alter tablespace tempts1
  add tempfile '/u01/app/oracle/oradata/JRCBDA2/temp01.dbf'
  size 20m reuse autoextend on next 640k maxsize unlimited;
```

Ejecutar nuevamente la siguiente sentencia para comprobar e incluir los resultados.

```
select * from v$tempfile;
```

| | FILE# | CREATION_CHANGE# | CREATION_TIME | TS# | RFILE# | STATUS | ENABLED | BYTES | BLOCKS | CREATE_BYTES | BLOCK_SIZE | NAME |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2066369 | 05-MAY-20 | 3 | 1 | ONLINE | READ WRITE | 20971520 | 2560 | 20971520 | 8192 | /u01/app/oracle/oradata/JRCBDA2/temp01.dbf |

- *Incluir en este ejercicio totas las sentencias SQL y sus respectivos resultados.*

#### *7.3.6. Vistas del diccionario de datos asociadas con un Control file.*

```
v$database
v$controlfile
v$controlfile_record_section
```

*Ejercicio 6:*

Explorar revisar las vistas anteriores, indicar la diferencia entre ellas, y generar consultas con los datos que contienen.