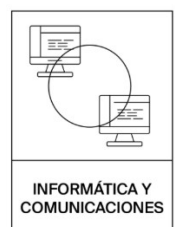


Administración de Sistemas Gestores de Bases de Datos



Configuración DE UN SGBD ORACLE



UD 2 . Configuración de un SGBD Oracle

Resumen del contenido:

1. Configuración del Entorno en Oracle

- Variables de entorno en Linux y Windows: ORACLE_HOME, ORACLE_SID, ORACLE_BASE, PATH, NLS_LANG.
- Comandos para ver la configuración (env, printenv, echo en Linux y set, echo en Windows).

2. Configuración de las Conexiones

- Necesidad de una IP accesible.
- Uso de VirtualBox con red de solo anfitrión o interna.
- Configuración de firewall para permitir el puerto 1521.
- Configuración de ficheros de conexión (listener.ora, sqlnet.ora, tnsnames.ora).

3. Primera Conexión a la Base de Datos

- Uso de sqlplus / as sysdba.
- Comandos básicos: show con_name, show user, show pdbs, show sga.

4. Administración de la Instancia Oracle

- Archivos SPFILE y su configuración (ALTER SYSTEM SET parametro=valor).
- Configuración de la instancia en versiones modernas de Oracle (21c).

5. Gestión de Conexiones y Usuarios

- Cuentas de administración (SYS, SYSTEM, PDBADMIN).
- Conexión a una base de datos específica (sqlplus sys@localhost/pdb1 as sysdba).

6. Comandos en SQL*Plus

- edit, help, list, save, run, shutdown, startup, etc.

7. Estados del Servidor Oracle

- Diferentes estados: shutdown, nomount, mount, open.
- Cómo cambiar entre estados (STARTUP, ALTER DATABASE OPEN).

8. Gestión de Tablespaces y Datafiles

- Creación de tablespaces (CREATE TABLESPACE), modificación (ALTER TABLESPACE), eliminación (DROP TABLESPACE).
- Permisos requeridos para gestionar tablespaces.

9. El Diccionario de Datos en Oracle

- Consultas útiles (`SELECT table_name FROM user_tables;`, `describe dba_tablespace`).
- Uso de vistas dinámicas (`V$INSTANCE`, `V$DATABASE`).

10. Gestión del Redo Log y Archive Log

- Funcionamiento del redo log (registro de cambios para recuperación en caso de fallo).
- Activación del modo ARCHIVELOG.

11. Gestión de Logs en Oracle

- Logs de alertas, procesos en background y logs de usuarios.
 - Comandos para visualizar logs (`SELECT * FROM V$LOGFILE;`).
-

Punto 1. Configuración del Entorno en Oracle

La configuración del entorno en Oracle es fundamental para que el sistema funcione correctamente. Se centra en definir y verificar las variables de entorno necesarias tanto en **Linux** como en **Windows**.

1.1. Variables de entorno principales en Oracle

Antes de iniciar una base de datos Oracle, es necesario configurar ciertas variables del entorno del sistema operativo. Estas variables ayudan a Oracle a encontrar los archivos, configuraciones y rutas necesarias para su funcionamiento.

Principales variables de entorno:

1. ORACLE_HOME

- Define el directorio donde está instalado Oracle Database.
- Ejemplo: `/u01/app/oracle/product/21.0.0/dbhome_1`

2. ORACLE_SID

- Identificador único de la instancia de Oracle.
- Ejemplo: `orcl`

3. ORACLE_BASE

- Directorio base donde se guardan archivos de Oracle.
- Ejemplo: `/u01/app/oracle`

4. PATH

- Contiene las rutas donde el sistema operativo busca los ejecutables de Oracle.

- Se debe incluir \$ORACLE_HOME/bin en Linux o %ORACLE_HOME%\bin en Windows.

5. NLS_LANG

- Configura el idioma, juego de caracteres y tipo de territorio de Oracle.
 - Ejemplo: AMERICAN_AMERICA.UTF8
-

1.2. Cómo configurar las variables en Linux

Para ver el valor actual de una variable, se pueden usar estos comandos en la terminal:

```
echo $ORACLE_SID
echo $ORACLE_HOME
```

Si necesitas cambiar el valor de una variable temporalmente, puedes usar el siguiente comando:

```
export ORACLE_SID=orcl
```

Para que estos valores sean permanentes, se deben añadir al archivo `.bashrc` o `.bash_profile`:

```
echo "export ORACLE_HOME=/u01/app/oracle/product/21.0.0/dbhome_1" >> ~/.bashrc
echo "export ORACLE_SID=orcl" >> ~/.bashrc
source ~/.bashrc
```

1.3. Cómo configurar las variables en Windows

Para ver la configuración actual en Windows, se pueden ejecutar los siguientes comandos en la línea de comandos (cmd):

```
set
echo %ORACLE_HOME%
echo %ORACLE_SID%
```

Para cambiar una variable temporalmente en una sesión de cmd:

```
set ORACLE_SID=orcl
```

Para hacer los cambios permanentes, hay que editar las variables de entorno del sistema:

1. Ir a **Panel de control** → **Sistema** → **Configuración avanzada del sistema**.
 2. Clic en **Variables de entorno**.
 3. Añadir o modificar:
 - ORACLE_HOME = C:\oracle\product\21.0.0\dbhome_1
 - ORACLE_SID = orcl
 - Agregar %ORACLE_HOME%\bin en la variable PATH.
-

1.4. Comprobación de la Configuración del Entorno

Una vez configuradas las variables, es recomendable verificarlas.

En **Linux**:

```
env | grep ORACLE  
printenv | grep ORACLE
```

En **Windows**:

```
set ORACLE
```

Conclusión

Conocer y configurar correctamente estas variables es clave para que Oracle Database funcione bien. En Linux, se gestiona de manera diferente a la utilizada en Windows

Punto 2. Configuración de las conexiones en Oracle

Para que los clientes puedan conectarse a una base de datos Oracle, es necesario configurar correctamente la red y los parámetros de conexión. En este punto del documento, se explican los requisitos y configuraciones necesarias tanto en Windows como en Linux.

2.1. Requisitos básicos para la conexión

Antes de configurar las conexiones, se debe asegurar que la máquina donde está el servidor Oracle: Tiene una **IP accesible** (fija o dentro de la red local).

No está utilizando **APIPA (Automatic Private IP Addressing)** en Windows, ya que este sistema asigna direcciones IP privadas automáticamente y puede impedir la conexión externa. Tiene una configuración de red adecuada si se usa **VirtualBox** para máquinas virtuales.

Configuración del Firewall

Para que los clientes se conecten a Oracle, se debe permitir la comunicación en el **puerto 1521**, que es el puerto por defecto del **listener de Oracle**.

En **Windows**:

1. Ir a **Configuración del Firewall** → "Reglas de Entrada".
 2. Crear una nueva regla que permita conexiones TCP en el puerto **1521**.
 3. Aplicar la regla y reiniciar el servicio de Oracle.
-



En **Linux**:

Se puede permitir el tráfico con el siguiente comando:

```
sudo firewall-cmd --zone=public --add-port=1521/tcp --permanent
sudo firewall-cmd --reload
```

2.2. Archivos de configuración de red en Oracle

Para establecer una conexión entre el servidor y los clientes, Oracle utiliza varios archivos de configuración ubicados en:

-  **\$ORACLE_HOME/network/admin/** (en Linux)
-  **C:\Oracle\product\xx.x.x\dbhome_x\network\admin** (en Windows)

Los archivos más importantes son:

1. **listener.ora** → Configura el listener, que gestiona las conexiones externas.
 2. **sqlnet.ora** → Configura opciones de red y seguridad.
 3. **tnsnames.ora** → Define cómo los clientes encuentran y se conectan al servidor Oracle.
-

2.3. Configuración del Listener (listener.ora)

El listener es un proceso que escucha las solicitudes de conexión en el puerto **1521** y las redirige a la base de datos. Se configura en el archivo **listener.ora**.

Ejemplo de **listener.ora** en un servidor Oracle:

```
LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.100)(PORT = 1521))
    )
  )
```

Explicación:

PROTOCOL = TCP → Utiliza el protocolo TCP para conexiones remotas.

HOST = 192.168.1.100 → Dirección IP del servidor Oracle.

PORT = 1521 → Puerto donde Oracle escucha conexiones.

Comandos para administrar el Listener

Desde la terminal o el símbolo del sistema en Windows, se pueden ejecutar estos comandos:

- ✓ **Iniciar el listener** `lsnrctl start`
 - ✓ **Detener el listener** `lsnrctl stop`
 - ✓ **Ver el estado del listener** `lsnrctl status`
-

2.4. Configuración del Cliente (tnsnames.ora)

Los clientes Oracle usan el archivo **tnsnames.ora** para encontrar y conectarse a la base de datos.

Ejemplo de **tnsnames.ora** en un cliente:

```
ORCL =  
  (DESCRIPTION =  
    (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.100)(PORT = 1521))  
    (CONNECT_DATA =  
      (SERVER = DEDICATED)  
      (SERVICE_NAME = orcl)  
    )  
  )  
)
```

Explicación:

HOST = 192.168.1.100 → IP del servidor Oracle.

PORT = 1521 → Puerto de comunicación.

SERVICE_NAME = orcl → Nombre del servicio de base de datos.

Comprobación de la conexión

Para probar si la conexión funciona, en el cliente se ejecuta:

```
tnsping ORCL
```

Si responde correctamente, significa que la configuración es correcta.

2.5. Uso del asistente de configuración de red (netca)

Oracle proporciona una herramienta gráfica llamada **netca** (Network Configuration Assistant) que permite configurar los archivos de red de manera más sencilla.

Para ejecutarlo en Linux:

```
netca
```

En Windows:

- Abrir el **Oracle Net Manager** desde el menú de Oracle.

Con esta herramienta se pueden:

- ✓ Crear y eliminar **listeners**.
 - ✓ Configurar el acceso de clientes.
 - ✓ Administrar la seguridad de conexiones.
-

2.6. Resumen de la configuración de conexión en Oracle

1. **Tener una IP accesible** y evitar direcciones APIPA.
 2. **Configurar VirtualBox** si se usa una máquina virtual.
 3. **Abrir el puerto 1521 en el firewall.**
 4. **Configurar el archivo listener.ora** en el servidor.
 5. **Configurar el archivo tnsnames.ora** en los clientes.
 6. **Probar la conexión con tnsping.**
 7. **Usar netca para configurar la red** de forma más sencilla.
-

Punto 3. Primera Conexión a la Base de Datos en Oracle

Este apartado del documento explica cómo realizar la primera conexión a una base de datos Oracle utilizando la herramienta **SQL*Plus**. SQL*Plus es una interfaz de línea de comandos que permite a los administradores y desarrolladores interactuar con la base de datos Oracle mediante comandos SQL y PL/SQL.

3.1. Iniciar SQL*Plus y conectarse como administrador

Para conectarse a una base de datos Oracle, se utiliza el siguiente comando en la terminal (tanto en Windows como en Linux):

```
sqlplus / as sysdba
```

Explicación del comando:

- `sqlplus` → Ejecuta la herramienta SQL*Plus.
- `/` → Indica que no se usará un usuario ni contraseña explícitamente, sino que se conectará como el usuario predeterminado.
- `as sysdba` → Se conecta como administrador de la base de datos (SYSDBA), que tiene privilegios especiales para gestionar Oracle.

Ejemplo en Windows:

```
C:\Users\enric> sqlplus / as sysdba
```

Si la conexión es exitosa, SQL*Plus mostrará un mensaje de bienvenida y se podrá empezar a ejecutar comandos.

3.2. Comandos básicos para verificar la conexión

Una vez dentro de SQL*Plus, se pueden ejecutar varios comandos para obtener información sobre la base de datos:

Mostrar el nombre de la base de datos actual:

```
SQL> show con_name;
```

Este comando muestra en qué contenedor (CDB o PDB) estamos conectados.

Si la base de datos es de tipo **multitenant**, puede haber varias bases de datos dentro de una misma instancia, llamadas **Pluggable Databases (PDBs)**.

Obtener el nombre de la base de datos

```
SQL> select name from v$database;
```

Este comando consulta la vista dinámica v\$database para obtener el nombre de la base de datos.

Ejemplo de salida:

```
NAME  
-----  
ORCL
```

Significa que estamos conectados a una base de datos llamada **ORCL**.

Mostrar el usuario actual

```
SQL> show user;
```

Indica con qué usuario estamos conectados a la base de datos.

Ejemplo de salida:

```
USER is "SYS"
```

Si aparece **SYS**, significa que estamos conectados con el usuario administrador o que estamos con otro usuario actuando con el ROL de SYSDBA (administrador).

Ver las PDBs disponibles

```
SQL> show pdbs;
```

Este comando muestra todas las Pluggable Databases (PDBs) dentro de la Container Database (CDB).

Ejemplo de salida:

| CON_ID | CON_NAME | OPEN MODE | RESTRICTED |
|--------|-----------|------------|------------|
| 2 | PDB\$SEED | READ ONLY | NO |
| 3 | PDB1 | READ WRITE | NO |
| 4 | PDB2 | MOUNTED | NO |

Explicación de la salida:

- **PDB\$SEED** → Es una PDB de plantilla usada para crear otras PDBs. Está en estado **READ ONLY** y no se puede usar.
- **PDB1** → Está en estado **READ WRITE**, lo que significa que está activa y se puede modificar.
- **PDB2** → Está en estado **MOUNTED**, por lo que no está completamente abierta y no se puede usar aún.

Si se quiere cambiar a otra PDB, se usa este comando:

```
SQL> alter session set CONTAINER=PDB1;
```

Mostrar la memoria asignada a la base de datos (SGA - System Global Area)

```
SQL> show sga;
```

Este comando muestra el tamaño total de la memoria asignada a Oracle.

Ejemplo de salida:

```
Total System Global Area 125829120 bytes
Fixed Size                 8908328 bytes
Variable Size              79691736 bytes
Database Buffers          33554432 bytes
Redo Buffers               3686400 bytes
```

Explicación de los valores:

- **Total System Global Area** → Memoria total asignada al sistema.
- **Fixed Size** → Memoria reservada para estructuras internas de Oracle.
- **Variable Size** → Memoria utilizada dinámicamente para procesos.
- **Database Buffers** → Memoria usada para almacenar datos de las tablas y registros.
- **Redo Buffers** → Memoria usada para gestionar registros de cambios (Redo Logs).

3.3. Configuración de Firewall para permitir la conexión

El documento menciona que el **puerto 1521** debe estar permitido en el firewall, ya que es el puerto predeterminado de Oracle.

En Windows, se puede permitir el puerto 1521 con el siguiente comando en la terminal de administración (CMD con permisos de administrador):

```
netsh advfirewall firewall add rule name="Permitir Oracle" dir=in action=allow
protocol=TCP localport=1521
```

En Linux, se puede permitir con:

```
sudo firewall-cmd --permanent --add-port=1521/tcp  
sudo firewall-cmd --reload
```

Esto es necesario si se quiere acceder a la base de datos desde otro equipo o cliente externo.

Resumen de los pasos para conectarse por primera vez

Abrir una terminal en Windows o Linux.

Ejecutar el comando:

```
sqlplus / as sysdba
```

Verificar el nombre de la base de datos con:

```
show con_name;
```

Consultar el usuario actual con:

```
show user;
```

Ver las bases de datos disponibles con:

```
show pdbs;
```

Si es necesario, cambiar a una PDB específica:

```
alter session set CONTAINER=PDB1;
```

Configurar el firewall para permitir conexiones externas en el puerto 1521.

Conclusión

Este apartado del documento explica cómo realizar la primera conexión a una base de datos Oracle utilizando **SQL*Plus** y cómo verificar la información básica de la base de datos. También introduce algunos conceptos importantes, como **Pluggable Databases (PDBs)** y la memoria del sistema (**SGA**).

Punto 4. Configuración de la Instancia de Oracle

En Oracle, una **instancia** es el conjunto de procesos en memoria que gestionan la base de datos. Su configuración es clave para el rendimiento y la estabilidad del sistema.

4.1. Archivos de Configuración de la Instancia

Oracle utiliza un archivo especial llamado **SPFILE (Server Parameter File)** para almacenar parámetros de configuración. Este archivo:

- Es **binario** y no puede editarse manualmente.
- Sustituye al antiguo **init.ora** (archivo de texto usado en versiones anteriores de Oracle).
- Solo puede modificarse a través de comandos SQL.

Ubicación del SPFILE:

Se encuentra en la ruta `$ORACLE_BASE/database` y su nombre es `spfile<SID>.ora`, donde `<SID>` es el identificador de la instancia.

4.2. Modificación de Parámetros de la Instancia

Para modificar los parámetros de la instancia, se usa el comando `ALTER SYSTEM SET`.

Ejemplo básico:

```
ALTER SYSTEM SET sga_target = 1024M;
```

Pero **hay tres formas de aplicar cambios**, según el ámbito del ajuste:

| Modo | Descripción |
|---------------|--|
| SPFILE | El cambio se guarda en el SPFILE y se aplicará en el próximo reinicio de la base de datos. |
| MEMORY | El cambio se aplica de inmediato en memoria, pero no persiste tras un reinicio. |
| BOTH | Se aplica en memoria y se guarda en el SPFILE para que persista tras un reinicio. |

Ejemplo:

```
ALTER SYSTEM SET processes = 300 SCOPE = BOTH;
```

4.3. Visualización de Parámetros de Configuración

Puedes consultar la configuración actual de la instancia con los siguientes comandos:

Mostrar todos los parámetros de Oracle:

```
SHOW PARAMETERS;
```

Ver los parámetros relacionados con la memoria (SGA):

```
SHOW PARAMETERS SGA;
```

Consultar la vista v\$parameter para más detalles:

```
SELECT name, value, isdefault FROM v$parameter;
```

Ver los parámetros guardados en el SPFILE:

```
SHOW SPPARAMETERS;
```

Esto muestra los valores almacenados en el **SPFILE**, incluso si aún no están activos.

4.4. Cambios en la Configuración de la Instancia

Existen parámetros que pueden cambiarse dinámicamente y otros que requieren reiniciar la base de datos.

Cambios sin reinicio (dinámicos):

```
ALTER SYSTEM SET shared_pool_size = 500M SCOPE = BOTH;
```

Este cambio se aplica al instante y se mantiene tras el reinicio.

Cambios que requieren reinicio:

```
ALTER SYSTEM SET processes = 500 SCOPE = SPFILE;
```

En este caso, como se usa `SCOPE = SPFILE`, el cambio solo entrará en vigor después de reiniciar la base de datos.

4.5. Configuración Gráfica del SPFILE

Además de la línea de comandos, también se pueden modificar parámetros de la instancia usando herramientas gráficas como:

- **Oracle Enterprise Manager (OEM)** (ya en desuso en algunas versiones).
 - **SQL Developer** (opciones de administración avanzadas).
-

Conclusión

La configuración de la instancia de Oracle es clave para un buen rendimiento del sistema. Los parámetros de configuración deben manejarse con cuidado, y los cambios permanentes se deben aplicar en el **SPFILE**. Para ajustes temporales, se pueden aplicar en **MEMORY** sin afectar la configuración persistente.

Punto 5. Gestión de Conexiones y Usuarios en Oracle

En Oracle, la gestión de conexiones y usuarios es fundamental para administrar correctamente el acceso a la base de datos. Este apartado trata sobre los diferentes tipos de cuentas de administración y cómo conectarse a la base de datos con ellas.

5.1. Tipos de Cuentas de Administración en Oracle

Oracle tiene tres cuentas principales para la administración del sistema:

1. SYS

- Es el usuario más poderoso de Oracle.
- Contiene todas las tablas del sistema y vistas del diccionario de datos.
- Puede realizar cualquier operación, como iniciar y detener la base de datos.
- Se recomienda **no usarlo** para tareas diarias debido a su alto nivel de privilegios.

2. SYSTEM

- Similar a SYS, pero con algunas restricciones:
 - No puede realizar tareas de **backup y recuperación**.
 - No puede **actualizar** el software del SGBD.
 - No puede **iniciar ni detener** la base de datos.
- Se usa para la administración diaria de la base de datos.

3. PDBADMIN

- Usuario administrador de cada **Pluggable Database (PDB)** en Oracle Multitenant.
- No tiene permisos avanzados por defecto, solo puede conectarse a su PDB específica.
- Puede recibir permisos adicionales según sea necesario.

Nota:

Cuando se crea una base de datos en Oracle, los usuarios SYS y SYSTEM se crean automáticamente y están activados. En cambio, PDBADMIN solo se activa si se trabaja con bases de datos pluggable (PDBs).

5.2. Conectar a una Base de Datos en Oracle

Para conectarse a Oracle desde una terminal o línea de comandos, se usa **SQL*Plus**, una herramienta para ejecutar comandos SQL.

Conexión a la Base de Datos por Defecto

El siguiente comando permite conectarse como administrador a la base de datos principal (CDB o PDB por defecto):

```
sqlplus / as sysdba
```

Esto se usa cuando se ejecuta desde el mismo servidor donde está instalada la base de datos.

5.3. Conexión a una Base de Datos Específica (CDB o PDB)

Si queremos conectarnos a una base de datos distinta de la que se abre por defecto, debemos especificarlo en la conexión.

Ejemplo de conexión como SYS en una PDB específica:

```
sqlplus sys@localhost/pdb1 as sysdba
```

Aquí `pdb1` es el nombre de la Pluggable Database (PDB) a la que nos queremos conectar.

Ejemplo de conexión como SYSTEM en una PDB específica:

```
sqlplus system@localhost/pdb1 as sysdba
```

Esto conecta al usuario `SYSTEM` a la PDB `pdb1` como administrador.

5.4. Conexión Remota a la Base de Datos

Si estamos en otro equipo y queremos conectarnos a la base de datos en un servidor remoto, debemos usar la dirección IP o el nombre del servidor.

Ejemplo de conexión remota con una dirección IP:

```
sqlplus system@192.168.0.214/nomPdb1
```

Se reemplaza `192.168.0.214` con la IP del servidor y `nomPdb1` con el nombre de la base de datos.

Conexión usando un nombre de dominio (DNS)

Si el servidor tiene un nombre de dominio configurado, podemos usarlo en lugar de la IP:

```
sqlplus system@nomdomini.com/orcl
```

Aquí `nomdomini.com` es el nombre del dominio del servidor y `orcl` es el nombre de la base de datos.

5.5. Conexión con Usuario y Contraseña en la Línea de Comandos

Si queremos conectarnos sin que nos pida la contraseña de forma interactiva, podemos incluirla en el comando (aunque no se recomienda por seguridad):

```
sqlplus system/password@192.168.0.214/nomPdb1
```

Aquí `password` es la contraseña del usuario `system`.

Advertencia:

Incluir la contraseña en la línea de comandos puede ser un riesgo de seguridad, ya que otros usuarios del sistema pueden verla.

Resumen

- **SYS:** Usuario con todos los privilegios, usado para tareas críticas.
- **SYSTEM:** Usuario administrativo con permisos limitados.
- **PDBADMIN:** Usuario para administrar una PDB específica.
- Podemos conectarnos a una base de datos usando `sqlplus` desde el servidor o de forma remota con IP o dominio.
- Es posible especificar la contraseña en la línea de comandos, aunque no es recomendable por seguridad.

Punto 6. Comandos en SQL*Plus

SQL*Plus es una herramienta de línea de comandos que permite interactuar con una base de datos Oracle. Se usa para ejecutar sentencias SQL, realizar tareas administrativas y gestionar la base de datos. A continuación, te explico los comandos más importantes mencionados en el documento.

Principales Comandos en SQL*Plus

Estos comandos son útiles para navegar y manipular la base de datos desde la línea de comandos de SQL*Plus.

Comandos de edición y ayuda

- **edit** → Abre un editor de texto para modificar la última consulta ejecutada.
- **define_editor=notepad** → Cambia el editor de texto predeterminado a Notepad en Windows.
- **help** → Muestra ayuda sobre comandos disponibles en SQL*Plus.

Comandos para listar y ejecutar sentencias

- **list** → Muestra la última consulta escrita en el buffer de SQL*Plus.
- **run** o **r** o **/** → Ejecuta la última consulta escrita en el buffer.
- **save nombre_fichero.ext** → Guarda el contenido actual del buffer en un archivo.
- **get nombre_fichero.ext** → Carga un archivo SQL en el buffer.

Comandos de manipulación del buffer

- **a texto** → Añade texto al final de la última línea del buffer.
- **c /texto_anterior/texto_nuevo/** → Sustituye `texto_anterior` por `texto_nuevo` en la última consulta escrita.
- **clear buffer** → Borra el contenido del buffer de SQL*Plus.

Comandos de eliminación

- **del** → Borra líneas de código del buffer.
-

Funcionamiento del buffer en SQL*Plus

Cuando escribes una consulta en SQL*Plus, esta se almacena en un buffer hasta que la ejecutas. Solo se guarda la última consulta, por lo que si escribes una nueva, la anterior se pierde a menos que la guardes en un archivo.

Ejemplo de uso del buffer:

```
SQL> SELECT * FROM empleados;
```

Para ver la consulta en el buffer:

```
SQL> list
```

Si quieres modificarla, puedes escribir:

```
SQL> c/empleados/clientes/
```

Esto cambiará `empleados` por `clientes` en la consulta.

Importancia de SQL*Plus

SQL*Plus es una herramienta fundamental en la administración de bases de datos Oracle. Se utiliza en entornos de producción, pruebas y desarrollo para ejecutar comandos, consultar datos y gestionar el sistema de bases de datos.

Punto 7. Estados del Servidor Oracle y su Administración

En Oracle, el servidor de base de datos pasa por diferentes estados durante su ciclo de vida, desde su inicio hasta su apagado. Estos estados determinan qué operaciones pueden realizarse sobre la base de datos en cada momento.

7.1. Estados del Servidor Oracle

Oracle tiene cuatro estados principales:

1. Shutdown (Apagado)

- La base de datos está completamente cerrada.
- Ningún usuario ni proceso puede acceder a la base de datos.
- Se puede iniciar con el comando **STARTUP**.

2. Nomount (Sin montar)

- La instancia de Oracle se inicia, pero no se ha montado ninguna base de datos.
- Se cargan los procesos en memoria y se asigna la SGA (System Global Area).
- En este estado, se pueden realizar operaciones como la creación de una nueva base de datos o la recuperación del control de archivos.
- Se activa con **STARTUP NOMOUNT**.

3. Mount (Montado)

- La base de datos se asocia a la instancia, pero aún no está accesible para los usuarios.
- Se leen los archivos de control (**control files**), pero los datos aún no están disponibles.
- Útil para realizar tareas de mantenimiento como restauraciones y recuperación de datos.
- Se activa con **STARTUP MOUNT** o **ALTER DATABASE MOUNT**.

4. Open (Abierta)

- La base de datos está completamente funcional y los usuarios pueden conectarse.
 - Se accede a los **datafiles** y **redo logs**.
 - Se activa con **STARTUP OPEN** o **ALTER DATABASE OPEN**.
 - Solo en este estado los usuarios pueden ejecutar consultas y transacciones.
-

7.2. Comandos para Arrancar y Apagar la Base de Datos

Los **administradores** de bases de datos (ROL SYSDBA) pueden controlar el estado de Oracle con comandos SQL y SQL*Plus.

Arrancar la Base de Datos

```
SQL> STARTUP;
```

➔ Este comando arranca la base de datos en estado OPEN por defecto (si no hay errores).

Opciones avanzadas:

```
SQL> STARTUP NOMOUNT;  -- Inicia la instancia pero no monta la base de datos.
SQL> STARTUP MOUNT;    -- Inicia la instancia y monta la base de datos, pero no
                        -- la abre.
SQL> STARTUP OPEN;     -- Inicia, monta y abre la base de datos.
```

Arrancar las PDB

Desde la CDB arrancada (open)

```
SQL> alter pluggable database <nomPDB> open;
SQL> alter pluggable database <nomPDB> save state;
SQL> alter pluggable database <nomPDB> close immediate;
```

Apagar la Base de Datos

Existen varios modos para apagar Oracle, dependiendo de si se quiere esperar a que los usuarios terminen su trabajo o forzar el cierre:

```
SQL> SHUTDOWN NORMAL;
```

➔ Espera a que todos los usuarios se desconecten antes de cerrar la base de datos.

```
SQL> SHUTDOWN TRANSACTIONAL;
```

➔ Espera a que se completen las transacciones activas antes de apagar la base de datos.

```
SQL> SHUTDOWN IMMEDIATE;
```

➔ Cierra inmediatamente todas las sesiones, revierte transacciones en curso y apaga la base de datos rápidamente (recomendado en la mayoría de los casos).

```
SQL> SHUTDOWN ABORT;
```

➔ Cierra la base de datos de inmediato sin esperar, pero deja la base de datos en un estado inconsistente. **No se recomienda**, salvo en emergencias.

7.3. Cambiar entre Estados

Si ya se ha iniciado la base de datos y se quiere cambiar entre estados, se pueden usar estos comandos:

➔ **Pasar de NOMOUNT a MOUNT:**

```
SQL> ALTER DATABASE MOUNT;
```

➔ **Pasar de MOUNT a OPEN:**

```
SQL> ALTER DATABASE OPEN;
```

➡ Pasar de OPEN a MOUNT:

```
SQL> ALTER DATABASE CLOSE;  
SQL> ALTER DATABASE MOUNT;
```

➡ Pasar de MOUNT a SHUTDOWN:

```
SQL> SHUTDOWN IMMEDIATE;
```

7.4. Cómo Ver el Estado Actual de la Base de Datos

Para verificar el estado actual de la base de datos en Oracle, se pueden usar estos comandos:

```
SQL> SELECT INSTANCE_NAME, STATUS, DATABASE_STATUS FROM V$INSTANCE;
```

Muestra el estado de la instancia Oracle.

```
SQL> SELECT DATABASE_NAME, OPEN_MODE FROM V$DATABASE;
```

Muestra el nombre de la base de datos y si está en modo abierto, montado o cerrado.

```
SQL> SHOW PARAMETER INSTANCE;
```

Muestra información sobre la instancia actual.

7.5. Modos Especiales de Inicio

Modo Restringido

Se usa cuando se necesita realizar mantenimiento y solo los administradores pueden conectarse.

```
SQL> STARTUP RESTRICT;  
SQL> ALTER SYSTEM ENABLE RESTRICTED SESSION;
```

Modo Actualización

Se usa para actualizar la base de datos sin permitir conexiones de usuarios.

```
SQL> STARTUP UPGRADE;
```

Modo Forzado

Si la base de datos está bloqueada y no responde, se puede forzar el arranque:

```
SQL> STARTUP FORCE;
```

Este comando equivale a hacer un SHUTDOWN ABORT y luego STARTUP.

Conclusión

- Oracle tiene diferentes estados (SHUTDOWN, NOMOUNT, MOUNT, OPEN).
 - Se pueden cambiar entre estados con comandos SQL (STARTUP, SHUTDOWN, ALTER DATABASE).
 - Se pueden verificar los estados con consultas a V\$INSTANCE y V\$DATABASE.
 - Existen modos especiales como RESTRICTED, UPGRADE y FORCE para tareas avanzadas.
-

Punto 8. Gestión de Tablespaces y Datafiles en Oracle

En Oracle, un **tablespace** es una unidad lógica de almacenamiento dentro de la base de datos. Los **datafiles** son los archivos físicos que contienen los datos de un tablespace.

El objetivo de esta sección es explicar cómo se crean, modifican y eliminan tablespaces y datafiles en Oracle.

8.1. ¿Qué es un Tablespace?

- Es un espacio lógico dentro de la base de datos donde se almacenan los objetos (tablas, índices, etc.).
 - Cada tablespace puede contener uno o más datafiles.
 - Se recomienda usar varios tablespaces para organizar y gestionar mejor los datos.
-

8.2. Tipos de Tablespaces en Oracle

Oracle crea algunos tablespaces por defecto al instalarse:

1. **SYSTEM** → Contiene la información básica del diccionario de datos.
2. **SYSAUX** → Almacena información auxiliar y estadísticas.
3. **UNDO** → Gestiona transacciones y permite realizar rollbacks.
4. **TEMP** → Se usa para operaciones temporales (ordenaciones, joins grandes, etc.).
5. **USERS** → Tablespace por defecto para usuarios.

Oracle también permite crear tablespaces adicionales según las necesidades.

8.3. Creación de un Tablespace

Para crear un tablespace, usamos la sentencia **CREATE TABLESPACE**:

```
CREATE TABLESPACE mitablespace  
DATAFILE '/ruta/del/datafile.dbf'
```

```
SIZE 100M  
AUTOEXTEND ON NEXT 10M  
MAXSIZE 1G;
```

Explicación de los parámetros:

- `DATAFILE '/ruta/del/datafile.dbf'` → Define el archivo físico donde se almacenarán los datos.
 - `SIZE 100M` → Establece el tamaño inicial del datafile.
 - `AUTOEXTEND ON NEXT 10M` → Permite que el datafile crezca automáticamente en bloques de 10MB.
 - `MAXSIZE 1G` → Limita el crecimiento del datafile a un máximo de 1GB.
-

8.4. Modificación de un Tablespace

Si queremos modificar un tablespace existente, podemos hacer lo siguiente:

A. Añadir un nuevo Datafile

```
ALTER TABLESPACE mitablespace  
ADD DATAFILE '/ruta/nuevo_datafile.dbf' SIZE 200M;
```

Esto agregará un nuevo archivo de datos de 200MB al tablespace.

B. Cambiar el tamaño de un Datafile existente

```
ALTER DATABASE DATAFILE '/ruta/del/datafile.dbf'  
RESIZE 300M;
```

Aumenta o reduce el tamaño del datafile a 300MB.

8.5. Eliminación de un Tablespace

Para eliminar un tablespace, usamos la instrucción `DROP TABLESPACE`.

A. Eliminar un tablespace sin borrar los archivos físicos

```
DROP TABLESPACE mitablespace;
```

Esto elimina el tablespace de la base de datos, pero los archivos de datos seguirán existiendo en el disco.

B. Eliminar un tablespace y sus datafiles

```
DROP TABLESPACE mitablespace INCLUDING CONTENTS AND DATAFILES;
```

Este comando borra el tablespace junto con los archivos físicos.

¡Importante!

Si un usuario tenía ese tablespace como predeterminado, seguirá vinculado a él aunque el tablespace haya sido eliminado. Para evitar problemas, es recomendable reasignarle otro tablespace antes de eliminarlo:

```
ALTER USER usuario DEFAULT TABLESPACE users;
```

8.6. Permisos para Gestionar Tablespaces

No cualquier usuario puede modificar tablespaces, solo aquellos con los permisos adecuados.

Los permisos necesarios son:

- `CREATE TABLESPACE` → Para crear tablespaces.
- `ALTER TABLESPACE` → Para modificar tablespaces.
- `DROP TABLESPACE` → Para eliminar tablespaces.
- `MANAGE TABLESPACE` → Para tareas avanzadas de administración.
- `ALTER DATABASE` → Permite cambiar parámetros de la base de datos, incluyendo tablespaces.

Estos permisos pueden asignarse con:

```
GRANT CREATE TABLESPACE TO usuario;  
GRANT DROP TABLESPACE TO usuario;
```

8.7. Localización de los Tablespaces en Oracle

- En una base de datos tradicional (CDB), los tablespaces pertenecen a la base de datos.
- En un entorno **multitenant** con varias PDBs (Pluggable Databases), cada PDB puede tener su propio conjunto de tablespaces.

Para ver los tablespaces disponibles en una base de datos:

```
SELECT tablespace_name FROM dba_tablespaces;
```

Para ver los archivos de datos asociados a un tablespace:

```
SELECT file_name, tablespace_name FROM dba_data_files;
```

8.8. Mover un Datafile a otra Ubicación

Si necesitas mover un datafile de un tablespace a otra ubicación, sigue estos pasos:

1. Poner el tablespace en modo offline

```
ALTER TABLESPACE mitablespace OFFLINE;
```

2. Mover el archivo de datos a la nueva ubicación (Desde el sistema operativo)

```
mv /ruta/original/datafile.dbf /nueva/ruta/datafile.dbf
```

3. Actualizar la ubicación en Oracle

```
ALTER DATABASE RENAME FILE '/ruta/original/datafile.dbf'  
TO '/nueva/ruta/datafile.dbf';
```

4. Volver a poner el tablespace online

```
ALTER TABLESPACE mitablespace ONLINE;
```

8.9. Uso de un Tablespace en una Tabla o Índice

Cuando creamos una tabla o un índice, podemos especificar en qué tablespace queremos que se almacene.

A. Crear una tabla en un tablespace específico

```
CREATE TABLE empleados (  
    id NUMBER(10),  
    nombre VARCHAR2(50)  
) TABLESPACE mitablespace;
```

B. Crear un índice en un tablespace específico

```
CREATE INDEX idx_empleados ON empleados(nombre)  
TABLESPACE mitablespace;
```

C. Mover una tabla a otro tablespace

```
ALTER TABLE empleados MOVE TABLESPACE otro_tablespace;
```

D. Cambiar el tablespace por defecto de un usuario

```
ALTER USER usuario DEFAULT TABLESPACE mitablespace;
```

Punto 9. El Diccionario de Datos en Oracle

El **diccionario de datos** es un conjunto de vistas que proporciona información sobre la base de datos, los objetos que contiene y los usuarios que tienen acceso a esos objetos. Oracle lo utiliza para gestionar los metadatos, es decir, la "información sobre los datos".

Los datos del diccionario de datos están almacenados en las vistas del esquema SYS. Las vistas del diccionario permiten consultar la estructura de la base de datos, los permisos de los usuarios, las

tablas, las columnas, etc. Este diccionario se encuentra dentro del **esquema SYS**, y contiene información acerca de los objetos, los usuarios, las sesiones, las transacciones, etc.

Vistas del Diccionario de Datos en Oracle

Existen varias vistas en Oracle que permiten acceder a la información sobre la base de datos. Estas vistas son categorizadas según el acceso que tenga el usuario. A continuación, se describen las principales:

1. **DBA_**xx:

Estas vistas contienen información sobre toda la base de datos y son accesibles solo por usuarios con privilegios de administrador (como el usuario SYS o SYSTEM).

Ejemplo:

- **DBA_TABLESPACES**: Información sobre los tablespaces.
- **DBA_USERS**: Información sobre todos los usuarios de la base de datos.

2. **USER_**xx:

Estas vistas contienen información sobre los objetos de la base de datos pertenecientes al usuario que está conectado. Solo el usuario propietario puede acceder a estas vistas.

Ejemplo:

- **USER_TABLES**: Muestra todas las tablas del usuario actual.

3. **ALL_**xx:

Estas vistas contienen información sobre los objetos que el usuario actual puede ver, es decir, los objetos que posee o para los que tiene permisos.

Ejemplo:

- **ALL_TABLES**: Muestra todas las tablas que el usuario puede acceder, independientemente de si es propietario de ellas.

4. **V\$**xx (**Vistas dinámicas**):

Las vistas V\$ son vistas dinámicas y muestran información sobre el estado actual de la base de datos. Estas vistas son útiles para monitorear el sistema en tiempo real.

Ejemplo:

- **V\$DATABASE**: Muestra información sobre la base de datos actual.
- **V\$SESSION**: Muestra información sobre las sesiones activas en la base de datos.
- **V\$INSTANCE**: Muestra información sobre la instancia de Oracle que está en ejecución.

Consultas Comunes al Diccionario de Datos

1. **Ver las tablas del usuario**

Si deseas consultar las tablas que existen en tu esquema (es decir, las tablas pertenecientes a tu usuario), puedes usar la siguiente consulta:

```
SELECT table_name FROM user_tables;
```

2. Ver las columnas de una tabla

Para obtener información detallada sobre las columnas de una tabla, como el nombre de las columnas, el tipo de datos, la precisión, etc., puedes utilizar la siguiente consulta:

```
SELECT column_name, data_type, data_default,  
       data_precision, data_scale, nullable  
FROM user_tab_columns  
WHERE table_name = 'EMPLOYEES';
```

Esta consulta te devolverá los detalles de las columnas de la tabla EMPLOYEES.

3. Ver las tablas a las que tienes acceso

Si deseas ver todas las tablas que puedes acceder, tanto si eres el propietario como si tienes permisos sobre ellas, puedes usar la siguiente consulta:

```
SELECT table_name FROM all_tables;
```

4. Ver las tablas de un propietario específico

Si tienes permisos sobre las tablas de otros esquemas y deseas ver las tablas de un propietario específico, puedes usar esta consulta:

```
SELECT table_name FROM all_tables WHERE owner = 'JUAN';
```

5. Describir una vista o tabla

Si deseas conocer la estructura de una tabla o vista, puedes utilizar el comando DESCRIBE. Este comando muestra las columnas, tipos de datos y otras propiedades de la tabla o vista:

```
DESCRIBE user_tables;
```

También puedes usar DESC como abreviatura.

Consultas sobre el Diccionario de Datos de Oracle

A continuación, algunas de las vistas más importantes y consultas relacionadas con ellas:

- **DBA_TABLESPACES:** Muestra información sobre los tablespaces en la base de datos.

```
SELECT tablespace_name FROM dba_tablespaces;
```
- **V\$DATABASE:** Muestra información sobre la base de datos actual, como su nombre, estado, etc.

```
SELECT name FROM v$database;
```
- **V\$SESSION:** Muestra información sobre las sesiones activas.

```
SELECT sid, username, status FROM v$session;
```
- **V\$LOGFILE:** Muestra información sobre los archivos de registro (logfiles).

```
SELECT * FROM v$logfile;
```
- **V\$SYSTEM_PARAMETER:** Muestra los parámetros del sistema de la base de datos.

```
SELECT name, value FROM v$system_parameter;
```

Acceso al Diccionario de Datos

Para acceder al diccionario de datos, se utilizan consultas SQL que pueden ser ejecutadas por usuarios con los permisos adecuados. Los usuarios con privilegios **SYSDBA** o **SYSTEM** pueden acceder a todas las vistas, mientras que los usuarios con privilegios más limitados solo pueden acceder a las vistas que les corresponden.

punto 10: Gestión del Redo Log y Archive Log

Trata sobre cómo Oracle maneja los registros de transacciones para garantizar la integridad de la base de datos y permitir su recuperación en caso de fallos. Vamos a desglosarlo en detalle.

¿Qué es el Redo Log en Oracle?

El **Redo Log** es un conjunto de archivos que registran todos los cambios realizados en la base de datos antes de que se escriban definitivamente en los **datafiles**. Su objetivo principal es la **recuperación de datos** en caso de fallo.

Funcionamiento:

1. Cuando se ejecuta una transacción en Oracle, los cambios no se escriben inmediatamente en los datafiles, sino que primero se guardan en memoria y en los **redo log buffers**.
2. Luego, el **LGWR (Log Writer Process)** escribe estos cambios en los archivos **Redo Log** en disco.
3. Si la base de datos sufre un fallo, estos registros permiten restaurar los datos hasta el último cambio registrado.

Estructura del Redo Log:

Los archivos de redo log están organizados en **grupos** y cada grupo tiene uno o más archivos físicos. El **LGWR** escribe en un grupo de redo log y, cuando se llena, pasa al siguiente grupo en un proceso llamado **log switch**.

Consulta de archivos Redo Log:

Puedes ver los redo logs con los siguientes comandos:

```
SQL> SELECT * FROM V$LOGFILE;  
SQL> SELECT * FROM V$LOG;
```

Estos comandos muestran información sobre los archivos redo log actuales.

¿Qué es el Archive Log en Oracle?

El **Archive Log** es un mecanismo que permite guardar copias de los archivos de redo log después de que se llenan y se cambian (log switch). Es fundamental para la **recuperación ante desastres** y para garantizar la disponibilidad de los datos.

Modos de operación de una base de datos Oracle:

1. Modo NOARCHIVELOG:

- Los redo logs antiguos se sobrescriben sin guardar una copia.
- No se pueden recuperar datos si hay una falla que afecte los datafiles.
- Es más rápido, pero menos seguro.

2. Modo ARCHIVELOG (Recomendado):

- Se genera una copia de cada redo log antes de sobrescribirse.
- Permite recuperación completa de la base de datos en caso de fallos.
- Requiere almacenamiento adicional.

Consulta del estado del Archive Log:

```
SQL> ARCHIVE LOG LIST;  
SQL> SELECT NAME, LOG_MODE FROM V$DATABASE;
```

Estos comandos muestran si la base de datos está en modo ARCHIVELOG o NOARCHIVELOG.

Cómo activar el Archive Log en Oracle

Para activar el **modo ARCHIVELOG**, debes seguir estos pasos:

1. Verificar el estado actual:

```
SQL> ARCHIVE LOG LIST;
```

2. Configurar la ubicación donde se guardarán los archivos Archive Log:

```
SQL> ALTER SYSTEM SET LOG_ARCHIVE_DEST_1='LOCATION=/ruta/a/archivelog'  
SCOPE=SPFILE;
```

3. Configurar el formato de los archivos de Archive Log:

```
SQL> ALTER SYSTEM SET LOG_ARCHIVE_FORMAT='arch_%r_%t_%s.arc' SCOPE=SPFILE;
```

4. Habilitar el Archive Log:

```
SQL> SHUTDOWN IMMEDIATE;  
SQL> STARTUP MOUNT;  
SQL> ALTER DATABASE ARCHIVELOG;  
SQL> ALTER DATABASE OPEN;
```

5. Verificar que el modo ARCHIVELOG está activado:

```
SQL> ARCHIVE LOG LIST;
```

¿Por qué es importante el Archive Log?

Permite recuperar la base de datos en caso de fallo: Si hay un corte de energía o un problema con el servidor, se pueden restaurar datos hasta el último redo log archivado.

Es obligatorio para estrategias de backup y recuperación con RMAN.

Permite la replicación de bases de datos en entornos de alta disponibilidad.

Rotación de Redo Logs (Switch Logfile)

Para forzar manualmente la rotación de redo logs, puedes usar:

```
SQL> ALTER SYSTEM SWITCH LOGFILE;
```

Esto fuerza el cambio de redo logs y puede ser útil para pruebas o mantenimiento.

Conclusión

- **Redo Log:** Registra todos los cambios antes de que se graben en disco.
 - **Archive Log:** Guarda copias de los redo logs antes de ser sobrescritos.
 - **Modo ARCHIVELOG:** Es clave para estrategias de recuperación de datos.
 - **Es recomendable habilitar el Archive Log en entornos de producción.**
-

Punto 11. Gestión de Logs en Oracle

Oracle Database genera diferentes tipos de logs (archivos de registro) para supervisar y depurar el sistema. Estos logs son fundamentales para el diagnóstico, la recuperación de datos en caso de fallo y el mantenimiento del sistema.

Tipos de Logs en Oracle

Oracle maneja varios tipos de archivos de registro, pero los más importantes son:

1. **Redo Log (Online Redo Log y Archived Redo Log)**
 2. **Alert Log**
 3. **Trace Logs**
 4. **Logs de Servicios**
-

Redo Log y Archive Log

El **Redo Log** es un componente crítico del sistema de recuperación de Oracle. Registra todas las modificaciones en la base de datos antes de que sean confirmadas en los datafiles.

Online Redo Log

- Se usa para la recuperación de la base de datos en caso de fallos.
- Contiene información sobre transacciones en curso y commit realizados.
- Se guarda en la vista dinámica V\$LOG.

Comando para visualizar los redo logs:

```
SELECT * FROM V$LOGFILE;  
SELECT * FROM V$LOG;
```

Funcionamiento:

- Oracle usa varios grupos de redo logs de manera cíclica.
- Cuando un grupo se llena, Oracle cambia automáticamente al siguiente.
- Si se produce un error antes de escribir en disco, el redo log permite recuperar la transacción.

Archived Redo Log (ARCHIVELOG Mode)

- Guarda copias de los redo logs después de ser llenados.
- Es obligatorio para bases de datos que necesitan alta disponibilidad y recuperación ante desastres.
- Se usa para restaurar cambios desde el último backup completo.
- Se guarda en la vista dinámica V\$ARCHIVED_LOG.

Activación del modo ARCHIVELOG El modo ARCHIVELOG no está activado por defecto en Oracle. Para activarlo, sigue estos pasos:

[1] Verificar el estado actual:

```
SQL> archive log list;
```

[2] Configurar el directorio de almacenamiento de logs archivados:

```
SQL> alter system set log_archive_dest_1='LOCATION=/archivelog/' SCOPE=SPFILE;
```

[3] Configurar el formato del nombre del archivo:

```
SQL> alter system set log_archive_format='arch_%r_%t_%s.arc' SCOPE=SPFILE;
```

[4] Habilitar el servicio de archivado:

```
SQL> alter system set LOG_ARCHIVE_START=TRUE SCOPE=spfile;
```

[5] Reiniciar la base de datos y activar ARCHIVELOG:

```
SQL> shutdown immediate;
```

```
SQL> startup mount;  
SQL> alter database archivelog;  
SQL> alter database open;
```

[6] Comprobar que está activo:

```
SQL> archive log list;  
SQL> select name, log_mode from v$database;
```

IMPORTANTE:

- **Modo NOARCHIVELOG:** Solo guarda redo logs en memoria y sobrescribe los más antiguos. Se usa para rendimiento, pero NO permite recuperación total de datos.
 - **Modo ARCHIVELOG:** Guarda redo logs en disco, permite recuperación completa en caso de fallo.
-

Alert Log (Registro de Alertas)

El **Alert Log** es un archivo de texto donde Oracle registra eventos críticos, como:

- ✓ Errores importantes.
- ✓ Arranques y paradas de la base de datos.
- ✓ Cambios en la configuración de parámetros.

Ubicación del Alert Log

En **Windows**:

C:\oracle\diag\rdbms\nombbdd\nombbdd\trace>alert_nombbdd.log

En **Linux**:

/u02/app/oracle/diag/rdbms/nombbdd/nombbdd/trace/alert_nombbdd.log

Para ver el log en Oracle:

```
SELECT * FROM V$DIAG_INFO WHERE NAME = 'Diag Trace';
```

Trace Logs (Registros de Depuración)

Los **Trace Logs** contienen información detallada sobre la ejecución de procesos internos y errores. Se generan automáticamente cuando ocurre un error grave.

- ✓ Se almacenan en la carpeta `trace`.
- ✓ Son usados por administradores para resolver problemas complejos.

Para ver la ubicación de estos logs:

```
SELECT * FROM V$DIAG_INFO WHERE NAME = 'Default Trace File';
```

Ejemplo de un error que generaría un trace log:

ORA-00600: internal error code

Si aparece este error, Oracle genera automáticamente un archivo de trazas en el directorio de logs.

Logs de Servicios (Listener, Startup)

Oracle genera logs de ciertos servicios, como el **Listener** o el arranque de la base de datos.

Listener Log (Conexiones a la BBDD)

- Contiene información sobre intentos de conexión a la base de datos.
- Se encuentra en `$ORACLE_HOME/network/log/listener.log`.

Para ver su contenido:

```
tail -f $ORACLE_HOME/network/log/listener.log
```

Startup Log (Arranque de la BBDD)

- Contiene información sobre el proceso de inicio de Oracle.
 - Se encuentra en `$ORACLE_HOME/startup.log`.
-

Resumen

| Tipo de Log | Función | Ubicación |
|--------------------------|---|-------------------|
| Online Redo Log | Registro de cambios en la BBDD para recuperación. | V\$LOGFILE |
| Archived Redo Log | Copias de redo logs para recuperación de desastres. | V\$ARCHIVED_LOG |
| Alert Log | Errores críticos y eventos importantes. | alert_nombbdd.log |
| Trace Log | Información detallada de errores internos. | V\$DIAG_INFO |
| Listener Log | Registro de conexiones al SGBD. | listener.log |
| Startup Log | Información sobre el arranque de Oracle. | startup.log |

Consejos clave:

- El modo **ARCHIVELOG** es crucial para recuperar la base de datos después de un fallo.
 - El **Alert Log** debe revisarse regularmente para detectar problemas en Oracle.
 - Los **Trace Logs** son útiles para depurar errores internos.
 - El **Listener Log** ayuda a diagnosticar problemas de conexión.
-