Tema I
Arquitectura básica: Instancia y Base de Datos

- **Instancia**: Es el conjunto de memoria (SGA) y procesos en ejecución que permiten interactuar con una base de datos.
- **Base de datos**: Conjunto de datos organizados almacenados en discos. Incluye archivos de datos, control y redo logs.
- Relación: Una instancia accede y gestiona una base de datos.

Memoria

- **SGA (System Global Area)**: Memoria compartida por todos los procesos. Componentes principales:
- *Buffer Cache*: Almacena datos leídos/escritos recientemente.
- *Shared Pool*: Guarda instrucciones SQL y PL/SQL compiladas.
- *Redo Log Buffer*: Registra cambios realizados para recuperación.
- *Java Pool* y *Large Pool*: Espacios adicionales para tareas específicas.
- **PGA (Program Global Area)**: Memoria privada asignada a cada proceso de usuario.

Procesos

- **Procesos de fondo**: Ejecutan tareas esenciales. Ejemplos clave:
- *DBWn (Database Writer)*: Escribe datos modificados en disco.
- *LGWR (Log Writer)*: Escribe entradas del Redo Log Buffer en los archivos redo log.
- *CKPT (Checkpoint)*: Marca puntos de recuperación en archivos de control.
- *SMON (System Monitor)*: Maneja recuperación de instancias.
- *PMON (Process Monitor)*: Limpia procesos fallidos.

- **Procesos de usuario**: Interactúan con la base de datos para ejecutar consultas y comandos.

Ficheros

- **Archivos de datos**: Contienen tablas, índices y otros objetos.
- **Archivos de control**: Guardan información de estructura y estado de la base de datos.
- **Redo Logs**: Registran todas las transacciones para recuperación.
- **Archivos temporales**: Soportan operaciones como sorting.
- **Archivos de parámetros (SPFILE/PFILE)**: Configuran la instancia.

Arquitectura Multitenant

- Introducida en Oracle 12c, permite consolidar múltiples bases de datos (PDBs) dentro de un contenedor único (CDB), optimizando recursos.

Diferencia entre CDB y PDB

- **CDB (Container Database)**:
- Base de datos contenedor que gestiona recursos compartidos y procesos.
- Contiene PDBs y el Root Container (información común).
- **PDB (Pluggable Database)**:
- Base de datos independiente, alojada dentro de un CDB.
- Cada PDB tiene sus propios datos y esquemas.

Ventajas: Aislamiento, gestión centralizada y facilidad de migración.

Tema II
Tema II: Creación de Base de Datos

Consejo Multitenant

- Oracle recomienda usar la arquitectura multitenant para consolidación de bases de datos, mejorando la administración, seguridad y escalabilidad.
- A partir de Oracle 21c, **multitenant es obligatorio**, eliminando la arquitectura tradicional.

DBCA (Database Configuration Assistant)

Herramienta gráfica para crear, configurar y administrar bases de datos Oracle.

1. Crear Base de Datos Tradicional (19c e inferiores):

- Selección manual de opciones para bases de datos autónomas (no multitenant).
- Incluye configuración de parámetros básicos como:
- Ubicación de archivos.
- Configuración de memoria (SGA y PGA).
- Creación de usuarios y esquemas básicos.

2. Crear Base de Datos Multitenant (21c):

- En versiones 21c, solo se pueden crear **CDBs con PDBs**.
- Pasos adicionales incluyen:
- Definición de cantidad de PDBs iniciales.
- Configuración de ficheros específicos para cada PDB.

Ficheros y Directorios

- **Creados por defecto:**
- Archivos de datos (*.dbf*), control y redo logs.

- Archivos de parámetros (SPFILE o PFILE).
- Directorio ORACLE_HOME y subdirectorios asociados a instancia.
- **Adicionales en 21c:**
- Directorios exclusivos para cada PDB dentro del contenedor (CDB).
- Archivos temporales y de recuperación asignados a cada PDB.

Procesos y Memoria

- **Tradicional:**
- Recursos dedicados a una única base de datos, gestionados por una instancia específica.
- **Multitenant:**
- **CDB (Contenedor):** Procesos y memoria compartidos.
- **PDBs (Bases de Datos Alojadas):** Reutilizan recursos del contenedor. Consumen menos memoria que instancias tradicionales.

Uso Avanzado de DBCA

1. Crear otra Base de Datos Tradicional (19c):

- Selección avanzada permite configurar:
- Ajuste detallado de memoria.
- Configuración de redo logs, archivos de control y almacenamiento.
- Personalización de esquemas iniciales.

2. Crear Multitenant:

- Configuración avanzada para definir el tamaño del contenedor, recursos, y número inicial de PDBs.
- Creación y asignación de directorios específicos.

Configurar Variables de Entorno (oraenv**)**

- Script para configurar variables necesarias para acceder a diferentes instancias de Oracle.
- **Pasos:**

1. Ejecutar oraenv.
2. Especificar el SID (identificador de la base de datos).
3. Variables configuradas: ORACLE_HOME, ORACLE_SID, PATH.

Ficheros, Memoria y Procesos en una Multitenant

- **Ficheros:** Cada PDB tiene archivos de datos y temporales separados dentro del almacenamiento asignado por el CDB.
- **Memoria:** CDB administra la memoria (SGA y PGA), compartiéndola entre las PDBs.
- **Procesos:** Procesos principales de la instancia del CDB gestionan el acceso y tareas comunes para todas las PDBs.

Tema III: SQL*Plus - Herramienta de Administración

Introducción a SQL*Plus

- **SQL*Plus** es una herramienta de línea de comandos proporcionada por Oracle para interactuar con la base de datos.
- Permite ejecutar comandos SQL, PL/SQL, y comandos específicos de SQL*Plus para administrar y gestionar bases de datos Oracle.
- **Características clave**:
- Fácil acceso para ejecutar consultas y administrar bases de datos.
- Soporte para scripts de automatización.
- Visualización y formato de resultados.

Un paseo por SQL*Plus

- **Acceso**:

4. Iniciar sesión desde la terminal usando:bash
   Copiar código

   ```
   sqlplus usuario/contraseña@base_datos
   ```

5. Para usar el entorno local, establecer el SID con oraenv y luego:bash
   Copiar código

   ```
   sqlplus / as sysdba
   ```

   Entorno básico:

1. Al iniciar, muestra un prompt con el símbolo SQL>.
2. Puedes ejecutar comandos SQL directamente o guardar scripts con extensión .sql.

- **Salida y formato**:

1. Los resultados se muestran en formato tabular por defecto.
2. Se puede personalizar con comandos como SET para ajustar el ancho de columnas, encabezados, etc.

Algunos comandos útiles de SQL*Plus

1. **Conexión y Desconexión**:

- CONNECT usuario/contraseña@db: Conecta a una base de datos.
- EXIT o QUIT: Salir de SQL*Plus.

2. **Comandos de entorno**:

- SET: Configura opciones de SQL*Plus.
  - SET LINESIZE n: Ajusta el ancho de las líneas.
  - SET PAGESIZE n: Establece el número de filas mostradas por página.
  - SET HEADING OFF: Oculta los encabezados de columna.
- CLEAR: Limpia buffers o ajustes:
  - CLEAR SCREEN: Limpia la pantalla.
  - CLEAR BUFFER: Limpia el buffer de comandos.

3. **Manipulación de scripts**:
- SPOOL archivo.log: Guarda la salida de la sesión en un archivo.
- @script.sql: Ejecuta un script guardado.
- EDIT script.sql: Abre el script para editar (requiere configurar un editor).

4. **Comandos de consulta**:
- DESCRIBE tabla: Muestra la estructura de una tabla.
- SELECT: Ejecuta consultas SQL.
- SHOW PARAMETER nombre: Muestra parámetros específicos de la base de datos.

5. **Comandos administrativos**:
- ALTER SYSTEM: Cambia parámetros a nivel de instancia.
- SHUTDOWN: Cierra la base de datos.
- STARTUP: Inicia la base de datos.

6. **Formateo de resultados**:
- COLUMN columna FORMAT formato: Ajusta el formato de una columna específica.
- BREAK ON columna: Agrupa resultados basados en una columna.

Tema IV: Administración Básica de la Base de Datos

Introducción al arranque de la Base de Datos

- El arranque de una base de datos Oracle implica inicializar la instancia, asociarla con la base de datos y abrirla para su uso.
- **Fases del arranque**:

1. **No Mount**: Solo se carga la instancia (procesos y memoria).
2. **Mount**: La instancia se conecta a los archivos de control.
3. **Open**: La base de datos está lista para operaciones.

Comando STARTUP

- Utilizado para arrancar la base de datos. Sintaxis básica:sql
  Copiar código

  ```
  STARTUP [OPEN|MOUNT|NOMOUNT];
  ```

- *NOMOUNT*: Solo instancia.
- *MOUNT*: Instancia + archivos de control.
- *OPEN*: Base de datos operativa (predeterminado).
- Ejemplo:sql
  Copiar código

  ```
  STARTUP MOUNT;
  ```

Parar la Base de Datos: Introducción

- Parar la base de datos implica cerrar conexiones activas, sincronizar cambios pendientes y liberar recursos de memoria y procesos.

Comando SHUTDOWN

- Sintaxis básica:sql
  Copiar código

  ```
  SHUTDOWN [NORMAL|IMMEDIATE|TRANSACTIONAL|ABORT];
  ```

- *NORMAL*: Espera a que los usuarios se desconecten.
- *IMMEDIATE*: Desconecta usuarios y guarda transacciones.
- *TRANSACTIONAL*: Completa transacciones activas antes de desconectar.
- *ABORT*: Parada inmediata, sin guardar cambios pendientes.

Diccionario de Datos: Tablas y Vistas

- **Diccionario de Datos**: Conjunto de tablas y vistas que contienen información sobre la estructura de la base de datos.
- Ejemplo: USER_TABLES, ALL_TABLES, DBA_TABLES.

V$FIXED_TABLE: Información de Vistas y Tablas Dinámicas

- **V$**: Prefijo de vistas dinámicas que muestran información en tiempo real.
- Para listar vistas dinámicas:sql
  Copiar código

  ```
  SELECT * FROM V$FIXED_TABLE;
  ```

- **Ejemplo**: Recuperar información de la SGA:sql
  Copiar código

  ```
  SELECT * FROM V$SGA;
  ```

Parámetros de la Base de Datos

- Configuran el comportamiento de la base de datos.
- Se almacenan en el archivo de parámetros:
- **SPFILE (Server Parameter File)**: Archivo binario usado por la instancia al arrancar.
- **PFILE (Parameter File)**: Archivo de texto alternativo.

Niveles de Parámetros:

1. **Sesión**: Cambios aplican solo a la sesión actual.
2. **Sistema**: Cambios aplican a toda la instancia.

Cambiar Parámetros

- **Nivel de Sesión**:
  sql
  Copiar código

  ```
  ALTER SESSION SET parametro = valor;
  ```

  Ejemplo:
  sql
  Copiar código

  ```
  ALTER SESSION SET SORT_AREA_SIZE = 1048576;
  ```

- **Nivel de Sistema**:
  sql
  Copiar código

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

- *MEMORY*: Solo en la memoria de la instancia actual.
- *SPFILE*: Se guarda para el siguiente arranque.
- *BOTH*: Aplica al momento y se guarda.

Ejemplo:

sql

Copiar código

```
ALTER SYSTEM SET LOG_BUFFER = 1048576 SCOPE = BOTH;
```

Tema V: Conexiones y Redes en Oracle - Listener

Introducción a las Conexiones y Redes en Oracle

- Oracle usa conexiones cliente-servidor para interactuar con bases de datos.

- **Conexión**: El cliente envía solicitudes al servidor, que las procesa y devuelve resultados.
- **Red**: Permite que clientes remotos se conecten a bases de datos Oracle usando protocolos de red como TCP/IP.

Conexiones locales y servicios

- **Conexión local**: Cliente y base de datos están en el mismo servidor. Usa ORACLE_SID para identificar la instancia.
- **Conexión de servicio**: Cliente y base de datos están en diferentes servidores. Usa un servicio definido en el archivo TNSNAMES.ORA para conectarse.

Formas de conexión

1. **Conexión local**:

- Se configura en el entorno usando ORACLE_HOME y ORACLE_SID.
- Ejemplo:bash
  Copiar código

  ```
  sqlplus / as sysdba
  ```

2. **Conexión remota**:

- Usa un servicio definido en TNSNAMES.ORA.
- Ejemplo:bash
  Copiar código

  ```
  sqlplus usuario/contraseña@servicio
  ```

LISTENER: Ficheros de Configuración

- **Listener**: Proceso que escucha solicitudes de conexión en un puerto específico (por defecto, 1521).
- Archivos principales:
- LISTENER.ORA: Configura el listener.
- TNSNAMES.ORA: Define servicios de red para clientes.
- SQLNET.ORA: Configura parámetros generales de red.

Crear un Listener

1. Modificar el archivo LISTENER.ORA (ubicado en $ORACLE_HOME/network/admin).

- Ejemplo básico:plaintext
  Copiar código

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

2. Iniciar el listener con lsnrctl:
   bash
   Copiar código

   ```
   lsnrctl start
   ```

Utilidad LSNRCTL

- Herramienta de línea de comandos para administrar el listener.
- Comandos comunes:
- lsnrctl start: Inicia el listener.
- lsnrctl stop: Detiene el listener.
- lsnrctl status: Muestra el estado actual del listener.
- lsnrctl reload: Recarga la configuración sin detener el listener.

Crear SQLNET y TNSNAMES

1. **SQLNET.ORA**:

- Configura el protocolo de red y las políticas de autenticación.
- Ejemplo:plaintext
  Copiar código

  ```
  SQLNET.AUTHENTICATION_SERVICES = (NTS)
  ```

2. **TNSNAMES.ORA**:

- Define servicios para conexiones remotas.
- Ejemplo:plaintext
  Copiar código

  SERVICIO_DB = (DESCRIPTION = (ADDRESS = (PROTOCOL = TCP) (HOST = hostname)(PORT = 1521)) (CONNECT_DATA = (SERVICE_NAME = nombre_base_de_datos)))

NETMGR: Otra Utilidad de Configuración de Red

- **Network Manager (NETMGR)**: Herramienta gráfica para configurar LISTENER.ORA, TNSNAMES.ORA, y SQLNET.ORA.
- Ventajas:
- Simplifica la edición de archivos de configuración.
- Útil para administradores menos familiarizados con archivos manuales.

Pasos básicos:

1. Abrir netmgr.
2. Configurar un nuevo listener o servicio en la base de datos.
3. Guardar los cambios, que se aplicarán a los archivos relevantes.

Tema VI: SQL Developer - Herramienta Gráfica para Administración

Introducción a SQL Developer

- **SQL Developer** es una herramienta gráfica gratuita de Oracle que permite interactuar con bases de datos de forma sencilla.
- Diseñado para desarrolladores y administradores, ofrece funcionalidades para:
- Ejecutar comandos SQL y PL/SQL.
- Administrar bases de datos y objetos (tablas, índices, vistas, etc.).
- Migrar datos entre bases de datos.

- Realizar tareas administrativas como monitoreo y configuración.

Características Principales

1. **Interfaz amigable**:
- Acceso rápido a esquemas, tablas, vistas, procedimientos almacenados y más.
- Paneles gráficos para explorar objetos de la base de datos.
2. **Ejecución de comandos SQL y PL/SQL**:
- Editor avanzado con resaltado de sintaxis y autocompletado.
- Ventana de salida para ver resultados y mensajes de error.
3. **Administración de Bases de Datos**:
- Creación y gestión de usuarios y roles.
- Configuración de privilegios.
- Visualización y modificación de parámetros de la base de datos.
4. **Exportación e importación de datos**:
- Exportar tablas a formatos como CSV, Excel, XML.
- Importar datos desde archivos externos.
5. **Monitoreo**:
- Paneles para visualizar el rendimiento de la base de datos.
- Análisis de sesiones activas y consultas lentas.

Instalación y Configuración

1. Descargar desde el sitio oficial de Oracle.
2. Requisitos:
- Java Development Kit (JDK) 8 o superior.
- Configurar la conexión con la base de datos, proporcionando:
    - *Host* y *Puerto*.
    - *SID* o *Service Name*.
    - Usuario y contraseña.

Usos Comunes como Administrador

1. **Conexión a la base de datos**:
- Crear una nueva conexión usando las credenciales de administrador (e.g., SYS o SYSTEM).
- Guardar conexiones para acceder rápidamente en el futuro.
2. **Exploración de esquemas y objetos**:
- Navegar por las tablas, vistas, índices y paquetes.
- Modificar objetos directamente desde la interfaz.
3. **Ejecución de scripts**:
- Ejecutar scripts SQL o PL/SQL desde el editor.
- Guardar y cargar scripts para reutilización.
4. **Administración de usuarios y privilegios**:
- Crear y modificar usuarios.
- Asignar roles y permisos.
5. **Gestión de rendimiento**:
- Visualizar estadísticas de uso de recursos.
- Identificar bloqueos y consultas que consumen recursos.

Ventajas

- Reducción de la complejidad en tareas administrativas.
- Mayor productividad gracias a su interfaz gráfica.
- Compatible con bases de datos Oracle locales y en la nube.

Tema VI: SQL Developer - Herramienta Gráfica para Trabajar como Administrador

Introducción a SQL Developer

- **SQL Developer** es una herramienta gráfica gratuita proporcionada por Oracle para interactuar con bases de datos de manera sencilla y eficiente.
- Diseñada para administradores y desarrolladores, ofrece:
    - Ejecución de consultas SQL y PL/SQL.
    - Administración de objetos de base de datos.
    - Gestión de usuarios y roles.
    - Migración y exportación de datos.

Funciones Principales

1. **Interfaz gráfica intuitiva**:
    - Navegador de base de datos para explorar esquemas, tablas, vistas y otros objetos.
    - Editor de consultas con resaltado de sintaxis y autocompletado.
2. **Administración de bases de datos**:
    - Creación y modificación de tablas, índices y vistas.
    - Administración de usuarios, roles y privilegios.
    - Configuración de parámetros y monitoreo de sesiones.
3. **Ejecución de comandos SQL y PL/SQL**:
    - Ventana de entrada para escribir y ejecutar comandos.
    - Soporte para scripts complejos.
4. **Exportación e importación de datos**:
    - Exportación a formatos como CSV, Excel, XML, etc.

- Importación de datos desde archivos externos.

5. **Monitoreo y gestión de rendimiento**:
   - Análisis de consultas y uso de recursos.
   - Identificación de sesiones activas y consultas costosas.

Instalación y Configuración

1. **Descargar SQL Developer**:
   - Disponible en el sitio oficial de Oracle: Descargar SQL Developer.
2. **Requisitos**:
   - Java Development Kit (JDK) 8 o superior.
   - Configuración del entorno: Conexión a base de datos (host, puerto, SID o servicio).
3. **Configuración inicial**:
   - Crear una conexión:
     - Introducir las credenciales del usuario (nombre, contraseña).
     - Especificar el nombre del host, puerto y SID o nombre del servicio.

Usos Comunes como Administrador

1. **Conexión a la base de datos**:
   - Crear conexiones a bases de datos locales o remotas.
   - Guardar conexiones para reutilización.
2. **Administración de esquemas**:
   - Navegar y modificar objetos como tablas, vistas y procedimientos almacenados.
   - Crear nuevos objetos o eliminar los existentes.
3. **Gestión de usuarios y roles**:
   - Crear usuarios y asignar privilegios.
   - Gestionar roles y perfiles de recursos.

4. **Optimización de consultas**:
    - Uso del **Explain Plan** para analizar el costo de consultas.
    - Identificación de índices faltantes.
5. **Exportación e importación de datos**:
    - Exportar datos para análisis externo o respaldos.
    - Importar datos para poblar tablas nuevas.

Ventajas de SQL Developer

- Simplifica tareas administrativas mediante una interfaz gráfica amigable.
- Reduce la necesidad de comandos manuales para configuraciones básicas.
- Permite monitoreo en tiempo real de actividades y uso de recursos.

Tema VII: MULTITENANT - Conceptos Adicionales

Repaso de la arquitectura Multitenant

- Introducida en Oracle 12c, la arquitectura multitenant permite consolidar varias bases de datos (PDBs) dentro de una base de datos contenedora (CDB).
- Componentes principales:
- **CDB (Container Database)**: Contiene estructuras comunes (archivos de control, memoria, procesos).
- **PDB (Pluggable Database)**: Bases de datos individuales alojadas dentro del contenedor.
- **Root Container**: Almacena datos y metadatos compartidos entre todas las PDBs.
- **Seed PDB**: Plantilla para crear nuevas PDBs.

Primeros pasos con una CDB: Arrancarla

1. **Arrancar la CDB**:

- Comando básico para arrancar la base de datos contenedora:sql
  Copiar código

  ```
  STARTUP;
  ```

- La CDB inicia en el estado abierto, pero las PDBs asociadas permanecen cerradas por defecto.

2. **Verificar PDBs en la CDB**:
   sql
   Copiar código

   ```
   SELECT CON_ID, NAME, OPEN_MODE FROM V$PDBS;
   ```

Abrir y cerrar PDBs

1. **Abrir una PDB**:
   sql
   Copiar código

   ```
   ALTER PLUGGABLE DATABASE nombre_pdb OPEN;
   ```

2. **Abrir todas las PDBs**:
   sql
   Copiar código

   ```
   ALTER PLUGGABLE DATABASE ALL OPEN;
   ```

   Cerrar una PDB:
   sql
   Copiar código

   ```
   ALTER PLUGGABLE DATABASE nombre_pdb CLOSE;
   ```

Arrancar PDBs de forma automática

- Configuración para abrir PDBs automáticamente al iniciar la CDB:
  sql
  Copiar código

  ```
  ALTER PLUGGABLE DATABASE nombre_pdb SAVE STATE;
  ```

- Verificar el estado guardado:
  sql

Copiar código

```
SELECT PDB_NAME, PDB_STATE FROM DBA_PDB_SAVED_STATES;
```

Conectarnos en una Multitenant

1. **Conexión a la CDB**:

- Con el usuario SYS o SYSTEM:bash
  Copiar código

```
sqlplus sys/contraseña@cdb_service_name AS SYSDBA
```

2. **Conexión a una PDB específica**:

- Indicar el servicio de la PDB:bash
  Copiar código

```
sqlplus usuario/contraseña@pdb_service_name
```

Más opciones de conexión

- **Cambiar entre CDB y PDBs** en una sesión abierta:sql
  Copiar código

```
ALTER SESSION SET CONTAINER = nombre_pdb;
```

Vistas para consultar CDBs

- **Consultar información de la CDB y sus PDBs**:
- V$PDBS: Detalles sobre las PDBs asociadas.
- CDB_TABLES, CDB_USERS: Información consolidada de objetos en todas las PDBs.
- DBA_PDBS: Detalles administrativos de las PDBs.

Ejemplo:

sql

Copiar código

```
SELECT NAME, OPEN_MODE, RESTRICTED FROM V$PDBS;
```

Crear PDBs desde DBCA

1. Iniciar el asistente de configuración (DBCA).

2. Seleccionar "Gestionar bases de datos pluggable (PDB)".
3. Configurar el nombre de la PDB, ubicación de los archivos y opciones iniciales.
4. Crear la PDB usando la **Seed PDB** como plantilla.

Crear una PDB de forma manual

1. **Conectar a la CDB** como administrador:

   bash
   Copiar código

   ```
   sqlplus sys/contraseña@cdb_service_name AS SYSDBA
   ```

2. **Crear la PDB usando la Seed**:

   sql
   Copiar código

   ```
   CREATE PLUGGABLE DATABASE nombre_pdb ADMIN USER admin IDENTIFIED BY contraseña FILE_NAME_CONVERT = ('/ruta_seed/', '/ruta_nueva_pdb/');
   ```

3. **Abrir la PDB recién creada**:

   sql
   Copiar código

   ```
   ALTER PLUGGABLE DATABASE nombre_pdb OPEN;
   ```

4. **Verificar la creación**:

   sql
   Copiar código

   ```
   SELECT NAME, OPEN_MODE FROM V$PDBS;
   ```

Tema VII: MULTITENANT - Conceptos Adicionales

Repaso de la arquitectura Multitenant

- Introducida en Oracle 12c, la arquitectura multitenant permite consolidar varias bases de datos (PDBs) dentro de una base de datos contenedora (CDB).
- Componentes principales:
- **CDB (Container Database)**: Contiene estructuras comunes (archivos de control, memoria, procesos).
- **PDB (Pluggable Database)**: Bases de datos individuales alojadas dentro del contenedor.
- **Root Container**: Almacena datos y metadatos compartidos entre todas las PDBs.
- **Seed PDB**: Plantilla para crear nuevas PDBs.

Primeros pasos con una CDB: Arrancarla

1. **Arrancar la CDB**:
- Comando básico para arrancar la base de datos contenedora:sql
  Copiar código

  ```
  STARTUP;
  ```

- La CDB inicia en el estado abierto, pero las PDBs asociadas permanecen cerradas por defecto.

2. **Verificar PDBs en la CDB**:
   sql
   Copiar código

```
SELECT CON_ID, NAME, OPEN_MODE FROM V$PDBS;
```

Abrir y cerrar PDBs

1. **Abrir una PDB**:
   sql
   Copiar código

   ```
   ALTER PLUGGABLE DATABASE nombre_pdb OPEN;
   ```

2. **Abrir todas las PDBs**:
   sql
   Copiar código

   ```
   ALTER PLUGGABLE DATABASE ALL OPEN;
   ```

3. **Cerrar una PDB**:
   sql
   Copiar código

   ```
   ALTER PLUGGABLE DATABASE nombre_pdb CLOSE;
   ```

Arrancar PDBs de forma automática

- Configuración para abrir PDBs automáticamente al iniciar la CDB:
  sql
  Copiar código

  ```
  ALTER PLUGGABLE DATABASE nombre_pdb SAVE STATE;
  ```

- Verificar el estado guardado:
  sql
  Copiar código

  ```
  SELECT PDB_NAME, PDB_STATE FROM DBA_PDB_SAVED_STATES;
  ```

Conectarnos en una Multitenant

1. **Conexión a la CDB**:

- Con el usuario SYS o SYSTEM:bash
  Copiar código

  ```
  sqlplus sys/contraseña@cdb_service_name AS SYSDBA
  ```

2. **Conexión a una PDB específica**:

- Indicar el servicio de la PDB:bash
  Copiar código

  ```
  sqlplus usuario/contraseña@pdb_service_name
  ```

Más opciones de conexión

- **Cambiar entre CDB y PDBs** en una sesión abierta:sql
  Copiar código

  ```
  ALTER SESSION SET CONTAINER = nombre_pdb;
  ```

Vistas para consultar CDBs

- **Consultar información de la CDB y sus PDBs**:
- V$PDBS: Detalles sobre las PDBs asociadas.
- CDB_TABLES, CDB_USERS: Información consolidada de objetos en todas las PDBs.
- DBA_PDBS: Detalles administrativos de las PDBs.

Ejemplo:

sql

Copiar código

```
SELECT NAME, OPEN_MODE, RESTRICTED FROM V$PDBS;
```

Crear PDBs desde DBCA

1. Iniciar el asistente de configuración (DBCA).

2. Seleccionar "Gestionar bases de datos pluggable (PDB)".
3. Configurar el nombre de la PDB, ubicación de los archivos y opciones iniciales.
4. Crear la PDB usando la **Seed PDB** como plantilla.

Crear una PDB de forma manual

1. **Conectar a la CDB** como administrador:

   bash
   Copiar código

   ```
   sqlplus sys/contraseña@cdb_service_name AS SYSDBA
   ```

2. **Crear la PDB usando la Seed**:

   sql
   Copiar código

   ```
   CREATE PLUGGABLE DATABASE nombre_pdb ADMIN USER admin IDENTIFIED BY contraseña FILE_NAME_CONVERT = ('/ruta_seed/', '/ruta_nueva_pdb/');
   ```

3. **Abrir la PDB recién creada**:

   sql
   Copiar código

   ```
   ALTER PLUGGABLE DATABASE nombre_pdb OPEN;
   ```

4. **Verificar la creación**:

   sql
   Copiar código

   ```
   SELECT NAME, OPEN_MODE FROM V$PDBS;
   ```

Tema VIII: Tablespaces y Gestión del Almacenamiento

Introducción al Almacenamiento

- Un **tablespace** es la unidad lógica de almacenamiento en una base de datos Oracle.
- Contiene uno o más **archivos de datos** físicos donde se almacenan objetos como tablas e índices.
- Tipos comunes:
- **Tablespace permanente**: Para datos persistentes.
- **Tablespace temporal**: Para operaciones transitorias como sorting.
- **Tablespace de deshacer**: Para gestionar rollback.

Vistas para ver Tablespaces

- **Vistas comunes** para consultar información de tablespaces:
- DBA_TABLESPACES: Información general.
- DBA_DATA_FILES: Archivos asociados.
- V$TABLESPACE: Estado de los tablespaces en tiempo real.

Ejemplo:

sql

Copiar código

```
SELECT TABLESPACE_NAME, STATUS, CONTENTS FROM DBA_TABLESPACES;
```

Crear una Tablespace

1. **Sintaxis básica**:
   sql
   Copiar código

   ```
   CREATE TABLESPACE nombre_tablespace DATAFILE '/ruta/archivo.dbf' SIZE 100M AUTOEXTEND ON NEXT 10M MAXSIZE UNLIMITED;
   ```

2. **Tablespace por defecto**:
   - Al crear usuarios, se les asigna un tablespace predeterminado:sql
     Copiar código

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

Tablespaces en la Multitenant

- Cada **PDB** tiene su propio conjunto de tablespaces, pero hereda configuraciones del contenedor (CDB).
- Consultar tablespaces de una PDB:sql
  Copiar código

  ```
  SELECT TABLESPACE_NAME FROM CDB_TABLESPACES WHERE CON_ID = (ID_DE_PDB);
  ```

Gestión de Extensiones

1. **AUTOALLOCATE**:
   - Oracle gestiona automáticamente el tamaño de los segmentos.
   - Ideal para bases de datos con tamaños de objetos variables.
2. sql
   Copiar código

   ```
   CREATE TABLESPACE nombre AUTOALLOCATE;
   ```

3. **UNIFORM**:
   - Asigna bloques del mismo tamaño.
   - Mejor para bases de datos con objetos similares.
4. sql
   Copiar código

   ```
   CREATE TABLESPACE nombre UNIFORM SIZE 1M;
   ```

Añadir ficheros a un Tablespace

- Permite expandir el tamaño del tablespace agregando más archivos:sql
  Copiar código

  ```
  ALTER TABLESPACE nombre ADD DATAFILE '/ruta/nuevo_archivo.dbf' SIZE 100M;
  ```

Cambiar el Estado de un Tablespace

- **OFFLINE**: No está disponible para uso:sql
  Copiar código

  ```
  ALTER TABLESPACE nombre OFFLINE;
  ```

- **READ_ONLY**: Disponible solo para consultas:sql
  Copiar código

  ```
  ALTER TABLESPACE nombre READ ONLY;
  ```

Autoextender Ficheros

- Activar autoextensión para un archivo de datos:sql
  Copiar código

  ```
  ALTER DATABASE DATAFILE '/ruta/archivo.dbf' AUTOEXTEND ON NEXT 10M MAXSIZE UNLIMITED;
  ```

Cambiar Manualmente el Tamaño de un Fichero

- Incrementar el tamaño de un archivo:sql
  Copiar código

  ```
  ALTER DATABASE DATAFILE '/ruta/archivo.dbf' RESIZE 200M;
  ```

Mover y Renombrar Ficheros

- Cambiar ubicación de un archivo:

5. Mover el archivo al nuevo destino y renombrarlo:sql
   Copiar código

```sql
ALTER DATABASE RENAME FILE '/ruta/archivo_viejo.dbf' TO '/ruta/archivo_nuevo.dbf';
```

Hacer online nuevamente:sql
Copiar código

```sql
ALTER TABLESPACE nombre ONLINE;
```

Tablespaces Temporales

- Usados para operaciones temporales como ordenamiento y unión.
- Consultar tablespaces temporales:sql
  Copiar código

  ```sql
  SELECT TABLESPACE_NAME FROM DBA_TABLESPACES WHERE CONTENTS = 'TEMPORARY';
  ```

Trabajar con tablespaces temporales:

- Crear:sql
  Copiar código

  ```sql
  CREATE TEMPORARY TABLESPACE temp_tbs TEMPFILE '/ruta/tempfile.dbf' SIZE 50M;
  ```

- Asignar a un usuario:sql
  Copiar código

  ```sql
  ALTER USER nombre TEMPORARY TABLESPACE temp_tbs;
  ```

Borrar una Tablespace

- Eliminar un tablespace junto con sus archivos:sql
  Copiar código

  ```sql
  DROP TABLESPACE nombre INCLUDING CONTENTS AND DATAFILES;
  ```

Uso de Enterprise Manager con Tablespaces

- **Enterprise Manager** proporciona una interfaz gráfica para:
- Crear, modificar y eliminar tablespaces.

- Monitorear el uso de espacio y rendimiento.
- Configurar opciones de autoextensión y agregar archivos de datos.

Tema IX: Ficheros de Control

Introducción a los Ficheros de Control

- Los **ficheros de control** son componentes esenciales en una base de datos Oracle.
- Contienen información clave sobre la estructura y el estado de la base de datos:
- Identificadores de la base de datos.
- Ubicación de los archivos de datos y redo logs.
- Información sobre puntos de recuperación (checkpoints).
- Registros de archivos archivados.

Características principales:

- Se actualizan constantemente durante el funcionamiento de la base de datos.
- Una base de datos debe tener al menos un fichero de control, pero se recomienda mantener múltiples copias en ubicaciones distintas para mayor redundancia.

Comprobar los Ficheros de Control

- Consultar los ficheros de control configurados:

  sql
  Copiar código

  ```
  SELECT NAME FROM V$CONTROLFILE;
  ```

- Verificar el estado de los ficheros:

  sql
  Copiar código

  ```
  SELECT STATUS FROM V$CONTROLFILE_RECORD_SECTION;
  ```

Crear un init.ora **desde un** SPFILE

1. **Extraer parámetros de un SPFILE**:

- Usar el comando:sql
  Copiar código

  ```
  CREATE PFILE='/ruta/init.ora' FROM SPFILE;
  ```

2. **Modificar el archivo** init.ora **según sea necesario**:

- Ajustar rutas, tamaños, etc.

3. **Arrancar la base de datos con el** init.ora:

- Asegurarse de que la base de datos esté cerrada:sql
  Copiar código

  ```
  SHUTDOWN IMMEDIATE;
  ```

- Arrancar la base de datos usando el PFILE:sql
  Copiar código

  ```
  STARTUP PFILE='/ruta/init.ora';
  ```

Añadir un ControlFile y Arrancar

1. **Editar el PFILE o SPFILE para incluir un nuevo fichero de control**:

- En el init.ora:plaintext
  Copiar código

  ```
  CONTROL_FILES = ('/ruta/control01.ctl', '/ruta/nuevo_control.ctl')
  ```

- Para un SPFILE:sql
  Copiar código

  ```
  ALTER SYSTEM SET CONTROL_FILES='/ruta/control01.ctl', '/ruta/nuevo_control.ctl' SCOPE=SPFILE;
  ```

2. **Crear el nuevo fichero de control como una copia de uno existente**:
   bash
   Copiar código

```
cp /ruta/control01.ctl /ruta/nuevo_control.ctl
```

3. **Arrancar la base de datos**:
   sql
   Copiar código

   ```
   STARTUP;
   ```

Crear un SPFILE **desde un** init.ora **y Arrancar la Base de Datos**

1. **Convertir un** init.ora **en un** SPFILE:
   sql
   Copiar código

   ```
   CREATE SPFILE='/ruta/spfile.ora' FROM PFILE='/ruta/init.ora';
   ```

2. **Arrancar la base de datos usando el nuevo SPFILE**:

- Apagar la base de datos si está en ejecución:sql
  Copiar código

  ```
  SHUTDOWN IMMEDIATE;
  ```

- Arrancar de nuevo:sql
  Copiar código

  ```
  STARTUP;
  ```

Enterprise Manager y Ficheros de Control

- **Enterprise Manager** proporciona una interfaz gráfica para:
- Visualizar los ficheros de control configurados.
- Monitorear su estado.
- Agregar o modificar ubicaciones de ficheros de control.
- Para realizar cambios, acceder al menú de configuración de almacenamiento en la consola de Enterprise Manager.

Tema X: UNDO

Introducción a UNDO

- **UNDO** es un mecanismo en Oracle que gestiona los cambios realizados en una transacción antes de que sean confirmados (COMMIT).
- Proporciona:
- **Consistencia en lectura**: Permite que los usuarios vean datos coherentes durante una transacción.
- **Recuperación de transacciones**: Facilita el rollback en caso de fallos o cancelación de operaciones.

Consistencia en Lectura

- Oracle utiliza segmentos de UNDO para mantener una copia consistente de los datos que están siendo modificados.

- Ejemplo:
- Un usuario modifica una fila, pero otro usuario la consulta antes del COMMIT. Oracle usa UNDO para mostrar la versión previa de la fila al segundo usuario.

Parámetros de UNDO y Tablespaces

1. **Parámetros clave**:

- UNDO_TABLESPACE: Define el tablespace usado para segmentos de UNDO.
- UNDO_RETENTION: Tiempo (en segundos) que Oracle retiene información UNDO para consultas de consistencia en lectura y recuperación.

2. **Configurar parámetros**:
   sql
   Copiar código

   ```
   ALTER SYSTEM SET UNDO_TABLESPACE = nombre_undo_tablespace;
   ALTER SYSTEM SET UNDO_RETENTION = 900;
   ```

Ver Transacciones

- Consultar las transacciones activas que usan UNDO:sql
  Copiar código

  ```
  SELECT * FROM V$TRANSACTION;
  ```

Crear Tablespace de UNDO y Convertirla en la Predeterminada

1. **Crear un tablespace de UNDO**:
   sql
   Copiar código

   ```
   CREATE UNDO TABLESPACE undo_tbs DATAFILE '/ruta/undo_tbs.dbf' SIZE 500M;
   ```

2. **Hacerlo el tablespace de UNDO por defecto**:
   sql
   Copiar código

   ```
   ALTER SYSTEM SET UNDO_TABLESPACE = undo_tbs;
   ```

Periodo de Retención de UNDO (UNDO_RETENTION)

- Especifica cuánto tiempo se retiene la información UNDO después del COMMIT.
- Configurar el tiempo de retención:sql
  Copiar código

  ```
  ALTER SYSTEM SET UNDO_RETENTION = 1200;
  ```

- Consultar el valor actual:sql
  Copiar código

```
SHOW PARAMETER UNDO_RETENTION;
```

Vistas para Consultar UNDO

- **Vistas importantes**:
- DBA_UNDO_EXTENTS: Información sobre extensiones UNDO.
- V$UNDOSTAT: Estadísticas de uso y retención de UNDO.
- DBA_TABLESPACES: Detalles sobre los tablespaces de UNDO.

Ejemplo:

sql

Copiar código

```
SELECT BEGIN_TIME, END_TIME, UNDOTSN, UNDOBLKS FROM V$UNDOSTAT;
```

Borrar Tablespace UNDO

1. **Mover transacciones activas a otro tablespace**:
- Configurar un nuevo tablespace de UNDO por defecto:sql
  Copiar código

  ```
  ALTER SYSTEM SET UNDO_TABLESPACE = nuevo_undo_tbs;
  ```

2. **Eliminar el tablespace antiguo**:
   sql
   Copiar código

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

Enterprise Manager y UNDO

- **Enterprise Manager** simplifica la administración de UNDO:
- Monitoriza el uso de UNDO en tiempo real.
- Configura el tablespace predeterminado y el periodo de retención.
- Optimiza el tamaño y la extensión de UNDO.

Tema XI: REDO LOGS

Introducción a los Redo Logs

- **Redo Logs** son un componente clave en Oracle para garantizar la recuperación de datos en caso de fallos.
- **Función principal**:
- Registrar todas las transacciones (modificaciones de datos) realizadas en la base de datos, ya sean confirmadas (COMMIT) o no.
- **Estructura**:
- Conjunto de **grupos de redo logs** que contienen uno o más **miembros** (copias idénticas del grupo para redundancia).
- Los Redo Logs permiten a Oracle restaurar los cambios en caso de fallos inesperados.

Ver el Funcionamiento de los Redo Logs

1. **Consultar los grupos de redo logs**:

sql
Copiar código

```
SELECT GROUP#, STATUS, MEMBERS, BYTES FROM V$LOG;
```

2. **Consultar los miembros de redo logs**:

sql
Copiar código

```
SELECT GROUP#, MEMBER FROM V$LOGFILE;
```

3. **Estados comunes de los grupos**:

- *CURRENT*: Grupo activo que está registrando cambios.
- *ACTIVE*: Cambios guardados en disco pero necesarios para recuperación.
- *INACTIVE*: Cambios ya archivados, no necesarios para recuperación inmediata.

Añadir un Miembro de Redo Log

- Para aumentar la redundancia de un grupo:sql
  Copiar código

  ```
  ALTER DATABASE ADD LOGFILE MEMBER '/ruta/nuevo_redo01.log' TO GROUP 1;
  ```

Añadir un Grupo de Redo Log

- Para mejorar la capacidad de registro y rendimiento:sql
  Copiar código

  ```
  ALTER DATABASE ADD LOGFILE GROUP 4 ('/ruta/redo04a.log', '/ruta/redo04b.log') SIZE 50M;
  ```

Borrar un Miembro de un Grupo de Redo Log

1. **Comprobar que el miembro no está en uso**:
- Consultar el estado del grupo:sql
  Copiar código

  ```
  SELECT GROUP#, STATUS FROM V$LOG;
  ```

2. **Eliminar un miembro del grupo**:
   sql
   Copiar código

   ```
   ALTER DATABASE DROP LOGFILE MEMBER '/ruta/miembro_a_eliminar.log';
   ```

Borrar un Grupo de Redo Log

1. **Asegurarse de que el grupo no está activo**:
- Cambiar el grupo activo si es necesario:sql
  Copiar código

  ```
  ALTER SYSTEM SWITCH LOGFILE;
  ```

2. **Eliminar el grupo**:
   sql
   Copiar código

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

Enterprise Manager y Redo Logs

- **Enterprise Manager** permite:
- Visualizar el estado y tamaño de los redo logs.
- Añadir o eliminar grupos y miembros mediante una interfaz gráfica.
- Monitorear la actividad y el uso de los redo logs en tiempo real.

Tema XII: ARCHIVELOG

Introducción a Archivelog

- **Modo ARCHIVELOG** en Oracle permite que los redo logs se almacenen en formato de archivo una vez llenos.
- **Función principal**:
- Facilitar la recuperación completa de datos en caso de fallos, incluyendo datos perdidos desde el último backup.
- **Ventaja clave**:
- Permite respaldos en línea, sin necesidad de apagar la base de datos.

Propiedades de los Archivelog

- Al habilitar ARCHIVELOG:
- Los redo logs llenos se copian a un destino de archivo (archivo de log archivado).

- Se conserva el historial de todas las transacciones de la base de datos.
- **Vista para consultar el estado del modo**:sql
  Copiar código

  ```
  SELECT LOG_MODE FROM V$DATABASE;
  ```

- *ARCHIVELOG*: El modo está habilitado.
- *NOARCHIVELOG*: El modo está deshabilitado.

Configurar Destinos y Formatos

1. **Configurar el destino donde se almacenarán los archivos de log**:

- Cambiar el destino:sql
  Copiar código

  ```
  ALTER SYSTEM SET LOG_ARCHIVE_DEST_1='LOCATION=/ruta/archivelogs/';
  ```

  Configurar múltiples destinos (opcional):sql
  Copiar código

  ```
  ALTER SYSTEM SET LOG_ARCHIVE_DEST_2='SERVICE=standby_db';
  ```

2. **Configurar el formato del archivo**:

- Definir formato de nombres:sql
  Copiar código

  ```
  ALTER SYSTEM SET LOG_ARCHIVE_FORMAT='log_%t_%s_%r.arc';
  ```

  %t: Número de thread.
  - %s: Número de secuencia.
  - %r: ID de restauración de la base de datos.

Cambiar la Base de Datos a ARCHIVELOG

1. **Preparar la base de datos**:

- Asegurarse de que la base de datos esté en modo MOUNT:sql
  Copiar código

```
SHUTDOWN IMMEDIATE; STARTUP MOUNT;
```

2. **Habilitar el modo ARCHIVELOG**:
   sql
   Copiar código

   ```
   ALTER DATABASE ARCHIVELOG;
   ```

3. **Abrir la base de datos**:
   sql
   Copiar código

   ```
   ALTER DATABASE OPEN;
   ```

4. **Verificar el cambio**:
   sql
   Copiar código

   ```
   ARCHIVE LOG LIST;
   ```

Enterprise Manager y Archivelog

- **Enterprise Manager** simplifica la configuración del modo ARCHIVELOG:
- Configura destinos y formatos de manera gráfica.
- Muestra el estado actual del modo.
- Permite habilitar o deshabilitar ARCHIVELOG desde el menú de configuración.

Tema XIII: Procesos en Oracle

Introducción a los Procesos en Oracle

- Los procesos son programas en ejecución que permiten la interacción entre el cliente y la base de datos Oracle.
- Se dividen en dos tipos principales:

1. **Procesos de usuario**:
    - Representan las solicitudes de los clientes (consultas SQL, PL/SQL, etc.).
2. **Procesos de servidor**:
    - Manejan las solicitudes de los usuarios interactuando con la base de datos.

- **Modos de configuración de procesos de servidor**:

1. **Servidor Dedicado**: Cada usuario tiene un proceso de servidor exclusivo.
2. **Servidor Compartido**: Un conjunto de procesos de servidor se comparte entre múltiples usuarios.

Repaso de Procesos Dedicados

- En el **modo dedicado**, cada conexión de cliente recibe un proceso de servidor único.
- **Ventajas**:
- Mejor rendimiento para usuarios individuales.
- Ideal para sistemas con pocas conexiones simultáneas.
- **Desventajas**:
- Requiere más recursos en sistemas con muchas conexiones.
- Ejemplo de conexión en modo dedicado:bash
  Copiar código

```
sqlplus usuario/contraseña@servidor
```

Configurar Servidores Compartidos

- En el **modo compartido**, múltiples conexiones de usuario utilizan un grupo de procesos de servidor.
- **Ventajas**:
- Reduce el uso de recursos al compartir procesos.
- Escalable para sistemas con muchas conexiones concurrentes.
- **Desventajas**:
- Latencia adicional debido al intercambio de procesos.

3. **Configurar parámetros en el archivo de inicialización**:

- Activar servidores compartidos:sql
  Copiar código

```
ALTER SYSTEM SET SHARED_SERVERS = 5;
```

- Definir procesos máximos:sql
  Copiar código

```
ALTER SYSTEM SET MAX_SHARED_SERVERS = 20;
```

- Configurar colas de despacho:sql
  Copiar código

```
ALTER SYSTEM SET DISPATCHERS = '(PROTOCOL=TCP)
(DISPATCHERS=2)';
```

2. **Verificar configuración de servidores compartidos**:

- Consultar la vista V$SHARED_SERVER:sql
  Copiar código

```
SELECT SERVER, STATUS FROM V$SHARED_SERVER;
```

Modificar dinámicamente la cantidad de servidores compartidos:
sql
Copiar código

```
ALTER SYSTEM SET SHARED_SERVERS = 10;
```

Tema XIV: Usuarios y Privilegios

Introducción a la Seguridad

- La seguridad en Oracle se basa en:
- **Usuarios**: Identidades que interactúan con la base de datos.
- **Privilegios**: Permisos que determinan las acciones que un usuario puede realizar.
- **Roles**: Conjuntos de privilegios asignados a usuarios.

Usuarios Creados por Defecto

- Oracle crea usuarios predeterminados al instalar la base de datos, como:
- SYS y SYSTEM: Administradores.
- HR, SCOTT, etc.: Usuarios de ejemplo.
- Consultar usuarios predeterminados:sql
  Copiar código

```sql
SELECT USERNAME FROM DBA_USERS;
```

Crear un Usuario

1. **Sintaxis básica**:sql
   Copiar código

```sql
CREATE USER nombre IDENTIFIED BY contraseña;
```

2. **Ejemplo**:sql
   Copiar código

```sql
CREATE USER juan IDENTIFIED BY 12345;
```

Usuarios en Multitenant

1. **Usuarios comunes**:
- Existen en el contenedor raíz (CDB) y en todas las PDBs.
- Se crean usando el prefijo C##:sql
  Copiar código

```sql
CREATE USER C##admin IDENTIFIED BY admin123 CONTAINER=ALL;
```

2. **Usuarios locales**:
- Existen solo dentro de una PDB específica.
- Crear usuario local en una PDB:sql
  Copiar código

```
CREATE USER local_user IDENTIFIED BY localpass;
```

3. **Usuarios globales**:
- Autenticados mediante servicios externos (LDAP).
- Ejemplo:sql
  Copiar código

```
CREATE USER global_user IDENTIFIED GLOBALLY AS 'CN=usuario,OU=unidad,DC=dominio';
```

Modificar un Usuario

- Cambiar contraseña:
  sql
  Copiar código

```
ALTER USER nombre IDENTIFIED BY nueva_contraseña;
```

- Cambiar parámetros (como límite de espacio):
  sql
  Copiar código

```
ALTER USER nombre QUOTA 100M ON tablespace_nombre;
```

Reservar Espacio para los Usuarios

- Asignar espacio en un tablespace:sql
  Copiar código

```
ALTER USER nombre QUOTA 100M ON nombre_tablespace;
```

Estado de los Usuarios

- Bloquear cuentas:
  sql
  Copiar código

  ```
  ALTER USER nombre ACCOUNT LOCK;
  ```

- Desbloquear cuentas:
  sql
  Copiar código

  ```
  ALTER USER nombre ACCOUNT UNLOCK;
  ```

  Expiración de contraseñas:
  sql
  Copiar código

  ```
  ALTER PROFILE DEFAULT LIMIT PASSWORD_LIFE_TIME 90;
  ```

Borrar un Usuario

- Eliminar usuario y sus objetos:sql
  Copiar código

  ```
  DROP USER nombre CASCADE;
  ```

Privilegios: Introducción

- Los privilegios se dividen en:
- **Privilegios de sistema**: Permiten acciones administrativas (crear tablas, sesiones, etc.).
- **Privilegios de objeto**: Permiten operaciones sobre objetos específicos (tablas, vistas, etc.).

Dar Privilegios de Sistema

- Sintaxis básica:sql
  Copiar código

  ```
  GRANT privilegio TO usuario;
  ```

- Ejemplo:sql
  Copiar código
  ```
  GRANT CREATE SESSION TO juan;
  ```

Heredar Privilegios

- Usar roles para agrupar privilegios:sql
  Copiar código

  ```
  CREATE ROLE manager_role; GRANT CREATE TABLE, CREATE VIEW TO manager_role; GRANT manager_role TO juan;
  ```

Eliminar Privilegios de Sistema

- Revocar privilegios:sql
  Copiar código

  ```
  REVOKE privilegio FROM usuario;
  ```

- Ejemplo:sql
  Copiar código

  ```
  REVOKE CREATE SESSION FROM juan;
  ```

Dar Privilegios de Objeto

- Permitir acceso a un objeto:sql
  Copiar código

  ```
  GRANT SELECT, INSERT ON tabla TO usuario;
  ```

Dar Privilegios sobre Columnas

- Especificar columnas al conceder privilegios:sql
  Copiar código

  ```
  GRANT SELECT (columna1, columna2) ON tabla TO usuario;
  ```

Heredar Privilegios de Objeto

- Usar un rol para agrupar privilegios de objeto y asignarlo a un usuario:sql
  Copiar código

  ```
  GRANT SELECT ON tabla TO rol; GRANT rol TO usuario;
  ```

Quitar Privilegios de Objeto

- Revocar privilegios de un objeto:sql
  Copiar código

  ```
  REVOKE SELECT ON tabla FROM usuario;
  ```

Vistas para Visualizar Privilegios

- Consultar privilegios otorgados:
- DBA_SYS_PRIVS: Privilegios de sistema.
- DBA_TAB_PRIVS: Privilegios de objetos.
- USER_ROLE_PRIVS: Roles asignados al usuario.
- Ejemplo:sql
  Copiar código

  ```
  SELECT * FROM DBA_SYS_PRIVS WHERE GRANTEE = 'JUAN';
  ```

Usar Enterprise Manager con Usuarios y Roles

- **Enterprise Manager** facilita:
- Crear, modificar y eliminar usuarios.
- Asignar roles y privilegios mediante una interfaz gráfica.
- Monitorear el uso de recursos por usuario.

Tema XV: Roles

Introducción a los Roles

- **Roles** en Oracle son conjuntos de privilegios agrupados que facilitan la gestión de permisos para usuarios.
- Beneficios:
- Simplifican la asignación y revocación de permisos.
- Mejoran la administración de seguridad en bases de datos con muchos usuarios.

Roles Predefinidos

- Oracle proporciona roles predefinidos con permisos comunes:
- **CONNECT**: Permite iniciar sesiones en la base de datos.
- **RESOURCE**: Permite crear objetos como tablas e índices.
- **DBA**: Privilegios administrativos completos.
- **SELECT_CATALOG_ROLE**: Permite acceder a vistas del diccionario de datos.

Ejemplo para consultar roles predefinidos:

sql

Copiar código

```
SELECT * FROM DBA_ROLES;
```

Crear un Rol

1. **Sintaxis básica**:
   sql
   Copiar código

   ```
   CREATE ROLE nombre_rol;
   ```

2. **Ejemplo**:
   sql
   Copiar código

   ```
   CREATE ROLE admin_role;
   ```

Crear un Rol Global en Multitenant

- Un rol global se autentica mediante un servicio externo (como LDAP).
- Ejemplo:sql
  Copiar código

  ```
  CREATE ROLE global_role IDENTIFIED GLOBALLY;
  ```

Otorgar y Quitar Permisos a un Rol

1. **Otorgar privilegios**:
   sql
   Copiar código

   ```
   GRANT CREATE SESSION, CREATE TABLE TO admin_role;
   ```

2. **Revocar privilegios**:
   sql
   Copiar código

   ```
   REVOKE CREATE TABLE FROM admin_role;
   ```

Asignar un Rol a un Usuario

1. **Otorgar un rol**:
   sql
   Copiar código

   ```
   GRANT admin_role TO juan;
   ```

2. **Revocar un rol**:
   sql
   Copiar código

   ```
   REVOKE admin_role FROM juan;
   ```

Vistas para Ver Información de los Roles

- Vistas clave:
- DBA_ROLES: Lista de roles existentes.
- DBA_ROLE_PRIVS: Roles asignados a usuarios.
- ROLE_TAB_PRIVS: Privilegios de objeto asociados a roles.

Ejemplo:

sql

Copiar código

```
SELECT * FROM DBA_ROLE_PRIVS WHERE GRANTEE = 'JUAN';
```

Borrar un Rol

- Eliminar un rol de la base de datos:sql
  Copiar código

  ```
  DROP ROLE admin_role;
  ```

Enterprise Manager para Roles

- **Enterprise Manager** permite:
- Crear, modificar y eliminar roles.
- Asignar y revocar privilegios y roles a usuarios.
- Monitorear el uso de roles y privilegios asignados.

Tema XVI: Profiles

Introducción a los Profiles

- **Profiles** son herramientas de Oracle para controlar y limitar el uso de recursos y definir políticas de contraseñas para usuarios.
- Objetivos principales:
- Gestionar recursos del sistema (CPU, sesiones, etc.).
- Aplicar políticas de seguridad de contraseñas.

Perfil Default

- Oracle crea un perfil predeterminado llamado **DEFAULT**.
- Todos los usuarios sin un perfil específico utilizan este.
- El perfil DEFAULT se puede modificar, pero no eliminar.

Ejemplo para consultar el perfil DEFAULT:

sql

Copiar código

```
SELECT * FROM DBA_PROFILES WHERE PROFILE = 'DEFAULT';
```

Crear y Modificar un Perfil (Recursos de Kernel)

1. **Crear un perfil**:
   sql
   Copiar código

```
CREATE PROFILE nombre_perfil LIMIT SESSIONS_PER_USER 5
```

```
CPU_PER_SESSION 10000 CPU_PER_CALL 3000;
```

2. **Modificar un perfil existente**:
   sql
   Copiar código

   ```
   ALTER PROFILE nombre_perfil LIMIT SESSIONS_PER_USER 3 IDLE_TIME 30;
   ```

Asignar un Perfil a un Usuario

- Para asignar un perfil a un usuario:sql
  Copiar código

  ```
  ALTER USER nombre_usuario PROFILE nombre_perfil;
  ```

Recursos de Password de un Perfil

- Políticas relacionadas con la seguridad de contraseñas, como:
- *LONGITUD MÍNIMA*: Número mínimo de caracteres.
- *EXPIRACIÓN*: Tiempo máximo de vida de una contraseña.
- *REUTILIZACIÓN*: Número de contraseñas previas que no pueden reutilizarse.

Ejemplo para definir recursos de contraseña:

sql

Copiar código

```
ALTER PROFILE nombre_perfil LIMIT PASSWORD_LIFE_TIME 90 PASSWORD_REUSE_TIME 365 PASSWORD_LOCK_TIME 1;
```

Trabajar con Recursos de Password

1. **Bloquear cuentas tras intentos fallidos**:
   sql
   Copiar código

   ```
   ALTER PROFILE nombre_perfil LIMIT FAILED_LOGIN_ATTEMPTS 5;
   ```

2. **Configurar tiempo de bloqueo tras intentos fallidos**:
   sql
   Copiar código

   ```
   ALTER PROFILE nombre_perfil LIMIT PASSWORD_LOCK_TIME 1/24;
   ```

Borrar un Perfil

- Un perfil no se puede eliminar si está asignado a usuarios:

3. Cambiar a los usuarios al perfil DEFAULT:sql
   Copiar código

   ```
   ALTER USER nombre_usuario PROFILE DEFAULT;
   ```

4. Eliminar el perfil:sql
   Copiar código

   ```
   DROP PROFILE nombre_perfil CASCADE;
   ```

Enterprise Manager para Profiles

- **Enterprise Manager** permite:
- Crear y modificar perfiles de manera gráfica.
- Configurar límites de recursos y políticas de contraseñas.
- Asignar perfiles a usuarios.

Tema XVII: Memoria

Introducción a la Gestión de Memoria

- Oracle utiliza **memoria compartida y privada** para optimizar el rendimiento y la gestión de la base de datos.
- **Componentes principales**:
- **SGA (System Global Area)**: Memoria compartida por todos los procesos de la base de datos.
- **PGA (Program Global Area)**: Memoria privada asignada a cada proceso del usuario.

Objetivo de la gestión de memoria: **Optimizar el rendimiento** ajustando el uso de recursos según la carga de trabajo.

Configurar la Memoria Automática en Oracle (AMM)

- **AMM (Automatic Memory Management)** permite que Oracle administre automáticamente SGA y PGA.
- Habilitar AMM:

1. Configurar el parámetro MEMORY_TARGET (tamaño total):sql
   Copiar código

   ```
   ALTER SYSTEM SET MEMORY_TARGET = 2G SCOPE = SPFILE;
   ```

2. Configurar el límite máximo:sql
   Copiar código

   ```
   ALTER SYSTEM SET MEMORY_MAX_TARGET = 2G SCOPE = SPFILE;
   ```

3. Reiniciar la base de datos:sql
   Copiar código

   ```
   SHUTDOWN IMMEDIATE; STARTUP;
   ```

Configurar la SGA de Forma Automática

- Habilitar **ASMM (Automatic Shared Memory Management)** para que Oracle ajuste automáticamente los componentes de la SGA.
- Parámetro clave: SGA_TARGET.sql
  Copiar código

  ```
  ALTER SYSTEM SET SGA_TARGET = 1G SCOPE = SPFILE;
  ```

- Desactivar AMM y habilitar ASMM:sql
  Copiar código

  ```
  ALTER SYSTEM SET MEMORY_TARGET = 0 SCOPE = SPFILE; ALTER SYSTEM SET SGA_TARGET = 1G SCOPE = SPFILE; ALTER SYSTEM SET PGA_AGGREGATE_TARGET = 500M SCOPE = SPFILE;
  ```

Configurar la SGA de Forma Manual

- Se gestionan manualmente los componentes clave:

- **Buffer Cache**:sql
  Copiar código

  ```
  ALTER SYSTEM SET DB_CACHE_SIZE = 500M;
  ```

- **Shared Pool**:sql
  Copiar código

  ```
  ALTER SYSTEM SET SHARED_POOL_SIZE = 300M;
  ```

- **Large Pool**:sql
  Copiar código

  ```
  ALTER SYSTEM SET LARGE_POOL_SIZE = 100M;
  ```

Crear Buffers de Datos y Tablespaces con Tamaños de Bloque Distintos

1. **Buffers de datos**:
- Crear un nuevo buffer para un tamaño de bloque específico:sql
  Copiar código

  ```
  ALTER SYSTEM SET DB_nK_CACHE_SIZE = 128M;
  ```

  Donde n es el tamaño de bloque (4, 8, 16, etc.).

2. **Tablespaces con tamaños de bloque distintos**:
- Crear un tablespace con tamaño de bloque personalizado:sql
  Copiar código

  ```
  CREATE TABLESPACE tbs_nk DATAFILE '/ruta/tbs_nk.dbf' SIZE 100M BLOCKSIZE 16K;
  ```

Configurar la PGA

- **PGA (Program Global Area)** es memoria privada utilizada para tareas como sorting y hash joins.
- Configuración:
- Habilitar PGA automática:sql
  Copiar código

  ```
  ALTER SYSTEM SET PGA_AGGREGATE_TARGET = 500M;
  ```

- Consultar el tamaño configurado:sql
  Copiar código

  ```
  SELECT VALUE FROM V$PARAMETER WHERE NAME = 'pga_aggregate_target';
  ```

Enterprise Manager y Memoria

- **Enterprise Manager** permite:
- Monitorear el uso de SGA y PGA en tiempo real.
- Configurar parámetros de memoria de manera gráfica.
- Ajustar valores como SGA_TARGET, PGA_AGGREGATE_TARGET, y otros componentes específicos.

Tema XVII: BACKUP y Recovery con RMAN

Introducción al Backup y Recovery

- **Backup y recovery** son pilares de la administración de bases de datos Oracle para garantizar la disponibilidad y recuperación de datos en caso de fallos.
- **RMAN (Recovery Manager)**:
- Herramienta de línea de comandos para gestionar respaldos y recuperaciones.
- Optimiza la creación, administración y recuperación de backups.

Repasemos la Importancia de ARCHIVELOG

- En modo **ARCHIVELOG**, los redo logs se archivan, permitiendo:
- Recuperación completa de datos en caso de fallos.
- Respaldo sin necesidad de apagar la base de datos.
- Consultar el estado del modo:sql
  Copiar código

  ```
  ARCHIVE LOG LIST;
  ```

Funcionamiento de los Archivelog

- Los redo logs llenos se copian a un archivo en el **destino de archivelogs**.
- Se usan en recuperaciones para reproducir cambios realizados después de un backup.

FRA - Fast Recovery Area

- **FRA (Fast Recovery Area)**: Espacio de almacenamiento dedicado para backups, archivelogs y otros archivos de recuperación.
- Configurar FRA:sql
  Copiar código

  ```
  ALTER SYSTEM SET DB_RECOVERY_FILE_DEST='/ruta/fra'; ALTER SYSTEM SET DB_RECOVERY_FILE_DEST_SIZE=10G;
  ```

Recuperación Automática de Instancia

- Oracle puede recuperar automáticamente la base de datos después de un fallo, aplicando redo logs y archivelogs según sea necesario.

Backup y Restore Completo en Frío

- **Backup en frío**: Base de datos apagada y sin usuarios conectados.
- Pasos:

1. Apagar la base de datos:sql
   Copiar código

   ```
   SHUTDOWN IMMEDIATE;
   ```
2. Copiar los archivos de datos, redo logs y controlfiles al destino de backup.
3. Reiniciar la base de datos:sql
   Copiar código

   ```
   STARTUP;
   ```

Activar ARCHIVELOG

1. Cambiar al modo MOUNT:sql
   Copiar código

   ```
   SHUTDOWN IMMEDIATE; STARTUP MOUNT;
   ```

2. Activar ARCHIVELOG:sql
   Copiar código

   ```
   ALTER DATABASE ARCHIVELOG;
   ```

3. Abrir la base de datos:sql
   Copiar código

   ```
   ALTER DATABASE OPEN;
   ```

Introducción a RMAN

- **RMAN** es la herramienta principal para realizar backups y recuperar datos en Oracle.

Conectarnos con RMAN

1. Desde el terminal:bash
   Copiar código

   ```
   rman target / # Conexión local
   ```

2. Con usuario y contraseña:bash
   Copiar código

   ```
   rman target sys/password@base_datos
   ```

Configuración Persistente de RMAN

- Definir configuraciones que persisten entre sesiones.
- Ejemplo:sql
  Copiar código

  ```
  CONFIGURE RETENTION POLICY TO REDUNDANCY 2; CONFIGURE CONTROLFILE AUTOBACKUP ON;
  ```

Formato de los Backups con RMAN

- Definir nombres de archivos de backup:sql
  Copiar código

  ```
  CONFIGURE CHANNEL DEVICE TYPE DISK FORMAT '/ruta/backup_%U.bkp';
  ```

Scripts de RMAN

- Ejecutar comandos almacenados en scripts.
- Ejemplo:bash
  Copiar código

  ```
  RUN { BACKUP DATABASE PLUS ARCHIVELOG; }
  ```

Backup Completo de la Base de Datos

- Realizar un backup completo:bash
  Copiar código

  ```
  BACKUP DATABASE;
  ```

LIST: Buscando Información de los Backups

- Consultar detalles de backups realizados:bash
  Copiar código

  ```
  LIST BACKUP;
  ```

Backup de Tablespaces

- Respaldo de un tablespace específico:bash
  Copiar código

  ```
  BACKUP TABLESPACE users;
  ```

Backup de Tablespace como Image Copy

- Crear una copia exacta de un tablespace:bash
  Copiar código

  ```
  BACKUP AS COPY TABLESPACE users;
  ```

Backup de Datafiles

- Respaldo de un archivo de datos específico:bash
  Copiar código

  ```
  BACKUP DATAFILE '/ruta/datafile01.dbf';
  ```

Backup de Controlfiles

- Incluir el controlfile en un backup:bash
  Copiar código

  ```
  BACKUP CURRENT CONTROLFILE;
  ```

Backup de Archivelogs

- Respaldar todos los archivelogs disponibles:bash
  Copiar código

  ```
  BACKUP ARCHIVELOG ALL;
  ```

Backups Incrementales: Introducción

- **Backups incrementales**: Solo respaldan bloques de datos modificados desde el último backup.
- Ejemplo:bash
  Copiar código

  ```
  BACKUP INCREMENTAL LEVEL 1 DATABASE;
  ```

Canales: Cambiar el Canal por Defecto

- Configurar un canal para dispositivos específicos:bash
  Copiar código

  ```
  CONFIGURE CHANNEL DEVICE TYPE DISK FORMAT '/ruta/canal_%U.bkp';
  ```

RUN: Lanzar Jobs en RMAN

- Crear un bloque de comandos:bash
  Copiar código

  ```
  RUN { ALLOCATE CHANNEL c1 DEVICE TYPE DISK; BACKUP DATABASE;
  RELEASE CHANNEL c1; }
  ```

RECOVER: Introducción

- Usado para aplicar archivelogs y restaurar el estado de la base de datos.

Recuperar la Base de Datos Completa

- Ejemplo:bash
  Copiar código

  ```
  RECOVER DATABASE;
  ```

Recuperar una Tablespace

- Ejemplo:bash
  Copiar código

  ```
  RECOVER TABLESPACE users;
  ```

Recuperación Hasta un Punto en el Tiempo

- Restaurar la base de datos a un estado específico:bash
  Copiar código

  ```
  RECOVER DATABASE UNTIL TIME '2024-11-14 10:00:00';
  ```

Política de Retención

- Configurar la retención de backups:bash
  Copiar código

  ```
  CONFIGURE RETENTION POLICY TO REDUNDANCY 2;
  ```

Comando REPORT

- Analizar información sobre backups:bash
  Copiar código

  ```
  REPORT OBSOLETE;
  ```

DELETE: Borrar Backups

- Eliminar backups innecesarios:bash
  Copiar código

  DELETE BACKUP;

TEMA XVIII: Resolución de Problemas

Introducción a la Gestión de Problemas

- La **gestión de problemas** en Oracle permite identificar, registrar y solucionar errores que ocurren en la base de datos.
- **Componentes clave**:
- **ADR (Automatic Diagnostic Repository)**: Framework para almacenar información diagnóstica.
- **Ficheros de trazas**: Contienen detalles técnicos sobre el funcionamiento y errores.
- **Fichero de Alert Log**: Registra eventos importantes como errores, cambios en la base de datos y operaciones administrativas.

Revisión del ADR

- **ADR (Automatic Diagnostic Repository)**:
- Almacena información estructurada sobre incidentes, problemas y diagnósticos.
- Incluye:
  - Ficheros de trazas.
  - Dumps (información detallada de errores graves).
  - Información de incidentes.

Consultar ubicación del ADR:

sql

Copiar código

```
SELECT VALUE FROM V$DIAG_INFO WHERE NAME = 'ADR Home';
```

Fichero de Alert Log

- **Alert Log** registra:
- Cambios en la base de datos (arranques, apagados).
- Errores de memoria, disco o transacciones.
- Problemas relacionados con procesos.
- Ubicación:
- En el directorio trace del ADR.
- Consultar el archivo directamente:bash
  Copiar código

  ```
  tail -f /ruta/alert_<SID>.log
  ```

Ficheros de Traza, Core y Dump

- **Trazas**: Información detallada sobre procesos específicos.
- Generadas para comandos problemáticos o fallos.
- Ejemplo: Logs de SQL problemáticos.
- **Core Dumps**: Datos detallados sobre fallos graves (errores de memoria o sistema).

- **Dumps de procesos**: Información de diagnóstico para sesiones y procesos problemáticos.

Trazas de los Comandos DDL

- Registrar trazas para comandos DDL:sql
  Copiar código

  ```
  ALTER SESSION SET SQL_TRACE = TRUE;
  ```

- Consultar los resultados en los ficheros de traza generados.

ADRCI: Herramienta en Modo Comando para ADR

- **ADRCI (ADR Command-Line Interface)**:
- Interfaz para acceder y gestionar información en el ADR.
- Comandos básicos:
  - Iniciar ADRCI:bash
    Copiar código

    ```
    adrci
    ```

Trabajar con ADRCI

- **Ver trazas, incidentes y problemas**:
- Listar problemas:bash
  Copiar código

  ```
  SHOW PROBLEM;
  ```

- Ver incidentes relacionados con un problema:bash
  Copiar código

  ```
  SHOW INCIDENT;
  ```

- Consultar trazas específicas:bash
  Copiar código

  ```
  SHOW TRACEFILE;
  ```

- **Ejemplo de problema e incidente**:
- Identificar un problema:bash
  Copiar código

  ```
  SHOW PROBLEM;
  ```

- Ver detalles del incidente asociado:bash
  Copiar código

  ```
  SHOW INCIDENT WHERE PROBLEM_ID = <ID>;
  ```

Crear un Paquete para Mandar a Oracle Soporte

1. Crear un paquete para incidentes:bash
   Copiar código

   ```
   IPS CREATE PACKAGE PROBLEM <problem_id>;
   ```

2. Agregar incidentes al paquete:bash
   Copiar código

   ```
   IPS ADD INCIDENT <incident_id> PACKAGE <package_id>;
   ```

3. Finalizar el paquete:bash
   Copiar código

   ```
   IPS GENERATE PACKAGE <package_id>;
   ```

4. Consultar la ubicación del paquete:bash
   Copiar código

   ```
   SHOW PACKAGE;
   ```