UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

IMPLEMENTACIÓN DE BASE DE DATOS II

INVESTIGACIÓN FINAL

“ASEGURAMIENTO DEL ACCESO DE LA BASE DE DATOS”

GRUPO: 1SF133

PROFESOR HENRY LEZCANO

FECHA:

24 DE JUNIO DE 2024

I SEMESTRE DE 2024

INTEGRANTES POR GRUPO

EQUIPO #1

Grupo #1					
#	Apellido	Nombre	Cedula		
1	Campos	Rodolfo	08	0905	002179
2	Achurra	Adriana	08	0990	000123
3	Arrocha	Victor	08	0996	001219
4	Alexander	Camargo	08	1002	001756
5	Sebastian	Ferrer	E	8211	003001

EQUIPO #2

Grupo #2					
#	Apellido	Nombre	Cedula		
1	Lucero	Pedro	08	0973	001067
2	Valdes	Andrés	06	0726	001938
3	Coronado	Arantxa	02	0752	001519
4	Hernandez	Jose	08	1002	002448
5	Candanedo	Rafaela	08	0905	001651

EQUIPO #3

Grupo #3					
#	Apellido	Nombre	Cedula		
1	Gomez	Alejandra	20	0014	007146
2	Ramirez	María	20	0014	006014
3	Ramos	Angel	08	0999	002133
4	Ibrahim	Lorenzo	04	0826	001268

EQUIPO #4

Grupo #4					
#	Apellido	Nombre	Cedula		
1	Zhong	Carlos	08	0992	002438
2	Rojas	Juan	08	0993	001590
3	Mo	Luis	03	0752	000562
4	Chavarria	Christopher	08	1002	000376
5	José	Herazo	08	0986	002492

EQUIPO #5

Grupo #5					
#	Apellido	Nombre	Cedula		
1	Somoza	Alder	08	1002	000814
2	Lopez	Alexis	08	0994	001521
3	Perez	Leonor	08	1004	002420
4	Morales	Julissa	08	0995	002383
5	Rivas	Daniel	20	0070	007101

EQUIPO #6

Grupo #6					
#	Apellido	Nombre	Cedula		
1	Adames	Jose	08	1008	001433
2	Archbold	Bryan	08	0957	002109
3	Rodríguez	César	08	0995	000494
4	Patiño	Anthony	08	1011	001099

EQUIPO #7

Grupo #7					
#	Apellido	Nombre	Cedula		
1	Arias	Alan	08	0997	002267
2	Rovira	Alejandro	08	1015	001268
3	Grant	Joseph	08	1005	001268
4	Povaz	María	08	0944	000285
5	Nelson	Adrián	08	0970	000201

EQUIPO #8

Grupo #8					
#	Apellido	Nombre	Cedula		
1	Batista	Enrique	08	1011	001855
2	Mejia	Sebastian			
3	Gomez	Jose	20	0014	007317
4	María	Belén			
5	Antonio	Carmona	08	1002	000766

INTRODUCCIÓN

El Grupo 133 se ha embarcado en una investigación integral sobre el aseguramiento del acceso a bases de datos. Esta iniciativa es crucial para comprender y aplicar las mejores prácticas en la administración y seguridad de bases de datos, un componente vital en cualquier sistema de información.

El objetivo principal de esta investigación es profundizar en diversos aspectos de la administración de bases de datos, asegurando su acceso y protección mediante estrategias robustas. A lo largo de este proyecto, los equipos abordarán áreas clave que incluyen la administración y autenticación de usuarios, la gestión de privilegios y funciones, los límites de recursos de base de datos, la administración de cuentas de usuario, así como los procedimientos de respaldo y recuperación.

Para la correcta ejecución de esta investigación, cada equipo deberá preparar y presentar un informe detallado y una presentación en formato PPT.

RESUMEN

Esta investigación se centrará en el aseguramiento del acceso a bases de datos, abarcando aspectos críticos para la administración y protección de los datos. Divididos en equipos, los estudiantes exploraron temas específicos para garantizar un enfoque detallado y especializado en cada área.

Cada equipo preparó un informe detallado y una presentación en PPT, contribuyendo a un informe general que encapsula las diversas dimensiones del aseguramiento del acceso a bases de datos.

INVESTIGACIÓN FINAL

ASEGURAMIENTO DEL ACCESO DE LA BASE DE DATOS

EQUIPO # 1

A. ADMINISTRACIÓN Y AUTENTICACIÓN DE USUARIOS

La administración y autenticación de usuarios son fundamentales para asegurar que solo personas autorizadas puedan acceder a la base de datos. Esto incluye la creación, gestión y monitoreo de cuentas de usuario, así como la implementación de mecanismos para verificar la identidad de los usuarios.

a. Cuentas de usuario bloqueadas y no bloqueadas

- **Cuentas bloqueadas:**
    - Las cuentas de usuario pueden ser bloqueadas por varias razones, como intentos fallidos de inicio de sesión, inactividad prolongada, o por razones de seguridad tras la detección de comportamientos sospechosos.
    - El bloqueo de cuentas ayuda a prevenir accesos no autorizados y protege la integridad de la base de datos.
    - Para bloquear sin eliminar, usamos el siguiente comando:
      ALTER USER user_example ACCOUNT **LOCK**;
    - Para desbloquear una cuenta bloqueada, usamos el siguiente comando:
      ALTER USER user_example ACCOUNT **UNLOCK**;

- **Cuentas no bloqueadas:**
    - Son cuentas activas y accesibles, que los usuarios autorizados pueden utilizar para acceder a la base de datos.
    - La administración de estas cuentas incluye asegurarse de que los usuarios mantengan contraseñas seguras y actualizadas, y que se cumplan las políticas de seguridad establecidas.

b. Usuarios predeterminados de la base de datos

Son cuentas de usuario que vienen preconfiguradas con la instalación de la base de datos. Ejemplos comunes incluyen cuentas como admin, root, guest, entre otras.

- Creemos un usuario admin con privilegios elevados:
  CREATE USER admin IDENTIFIED **BY** Admin#123;
  GRANT DBA TO admin;
  GRANT SYSDBA **TO** admin;
  GRANT CONNECT, **RESOURCE TO** admin;

- Creemos un usuario root con privilegios elevados:
  CREATE USER root IDENTIFIED **BY** Root#123;
  GRANT DBA **TO** root;
  GRANT SYSDBA **TO** root;
  GRANT CONNECT, **RESOURCE TO** root;

- Creemos un usuario guest con privilegios limitados:
  CREATE USER guest IDENTIFIED **BY** Guest#123;
  GRANT CONNECT TO guest;
  GRANT SELECT ON all_tables **TO** guest;

Estas cuentas suelen tener privilegios elevados y, por lo tanto, son un objetivo potencial para los atacantes.

Es crucial cambiar las contraseñas predeterminadas de estas cuentas y deshabilitar aquellas que no sean necesarias**.**

c. Buenas prácticas para la administración y autenticación de usuarios

- **Políticas de Contraseñas Fuertes:**
    - Implementar requisitos de complejidad para las contraseñas (longitud mínima, uso de caracteres especiales, mayúsculas y minúsculas, etc.).
    - Forzar el cambio de contraseñas periódicamente.

      CREATE PROFILE secure_profile

      LIMIT

      FAILED_LOGIN_ATTEMPTS 5

      PASSWORD_LIFE_TIME 90

      PASSWORD_REUSE_TIME 365

      PASSWORD_REUSE_MAX 5

      PASSWORD_VERIFY_FUNCTION custom_verify_function;

- **Autenticación Multifactor (MFA):**
    - Añadir una capa adicional de seguridad requiriendo que los usuarios verifiquen su identidad mediante algo que saben (contraseña) y algo que tienen (token, código enviado a un dispositivo móvil, etc.).

- Creamos una tabla de contraseñas a la que podamos referenciar luego:
  CREATE TABLE mfa_codes (

  username **VARCHAR2**(30),

  code **VARCHAR2**(10),

  expiration **TIMESTAMP**,

  PRIMARY KEY (username));

- **Monitoreo y Auditoría:**
  - Registrar y revisar los intentos de inicio de sesión y otras actividades relevantes de los usuarios.
  - Auditar regularmente las cuentas de usuario para detectar y responder a comportamientos inusuales o no autorizados.
  - Para habilitar auditoría de intentos de inicio de sesión
    AUDIT SESSION
  - Para consultar los registros de auditoría, usamos:
    SELECT username, **timestamp**, action_name

    FROM dba_audit_trail

    WHERE action_name = 'SESSION':
  - Para habilitar auditoría de fallos de inicio de sesión, usamos:
    AUDIT SESSION WHENEVER **NOT** SUCCESSFUL:

- **Principio de Mínimos Privilegios:**
    - Asignar a cada usuario solo los permisos necesarios para realizar su trabajo. Esto minimiza el riesgo en caso de que una cuenta sea comprometida.
    - Para crear roles con permisos mínimos, podemos usar:
      CREATE ROLE read_only_role;
      GRANT SELECT ON some_table **TO** read_only_role;

- **Capacitación de Usuarios:**
    - Educar a los usuarios sobre la importancia de la seguridad de la información y las mejores prácticas para proteger sus credenciales de acceso.

EQUIPO #2

B. ADMINISTRACIÓN DE PRIVILEGIOS Y FUNCIONES

a. Tipos de privilegios y funciones:

○ Privilegios del Sistema

Los privilegios del sistema permiten realizar acciones a nivel de base de datos. Estos privilegios son necesarios para crear, alterar o eliminar objetos de la base de datos y gestionar el entorno de la base de datos. Algunos ejemplos comunes incluyen:

- **CREATE SESSION**: Permite a un usuario crear una conexión a la base de datos.
- **CREATE TABLE**: Permite al usuario crear tablas en su propio esquema.
- **CREATE VIEW**: Permite al usuario crear vistas.
- **CREATE ANY TABLE**: Permite al usuario crear tablas en cualquier esquema.
- **DROP ANY TABLE**: Permite al usuario eliminar tablas de cualquier esquema.
- **CREATE USER**: Permite al usuario crear otros usuarios.
- **ALTER USER**: Permite al usuario modificar otros usuarios.
- **DROP USER**: Permite al usuario eliminar otros usuarios.

○ Privilegios Administrativos

Los privilegios administrativos están diseñados para tareas administrativas comunes, como realizar operaciones de respaldo y recuperación. Oracle Database proporciona privilegios administrativos específicos para tareas administrativas específicas, como el privilegio administrativo SYSKM para realizar tareas de Cifrado Transparente de Datos (Transparent Data Encryption).

- **SYSDBA**: Permite realizar casi todas las operaciones administrativas, incluyendo el arranque y la detención de la base de datos, recuperación y respaldo.

- **SYSOPER**: Permite realizar un subconjunto de operaciones administrativas, como el arranque y la detención de la base de datos.
- **SYSKM**: Permite realizar tareas de administración de cifrado transparente de datos.
- **SYSBACKUP**: Permite realizar operaciones de respaldo y recuperación de la base de datos.
- **SYSDG**: Permite realizar operaciones de administración de Data Guard.
- **SYSRAC**: Permite realizar operaciones de administración de Oracle Real Application Clusters (RAC).
- **SYSASM**: Privilegio para realizar operaciones de administración de Automatic Storage Management (ASM).

    - **Privilegios de Objeto**

Los privilegios de objeto permiten a los usuarios realizar acciones específicas en objetos particulares dentro de la base de datos, como tablas, vistas, secuencias y procedimientos almacenados. Algunos ejemplos incluyen:

- **DELETE**: Permite a un usuario eliminar de una tabla.
- **EXECUTE**: Permite a un usuario ejecutar un paquete, procedimiento o función PL/SQL directamente.
- **FLUSH**: Permite a un usuario vaciar un grupo de caché.
- **INDEX**: Permite a un usuario crear un índice en una tabla o vista materializada.
- **INSERT**: Permite a un usuario insertar en una tabla o en la tabla a través de un sinónimo.
- **LOAD**: Permite a un usuario cargar un grupo de caché.
- **REFERENCES**: Permite a un usuario crear una dependencia de clave externa en una tabla o vista materializada.
- **REFRESH**: Permite a un usuario actualizar un grupo de caché.

- **SELECT**: Permite a un usuario seleccionar entre una tabla, secuencia, vista, vista materializada o sinónimo.
- **UPDATE**: Permite a un usuario actualizar una tabla.

EQUIPO #3

b. Conceder y restaurar privilegios de usuario

- **Roles en Oracle**

En Oracle, los roles se utilizan para agrupar privilegios relacionados que pertenecen a requisitos específicos. Los roles permiten la continuidad y el límite del acceso en función de la responsabilidad. Los roles más comunes en Oracle incluyen:

DBA: El rol predefinido de DBA normalmente permite todos los privilegios administrativos en una base de datos de Oracle.

SYSDBA: El rol SYSDBA proporciona privilegios de administración avanzados, como la capacidad de crear y eliminar usuarios, así como de realizar operaciones de mantenimiento en la base de datos.

SYSTEM: El rol SYSTEM proporciona privilegios de sistema, como la capacidad de crear y eliminar usuarios, así como de realizar operaciones de mantenimiento en la base de datos.

- **Privilegios Individuales**

Además de los roles, se pueden conceder privilegios individuales a un usuario. Los privilegios individuales permiten conceder permisos específicos a un usuario. Algunos de los privilegios individuales más comunes en Oracle incluyen:

- **CREATE VIEW**: Permite a un usuario crear vistas en la base de datos.
- **CREATE SEQUENCE**: Permite a un usuario crear secuencias en la base de datos.
- **CREATE TABLESPACE**: Permite a un usuario crear tablespaces en la base de datos.

- **ALTER DATABASE**: Permite a un usuario realizar operaciones de mantenimiento en la base de datos, como la modificación de parámetros de inicialización.
- **ALTER SYSTEM**: Permite a un usuario realizar operaciones de mantenimiento en el sistema, como la modificación de parámetros de inicialización.

Conceder privilegios de sistema

Se usa con la instrucción GRANT que funciona así:

```
GRANT privilegio1 [,privilegio2[,...]] TO usuario
[WITH ADMIN OPTION];
```

La opción **WITH ADMIN OPTION** permite que el usuario al que se le concede el privilegio puede conceder dicho privilegio a otros usuarios. Es, por tanto, una opción a utilizar con cautela. Se utiliza únicamente con privilegios de sistema.

Conceder privilegios de objeto

Se trata de privilegios que se colocan a un objeto para dar permiso de uso a un usuario.

Sintaxis:

```
GRANT {privilegio [(listaColumnas)] [,...]] | ALL [PRIVILEGES]}
ON [esquema.]objeto
TO {usuario | rol} [,{usuario | rol } [,...]]
[WITH GRANT OPTION];
```

Podemos conceder diferentes permisos a nuestros usuarios y así controlar exactamente que acciones queremos que puedan realizar y que acciones no van a ser permitidas.

```
GRANT privilegio1 [[,privilegio2, ...] | ALL]
```

```
[(columna1[,columna2,...])]

[ON usuario[.objeto] | ANY TABLE]

TO {nombreUsuario | rol | PUBLIC}

[WITH GRANT OPTION];
```

- **ON**: Objeto sobre el que aplico los privilegios
- **TO**: Usuario al que concedo los privilegios
- **ALL**: Permite asignar todos los permisos
- **PUBLIC**: Asigna el privilegio o privilegios a todos los usuarios del sistema (también a los futuros)
- **WITH GRANT OPTION**: Permite que el usuario que lo reciba pueda conceder permisos a otros usuarios

La opción **ALL** concede todos los privilegios posibles sobre el objeto. Se pueden asignar varios privilegios a la vez y también varios posibles usuarios.

La opción **WITH GRANT OPTION** permite al usuario al que se le conceden los privilegios, que pueda, a su vez, concederlos a otro. Se utiliza únicamente con privilegios de objeto.

Podemos también asignar ciertos permisos sobre el sistema a los usuarios que tengamos creados en nuestra base de datos. Los privilegios del sistema no se asignan sobre objetos concretos, sino que especifican que acciones se podrán realizar sobre el sistema gestor de base de datos.

```
GRANT permiso1[,permiso2,...]

TO nombreUsuario[,nombreUsuario2,...] | nombreRol;
```

Dónde permiso puede ser alguno de los siguientes:

create

- session: Permite conectarse a la base de datos
- table: Permite crear tablas
- sequence: Permite crear secuencias
- view: Permite crear vistas
- trigger: Permite crear disparadores
- procedure: Permite crear procedimientos
- profile: Permite crear perfiles
- synonym: Permite crear sinónimos
- execute any procedure: Permite ejecutar cualquier procedimiento

create

- user: Permite crear usuarios. WITH ADMIN OPTIONS permite que el nuevo usuario tenga permisos administrativos, por ejemplo, para crear nuevos usuarios.
- role: Permite crear roles

drop

- table: Permite eliminar tables
- sequence: Permite eliminar secuencias
- view: Permite eliminar vistas
- trigger: Permite eliminar disparadores
- procedure: Permite eliminar procedimientos
- profile: Permite eliminar perfiles
- synonym: Permite eliminar sinónimos
- user: Permite eliminar usuarios
- role: Permite eliminar roles
- session: Permite eliminar sesiones

grant

- privilege: Permite asignar privilegios
- role: Permite asignar roles

Ejemplos de Conceder y Restaurar Privilegios

A continuación, se proporcionan algunos ejemplos de cómo conceder y restaurar privilegios en Oracle:

```sql
-- Conceder el privilegio CREATE SESSION a un usuario
GRANT CREATE SESSION TO usuario;

-- Conceder el privilegio ALTER DATABASE a un usuario
GRANT ALTER DATABASE TO usuario;

-- Conceder el privilegio ALTER SYSTEM a un usuario
GRANT ALTER SYSTEM TO usuario;

-- Revocar el privilegio CREATE SESSION a un usuario
REVOKE CREATE SESSION FROM usuario;

-- Revocar el privilegio ALTER DATABASE a un usuario
REVOKE ALTER DATABASE FROM usuario;

-- Revocar el privilegio ALTER SYSTEM a un usuario
REVOKE ALTER SYSTEM FROM usuario;
```

EQUIPO #2 (CONTINUACIÓN)

c. Administración de privilegios con tareas

En la administración de una base de datos Oracle, es crucial gestionar correctamente los privilegios para asegurar que los usuarios puedan realizar sus tareas de manera eficiente y segura. La administración de privilegios con tareas implica otorgar y revocar permisos específicos a los usuarios en función de sus roles y responsabilidades dentro de la organización.

- **Evaluación de Necesidades**

Antes de otorgar privilegios, se debe evaluar qué permisos son realmente necesarios para que un usuario realice sus tareas. Esto ayuda a evitar la concesión excesiva de privilegios que podrían ser explotados. Por ejemplo, un usuario que necesita insertar y actualizar registros en la tabla 'empleados' no debería tener privilegios para eliminar tablas o modificar la estructura de la base de datos.

- **Uso de Roles**

En lugar de otorgar privilegios directamente a los usuarios individuales, es recomendable agrupar los privilegios en roles. Los roles son colecciones de privilegios que se pueden asignar a múltiples usuarios, facilitando la gestión de permisos. Un rol llamado ‘editor_empleado’ podría incluir

permisos para insertar y actualizar datos en la tabla 'empleados'. Este rol se crea y se le otorgan los privilegios necesarios, y luego se asigna a los usuarios que desempeñan funciones relacionadas.

Para gestionar estos roles, primero se crea un rol llamado ' editor_empleado':

```
CREATE ROLE editor_empleado;
```

Luego, se le otorgan los privilegios necesarios para insertar y actualizar registros en la tabla 'empleados':

```
GRANT INSERT, UPDATE ON employees TO editor_empleado;
```

Finalmente, se asigna este rol al usuario 'jdoe':

```
GRANT editor_empleado TO jdoe;
```

Este enfoque facilita la administración y permite una fácil revocación de permisos cuando ya no sean necesarios.

- **Revisión y Auditoría**

Regularmente, se deben revisar y auditar los privilegios otorgados para asegurarse de que todavía son necesarios y no se han otorgado de manera excesiva. La auditoría ayuda a identificar y corregir cualquier concesión indebida de privilegios. Es importante mantener un registro de los privilegios otorgados y revocados para asegurarse de que se siguen las mejores prácticas de seguridad.

- **Revocación de rol**

Como ya se mencionó cuando un usuario ya no necesita ciertos privilegios, es importante revocarlos para mantener la seguridad. Por ejemplo, si el usuario 'jdoe' ya no necesita editar los registros en la tabla 'empleados, se revoca el rol ‘editor_empleado’ asignado a este usuario:

```
REVOKE editor_empleado FROM jdoe;
```

De esta manera, todos los privilegios asociados con ese rol también se revocan, asegurando que 'jdoe' no pueda realizar acciones no autorizadas.

En entornos multitenant, la administración de privilegios se complica por la existencia de múltiples bases de datos conectables (PDBs). Es crucial distinguir entre privilegios otorgados común y localmente. Los privilegios comunes se aplican a todos los PDBs, mientras que los privilegios locales solo se aplican al PDB específico en el que se otorgan. Por ejemplo, un privilegio otorgado comúnmente a un usuario común permite que este usuario utilice el privilegio en todos los contenedores actuales y futuros. Para otorgar un privilegio comúnmente, se usaría la siguiente instrucción:

```
GRANT CREATE ANY TABLE TO c##hr_admin CONTAINER=ALL;
```

Sin embargo, un privilegio otorgado localmente solo es efectivo en el contenedor específico donde se otorgó. Para otorgar un privilegio localmente, se usaría la siguiente instrucción:

```
GRANT CREATE TABLE TO hr_admin CONTAINER=CURRENT;
```

La administración de privilegios con tareas es esencial para mantener la seguridad y eficiencia en la gestión de bases de datos Oracle. Utilizar roles para agrupar privilegios y revisar regularmente los permisos otorgados ayuda a mantener un entorno seguro y bien gestionado. La correcta asignación y revocación de privilegios asegura que los usuarios solo tengan los accesos necesarios para cumplir con sus responsabilidades, minimizando los riesgos de seguridad.

EQUIPO #4

C. LÍMITES DE RECURSO DE BASE DE DATOS

a. Cuotas de espacio de tablas

Cuotas: Es una ubicación de almacenamiento donde pueden ser guardados los datos correspondientes a los objetos de una base de datos. Este provee una capa de abstracción entre los datos físicos y lógicos y sirve para asignar espacio para todos los segmentos administrados del

sistema de gestión de base de datos. Si no se especifica el espacio por defecto es 0 con lo cual el usuario no podrá crear nada.

Cuotas de espacio de tablas: son límites específicos establecidos para el almacenamiento de datos de cada tabla dentro de una base de datos.

Definición de Cuotas: Las cuotas pueden definirse en términos de tamaño máximo de datos que una tabla puede almacenar.

Aplicación de Cuotas: Las cuotas se aplican para controlar el uso excesivo de espacio por parte de tablas individuales. Esto ayuda a prevenir el agotamiento de recursos y a mantener un rendimiento consistente del sistema.

Gestión de Cuotas: Los administradores de bases de datos pueden establecer, modificar y monitorear las cuotas de espacio de tablas para asegurarse de que se cumplan los límites establecidos.

Impacto en el Rendimiento: Superar las cuotas de espacio puede tener un impacto negativo en el rendimiento de consultas y operaciones de escritura, ya que la base de datos podría necesitar distribuir datos o fragmentar tablas para acomodar el exceso de almacenamiento.

A continuación se mostrará un ejemplo sobre cómo aplicar estos principios a Oracle:

Una empresa llamada "Ventas Global" tiene una base de datos Oracle para gestionar sus datos de ventas. La empresa quiere asegurarse de que los datos de ventas y los datos de inventario no ocupen más espacio del necesario en la base de datos. Para esto, decide establecer cuotas de espacio para diferentes usuarios responsables de estos datos.

Requisitos:

- Crear dos tablespaces: uno para datos de ventas y otro para datos de inventario.

- Crear dos usuarios: uno para gestionar datos de ventas y otro para gestionar datos de inventario.

- Asignar cuotas específicas de espacio a cada usuario en sus respectivos tablespaces.

- Monitorizar el uso de espacio por cada usuario.

1- Primero, creamos dos tablespaces: ventas_tablespace y inventario_tablespace.

```
SQL>
SQL> -- Crear tablespace para datos de inventario
SQL> CREATE TABLESPACE inventario_tablespace
  2  DATAFILE 'inventario_datafile.dbf'
  3  SIZE 100M
  4  AUTOEXTEND ON
  5  NEXT 10M
  6  MAXSIZE 500M;

Tablespace created.
```

2- Creamos dos usuarios, ventas_user e inventario_user, y les asignamos cuotas de espacio.

```
SQL>
SQL> -- Crear usuario para gestionar datos de inventario
SQL> CREATE USER inventario_user
  2  IDENTIFIED BY inventario_password
  3  DEFAULT TABLESPACE inventario_tablespace
  4  QUOTA 100M ON inventario_tablespace;

User created.
```

3- Asignamos los privilegios necesarios para que los usuarios puedan crear y gestionar sus tablas.

```
SQL> -- Asignar privilegios al usuario de ventas
SQL> GRANT CONNECT, RESOURCE TO ventas_user;

Grant succeeded.

SQL>
SQL> -- Asignar privilegios al usuario de inventario
SQL> GRANT CONNECT, RESOURCE TO inventario_user;

Grant succeeded.
```

4- En cada usuario, creamos algunas tablas y verificamos el uso de espacio.

```
SQL> COMMIT;

Commit complete.

SQL>
SQL> -- Verificar el uso de espacio por el usuario de ventas
SQL> SELECT
  2      segment_name AS table_name,
  3      SUM(bytes)/1024/1024 AS size_mb
  4  FROM
  5      user_segments
  6  WHERE
  7      segment_type = 'TABLE'
  8  GROUP BY
  9      segment_name;

TABLE_NAME
--------------------------------------------------------------------------------
   SIZE_MB
----------
VENTAS
     .0625


SQL>
SQL> -- Conectar como usuario de inventario
SQL> CONNECT inventario_user/inventario_password;
Connected.
SQL>
```

```
SQL> SELECT
  2      segment_name AS table_name,
  3      SUM(bytes)/1024/1024 AS size_mb
  4  FROM
  5      user_segments
  6  WHERE
  7      segment_type = 'TABLE'
  8  GROUP BY
  9      segment_name;

TABLE_NAME
--------------------------------------------------------------------------------
   SIZE_MB
----------
INVENTARIO
     .0625


SQL>
```

```
SQL> GRANT SELECT ON dba_segments TO carlos;

Grant succeeded.
```

- **Monitorización del Uso de Espacio**

Para monitorizar el uso de espacio por cada usuario, utilizamos las siguientes consultas:

1- Uso de espacio por el usuario de ventas:

```
SQL> SELECT
  2      tablespace_name,
  3      SUM(bytes)/1024/1024 AS size_mb
  4  FROM
  5      dba_segments
  6  WHERE
  7      owner = 'VENTAS_USER'
  8  GROUP BY
  9      tablespace_name;

TABLESPACE_NAME                 SIZE_MB
------------------------------ ----------
VENTAS_TABLESPACE                    .125
```

2- Uso de espacio por el usuario de inventario:

```
SQL> SELECT
  2      tablespace_name,
  3      SUM(bytes)/1024/1024 AS size_mb
  4  FROM
  5      dba_segments
  6  WHERE
  7      owner = 'INVENTARIO_USER'
  8  GROUP BY
  9      tablespace_name;

TABLESPACE_NAME                 SIZE_MB
------------------------------ ----------
INVENTARIO_TABLESPACE                .125
```

b. Perfiles de límite de recursos

Permite limitar el uso de recursos, las operaciones de enlace y las modalidades de paralelismo en el proceso de determinadas sentencias.

Definición de Perfiles: Los perfiles de límite de recursos especifican límites para recursos como la CPU, el número de conexiones concurrentes, el tiempo de ejecución de consultas, la cantidad de memoria utilizada, etc.

Asignación a Usuarios: Los perfiles de límite de recursos suelen asignarse a roles o usuarios específicos dentro de la base de datos. Esto permite a los administradores controlar de manera precisa cómo se utilizan los recursos del sistema.

Propósitos de Control: Ayudan a prevenir la sobrecarga del sistema, el abuso de recursos y aseguran un uso justo y equitativo de los recursos de la base de datos entre múltiples usuarios y aplicaciones.

Implementación y Monitoreo: Es fundamental implementar correctamente los perfiles de límite de recursos y monitorear su efectividad para ajustarlos según sea necesario. Esto asegura que el rendimiento de la base de datos se mantenga óptimo y predecible.

Escenario:

La empresa tiene un equipo de desarrollo que trabaja en múltiples proyectos simultáneamente. El administrador de la base de datos quiere asegurarse de que ningún usuario de este equipo consume excesivamente los recursos de la base de datos, lo que podría afectar el rendimiento general. Queremos limitar el uso de sesiones, CPU, y lecturas lógicas por sesión y por llamada.

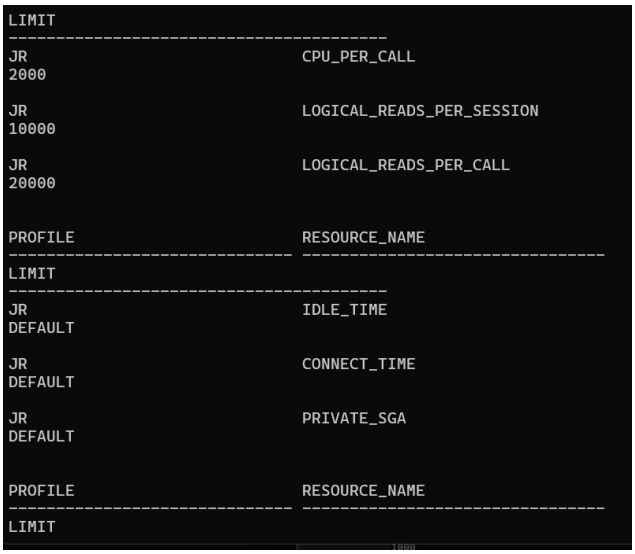
Límites a establecer:

- Máximo de 10 sesiones simultáneas por usuario.
- Máximo de 1000 décimas de segundo de CPU por sesión.
- Máximo de 2000 décimas de segundo de CPU por llamada.
- Máximo de 10,000 lecturas lógicas por sesión.
- Máximo de 20,000 lecturas lógicas por llamada.

```
SQL> grant resource to JR;

Grant succeeded.

SQL> grant create view to JR;

Grant succeeded.

SQL> SELECT profile, resource_name, limit
  2  FROM dba_profiles
  3  WHERE profile = 'JR';

PROFILE                        RESOURCE_NAME
------------------------------ --------------------------------
LIMIT
----------------------------------------
JR                             COMPOSITE_LIMIT
DEFAULT

JR                             SESSIONS_PER_USER
10

JR                             CPU_PER_SESSION
1000


PROFILE                        RESOURCE_NAME
------------------------------ --------------------------------
```

- Se crea el usuario y posteriormente se verifican los privilegios. Se pasa a crear el perfil de límite de recursos y verificar su creación.

- Se procede a asignar a los usuarios del equipo de desarrollo y se monitorea los eventos relacionados con los límites de recursos (recursos requeridos).



La gestión de recursos en bases de datos es clave para asegurar un buen rendimiento y estabilidad del sistema.

Los límites de recursos, como las cuotas de espacio de tablas, ayudan a controlar el uso de almacenamiento, evitando que usuarios o aplicaciones acaparen recursos y ralentiza el sistema.

Estas cuotas definen cuánta memoria puede usar cada tabla, impidiendo que se agoten los recursos disponibles. Así, los administradores pueden manejar mejor el crecimiento y la capacidad de la base de datos, asegurando que el sistema funcione bien para todos los usuarios.

EQUIPO #5

D. ADMINISTRACIÓN DE CUENTAS DE USUARIO.

a. Perfil predeterminado

Un perfil predeterminado en la base de datos es un conjunto de parámetros que definen las políticas de recursos y seguridad que se aplican automáticamente a los nuevos usuarios, a menos que se especifique un perfil diferente. Este perfil actúa como una plantilla inicial para las configuraciones de recursos y seguridad de los usuarios.

Un perfil predeterminado en una base de datos es importante por varias razones:

- Control de Recursos: Ayuda a prevenir el uso excesivo de recursos de la base de datos por parte de los usuarios, estableciendo límites en factores como tiempo de CPU, uso de memoria, y número de conexiones concurrentes.
- Seguridad: Establece un nivel base de seguridad para nuevos usuarios, asegurando que todos tengan los mínimos privilegios necesarios y evitando el acceso excesivo desde el inicio.
- Consistencia: Garantiza que todos los usuarios nuevos tengan un conjunto estándar de privilegios y configuraciones, lo que facilita la administración y el mantenimiento del sistema.
- Eficiencia: Simplifica el proceso de creación de cuentas nuevas, ya que no es necesario configurar cada cuenta individualmente.

Los beneficios de utilizar un perfil predeterminado en términos de gestión de recursos y seguridad incluyen:

- Optimización de Recursos: Los perfiles predeterminados ayudan a prevenir el uso excesivo de recursos al establecer límites en el tiempo de CPU, sesiones simultáneas, y uso de memoria, lo que contribuye a una mejor performance general del sistema.
- Prevención de Abusos: Al limitar los recursos disponibles para los usuarios, se reduce el riesgo de que actividades como consultas mal diseñadas o intentos de ataque consuman demasiados recursos y afecten la disponibilidad del sistema.

- Cumplimiento de Políticas: Al establecer políticas de seguridad y privacidad en los perfiles predeterminados, se facilita el cumplimiento de regulaciones y normativas en relación con la protección de datos sensibles.
- Administración Centralizada: Permite a los administradores gestionar y actualizar políticas para múltiples usuarios de manera centralizada, lo que simplifica la administración y reduce el riesgo de errores.

Creación de un Perfil Predeterminado:

- ❖ Conectar a la base de datos como administrador
- ❖ Ejecutar el comando CREATE PROFILE seguido del nombre que desees para el perfil. Por ejemplo:

```
CREATE PROFILE perfil_personalizado LIMIT
  SESSIONS_PER_USER 5
  CPU_PER_SESSION 600
  CPU_PER_CALL 100
  CONNECT_TIME 180
  IDLE_TIME 30
  LOGICAL_READS_PER_SESSION DEFAULT
  LOGICAL_READS_PER_CALL 1000
  COMPOSITE_LIMIT 5000000;
```

Este comando crea un nuevo perfil llamado Perfil_personalizado con límites específicos para recursos como sesiones por usuario, tiempo de CPU y tiempo de conexión.

- ❖ Asignar el perfil a los usuarios con el comando ALTER USER. Por ejemplo:

```
ALTER USER usuario IDENTIFIED BY contraseña
PROFILE perfil_personalizado;
```

O Asignar el perfil cuando se está creando un usuario con el comando DEFAULT:

```
CREATE USER new_user IDENTIFIED BY password DEFAULT PROFILE default_profile;
```

Modificación de un Perfil Predeterminado:

- Si se necesita ajustar las políticas del perfil predeterminado, se puede utilizar el comando ALTER PROFILE. Por ejemplo:

```
ALTER PROFILE default_profile LIMIT
   SESSIONS_PER_USER 15
   CONNECT_TIME 180;
```

Parámetros utilizados en la creación de Perfiles Predeterminados:

Límites de Uso de CPU y Memoria por Sesión

Parámetro	Descripción
CPU_PER_SESSION	Límite en la cantidad de tiempo de CPU que una sesión puede usar. Medido en centésimas de segundo.
CPU_PER_CALL	Límite en la cantidad de tiempo de CPU que una sola llamada puede usar. Medido en centésimas de segundo.
PRIVATE_SGA	Límite de espacio de memoria SGA privada que una sesión puede usar. Medido en bytes.
LOGICAL_READS_PER_SESSION	Límite en el número de lecturas lógicas (buffers de bloques) que una sesión puede realizar.
LOGICAL_READS_PER_CALL	Límite en el número de lecturas lógicas que una llamada puede realizar.

Tiempo de Conexión y Tiempo de Inactividad Permitido

Parámetro	Descripción
CONNECT_TIME	Límite en el tiempo total que una sesión puede estar conectada. Medido en minutos.
IDLE_TIME	Límite en el tiempo de inactividad permitido para una sesión. Medido en minutos.

Otras Restricciones de Recursos

Parámetro	Descripción
SESSIONS_PER_USER	Límite en el número de sesiones simultáneas que un usuario puede tener.
FAILED_LOGIN_ATTEMPTS	Número máximo de intentos fallidos de inicio de sesión permitidos antes de bloquear la cuenta.
PASSWORD_LIFE_TIME	Tiempo máximo que una contraseña puede ser utilizada antes de expirar. Medido en días.
PASSWORD_REUSE_TIME	Período durante el cual una contraseña no puede ser reutilizada. Medido en días.
PASSWORD_REUSE_MAX	Número máximo de veces que una contraseña antigua puede ser reutilizada.
PASSWORD_VERIFY_FUNCTION	Función PL/SQL utilizada para verificar la complejidad y fortaleza de las contraseñas.
PASSWORD_LOCK_TIME	Tiempo que una cuenta permanece bloqueada después de exceder los intentos fallidos de inicio de sesión. Medido en días.
PASSWORD_GRACE_TIME	Período durante el cual los usuarios pueden cambiar su contraseña después de recibir una advertencia de expiración. Medido en días.

Políticas de Seguridad en el Perfil Predeterminado de Base de Datos

Las políticas de seguridad en el perfil predeterminado de una base de datos son un conjunto de directrices y configuraciones que se aplican para proteger los datos y garantizar que solo usuarios autorizados puedan acceder y manipular la información. Estas políticas abarcan varios aspectos, incluyendo la gestión de contraseñas, la autenticación de usuarios, la auditoría de actividades y la protección contra ataques maliciosos. Los principales aspectos de estas políticas incluyen:

-Longitud Mínima de la Contraseña: Establece el número mínimo de caracteres que debe tener una contraseña para ser considerada válida.

-Complejidad de la Contraseña: Se refiere a la necesidad de incluir una combinación de diferentes tipos de caracteres en una contraseña, tales como letras mayúsculas y minúsculas, números y caracteres especiales.

-Expiración de la Contraseña: Determina el periodo de tiempo tras el cual se debe cambiar una contraseña.

-Historial de Contraseñas: Controla la reutilización de contraseñas anteriores, evitando que los usuarios vuelvan a usar contraseñas viejas.

-Bloqueo de Cuenta y Manejo de Intentos Fallidos de Inicio de Sesión: Establece políticas para el bloqueo de cuentas después de un número específico de intentos fallidos de inicio de sesión y define cómo se manejan estos bloqueos.

-Verificación de Contraseñas y Funciones de Verificación: Uso de funciones específicas para validar las contraseñas y asegurar que cumplen con las políticas de seguridad.

Monitoreo y Auditoría de Perfiles Predeterminados

Para verificar los detalles y configuraciones de un perfil predeterminado en Oracle PL/SQL, se puede utilizar el comando SELECT para consultar la vista del diccionario de datos DBA_PROFILES. Por ejemplo:

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

Este comando mostrara todos los parámetros y sus valores para el perfil DEFAULT.

También se puede consultar la vista 'USER_PROFILES' para obtener información sobre los perfiles asociados con el usuario actual.

```
SELECT PROFILE, RESOURCE_NAME, LIMIT
FROM USER_PROFILES
WHERE PROFILE = 'DEFAULT';
```

Para monitorear el cumplimiento de las políticas de perfiles en Oracle Database, existen varias herramientas y vistas del sistema que pueden ser utilizadas:

- Vistas del Sistema:
  - DBA_PROFILES: Permite ver los límites de recursos establecidos para cada perfil.
  - DBA_USERS: Muestra información sobre los usuarios, incluyendo el perfil que se les ha asignado.
  - DBA_TS_QUOTAS: Permite monitorear las cuotas de espacio en tablespaces asignadas a los usuarios, lo cual es parte de la gestión de perfiles.
  - DBA_USERS_WITH_DEFPWD: Identifica los usuarios de la base de datos que aún tienen la contraseña por defecto. Es útil para verificar el cumplimiento de las políticas de seguridad de contraseñas.
  - DBA_AUDIT_TRAIL: Contiene registros de auditoria para actividades de la base de datos. Se puede utilizar para monitorear eventos relacionados con el cumplimiento de las políticas de perfiles.
  - V$SESSION: Proporciona información sobre las sesiones activas en la base de datos. Puede ser útil para monitorear el uso de recursos en tiempo real.

- Herramientas de Monitoreo:
    - Oracle Enterprise Manager (OEM):
    - Auditoria de Base de Datos: Oracle proporciona capacidades de auditoría que te permiten rastrear y monitorear el uso de recursos y la actividad de los usuarios.
    - Enterprise Manager o Cloud Control: Estas son herramientas gráficas proporcionadas por Oracle que facilitan la monitorización y gestión de perfiles y políticas de seguridad.

Para la administración efectiva de perfiles en la base de datos, es fundamental contar con reportes y alertas que proporcionen una visión clara y en tiempo real del cumplimiento de las políticas de perfiles y el uso de recursos.

1. Reportes
    a. Reportes de Configuración de Perfiles

Un reporte que detalle la configuración de todos los perfiles puede ser útil para revisar y validar las políticas implementadas.

```
SELECT PROFILE, RESOURCE_NAME, LIMIT
FROM DBA_PROFILES
ORDER BY PROFILE, RESOURCE_NAME;
```

    b. Reportes de Uso de Recursos

Generar reportes sobre el uso de recursos por parte de los usuarios y sesiones permite identificar posibles problemas de desempeño y cumplimiento.

```
SELECT USERNAME, SID, VALUE AS CPU_USAGE
FROM V$SESSTAT
JOIN V$STATNAME ON V$SESSTAT.STATISTIC# = V$STATNAME.STATISTIC#
WHERE NAME = 'CPU used by this session'
ORDER BY VALUE DESC;
```

c. Reportes de Sesiones Activas

Monitorear las sesiones activas y su estado es esencial para la administración de recursos.

```
SELECT SID, SERIAL#, USERNAME, STATUS, SCHEMANAME, OSUSER, MACHINE, PROGRAM
FROM V$SESSION;
```

d. Reportes de Auditoría de Sesiones

La auditoría de sesiones y los intentos de inicio de sesión pueden ayudar a identificar problemas de seguridad y cumplimiento.

```
SELECT USERNAME, TIMESTAMP, ACTION_NAME, RETURNCODE
FROM DBA_AUDIT_TRAIL
WHERE ACTION_NAME IN ('LOGON', 'LOGOFF', 'FAILED LOGON');
```

2. Alertas
   a. Oracle Enterprise Manager (OEM): Permite configurar alertas basadas en umbrales específicos para diversos parámetros y eventos.
   b. Scripts de Monitoreo Personalizados: Se puede crear scripts personalizados que se ejecuten periódicamente para verificar condiciones específicas y enviar alertas. Por ejemplo:

```
DECLARE
  v_cpu_usage NUMBER;
  v_email_body VARCHAR2(4000);
BEGIN
  -- Verificar uso de CPU
  SELECT VALUE INTO v_cpu_usage
  FROM V$SESSTAT
  JOIN V$STATNAME ON V$SESSTAT.STATISTIC# = V$STATNAME.STATISTIC#
  WHERE NAME = 'CPU used by this session'
  AND ROWNUM = 1
  ORDER BY VALUE DESC;

  -- Si el uso de CPU excede el umbral, enviar alerta
  IF v_cpu_usage > 10000 THEN
    v_email_body := 'Alerta: El uso de CPU ha excedido el umbral. Uso actual: ' || v_cpu_usage;
    UTL_MAIL.SEND(
      sender => 'dba@example.com',
      recipients => 'admin@example.com',
      subject => 'Alerta de Uso de CPU',
      message => v_email_body
    );
  END IF;
END;
/
```

Este script comprueba el uso de CPU y envía un correo electrónico si se supera un umbral especificado.

c. Uso de DBMS_SCHEDULER para Ejecutar Scripts: Se puede programar la ejecución de scripts de monitoreo utilizando ‘DBMS_SCHEDULER’.

Ejemplo:

```
SQL> DECLARE
  2    v_cpu_usage NUMBER;
  3    v_email_body VARCHAR2(4000);
  4  BEGIN
  5    -- Verificar uso de CPU
  6    SELECT VALUE INTO v_cpu_usage
  7    FROM V$SESSTAT
  8    JOIN V$STATNAME ON V$SESSTAT.STATISTIC# = V$STATNAME.STATISTIC#
  9    WHERE NAME = 'CPU used by this session'
 10    AND ROWNUM = 1
 11    ORDER BY VALUE DESC;
 12
 13    -- Si el uso de CPU excede el umbral, enviar alerta
 14    IF v_cpu_usage > 10000 THEN
 15      v_email_body := 'Alerta: El uso de CPU ha excedido el umbral. Uso actual: ' || v_cpu_usage;
 16      UTL_MAIL.SEND(
 17        sender => 'dba@example.com',
 18        recipients => 'admin@example.com',
 19        subject => 'Alerta de Uso de CPU',
 20        message => v_email_body
 21      );
 22    END IF;
 23  END;
 24  /
```

Practicas recomendadas y casos de uso.

Administrar el perfil predeterminado de una base de datos Oracle implica establecer configuraciones y políticas que garanticen la seguridad, eficiencia y estabilidad de la base de datos. Algunos de las practicas más recomendadas pueden ser las siguientes:

Definir Políticas Claras y Estrictas: Implementar políticas de longitud mínima, complejidad, expiración y reutilización de contraseñas para asegurar que las contraseñas sean seguras, asimismo, configurar el bloqueo de cuentas tras un número determinado de intentos fallidos.

Monitoreo y Auditoría: Habilitar la auditoría para monitorear las actividades de los usuarios y detectar comportamientos sospechosos Y procurar realizar revisiones periódicas de las políticas de seguridad y ajustes en el perfil predeterminado para asegurar que se mantengan efectivas.

Documentación y Capacitación: Mantener una documentación clara y accesible sobre las políticas de perfil predeterminado y las configuraciones de seguridad. Teniendo siempre capacitados a los administradores de bases de datos y usuarios relevantes sobre las políticas de seguridad y las mejores prácticas.

- Casos de uso comunes en diferentes entornos:

Dependiendo el entorno donde se va a utilizar una base de datos las políticas de seguridad de estas pueden variar las dos siguientes son ejemplos:

1. Bases de datos empresariales:

Perfiles más restrictivos: En entornos empresariales, aplica políticas más estrictas. Limita el acceso a funciones sensibles y establece contraseñas fuertes.

Auditoría rigurosa: Realiza un seguimiento exhaustivo de las actividades para cumplir con regulaciones y detectar posibles amenazas.

2. Bases de datos de desarrollo:

Flexibilidad: En entornos de desarrollo, puedes ser más flexible. Los desarrolladores pueden necesitar acceso a funciones adicionales para probar y depurar.

Políticas menos restrictivas: Ajusta las políticas de contraseñas y recursos para facilitar el trabajo de desarrollo.

Impacto y Consecuencias de No Utilizar un Perfil Predeterminado en una Base de Datos

Riesgos Asociados con la Falta de Políticas de Recursos y Seguridad

- Consumo excesivo de recursos: Si no aplicas políticas adecuadas en el perfil predeterminado, los usuarios podrían ejecutar consultas o procesos que consuman más recursos de los necesarios. Esto podría afectar el rendimiento general de la base de datos y ralentizar otras operaciones.
- Desperdicio de recursos: Sin restricciones, los usuarios podrían ocupar más espacio en disco, memoria o CPU del necesario, lo que podría afectar la escalabilidad y la eficiencia.
- Acceso no autorizado: Sin políticas de seguridad adecuadas, las contraseñas pueden ser débiles, lo que facilita a los atacantes el acceso no autorizado provocando que los datos sean sensibles y pueden ser robados, alterados o eliminados.

Gracias a no seguir las practicas dadas al principio se pueden provocar riesgos que pueden escalar peligrosamente a una tragedia que puede afectar una empresa estos son algunos ejemplos:

Consumo Excesivo de Recursos: Un usuario ejecuta una consulta mal optimizada que consume una cantidad excesiva de CPU o memoria, esto genera que el rendimiento de la base de datos se ve gravemente afectado, ralentizando o incluso deteniendo las operaciones para otros usuarios.

Pérdida de Integridad de los Datos: Sin políticas de auditoría, las acciones maliciosas o accidentales no se detectan, los cambios no autorizados en los datos pueden pasar desapercibidos, comprometiendo la integridad de la información.

Acceso No Autorizado y Robo de Datos: Un atacante aprovecha una contraseña débil para acceder a la base de datos, los datos sensibles son exfiltrados, lo que puede llevar a la pérdida de confianza del cliente, daño a la reputación de la empresa y posibles repercusiones legales.

b. Administrar perfiles de límite de recursos

La administración de perfiles de límite de recursos en Oracle SQL es una práctica esencial que permite a los administradores de bases de datos controlar y optimizar el uso de recursos. A través de los perfiles, los administradores pueden establecer restricciones específicas que aseguran un uso equitativo y eficiente de los recursos del sistema. Esto es crucial en entornos donde múltiples

usuarios y aplicaciones compiten por los mismos recursos, garantizando que ninguno consuma más de lo debido y perjudique a otros usuarios o procesos.

En una base de datos Oracle, los recursos como CPU, memoria, E/S (entrada/salida) y sesiones de usuario son compartidos entre múltiples usuarios y aplicaciones. Sin una gestión adecuada, algunos usuarios o procesos pueden consumir recursos excesivos, afectando negativamente el rendimiento y la disponibilidad del sistema para otros usuarios. Los perfiles de recursos permiten establecer límites precisos para prevenir estos problemas.

Definición detallada de perfiles, límites y asignaciones en el contexto de Oracle SQL.

Un perfil es un conjunto de parámetros y límites que se puede aplicar a uno o más usuarios. Los perfiles se definen con el comando CREATE PROFILE.

- Tipos de Límites:
    - Límites de Sesión: Controlan aspectos como el número máximo de sesiones simultáneas (SESSIONS_PER_USER) y el tiempo máximo que una sesión puede estar conectada (CONNECT_TIME).
    - Límites de CPU: Establecen el tiempo máximo de CPU que una sesión o llamada puede utilizar (CPU_PER_SESSION y CPU_PER_CALL).
    - Límites de Memoria: Regulan la cantidad de memoria SGA privada que puede usar una sesión (PRIVATE_SGA).
    - Límites de E/S: Definen el número máximo de lecturas lógicas que una sesión o llamada puede realizar (LOGICAL_READS_PER_SESSION y LOGICAL_READS_PER_CALL).
    - Límites de Contraseñas: Incluyen parámetros como la duración máxima de una contraseña antes de que deba ser cambiada (PASSWORD_LIFE_TIME) y el número de veces que una contraseña puede ser reutilizada (PASSWORD_REUSE_MAX).

- Asignación de Perfiles: Una vez que un perfil ha sido definido, se puede asignar a usuarios específicos mediante el comando ALTER USER.

Importancia y beneficios de la administración de límites de perfiles.

- Control de Recursos: Permite a los administradores asegurar que ningún usuario o proceso consuma recursos excesivos, garantizando un rendimiento equilibrado para todos.
- Optimización del Rendimiento: Estableciendo límites, se puede prevenir que aplicaciones o usuarios individuales monopolicen los recursos, manteniendo un rendimiento óptimo.
- Seguridad: Los límites en las contraseñas y sesiones inactivas ayudan a proteger la base de datos contra accesos no autorizados y posibles ataques de fuerza bruta.
- Estabilidad del Sistema: Al controlar el uso excesivo de recursos, los perfiles contribuyen a la estabilidad y disponibilidad del sistema.

Diferencias entre la administración de límites de perfiles y otros mecanismos de control de recursos, como roles y privilegios.

En Oracle SQL, la administración de perfiles de recursos es solo una de las formas en que se puede controlar el acceso y uso de la base de datos. Otros mecanismos incluyen roles y privilegios.

- Roles:

  En Oracle son conjuntos de privilegios agrupados que se pueden asignar a usuarios o a otros roles. Los roles simplifican la administración de permisos al permitir que los privilegios se gestionen de manera colectiva en lugar de individual.
    - Funcionalidad:
        - Agrupación de Privilegios: Un rol puede contener múltiples privilegios de sistema y de objeto. Esto permite asignar un conjunto coherente de permisos a varios usuarios de una sola vez.
        - Herencia: Los roles pueden incluir otros roles, lo que permite una estructura jerárquica de permisos.
- Privilegios:

  Son permisos específicos otorgados a usuarios o roles para realizar determinadas operaciones en la base de datos. Existen dos tipos principales de privilegios:
    - Privilegios de Sistema:

        - Permiten realizar acciones que afectan al sistema de la base de datos en su totalidad, como CREATE USER, DROP DATABASE, ALTER SYSTEM.
        - Son esenciales para la administración y configuración de la base de datos.
    - Privilegios de Objeto:
        - Permiten realizar acciones específicas sobre objetos de la base de datos, como tablas, vistas, procedimientos, etc. Ejemplos incluyen SELECT, INSERT, UPDATE, DELETE en una tabla determinada.
        - Controlan el acceso y modificación de los datos a nivel de objeto.

Creación y gestión de perfiles

Las bases de datos tienen una lista válida de usuarios a los que se le permite la conexión al sistema. Por ese motivo se debe crear una cuenta para cada usuario en la que se especifique: nombre de usuario, método de autentificación, tablespace permanente o temporal y perfil de usuario. Vamos a diferenciar las dos formas por las que se puede autentificar un usuario:

- **Autentificación por base de datos:** la administración de la cuenta es de tipo usuario/contraseña, de forma que se va a guardar encriptada y su autentificación se realizada por Oracle.
- **Autentificación externa:** la cuenta la mantiene Oracle, aunque la administración de la contraseña y la autentificación de usuario es realizada externamente por el sistema operativo. El usuario tiene la posibilidad de conectarse a la base de datos sin necesidad de indicar el nombre del usuario o la clave (es el mismo nombre de usuario del S.O.).

Características de los usuarios de Oracle

Los usuarios, como hemos indicado anteriormente, deben poseer un nombre. Veamos las restricciones que existen a la hora de su creación:

- Nombre usuario: debe ser único e irrepetible. Su longitud máxima no debe sobrepasar los 30 caracteres. Además, solamente puede contener caracteres alfanuméricos y los signos '$' y '_' como caracteres especiales.
- Configuración física: espacio que posee el usuario para almacenar su información y límite de almacenamiento. En Oracle se denomina tablespace.

- Perfil asociado: son los diferentes recursos de los que dispone el usuario del sistema.
- Privilegios y roles: concesión de funciones que pueden realizar los usuarios.

Estado de una cuenta de usuario

- Abierta: el usuario puede trabajar sin problema en las acciones habilitadas.
- Bloqueada: el usuario no puede realizar ninguna acción mientras que se encuentre en este estado.
- Expirada: la cuenta de usuario ha agotado el tiempo máximo del que disponía.
- Expirada y bloqueada.
- Expirada en periodo de gracia: está en los últimos momentos de uso antes de pasar a estado de expirada.

Comandos SQL para crear, modificar y eliminar perfiles.

- **Creación de usuario**

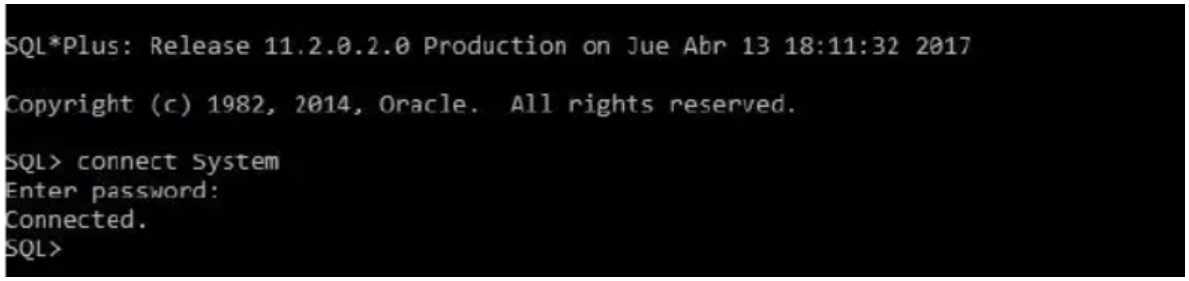
Antes de comenzar con la creación de usuarios, debemos asegurarnos de que nos conectamos a un usuario que posee permisos concedidos para la creación de usuarios.

Existen dos formas diferentes para conectarnos a un usuario:

1. Mediante la creación de una conexión.

Como hemos visto en el apartado anterior con el usuario SYSTEM.

2. Mediante comandos en Oracle (Run SQL Command Line).



La sentencia para crear una cuenta de usuario que permita la autentificación de este en el SGBD con un nivel determinado de privilegios es:

CÓDIGO:

```
-- Sintaxis
CREATE USER nombre_usuario IDENTIFIED BY 'passwords' [opciones];
```

CÓDIGO:

```
-- Ejemplo
CREATE USER ilerna IDENTIFIED BY 'root';
```

Para saber a qué estamos conectados utilizaremos el siguiente comando:

CÓDIGO:

```
-- Ejemplo
SHOW USER
```

Si estamos trabajando con las hojas de trabajo del cliente de Oracle SQL Developer, hay que tener en cuenta que, al crear un usuario y conectarnos a él, una vez terminemos el script, este se desconectará automáticamente. En el caso de estar utilizando la consola de comandos, este usuario se quedará conectado hasta que cerremos la conexión o la consola.

Sentencia de creación y conexión:

```
CREATE USER ilerna IDENTIFIED BY 'root';
GRANT CREATE SESSION TO ilerna;
CONN ilerna / root
```

Cuando se crea un usuario también podemos elegir las siguientes opciones:

CÓDIGO:

```
-- Sintaxis
CREATE UER nombre {IDENTIFIED BY 'contraseña'|
EXTERNALLY |  GLOBALLY AS nombreGlobal}
[DEFAULT TABLESPACE tableSpacePorDefecto]
[TEMPORARY TABLESPACE tableSpacetTemporal]
[QUOTA {cantidad [K|M] | UNLIMITED} ON tablespace
[QUOTA {cantidad [K|M] | UNLIMITED} ON tablespace [...]] ]
[PASSWORD EXPIRE]
[ACCOUNT {UNLOCK|LOCK}];
[PROFILE {perfil | DEFAULT}]
```

CÓDIGO:

```
-- Ejemplo
CREATE USER ilerna IDENTIFIED BY 'root'
DEFAULT TABLESPACE 'Alumnos'
QUOTA 15M ON 'Alumnos'   /*Se dan 15MBytes de espacio en el tablespace*/
```

- **Modificación de usuario**

CÓDIGO:

```
-- Sintaxis
ALTER USER nombre_usuario [opciones];
-- Ejemplo
ALTER USER ilerna ACCOUNT UNLOCK;
```

Para quitarle el límite de cuota:

CÓDIGO:

```
-- Ejemplo
ALTER USER ilerna QUOTA UNLIMITED ON Alumnos;
```

- **Borrado de usuario**

CÓDIGO:

```
-- Sintaxis
DROP USER nombre_usuario [CASCADE];
```

La opción CASCADE elimina primero los objetos que están asociados al usuario y después el usuario.

CÓDIGO:

```
-- Ejemplo
DROP USER ilerna;
```

Cada vez que borremos un usuario, borraremos también el esquema asociado a sus

objetos.

Una vez que ya hemos visto la creación, modificación y borrado de usuarios, pasaremos a ver cómo realizar consultas sobre usuarios ya creados.

Mediante DBA_USERS se muestra la lista y configuración de los usuarios del sistema. Es conveniente utilizar DESCRIBE DBA USERS para visualizar su estructura.

CÓDIGO:

```
-- Ejemplo
DESC DBA_USERS;
```

Opciones de límite de recursos

Los perfiles de límite de recursos admiten una amplia gama de opciones para definir límites para diferentes tipos de recursos del sistema:

- **Tiempo de CPU**: La cantidad total de tiempo de CPU que un usuario o proceso puede consumir en un período determinado.
- **Tiempo de conexión**: La cantidad total de tiempo que un usuario puede estar conectado a la base de datos.

- **Sesiones por usuario**: El número máximo de sesiones simultáneas que un usuario puede tener.
- **Tiempo de inactividad**: La cantidad de tiempo máximo que una sesión puede estar inactiva antes de ser terminada automáticamente.
- **Espacio en disco**: La cantidad total de espacio en disco que un usuario puede consumir.
- **Eventos de E/S**: El número máximo de eventos de E/S que un usuario o proceso puede generar en un período determinado.

Consideraciones para establecer límites de recursos

Al establecer límites de recursos, es importante considerar las siguientes características del sistema y las necesidades de los usuarios:

- **Carga del sistema**: La carga general del sistema y la cantidad de recursos disponibles.
- **Tipos de usuarios**: Los tipos de usuarios que acceden a la base de datos y sus necesidades de recursos.
- **Aplicaciones**: Las aplicaciones que se ejecutan en la base de datos y sus requisitos de recursos.
- **Impacto en el rendimiento**: El impacto potencial de los límites de recursos en el rendimiento del sistema y de las aplicaciones.

Subtemas para la investigación sobre la administración de límites de perfiles en Oracle SQL:

3. Asignación y revocación de perfiles:

- Comandos SQL para asignar y revocar perfiles a usuarios y grupos y Casos de uso para la asignación de diferentes perfiles a distintos tipos de usuarios.

Asignar perfiles a usuarios en Oracle SQL

Para asignar un perfil a un usuario individual:

GRANT perfil TO usuario;

Ejemplo:

```
GRANT HR TO scott;
```

Para asignar un perfil a un grupo de usuarios:

```
GRANT perfil TO grupo;
```

Ejemplo:

```
GRANT HR TO managers;
```

Opciones adicionales:

WITH ADMIN OPTION: Permite al usuario conceder el perfil a otros usuarios.

WITH GRANT OPTION: Permite al usuario conceder el perfil a otros usuarios, con la opción de conceder la opción ADMIN.

Ejemplo:

```
GRANT HR TO scott WITH ADMIN OPTION;
```

Revocar perfiles a usuarios en Oracle SQL

Para revocar un perfil de un usuario individual:

```
REVOKE perfil FROM usuario;
```

Ejemplo:

```
REVOKE HR FROM scott;
```

Para revocar un perfil de un grupo de usuarios:

```
REVOKE perfil FROM grupo;
```

Ejemplo:

```
REVOKE HR FROM managers;
```

Opciones adicionales:

CASCADE: Revoca el perfil del usuario y de cualquier otro usuario al que se le haya concedido mediante la opción WITH GRANT OPTION.

Ejemplo:

```
REVOKE HR FROM scott CASCADE;
```

Impacto de la revocación de un perfil en las capacidades de un usuario

La revocación de un perfil de un usuario en Oracle SQL puede tener un impacto significativo en sus capacidades dentro del sistema. Un perfil encapsula un conjunto de privilegios y roles que definen lo que un usuario puede hacer dentro de la base de datos. Al revocar un perfil, se elimina el acceso a todos los privilegios y roles asociados a ese perfil.

Consecuencias de revocar un perfil:

Pérdida de acceso a objetos: Si el perfil concede acceso a tablas, vistas, procedimientos, funciones u otros objetos de la base de datos, el usuario perderá la capacidad de interactuar con

esos objetos. Esto puede impedirle realizar tareas como consultar datos, insertar o modificar registros, o ejecutar procedimientos.

Restricción de acciones: Si el perfil concede privilegios específicos, como la capacidad de crear objetos, otorgar privilegios a otros usuarios o ejecutar comandos administrativos, el usuario perderá la posibilidad de realizar esas acciones.

Limitación de roles: Si el perfil asigna roles específicos al usuario, estos roles y sus privilegios asociados también se revocarán. Esto puede afectar la capacidad del usuario para realizar tareas dentro de áreas específicas del sistema.

Ejemplos del impacto de la revocación de un perfil:

Si se revoca el perfil HR a un usuario, este ya no podrá acceder a las tablas de recursos humanos, ni podrá contratar o despedir empleados.

Si se revoca el perfil DBA a un usuario, este ya no podrá crear usuarios, otorgar privilegios o realizar otras tareas administrativas en la base de datos.

Si se revoca el perfil SALES a un grupo de usuarios, estos ya no podrán consultar los datos de ventas ni realizar pedidos.

Consideraciones adicionales:

La revocación de un perfil puede tener un impacto en cascada, ya que puede afectar a otros usuarios a los que el usuario revocado haya concedido privilegios mediante la opción WITH GRANT OPTION.

Es importante evaluar cuidadosamente las consecuencias de revocar un perfil antes de hacerlo, ya que puede tener un impacto significativo en la capacidad del usuario para trabajar dentro del sistema.

Se recomienda utilizar la revocación de perfiles con moderación y solo cuando sea necesario para restringir el acceso a la base de datos o para cumplir con requisitos de seguridad específicos.

EQUIPO #6

E. RESPALDO Y RECUPERACIÓN DE UNA BASE DE DATOS

a. Importación de datos y objetos

La importación de objetos y datos en la recuperación de bases de datos es un proceso fundamental para restaurar el estado de una base de datos después de una falla o pérdida de datos. Este proceso implica extraer objetos y datos de una copia de seguridad o de un origen externo y colocarlos en una base de datos existente o recién creada. Esta guía proporciona una revisión exhaustiva de la importación de objetos y datos, incluyendo sus beneficios, aplicaciones, métodos comunes y consideraciones importantes.

La importación de datos se refiere al proceso de obtener o recuperar datos desde una fuente externa, como una base de datos, un archivo de texto, una aplicación o un sistema de información, y llevarlos a una aplicación, sistema o base de datos local para su procesamiento, análisis o almacenamiento.

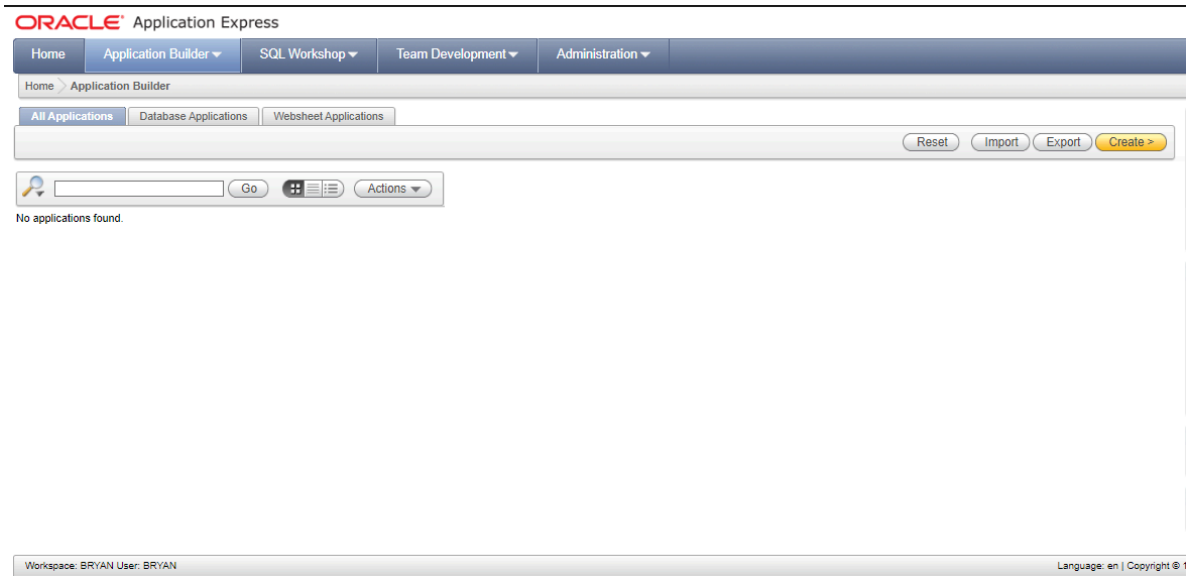
La importación de datos puede involucrar varios pasos, como:

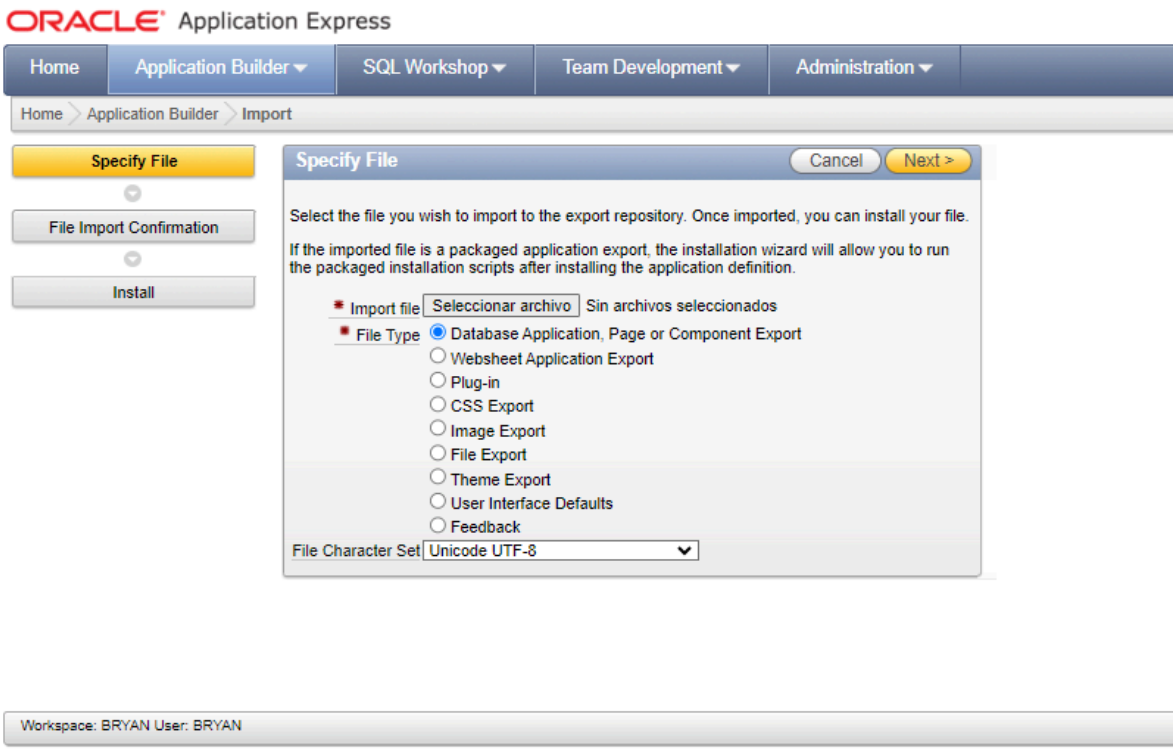
- Recopilación: Recopilar los datos desde la fuente externa.
- Transformación: Convertir los datos en un formato compatible con el sistema o aplicación local.
- Almacenamiento: Almacenar los datos en la base de datos o archivo local.

La importación de datos es común en diversas áreas, como:

- Análisis de datos: Importar datos para realizar análisis estadísticos, visualización de datos o minería de datos.
- Integración de sistemas: Importar datos de un sistema o aplicación para integrarlos con otro sistema o aplicación.
- Automatización de procesos: Importar datos para automatizar procesos empresariales, como la gestión de inventarios o la facturación.

La importación de datos es un proceso importante en la gestión de datos y puede ser realizada utilizando herramientas y tecnologías específicas, como APIs, ETL (Extract, Transform, Load), o herramientas de importación de datos específicas para cada aplicación o sistema.





Importación de objetos y datos en PL/SQL: Ventajas y desventajas

Ventajas:

- Facilidad de uso: PL/SQL ofrece una sintaxis sencilla y directa para importar objetos y datos, lo que la hace accesible para usuarios con diferentes niveles de experiencia.
- Flexibilidad: PL/SQL permite importar datos desde una variedad de fuentes, incluyendo archivos planos, bases de datos y aplicaciones empresariales.
- Control granular: PL/SQL ofrece un control preciso sobre el proceso de importación, lo que permite a los usuarios especificar qué datos se importan, cómo se transforman y dónde se almacenan.
- Automatización: Las tareas de importación de PL/SQL se pueden automatizar mediante scripts, lo que las hace ideales para procesos repetitivos.

Desventajas:

- Complejidad: Importar objetos y datos complejos con PL/SQL puede requerir código extenso y detallado.
- Rendimiento: La importación de grandes volúmenes de datos con PL/SQL puede ser lenta, especialmente si los datos no están formateados adecuadamente.
- Dependencia de PL/SQL: La importación de objetos y datos con PL/SQL requiere que los usuarios tengan conocimientos de este lenguaje, lo que puede limitar su uso.

Tipos de importaciones en PL/SQL

Existen dos tipos principales de importaciones en PL/SQL:

1. Importación de datos: Este tipo de importación se utiliza para cargar datos en tablas de Oracle desde una fuente externa. PL/SQL ofrece varios métodos para importar datos, incluyendo:

- INSERT: Inserta datos directamente en una tabla.
- LOAD DATA: Carga datos de un archivo plano en una tabla.
- SQL*Loader: Una herramienta especializada para la carga de datos a gran escala.

2. Importación de objetos: Este tipo de importación se utiliza para crear o reemplazar objetos de base de datos, como tablas, vistas, procedimientos y paquetes. PL/SQL ofrece el comando CREATE para importar objetos de base de datos.

Beneficios de la importación de objetos y datos

La importación de objetos y datos ofrece diversos beneficios para la recuperación de bases de datos:

- **Restauración rápida:** Permite restaurar rápidamente una base de datos a un estado anterior, minimizando el tiempo de inactividad y el impacto en las operaciones comerciales.
- **Recuperación granular:** Permite restaurar objetos y datos específicos, en lugar de toda la base de datos, lo que reduce el tiempo y los recursos necesarios para la recuperación.
- **Flexibilidad:** Permite importar objetos y datos desde diversas fuentes, incluyendo copias de seguridad, bases de datos existentes y archivos de texto.
- **Escalabilidad:** Admite la recuperación de bases de datos de gran tamaño y complejas.
- **Protección de datos:** Garantiza la integridad y disponibilidad de los datos críticos en caso de fallos del sistema o errores humanos.

Aplicaciones de la importación de objetos y datos

La importación de objetos y datos se utiliza en diversas situaciones, como:

- **Recuperación de desastres:** Para restaurar una base de datos después de un desastre natural o un fallo del sistema.
- **Migración de datos:** Para mover datos entre diferentes plataformas de bases de datos o entornos.
- **Pruebas y desarrollo:** Para crear entornos de prueba o desarrollo con datos reales.
- **Análisis forense:** Para investigar incidentes de seguridad o corrupción de datos.
- **Restauración de datos eliminados accidentalmente:** Para recuperar datos que se han eliminado por error.

Métodos comunes para la importación de objetos y datos

Existen diversos métodos para importar objetos y datos en una base de de datos:

- **Herramientas de administración de bases de datos:** La mayoría de los sistemas de gestión de bases de datos (SGBD) proporcionan herramientas integradas para la importación y exportación de datos. Estas herramientas suelen ofrecer interfaces

gráficas de usuario (GUI) fáciles de usar y opciones configurables para personalizar el proceso de importación.

- **Utilidades de línea de comandos:** Algunos SGBD también ofrecen utilidades de línea de comandos para la importación y exportación de datos. Estas utilidades pueden ser útiles para automatizar tareas de importación o para realizar importaciones complejas que requieren opciones avanzadas.
- **Scripts y lenguajes de programación:** Los scripts y lenguajes de programación, como SQL o Python, se pueden utilizar para crear scripts personalizados para la importación de datos. Esta opción ofrece mayor flexibilidad y control sobre el proceso de importación, pero requiere conocimientos de programación.

Consideraciones importantes para la importación de objetos y datos

Al importar objetos y datos en una base de datos, es importante tener en cuenta las siguientes consideraciones:

- **Compatibilidad:** Asegúrese de que los objetos y datos que se importan sean compatibles con el esquema de la base de datos de destino.
- **Validación de datos:** Valide los datos que se importan para garantizar su integridad y consistencia.
- **Conflictos de datos:** Identifique y resuelva posibles conflictos de datos que puedan surgir durante la importación.
- **Seguridad:** Implemente medidas de seguridad adecuadas para proteger los datos durante el proceso de importación.
- **Documentación:** Documente el proceso de importación para facilitar futuras restauraciones o migraciones.

Ejemplos de importación de datos, mediante scripts y líneas de comandos

Para los siguientes ejemplos, se usará la tabla:

```
CREATE TABLE empleados (
    id      NUMBER,
```

```
    nombre          VARCHAR2(50),
    departamento VARCHAR2(50),
    salario         NUMBER
);
```

Método Insert:

[

```
BEGIN
    INSERT INTO empleados (id, nombre, departamento, salario)
    VALUES (1, 'Juan Perez', 'IT', 5000);
    INSERT INTO empleados (id, nombre, departamento, salario)
    VALUES (2, 'Ana Gomez', 'HR', 4500);
    INSERT INTO empleados (id, nombre, departamento, salario)
    VALUES (3, 'Carlos Ruiz', 'Finance', 6000);
    COMMIT;
END;
```

]

Este método no utiliza un archivo externo, sino que es un método directo para insertar datos a la tabla. Se puede usar en cualquier SGBD con SQL

Método LOAD DATA:

[

LOAD DATA INFILE 'empleados.csv'

INTO TABLE empleados

FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY ""

(id, nombre, departamento, salario);

]

Esta vez, el archivo del que estamos importando es “empleados.csv”. Indicamos la tabla después del “into table”, determinamos dónde acaba un dato y dónde empieza otro con “fields terminated by ‘,’”, donde la , (coma) indica dónde acaba. Este método es específico de MySQL, por lo que no se puede usar en Oracle.

Método SQL*Loader:

[

LOAD DATA

INFILE 'empleados.csv'

INTO TABLE empleados

INSERT

FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY ""

(

    id   INTEGER EXTERNAL,

    nombre   CHAR,

    departamento CHAR,

    salario   INTEGER EXTERNAL

)

]

Este código se escribe en un archivo aparte, con terminación .ctl. Este método es exclusivo de Oracle. Al igual que el método anterior, está importando desde el archivo empleados.csv a la tabla empleados.

EQUIPO #7

1. Definición y Propósito de la Exportación de Datos

¿Qué es la exportación de datos?

La exportación de datos es el proceso de extraer datos de una base de datos y almacenarlos en un formato que se puede transferir a otra ubicación, sistema o formato de almacenamiento. Esto implica tomar una instantánea de los datos y objetos de la base de datos (como tablas, vistas, esquemas, etc.) y guardarla en un archivo externo.

Importancia

- Seguridad y Respaldo: La exportación de datos es crucial para crear copias de seguridad de la base de datos. En caso de un fallo del sistema, pérdida de datos o corrupción, se puede restaurar la base de datos a partir de estos archivos exportados.

- Migración y Transferencia: Permite mover los datos entre diferentes sistemas, versiones de bases de datos o también entre diferentes entornos como producción, desarrollo y pruebas.

- Análisis y Reportes: Los datos exportados se pueden usar para análisis externo, generación de informes o para alimentar sistemas de análisis de datos.

- Compatibilidad e Interoperabilidad: La compatibilidad ayuda en la integración de datos entre sistemas heterogéneos que pueden no estar directamente conectados entre sí.

Casos de uso comunes y ejemplos

- Migración a una nueva versión de base de datos: Al actualizar por ejemplo de un sistema como Oracle 11g a 12c, se puede exportar la base de datos y luego importarla en el nuevo sistema.

- Creación de entornos de prueba: Exportar datos de producción para cargar un entorno de prueba con datos realistas.

- Compartición de datos: Exportar datos para compartir con otras organizaciones o equipos que utilizan diferentes sistemas de bases de datos.

2. Métodos y Herramientas para la Exportación de Datos

Herramientas y software comúnmente utilizados

- Oracle Data Pump: Una herramienta de exportación e importación de alto rendimiento para bases de datos Oracle, que incluye utilidades como `expdp` (exportación) e `impdp` (importación).

- Oracle SQL Developer: Un entorno de desarrollo integrado que proporciona herramientas gráficas para exportar datos y objetos de la base de datos.

- Oracle Export Utility (`exp`): Una herramienta más antigua y menos recomendada para nuevas aplicaciones, pero aún utilizada en sistemas heredados.

Métodos Manuales vs. Automatizados

Manuales: Los métodos manuales implican ejecutar comandos directamente desde la línea de comandos o herramientas como SQL*Plus. Requieren intervención humana para iniciar y supervisar cada proceso de exportación de datos. Son útiles para exportaciones para un fin determinado o situaciones donde se necesita control directo sobre el proceso de exportación.

Automatizados: Los métodos automatizados se basan en la programación de tareas mediante herramientas como Oracle Scheduler y procedimientos PL/SQL. Estos métodos ejecutan exportaciones de datos de manera programada según horarios predefinidos. Es ideal para realizar exportaciones regulares y automáticas, minimizando la intervención humana y asegurando la consistencia en la ejecución de tareas.

Scripts y Comandos Específicos para la Exportación de Datos

En Oracle Database, la herramienta principal para la exportación de datos se llama expdp. A continuación, se explican los comandos específicos que se utilizan con expdp:

1. expdp

· **Descripción**: expdp es el comando principal utilizado para iniciar una operación de exportación de datos en Oracle Database.

· **Uso**: Se utiliza para exportar datos, metadatos y definiciones de objetos desde la base de datos

2. SCHEMAS

· **Descripción**: Parámetro que especifica los esquemas de la base de datos que se van a exportar.

· **Ejemplo**:

expdp hr/password@XE SCHEMAS=hr

Este comando exporta todos los objetos del esquema HR.

3. DIRECTORY

· **Descripción**: Parámetro que indica el nombre del directorio de Oracle Data Pump donde se guardarán los archivos de exportación.

· **Ejemplo**:

expdp hr/password@XE DIRECTORY=dpump_dir1

4. DUMPFILE

· **Descripción**: Parámetro que especifica el nombre del archivo de volcado que se generará durante la exportación.

· **Ejemplo**:

expdp hr/password@XE DUMPFILE=hr_export.dmp

Este comando generará un archivo de volcado llamado hr_export.dmp que contendrá los datos exportados del esquema HR.

5. LOGFILE

· **Descripción**: Parámetro que especifica el nombre del archivo de registro donde se registrarán los mensajes y el progreso de la operación de exportación.

· **Ejemplo**:

expdp hr/password@XE LOGFILE=hr_export.log

Este comando generará un archivo de registro llamado hr_export.log que contendrá los registros detallados de la exportación.

6. OTROS PARÁMETROS OPCIONALES

· TABLES: Para exportar tablas específicas.

· QUERY: Para exportar datos basados en una consulta SQL.

· JOB_NAME, JOB_INSTANCE, entre otros: Para gestionar y controlar operaciones de exportación más complejas y automatizadas.

3. Formatos de Exportación de Datos

a) **Archivos de Script SQL** (.sql)

VENTAJAS

Los scripts SQL pueden ser ejecutados en diferentes entornos de bases de datos Oracle sin necesidad de cambios significativos.

Son fáciles de crear y entender, especialmente para tareas de definición y manipulación de datos.

DESVENTAJAS

Los scripts SQL pueden volverse desordenados y difíciles de mantener si contienen muchas instrucciones.

Los scripts pueden contener información sensible como contraseñas si no se manejan adecuadamente.

CASO DE USO: Cuando una empresa necesita migrar datos de un sistema antiguo a un nuevo sistema basado en Oracle, se pueden escribir scripts SQL para extraer, transformar y cargar los datos en la nueva base de datos.

b) **Archivos de Procedimiento PL/SQL** (.pls)

VENTAJAS

Los procedimientos almacenados pueden mejorar el desempeño al reducir el tráfico entre la aplicación y la base de datos.

Facilitan el mantenimiento y actualización del código

DESVENTAJAS

La creación y mantenimiento de procedimientos PL/SQL pueden ser más complejos comparados con scripts SQL simples

CASO DE USO: En una cadena de tiendas, los procedimientos PL/SQL pueden ser utilizados para automatizar la actualización de inventarios, realizar pedidos automáticos cuando el stock es bajo y generar informes de inventario.

c) **Archivos de Paquetes PL/SQL** (.pks y .pkb)

VENTAJAS

La carga completa del paquete en la memoria puede consumir más recursos, especialmente si el paquete es grande

DESVENTAJAS

La creación y mantenimiento de procedimientos PL/SQL pueden ser más complejos comparados con scripts SQL simples

CASO DE USO: En un sistema de nómina, los paquetes PL/SQL pueden gestionar el procesamiento por lotes de salarios, deducciones y pagos, asegurando que todos los cálculos se realicen de manera eficiente y segura.

d) **Archivos de Control de Datos** (.ctl)

VENTAJAS

Los archivos de texto son fáciles de crear, leer y manipular con herramientas básicas.

Son ampliamente compatibles con diferentes sistemas y aplicaciones, facilitando el intercambio de datos.

DESVENTAJAS

Los archivos de texto pueden no ser seguros para almacenar información sensible si no se protegen adecuadamente.

CASO DE USO: Una compañía de telecomunicaciones utiliza archivos de control para cargar grandes volúmenes de datos de clientes desde sistemas externos a su base de datos central, manteniendo la información actualizada y precisa.

e) **Archivos de Texto** (.txt)

VENTAJAS

Los archivos .ctl permiten definir cómo se deben cargar los datos en la base de datos desde archivos externos, ofreciendo un alto grado de control sobre el proceso.

Pueden ser utilizados con SQL*Loader para automatizar el proceso de carga de grandes volúmenes de datos.

DESVENTAJAS

La creación y configuración de archivos .ctl puede ser complicada para los usuarios nuevos

CASO DE USO: En un proyecto colaborativo, diferentes equipos pueden intercambiar datos de configuración y resultados de pruebas en archivos de texto simples para facilitar la comunicación y coordinación.

4. Procedimientos y Mejores Prácticas

Este es un ejemplo de exportación de datos a un fichero plano desde SQLPlus

```
SQL> SET HEADING OFF
 SQL> SET FEEDBACK OFF
 SQL> SPOOL C:\datos_de_clientes.txt
   SQL> SELECT 'Cliente ' || CLI_NOMBRE || ', ' || CLI_NIF || '. Fecha alta: ' ||
TO_CHAR(CLI_FECHAALTA,'YYYY-MM-DD')
        FROM TABLA_CLIENTES
        ORDER BY CLI_FECHAALTA DESC;
 SQL> SPOOL OFF;
 SQL> SET FEEDBACK ON
 SQL> SET HEADING ON
```

- Prácticas recomendadas para garantizar la integridad y seguridad de los datos exportados.

· Realizar Backups Regulares

· Verificación y Validación de Datos

· Uso de Transacciones

· Encriptación de Datos

5. Seguridad y Cumplimiento

Medidas de seguridad durante la exportación

Para proteger la integridad de los datos existen algunas medidas de seguridad que se recomiendan:

- Una buena práctica es asegurarse de que la exportación de datos solo pueda ser realizada por los usuarios autorizados definiendo roles con permisos específicos. Además, se deberían implementar contraseñas seguras que cambien periódicamente.
- Habilitar el registro de todas las exportaciones para saber quién hizo cada una y configurar alertas para actividades inusuales.
- Aplicar protocolos de cifrado seguro como SSL/tLS mientras se exporta.
- Mantener los entornos de base de datos y exportación en segmentos de red separados.
- Adicionalmente, es recomendable proteger las conexiones de red usando firewalls y VPNs para prevenir accesos no autorizados.

Cumplimiento de normativas & regulaciones

Es importante mantener el cumplimiento de ciertas regulaciones para garantizar que la gestión de datos que se practica sea tanto segura como legal. Algunas de las principales normativas son las siguientes:

- **GDPR:** Se centra en protección de datos personales de ciudadanos para ciudadanos de la unión europea. Trata con asuntos de consentimiento para el procesamiento de datos personales y derechos de usuarios.
- **HIPAA:** Es una normativa que trata con la protección de datos pertenecientes al área de salud protegida, políticas de seguridad y privacidad y evaluación de riesgos.
- **SOX:** Su principal foco es la transparencia en los informes financieros, controles internos y auditorías regulares.

Cabe mencionar que estas regulaciones solo aplican en ciertas regiones, pero deberían ser puestas en acción para mantener buenas prácticas de manejo de una base de datos.

Manejo de datos sensibles

Para proteger la privacidad de la información de datos sensibles existen ciertas prácticas que resultan útiles:

- Identificación y clasificación de datos sensibles para manejar de manera adecuada la protección de datos como información personal, datos de salud, datos financieros, etc.
- Aplicar técnicas de mascarado para ocultar datos sensibles durante la exportación. De esta forma se pueden transformar los datos de manera que los individuos no puedan ser identificados.
- Mantener registros detallados de las actividades relacionadas a la exportación, así como identificación de la persona responsable de cualquier manipulación con los datos en cuestión. Además es recomendable implementar sistemas de monitorea para detectar cualquier acceso no autorizado.

  La implementación de estas prácticas aportará en la protección de la privacidad de datos sensibles al eliminar varios escenarios no deseados.

6. Casos Prácticos y Ejemplos Reales

Caso de Uso: Exportación de Datos de Estudiantes en la Escuela Pedro Pablo Sánchez

Objetivo: Exportar el número de identificación, nombre y apellido de los estudiantes en la Escuela Pedro Pablo Sánchez a un archivo de texto plano.

Pre-Requisitos

```
GRANT EXECUTE ON UTL_FILE TO nombreUsuario;

CREATE OR REPLACE DIRECTORY export_dir AS 'DireccionDeCarpeta';

GRANT READ, WRITE ON DIRECTORY export_dir TO nombreUsuario;
```

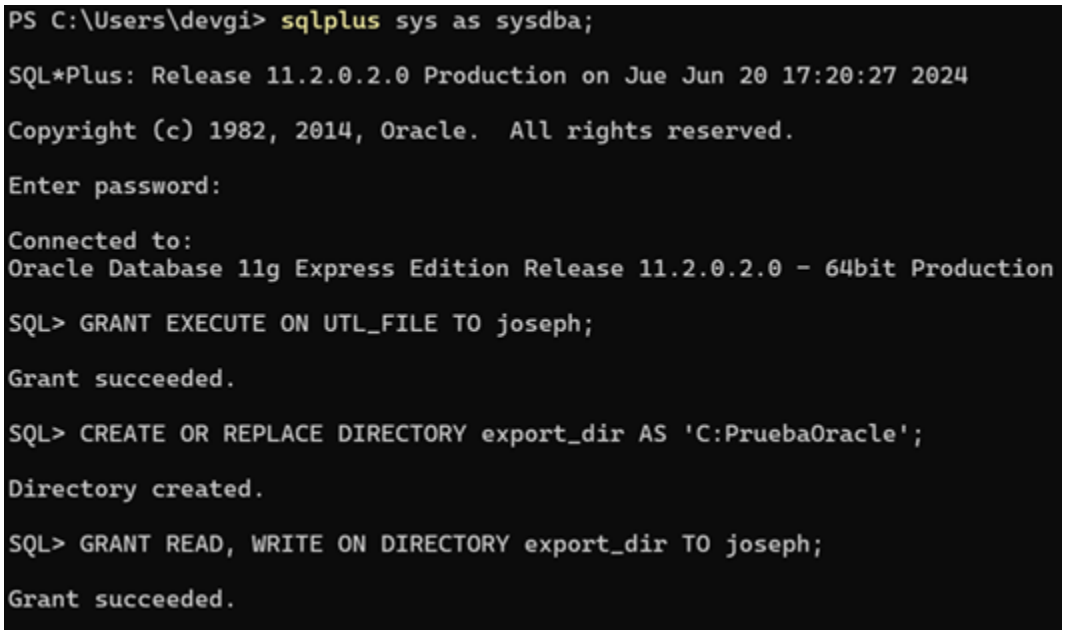
Método#1 Utilizando Archivos txt

```
DECLARE
  l_file UTL_FILE.FILE_TYPE;
  l_line VARCHAR2(32767);
BEGIN
  -- Abrir el archivo para escribir
  l_file := UTL_FILE.FOPEN('EXPORT_DIR', nombrearchivo.txt', 'w');

  -- Seleccionar los datos de la tabla y escribir en el archivo
  FOR rec IN (SELECT atributos FROM tabla) LOOP
        l_line := rec.atributo1 || ', ' || rec.atributo2 || ', ' || rec.atributo3; -- Agregar dependiendo del número de atributos
        UTL_FILE.PUT_LINE(l_file, l_line);
  END LOOP;

  -- Cerrar el archivo
  UTL_FILE.FCLOSE(l_file);
  DBMS_OUTPUT.PUT_LINE('Exportación completada');
EXCEPTION
  WHEN OTHERS THEN
        -- En caso de error, cerrar el archivo si está abierto
        IF UTL_FILE.IS_OPEN(l_file) THEN
```

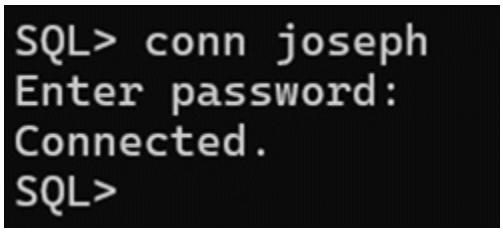
```
    UTL_FILE.FCLOSE(l_file);

        END IF;

        -- Re-lanzar la excepción

        RAISE;

END;
```

Ejemplo de uso:

Aplicar los pre-requisitos



Nos conectaremos al usuario que tiene los permisos



Utilizaremos la tabla estudiantes previamente creada y rellenada con datos para este caso práctico.

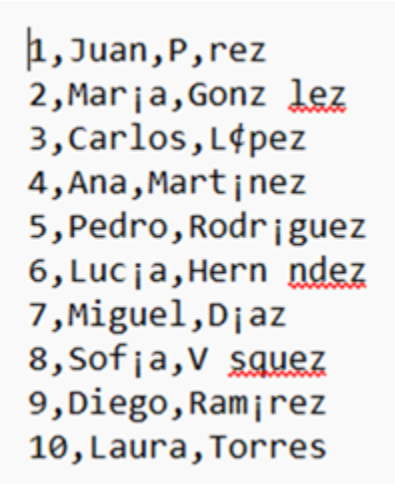
```
SQL> select * from estudiantes;

    EST_ID EST_NOMBRE                     EST_APELLIDO
---------- ------------------------------ ------------------------------
         1 Juan                           Pérez
         2 María                          González
         3 Carlos                         López
         4 Ana                            Martínez
         5 Pedro                          Rodríguez
         6 Lucía                          Hernández
         7 Miguel                         Díaz
         8 Sofía                          Vásquez
         9 Diego                          Ramírez
        10 Laura                          Torres

10 rows selected.
```

Aplicamos el comando para exportar los archivos a un archivo.txt

```
SQL> DECLARE
  2    l_file UTL_FILE.FILE_TYPE;
  3    l_line VARCHAR2(32767);
  4  BEGIN
  5    -- Abrir el archivo para escribir
  6    l_file := UTL_FILE.FOPEN('EXPORT_DIR', 'estudiantes_export.txt', 'w');
  7
  8    -- Seleccionar los datos de la tabla y escribir en el archivo
  9    FOR rec IN (SELECT est_id, est_nombre, est_apellido FROM estudiantes) LOOP
 10      l_line := rec.est_id || ',' || rec.est_nombre || ',' || rec.est_apellido;
 11      UTL_FILE.PUT_LINE(l_file, l_line);
 12    END LOOP;
 13
 14    -- Cerrar el archivo
 15    UTL_FILE.FCLOSE(l_file);
 16
 17    DBMS_OUTPUT.PUT_LINE('Exportación completada');
 18  EXCEPTION
 19    WHEN OTHERS THEN
 20      -- En caso de error, cerrar el archivo si está abierto
 21      IF UTL_FILE.IS_OPEN(l_file) THEN
 22        UTL_FILE.FCLOSE(l_file);
 23      END IF;
 24      -- Re-lanzar la excepción
 25      RAISE;
 26  END;
 27  /

PL/SQL procedure successfully completed.
```

Podremos ver que el archivo se ha creado exitosamente en la carpeta

Revisar los datos



Método#2 Utilizando SPOON

SPOOL DIRECCION\nombrearchivo.txt

SELECT atributos

FROM estudiantes;

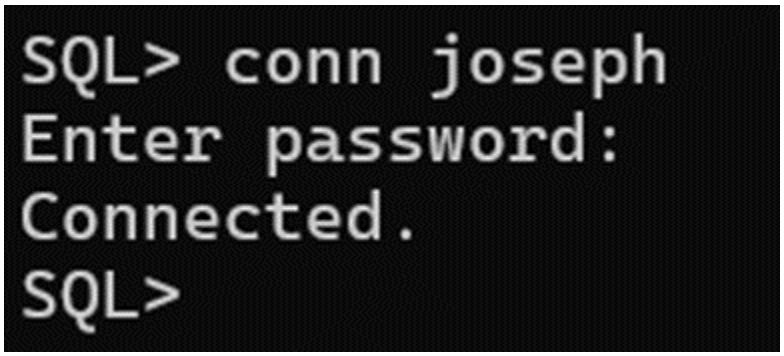
SPOOL OFF

Ejemplo de uso:

Aplicar los pre-requisitos

```
PS C:\Users\devgi> sqlplus sys as sysdba;

SQL*Plus: Release 11.2.0.2.0 Production on Jue Jun 20 17:20:27 2024

Copyright (c) 1982, 2014, Oracle.  All rights reserved.

Enter password:

Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production

SQL> GRANT EXECUTE ON UTL_FILE TO joseph;

Grant succeeded.

SQL> CREATE OR REPLACE DIRECTORY export_dir AS 'C:PruebaOracle';

Directory created.

SQL> GRANT READ, WRITE ON DIRECTORY export_dir TO joseph;

Grant succeeded.
```

Nos conectaremos al usuario que tiene los permisos:



Aplicamos el comando para exportar los archivos a un archivo.txt

SPOOL C:\PruebaOracle\estudiantes_export2.txt

SELECT est_id || ',' || est_nombre || ',' || est_apellido

FROM estudiantes;

SPOOL OFF

```
SQL> SPOOL C:\PruebaOracle\estudiantes_export2.txt
SQL>
SQL> SELECT est_id || ',' || est_nombre || ',' || est_apellido
  2  FROM estudiantes;

EST_ID||','||EST_NOMBRE||','||EST_APELLIDO
--------------------------------------------------------------------------------
1,Juan,Pérez
2,María,González
3,Carlos,López
4,Ana,Martínez
5,Pedro,Rodríguez
6,Lucía,Hernández
7,Miguel,Díaz
8,Sofía,Vásquez
9,Diego,Ramírez
10,Laura,Torres

10 rows selected.

SQL>
SQL> SPOOL OFF
```

Podremos ver que el archivo se ha creado exitosamente en la carpeta

Revisar los datos

```
SQL>
SQL> SELECT est_id || ',' || est_nombre || ',' || est_apellido
  2  FROM estudiantes;

EST_ID||','||EST_NOMBRE||','||EST_APELLIDO
--------------------------------------------------------------------------------
-------
1,Juan,P,rez
2,Mar¡a,Gonz lez
3,Carlos,L¢pez
4,Ana,Mart¡nez
5,Pedro,Rodr¡guez
6,Luc¡a,Hern ndez
7,Miguel,D¡az
8,Sof¡a,V squez
9,Diego,Ram¡rez
10,Laura,Torres

10 rows selected.

SQL>
SQL> SPOOL OFF
```

EQUIPO #8

F. DESARROLLO DE BASE DE DATOS EN LA WEB

El desarrollo de bases de datos en la web es un tema fundamental en la creación de aplicaciones web que requieren almacenar y gestionar grandes cantidades de datos. Las bases de datos web ofrecen varias ventajas, pero también presentan desafíos en cuanto a la conectividad, la latencia y la gestión de copias de seguridad. Las tecnologías y herramientas disponibles permiten diseñar e implementar bases de datos web escalables y seguras que satisfacen las necesidades de las aplicaciones web.

El objetivo principal es desarrollar una comprensión sólida de las bases de datos web y cómo se utilizan en la creación de aplicaciones web. Esto incluye la elección de tecnologías y herramientas adecuadas, el diseño y arquitectura de bases de datos, la implementación de bases de datos, y la integración con aplicaciones web. Al entender estos conceptos, se puede crear una base de datos web eficiente y escalable que satisfaga las necesidades de las aplicaciones web.

a. Conceptos Básicos

Definición de bases de datos web

Es un sistema para almacenar información al que luego se puede acceder a través de un sitio web. Por ejemplo, una comunidad en línea puede tener una base de datos que almacena el nombre de usuario, la contraseña y otros detalles de todos sus miembros. El sistema de base de datos más utilizado para Internet es MySQL debido a su integración con PHP, uno de los lenguajes de programación del lado del servidor más utilizados. En su nivel más simple, una base de datos web es un conjunto de una o más tablas que contienen datos. Cada tabla tiene diferentes campos para almacenar información de varios tipos.

Ventajas y desventajas de las bases de datos en la web

Base de datos en la web	
Ventajas	**Desventajas**
Los usuarios pueden acceder y gestionar datos desde cualquier lugar con conexión a Internet, lo que facilita el trabajo remoto y la colaboración.	El acceso y la gestión de los datos dependen de una conexión a Internet fiable. Cualquier interrupción en la conectividad puede afectar el acceso a los datos.
Permite que las aplicaciones y los usuarios actualicen y consulten datos en tiempo real, mejorando la eficiencia y la productividad.	La latencia puede ser un problema, especialmente en aplicaciones que requieren acceso y procesamiento de datos en tiempo real y que están ubicadas en diferentes regiones geográficas.
Las bases de datos web pueden escalarse fácilmente para manejar un mayor volumen de datos y usuarios mediante técnicas como el clustering y la distribución de datos.	La gestión de copias de seguridad y la recuperación de datos pueden ser complicadas y requerir un plan robusto para evitar la pérdida de datos.

b. **Tecnologías y Herramientas**

Sistemas de gestión de bases de datos (DBMS) utilizados en la web (MySQL, PostgreSQL, MongoDB, etc.).

- **MySQL:** Uno de los sistemas de gestión de bases de datos más populares y ampliamente utilizados. Es conocido por su fiabilidad, facilidad de uso y soporte para grandes volúmenes de datos. Es soporte para SQL, transacciones, replicación y clustering.
- **PostgreSQL:** Es un sistema de gestión de bases de datos relacionales avanzado y de código abierto. Conocido por su robustez, extensibilidad y soporte para estándares SQL. Soporte para JSON, índices avanzados, replicación, y transacciones ACID.
- **MongoDB:** Una base de datos NoSQL orientada a documentos que almacena datos en formato BSON (similar a JSON). Es de Escalabilidad horizontal, consultas flexibles, y soporte para datos geoespaciales.

- **Microsoft SQL Server:** Sistema de gestión de bases de datos relacionales desarrollado por Microsoft, diseñado para entornos empresariales con alta demanda de rendimiento y seguridad. Integración con servicios de Microsoft, herramientas de desarrollo, y capacidades de análisis avanzadas.

- **Cassandra:** Base de datos NoSQL distribuida diseñada para manejar grandes cantidades de datos en múltiples nodos sin un punto único de falla. Contiene una Alta disponibilidad, escalabilidad horizontal, y replicación automática.

c. <u>Lenguajes de programación y frameworks comunes (PHP, Node.js, Python, Ruby on Rails, etc.).</u>

- **React**: Biblioteca de JavaScript para construir interfaces de usuario. Útil en el desarrollo de aplicaciones de una sola página (SPA) y componentes reutilizables.

- **Flask**: Microframework ligero para aplicaciones web. Usado en Proyectos pequeños a medianos, APIs RESTful. Se caracteriza por Flexibilidad, simplicidad, y facilidad para extender con bibliotecas adicionales.

- **Ruby On Rails**: Framework de desarrollo web que sigue el paradigma de "Convención sobre configuración" (CoC). Es de Desarrollo rápido de aplicaciones web y prototipos.

- **Laravel:** Framework de PHP para el desarrollo de aplicaciones web. Se usa en Aplicaciones web con estructuras complejas y necesidades de backend robusto.

- **Django:** Framework de alto nivel para el desarrollo rápido de aplicaciones web. Se enfoca en la simplicidad y reutilización de código, ORM, y administración automática.

d. <u>Arquitecturas de base de datos web</u>

- **Arquitectura Cliente-Servidor:** La arquitectura cliente-servidor es un modelo de diseño que separa las responsabilidades de las aplicaciones en dos roles principales: cliente y servidor.

- **Cliente:** Solicita servicios o recursos del servidor. Puede ser una aplicación web, una aplicación móvil, o cualquier otro tipo de software que necesite comunicarse con un servidor.
- **Servidor:** Provee servicios o recursos al cliente. Puede ser un servidor web, un servidor de base de datos, o cualquier otro tipo de servidor.

e. API y Servicios Web

REST

REST es un estilo de arquitectura que utiliza los estándares HTTP para crear servicios web escalables y mantenibles.

Características clave de REST:

- **Recursos identificables:** Cada recurso se identifica mediante una URI única.
- **Métodos HTTP:** Utiliza métodos estándar como GET, POST, PUT, DELETE para realizar operaciones sobre los recursos.
- **Sin estado:** Cada solicitud del cliente al servidor debe contener toda la información necesaria para entender y procesar la solicitudes.
- **Representaciones múltiples:** Los recursos pueden tener diferentes representaciones, como JSON, XML, etc.

GraphQL

GraphQL es un lenguaje de consulta para APIs desarrollado por Facebook, que permite a los clientes pedir exactamente los datos que necesitan y nada más.

Características clave de GraphQL:

- **Consulta precisa:** Los clientes pueden especificar la forma exacta y la cantidad de datos requeridos.
- **Jerárquico:** Los datos solicitados reflejan la estructura de los objetos anidados.
- **Introspección:** Permite a los clientes descubrir la estructura de la API, incluyendo los tipos y operaciones disponibles.

- **Un solo endpoint:** En lugar de tener múltiples endpoints para diferentes recursos, GraphQL usa un solo endpoint para todas las operaciones.

f. Procedimientos para la Creación de Bases de Datos

1. Planificación y Diseño:

- **Requisitos:**
  - Recopilar y analizar los requisitos específicos de la aplicación web.
  - Considerar necesidades de seguridad y acceso de usuarios.
- **Modelo ER:**
  - Crear un diagrama entidad-relación (ER) para definir entidades, atributos y relaciones.
  - Incluir relaciones de usuario y permisos.
- **Normalización:**
  - Aplicar reglas de normalización para evitar redundancias y mejorar integridad de datos.

2. Configuración del Entorno:

- **Seleccionar DBMS:**
  - Elegir un Sistema de Gestión de Bases de Datos (DBMS) adecuado para la web (e.g., MySQL, PostgreSQL).
- **Instalar DBMS:**
  - Descargar e instalar el DBMS en el servidor adecuado (Linux, Windows).

Población de Datos

Insertar Datos: Utilizar DML (Lenguaje de Manipulación de Datos) para insertar datos iniciales.

Creación de Índices y Vistas

Índices: Mejorar el rendimiento de las consultas creando índices en columnas frecuentemente consultadas.

```
CREATE INDEX idx_nombre ON Usuarios (nombre);
```

Vistas: Crear vistas para simplificar las consultas y mejorar la seguridad.

Configuración del Servidor de Base de Datos

3. Instalación:

- **Instalación en Linux:**

```
sudo apt-get update
sudo apt-get install mysql-server
```

Configuración Inicial:

- **Seguridad:**

```
sudo mysql_secure_installation
```

- **Archivos de Configuración:**

```
[mysqld]
bind-address = 0.0.0.0
max_connections = 200
```

- **Usuarios y Permisos:**

```
CREATE USER 'usuario'@'localhost' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON MiBaseDeDatos.* TO 'usuario'@'localhost';
FLUSH PRIVILEGES;
```

Backup y Recuperación:

- **Estrategias de respaldo:**

```
mysqldump -u usuario -p MiBaseDeDatos > backup_completo.sql
```

- · **Incremental y Diferencial:**

```
mysqlbackup --incremental --incremental-base=dir:/path/to/last/backup
mysqlbackup --differential --incremental-base=dir:/path/to/full/backup
```

Integración con Aplicaciones Web

4. Configuración del Entorno de Desarrollo:

- **Instalar herramientas y frameworks:**
  - Node.js, Django, Spring, etc.
- **Conexión a la Base de Datos:**
  - Usar bibliotecas o drivers específicos.
  - Ejemplo con Node.js:

```
const mysql = require('mysql');
const connection = mysql.createConnection({
    host: 'localhost',
    user: 'usuario',
    password: 'password',
    database: 'MiBaseDeDatos'
});

connection.connect((err) => {
    if (err) throw err;
    console.log('Conectado a la base de datos.');
});
```

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'MiBaseDeDatos',
        'USER': 'usuario',
        'PASSWORD': 'password',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}
```

Operaciones CRUD:

- **Implementar operaciones CRUD:**

```
from django.shortcuts import render
from .models import Usuario

def listar_usuarios(request):
    usuarios = Usuario.objects.all()
    return render(request, 'listar_usuarios.html', {'usuarios': usuarios})
```

g. **<u>Monitoreo del Rendimiento</u>**

Herramientas de Monitoreo:

- **MySQL Performance Schema:** Herramienta integrada para monitorear el rendimiento y diagnosticar problemas.
- **Herramientas Externas:** Utilizar herramientas como Percona Monitoring and Management (PMM), New Relic, o Nagios para monitoreo avanzado.
- **Monitoreo de Métricas Clave:**

· **Uso de CPU y Memoria:** Vigilar el uso de recursos del servidor.

- **Latencia de Consultas:** Medir el tiempo que tardan las consultas en ejecutarse.
- **Tasa de Aciertos en Caché:** Verificar cuántas consultas se resuelven desde la caché en lugar de leer del disco.
- **Conexiones Activas:** Monitorear el número de conexiones activas a la base de datos.

Alertas y Notificaciones: Configurar alertas para notificar sobre condiciones anómalas o problemas de rendimiento.

Análisis de Rendimiento Regular: Revisar regularmente los informes de rendimiento y ajustar la configuración y consultas según sea necesario.

h. Control de Acceso en Bases de Datos Web

Conceptos Básicos

Definición y objetivos del control de acceso

El control de acceso en bases de datos web se refiere a los mecanismos y políticas que restringen el acceso a la base de datos para garantizar que solo los usuarios autorizados puedan acceder, modificar o eliminar datos. Esto incluye tanto la verificación de identidad (autenticación) como la determinación de qué recursos puede acceder un usuario (autorización).

Objetivos:

1. **Protección de Datos Sensibles:** Asegurar que solo los usuarios autorizados tengan acceso a información crítica o confidencial.
2. **Integridad de los Datos:** Prevenir modificaciones no autorizadas que podrían comprometer la exactitud y consistencia de la información almacenada.
3. **Disponibilidad:** Garantizar que los usuarios legítimos puedan acceder a la base de datos cuando lo necesiten.
4. **Auditoría y Responsabilidad:** Registrar y monitorear las acciones de los usuarios para detectar actividades sospechosas y asignar responsabilidades.

Importancia del control de acceso en las bases de datos web

Las bases de datos web son particularmente vulnerables a ataques debido a su accesibilidad global. La importancia del control de acceso en este contexto incluye:

1. **Protección contra Accesos No Autorizados:** Evita que individuos no autorizados accedan a la base de datos y potencialmente roben o alteren información.
2. **Prevención de Ataques Cibernéticos:** Reduce la probabilidad de ataques como SQL injection, phishing y ataques de fuerza bruta al implementar medidas de control de acceso robustas.
3. **Cumplimiento Normativo:** Asegura el cumplimiento de regulaciones y estándares de seguridad de la información, como GDPR, HIPAA y PCI DSS, que requieren controles estrictos sobre el acceso a datos sensibles.
4. **Confianza del Usuario:** Mantiene la confianza de los usuarios y clientes al garantizar que sus datos están protegidos contra accesos no autorizados.

Métodos de Autenticación

- **Autenticación basada en contraseñas**

La autenticación basada en contraseñas es el método más común, donde el usuario ingresa un nombre de usuario y una contraseña para acceder a un sistema. Si bien es fácil de implementar y usar, presenta vulnerabilidades como ataques de fuerza bruta, phishing y contraseñas débiles. Para mitigar estos riesgos, se deben implementar políticas de contraseñas sólidas que incluyan longitud mínima, complejidad y cambios periódicos.

- **Autenticación de dos factores (2FA)**

La autenticación de dos factores (2FA) agrega una capa adicional de seguridad al requerir dos tipos de autenticación: algo que el usuario sabe (contraseña) y algo que el usuario tiene (token, smartphone) o es (biometría). Esto aumenta significativamente la seguridad, ya que incluso si un factor se compromete, el otro sirve como barrera de protección. Sin embargo, puede ser menos conveniente para los usuarios y requiere infraestructura adicional.

Uso de certificados y claves públicas:

El uso de certificados y claves públicas emplea criptografía de clave pública para autenticar usuarios mediante certificados digitales emitidos por una autoridad de certificación (CA). Este método ofrece la máxima seguridad y resistencia a la falsificación, eliminando la necesidad de

recordar contraseñas y protegiéndolas con criptografía. No obstante, su implementación y gestión son complejas, especialmente en grandes organizaciones, y requiere un manejo cuidadoso de certificados y claves privadas.

En resumen:

- La autenticación basada en contraseñas es simple pero menos segura.
- La 2FA aumenta la seguridad pero puede ser menos práctica.
- Los certificados y claves públicas ofrecen la máxima seguridad pero son complejos de implementar.

I. Métodos de Autorización

Roles y permisos

Asigna permisos específicos a roles predefinidos, y los usuarios se asocian a uno o más roles. Los permisos determinan qué acciones puede realizar un usuario en el sistema. Simplifica la administración de permisos al agruparlos en roles, pero puede ser complejo en sistemas con un gran número de roles y permisos, y requiere una planificación cuidadosa para definirlos de manera adecuada.

· **Control de acceso basado en roles (RBAC)**

Asigna permisos de acceso a roles específicos dentro de la organización, y los usuarios se asignan a estos roles. Cada rol tiene un conjunto de permisos asociados que dictan lo que el usuario puede hacer en el sistema. Facilita la gestión de permisos agrupándolos por roles, lo que es ideal para grandes organizaciones. Es flexible y escalable, adaptándose a las necesidades cambiantes de la organización, pero requiere un análisis detallado para definir roles y permisos de manera efectiva, y la gestión de roles puede volverse complicada en organizaciones muy grandes o complejas.

· **Control de acceso basado en atributos (ABAC)**

Utiliza atributos de usuarios, recursos y el entorno para tomar decisiones de acceso. Los atributos pueden incluir el rol del usuario, la hora del día, la ubicación, entre otros. Ofrece una granularidad y flexibilidad mucho mayores en las políticas de acceso, permitiendo definir políticas dinámicas basadas en múltiples factores contextuales. Es complejo de implementar y mantener por la

necesidad de gestionar múltiples atributos, y requiere una infraestructura robusta para manejar y evaluar los atributos en tiempo real.

Implementación en Bases de Datos Web

La implementación de bases de datos en aplicaciones web implica la configuración y administración de un sistema de gestión de bases de datos (DBMS) que permita almacenar, recuperar y manipular datos a través de aplicaciones web.

Configuración de Usuarios y Permisos en DBMS

La configuración de usuarios y permisos es esencial para asegurar la base de datos. Los pasos básicos incluyen:

- **Creación de Usuarios:**
  - Generación de cuentas de usuario específicas para diferentes roles, como administradores, desarrolladores y usuarios finales.
- **Asignación de Roles y Permisos:**
  - Definición de roles y asignación de permisos específicos (SELECT, INSERT, UPDATE, DELETE) a cada rol.
- **Grupos de Seguridad:**
  - Uso de grupos para facilitar la gestión de permisos cuando múltiples usuarios comparten los mismos permisos.
- **Revisión y Auditoría:**
  - Monitoreo regular de permisos y actividades para detectar y corregir cualquier configuración inapropiada.

Integración con Sistemas de Autenticación Externa (OAuth, LDAP)

Integrar un DBMS con sistemas de autenticación externa mejora la seguridad y simplifica la gestión de identidades. Dos enfoques comunes son:

- **OAuth (Open Authorization):**
  - Permite a los usuarios autenticarse utilizando sus credenciales de servicios externos (como Google o Facebook).

    - Implementación a través de la configuración de endpoints y la validación de tokens de acceso en el backend.
- **LDAP (Lightweight Directory Access Protocol):**
    - Autentica usuarios contra un directorio de usuarios centralizado (como Active Directory).
    - Configuración del DBMS para consultar y autenticar usuarios mediante el servidor LDAP.

CONCLUSIÓN

El desarrollo de bases de datos web es crucial para aplicaciones que gestionan grandes volúmenes de datos. Ofrecen ventajas como accesibilidad desde cualquier lugar y escalabilidad, pero presentan desafíos en conectividad, latencia y gestión de copias de seguridad. Las tecnologías disponibles, como MySQL, PostgreSQL y MongoDB, permiten diseñar e implementar bases de datos seguras y eficientes.

Además, el desarrollo de bases de datos en la web y el control de acceso en entornos en línea ha destacado la importancia de adaptar nuestras estrategias a las necesidades modernas de seguridad.

Es esencial entender la arquitectura, seleccionar herramientas adecuadas y seguir buenas prácticas en diseño, implementación y seguridad. Así, se puede crear una base de datos web que satisfaga las necesidades específicas de las aplicaciones web, asegurando rendimiento, integridad y protección de los datos.

La culminación de esta investigación ha proporcionado al Grupo 133 de Ingeniería de Software una comprensión profunda y multifacética del aseguramiento del acceso a bases de datos. A través de un enfoque colaborativo y especializado, cada equipo ha contribuido con valiosos hallazgos y soluciones prácticas a problemas específicos de administración y seguridad en bases de datos.

REFERENCIAS BIBLIOGRÁFICAS

Jeloka, S. (2024, June 14). *Managing User Authentication and Authorization*. Oracle Help Center. https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/part_1.html

June2024. (2024, June 14). *Configuring privilege and role authorization*. Oracle Help Center. https://docs.oracle.com/en/database/oracle/oracle-database/23/dbseg/configuring-privilege-and-role-authorization.html#GUID-AEAC1E85-D55E-41B4-83B4-82740CB5A82C

TimesTen In-Memory Database SQL Reference. (n.d.). https://docs.oracle.com/database/timesten-18.1/TTSQL/privileges.htm#TTSQL339

ArcGIS Pro. (n.d.). Privilegios para geodatabases en Oracle—ArcGIS Pro | Documentación. Recuperado de https://pro.arcgis.com/es/pro-app/latest/help/data/geodatabases/manage-oracle/privileges-oracle.htm

ArcGIS Online. (n.d.). Privilegios para geodatabases en Oracle—ArcMap - ArcGIS Online. Recuperado de https://desktop.arcgis.com/es/arcmap/latest/manage-data/gdbs-in-oracle/privileges-oracle.htm

Amazon RDS. (n.d.). Usuarios y privilegios de RDS para Oracle - Amazon. Recuperado de https://docs.aws.amazon.com/es_es/AmazonRDS/latest/UserGuide/Oracle.Concepts.Privileges.html

Oracle. (n.d.). Privilegios de usuario de base de datos necesarios para Database Management. Recuperado de https://docs.oracle.com/es-ww/iaas/database-management/doc/database-user-privileges-required-database-management.html

Oracle. (n.d.). Acerca de los Privilegios y los Permisos. Recuperado de https://docs.oracle.com/middleware/12211help/biee/es/bi.12211/e73374/GUID-29A5CBE3-8C06-4879-A2FD-1A313ABEEDB4.htm

DBAgroup. (2009, junio 2). Usuarios y privilegios en Oracle - DBAgroup. Recuperado de https://dbagroup.cl/usuarios-y-privilegios-en-oracle/

ArcGIS Pro. (n.d.). Privilegios para geodatabases en Oracle—ArcGIS Pro | Documentación. Recuperado de https://pro.arcgis.com/es/pro-app/latest/help/data/geodatabases/manage-oracle/privileges-oracle.htm

ArcGIS Online. (n.d.). Privilegios para geodatabases en Oracle—ArcMap - ArcGIS Online. Recuperado de https://desktop.arcgis.com/es/arcmap/latest/manage-data/gdbs-in-oracle/privileges-oracle.htm

Amazon RDS. (n.d.). Usuarios y privilegios de RDS para Oracle - Amazon. Recuperado de https://docs.aws.amazon.com/es_es/AmazonRDS/latest/UserGuide/Oracle.Concepts.Privileges.html

Oracle. (n.d.). Privilegios de usuario de base de datos necesarios para Database Management. Recuperado de https://docs.oracle.com/es-ww/iaas/database-management/doc/database-user-privileges-required-database-management.html

Oracle. (n.d.). Acerca de los Privilegios y los Permisos. Recuperado de https://docs.oracle.com/middleware/12211help/biee/es/bi.12211/e73374/GUID-29A5CBE3-8C06-4879-A2FD-1A313ABEEDB4.htm

DBAgroup. (2009, junio 2). Usuarios y privilegios en Oracle - DBAgroup. Recuperado de https://dbagroup.cl/usuarios-y-privilegios-en-oracle/

6.1. CONFIGURACIÓN — Gestión de Bases de Datos. (n.d.). https://gestionbasesdatos.readthedocs.io/es/latest/Tema6/Teoria.html

Aitor. (2015, October 1). *Gestión de privilegios en Oracle*. Blog De Aitor. https://blogdeaitor.wordpress.com/2008/10/30/comandos-oracle-%E2%80%93-tercera-parte-%E2%80%93/

IBM DB2 Warehouse on Cloud. (n.d.). https://www.ibm.com/docs/es/db2woc?topic=statements-grant-table-view-nickname-privileges

osceda@hotmail.com. (2024, 18 enero). *Oracle ALTER PROFILE: Prácticos ejemplos – Alter profile in Oracle*. Oscar Aguado Web. https://oscaraguadoweb.com/oracle/oracle-alter-profile-practicos-ejemplos-alter-profile-in-oracle/

Cómo activar los límites de recursos - Administración de la gestión de recursos en Oracle® Solaris 11.2. (2015, 20 enero). https://docs.oracle.com/cd/E56339_01/html/E54030/rm.rcapd.task-10.html

Cambio y eliminación de cuotas de UFS - Guía de administración del sistema: administración avanzada. (2011, 1 enero). https://docs.oracle.com/cd/E24842_01/html/E23086/sysresquotas-10332.html

Archiveddocs. (2014, 24 septiembre). *Límites de uso de recursos en las soluciones de espacio aislado*. Microsoft Learn. https://learn.microsoft.com/es-es/previous-versions/office/developer/sharepoint-2010/gg615462(v=office.14)

colaboradores de Wikipedia. (2020, 26 enero). *Espacio de tabla (base de datos)*. Wikipedia, la Enciclopedia Libre. https://es.wikipedia.org/wiki/Espacio_de_tabla_(base_de_datos)

August2023. (2023, 17 agosto). Managing Users and Securing the Database. Oracle Help Center. https://docs.oracle.com/en/database/oracle/oracle-database/19/admin/managing-users-and-securing-the-database.html

August2023. (2023b, agosto 17). Managing Users and Securing the Database. Oracle Help Center. https://docs.oracle.com/en/database/oracle/oracle-database/19/admin/managing-users-and-securing-the-database.html#GUID-A294CD5E-BC73-4FA7-980A-8A2538950F2D

Oracle ALTER PROFILE Statement by Practical Examples. (2023, 5 octubre). Oracle Tutorial. https://www.oracletutorial.com/oracle-administration/oracle-alter-profile/

Oracle CREATE PROFILE: Setting database resource & password Limits. (2023, 5 octubre). Oracle Tutorial. https://www.oracletutorial.com/oracle-administration/oracle-create-profile/

March2024. (2024, 4 marzo). CREATE PROFILE. Oracle Help Center. https://docs.oracle.com/en/database/oracle/oracle-database/19/sqlrf/CREATE-PROFILE.html

Oracle® Fusion Cloud EPM Trabajo con Planning. (s. f.). https://docs.oracle.com/cloud/help/es/pbcs_common/PFUSU/working_with_mr_reports.htm#PFUSU-Document1_6

Servicios de Oracle Reports | Oracle México. (s. f.). https://www.oracle.com/mx/middleware/technologies/reports.html

Administración de perfiles - Configuración y administración de componentes de red en Oracle® Solaris 11.2. (2015, 20 enero). https://docs.oracle.com/cd/E56339_01/html/E53785/gmyaf.html

Gopalakrishnan, K., & Alapati, S. R. (2018). Oracle Database 12C Release 2 Real Application Clusters Handbook: Concepts, Administration, Tuning & Troubleshooting. McGraw Hill Professional.

Bryla, B. (2015). Oracle Database 12C DBA Handbook. McGraw Hill Professional.

Feuerstein, S., & Pribyl, B. (2009). Oracle PL/SQL Programming (5th ed.). O'Reilly Media.

Loney, K. (2016). Oracle Database 12c The Complete Reference (1st ed.). McGraw-Hill Education.

Urman, S., Hardman, R., & McLaughlin, M. (2004). Oracle Database 10g PL/SQL Programming. McGraw-Hill Education.

Price, M., & Viscusi, E. (2013). Oracle Database 12c Security Cookbook. Packt Publishing.

Kyte, T., & Kuhn, D. (2013). Expert Oracle Database Architecture (3rd ed.). Apress.

Vives, J. (n.d.). Bases de Datos (B) (Material Didáctico). Scribd. https://es.scribd.com/document/493646287/Bases-de-Datos-B-Material-Didactico

ORACLE: Seguridad. (n.d.). https://www.infor.uva.es/~jvegas/cursos/bd/oraseg/oraseg.html

Oracle Corporation. (2023). *Oracle Database Backup and Recovery User's Guide*. Oracle Corporation. https://docs.oracle.com/en/database/oracle/oracle-database/21/bradv/database-backup-and-recovery-users-guide.pdf

Microsoft Corporation. (2023). *SQL Server Backup and Restore*. Microsoft Corporation. https://learn.microsoft.com/en-us/sql/relational-databases/backup-restore/restore-a-database-backup-using-ssms?view=sql-server-ver16

Vogel, J. (2023). *Database Recovery: Strategies and Techniques for Protecting and Restoring Your Data*. Apress. [se quitó una URL no válida]

Armbrust, M., Curtis, T., Fordham, B., Kozyrakov, A., Li, K., Patterson, D., ... & Robin, S. (2020). *DataStax Cassandra: The Definitive Guide*. O'Reilly Media.
Aldaz, L., Eguía, B., & Urcola, L. (2009). Importación y Exportación de Datos. Recuperado de https://ocw.ehu.eus/file.php/129/herrami_gestion/m1t6eg.pdf

Google Analytics. (s.f.). Información sobre Importación de datos. Recuperado de https://support.google.com/analytics/answer/3191589?hl=es

Gómez Orea, D. (1999). Evaluación del impacto ambiental de proyectos de desarrollo. Recuperado de https://biblioteca.semarnat.gob.mx/janium/Documentos/Ciga/Libros2011/CD001413.pdf

Organización de los Estados Americanos. (2007). Evaluación Ambiental y de Capacidad Institucional de Panamá Frente al Tratado de Promoción Comercial con los Estados Unidos. Recuperado de

http://www.oas.org/dsd/environmentlaw/documents/estudio_panama_oea__informe_final__juliorevisado.pdf
Autoridad Nacional del Ambiente. (2017). Estudio de Impacto Ambiental Categoría II. Recuperado de https://ana.gob.pa/w_ana/images/ANA_pdf/noticias_/estudio_impacto_ambiental.pdf

Garrido, G. (n.d.). *Programación en PL/SQL: Teoría y ejemplos* [Presentación de diapositivas]. SlideShare. https://www.slideshare.net/slideshow/programacion-en-plsql-teoria-y-ejemplos/266105721#12

Rodríguez, C. (2008, diciembre 11). *Exportar fácilmente datos de Oracle a un fichero plano*. Dataprix. https://www.dataprix.com/es/blog-it/carlos/exportar-facilmente-datos-oracle-un-fichero-plano

Oracle. (s.f.). **Exportación con Data Pump**. Documentación de Oracle. Recuperado el 20 de junio de 2024, de https://docs.oracle.com/en/database/oracle/oracle-database/19/dmpug/data-pump-export.html

Oracle. (s.f.). **Utilidades de la base de datos: Utilidad de Exportación (exp)**. Documentación de Oracle. Recuperado el 20 de junio de 2024, de https://docs.oracle.com/cd/B28359_01/server.111/b28319/exp_imp.htm

IBM. (s.f.). **Comprendiendo la Exportación e Importación de Datos**. Centro de Conocimiento de IBM. Recuperado el 20 de junio de 2024, de https://www.ibm.com/support/knowledgecenter/SSLTBW_2.3.0/com.ibm.zvm.v23.hcpa0/hcpacu0088.htm

TechTarget. (2021). **¿Qué es la Exportación de Datos?**. Recuperado el 20 de junio de 2024, de https://www.techtarget.com/searchdatamanagement/definition/data-export

Oracle. (s.f.). **Oracle SQL Developer: Exportación de Datos**. Documentación de Oracle. Recuperado el 20 de junio de 2024, de https://docs.oracle.com/en/database/oracle/sql-developer/19.1/rptug/exporting-data.html

GeeksforGeeks. (2021). **Introducción a la Exportación e Importación de Bases de Datos Oracle**. Recuperado el 20 de junio de 2024, de https://www.geeksforgeeks.org/introduction-to-oracle-database-export-and-import/

Datadog. (2023). **Automatizando las Copias de Seguridad de la Base de Datos Oracle con Data Pump**. Recuperado el 20 de junio de 2024, de https://www.datadoghq.com/blog/oracle-database-backups/