CENTRE ESPECÍFIC
D´EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

TEMA7
ADMINISTRACIÓN DE USUARIOS ORACLE

Bases de Datos
CFGS DAW

Raquel Torres
raquel.torres@ceedcv.es
Versión:190225.1229

Licencia

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

Revisiones

- Pg. 5: Se modifica el TABLESPACE por defecto de USERS a SYSTEM.
- Pg. 7: En la línea que comienza con “PROFILE profile_name...” falta realizar un enter y crear un nuevo punto empezando con “PASSWORD EXPIRE...”

ÍNDICE DE CONTENIDO

1. Tareas esenciales de un BDA....4
2. Administración de Usuarios....5
2.1 Concepto de esquema, usuario, privilegio y rol....5
2.2 Creación de usuarios....7
2.3 Eliminación de usuarios....8
3. Privilegios....9
3.1 Privilegios de sistema....9
3.1.1 Instrucción GRANT....9
3.1.2 Instrucción REVOKE....11
3.2 Privilegios sobre objetos....12
3.2.1 Instrucción GRANT....12
3.2.2 Instrucción REVOKE....14
4. Creación de roles....15
5. Perfiles de Usuario....16
6. Listar Privilegios Otorgados. Vistas del Diccionario....18
7. Casos prácticos usuarios, privilegios y roles....20
7.1 Caso práctico 1....20
7.2 Caso práctico 2....21

UD7 ADMINISTRACIÓN DE USUARIOS ORACLE

1. TAREAS ESENCIALES DE UN BDA

Algunas de las tareas esenciales que debe realizar un administrador de bases de datos son las que aparecen a continuación, no obstante esto dependerá del tipo y del tamaño del sistema a administrar.

- Decidir el SGBD idóneo, instalarlo y configurarlo inicialmente.
- Supervisar el diseño lógico de la BD.
- Realizar el diseño físico de la BD: Estructura de almacenamiento.
- Crear y mantener el esquema de la BD.
- Colaborar en la formación de usuarios y programadores.
- Detectar y resolver problemas de rendimiento de la BD usando herramientas de monitorización.
- Crear y mantener cuentas de usuario.
- Mantener la disponibilidad de la base de datos. Y en caso de fallos será su tarea garantizar la mínima pérdida de información. Para ello, el DBA (Data Base Administrator) deberá:
  - Realizar **copias de seguridad** de la base de datos para asegurarse de poder poner en marcha la base de datos ante cualquier fallo que se produzca.
  - **Minimizar** el **tiempo** en que la **base de datos no** está **disponible**. En definitiva debe reducir lo máximo posible el tiempo de recuperación.
  - **Proteger** la **base de datos contra** varios tipos de **fallos**. Configurar la base de datos de forma que no se produzcan estos fallos a menudo y así aumentar el tiempo medio entre fallos.

2. ADMINISTRACIÓN DE USUARIOS

2.1 Concepto de esquema, usuario, privilegio y rol

Se denomina **esquema** al conjunto de objetos que son propiedad de un usuario concreto.

Dichos objetos pueden ser tablas, vistas, índices, procedimientos, funciones, triggers, etc.

El esquema de cada usuario se almacena físicamente en un TABLESPACE. Si no se especifica nada en la sintaxis de creación del usuario, se le asignará el TABLESPACE por defecto, que es SYSTEM. Aunque en un mismo TABLESPACE se pueden almacenar los esquemas de distintos usuarios, como ocurre en SYSTEM, es recomendable crear un TABLESPACE para cada uno de los usuarios.

Recordemos la sintaxis de creación de un TABLESPACE:

```
CREATE TABLESPACE nombre_tablespace
DATAFILE 'C:\oraclexe\app\oracle\oradata\XE\nombre_tablespace.dbf'
SIZE 50M
AUTOEXTEND ON NEXT 50M MAXSIZE 300M
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 10M
SEGMENT SPACE MANAGEMENT AUTO;
```

Veamos de qué se compone el comando anterior:

- Crea un espacio de nombres llamado *nombre_tablespace*.
- El fichero (datafile) que almacenará los objetos que se creen dentro de dicho espacio se llamará *nombre_tablespace.dbf* y se encontrará en la ruta donde están los espacios para objetos que Oracle tiene por defecto.
- El tamaño inicial del fichero será de 50MB, pudiendo aumentar otros 50MB si se llena. Este proceso se puede repetir hasta llegar a un tamaño máximo de 300MB.
- La gestión de las extensiones de este espacio de objetos se hará localmente.
- La gestión de segmentos de este espacio de objetos se hará automáticamente.

Nota: Si en algún momento nos quedamos sin espacio dentro del fichero creado, Oracle nos dará error al crear nuevos objetos o agregar más información. La solución reside en redimensionar el fichero, no el TABLESPACE (de nada sirve hacer más grande el TABLESPACE si el fichero tiene un límite inferior).

```
ALTER DATABASE DATAFILE AUTOEXTEND ON NEXT 5M MAXSIZE 1G;
```

Es un planteamiento totalmente distinto al de otros SGBD (como MySQL), donde un esquema es, a grandes rasgos, una base de datos tal y como se entiende en un SGBD de escritorio.

A la hora de establecer una conexión con un servidor Oracle lo hacemos utilizando un nombre de usuario. Cada uno de los usuarios que acceden a la base de datos lo hace con un nivel diferente de seguridad, podrá acceder a unos u otros objetos y podrá realizar unas acciones u otras sobre ellos.

Un **usuario** no es más que un conjunto de permisos que se aplican a una conexión de base de datos.

Así mismo, el usuario también tiene otras funciones:

- Ser el propietario de ciertos objetos y de su esquema.
- Definición del *tablespace* por defecto para los objetos de un usuario.
- Copias de seguridad.
- Cuotas de almacenamiento.

Un **privilegio** no es más que un permiso dado a un usuario para que realice cierta operación.

Estas operaciones pueden ser de dos tipos:

- Privilegio de sistema: necesita el permiso de sistema correspondiente. Permiten realizar una determinada operación o ejecutar un comando concreto, por ejemplo create, select, alter, etc.
- Privilegio sobre objeto: necesita el permiso sobre el objeto en cuestión. Cuando a un usuario se le da un privilegio sobre un objeto, se le permite hacer algo con ese objeto.

Y por último,

Un **rol** de base de datos no es más que una agrupación de permisos tanto de sistema como de objeto.

Un **perfil** es un conjunto de límites de uso de los recursos del sistema y/o un conjunto de características que debe cumplir una contraseña.

2.2 Creación de usuarios

La creación de usuarios se hace a través de la sentencia *SQL CREATE USER*. Su sintaxis básica es:

```
CREATE USER username
IDENTIFIED {BY password | EXTERNALLY }
DEFAULT TABLESPACE tablespace
TEMPORARY TABLESPACE tablespace
QUOTA int {K | M} ON tablespace
QUOTA UNLIMITED ON tablespace
PROFILE profile_name
PASSWORD EXPIRE
ACCOUNT {LOCK|UNLOCK}
```

La cláusula *IDENTIFIED BY* permite indicar el tipo de autorización que se utilizará:

- Interna de Oracle: una clave para cada usuario de base de datos.
- Interna del SO: utilizando la seguridad del SO.

La cláusula *DEFAULT TABLESPACE* será el tablespace por defecto en la creación de objetos del usuario que estamos creando. Si se omite se utilizará el tablespace *SYSTEM*, como hemos dicho, no esrecomendable utilizar el tablespace SYSTEM para la creación de objetos de usuarios.

La cláusula *TEMPORARY TABLESPACE* indica el tablespace que se utilizará para la creación de objetos temporales en la operaciones internas de Oracle. Si se omite se utilizará el tablespace *SYSTEM*. Un tablespace temporal contiene objetos del esquema durante el tiempo que dura la sesión del usuario de dicho esquema. Los objetos que se encuentran en un tablespace temporal son almacenados en tempfiles (no en datafiles).

Para crear un TABLESPACE temporal llamado TEMP utilizaremos la sintaxis:

```
CREATE TEMPORARY TABLESPACE TEMP TEMPFILE 'ruta_temp.dbf' SIZE 2000M;
```

QUOTA int {K | M} ON tablespace permite asignar una cantidad máxima de memoria usada por el usuario dentro de un tablespace, no tiene porque ser su tablespace por defecto.

QUOTA UNLIMITED ON tablespace asigna uso ilimitado de un tablespace, no tiene porque ser su tablespace por defecto.

PROFILE profile_name proporciona un perfil al usuario. Si queremos asignarle uno diferente del que tienen todos los usuarios que es *DEFAULT.*

PASSWORD EXPIRE obliga al usuario a cambiar la contraseña en la primera conexión.

ACCOUNT {LOCK|UNLOCK} cuenta bloqueada o desbloqueada respectivamente.

Ejemplo

```
CREATE USER administrador
IDENTIFIED BY manager
DEFAULT TABLESPACE system
TEMPORARY TABLESPACE temporary_data
PASSWORD EXPIRE;

CREATE USER pepote
IDENTIFIED BY toro;
CREATE USER juancito
IDENTIFIED BY perez
DEFAULT TABLESPACE datos_contabilidad
TEMPORARY TABLESPACE temporary_data;
```

2.3 Eliminación de usuarios

La eliminación de usuarios se hace a través de la instrucción *DROP USER*.

Su sintaxis es:

```
DROP USER usuario {CASCADE};
```

Cláusula *CASCADE* permite borrar el usuario y todos los objetos que posea.

3. PRIVILEGIOS

3.1 Privilegios de sistema

Ya hemos dicho que los privilegios de sistema son permisos para realizar ciertas operaciones en la base de datos.

El modo de asignar un privilegio es a través de la instrucción *GRANT* y el modo de cancelar un privilegio es a través de la instrucción *REVOKE*.

La sintaxis básica de ambas instrucciones es:

3.1.1 Instrucción GRANT

```
GRANT [privilegios_de_sistema | roles]
TO [usuarios | roles |PUBLIC]
{ WITH ADMIN OPTION };
```

Es posible dar más de un privilegio de sistema o rol, separándolos por comas.

También es posible asignarle uno (o varios) privilegios a varios usuarios, separándolos por comas.

> Si se le asigna el privilegio a un rol, se asignará a todos los usuarios que tengan ese rol.

> Si se asigna el privilegio a *PUBLIC*, se asignará a todos los usuarios actuales y futuros de la base de datos.

La cláusula *WITH ADMIN OPTION* permite que el privilegio/rol que hemos concedido, pueda ser concedido a otros usuarios por el usuario al que estamos asignando.

La palabra clave *ANY* significa que ese usuario tiene el privilegio para todos los esquemas en la BD.

Algunos privilegios de sistema que tiene Oracle[1]:

Privilegio de Sistema	Capacidades
CREATE PROFILE	Crear perfiles de usuario
CREATE ROLE	Crear roles
CREATE ROLLBACK SEGMENT	Creación de segmentos de rollback
CREATE TABLESPACE	Crear espacios de tablas
CREATE USER	Crear usuarios
ALTER PROFILE	Alterar perfiles existentes
ALTER ANY ROLE	Alterar cualquier rol
ALTER ROLLBACK SEGMENT	Alterar segmentos de rollback
ALTER TABLESPACE	Alterar espacios de tablas
ALTER USER	Alterar usuarios
DROP PROFILE	Borrar un perfil existente
DROP ANY ROLE	Borrar cualquier rol
DROP ROLLBACK SEGMENT	Borrar un segmento de rollback existente
DROP TABLESPACE	Borrar un espacio de tablas
DROP USER	Borrar un usuario. Añadir CASCADE si el usuario posee objetos.
ALTER DATABASE	Permite una sentencia ALTER DATABASE
GRANT ANY PRIVILEGE	Otorgar cualquiera de estos privilegios
GRANT ANY ROLE	Otorgar cualquier rol a un usuario
UNLIMITED TABLESPACE	Puede usar una cantidad de almacenamiento ilimitada.

1 Ver todos los privilegios de sistema

Ejemplos

```
GRANT DBA TO administrador;
GRANT CREATE USER TO pepote WITH ADMIN OPTION;
GRANT DROP USER TO juancito;
GRANT CONNECT, RESOURCE TO pepote, juancito;
GRANT CONNECT, RESOURCE, DBA, EXP_FULL_DATABASE, IMP_FULL_DATABASE   TO control_total;
GRANT CONTROL_TOTAL TO administrador;
```

3.1.2 Instrucción REVOKE

```
REVOKE [privilegios_de_sistema | roles]
FROM [usuarios | roles |PUBLIC];
```

Es posible eliminar más de un privilegio de sistema o rol, separándolos por comas.

También es posible eliminar uno (o varios) privilegios a varios usuarios, separándolos por comas.

> Si se le elimina el privilegio de un rol, se eliminará de todos los usuarios que tengan ese rol.

> Si se elimina el privilegio de *PUBLIC*, se eliminará de todos los usuarios actuales y futuros de la base de datos.

Como es lógico, sólo se podrá eliminar un privilegio/rol, si previamente ha sido concedido a través de la instrucción *GRANT*.

Ejemplos

```
REVOKE DBA FROM administrador;
REVOKE CREATE USER FROM pepote;
REVOKE DROP USER FROM juancito;
RECOKE CONNECT, RESOURCE FROM pepote, juancito;
REVOKE CONNECT, RESOURCE, DBA, EXP_FULL_DATABASE, IMP_FULL_DATABASE FROM control_total;
REVOKE control_total FROM administrador;
```

3.2 Privilegios sobre objetos

Los privilegios sobre objetos permiten que cierto objeto (creado por un usuario) pueda ser accedido por otros usuarios. El nivel de acceso depende del permiso que le demos: podemos darle permiso de *SELECT*, de *UPDATE*, de *DELETE*, de *INSERT* o de todos ellos.

3.2.1 Instrucción GRANT

La sintaxis básica es:

```
GRANT [ALL {PRIVILEGES} | SELECT | INSERT | UPDATE | DELETE]
ON propietario.objeto
TO [usuario | rol | PUBLIC]
{WITH GRANT OPTION};
```

Al igual que con los permisos de sistema, es posible asignar un permiso de objeto sobre uno o varios (separados por comas) usuario y/o roles. Si se asigna a *PUBLIC* será accesible en toda la base de datos.

Si se incluye la cláusula *WITH GRANT OPTION*, este permiso podrá ser concedido por el usuario al que se le ha asignado.

Algunos privilegios de objeto que tiene Oracle[2]:

Manejo de Objetos	Capacidades
SELECT	Puede consultar a un objeto
INSERT	Puede insertar filas en una tabla o vista. Puede especificarse las columnas donde se permite insertar dentro de la tabla o vista
UPDATE	Puede actualizar filas en una tabla o vista. Puede especificarse las columnas donde se permite actualizar dentro de la tabla o vista
DELETE	Puede borrar filas dentro de la tabla o vista
ALTER	Puede alterar la tabla
CREATE INDEX	Puede crear índices de una tabla
REFERENCES	Puede crear claves ajenas que referencien a esta tabla

2 Ver todos los privilegios sobre objetos

EXECUTE	Puede ejecutar un procedimiento, paquete o función
CREATE ANY INDEX	Crear cualquier índice
CREATE [PUBLIC] SYNONYM	Crear sinónimos [públicos]
CREATE [ANY] TABLE	Crear tablas. El usuario debe tener cuota en el espacio de tablas, o ha de tener asignado el privilegio *UNLIMITED TABLESPACE*
CREATE [ANY] VIEW	Crear vistas
ALTER ANY INDEX	Alterar cualquier índice
ALTER ANY TABLE	Alterar cualquier tabla
DROP ANY INDEX	Borrar cualquier índice
DROP ANY SYNONYM	Borrar cualquier sinónimo
DROP PUBLIC SYNONYM	Borrar sinónimos públicos
DROP ANY VIEW	Borrar cualquier vista
DROP ANY TABLE	Borrar cualquier tabla
SELECT ANY TABLE	Efectuar selecciones de cualquier tabla o vista
INSERT ANY TABLE	Insertar en cualquier tabla o vista
DELETE ANY TABLE	Borrar filas de cualquier tabla o vista, y también truncar
ALTER SESSION	Alterar los parámetros de la sesión
CREATE SESSION	Conectarse a la BD

Nota: No todos los objetos se aplican a todas las clases de objetos.

La palabra clave *ANY* significa que ese usuario tiene el privilegio para todos los esquemas en la BD.

Ejemplos

GRANT ALL ON factura TO control_total;

GRANT SELECT, UPDATE ON alumno TO pepote, juancito WITH ADMIN OPTION;

GRANT SELECT ON profesor TO PUBLIC;

GRANT SELECT ON apunte TO acceso_contabilidad;

3.2.2 Instrucción REVOKE

El modo de eliminar permisos de objeto es con la instrucción *REVOKE*:

```
REVOKE [ALL {PRIVILEGES} | SELECT | INSERT | UPDATE | DELETE]
ON objeto
FROM [usuario | rol | PUBLIC]
```

Al igual que con los permisos de sistema, es posible revocar un permiso de objeto sobre uno o varios (separados por comas) usuario y/o roles. Y si se elimina el privilegio de *PUBLIC*, se eliminará de todos los usuarios actuales y futuros de la base de datos.

Ejemplo1

Otorgar todos los privilegios, *SELECT, INSERT, UPDATE, DELETE*, sobre el objeto *factura* al rol *control_total*.

GRANT ALL ON factura TO control_total;

Ejemplo2

Otorgar el privilegio de *UPDATE* sobre el objeto *alumno* a los usuarios *pepote* y *juancillo*, permitiendo que estos usuarios, a su vez puedan otorgar este privilegio a otros usuarios.

GRANT UPDATE ON alumno TO pepote, juancito WITH GRANT OPTION;

Ejemplo3

Todos los usuarios de la base de datos pueden hacer *SELECT* en el objeto *profesor*.

GRANT SELECT ON profesor TO PUBLIC;

Ejemplo4

Permitir hacer *SELECT* en el objeto *apunte* al rol *acceso_contabilidad*.

GRANT SELECT ON apunte TO acceso_contabilidad;

4. CREACIÓN DE ROLES

Los privilegios se pueden agrupar en ROLES para asignar privilegios conjuntamente a los diferentes tipos de usuarios.

> La creación de **roles** permite asignar un grupo de permisos a un usuario, y poder modificar este grupo de permisos sin tener que ir modificando todos los usuarios.

Podemos decir que un usuario normal, debe tener al menos los permisos de *CONNECT* (para conectarse) y de *RESOURCE* (para poder crear objetos).

Si asignamos un rol con 10 permisos a 300 usuarios, y posteriormente añadimos un permiso nuevo al rol, no será necesario ir añadiendo este nuevo permiso a los 300 usuarios, ya que el rol se encarga automáticamente de propagarlo.

La sintaxis básica es:

```
CREATE ROLE nombre_rol
```

Una vez que el rol ha sido creado será necesario añadirle permisos a través de instrucción *GRANT*.

Roles predefinidos de Oracle:

Rol	Privilegios
CONNECT	`alter session, create session, create cluster, create table, create view, create synonym, create sequence, create database link`
RESOURCE	`create cluster, create table, create procedure, create sequence, create trigger`
DBA	Todos los privilegios de sistema con la opción `with admin option`
EXP_FULL_DATABASE	Permisos para poder exportar toda la base de datos
IMP_FULL_DATABASE	Permisos para poder importar toda la base de datos

Ejemplo1

Podemos crear un rol llamado *creadorCuentas* que sólo pueda crear usuarios y no pueda realizar ninguna otra operación. Las sentencias que permiten hacer esto son las siguientes, podemos ver que le asignamos el privilegio de *create user* (crear usuarios) y por supuesto *create session* porque si no, no se podría conectar a la base de datos.

CREATE ROLE creadorCuentas;

GRANT CREATE SESSION, CREATE USER TO creadorCuentas;

Ejemplo2

En las tres siguientes instrucciones creamos 3 roles, *control_total, creador* y *acceso_contabilidad* pero no se les ha asignado ningún privilegio.

CREATE ROL control_total;

CREATE ROL creador;

CREATE ROL acceso_contabilidad;

Asignamos diferentes privilegios al rol *creador:*

GRANT CREATE SESSION, CREATE ANY TABLE, CREATE ANY VIEW to creador;

Faltaría asignar el rol a los diferentes usuarios, en este caso le asignaremos el ROL *creador* al usuario Pepito.

GRANT creador to Pepito;

5. PERFILES DE USUARIO

Los **perfiles** se utilizan para limitar la cantidad de recursos del sistema y de la BD disponibles para un usuario.

Si no se definen perfiles para un usuario se utiliza el perfil por defecto (**default**), que especifica recursos ilimitados.

Se pueden agrupar en dos conjuntos los parámetros:

- Los que **impiden** un **uso abusivo del sistema**.
- Parámetros referidos a las **contraseñas**.

La sintaxis básica es:

```
CREATE PROFILE nombre_perfil LIMIT
[recurso1 valor1
 recurso 2 valor2
 …];
```

Los perfiles se pueden modificar con la sentencia *ALTER PROFILE*.

Los **recursos** que pueden ser **limitados vía perfil** son los siguientes:

Recurso	**Descripción**
SESSIONS_PER_USER	Número de sesiones concurrentes que un usuario puede tener en una instancia
CPU_PER_SESSION	Tiempo de CPU, en centenas de segundos, que una sesión puede utilizar
CONNECT_TIME	Número de minutos que una sesión puede permanecer activa
IDLE_TIME	Número de minutos que una sesión puede permanecer sin que sea utilizada de manera activa
LOGICAL_READS_PER_SESSION	Número de bloques de datos que se pueden leer en una sesión
LOGICAL_READS_PER_CALL	Número de bloques de datos que se pueden leer en una operación
PRIVATE_SGA	Cantidad de espacio privado que una sesión puede reservar en la zona de SQL compartido de la SGA

Parámetros relativos a la seguridad de las **contraseñas**:

Parámetro	**Descripción**
FAILED_LOGIN_ATTEMPTS	Número máximo de intentos fallidos antes de bloquear la cuenta
PASSWORD_LIFE_TIME	Número de días de vida de la contraseña para reutilizar una contraseña
PASSWORD_REUSE_TIME	Número de días que deben transcurrir una contraseña antes de reutilizarla
PASSWORD_REUSE_MAX	Número de veces que se debe cambiar cuenta
PASSWORD_LOCK_TIME	Número de días que queda bloqueada la cuenta
PASSWORD_GRACE_TIME	Periodo de gracia de contraseñas caducadas
PASSWORD_VERIFY_FUNCTION	Función PL/SQL que dará el visto bueno a la complejidad de la contraseña

En general, el perfil por defecto debe ser adecuado para los usuarios “normales”. Los usuarios con requerimientos especiales deberían tener perfiles especiales.

Al igual que podemos asignar un *tablespace* a un usuario, también podemos asignarle un perfil (*profile*). Como hemos dicho, el principal perfil (*profile*) por defecto se denomina *default*.

Ejemplo1

Crear un perfil llamado “*desarrollo*” con los siguientes parámetros: sessions_per_user=2, Cpu_per_session=unlimited, Connect_time=480, Failed_login_attempts=2, Password_life_time=120.

```
CREATE PROFILE desarrollo LIMIT
SESSIONS_PER_USER 2
CPU_PER_SESSION UNLIMITED
CONNECT_TIME 480
FAILED_LOGIN_ATTEMPTS 2
PASSWORD_LIFE_TIME 120;
```

Ahora creamos un usuario, *david*, al cual le asignamos el perfil *desarrollo.*

```
CREATE USER david IDENTIFIED BY tititus
DEFAULT TABLESPACE users
TEMPORARY TABLESPACE temp
PROFILE desarrollo;
```

En caso de que el usuario ya esté creado, utilizamos la sentencia *ALTER USER*.

```
ALTER USER david PROFILE desarrollo;
```

6. LISTAR PRIVILEGIOS OTORGADOS. VISTAS DEL DICCIONARIO

La información de los privilegios otorgados se almacena en el diccionario de datos.

Estos datos son accesibles a través de las siguientes vistas del diccionario de datos: (Recordad que la consulta de las vistas se realiza a través de una SELECT → SELECT * FROM DBA_USERS;)

Vista	Contenidos
DBA_USERS	Todos los usuarios de la base de datos y su información, perfiles...
DBA_TS_QUOTAS	Cuotas de tablespace de los usuarios
DBA_SYS_PRIVS	Privilegios de sistema concedidos a usuarios
DBA_ROLES	Nombres de los roles y su estado del password
DBA_ROLE_PRIVS	Usuarios a los que han sido otorgados roles
DBA_SYS_PRIVS	Usuarios a los que han sido otorgados privilegios del sistema
DBA_TAB_PRIVS	Usuarios a los que han sido otorgados privilegios sobre objetos
DBA_COL_PRIVS	Usuarios a los que han sido otorgados privilegios sobre columnas de tablas.
ROLE_ROLE_PRIVS	Roles que han sido otorgados a otros roles
ROLE_SYS_PRIVS	Privilegios de sistema que han sido otorgados a roles
ROLE_TAB_PRIVS	Privilegios de tabla que han sido otorgados a roles
DBA_PROFILES	Muestra los perfiles existentes con sus límites
DBA_TABLESPACES	Tablespaces existentes en una BD

7. CASOS PRÁCTICOS USUARIOS, PRIVILEGIOS Y ROLES

7.1 Caso práctico 1

Vamos a dar privilegio *CONNECT* y *RESOURCE* a *PUBLIC*, de manera que todos los usuarios que creemos a partir de este momento tendrán permiso para conectarse.

GRANT CONNECT, RESOURCE TO PUBLIC;

Creamos un usuarios *user1.* No le asignamos ningún *tablespace*, por lo que tomará el *tablespace SYSTEM*.

CREATE USER user1
IDENTIFIED BY usuario1;

Creamos un usuario *user2.* No le asignamos ningún *tablespace*, por lo que tomará el *tablespace SYSTEM*. Recuerda que estará compartiendo *tablespace* con *user1* pero cada uno tendrá su esquema.

CREATE USER user2
IDENTIFIED BY usuario2;

Nos conectamos como *user1* y creamos una tabla ejemplo:

CREATE TABLE ejemplo (
id numeric(2,0) PRIMARY KEY,
nombre varchar(14),
loc varchar(13));

INSERT INTO ejemplo VALUES (10, 'Pepe', 'Valencia');
INSERT INTO ejemplo VALUES (20, 'María', 'Valencia');
INSERT INTO ejemplo VALUES (30, 'Juan', 'Madrid');
INSERT INTO ejemplo VALUES (40, 'Lucas', 'Alicante');

Consultamos la tabla creada desde *user1:*

*SELECT * FROM ejemplo;*

Conectamos como *user2* e intentamos consultar la tabla *ejemplo* de *user 1*, para esto tenemos que anteponer el nombre del usuario al que pertenece la tabla al nombre de la tabla:

*SELECT * FROM* user1.ejemplo;

No podemos realizar esta consulta porque no tenemos permiso para poder consultar esta tabla que pertenece a otro usuario, está en otro esquema. El mensaje que muestra es que no encuentra la tabla.

Otorgaremos permiso de consulta sobre la tabla ejemplo a *user2*. Este permiso lo puede conceder *user1*, ya que se trata de un permiso sobre un objeto que le pertenece. Por supuesto también lo podemos conceder desde *SYSTEM*.

GRANT SELECT ON ejemplo TO user2;

Ahora, desde la conexión de *user2* volvemos a intentar consultar la tabla *ejemplo* de *user1:*

*SELECT * FROM* user1.ejemplo;

Si queremos darle permiso a *user2* para que haga cualquier acción con la tabla *ejemplo* de *user1:*

GRANT ALL ON ejemplo TO user2

Lo haremos desde *user1* o desde *SYSTEM*.

7.2 Caso práctico 2

Cuando otorgamos un privilegio sobre un objeto a un usuario y este a su vez otorga ese mismo privilegio a un tercero, si revocamos el privilegio al segundo, el tercero también lo pierde. Veamos un ejemplo:

El *usuario1* le concede el privilegio al *usuario2* para poder hacer *SELECT* sobre el objeto (por ejemplo tabla) *facturas*, con la posibilidad de que el *usuario2* pueda conceder también este privilegio a otro usuario.

GRANT SELECT ON facturas WITH GRANT OPTION TO usuario2;

El usuario2 le concede al usuario3 el mismo privilegio que él recibió para el objeto *facturas*, en este caso sin *WITH GRANT OPTION:*

GRANT SELECT ON usuario1.facturas TO usuario3;

En el caso que se revoque el permiso al *usuario2* o se borre *usuario2*, *usuario3* perderá el privilegio sobre el objeto *facturas* porque fue *usuario2* quien le concedió el privilegio.

Si un usuario asigna privilegios de sistema a otro con la opción *WITH ADMIN OPTION* y a la vez este segundo asigna el mismo privilegio a un tercero, si se le revoca o se elimina el segundo usuario, este tercero mantiene el privilegio.

Es lo contrario que ocurre con los privilegios sobre objeto!