Teórico

DCL

CC BY NC SA

DCL

- Con el nombre de lenguaje de control de datos se hace referencia a la parte del lenguaje SQL que se ocupa de la seguridad, y acceso a los recursos.

CC BY NC SA

SEGURIDAD

Para Brindar Seguridad SQL provee funcionalidades que permiten proteger los datos frente a accesos, modificaciones o borrados no autorizados:

- Permite la definición de usuarios y roles mediante un mecanismo determinado, que puede ser específico de cada RDBMS(los motores respetan poco el Estándar).

- Asignación de permisos (privilegios) a usuarios y roles.

- Control de acceso a los recursos

CC BY NC SA

Usuarios

- El estándar SQL define el concepto de usuario y rol para acceder a bases de datos.

- Cada manejador de Bases de Datos implementa su propia manera de administrar los usuarios.

CC BY NC SA

Asignación de Privilegios: Comando GRANT

Sintaxis:

GRANT {ALL | *listaPrivilegios*}
ON TABLE *listaTabla*
TO {PUBLIC | *listaUsuarios*}
[WITH GRANT OPTION]

Concede todos (ALL) o un subconjunto de privilegios (*listaPrivilegios*) sobre una tabla o un conjunto de tablas (*listaTablas*) a una lista de usuarios o roles (*listaUsuarios*).

CC BY NC SA

Parámetros del comando GRANT

listaPrivilegios	Lista de privilegios, alguno de estos son: • DELETE, INSERT, SELECT, UPDATE para manipulación de datos pudiéndose especificar sobre que atributos. • References: para crear claves foráneas y cláusulas ckeck. • Execute: para poder ejecutar funciones o procedimientos almacenados. • Otros dependiendo del motor que se utilice.
ALL	Indica que todos los permisos serán asignados.
listaTablas	Lista de tabla a las cuales se asignan los privilegios, en el SQL estándar solo una.
listaUsuarios	Lista de usuarios o roles a los cuales se le otorgará el privilegio.
WITH GRANT OPTION	Define que a los usuarios a los que se ha concedido los privilegios pueden a su vez pasárselos (sólo los que se tienen actualmente) a otros usuarios (por medio de sentencias GRANT).

Ejemplo de Privilegios en MySQL

Privilegios	**Contexto**
CREATE	bases de datos, tablas e indices
DROP	bases de datos, tablas e indices
GRANT OPTION	bases de datos, tablas o procedimientos almacenados (store procedure)
REFERENCES	bases de datos o tablas
ALTER	tablas
DELETE	tablas
INDEX	tablas
INSERT	tablas
SELECT	tablas

CC BY NC SA

Privilegios	**Contexto**
UPDATE	tablas
CREATE VIEW	vistas
SHOW VIEW	vistas
ALTER ROUTINE	procedimientos almacenados
CREATE ROUTINE	procedimientos almacenados
EXECUTE	procedimientos almacenados

CC BY NC SA

Ejemplos

1) GRANT SELECT ON *personas* **TO** U1, U2

Esta instrucción otorga el permiso SELECT en la tabla *personas* a los usuarios (o Roles) U1 y U2

2) GRANT REFERENCES (*dni*) **ON** *personas* **TO** U1

Esta instrucción permite que el usuario U1(o Role) cree relaciones que hagan referencia a la clave *dni* de la tabla *personas* como clave externa (una aplicación es para que no interfieran usuarios no autorizados en la actualización de datos del campo).

3) GRANT SELECT ON *personas* **TO** U1 **WITH GRANT OPTION**

Otorga a U1 el privilegio select y permite que U1 otorgue a otros usuarios ese privilegio.

4) GRANT SELECT(*apellido, nombre***) ON** *personas* **TO** U1

Otorga a U1 el privilegio select sólo para los campos apellido y nombre. En Postgres disponibles desde la Versión 9.

Diferentes Implementaciones del Comando GRANT

En la siguientes URL se puede ver las implementaciones del comando GRANT en Postgres y MySQL respectivamente:

- http://www.postgresql.org/docs/9.3/static/sql-grant.html
- http://dev.mysql.com/doc/refman/5.5/en/grant.html

cc BY NC SA

Eliminar Privilegios: Comando REVOKE

Elimina los privilegio de usuarios o roles, su sintaxis es:

REVOKE {ALL | *listaPrivilegios*}

ON *listaTablas*

FROM {PUBLIC | *listaUsuarios*}[RESTRICT|CASCADE]

Ejemplo:

REVOKE SELECT ON *personas* **FROM** U1 **RESTRICT**

Retira al usuario(o Role) U1 el privilegio **SELECT** en la tabla *personas*, si se produce una eliminación de privilegios en cascada la instrucción fallará. Por defecto **REVOKE** es en cascada.

CC BY NC SA

RESTRICTED/CASCADE

- Si user1 da un privilegio con la opción GRANT a user2 y user2 se lo da a user3, entonces user1 puede revocar este privilegio en cascada usando la palabra clave CASCADE.

- Si user1 da un privilegio con GRANT a user2 y user2 se lo da a user3, entonces si user1 intenta revocar este privilegio, fallará si ha especificado la palabra clave RESTRICT.

  Nota: Postgres implenta estas características, Mysql no.

CC BY NC SA

Aclaración

- PostgreSQL permite al dueño del objeto revocar sus propios privilegios: por ejemplo, el dueño de una tabla puede hacer la tabla de sólo lectura revocando los privilegios INSERT, UPDATE, DELETE, y TRUNCATE. Esto no es posible en el SQL estándar.

- En Postgres los usuarios (o roles) sin permiso de login pueden considerarse grupos.

CC BY NC SA

Permisos sobre el esquema

- Sólo el propietario de un objeto puede conceder los privilegios del mismo. El propietario es siempre el creador del objeto.

- Las operaciones con el esquema de la base de datos (CREATE, DROP, etc.) sólo pueden ser realizadas por el propietario del esquema.

CC BY NC SA

Grafo de Concesión de Permisos

- Se construye un grafo por recurso y tipo de permiso.
- Los nodos del grafo son los usuarios.
- Se crea un arco de Ui a Uj, si Ui concede permisos a Uj.
- Un usuario tiene permiso si existe un camino desde la Raíz (usuario DBA o dueño del objeto) al nodo del usuario.

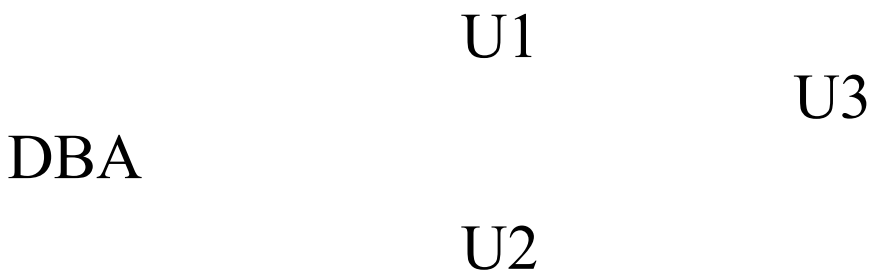
CC BY NC SA

Ejemplo

Considere los usuarios DBA, U1 , U2 y U3.

Construir el grafo para modificación de la relación personas.

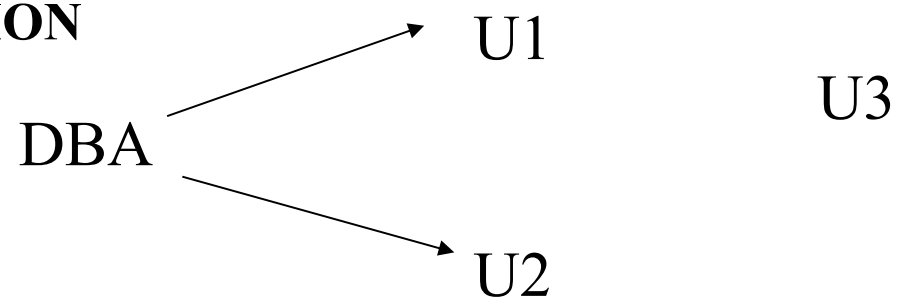
Inicialmente el grafo es:



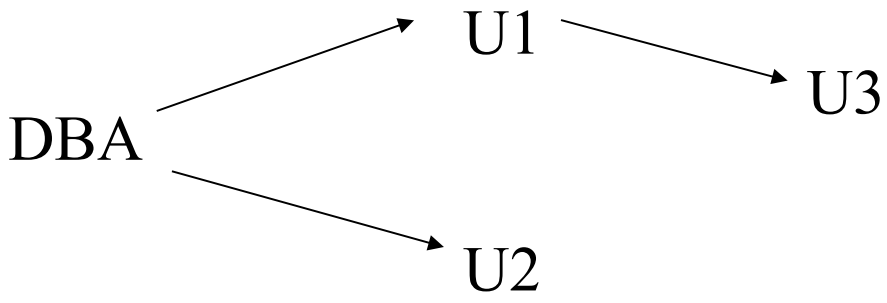
BY NC SA

Ejemplo (sigue)

- Luego de ejecutar como DBA:
  - **GRANT UPDATE ON** personas **TO** U1, U2 **WITH GRANT OPTION**

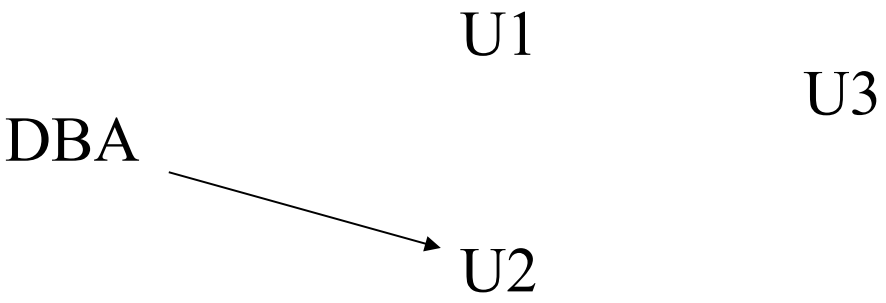


- Luego el usuario U1 ejecuta:
  - **GRANT UPDATE ON** personas **TO** U3



Ejemplo (sigue)

- Luego el usuario DBA ejecuta:
  - **REVOKE UPDATE ON** personas **TO** U1 **CASCADE**



Como DBA se quito el permisos a U1, con lo cual U3, también, perdió los privilegios.

Roles

- Se utilizan para definir “tipos de usuarios”. Por Ej.: se puede definir un role cajero con determinados privilegios, y cada vez que ingresa un cajero nuevo se le asigna el rol “cajero”, de esta forma no es necesario definir nuevamente los privilegios. Simplemente al nuevo cajero se le asigna el rol de cajero.

CC BY NC SA

Administración de Roles

- Creación de roles: **CREATE ROLE** *nombreRole*.

- Eliminación de Roles: **DROP ROLE** *nombreRole*.

- Concesión de roles a usuarios:

GRANT *role* **TO** *usuario*

- Concesión de roles a otros roles:

GRANT *role1* **TO** *role2*

CC BY NC SA

Administración de usuarios

- Es una funcionalidad de los RDBMS no estandarizada, cada RDBMS implementa su propia forma de administración.

CC BY NC SA

Manejo de Usuario en Mysql

- Para el acceso a base de datos MySQL, MySQL cuenta con usuarios, pero no implementa el concepto de role del estándar SQL.

- La instalación de mysql crea un superusuario, denominado “root”, desde la versión 5 solicita durante la instalación la contraseña. En versiones anteriores(y en algunas actuales) por defecto no tenia contraseña.

- Nombres de usuario en el sistema operativo están completamente desligados de los nombres de usuario de MySQL.

- Mysql almacena la información de usuarios y permisos en tablas de la base de datos “mysql”.

CC BY NC SA

Creación de usuarios en MYSQL

•Las cuentas de usuario están compuestas por el nombre de usuario, seguido del host desde donde accederá el usuario, por ejemplo basededatos@localhost.

•Los permisos pueden ser configurados dependiendo desde donde se conecte el usuario.

•Se puede crear cuentas MySQL de tres formas:

- Usando comando CREATE USER.
- Usando comando GRANT.
- Manipulando las tablas de permisos MySQL directamente.

CC BY NC SA

Comando CREATE USER

•La creación de cuentas mysql se puede hacer por medio de sentencia SQL:

CREATE USER *user* [IDENTIFIED BY [PASSWORD] '*password*']

•Las cuentas de usuario están compuestas por el nombre de usuario, seguido del host desde donde accederá el usuario, por ejemplo basededatos@localhost.

•Si se quiere que el usuario pueda acceder desde cualquier host se utiliza el comodín "%", por ej. basededatos@%.

•Cada usuario puede tener diferente password dependiendo del host desde donde se conecte.

Borrado de cuentas de usuario

- Se utiliza el comando DROP *user.*
- Antes de la versión 5.0.2 se podían eliminar sólo las cuentas que no tuvieran asignado permisos.
- Desde MySQL 5.0.2, se pueden borrar cuentas y sus permisos.

CC BY NC SA

Asignación de Límites en las Cuentas

•MySQL desde la versión 5.0, puede limitar los siguientes recursos de servidor para cuentas individuales:

- MAX_QUERIES_PER_HOUR: número de consultas que una cuenta puede realizar por hora.
- MAX_UPDATES_PER_HOUR: número de actualizaciones que una cuenta puede hacer por hora.
- MAX_CONNECTIONS_PER_HOUR: número de veces que una cuenta puede conectar con el servidor por hora.
- MAX_USER_CONNECTIONS (Desde MySQL 5.0.3): cantidad de conexiones simultaneas que puede tener el usuario

• Estos controles son interesantes para muchos administradores de MySQL, particularmente aquellos que trabajan en servicios de hosting.

cc BY NC SA

Ejemplo

Ejemplo de utilización del comando GRANT para limitar recursos:

```
GRANT ALL ON *.* TO ‘basededatos'@'localhost' WITH
        MAX_QUERIES_PER_HOUR 20
        MAX_UPDATES_PER_HOUR 10
        MAX_CONNECTIONS_PER_HOUR 5
        MAX_USER_CONNECTIONS 2;
```

CC BY NC SA

Administración de MySQL

- Utilizar la herramienta MysqlWorkbench(actual), phpMyAdmin(utilizada en servicios de Hosting) o MySQLAdministrator(va perdiendo compatibilidad con las nuevas versiones del Motor), entre otras...

BY NC SA

Manejo de Usuarios y Roles en Postgres

- En PostgreSQL los usuarios son Roles, Role es el concepto general.

- En PostgreSQL el concepto de Usuario, Role y Grupo son lo mismo. La diferencia es si tiene permiso de login a la bases de datos o no.

- La instalación del motor crea un superusuario postgres.

- La instalación de PostgreSQL crea una cuenta de usuario postgres en el sistema operativo.

CC BY NC SA

Creación de Usuarios PostgreSQL

Se pueden crear usuarios de dos formas:

- Programa "createuser", aplicación provista en la instalación del motor.
- Utilizando las sentencia SQL CREATE USER o CREATE ROLE.

CC BY NC SA

CREATE ROLE

CREATE ROLE *name* [[WITH] *option* [...]]

Donde *option* puede ser:

| {SUPERUSER | NOSUPERUSER }
| {CREATEDB | NOCREATEDB} // Puede crear o no base de datos
| {CREATEROLE | NOCREATEROLE} // Puede o no crear nuevos Usuarios
| {CREATEUSER | NOCREATEUSER} // Puede o no crear nuevos Usuarios
| {INHERIT | NOINHERIT } // El nuevo Role hereda los permisos de los Roles que es miembro
| {LOGIN | NOLOGIN}
| CONNECTION LIMIT *connlimit* // Cuantas conexiones concurrentes puede tener el nuevo Role
| {[ENCRYPTED | UNENCRYPTED] PASSWORD '*password*' }
| VALID UNTIL '*timestamp*'
| IN ROLE *rolename* [, ...] // El nuevo Role será agregado como miembro de estos Roles
| IN GROUP *rolename* [, ...] // El nuevo Role será agregado como miembro de estos Roles
| ROLE *rolename* [, ...] // Roles que serán agregados como miembros del nuevo Role
....

Programa createuser

createuser [opciones] [nombre_usuario]

Algunas opciones:

-a (o –r) : el nuevo usuario puede crear usuarios.

-A(o –R) : el nuevo usuario NO puede crear usuarios.

-d : nuevo usuario puede crear base de datos.

-D: nuevo usuario NO puede crear base de datos.

-P : pide el password del nuevo usuario.

-E : encripta el password.

-U : usuario que creara el nuevo usuario (por defecto se conecta usando el nombre de usuario del sistema).

-h :nombre_maquina: Especifica desde qué maquina se conectará.

BY NC SA

Eliminación de usuarios

Se pueden eliminar usuarios de dos formas:

- Programa "dropuser", aplicación provista en la instalación del motor.
- Utilizando la sentencia SQL DROP USER o DROP ROLE.

CC BY NC SA

Acceso de clientes

- PostgreSQL, al igual que MySQL, permite configurar los host desde donde se pueden establecer conexiones.
- Por defecto PostgreSQL permite conexiones al servidor sólo desde localhost.
- Para permitir accesos desde otros host se debe configurar desde el archivo pg_hba.conf, por ejemplo para permitir que se pueda acceder desde cualquier host:

```
# TYPE  DATABASE    USER        CIDR-ADDRESS          METHOD
host          all          all              0.0.0.0/0          md5
```

CC BY NC SA

Administración de grupos

Para la creación de grupos se utiliza el comando SQL (es un alias de create user):

CREATE GROUP *groupName*
[WITH [SYSID *groupid*]
[USER *username* [, ...]]]

- Para agregar o quitar usuarios del grupo:

ALTER GROUP *groupName* { ADD | DROP } USER *username* [,...]
(Equivalente a GRANT *groupName* TO *username* [,...])

- Para eliminar un grupo:

DROP GROUP *groupName*

CC BY NC SA

Administración de Postgres

- Utilizar la herramienta PGAdmin III o phpPgAdmin.
- Datos de usuario DBA por defecto:

  usuario:postgres

  clave: la solicita el instalador o se define en la configuración del servidor.

BY NC SA

Manejo de Usuarios y Roles en Oracle XE

- La creación de un usuario en Oracle implica la creación de un esquema de base de datos(Versión 10XE)(11XE similar).

- Permite la creación de roles de usuario.

- La instalación por defecto sólo permite conexiones desde localhost.

CC BY NC SA

Creación de usuario (ejemplo)

```
CREATE USER usuario
        IDENTIFIED BY usuario
        DEFAULT TABLESPACE user
        QUOTA 10M ON user
        TEMPORARY TABLESPACE temp
        QUOTA 5M ON system
        PASSWORD EXPIRE;
```

CC BY NC SA

Administración de roles

- Oracle XE tiene por defecto tres roles para asignar a los usuario, estos son:
  - CONNECT: Para conectarse a la base de datos,
  - RESOURCE: Para desarrolladores (permite crear objetos de la base de datos, pero no vistas) .
  - DBA: Para administradores de la base de datos.

CC BY NC SA

Asignación de Limites(Perfiles)

- Oracle permite definir limites al uso de recursos a los usuario, tanto en espacio de almacenamiento, como uso de CPU.

- Oracle permite definir limites mediante perfiles de usuario (profiles), por ejemplo:

```
CREATE PROFILE desarrolador LIMIT
        SESSIONS_PER_USER 1
        IDLE_TIME 30
        CONNECT_TIME 600;
```

CC BY NC SA

Tablespace en Oracle(10XE)

- Un tablespace es una unidad lógica de almacenamiento dentro de una base de datos Oracle.

- Es un puente entre el sistema de archivos del sistema operativo y la base de datos.

- Cada tablespace se compone de, al menos, un datafile y un datafile sólo puede pertenecer a un tablespace.

- Cada tabla o indice de Oracle pertenece a un tablespace, es decir cuando se crea una tabla o indice se crea en un tablespace determinado.

CC BY NC SA

Tipos de Tablespace

- SYSTEM
  - Te crea automáticamente con la instalación de Oracle o al crear una base de datos.
  - Contiene el diccionario de datos.
- TEMP
  - Utilizado para objetos temporales.

Nota: No hay ninguna relación entre schema y tablespace.

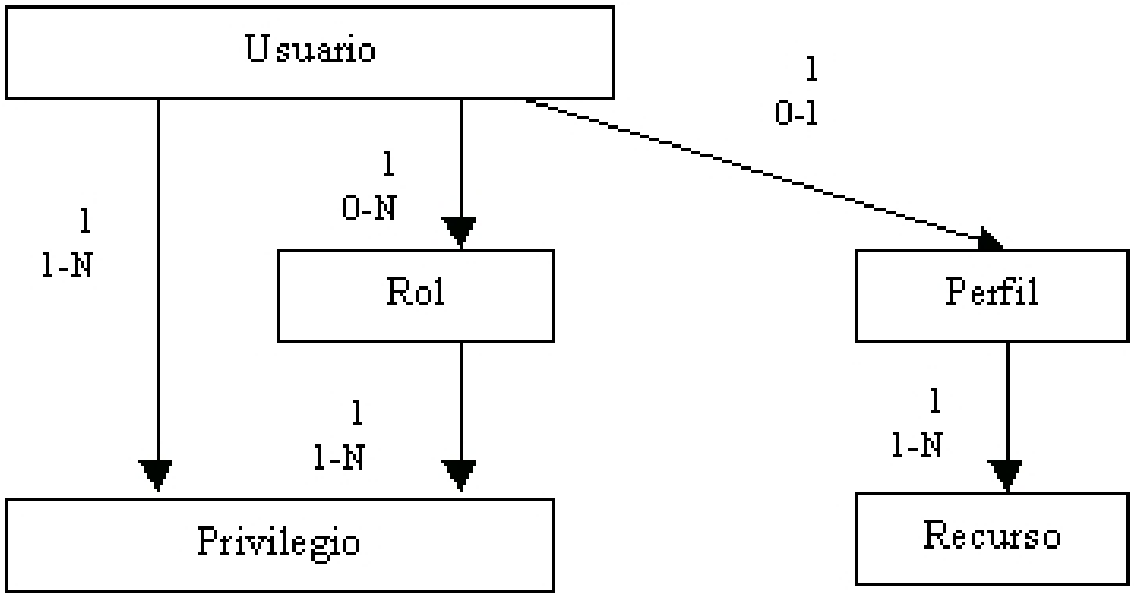
CC BY NC SA

Oracle (11XE)

- Usuario DBA admin (clave la ingresada en la instalación)
- Internal es el Workspace por defecto, que permite la administración.
- Cada Workspace tiene sus propios usuarios y esquemas.

cc BY NC SA

Resumen de conceptos



CC BY NC SA

Accesos a aplicación de administración de Oracle

- Una vez instalado Oracle podrá administrar el motor de bases de datos por medio de http://localhost:8080/apex

Versión 10XE
usuario: system
clave: el instalador la solicita

Versión 11XE

workspace:internal
usuario: ADMIN
clave: el instalador la solicita

CC BY NC SA