# Unidad. Lenguaje de control de datos (DCL)

Apuntes de BD para DAW, DAM y ASIR

José Juan Sánchez Hernández

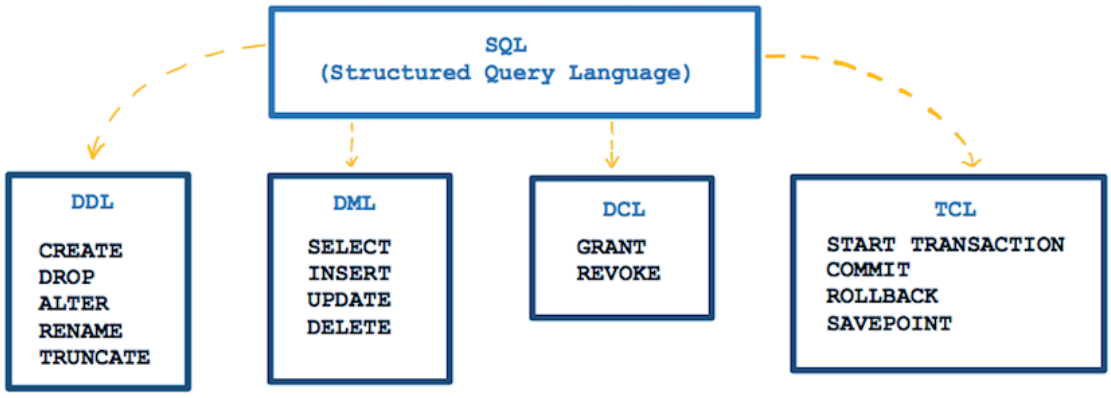
Curso 2023/2024

# Índice

# Índice de figuras

# Índice de cuadros

# 1 El lenguaje DCL de SQL






El **DCL** (*Data Control Language*) o **Lenguaje de Control de Datos** es la parte de SQL dedicada a controlar el acceso a los datos de una base de datos. Las sentencias **DCL** más utilizadas son las siguientes:

- `GRANT`: se utiliza para asignar permisos a un usuario o rol.
- `REVOKE`: se utiliza para eliminar los permisos que se han asignado a un usuario o rol.

## 1.1 Gestión de usuarios en MySQL Server

### 1.1.1 Crear un nuevo usuario

La sintaxis simplificada para crear un usuario en MySQL es la siguiente:

```
CREATE USER [IF NOT EXISTS]
    user [auth_option] [, user [auth_option]] ...
    DEFAULT ROLE role [, role ] ...
    [REQUIRE {NONE | tls_option [[AND] tls_option] ...}]
    [WITH resource_option [resource_option] ...]
    [password_option | lock_option] ...
    [COMMENT 'comment_string' | ATTRIBUTE 'json_object']
```

**Ejemplo:**

Habrá que reemplazar *nombre_usuario* y *contraseña* por los datos del nuevo usuario que desea crear:

```
CREATE USER 'nombre_usuario'@'localhost' IDENTIFIED BY 'contraseña';
```

Una vez que hemos creado el usuario hay que asignarle permisos para que pueda acceder a la/s base/s de datos que queramos.

Referencia:

- Documentación oficial de CREATE USER en MySQL

### 1.1.2 Eliminar un usuario

La sintaxis para eliminar un usuario en MySQL es la siguiente:

```
DROP USER [IF EXISTS] nombre_usuario [, nombre_usuario] ...
```

**Ejemplo:**

```
DROP USER IF EXISTS 'nombre_usuario'@'localhost';
```

Referencia:

- Documentación oficial de DROP USER en MySQL

### 1.1.3 Obtener el listado de usuarios

Los usuarios de MySQL se almacenan en la tabla `mysql.user`. La clave primaria de esta tabla está formada por los valores `user` y `host`, de modo que cada fila vendrá identificada por un nombre de usuario y el host desde el que puede conectarse.

La siguiente consulta nos devuelve el listado de usuarios que tenemos en MySQL y desde qué host pueden conectarse:

```
SELECT user,host FROM mysql.user;
```

En nuestra caso la consulta anterior devuelve el siguiente resultado:

```
+------------------+--------------+
| user             | host         |
+------------------+--------------+
| root             | localhost    |
| debian-sys-maint | localhost    |
| mysql.session    | localhost    |
| mysql.sys        | localhost    |
+------------------+--------------+
```

La columnna `host` indica desde qué host puede conectarse el usuario. Algunos de los valores que podemos encontrar en esta columna son los siguientes:

- `localhost`: el usuario sólo puede conectarse desde la máquina local.
- %: el usuario puede conectarse desde cualquier dirección IP.
- Una dirección IP específica, para indicar que el usuario sólo podrá conectarse desde esa dirección IP.
- Una dirección IP con el comodín % para indicar un rango de direcciones IP.

**Ejemplo:**

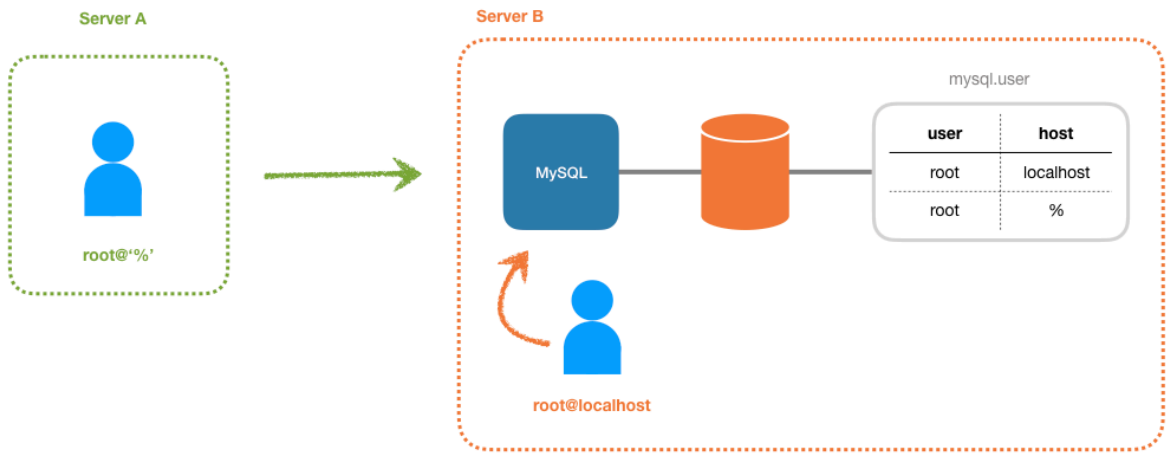
Imagina que la tabla `mysql.user` tiene los siguientes valores:

```
+------------------+--------------+
| user             | host         |
+------------------+--------------+
| user1            | localhost    |
| user2            | %            |
| user3            | 172.16.0.11  |
| user4            | 172.16.%     |
+------------------+--------------+
```

- En este ejemplo el usuario `user1` sólo puede conectarse desde `localhost`.
- El usuario `user2` puede conectarse desde cualquier dirección IP.
- El usuario `user3` sólo puede conectarse desde la dirección IP `172.16.0.11`.
- El usuario `user4` sólo puede conectarse desde direcciones IP que empiecen por `172.16.`, que sería equivalente a la red `172.16.0.0/16`.

**Ejemplo:**

El siguiente diagrama muestra un ejemplo de dos usuarios que se están conectando a una máquina con MySQL Server. El usuario `root@localhost` es un usuario que sólo puede conectarse desde la máquina local, mientras que el usuario `root@'%'` es un usuario que se puede conectar desde una máquina remota.



## 1.2 Gestión de privilegios o permisos

Los privilegios o permisos en MySQL se utilizan para determinar qué operaciones puede realizar un usuario.

Las operaciones que puede realizar un usuario pueden ser:

- operaciones de administración del servidor de MySQL,
- operaciones sobre todas las bases de datos y todos los objetos que contienen,
- operaciones sobre objetos específicos de una base de datos.

MySQL hace una distinción entre privilegios estáticos y dinámicos:

- **Privilegios estáticos**: Están integrados en el servidor.
- **Privilegios dinámicos**: Están disponibles sólo cuando se ha habilitado el componente que los implementa.

### 1.2.1 Tipos de permisos estáticos que podemos aplicar

La lista de permisos estáticos que podemos asignar a un usuario en MySQL es muy extensa. A continuación, se muestra una lista con algunos de los permisos que vamos a utilizar con más frecuencia:

- `ALL PRIVILEGES`: con esta opción podemos asignar todos los privilegios de una vez.
- `CREATE`: permite crear nuevas tablas o bases de datos.
- `DROP`: permite eliminar tablas o bases de datos
- `DELETE`: permite eliminar registros de tablas.
- `INSERT`: permite insertar registros en tablas.
- `SELECT`: permite leer registros en las tablas.
- `UPDATE`: permite actualizar registros seleccionados en tablas.
- `WITH GRANT OPTION`: permite a un usuario asignar sus privilegios a otros usuarios.

Puede consultar la lista completa de permisos estáticos en la documentación oficial de MySQL.

### 1.2.2 Asignar permisos a un usuario

La sintaxis simplificada para asignar permisos a un usuario en MySQL es la siguiente:

```
1  GRANT permiso ON nombre_base_de_datos.nombre_tabla TO 'nombre_usuario'@'
       localhost';
```

**Ejemplo 1:**

```
1  GRANT ALL PRIVILEGES ON *.* TO 'nombre_usuario'@'localhost';
```

En este comando, los asteriscos indican que estamos aplicando el permiso `ALL PRIVILEGES` al usuario `nombre_usuario` para todas las tablas de cada una de las bases de datos.

Después de este comando habrá que ejecutar el siguiente comando para refrescar todos los privilegios a los usuarios.

```
1  FLUSH PRIVILEGES;
```

Referencia:

- Documentación oficial de GRANT en MySQL

**Ejemplo 2:**

En este ejemplo vamos a crear la base de datos `prestashop` y el usuario `user@localhost`, y le vamos a asignar todos los permisos sobre la base datos.

```
1  -- Creamos la base de datos prestashop
2  DROP DATABASE IF EXISTS prestashop;
3  CREATE DATABASE prestashop CHARACTER SET utf8mb4;
4  USE prestashop;
5
6  -- Creamos un usuario y le asignamos todos los persisos
7  DROP USER IF EXISTS 'user'@'localhost';
8  CREATE USER 'user'@'localhost' IDENTIFIED BY 'password';
9  GRANT ALL PRIVILEGES ON prestashop.* TO 'user'@'localhost';
10
11 -- Refrescamos los privilegios si queremos que se apliquen los cambios
       inmediatamente
12 FLUSH PRIVILEGES;
```

**Ejemplo 3:**

En este ejemplo vamos a crear la base de datos `prestashop` y el usuario `user@localhost`, pero esta vez sólo le vamos a asignar al usuario los permisos de `SELECT`, `INSERT`, `UPDATE` y `DELETE`.

```
1  -- Creamos la base de datos prestashop
2  DROP DATABASE IF EXISTS prestashop;
3  CREATE DATABASE prestashop CHARACTER SET utf8mb4;
4  USE prestashop;
5
6  -- Creamos un usuario y le asignamos todos los persisos
7  DROP USER IF EXISTS 'user'@'localhost';
8  CREATE USER 'user'@'localhost' IDENTIFIED BY 'password';
9  GRANT SELECT, INSERT, UPDATE, DELETE ON prestashop.* TO 'user'@'localhost';
10
11 -- Refrescamos los privilegios si queremos que se apliquen los cambios
       inmediatamente
12 FLUSH PRIVILEGES;
```

**Ejemplo 4:**

En este ejemplo vamos a utilizar la opción `WITH GRANT OPTION` para que el usuario pueda asignar sus privilegios a otros usuarios.

```
1  -- Creamos la base de datos prestashop
2  DROP DATABASE IF EXISTS prestashop;
3  CREATE DATABASE prestashop CHARACTER SET utf8mb4;
4  USE prestashop;
5
6  -- Creamos un usuario y le asignamos todos los persisos
```

```
7  DROP USER IF EXISTS 'user'@'localhost';
8  CREATE USER 'user'@'localhost' IDENTIFIED BY 'password';
9  GRANT ALL PRIVILEGES ON prestashop.* TO 'user'@'localhost' WITH GRANT OPTION;
10
11 -- Refrescamos los privilegios si queremos que se apliquen los cambios
       inmediatamente
12 FLUSH PRIVILEGES;
```

Como el usuario `'user'@'localhost'` tiene la opción `WITH GRANT OPTION` y tiene todos los permisos (`ALL PRIVILEGES`) sobre todas las tablas de base de datos `prestashop`, podrá asignar asignar estos mismos permisos a otros usuarios sobre las mismas tablas de la base de datos `prestashop`.

**Ejemplo 5:**

En el siguiente ejemplo el usuario `'user'@'localhost'` sólo podrá asignar a otros usuarios el permiso de `SELECT` sobre la tabla `customer` de la base de datos `prestashop`.

```
1  GRANT SELECT ON prestashop.customer TO 'user'@'localhost' WITH GRANT OPTION;
```

### 1.2.3 Eliminar permisos a un usuario

La sintaxis simplificada para elminar permisos a un usuario en MySQL es la siguiente:

```
1  REVOKE permiso ON nombre_base_de_datos.nombre_tabla FROM 'nombre_usuario'@'
       localhost';
```

Referencia:

- Documentación oficial de REVOKE en MySQL

**Ejemplo:**

En este ejemplo vamos a eliminar los permisos de `INSERT`, `UPDATE` y `DELETE`, que tiene el usuario `user@localhost` sobre todas las tablas de la base de datos `prestashop`.

```
1  REVOKE INSERT, UPDATE, DELETE ON prestashop.* FROM 'user'@'localhost';
```

Si queremos que los cambios se apliquen inmediatamente, tendremos que ejecutar la sentencia:

```
1  FLUSH PRIVILEGES;
```

### 1.2.4 Cómo consultar los permisos de un usuario

También podemos consultar qué permisos específicos tiene un determinado usuario. La siguiente consulta nos devuelve los permisos que tiene el usuario `root`:

```
1  SHOW GRANTS FOR root@localhost;
```

```
1  +---------------------------------------------------+
2  | Grants for root@localhost                         |
3  +---------------------------------------------------+
```

```
| GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' |
+---------------------------------------------------+
```

## 1.3 Gestión de roles

Un rol es un conjunto de privilegios que se pueden asignar a uno o más usuarios.

### 1.3.1 Crear un rol

La sintaxis para crear un rol en MySQL es la siguiente:

```
CREATE ROLE [IF NOT EXISTS] nombre_rol [, nombre_rol ] ...
```

Para asignar privilegios a un rol se utiliza la sentencia `GRANT`:

```
GRANT permiso ON nombre_base_de_datos.nombre_tabla TO nombre_rol;
```

Para asignar un rol a un usuario también se utiliza la sentencia `GRANT`:

```
GRANT nombre_rol TO nombre_usuario;
```

Referencia:

- Documentación oficial de `CREATE ROLE` en MySQL

**Ejemplo:**

En este ejemplo vamos a crear tres roles: `rol_lectura_escritura`, `rol_lectura` y `rol_escritura`.

```
DROP ROLE IF EXISTS 'rol_lectura_escritura', 'rol_lectura', 'rol_escritura';
CREATE ROLE 'rol_lectura_escritura', 'rol_lectura', 'rol_escritura';
```

Ahora vamos a asignar los privilegios que tendrá cada rol y sobre qué base datos tendrá estos privilegios.

```
GRANT ALL ON base_de_datos.* TO 'rol_lectura_escritura';
GRANT SELECT ON base_de_datos.* TO 'rol_lectura';
GRANT INSERT, UPDATE, DELETE ON base_de_datos.* TO 'rol_escritura';
```

Finalmente, vamos a crear varios usuarios y vamos a asignarles los roles que hemos creado:

```
-- Creamos los usuarios
DROP USER IF EXISTS admin;
CREATE USER admin@'localhost' IDENTIFIED BY 'password1';

DROP USER IF EXISTS usuario_lectura_1;
CREATE USER usuario_lectura_1@'localhost' IDENTIFIED BY 'password2';

DROP USER IF EXISTS usuario_lectura_2;
CREATE USER usuario_lectura_2@'localhost' IDENTIFIED BY 'password3';

```

```
DROP USER IF EXISTS usuario_escritura_1;
CREATE USER usuario_escritura_1@'localhost' IDENTIFIED BY 'password4';

DROP USER IF EXISTS usuario_escritura_2;
CREATE USER usuario_escritura_2@'localhost' IDENTIFIED BY 'password5';

-- Asignamos los roles a los usuarios
GRANT 'rol_lectura_escritura' TO admin@'localhost';
GRANT 'rol_lectura' TO usuario_lectura_1@'localhost', usuario_lectura_2@'
    localhost';
GRANT 'rol_escritura' TO usuario_escritura_1@'localhost', usuario_escritura_2@'
    localhost';
```

### 1.3.2 Eliminar un rol de un usuario

```
REVOKE nombre_rol FROM nombre_usuario;
```

### 1.3.3 Eliminar un rol

La sintaxis para eliminar un rol en MySQL es la siguiente:

```
DROP ROLE [IF NOT EXISTS] nombre_rol [, nombre_rol ] ...
```

Referencia:

- Documentación oficial de `REMOVE ROLE` en MySQL

### 1.3.4 Obtener un listado de los roles existentes

Los roles se almacenan en la tabla `mysql.user` como si fueran usuarios pero con algunos valores específicos las columnas `account_locked`, `password_expired` y `authentication_string`.

- `account_locked`: Indica si la cuenta de usuario está bloqueada.
- `password_expired`: Indica si el password del usuario ha expirado.
- `authentication_string`: Contiene el hash del password del usuario.

Para obtener el listado de roles en MySQL podemos ejecutar la siguiente consulta:

```
SELECT mysql.user.user
FROM mysql.user
WHERE mysql.user.account_locked='Y' AND mysql.user.password_expired='Y' AND
    mysql.user.authentication_string='';
```

## 1.4 Ejercicios

1. Crea una base de datos llamada `wordpress` para la aplicación web WordPress.

2. Crea un usuario llamado `wp_local_user` que tenga todos los privilegios sobre la base de datos `wordpress`. Tenga en cuenta que el usuario `wp_local_user` sólo podrá conectarse desde la máquina local.
3. Crea un usuario llamado `wp_remote_user` que tenga todos los privilegios sobre la base de datos `wordpress` y que pueda conectarse desde cualquier máquina.
4. Crea un usuario llamado `wp_read_user` que sólo tenga permisos de lectura sobre la base de datos `wordpress` y que pueda conectarse desde cualquier máquina.
5. Vuelva a crear un usuario llamado `wp_read_user` que tenga todos los privilegios sobre la base de datos `wordpress` y que sí pueda conectarse desde cualquier máquina. Utilice una contraseña diferente a la que utilizó para el usuario anterior.
6. Quítele los privilegios de `CREATE`, `DROP`, `INSERT`, `DELETE` y `UPDATE` al usuario `wp_read_user` que puede conectarse desde cualquier máquina sobre a base de datos `wordpress`.
7. Muestre un listado de todos lo usuarios que ha creado en MySQL.
8. Muestre los permisos que tiene el usuario `wp_read_user` que puede conectarse desde cualquier máquina.
9. Elimine el usuario `wp_read_user` que puede conectarse desde cualquier máquina.

# 2 Licencia