

Tema 13

Usuarios y privilegios.

13.1. Usuarios.

Podemos ver el usuario actual:

```
SELECT USER();
```

Muestra cómo intentamos entrar (*loguearnos*).

```
SELECT CURRENT_USER();
```

Muestra el usuario con el que nos hemos conectado realmente.

La mayoría de las veces mostrarán la misma información.

Podemos ver todos los usuarios que hay:

```
SELECT user FROM mysql.user;
```

La tabla contiene mucha más información:

```
DESCRIBE mysql.user;
```

```
SELECT User, Host, authentication_string FROM mysql.user;
```

Vemos que en *Host* aparecen dos valores (los más comunes) (Figura 13.1):

- **localhost**: el *host* aparece especificado en la conexión y será *localhost*. Esto impide que dichos usuarios se conecten a la BdD si es en local.
Se podría especificar otra máquina y el usuario solo podría conectarse desde ella.
- **%**: el *host* NO aparece especificado por lo que deberá hacerse y por lo tanto permite conectarse desde otras máquinas.

	User	Host
▶	admin00	%
	mysql.infoschema	localhost
	mysql.session	localhost
	mysql.sys	localhost
	root	localhost

Figura 13.1: Columna *Host*.

También podemos ver qué usuarios están conectados en este momento:

```
SELECT user, host, db, command FROM information_schema.processlist;
```

13.1.1. Manipular usuarios.

La sentencia para **crear** usuarios es muy compleja pero nosotros solo veremos la forma más básica:

```
CREATE USER nombre_usuario IDENTIFIED BY contrasenia;
```

Por ejemplo:

```
CREATE USER 'cristina'@'localhost' IDENTIFIED BY 'htRd34f$rr';
```

O la usada para crear el usuario administrador de nuestra máquina virtual:

```
CREATE USER 'admin00'@'%' IDENTIFIED BY 'alumno';
```

Podemos **eliminar** un usuario:

```
DROP USER nombre_usuario;
```

por ejemplo:

```
DROP USER 'cristina'@'localhost';
```

Aunque permite muchas opciones (como la creación) podemos **modificar** los usuarios pero no entraremos en detalles.

Para modificar la contraseña:

```
ALTER USER 'cristina'@'localhost' IDENTIFIED BY 'po5TYf&hj&n';
```

NOTA:

En MySQL se puede usar SET PASSWORD pero en la propia documentación se indica que es preferible hacerlo con ALTER USER.

13.2. Privilegios (permisos).

En MySQL hay tres niveles de privilegios:

1. Administrativos: permiten manipular el funcionamiento del servidor.
2. De base de datos: afectan a una base de datos y todos los objetos que contiene. Se pueden conceder para una base de datos concreta o de manera global (sobre todas las BdD).
3. De objetos de BdD (tablas, vistas, índices, etc.): se pueden conceder para un objeto, para todos los de una BdD o de manera global.

Puedes consultar todos los **privilegios** existentes en la documentación oficial. Verás que los hay dinámicos y estáticos. No entramos en detalles pero brevemente:

- Estáticos: existen siempre por lo que pueden concederse o revocarse en cualquier momento.

- Dinámicos: van asociados a elementos de la ejecución del SGBD por lo que para poder concederse o revocarse dicho elemento debe encontrarse en ejecución. NO VEREMOS NINGUNO DE ESTE TIPO.

13.2.1. Conceder privilegios.

Sintaxis sencilla:

```
GRANT <privilege_name>  
ON <object_name>  
TO [ <user_name> | <role_name> ]  
[WITH GRANT OPTION];
```

Por ejemplo:

```
GRANT CREATE, ALTER, DROP, INSERT, UPDATE, DELETE, SELECT,  
REFERENCES, RELOAD  
ON *.*  
TO 'cristina'@'localhost'  
WITH GRANT OPTION;
```

Concede los permisos de la lista, en todos los objetos de todas las BdD al usuario cristina.

WITH GRANT OPTION permite que el usuario asigne los permisos que tiene a otros usuarios.

Si quisiéramos conceder los permisos a solo determinados elementos:

- *.*: todos los objetos de todas las bases de datos.
- nombre_bdd.*: todos los objetos de la base de datos indicada por nombre_bdd
- nombre_bdd.nombre_objeto: el objeto indicado por nombre_objeto de la base de datos indicada por nombre_bdd

Por ejemplo:

```
GRANT INSERT, UPDATE, DELETE, SELECT
ON Conciertos.Compositor
TO 'carmen'@'localhost';
```

Es posible incluso proporcionar privilegios para determinadas columnas:

```
GRANT
    SELECT (id_Compositor, nombre, apellido),
    UPDATE(apellido)
ON Conciertos.Compositor
TO 'carmen'@'localhost';
```

Cuando creamos el usuario admin00 lo hicimos con:

```
GRANT ALL PRIVILEGES ON *.* TO 'admin00'@'%' WITH GRANT OPTION;
```

Que es la forma de asignarle todos los privilegios.

En gran parte de la bibliografía se indica que se debe realizar el siguiente comando (y nosotros lo hicimos):

```
FLUSH PRIVILEGES;
```

Para confirmar la asignación de privilegios. Sin embargo, en la propia [documentación](#) de MySQL nos indican que no es necesario. Esta documentación no pretende ser una guía únicamente de MySQL aunque sea el SGBD que usamos.

Para consultar los permisos que tiene un usuario se haría con:

```
SHOW GRANTS FOR 'cristina'@'localhost';
```

13.2.2. Revocar privilegios.

Sintaxis sencilla:

```
REVOKE [IF EXISTS]
privilege_name
ON object_name
FROM user_or_role;
```

```
REVOKE [IF EXISTS] ALL [PRIVILEGES], GRANT OPTION
FROM user_or_role;
```

Por ejemplo:

```
REVOKE INSERT, UPDATE, DELETE, SELECT
ON Conciertos.Compositor FROM 'carmen'@'host';
```

```
REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'carmen'@'host';
```

13.3. Roles.

Permiten agrupar usuarios que deban tener los mismos privilegios.

Es un conjunto de privilegios a los que se les da un nombre.

Para asignar los mismos privilegios a un conjunto de usuarios a través de un rol:

1. Crea el rol.
2. Concede los privilegios a ese rol.
3. Asigna el rol a los usuarios.
4. Activar el rol.

13.3.1. Crear y asignar roles.

Podemos crear un rol:

```
CREATE ROLE conciertos_admin;
```

O varios:

```
CREATE ROLE
    conciertos_consultar,
    conciertos_modificar;
```

Asignamos privilegios a los roles:

```
GRANT ALL ON Conciertos.* TO conciertos_admin;
GRANT SELECT ON Conciertos.Musico TO conciertos_consultar;
GRANT SELECT ON Conciertos.Compositor TO conciertos_consultar;
GRANT INSERT, UPDATE, DELETE ON Conciertos.*
    TO conciertos_modificar;
```

Solo para el ejemplo, vamos a crear unos usuarios a los que asignarles los roles:

```
CREATE USER 'ludwig'@'localhost' IDENTIFIED BY 'uno';
CREATE USER 'johann'@'%' IDENTIFIED BY 'dos';
CREATE USER 'sebastian'@'localhost' IDENTIFIED BY 'tres';
CREATE USER 'friedrich'@'localhost' IDENTIFIED BY 'cuatro';
```

Y por último asignamos los roles a los usuarios:

```
GRANT conciertos_admin TO 'ludwig'@'localhost';
GRANT conciertos_consultar
    TO 'johann'@'%', 'sebastian'@'localhost';
GRANT conciertos_consultar, conciertos_modificar
    TO 'friedrich'@'localhost';
```

En MySQL es necesario **activar** los privilegios de los usuarios si se han asignado a través de un rol:

```
SET DEFAULT ROLE ALL TO
    'ludwig'@'localhost',
    'johann'@'localhost',
    'sebastian'@'localhost',
    'friedrich'@'localhost';
```

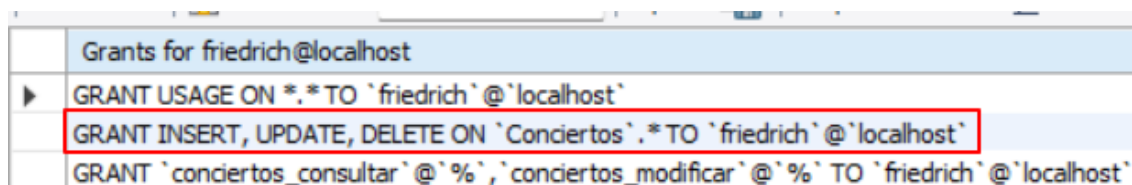
Ahora podemos **consultar** los privilegios:

```
SHOW GRANTS FOR 'friedrich'@'localhost';
```

Pero nos mostrará los roles que tiene asignados.

Si queremos ver qué le permiten dichos permisos (Figura 13.2):

```
SHOW GRANTS FOR 'friedrich'@'localhost'  
    USING conciertos_modificar;
```



Grants for friedrich@localhost
GRANT USAGE ON *.* TO `friedrich`@`localhost`
GRANT INSERT, UPDATE, DELETE ON `Conciertos`.* TO `friedrich`@`localhost`
GRANT `conciertos_consultar`@`%`,`conciertos_modificar`@`%` TO `friedrich`@`localhost`

Figura 13.2: Privilegios de un usuario por un rol.

13.3.2. Revocar privilegios de roles en MySQL.

Se hace de manera análoga a con los usuarios:

```
REVOKE INSERT, UPDATE, DELETE  
ON Conciertos.*  
FROM conciertos_modificar;
```

Para eliminar la asignación de un rol a un usuario:

```
REVOKE conciertos_consultar  
FROM 'sebastian'@'localhost';
```

Y por último, para eliminar uno o más roles:

```
DROP ROLE conciertos_consultar, conciertos_modificar;
```

Ejercicios 13.1

1. Crea un usuario local con tu nombre y la contraseña que quieras. Prueba a entrar. Al ser local, tendrás que hacerlo desde la propia máquina del servidor.
Recuerda que se hace desde la línea de comandos con:


```
mysql -u nombre_usuario -p
```

2. Modifica la contraseña y comprueba que se ha aplicado intentando entrar.
3. Elimina el usuario creado.
4. Crea el usuario con tu nombre pero ahora debe poder conectarse en remoto. Pruébalo desde *Workbench*.
5. Consulta en la documentación los siguientes privilegios y explica qué permiten hacer (ROUTINE, EXECUTE y TRIGGER tienen que ver con los procedimientos almacenados que veremos el tema siguiente. Intenta entender qué hacen pero no te preocupes si no los comprendes totalmente.):
 - ALL [PRIVILEGES]
 - ALTER
 - ALTER ROUTINE
 - CREATE
 - CREATE ROLE
 - CREATE ROUTINE
 - CREATE TABLESPACE
 - CREATE USER
 - CREATE VIEW
 - DELETE
 - DROP
 - DROP ROLE
 - EXECUTE
 - GRANT OPTION
 - INDEX
 - INSERT
 - REFERENCES
 - SELECT
 - SHOW DATABASES
 - SHOW VIEW
 - TRIGGER
 - UPDATE
6. Crea cuatro usuarios con los nombres de cuatro compañeros.

Sobre una base de datos que crees nueva (sin tablas), se necesitan los siguientes

roles:

- Rol de administración que tenga todos los privilegios en esa BdD. `rol_admin`.
- Rol que permita insertar y borrar y modificar datos en las tablas de dicha base de datos. `rol_modif`.
- Rol que pueda insertar y modificar pero no borrar. `rol_anadir`.
- Rol que permita únicamente seleccionar datos. `rol_select`.

Asigna los roles a los usuarios de la siguiente forma:

- Primer usuario: `rol_admin`.
- Segundo usuario: `rol_modif` y `rol_select`.
- Tercer usuario: `rol_anadir`.
- Cuarto usuario: `rol_select`.

Se pide que diseñes unas operaciones SQL que permitan comprobar que cada usuario puede ejecutar las instrucciones SQL que indican sus permisos y **que no** puede hacer aquellas que para las que no tiene privilegios (con un par de las que no, por usuario, es suficiente).

¿El tercer usuario puede leer (SELECT)?

7. Crea una vista en la BdD de Conciertos que contenga el nombre de las obras y el nombre y apellido del compositor de cada una.

Crea un rol que permita seleccionar (consultar) los datos sobre esa vista y dos usuarios que debes asignar al mismo. Comprueba lo que acabas de crear.

Crea otro rol que permita insertar y modificar sobre la vista y asígnaselo a uno de los usuarios. ¿Funciona? ¿por qué?

8. Realiza un script para eliminar los usuarios y roles siguiendo todos los pasos: quitar permisos, desasignar, borrar, etc.

Independientemente de que te deje o no, piensa el orden en que se deberían realizar las operaciones. Justifica tu respuesta.

13.4. Bibliografía.

- [MySQL 8.0 Reference Manual](#).
- [MySQL Administration](#). MySQLTutorial.
- [Structured Query Language](#). Wikibook. Wikipedia.
- Iván López Montalbán, Manuel de Castro Vázquez, John Ospino Rivas (2022). Bases de Datos (2ª Edición) . Garceta.

- [Is there a way to know your current username in mysql?](#). StackOverflow.
- [How to List All Users in a MySQL Database](#). PhoenixNAP.
- [Using % for host when creating a MySQL user](#). StackOverflow.
- [How To Create a New User and Grant Permissions in MySQL](#). Etel Sverdlov (DigitalOcean).