



Clase 19. Curso SQL

SUBLENGUAJE DCL

***RECUERDA PONER A GRABAR LA
CLASE***





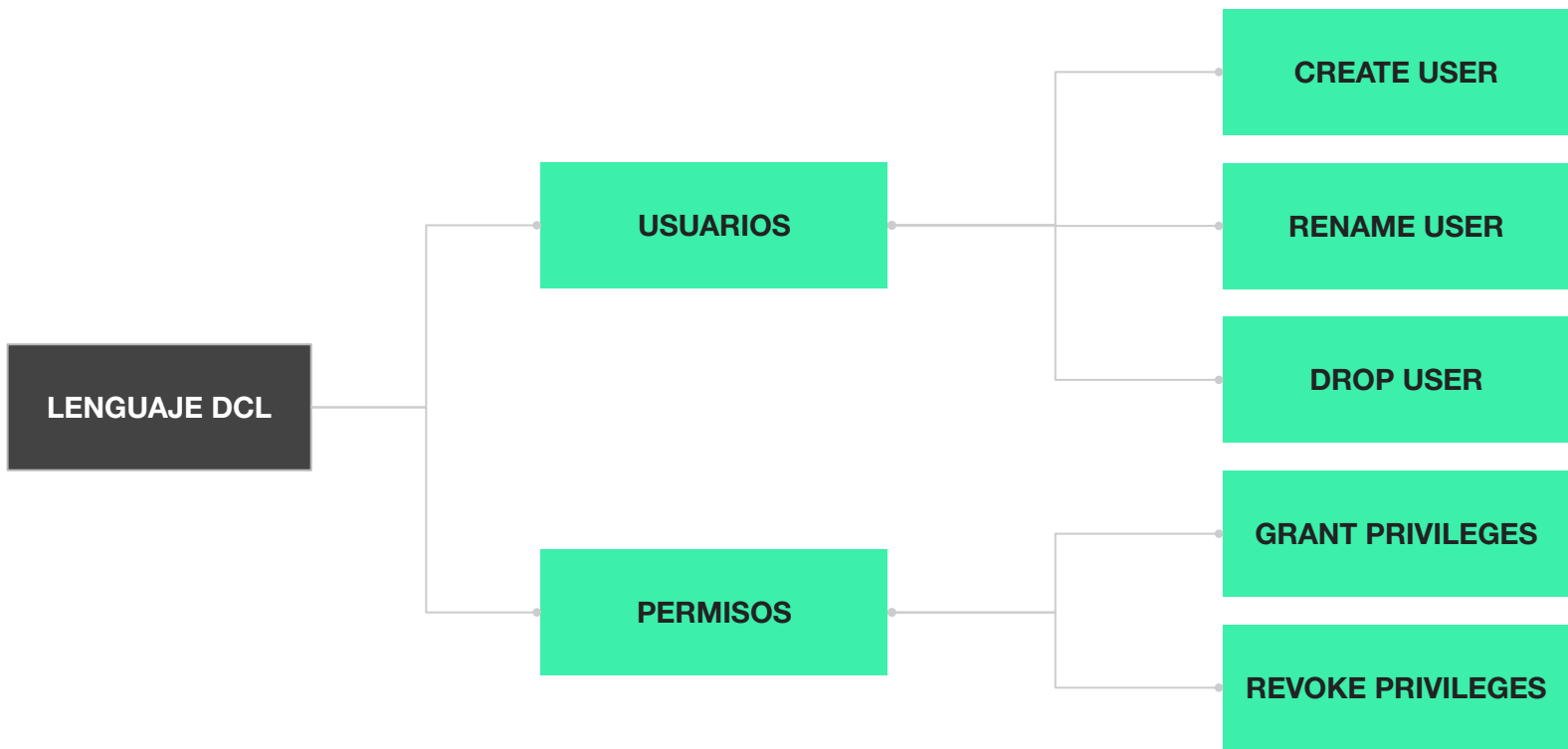
OBJETIVOS DE LA CLASE

- Reconocer e implementar las sentencias del sublenguaje DCL
- Identificar en qué situación usar cada sentencia.

MAPA DE CONCEPTOS

MAPA DE CONCEPTOS CLASE 19

¡Para
recordar!



EL LENGUAJE DCL

CONCEPTO GENERAL

DATA CONTROL LANGUAGE

El **Lenguaje de Control de Datos** (DCL) permite definir diferentes usuarios dentro del motor de base de datos Mysql, y establecer para cada uno de ellos, permisos totales, parciales, o negar el acceso sobre los diferentes Objetos que conforman la Base de Datos.

DATA CONTROL LANGUAGE

Al igual que con **DDL** y **DML**, **DCL** provee una serie de cláusulas y comandos para poder crear, renombrar y eliminar usuarios dentro de uno o más servidores de base de datos, como así también establecer y/o modificar un password o contraseña de acceso.



DATA CONTROL LANGUAGE

Definidos los usuarios a través de **DCL**, contamos con otro set de comandos y cláusulas para **permitir o revocar el acceso a diferentes Objetos de la base de datos.**

Entendemos por Objetos a: Tablas, Campos, Vistas, Stored Procedures y Funciones Almacenadas.



BASE DE DATOS DEL SISTEMA

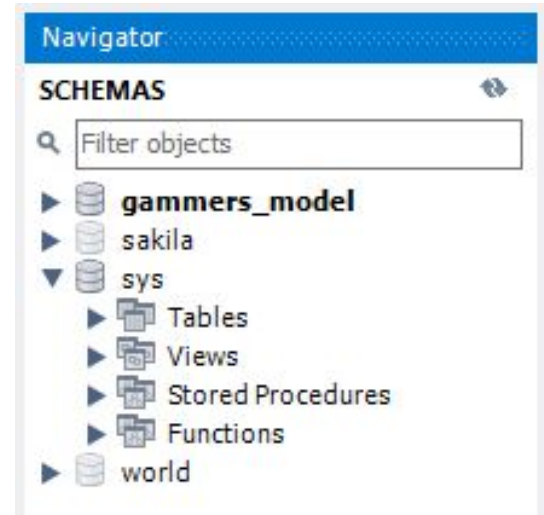
DB DE SISTEMA

Mysql cuenta con lo que se conoce como **Base de Datos del Sistema**, donde se almacena información referente al motor de base de datos en sí, performance y demás cuestiones propias de un sistema de software.

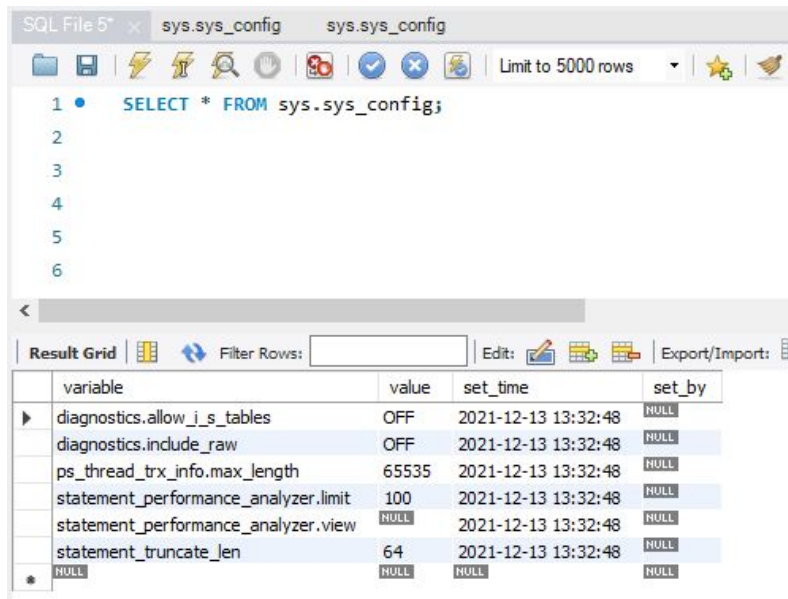
DB DE SISTEMA

Cuando instalamos por primera vez a Mysql vemos que, por defecto, se crea la **base de datos SYS** junto al motor de Mysql.

Despleguemos la misma para ver su contenido.



DB DE SISTEMA



The screenshot shows a SQL IDE window with a query editor and a result grid. The query editor contains the SQL statement: `SELECT * FROM sys.sys_config;`. The result grid displays the contents of the `sys.sys_config` table, which has four columns: `variable`, `value`, `set_time`, and `set_by`. The table contains several rows of configuration data, including `diagnostics.allow_i_s_tables`, `diagnostics.include_raw`, `ps_thread_trx_info.max_length`, `statement_performance_analyzer.limit`, `statement_performance_analyzer.view`, and `statement_truncate_len`. The last row shows `NULL` values for all columns.

variable	value	set_time	set_by
diagnostics.allow_i_s_tables	OFF	2021-12-13 13:32:48	NULL
diagnostics.include_raw	OFF	2021-12-13 13:32:48	NULL
ps_thread_trx_info.max_length	65535	2021-12-13 13:32:48	NULL
statement_performance_analyzer.limit	100	2021-12-13 13:32:48	NULL
statement_performance_analyzer.view	NULL	2021-12-13 13:32:48	NULL
statement_truncate_len	64	2021-12-13 13:32:48	NULL
NULL	NULL	NULL	NULL

Veremos que apenas tiene una tabla visible, llamada **sys_config**, con información sobre performance de carga. Si revisamos las Vistas y los Stored Procedures, encontraremos mucho más datos que aportan a la misma causa.

DB DE SISTEMA

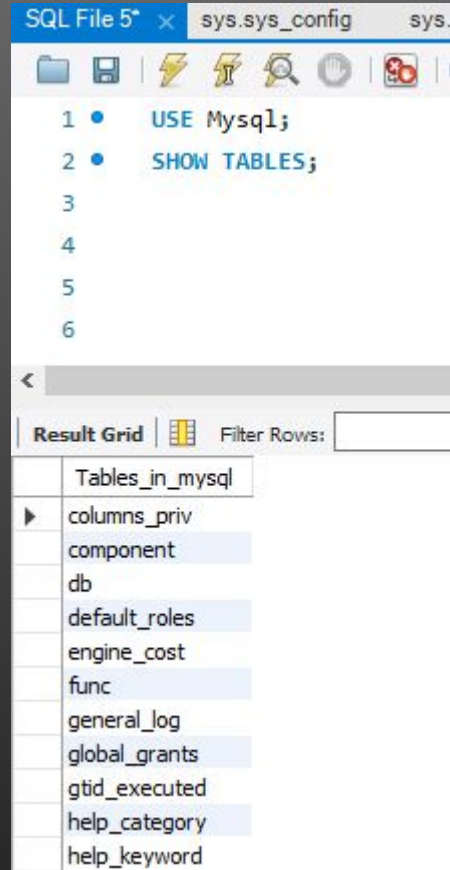
Además de **sys**, Mysql tiene otra base de datos de sistema llamada **mysql**. En la misma encontrarás otro set de tablas con información general para el sistema Mysql, más una tabla llamada **user**, donde se almacena la información de usuarios de la o las bases de datos de Mysql.

DB DE SISTEMA

Puedes verificar esto mismo, directamente desde **Mysql Workbench**, abriendo una pestaña de script y escribiendo los comandos:

```
USE mysql;
```

```
SHOW tables;
```

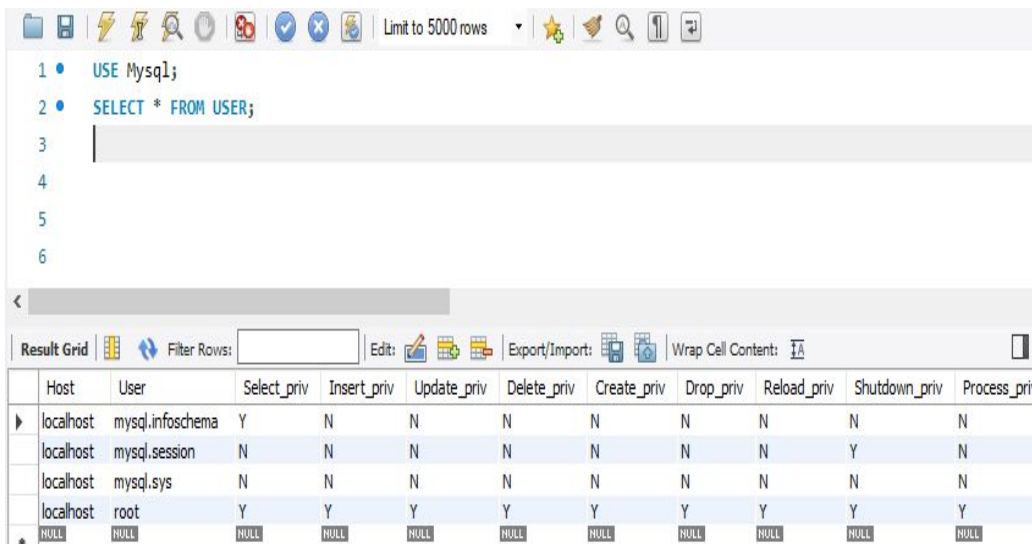


LA TABLA mysql.user

LA TABLA `mysql.user`

En **la tabla user**, Mysql almacena la información de usuarios junto a cada uno de los permisos o bloqueo para trabajar sobre la base de datos Mysql y los objetos de ésta.

LA TABLA *mysql.user*



The screenshot shows a MySQL client window with a toolbar at the top. The SQL editor contains the following queries:

```
1 • USE MySQL;  
2 • SELECT * FROM USER;  
3  
4  
5  
6
```

Below the editor, the 'Result Grid' tab is active, displaying the results of the query. The table has 11 columns: Host, User, Select_priv, Insert_priv, Update_priv, Delete_priv, Create_priv, Drop_priv, Reload_priv, Shutdown_priv, and Process_priv. The data is as follows:

Host	User	Select_priv	Insert_priv	Update_priv	Delete_priv	Create_priv	Drop_priv	Reload_priv	Shutdown_priv	Process_priv
localhost	mysql.infoschema	Y	N	N	N	N	N	N	N	N
localhost	mysql.session	N	N	N	N	N	N	N	Y	N
localhost	mysql.sys	N	N	N	N	N	N	N	N	N
localhost	root	Y	Y	Y	Y	Y	Y	Y	Y	Y
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Si invocamos la misma mediante la instrucción **SELECT**, veremos información sobre el host o máquina, el usuario en cuestión, y la lista de permisos totales que podemos aplicar sobre cada usuario.

LA TABLA `mysql.user`;

En esta imagen podemos ver los **Campos** que definen los permisos de usuario sobre **DML**

Select_priv	Insert_priv	Update_priv	Delete_priv
Y	N	N	N
N	N	N	N
N	N	N	N
Y	Y	Y	Y
NULL	NULL	NULL	NULL

*Cada permiso se define con un **Y** o **N**, si queremos brindarle al usuario Acceso o Restricción (respectivamente), sobre una acción determinada*

LA TABLA `mysql.user`;

Además de **DML**, también encontramos las tablas que permiten o impiden al usuario realizar sentencias del **DDL**.

Create_priv	Drop_priv
N	N
N	N
N	N
Y	Y
NULL	NULL

Y si recorremos el resto de la tabla, veremos en detalle que podemos tener un control específico sobre la ejecución, inserción, visualización y eliminación de objetos dentro de la o las bases de datos Mysql.

GESTIONAR USUARIOS

GESTIONAR USUARIOS



Veamos a continuación, cómo gestionar nuestros propios usuarios dentro de Mysql a través de los diferentes comandos que este motor de base de datos pone a nuestra disposición.

CREATE USER

CODER HOUSE

CREATE USER

A través de la sentencia **CREATE USER 'nombre'**, creamos un nuevo usuario en la base de datos. Podemos definir solo su nombre, o el nombre de éste seguido del dominio al cual pertenece.



```
CREATE USER 'prueba@dominio';
```

CREATE USER + DOMINIO

El dominio en sí hace referencia a la máquina o computadora donde se encuentra instalado Mysql. Podemos referirnos a ésta mediante **su nombre, IP**, o si es local, utilizando **localhost**.

```
CREATE USER 'prueba@dbProdServer'; //su nombre
```

```
CREATE USER 'prueba@192.168.0.213; //su dirección IP
```

```
CREATE USER 'prueba@localhost'; //local
```

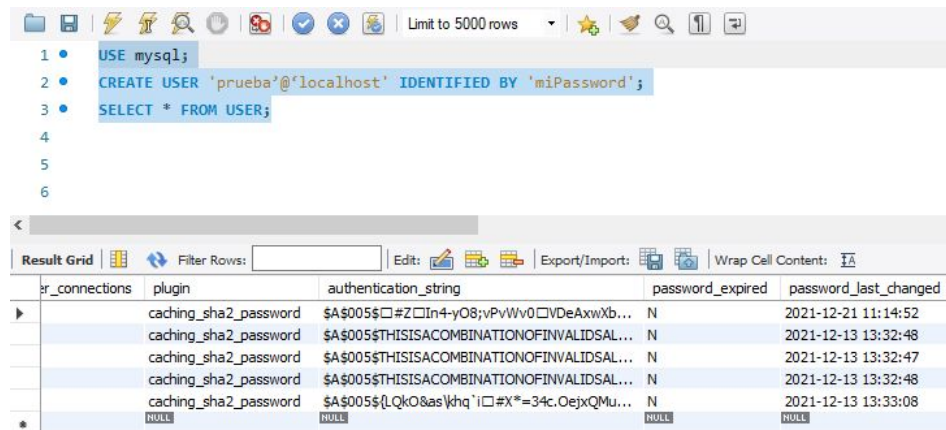
CREATE USER + DOMINIO + PASSWORD

Si también deseamos especificar un password para el usuario en cuestión, podemos hacerlo incorporando a la sentencia, el comando

IDENTIFIED BY. El password será visible en el comando en cuestión, pero se encriptará una vez almacenado en la tabla **user**.

```
CREATE USER 'prueba@localhost' IDENTIFIED BY 'miPassword';
```

CREATE USER



The screenshot shows a MySQL command-line interface. The top toolbar includes icons for file operations, execution, and search. Below the toolbar, a list of SQL commands is shown with line numbers 1 through 6. The commands are: 1. `USE mysql;`, 2. `CREATE USER 'prueba'@'localhost' IDENTIFIED BY 'miPassword';`, 3. `SELECT * FROM USER;`, 4. (empty line), 5. (empty line), and 6. (empty line). Below the commands, the 'Result Grid' is displayed, showing the output of the `SELECT * FROM USER;` command. The grid has five columns: `user_connections`, `plugin`, `authentication_string`, `password_expired`, and `password_last_changed`. The first five rows show the 'mysql' database user with the 'caching_sha2_password' plugin and a long, hashed authentication string. The last row shows a 'NULL' entry for all columns.

```
1 • USE mysql;
2 • CREATE USER 'prueba'@'localhost' IDENTIFIED BY 'miPassword';
3 • SELECT * FROM USER;
4
5
6
```

user_connections	plugin	authentication_string	password_expired	password_last_changed
1	caching_sha2_password	\$A\$005\$□#Z□In4-yO8;vPvWv0□VDeAxwXb...	N	2021-12-21 11:14:52
1	caching_sha2_password	\$A\$005\$THISISACOMBINATIONOFINVALIDSAL...	N	2021-12-13 13:32:48
1	caching_sha2_password	\$A\$005\$THISISACOMBINATIONOFINVALIDSAL...	N	2021-12-13 13:32:47
1	caching_sha2_password	\$A\$005\$THISISACOMBINATIONOFINVALIDSAL...	N	2021-12-13 13:32:48
1	caching_sha2_password	\$A\$005\${LQkO8as\khq`i□#X*=34c.OejxQM...	N	2021-12-13 13:33:08
*	NULL	NULL	NULL	NULL

La columna **authentication_string** permite validar que, el password ingresado, se almacena de manera encriptada.



BREAK

¡5/10 MINUTOS Y VOLVEMOS!

MODIFICAR UN PASSWORD

MODIFICAR UN PASSWORD

Podemos cambiar el password de un usuario, mediante el comando **ALTER USER**:

```
ALTER USER 'prueba@dominio' IDENTIFIED BY 'nuevoPassword';
```

.

También podemos hacerlo mediante la sentencia **UPDATE**:

```
UPDATE mysql.user SET Password=PASSWORD('nuevoPassword') WHERE user =  
    'prueba' AND host = 'dominio';
```

MODIFICAR UN PASSWORD

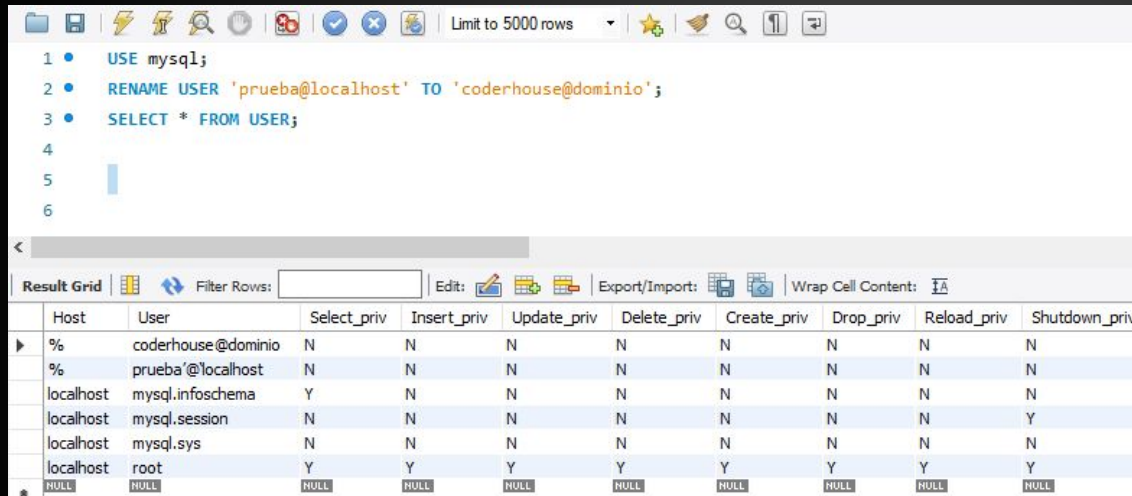
Si en algún momento debemos trabajar con una **base de datos Mysql v5.7.5 o inferior**, debemos utilizar el comando **SET PASSWORD**.

```
SET PASSWORD FOR 'prueba@dominio' = PASSWORD('nuevoPassword');  
o para el usuario logeado  
SET PASSWORD = PASSWORD('nuevoPassword')
```


RENOMBRAR Y ELIMINAR USUARIOS

RENOMBRAR UN USUARIO

También podemos renombrar un usuario, una vez creado, utilizando el comando **RENAME USER**.



```
1 • USE mysql;
2 • RENAME USER 'prueba@localhost' TO 'coderhouse@dominio';
3 • SELECT * FROM USER;
4
5
6
```

Result Grid

	Host	User	Select_priv	Insert_priv	Update_priv	Delete_priv	Create_priv	Drop_priv	Reload_priv	Shutdown_priv
▶	%	coderhouse@dominio	N	N	N	N	N	N	N	N
	%	prueba@localhost	N	N	N	N	N	N	N	N
	localhost	mysql.infoschema	Y	N	N	N	N	N	N	N
	localhost	mysql.session	N	N	N	N	N	N	N	Y
	localhost	mysql.sys	N	N	N	N	N	N	N	N
	localhost	root	Y	Y	Y	Y	Y	Y	Y	Y
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

RENAME USER

'prueba@localhost' TO
coderhouse@dominio;

CODER HOUSE

ELIMINAR UN USUARIO

Finalmente, si debemos eliminar un usuario, recurrimos al comando **DROP USER**:

```
DROP USER 'usuario@dominio';
```



EJEMPLO EN VIVO

Implementaremos las sentencias create, drop y rename user.

PERMISOS DEFINIDOS POR DEFECTO

Por cada usuario que creamos, solo definimos un nombre de usuario y sobre qué SERVIDOR trabajará el mismo.

Nos queda por delante comenzar a definir los permisos que cada usuario tendrá, sobre un dominio, base de datos, tabla(s) y/o campo(s) específicos.

¡Veamos entonces cómo hacerlo!

VERIFICAR EL USUARIO CREADO



VERIFICAR EL USUARIO CREADO



Si queremos verificar los permisos de un usuario específico, podemos realizar una consulta de selección filtrando específicamente por el usuario en cuestión.



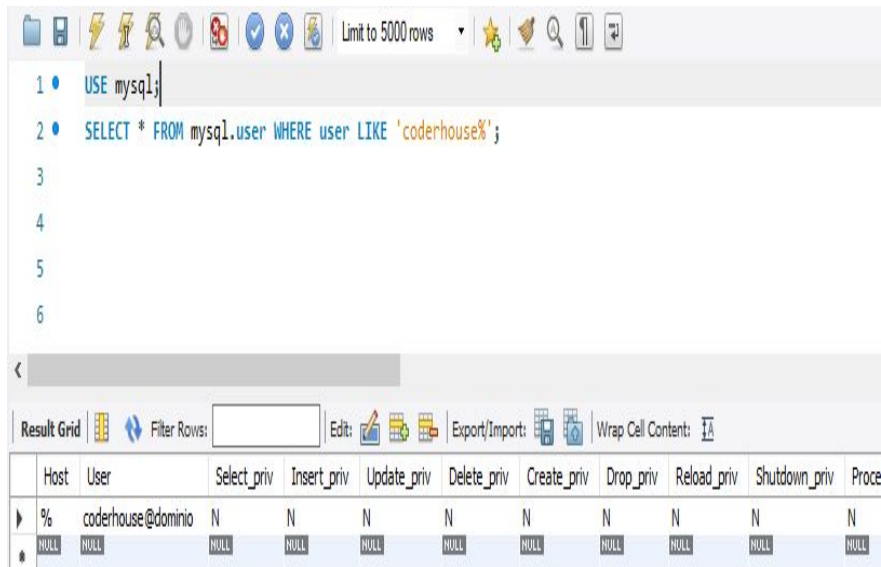
VERIFICAR EL USUARIO CREADO

Ejecutemos para ello, la sentencia **SELECT** sobre la tabla **user**, filtrando por el usuario en particular:

```
SELECT * FROM mysql.user WHERE user LIKE 'coderhouse%';
```




VERIFICAR EL USUARIO CREADO



The screenshot shows a MySQL client window with a toolbar at the top. The query editor contains two lines of SQL: `USE mysql;` and `SELECT * FROM mysql.user WHERE user LIKE 'coderhouse%';`. Below the editor is a horizontal scrollbar. At the bottom, the 'Result Grid' tab is active, displaying a table with 11 columns: Host, User, Select_priv, Insert_priv, Update_priv, Delete_priv, Create_priv, Drop_priv, Reload_priv, Shutdown_priv, and Process_priv. The table contains two rows: the first row shows a user with host '%' and username 'coderhouse@dominio', and the second row shows a user with host 'NULL' and username 'NULL'. All privilege columns for both users are set to 'N' or 'NULL'.

Host	User	Select_priv	Insert_priv	Update_priv	Delete_priv	Create_priv	Drop_priv	Reload_priv	Shutdown_priv	Process_priv
%	coderhouse@dominio	N	N	N	N	N	N	N	N	N
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

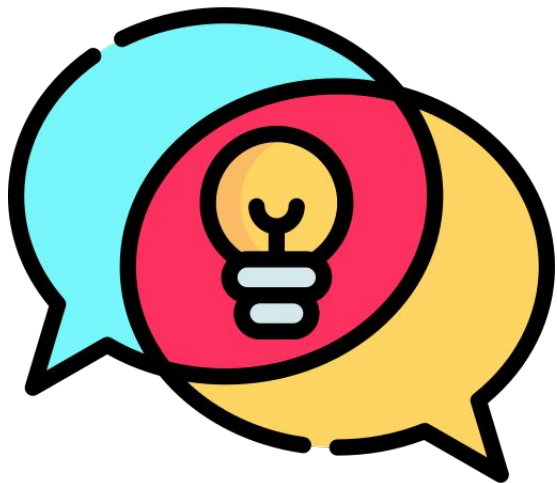
Como podemos apreciar, en cada campo de la tabla, el parámetro correspondiente al permiso está seteado en **N**, lo cual indica que no tiene establecido ningún permiso.

ESTABLECER PERMISOS SOBRE OBJETOS MYSQL

SENTENCIA GRANT

CODER HOUSE

SENTENCIA GRANT



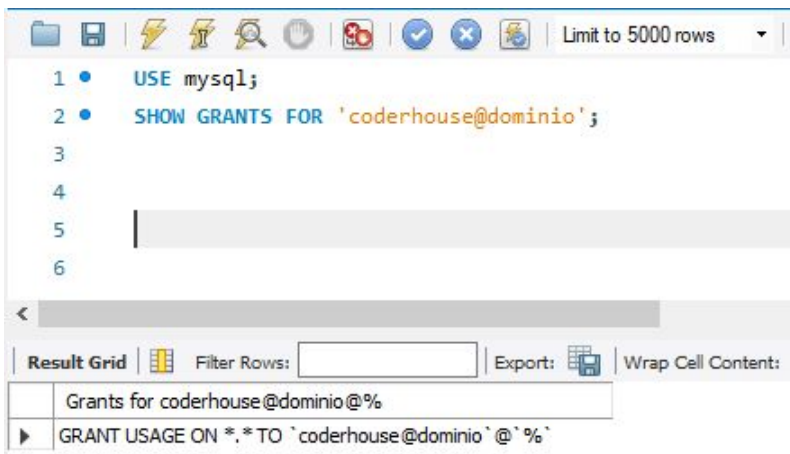
A través de sentencia **GRANT**, podemos definir en detalle, los permisos de escritura, modificación, eliminación, y/o lectura de datos y creación, alteración y borrado de objetos de la base de datos, para un usuario

SENTENCIA GRANT

SHOW GRANTS nos permite detallar sobre el usuario específico, cuáles permisos tiene asociado sobre los diferentes objetos de la DB.

```
SHOW GRANTS FOR 'coderhouse@dominio';
```

SENTENCIA GRANT



The screenshot shows a MySQL client window with a toolbar at the top. The command history on the left lists two commands: `USE mysql;` and `SHOW GRANTS FOR 'coderhouse@dominio';`. The main text area shows the execution of the second command. Below the text area, the 'Result Grid' tab is active, displaying the output of the `SHOW GRANTS` command. The output is a table with two rows: the first row is the header 'Grants for coderhouse@dominio@%', and the second row contains the grant statement `GRANT USAGE ON *,* TO 'coderhouse@dominio'@'%'`.

```
1 • USE mysql;
2 • SHOW GRANTS FOR 'coderhouse@dominio';
3
4
5 |
6
```

Grants for coderhouse@dominio@%
GRANT USAGE ON *,* TO 'coderhouse@dominio'@'%'

Su resultado no detalla **ningún permiso** sobre sentencias DML, aún para el usuario seleccionado.

Veamos entonces cómo comenzar a otorgarlos.

OTORGARLE TODOS LOS PERMISOS

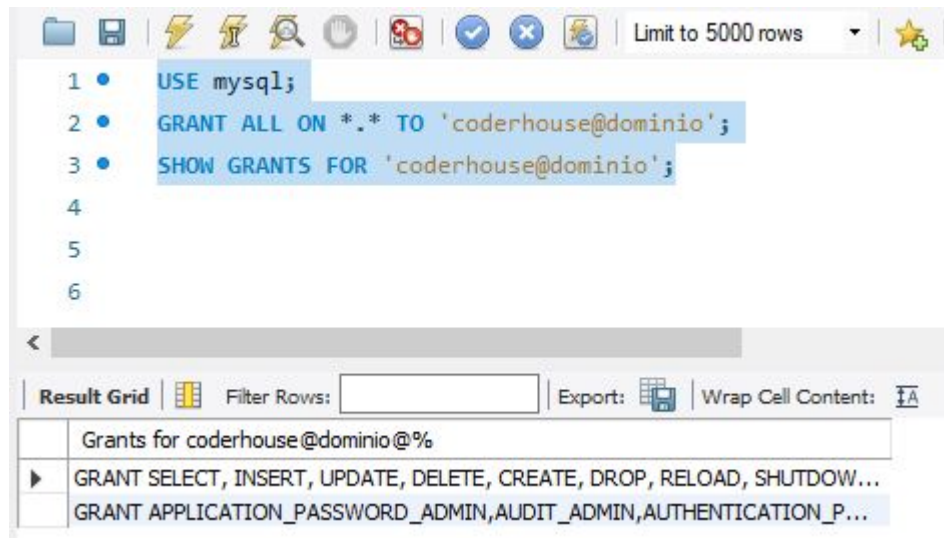
OTORGARLE TODOS LOS PERMISOS

Para otorgarle permisos completos a un usuario, sobre todos los objetos de todos los dominios, utilizamos la sentencia **GRANT ALL**.

```
GRANT ALL ON *.* TO 'coderhouse@dominio';
```

El uso de ***.*** refiere a **objetos.dominio**.

OTORGARLE TODOS LOS PERMISOS



The screenshot shows a MySQL command-line interface with a toolbar at the top. The toolbar includes icons for file operations (folder, save, lightning bolt, copy, search, undo, redo), a status bar (Limit to 5000 rows), and a star icon. The command prompt shows three lines of SQL code:

```
1 • USE mysql;  
2 • GRANT ALL ON *.* TO 'coderhouse@dominio';  
3 • SHOW GRANTS FOR 'coderhouse@dominio';  
4  
5  
6
```

Below the command prompt, the output is displayed in a table with the following content:

	Grants for coderhouse@dominio@%
▶	GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, RELOAD, SHUTDOWN...
	GRANT APPLICATION_PASSWORD_ADMIN,AUDIT_ADMIN,AUTHENTICATION_P...

A través de la pestaña **Action Output** veremos la correcta aplicación de los permisos sobre el usuario referenciado.

OTORGAR PERMISOS SOBRE TABLA(S)

OTORGAR PERMISOS SOBRE UNA TABLA

Para otorgarle permisos a un usuario sobre una tabla específica de una base de datos puntual, debemos referenciar la sentencia de la siguiente forma:

```
GRANT ALL ON gamer.level_game TO 'coderhouse@localhost';
```

PERMISOS EN MÁS DE UNA TABLA

Deberemos definir una línea específica por cada tabla en la que un usuario específico tendrá permisos:

```
GRANT ALL ON gamer.class TO 'coderhouse@localhost';  
GRANT ALL ON gamer.game TO 'coderhouse@localhost';
```

OTORGAR PERMISOS SELECTIVOS

OTORGAR PERMISOS SELECTIVOS

Si por ejemplo deseamos que un usuario tenga permisos selectivos referidos al DML sobre una tabla, debemos estructurar la cláusula de la siguiente forma:

```
GRANT SELECT, UPDATE ON gamer.level_game TO 'coderhouse@localhost';
```

OTORGAR PERMISOS SELECTIVOS

De esta manera, definimos que un usuario pueda realizar determinadas acciones sobre una tabla.

El usuario puede, por ejemplo sólo leer registros, agregar, o realizar todo tipo de operaciones menos eliminar.

Tengamos presente para esto, pensar bien la relación operativa entre las diferentes acciones, previo a establecer las mismas.

OTORGAR PERMISOS SOBRE COLUMNAS

OTORGAR PERMISOS SOBRE COLUMNAS

También podemos ir con más profundidad, estableciendo ciertos permisos sobre determinadas columnas de una tabla.

Por ejemplo así podemos permitir que un usuario modifique solamente ciertos campos y evitar cambios sobre otros claves, como ser un documento de identidad o importes monetarios, entre otros

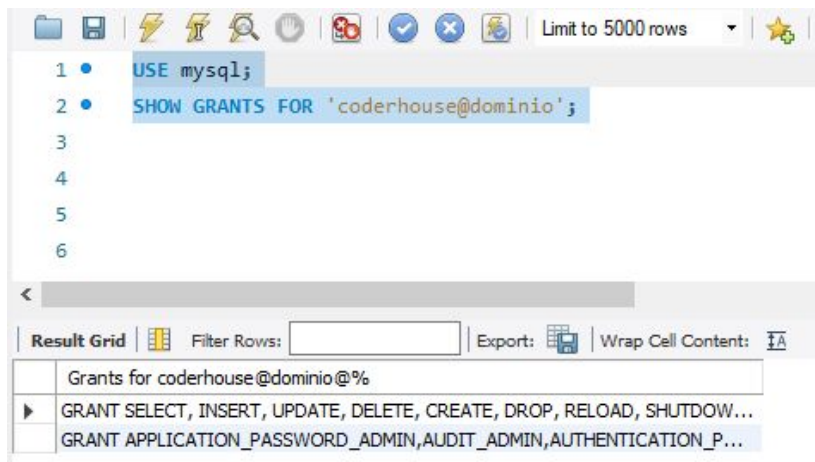
OTORGAR PERMISOS SOBRE COLUMNAS

Para esto, debemos definir cada campo específico separado por una coma, tal como hacemos una consulta del tipo SELECT, definiendo qué campos visualizar.

```
GRANT UPDATE, SELECT (description)  
ON gamer.level_game  
TO 'coderhouse@localhost';
```

***VERIFICAR PERMISOS
ESTABLECIDOS***

PERMISOS ESTABLECIDOS



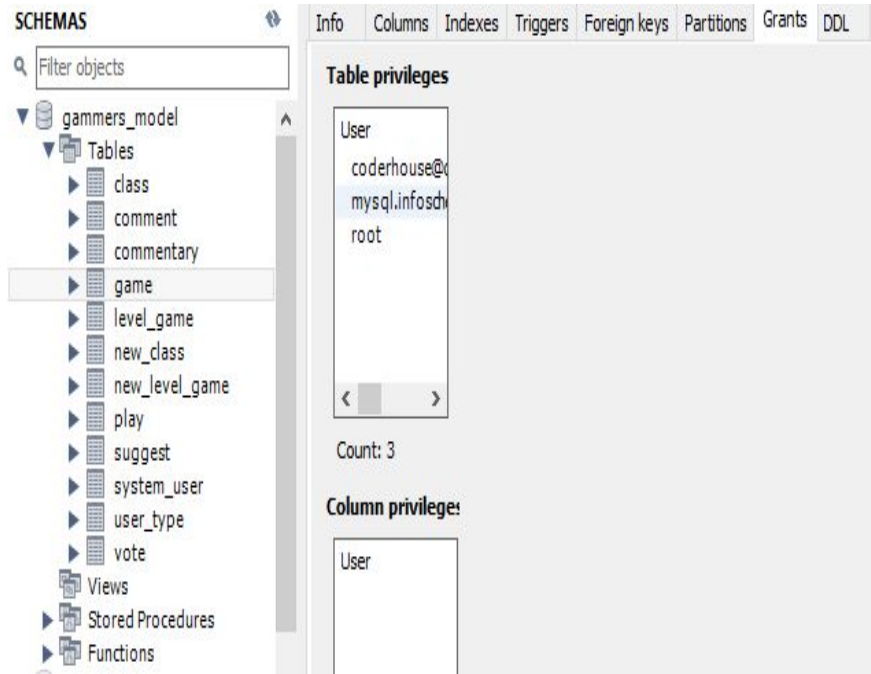
```
1 • USE mysql;
2 • SHOW GRANTS FOR 'coderhouse@dominio';
3
4
5
6
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

Grants for coderhouse@dominio@%	
▶	GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, RELOAD, SHUTDOWN...
	GRANT APPLICATION_PASSWORD_ADMIN,AUDIT_ADMIN,AUTHENTICATION_P...

Ejecutando nuevamente el comando **SHOW GRANTS** para el usuario en cuestión, podremos ver los diferentes permisos que le han sido otorgados al mismo.

PERMISOS ESTABLECIDOS



También, accediendo a las propiedades de la tabla mediante **TABLE INSPECTOR**, podremos ver un detalle de los permisos por usuario establecidos.

SENTENCIA REVOKE: QUITAR PERMISOS

SENTENCIA REVOKE

Tal como existe una sentencia para otorgar diferentes permisos a un usuario de base de datos, también existe la sentencia que le quita dichos permisos. Se llama **REVOKE**. y funciona de igual forma a **GRANT**, pero a la inversa.

Veamos unos ejemplos a continuación...

QUITAR TODOS LOS PERMISOS

Para quitarle todos los permisos a un usuario, sobre todos los objetos de todos los dominios, utilizamos la sentencia

REVOKE ALL.

```
REVOKE ALL ON *.* FROM 'coderhouse@localhost';
```

El uso de ***.*** refiere a **objetos.dominio**.

QUITAR UN PERMISO DETERMINADO

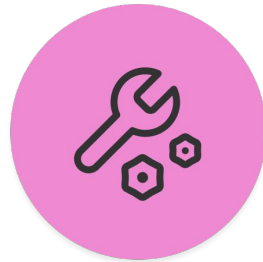
Para quitarle un permiso específico a un usuario, por ejemplo actualizar registros, sobre todos los objetos de todos los dominios, realizamos lo siguiente.

```
REVOKE UPDATE ON *.* FROM 'coderhouse@localhost';
```



EJEMPLO EN VIVO

Implementaremos las sentencias GRANT y REVOKE.



CREAR USUARIO Y PERMISOS SOBRE UNA DB

Practicaremos la implementación del sublenguaje DCL

Tiempo estimado: 15 minutos

CREAR USUARIO Y PERMISOS SOBRE UNA BB.DD.

Desafío
generico



Utilizaremos la base de datos GAMERS para crear un nuevo usuario y le estableceremos determinados permisos.

- Crea un usuario llamado *coderhouse* y asígnale lo mismo como contraseña.
- Establece permisos de solo lectura de datos sobre la tabla **GAME**.
- Establece permisos de lectura e inserción sobre la tabla **CLASS**.
- Abre una nueva ventana de conexión e inicia sesión con este usuario.
- Modifica un registro cualquiera de la tabla **GAME** y aplica los cambios.
- Agrega un registro en la tabla **CLASS**.
- Elimina este último registro agregado.



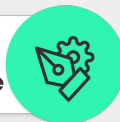
IMPLEMENTACIÓN DE SENTENCIAS

Implementaremos cada una de las sentencias que componen el sublenguaje DCL.

IMPLEMENTACIÓN DE SENTENCIAS

Formato: El archivo a presentar debe ser del tipo .sql nombrado como “Sentencias+Apellido”.

Desafío
entregable



>> Consigna: Se deberán crear dos usuarios nuevos a los cuales se le asignará una serie de permisos, a saber:

>>Aspectos a incluir en el entregable:

Uno de los usuarios creados deberá tener permisos de sólo lectura sobre todas las tablas.

El otro usuario deberá tener permisos de Lectura, Inserción y Modificación de datos.

Ninguno de ellos podrá eliminar registros de ninguna tabla.

Cada sentencia **GRANT** y **CREATE USER** deberá estar comentada con lo que realiza la misma.

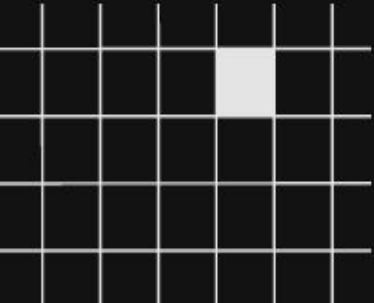
¿PREGUNTAS?





¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

- Creación de Usuarios.
 - Asignación de Permisos totales y parciales.
 - Eliminación de Permisos totales y parciales.
- 



OPINA Y VALORA ESTA CLASE

#DEMOCRATIZANDO LA EDUCACIÓN

CODER HOUSE