

Introducción

En este laboratorio, pondremos en práctica los mecanismos de control de acceso discrecional que hemos visto en clase. En concreto, crearemos una base de datos y configuraremos nuestro servidor para que pueda ser utilizado desde una aplicación cliente implementada en *Java*. Para que la situación sea más realista, en cada grupo de 2 personas cada uno hará de administrador de bases de datos de su propio servidor virtual y de cliente del servidor de bases de datos de su compañera o compañero. De esta forma, veremos la dificultad de configurar adecuadamente nuestro servidor para que los usuarios puedan utilizar todos los recursos que necesiten y de una forma adecuada pero sin que tengan opción de realizar ninguna acción más allá de las estrictamente necesarias.

Objetivos

Los objetivos de este laboratorio son:

- ☐ Crear usuarios de *MySQL* y asignarles los privilegios necesarios.
- ☐ Crear una primera base de datos de ejemplo.
- ☐ Comprobar si la configuración de privilegios es adecuada para realizar las tareas previstas.
- ☐ Utilizar un cliente *Java* para acceder a una base de datos.

Requisitos previos

Para realizar este laboratorio utilizaremos el servidor de bases de datos que obtuvimos como resultado del Laboratorio 1. El servidor tiene que estar en funcionamiento y correctamente configurado.

Para utilizar la aplicación cliente será necesario tener instalado un entorno *Java*, bien un entorno de ejecución (JRE) o un entorno de desarrollo (JDK).

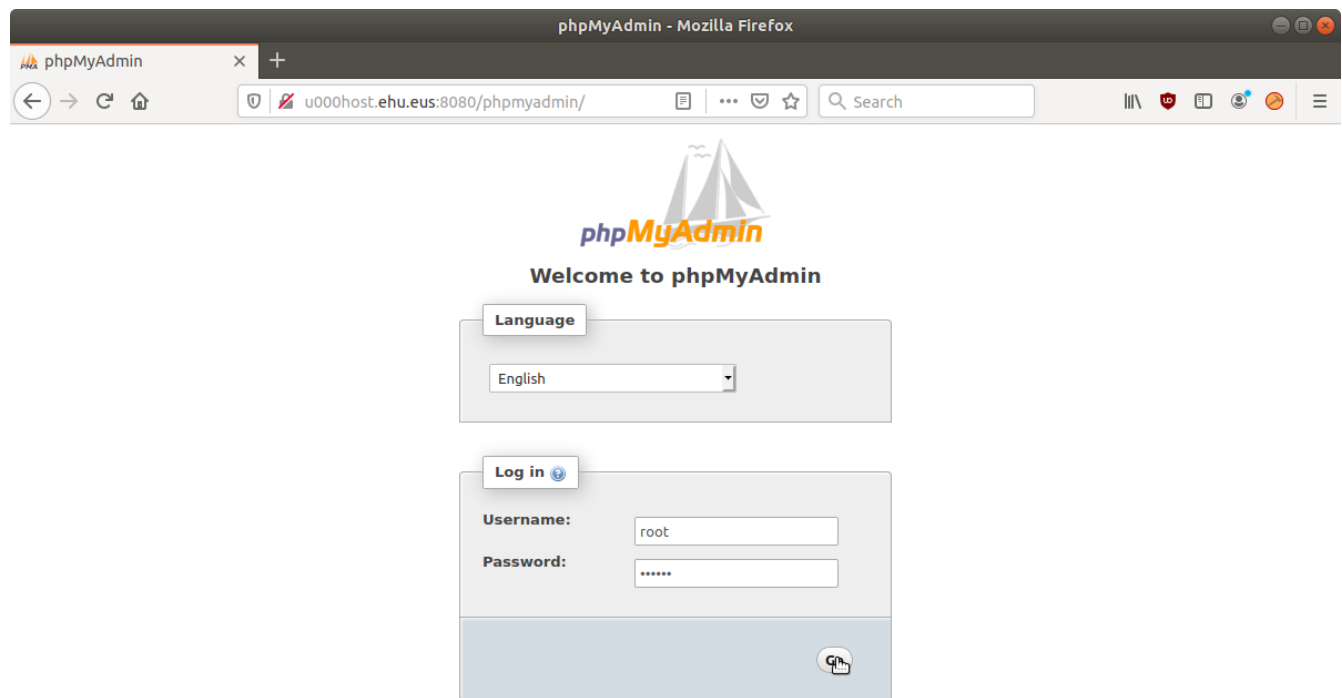
Procedimiento a seguir

A continuación, se describen los pasos a realizar suponiendo que el servidor de bases de datos está en marcha y correctamente configurado.

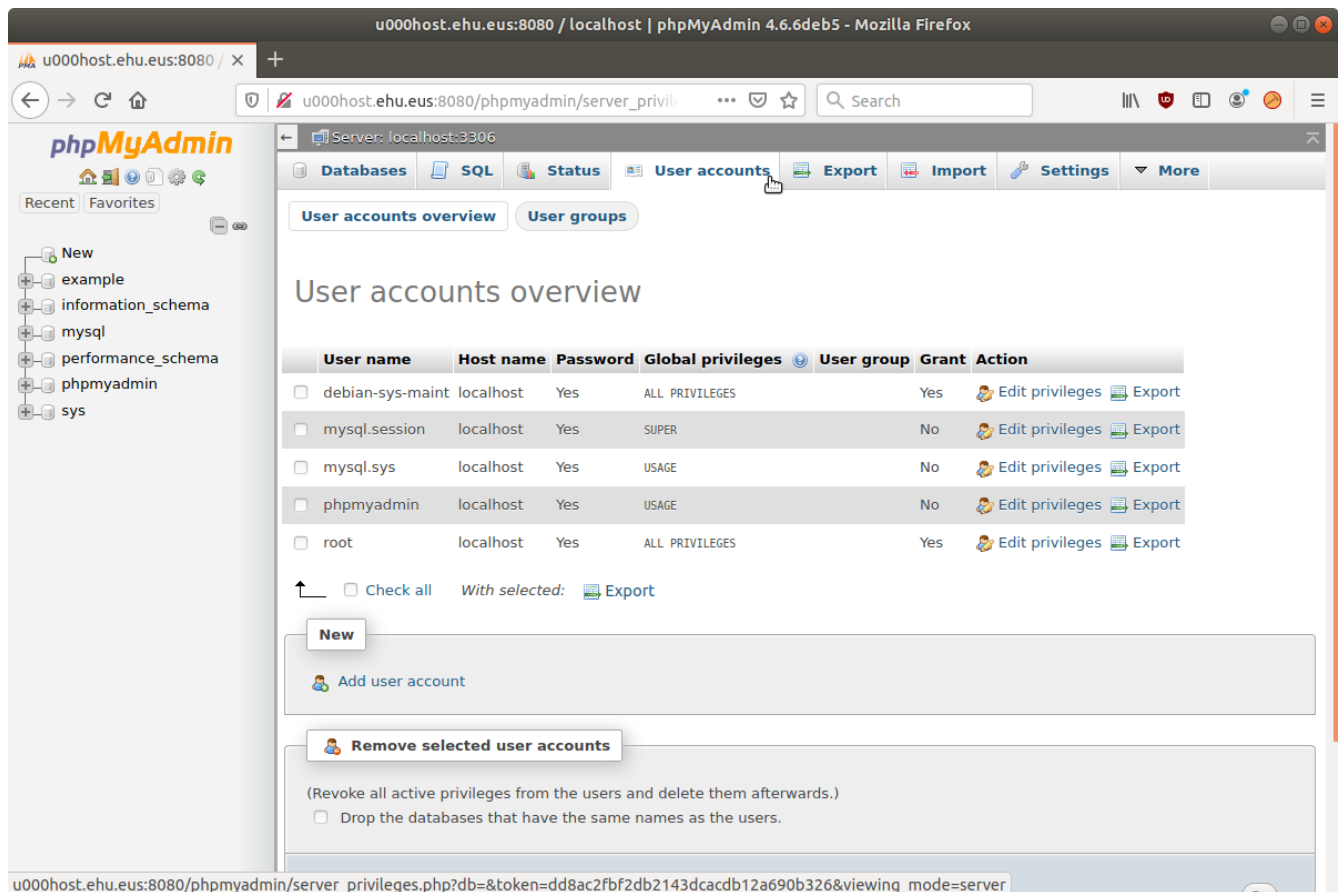
1 Creación de usuarios en MySQL

Para esto, utilizaremos el usuario “*root*” de *MySQL*, cuyo password especificamos en el Laboratorio 1.

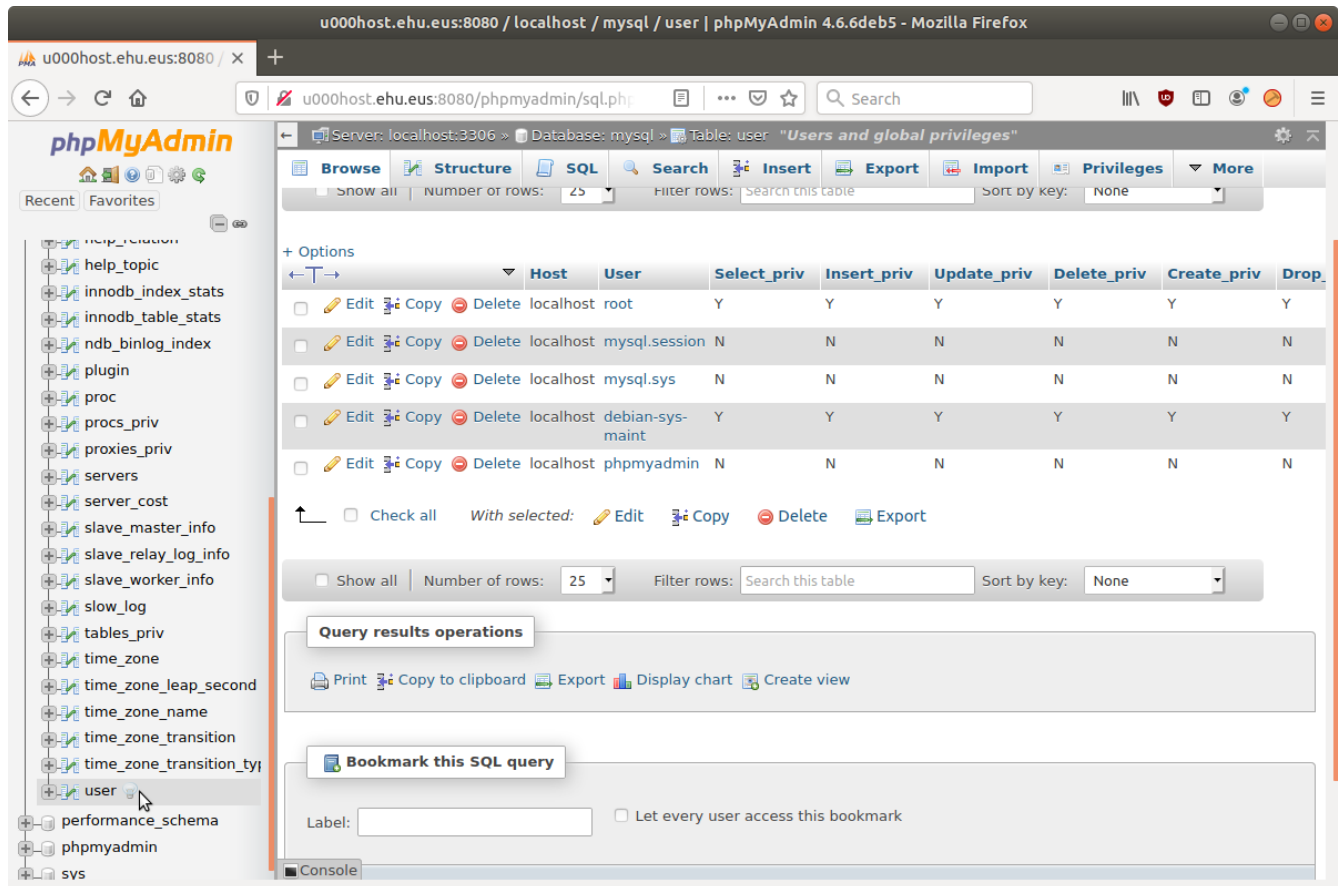
1. A través de *phpMyAdmin*, conéctate a tu servidor de bases de datos *MySQL* usando la dirección IP de la máquina virtual. Para iniciar sesión en *MySQL*, tan solo puedes utilizar el usuario “*root*”.



- En la ventana principal, haz click en la pestaña “*User accounts*”. Esta pestaña da acceso a la información referente a usuarios y permisos.



3. En particular, se pueden observar los usuarios que existen actualmente en *MySQL* y un resumen de los permisos que tienen asignados. En realidad, esta información está almacenada en la tabla “*user*” de la base de datos “*mysql*”, que almacena información de sistema y que puede consultarse como cualquier otra tabla. Por ejemplo, a través del propio *phpMyAdmin*.



u000host.ehu.eus:8080 / localhost / mysql / user | phpMyAdmin 4.6.6deb5 - Mozilla Firefox

u000host.ehu.eus:8080/phpmyadmin/sql.php

Server: localhost:3306 » Database: mysql » Table: user "Users and global privileges"

Options: Host, User, Select_priv, Insert_priv, Update_priv, Delete_priv, Create_priv, Drop_priv

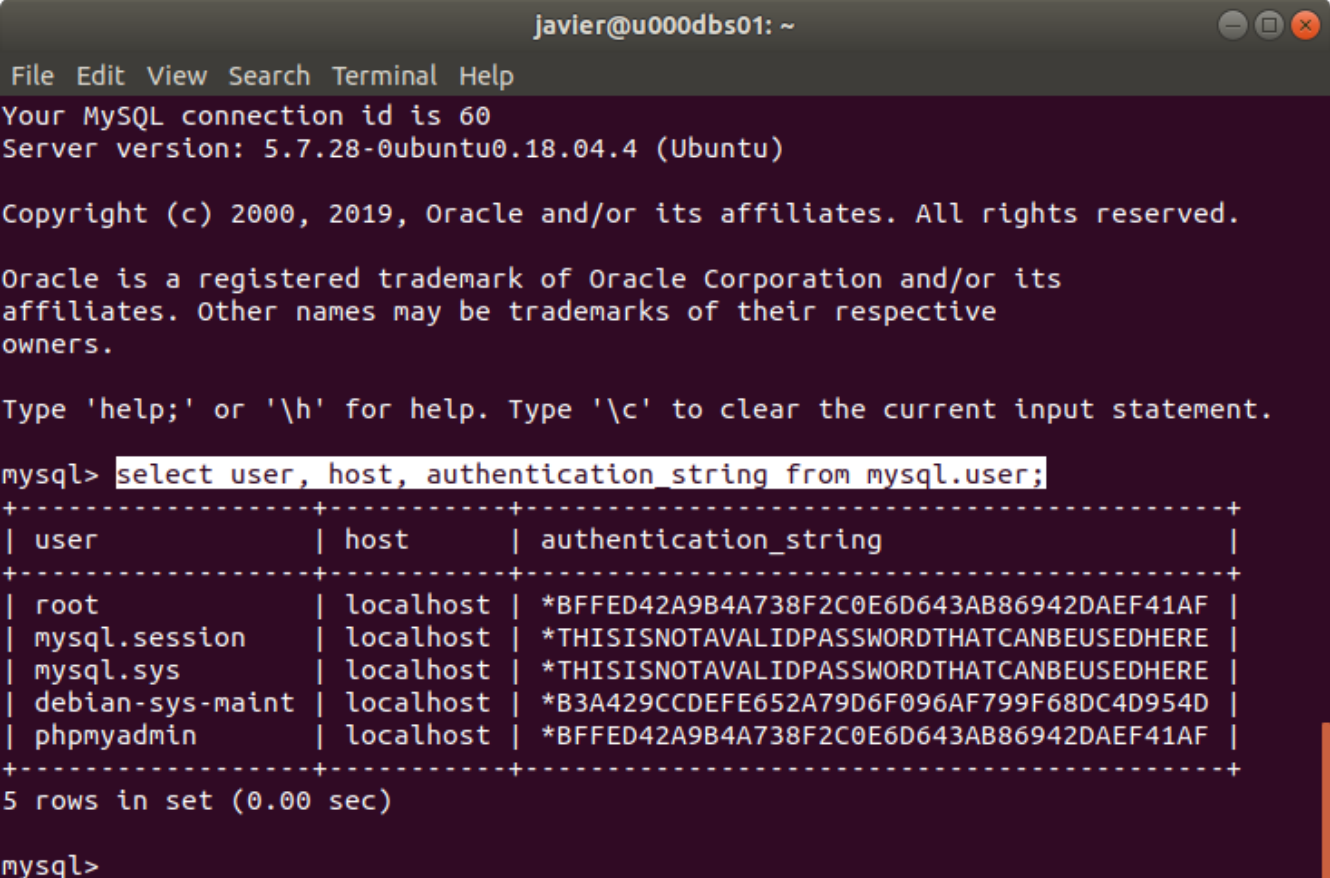
	Host	User	Select_priv	Insert_priv	Update_priv	Delete_priv	Create_priv	Drop_priv
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	localhost	root	Y	Y	Y	Y	Y	Y
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	localhost	mysql.session	N	N	N	N	N	N
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	localhost	mysql.sys	N	N	N	N	N	N
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	localhost	debian-sys-maint	Y	Y	Y	Y	Y	Y
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	localhost	phpmyadmin	N	N	N	N	N	N

Query results operations: Print, Copy to clipboard, Export, Display chart, Create view

Bookmark this SQL query: Label: ☐ Let every user access this bookmark

4. O bien a través del cliente de consola de *MySQL* si nos conectamos a nuestra máquina remota mediante *SSH*.

```
mysql> select user, host, authentication_string from mysql.user;
```

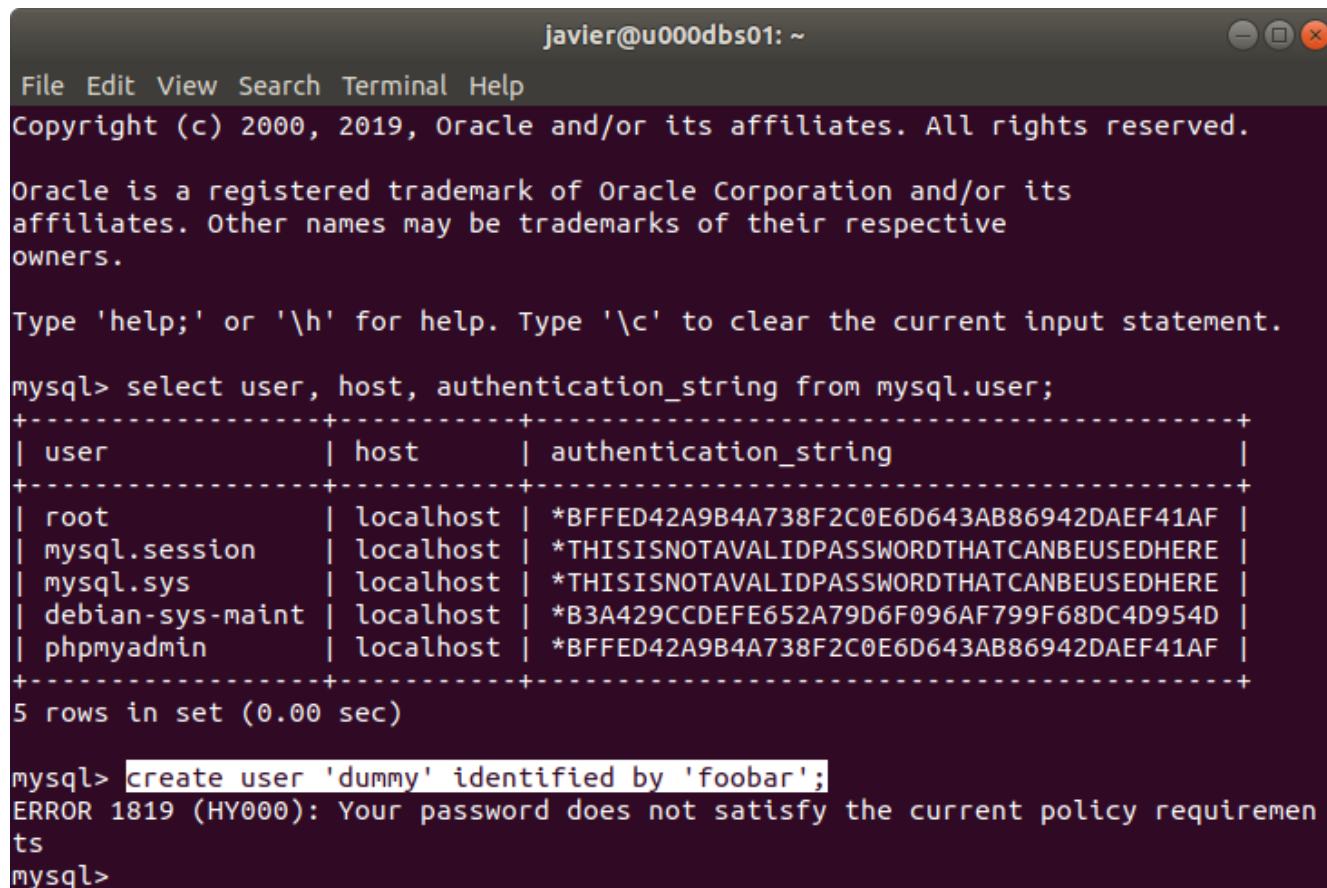


```
javier@u000db01: ~  
File Edit View Search Terminal Help  
Your MySQL connection id is 60  
Server version: 5.7.28-0ubuntu0.18.04.4 (Ubuntu)  
Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql> select user, host, authentication_string from mysql.user;  
+-----+-----+-----+  
| user          | host      | authentication_string |  
+-----+-----+-----+  
| root          | localhost | *BFFED42A9B4A738F2C0E6D643AB86942DAEF41AF |  
| mysql.session | localhost | *THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE |  
| mysql.sys     | localhost | *THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE |  
| debian-sys-maint | localhost | *B3A429CCDEF652A79D6F096AF799F68DC4D954D |  
| phpmyadmin    | localhost | *BFFED42A9B4A738F2C0E6D643AB86942DAEF41AF |  
+-----+-----+-----+  
5 rows in set (0.00 sec)  
mysql>
```

Como se puede observar, el campo *authentication_string* contiene la contraseña encriptada.

5. Desde el cliente de consola de *MySQL*, crea el usuario “*dummy*” y asignarle la contraseña “*foobar*”.

```
mysql> create user 'dummy' identified by 'foobar';
```



The screenshot shows a terminal window titled 'javier@u000db01: ~'. The terminal displays the MySQL command-line interface. It starts with a copyright notice and instructions. The user enters the command 'select user, host, authentication_string from mysql.user;'. The output is a table with 5 rows. Then, the user enters the command 'create user 'dummy' identified by 'foobar';'. This results in an error message: 'ERROR 1819 (HY000): Your password does not satisfy the current policy requirements'.

```
javier@u000db01: ~
File Edit View Search Terminal Help
Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

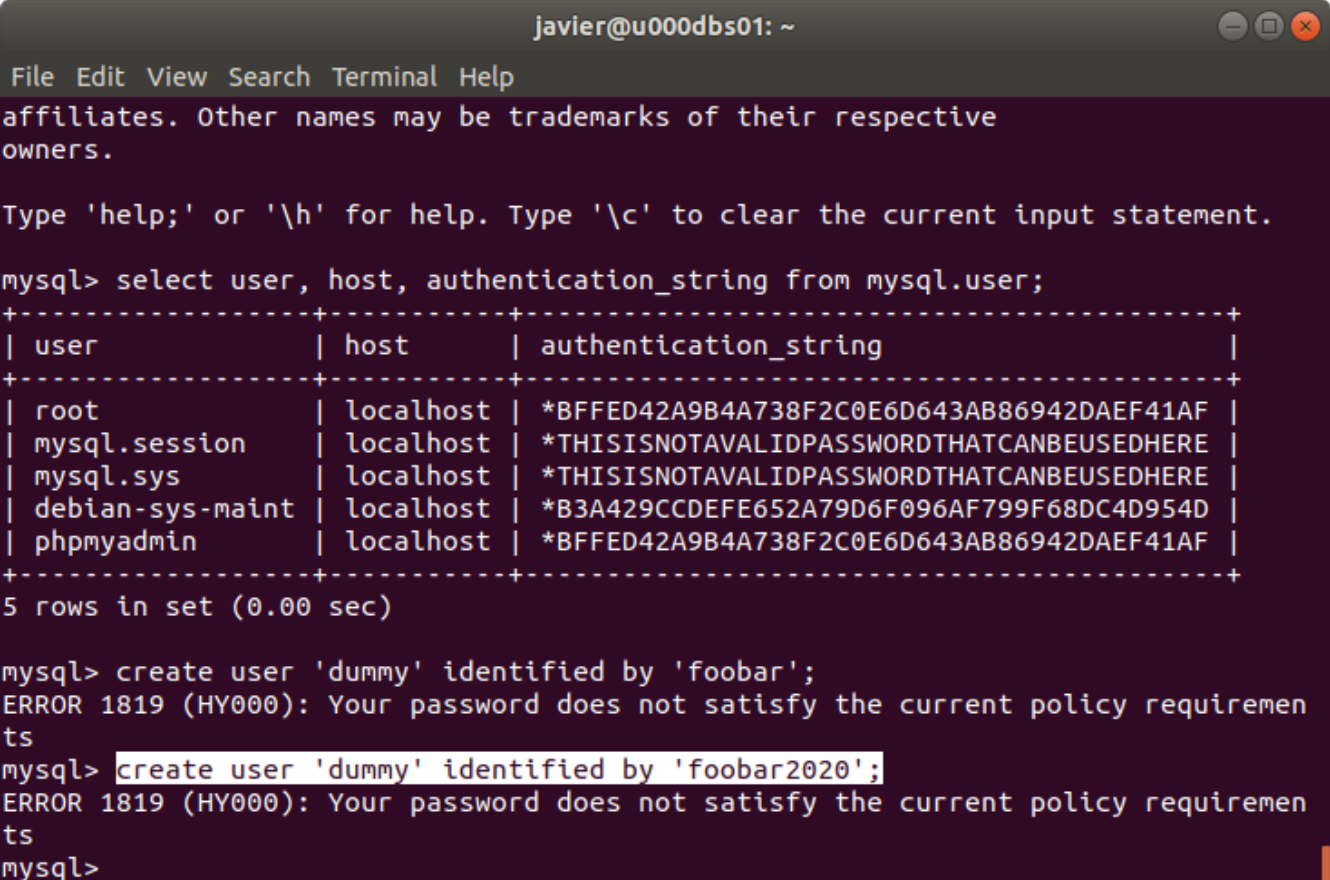
mysql> select user, host, authentication_string from mysql.user;
+-----+-----+-----+
| user          | host      | authentication_string |
+-----+-----+-----+
| root          | localhost | *BFFED42A9B4A738F2C0E6D643AB86942DAEF41AF |
| mysql.session | localhost | *THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE |
| mysql.sys     | localhost | *THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE |
| debian-sys-maint | localhost | *B3A429CCDEF652A79D6F096AF799F68DC4D954D |
| phpmyadmin    | localhost | *BFFED42A9B4A738F2C0E6D643AB86942DAEF41AF |
+-----+-----+-----+
5 rows in set (0.00 sec)

mysql> create user 'dummy' identified by 'foobar';
ERROR 1819 (HY000): Your password does not satisfy the current policy requirements
mysql>
```

El comando devuelve un error: la contraseña “foodbar” no cumple la política de *passwords* que se estableció en el laboratorio anterior.

6. Probemos con una contraseña más fuerte que combine letras y números:

```
mysql> create user 'dummy' identified by 'foobar2020';
```



The screenshot shows a terminal window titled 'javier@u000db01: ~'. The terminal displays the output of a MySQL query and an error message. The query 'select user, host, authentication_string from mysql.user;' returns a table with 5 rows. The error message 'ERROR 1819 (HY000): Your password does not satisfy the current policy requirements' is shown twice, once for 'foobar' and once for 'foobar2020'.

```
javier@u000db01: ~
File Edit View Search Terminal Help
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select user, host, authentication_string from mysql.user;
+-----+-----+-----+
| user          | host      | authentication_string          |
+-----+-----+-----+
| root          | localhost | *BFFED42A9B4A738F2C0E6D643AB86942DAEF41AF |
| mysql.session | localhost | *THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE |
| mysql.sys     | localhost | *THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE |
| debian-sys-maint | localhost | *B3A429CCDEF652A79D6F096AF799F68DC4D954D |
| phpmyadmin    | localhost | *BFFED42A9B4A738F2C0E6D643AB86942DAEF41AF |
+-----+-----+-----+
5 rows in set (0.00 sec)

mysql> create user 'dummy' identified by 'foobar';
ERROR 1819 (HY000): Your password does not satisfy the current policy requirements

mysql> create user 'dummy' identified by 'foobar2020';
ERROR 1819 (HY000): Your password does not satisfy the current policy requirements

mysql>
```

Este error es más confuso: aparentemente, “foobar2020” cumple la política establecida pero no se puede establecer como contraseña.

7. El problema es que el valor por defecto para el nivel de seguridad del *plugin* “*Validate Password*” de *MySQL* es “*MEDIUM*”, y éste se establece cada vez que se reinicia el servicio de *MySQL*:

```
mysql> show global variables like 'validate_password%';
```



```
mysql> show global variables like 'validate_password%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| validate_password_check_user_name | OFF |
| validate_password_dictionary_file |  |
| validate_password_length | 8 |
| validate_password_mixed_case_count | 1 |
| validate_password_number_count | 1 |
| validate_password_policy | MEDIUM |
| validate_password_special_char_count | 1 |
+-----+-----+
7 rows in set (0.01 sec)


mysql>
```

Sin embargo, nuestro *password* solo cumple los criterios del nivel “*LOW*”, que fue el nivel seleccionado al instalar el *plugin*.

8. Podemos solucionar este problema de forma provisional seleccionando de nuevo el nivel “*LOW*” del *plugin* “*Validate Password*” mediante la siguiente sentencia:

```
mysql> set global validate_password_policy=LOW;
```

Pero el *plugin* “*Validate Password*” volverá a establecerse al nivel “*MEDIUM*” después de cada reinicio del servicio de *MySQL*.



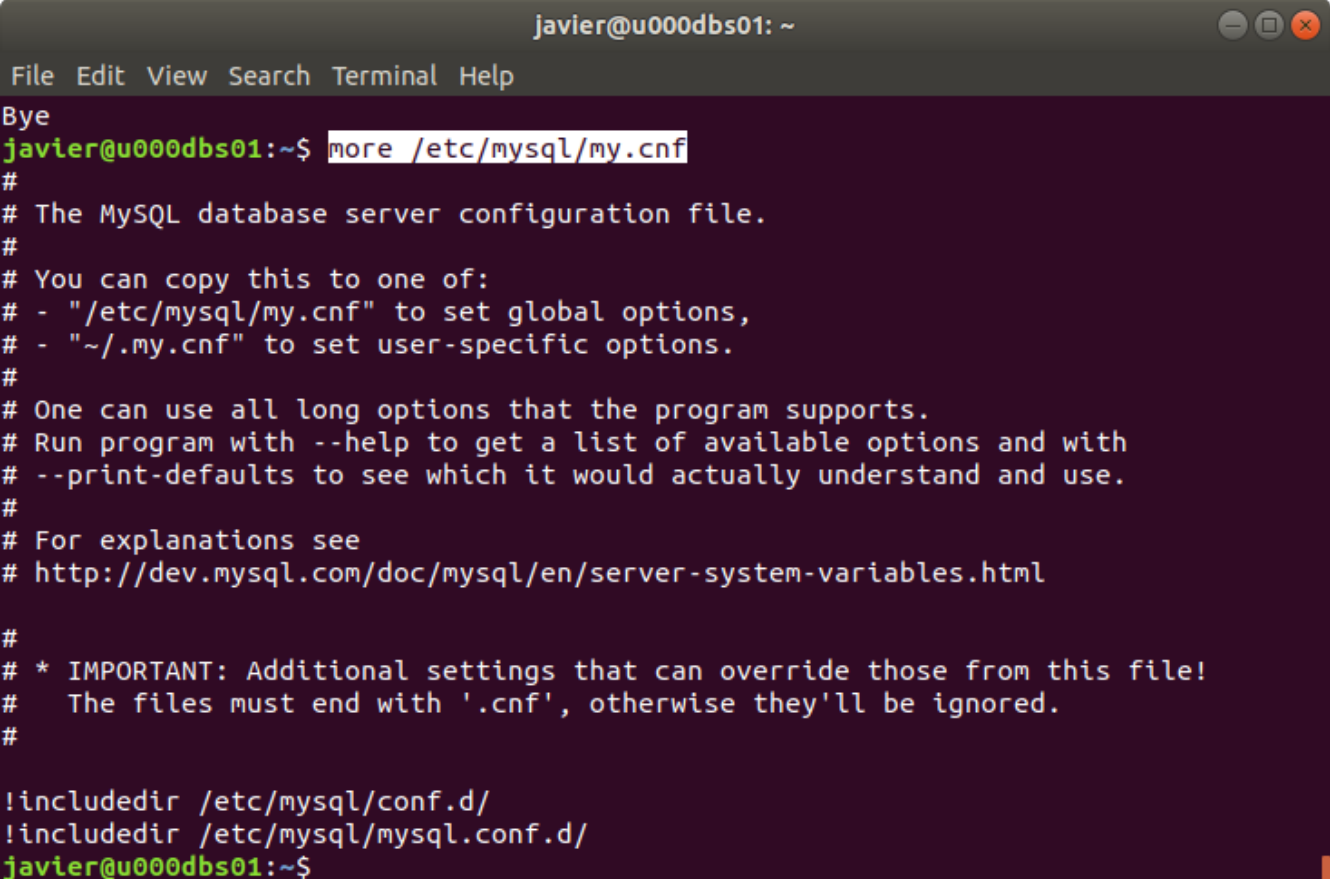
```
mysql> set global validate_password_policy=LOW;
Query OK, 0 rows affected (0.00 sec)

mysql> show global variables like 'validate_password%';
+-----+
| Variable_name | Value |
+-----+
| validate_password_check_user_name | OFF |
| validate_password_dictionary_file | |
| validate_password_length | 8 |
| validate_password_mixed_case_count | 1 |
| validate_password_number_count | 1 |
| validate_password_policy | LOW |
| validate_password_special_char_count | 1 |
+-----+
7 rows in set (0.00 sec)

mysql>
```

9. Una solución permanente sería establecer la configuración del *plugin* “*Validate Password*” en el propio fichero de configuración de *MySQL*. En versiones anteriores, toda la configuración se centralizaba en el fichero “*/etc/mysql/my.cnf*”:

```
:~$ more /etc/mysql/my.cnf
```

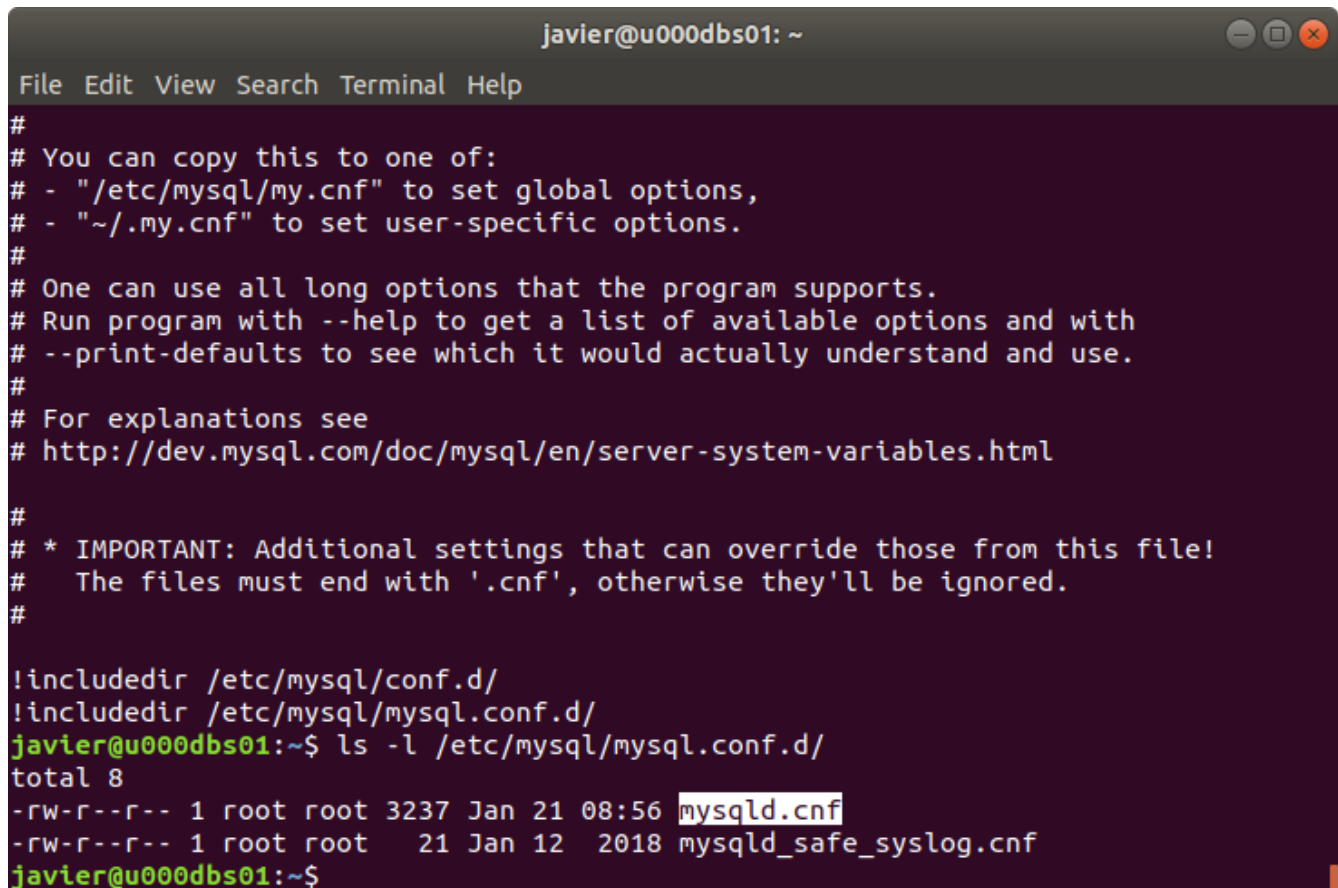


```
javier@u000db01: ~  
File Edit View Search Terminal Help  
Bye  
javier@u000db01:~$ more /etc/mysql/my.cnf  
#  
# The MySQL database server configuration file.  
#  
# You can copy this to one of:  
# - "/etc/mysql/my.cnf" to set global options,  
# - "~/.my.cnf" to set user-specific options.  
#  
# One can use all long options that the program supports.  
# Run program with --help to get a list of available options and with  
# --print-defaults to see which it would actually understand and use.  
#  
# For explanations see  
# http://dev.mysql.com/doc/mysql/en/server-system-variables.html  
#  
# * IMPORTANT: Additional settings that can override those from this file!  
#   The files must end with '.cnf', otherwise they'll be ignored.  
#  
!includedir /etc/mysql/conf.d/  
!includedir /etc/mysql/mysql.conf.d/  
javier@u000db01:~$
```

Sin embargo, en la versión actual la configuración está repartida en diferentes ficheros.

10. Para simplificar, en todas las prácticas de laboratorio utilizaremos el fichero “`/etc/mysql/mysql.conf.d/mysqld.cnf`” para centralizar los cambios de configuración que realicemos. Utilizar el siguiente comando para visualizar las propiedades del fichero:

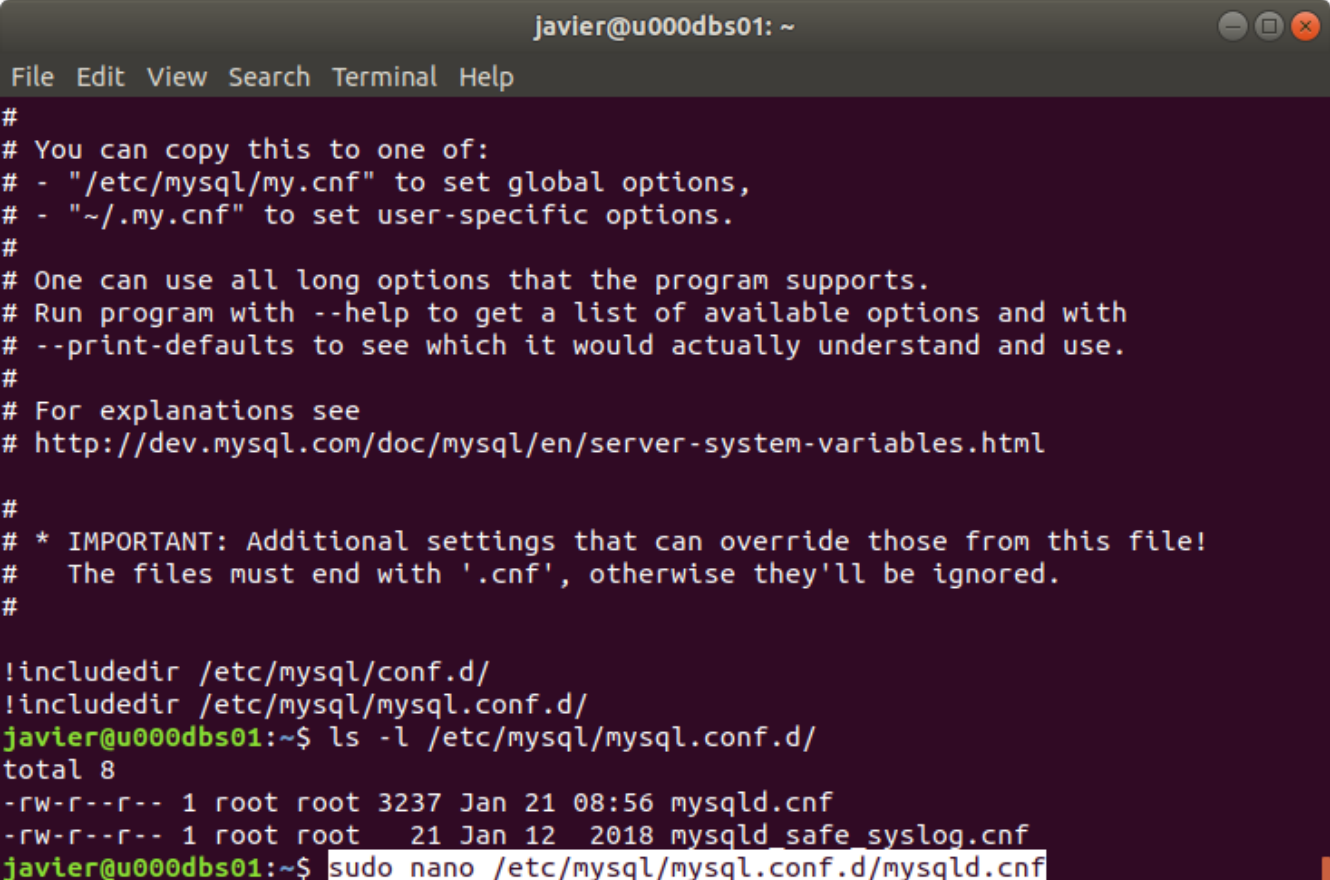
```
:~$ ls -l /etc/mysql/mysql.conf.d/mysqld.cnf
```



```
javier@u000db01: ~  
File Edit View Search Terminal Help  
#  
# You can copy this to one of:  
# - "/etc/mysql/my.cnf" to set global options,  
# - "~/.my.cnf" to set user-specific options.  
#  
# One can use all long options that the program supports.  
# Run program with --help to get a list of available options and with  
# --print-defaults to see which it would actually understand and use.  
#  
# For explanations see  
# http://dev.mysql.com/doc/mysql/en/server-system-variables.html  
#  
# * IMPORTANT: Additional settings that can override those from this file!  
# The files must end with '.cnf', otherwise they'll be ignored.  
#  
!includedir /etc/mysql/conf.d/  
!includedir /etc/mysql/mysql.conf.d/  
javier@u000db01:~$ ls -l /etc/mysql/mysql.conf.d/  
total 8  
-rw-r--r-- 1 root root 3237 Jan 21 08:56 mysqld.cnf  
-rw-r--r-- 1 root root  21 Jan 12 2018 mysqld_safe_syslog.cnf  
javier@u000db01:~$
```

11. Para editarlo, como son necesarios permisos de *root*, utilizaremos el siguiente comando:

```
:~$ sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```



```
javier@u000db01: ~  
File Edit View Search Terminal Help  
#  
# You can copy this to one of:  
# - "/etc/mysql/my.cnf" to set global options,  
# - "~/.my.cnf" to set user-specific options.  
#  
# One can use all long options that the program supports.  
# Run program with --help to get a list of available options and with  
# --print-defaults to see which it would actually understand and use.  
#  
# For explanations see  
# http://dev.mysql.com/doc/mysql/en/server-system-variables.html  
#  
# * IMPORTANT: Additional settings that can override those from this file!  
#   The files must end with '.cnf', otherwise they'll be ignored.  
#  
!includedir /etc/mysql/conf.d/  
!includedir /etc/mysql/mysql.conf.d/  
javier@u000db01:~$ ls -l /etc/mysql/mysql.conf.d/  
total 8  
-rw-r--r-- 1 root root 3237 Jan 21 08:56 mysqld.cnf  
-rw-r--r-- 1 root root  21 Jan 12  2018 mysqld_safe_syslog.cnf  
javier@u000db01:~$ sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

12. Al final del apartado *[mysqld]* (nombre del proceso que implementa el servicio de *MySQL*) del fichero de configuración (que coincide con el final del fichero), introduciremos la siguiente entrada para hacer que la configuración de la política sea permanente:

```
#
# * MySQL Validation Password plugin
#
validate_password_policy=LOW
```

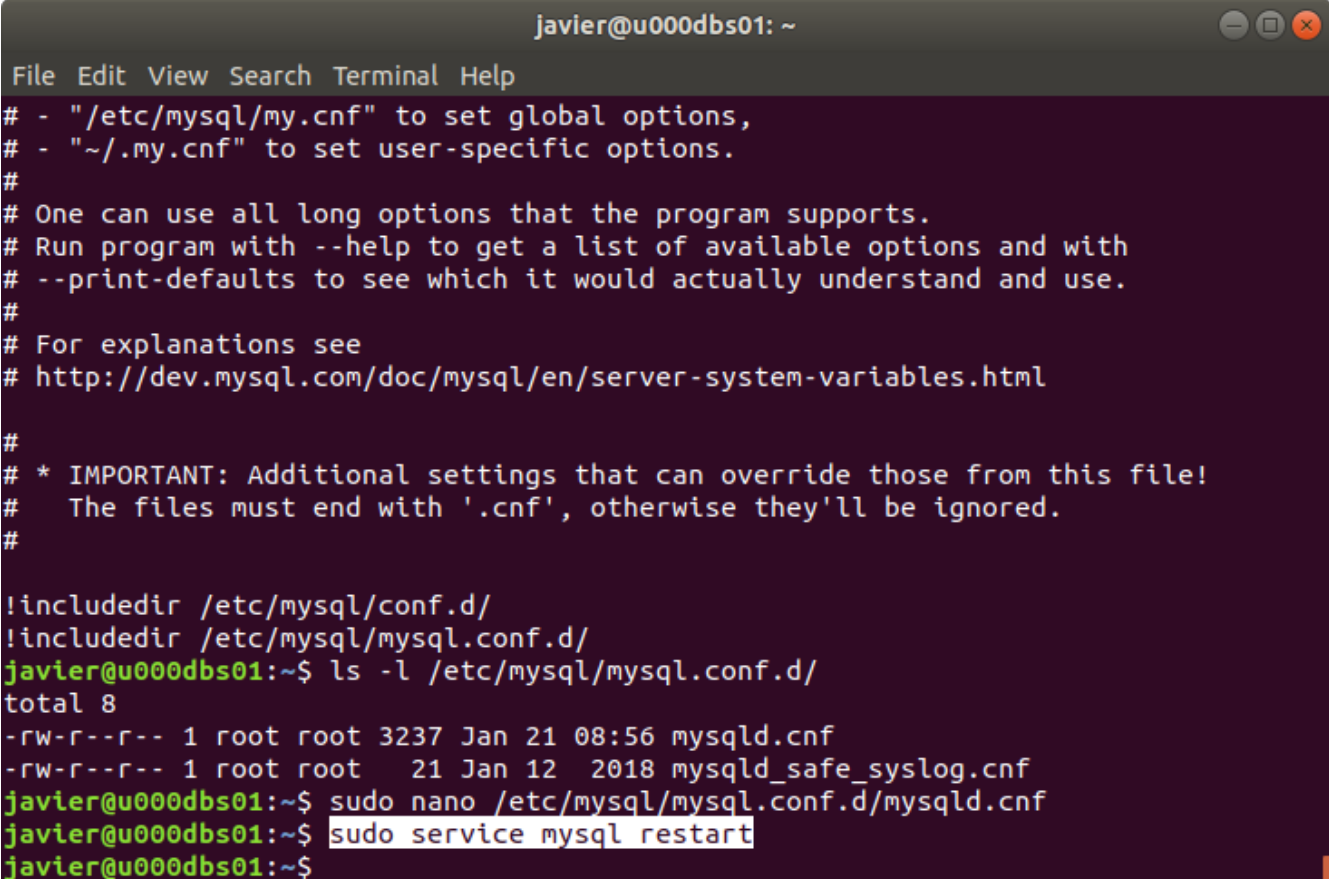


```
GNU nano 4.8 /etc/mysql/mysql.conf.d/mysqld.cnf Modified
# log-queries-not-using-indexes
#
# The following can be used as easy to replay backup logs or for replication.
# note: if you are setting up a replication slave, see README.Debian about
#       other settings you may need to change.
# server-id             = 1
# log_bin               = /var/log/mysql/mysql-bin.log
# binlog_expire_logs_seconds = 2592000
max_binlog_size        = 100M
# binlog_do_db          = include_database_name
# binlog_ignore_db      = include_database_name
#
#
# * MySQL Validation Password plugin
#
validate_password.policy=LOW
█

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File  ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line
```

13. Guardamos la configuración y reiniciamos el servicio de *MySQL*.

```
:~$ sudo service mysql restart
```

A screenshot of a terminal window titled 'javier@u000db01: ~'. The terminal shows the contents of the MySQL configuration file, followed by a command to list files in the configuration directory, and then commands to edit the configuration file and restart the MySQL service. The terminal output is as follows:

```
javier@u000db01: ~  
File Edit View Search Terminal Help  
# - "/etc/mysql/my.cnf" to set global options,  
# - "~/.my.cnf" to set user-specific options.  
#  
# One can use all long options that the program supports.  
# Run program with --help to get a list of available options and with  
# --print-defaults to see which it would actually understand and use.  
#  
# For explanations see  
# http://dev.mysql.com/doc/mysql/en/server-system-variables.html  
#  
# * IMPORTANT: Additional settings that can override those from this file!  
# The files must end with '.cnf', otherwise they'll be ignored.  
#  
!includedir /etc/mysql/conf.d/  
!includedir /etc/mysql/mysql.conf.d/  
javier@u000db01:~$ ls -l /etc/mysql/mysql.conf.d/  
total 8  
-rw-r--r-- 1 root root 3237 Jan 21 08:56 mysqld.cnf  
-rw-r--r-- 1 root root 21 Jan 12 2018 mysqld_safe_syslog.cnf  
javier@u000db01:~$ sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf  
javier@u000db01:~$ sudo service mysql restart  
javier@u000db01:~$
```

Si se obtiene un error antes a la hora de iniciar el servicio, es porque se intenta asignar la variable antes de que esté creada. Es decir, hay que cargar el módulo antes. Para ello, añadimos justo antes de la asignación, la siguiente línea:

```
[mysqld]  
plugin-load-add=validate_password.so  
validate_password_policy=LOW
```

14. Comprobar que la variable global `validate_password.policy` mantiene el valor “LOW” y probar de nuevo la sentencia para crear el usuario ‘dummy’:

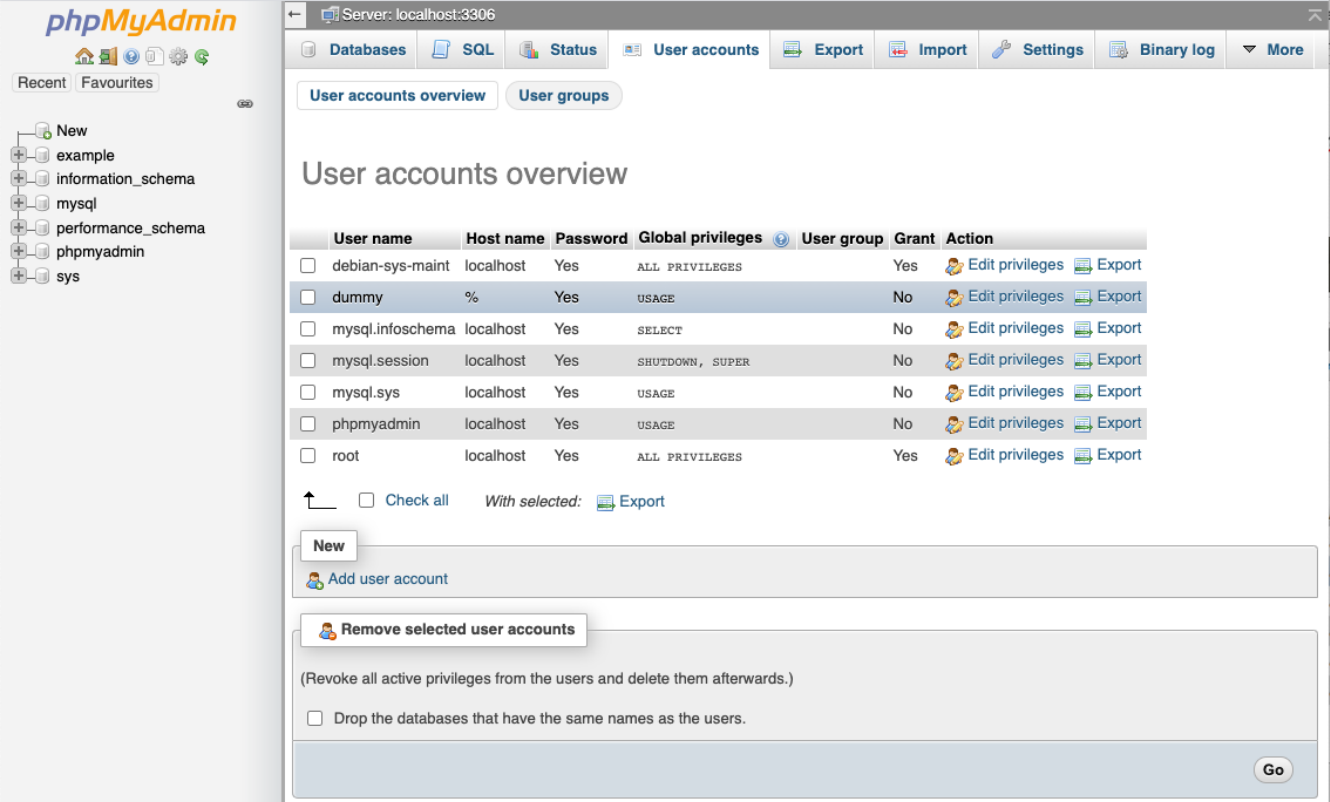
```
mysql>
mysql> show global variables like 'validate_password%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| validate_password.check_user_name | ON |
| validate_password.dictionary_file | |
| validate_password.length | 8 |
| validate_password.mixed_case_count | 1 |
| validate_password.number_count | 1 |
| validate_password.policy | LOW |
| validate_password.special_char_count | 1 |
+-----+-----+
7 rows in set (0.01 sec)

mysql>
mysql> create user 'dummy' identified by 'foobar2020';
Query OK, 0 rows affected (0.01 sec)

mysql>
```

Si la creación del usuario no ha sido correcta, revisar los pasos realizados hasta ahora antes de continuar con el laboratorio.

15. Comprueba que efectivamente se ha creado el usuario “dummy”. Por ejemplo, desde *phpMyAdmin*:



The screenshot shows the phpMyAdmin interface for a MySQL server at localhost:3306. The 'User accounts overview' page is displayed, showing a list of users and their privileges. The 'dummy' user is highlighted in blue, indicating it is selected. Below the table, there are options to 'Add user account' and 'Remove selected user accounts'.

User name	Host name	Password	Global privileges	User group	Grant	Action
<input type="checkbox"/> debian-sys-maint	localhost	Yes	ALL PRIVILEGES		Yes	Edit privileges Export
<input checked="" type="checkbox"/> dummy	%	Yes	USAGE		No	Edit privileges Export
<input type="checkbox"/> mysql.infoschema	localhost	Yes	SELECT		No	Edit privileges Export
<input type="checkbox"/> mysql.session	localhost	Yes	SHUTDOWN, SUPER		No	Edit privileges Export
<input type="checkbox"/> mysql.sys	localhost	Yes	USAGE		No	Edit privileges Export
<input type="checkbox"/> phpmyadmin	localhost	Yes	USAGE		No	Edit privileges Export
<input type="checkbox"/> root	localhost	Yes	ALL PRIVILEGES		Yes	Edit privileges Export

↑ ☐ Check all With selected: [Export](#)

New

[Add user account](#)

Remove selected user accounts

(Revoke all active privileges from the users and delete them afterwards.)

☐ Drop the databases that have the same names as the users.

[Go](#)

16. O bien desde el cliente de *MySQL* en consola:

```
mysql> select user, host, authentication_string from mysql.user;
```

```
mysql>
mysql> select user, host, authentication_string from mysql.user;
+-----+-----+-----+
| user          | host      | authentication_string |
+-----+-----+-----+
| W%NRK@Yqbay/1fC.TCcvDYan4gb1euKBWxB1c1PZK1KTFmci90 |
| debian-sys-maint | localhost | $A$005$QGJCThOW0cJ,0!SG2rLcVlr9RNkRGCBWmE5WQM.jjG.8GCLe.C31NrOW0o7 | |
| mysql.infoschema | localhost | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED |
| mysql.session    | localhost | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED |
| mysql.sys        | localhost | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED |
| phpmyadmin       | localhost | $A$005$0<SZ^|T3)"AoX /V48EOsSsf0CktFQhtpUw7Pb1Dn90ydGYsQIq6DsL15 |
| root             | localhost | *EC80DBDDF24974DD6D5998105958CD1CD1383694 |
+-----+-----+-----+
7 rows in set (0.00 sec)

mysql>
```

Con este comando, la tabla se recupera correctamente pero no conseguimos visualizar todos los datos de la 1ª fila debido a la longitud del campo “authentication_string”. Ya que ahora mismo no nos interesa visualizar este campo al completo, podemos mostrar sólo sus primeros caracteres con la función *substring*. El comando *substring* muestra un fragmento de un *string* y recibe 3 parámetros:

- El campo a seleccionar.
- El carácter en el que comenzar.
- El número de caracteres a mostrar.

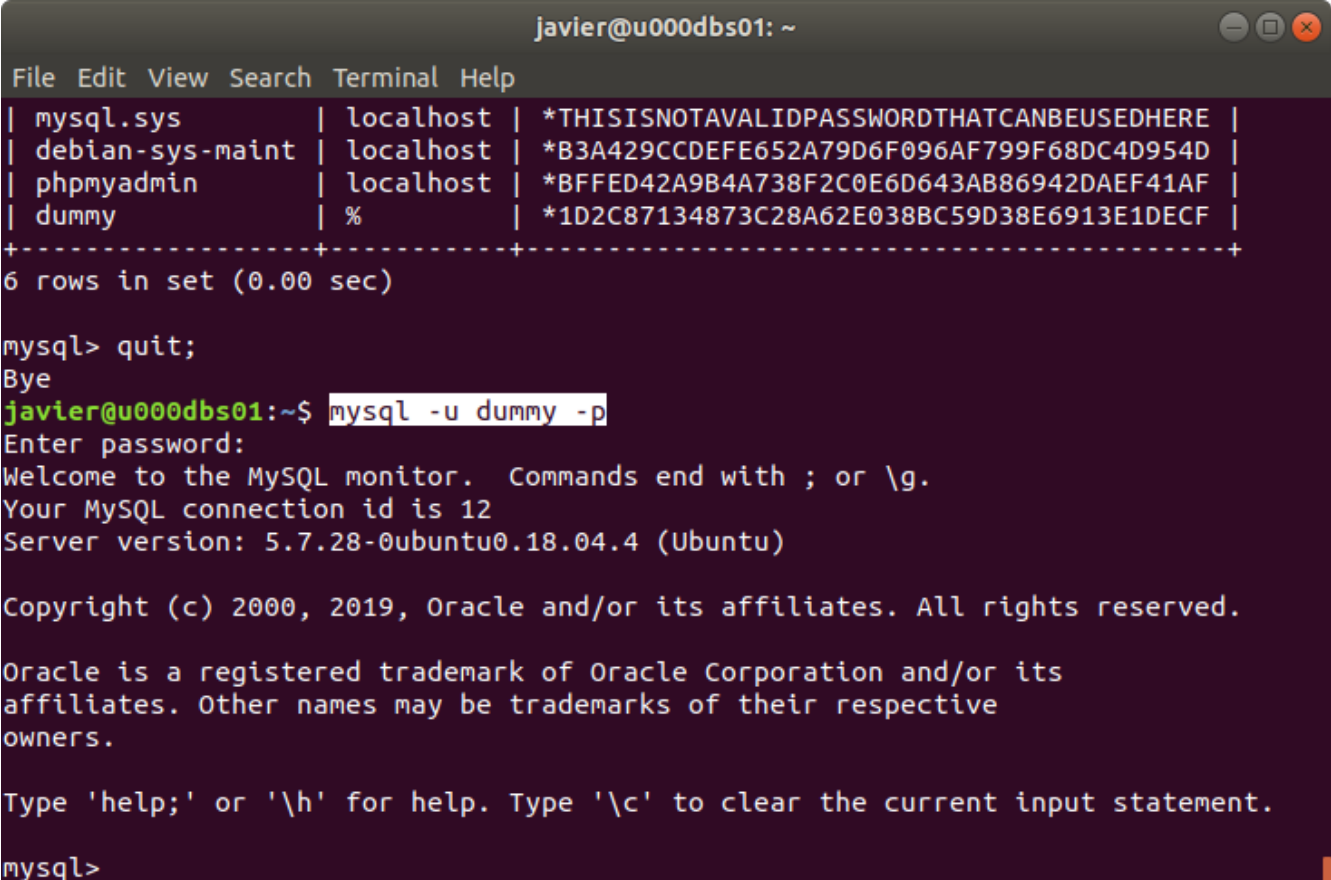
```
mysql> select user, host, substring(authentication_string,1,10) from
mysql.user;
```

```
mysql>
mysql> select user, host, substring(authentication_string,1,10) from mysql.user;
+-----+-----+-----+
| user          | host      | substring(authentication_string,1,10) |
+-----+-----+-----+
| dummy         | %         | $A$005$M |
| debian-sys-maint | localhost | $A$005$QG |
| mysql.infoschema | localhost | $A$005$THI |
| mysql.session   | localhost | $A$005$THI |
| mysql.sys       | localhost | $A$005$THI |
| phpmyadmin      | localhost | $A$005$0 |
| root           | localhost | *EC80DBDDF |
+-----+-----+-----+
7 rows in set (0.00 sec)

mysql>
mysql>
```

17. Cierra la sesión abierta en el cliente de consola de *MySQL* y conéctate utilizando el usuario “dummy”. ¿Qué ocurre?

```
:~$ mysql -u dummy -p
```



The screenshot shows a terminal window titled 'javier@u000db01: ~'. The terminal displays the output of a MySQL query, followed by the user exiting the MySQL monitor and then re-entering it as the 'dummy' user. The output of the query is as follows:

mysql.sys	localhost	*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE
debian-sys-maint	localhost	*B3A429CCDEF652A79D6F096AF799F68DC4D954D
phpmyadmin	localhost	*BFFED42A9B4A738F2C0E6D643AB86942DAEF41AF
dummy	%	*1D2C87134873C28A62E038BC59D38E6913E1DEC

6 rows in set (0.00 sec)

mysql> quit;
Bye
javier@u000db01:~\$ mysql -u dummy -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.7.28-0ubuntu0.18.04.4 (Ubuntu)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

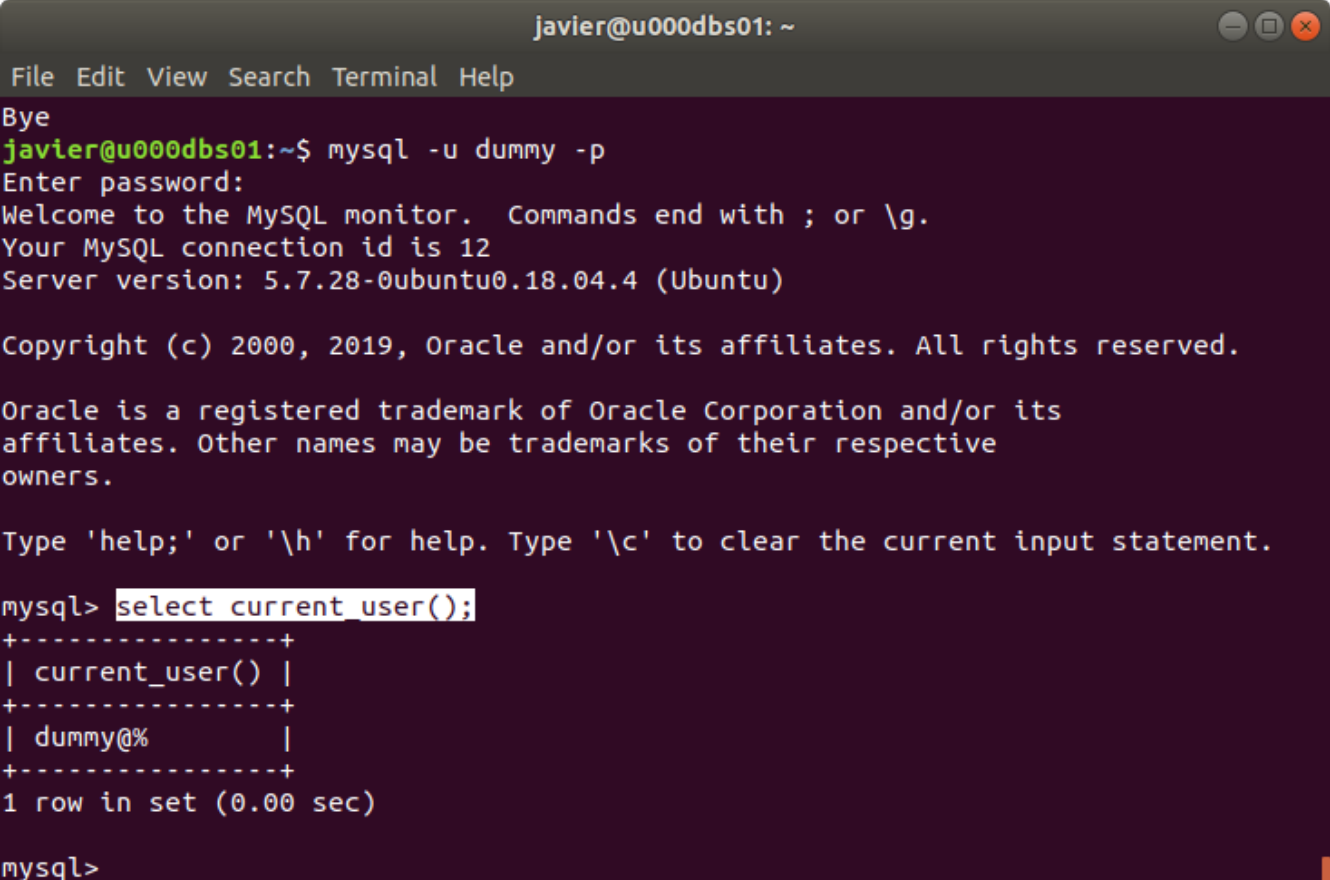
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>

Si no has conseguido iniciar sesión correctamente, revisa los pasos anteriores antes de continuar.

18. Usando la función *current_user()*, comprueba cuál es el usuario que está actualmente conectado.

```
mysql> select current_user();
```



The screenshot shows a terminal window titled 'javier@u000db01: ~'. The terminal content is as follows:

```
File Edit View Search Terminal Help
Bye
javier@u000db01:~$ mysql -u dummy -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.7.28-0ubuntu0.18.04.4 (Ubuntu)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

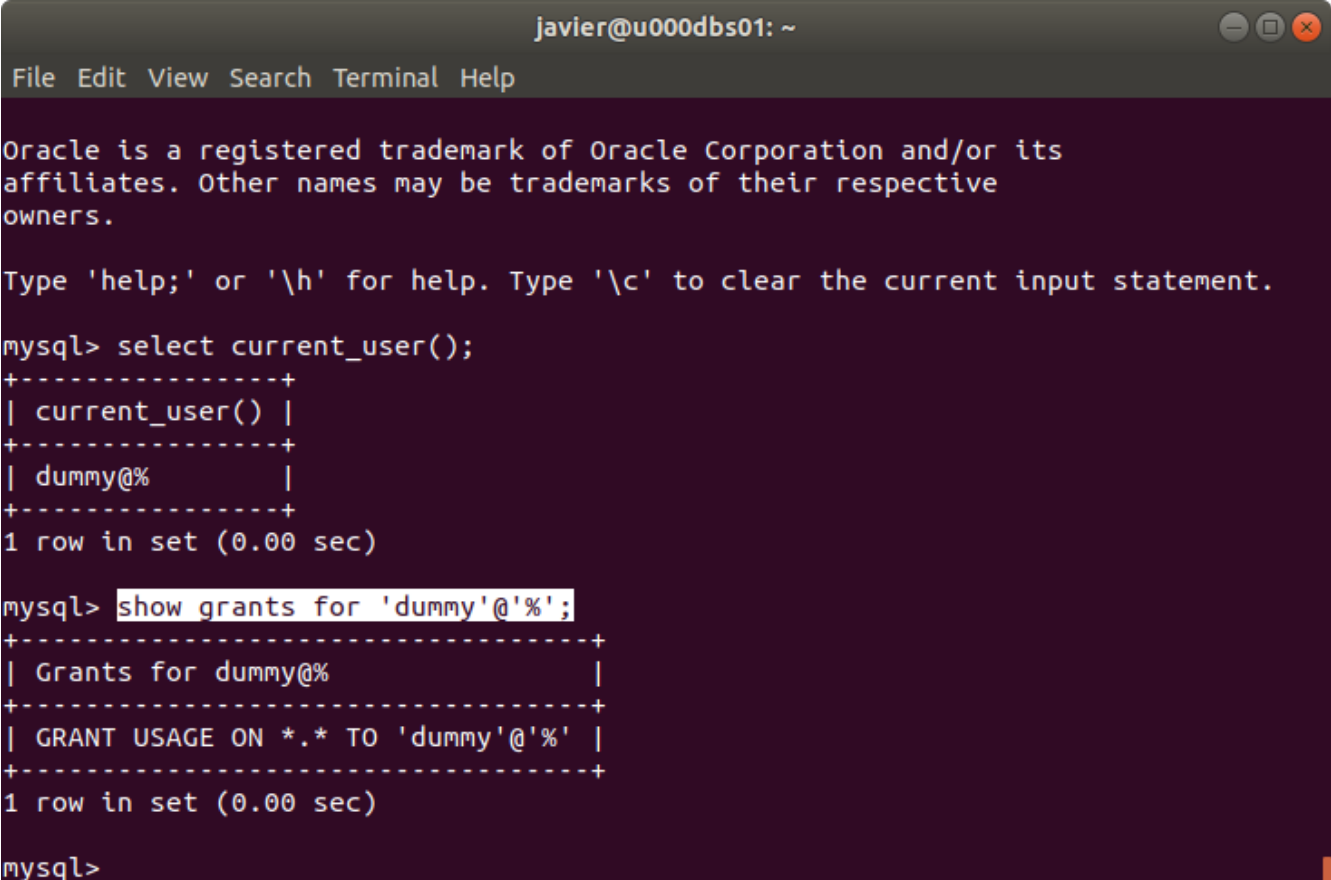
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select current_user();
+-----+
| current_user() |
+-----+
| dummy@%       |
+-----+
1 row in set (0.00 sec)

mysql>
```

19. Un comando muy útil para realizar esta práctica de laboratorio es “*show grants for*”, que nos lista los permisos de los que dispone un usuario actualmente:

```
mysql> show grants for 'dummy'@'%';
```



The screenshot shows a terminal window titled "javier@u000db01: ~". The terminal displays the following text:

```
File Edit View Search Terminal Help

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select current_user();
+-----+
| current_user() |
+-----+
| dummy@%       |
+-----+
1 row in set (0.00 sec)

mysql> show grants for 'dummy'@'%';
+-----+
| Grants for dummy@%          |
+-----+
| GRANT USAGE ON *.* TO 'dummy'@'%' |
+-----+
1 row in set (0.00 sec)

mysql>
```

Como acabamos de crear el usuario, tan solo dispone de los permisos básicos (*usage*).

En la siguiente práctica de laboratorio ya veremos más comandos útiles como éste.

20. Desconéctate del cliente de consola de *MySQL* y conéctate de nuevo con el usuario “*root*” para crear los siguientes usuarios anónimos:

```
mysql> create user '' identified by 'anyAnonymous';
```

```
mysql> create user ''@'localhost' identified by 'localAnonymous';
```

```
mysql> create user '' identified by 'anyAnonymous';
Query OK, 0 rows affected (0.01 sec)

mysql> create user ''@'localhost' identified by 'localAnonymous';
Query OK, 0 rows affected (0.01 sec)

mysql> select user, host, substring(authentication_string, 1, 10) from mysql.user;
+-----+-----+-----+
| user          | host      | substring(authentication_string, 1, 10) |
+-----+-----+-----+
| dummy         | %         | $$005$uR+                             |
|               | %         | $$005$M                               |
|               | localhost | $$005$Ipm                             |
| debian-sys-maint | localhost | $$005$QG                             |
| mysql.infoschema | localhost | $$005$THI                             |
| mysql.session  | localhost | $$005$THI                             |
| mysql.sys      | localhost | $$005$THI                             |
| phpmyadmin     | localhost | $$005$O                               |
| root          | localhost | *EC80DBDDF                           |
+-----+-----+-----+
9 rows in set (0.00 sec)

mysql>
```

21. Desconéctate de nuevo del cliente de consola de *MySQL* y conéctate utilizando el usuario “*dummy*”. ¿Qué ocurre ahora?

```
mysql>
mysql> select user, host, substring(authentication_string, 1, 10) from mysql.user;
```

user	host	substring(authentication_string, 1, 10)
	%	\$A\$005\$uR+
dummy	%	\$A\$005\$M
	localhost	\$A\$005\$Ipm
debian-sys-maint	localhost	\$A\$005\$QG
mysql.infoschema	localhost	\$A\$005\$THI
mysql.session	localhost	\$A\$005\$THI
mysql.sys	localhost	\$A\$005\$THI
phpmyadmin	localhost	\$A\$005\$0
root	localhost	*EC80DBDDF

```
9 rows in set (0.00 sec)

mysql>
mysql> ^DBye
ubuntu@ip-172-31-31-78:~$ mysql -u dummy -p
Enter password:
ERROR 1045 (28000): Access denied for user 'dummy'@'localhost' (using password: YES)
ubuntu@ip-172-31-31-78:~$
```

Continúa con el laboratorio para entender el motivo de este error.

2 Identificación de usuarios en MySQL

El sistema de control de acceso de *MySQL* se basa en 2 aspectos: por una parte, el identificador utilizado para el inicio de la sesión; por otra parte, la ubicación desde la cual se inicia la conexión. En cuanto a la ubicación, nos centraremos en 2 casos: ubicación “*localhost*”, cuando la conexión se realiza desde la propia máquina donde se está ejecutando *MySQL*; ubicación “%” (o *any*), cuando la conexión se realiza desde cualquier máquina.

A la hora de asignar privilegios en *MySQL*, se ha de especificar un identificador y una ubicación (si no se especifica una ubicación, por defecto se suele considerar “%”). Es decir, aunque se use un único identificador, podemos asignar privilegios distintos en función de la ubicación desde la cual se conecte. En la práctica, esto implica que en realidad no se trataría de un único usuario, si no que se estaría definiendo un usuario por cada ubicación desde la cual se conecte (aunque el identificador sea el mismo).

En nuestro caso, en el paso anterior hemos creado un usuario con identificador “*dummy*” para la ubicación “%” (*any*), como podemos observar en la tabla *mysql.users*.

```
mysql>
mysql> select user, host, substring(authentication_string, 1, 10) from mysql.user;
```

user	host	substring(authentication_string, 1, 10)
dummy	%	\$A\$005\$uR+
dummy	%	\$A\$005\$M
debian-sys-maint	localhost	\$A\$005\$Ipm
mysql.infoschema	localhost	\$A\$005\$QG
mysql.session	localhost	\$A\$005\$THI
mysql.sys	localhost	\$A\$005\$THI
phpmyadmin	localhost	\$A\$005\$0
root	localhost	*EC80DBDDF

```
9 rows in set (0.00 sec)

mysql>
mysql> ^DBye
ubuntu@ip-172-31-31-78:~$ mysql -u dummy -p
Enter password:
ERROR 1045 (28000): Access denied for user 'dummy'@'localhost' (using password: YES)
ubuntu@ip-172-31-31-78:~$
```

Después de definir “dummy” para la ubicación “%” (cualquier ubicación o *any*), inicialmente hemos podido conectarnos. Pero después de crear los usuarios anónimos, no.

¿Y por qué antes de crear los usuarios anónimos sí que podíamos conectarnos y ahora no? La respuesta es que *MySQL* ordena las cuentas de usuario siguiendo un criterio prefijado y sólo trata de autenticar al usuario con la primera cuenta que ajusta siguiendo dicho orden. Si consultas la siguiente dirección del manual

<https://dev.mysql.com/doc/refman/8.0/en/connection-access.html>

podrás conocer cuál es el criterio de ordenación que utiliza.

The screenshot shows the MySQL 8.0 Reference Manual page for 'connection-access.html'. On the left is a navigation sidebar with a search bar and a table of contents. The main content area explains how MySQL sorts user accounts for connection verification.

Search this Manual

Documentation Home

MySQL 8.0 Reference Manual

- Preface and Legal Notices
- General Information
- Installing and Upgrading MySQL
- Tutorial
- MySQL Programs
- MySQL Server Administration
- ▼ Security
 - General Security Issues
 - ▼ Access Control and Account Management
 - Account User Names and Passwords
 - Privileges Provided by MySQL
 - Grant Tables
 - Specifying Account Names
 - Specifying Role Names
 - **Access Control, Stage 1: Connection Verification**
 - Access Control, Stage 2: Request Verification
 - Adding Accounts, Assigning Privileges, and Dropping Accounts
 - Reserved Accounts

To see how this works, suppose that the `user` table looks like this:

	Host	User	...
1	%	root	...
2	%	jeffrey	...
3	localhost	root	...
4	localhost		...

When the server reads the table into memory, it sorts the rows using the rules just described. The result after sorting looks like this:

	Host	User	...
1	localhost	root	...
2	localhost		...
3	%	jeffrey	...
4	%	root	...

When a client attempts to connect, the server looks through the sorted rows and uses the first match found. For a connection from `localhost` by `jeffrey`, two of the rows from the table match: the one with `Host` and `User` values of `'localhost'` and `''`, and the one with values of `'%'` and `'jeffrey'`. The `'localhost'` row appears first in sorted order, so that is the one the server uses.

Here is another example. Suppose that the `user` table looks like this:

En primer lugar, *MySQL* ordena las cuentas de usuario según su ubicación, de tal manera que aquellas cuentas con una ubicación específica preceden a las cuentas con ubicación genérica. Las direcciones DNS y las direcciones IP se consideran como completamente específicas, mientras que la ubicación “%” (cualquier ubicación o *any*) se considera como genérica. Además, “” también es una ubicación válida que se interpreta de la misma manera que “%” (cualquier ubicación o *any*), pero las cuentas de usuario con ubicación “%” preceden a aquellas cuentas que tienen ubicación “”. Es decir, “” es la ubicación más genérica.

En segundo lugar, las cuentas de usuario que tienen la misma ubicación son ordenadas según su identificador, donde los identificadores más específicos preceden a los más genéricos. El identificador “” (usuario anónimo o cualquier usuario) es el más genérico, mientras que el resto son igual de específicos.

Por último, las cuentas de usuario con identificadores y ubicaciones igual de específicas respectivamente son ordenadas de forma indeterminada.

Según este criterio, es fácil comprobar que *MySQL* ha tratado de validar el último intento de conexión con la cuenta de identificador “” y ubicación “localhost” (usuario anónimo).

Como no tenemos control (en principio) sobre la ordenación de usuarios que hace *MySQL* (y porque es interesante entender correctamente cómo funciona *MySQL*), para conectarnos a *MySQL* usando el identificador “dummy” necesitamos crear un usuario con ese identificador asociado a la ubicación “localhost”. Para indicar la ubicación asociada a un identificador, en *MySQL* se utilizan expresiones de la siguiente forma:

[identificador] @ [ubicación]

Por defecto, la ubicación es “%”.

Por lo tanto, para crear el usuario “*dummy*” para la ubicación “*localhost*” utiliza la siguiente instrucción:

```
mysql> create user 'dummy'@'localhost' identified by 'localFoobar';
```

```
ubuntu@ip-172-31-31-78:~$ mysql -u dummy -p
Enter password:
ERROR 1045 (28000): Access denied for user 'dummy'@'localhost' (using password: YES)
ubuntu@ip-172-31-31-78:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 155
Server version: 8.0.22-0ubuntu0.20.04.3 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create user 'dummy'@'localhost' identified by 'localFoobar';
Query OK, 0 rows affected (0.00 sec)
```

Como resultado de la acción anterior, ahora existen 2 usuarios en *MySQL* que utilizan el identificador “dummy”, uno para ubicación “localhost” y otro para la ubicación “%”.

```
mysql> create user 'dummy'@'localhost' identified by 'localFoobar';
Query OK, 0 rows affected (0.00 sec)

mysql> select user, host, substring(authentication_string, 1, 10) from mysql.user;
```

user	host	substring(authentication_string, 1, 10)
	%	\$A\$005\$uR+
dummy	%	\$A\$005\$M
	localhost	\$A\$005\$Ipm
debian-sys-maint	localhost	\$A\$005\$QG
dummy	localhost	\$A\$005\$:DU
mysql.infoschema	localhost	\$A\$005\$THI
mysql.session	localhost	\$A\$005\$THI
mysql.sys	localhost	\$A\$005\$THI
phpmyadmin	localhost	\$A\$005\$0
root	localhost	*EC80DBDDF

```
10 rows in set (0.00 sec)

mysql> █
```

¿Qué *password* habrá que utilizar ahora para conectarse usando el identificador “*dummy*”?

```
ubuntu@ip-172-31-31-78:~$ mysql -u dummy -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 158
Server version: 8.0.22-0ubuntu0.20.04.3 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> █
```

IMPORTANTE: tened en cuenta que la asignación de privilegios se tiene que hacer en función de la ubicación, puesto que en la práctica se trata de cuentas de usuario diferentes.

Antes de pasar a la siguiente y si no quieres tener problemas más adelante, es conveniente que elimines los usuarios anónimos y el usuario “*dummy@localhost*” que acabas de crear.

```
ubuntu@ip-172-31-31-78:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 159
Server version: 8.0.22-0ubuntu0.20.04.3 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> drop user '@%';
Query OK, 0 rows affected (0.01 sec)

mysql> drop user '@localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> drop user 'dummy'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> select user, host, substring(authentication_string, 1, 10) from mysql.user;
+-----+-----+-----+
| user          | host      | substring(authentication_string, 1, 10) |
+-----+-----+-----+
| dummy         | %         | $$005$M                                |
| debian-sys-maint | localhost | $$005$QG                                |
| mysql.infoschema | localhost | $$005$THI                              |
| mysql.session  | localhost | $$005$THI                              |
| mysql.sys      | localhost | $$005$THI                              |
| phpmyadmin     | localhost | $$005$0                                |
| root           | localhost | *EC80DBDDF                             |
+-----+-----+-----+
7 rows in set (0.00 sec)

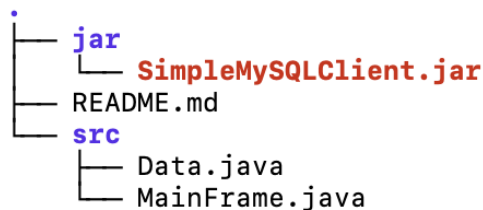
mysql> █
```

3 Cliente Java para MySQL

En esta sección veremos cómo poner en marcha una aplicación Java muy simple que hará las veces de cliente MySQL. Esta aplicación servirá para comprobar mejor la necesidad de diferentes roles al administrar una BBDD. La aplicación se encuentra en el siguiente repositorio de código:

<https://github.com/ulopeznovoa/ABD-DBK-Lab2-MySQL-Client>

El contenido del repositorio es el siguiente, representado en forma de árbol:



- Carpeta **jar**: contiene el paquete Java “SimpleMySQLClient.jar” con la aplicación lista para ser usada.
- Fichero **Readme.md**: contiene unas notas sobre el repositorio de código.
- Carpeta **src**: contiene los ficheros de código fuente de la aplicación.

En vuestro equipo local, utilizad el siguiente comando descargar el repositorio en una carpeta de vuestra elección y después, verificar que se ha descargado correctamente:

```
:~$ git clone https://github.com/ulopeznovoa/ABD-DBK-Lab2-MySQL-Client
:~$ ls -l ABD-DBK-Lab2-MySQL-Client
```

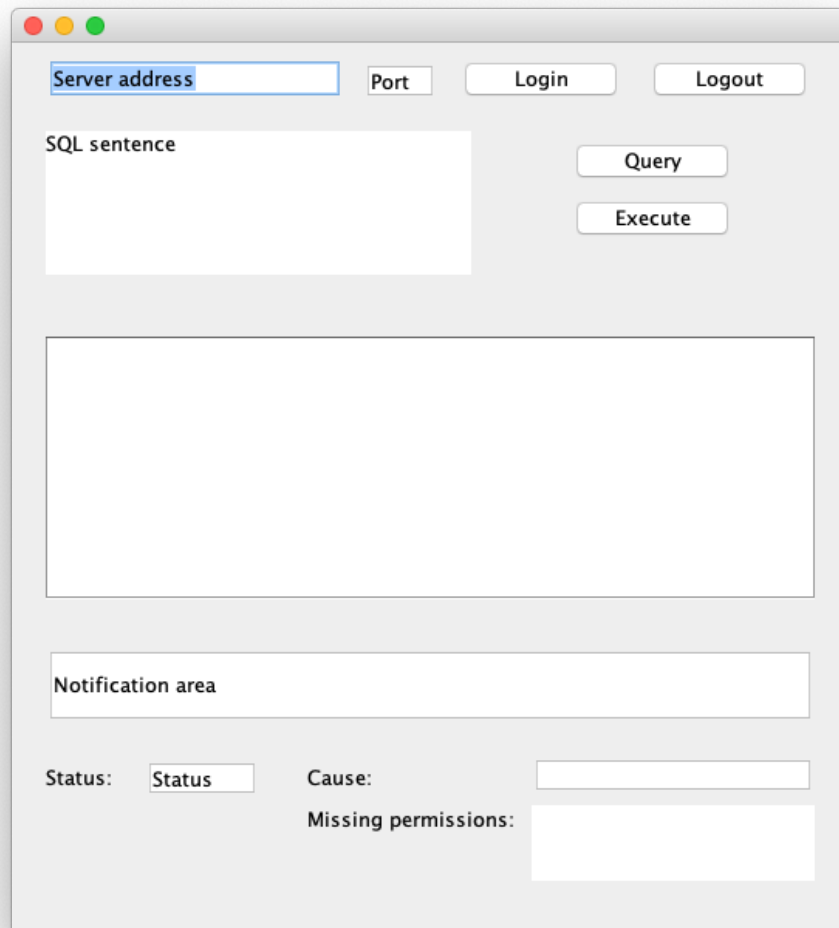
```
unai@MBP /tmp % git clone https://github.com/ulopeznovoa/ABD-DBK-Lab2-MySQL-Client
Cloning into 'ABD-DBK-Lab2-MySQL-Client'...
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 13 (delta 0), reused 10 (delta 0), pack-reused 0
Unpacking objects: 100% (13/13), done.
unai@MBP /tmp % ls -l ABD-DBK-Lab2-MySQL-Client
total 8
-rw-r--r--  1 unai  wheel   759 26 ene 10:09 README.md
drwxr-xr-x  3 unai  wheel    96 26 ene 10:09 jar
drwxr-xr-x  4 unai  wheel   128 26 ene 10:09 src
```

El resultado del segundo comando “ls -l” tiene que mostrar el contenido del repositorio. Si el primer comando genera un error por falta de tener un cliente “git” en el equipo local, se puede descargar el contenido del repositorio en un fichero comprimido .zip desde el repositorio: <https://github.com/ulopeznovoa/ABD-DBK-Lab2-MySQL-Client/archive/main.zip>

Para poner en marcha el cliente, utilizar el comando “java” desde el directorio “jar”

```
:~$ cd ABD-DBK-Lab2-MySQL-Client/jar  
:~$ java -jar SimpleMySQLClient.jar
```

Se mostrará la interfaz principal de la aplicación:



Si la interfaz no se muestra y en su lugar la terminal devuelve un error relacionado con la falta de un entorno Java, posiblemente habrá que instalar un entorno de ejecución de Java (JRE) o un entorno de desarrollo de Java (JDK). Se recomienda descargar el instalador adecuado en la web de Oracle: <https://www.oracle.com/java/technologies/javase-downloads.html>

La aplicación tiene las siguientes partes:

- Un *área de conexión*, con las cajas de texto “*Server address*” y “*Port*” y los botones (“*Login*” y “*Logout*”). En este área se indica la dirección y el puerto del servidor *MySQL* y al pulsar el botón “*Login*”, la aplicación mostrará una ventana emergente en la que se solicitará el nombre de usuario y contraseña para realizar la conexión. Al pulsar el botón “*Logout*”, se cerrará la conexión.
- Un *área de trabajo*, compuesta por 1 panel de texto (“*SQL sentence*”) y 2 botones (“*Query*” y “*Execute*”). En el panel de texto, se introduce la sentencia SQL a ejecutar. A continuación, si la sentencia es una consulta, se ejecutará tras pulsar el botón “*Query*”. De lo contrario, se ejecutará tras pulsar el botón “*Execute*”.
- Un *área de resultados* de consultas, donde se mostrarán los resultados devueltos por la consulta SQL que se han ejecutado correctamente.
- Un *área de notificaciones*, donde se mostrará el número de filas actualizadas (en el caso de actualizaciones) o bien el número de filas devueltas (en el caso de consultas) para las sentencias que se han ejecutado correctamente.
- Un *área de estado*, compuesto por 2 cajas de texto (“*Status*” y “*Cause*”) y 1 panel de texto (“*Missing permissions*”). En este área, se indica si la ejecución de la sentencia ha sido correcta (“*Status: Ok*”) o no (“*Status: Failure*”) en la caja de texto “*Status*”. En caso de las sentencias que no se hayan ejecutado correctamente, se indica si el error se debe a la falta de permisos (“*Cause: Denied*”) o bien a que la sentencia no es correcta (“*Cause: Syntax error*”) en la caja de texto “*Cause*”. Si la ejecución de la sentencia no ha sido correcta debido a un problema de permisos, se indican los permisos que han motivado el error (“*Missing permissions: ... for ... to ...*”) en el panel de texto “*Missing permissions*”.

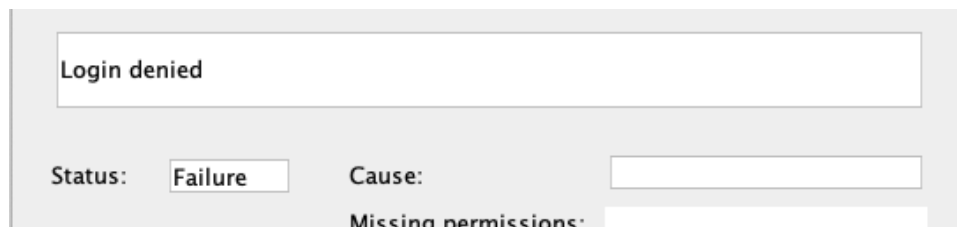
A continuación, se muestra una captura con una sentencia ejecutada correctamente:

Para comenzar a utilizar el cliente *MySQL*, introducir la dirección IP del servidor y el puerto 3306 (puerto por defecto de *MySQL*) en los campos superiores, y pulsar el botón “*Login*”:

Una vez pulsado *Login*, introducir el nombre de usuario y contraseña con el que nos queremos conectar en las cajas de diálogo. Para este primer intento usaremos “*dummy*” y “*foobar2020*” como credenciales:



Al realizar esto, el cliente nos devolverá un error:



Aunque las credenciales son correctas, el cliente no ha conseguido autenticarse contra el servidor. Para encontrar la causa, el primer paso es verificar que nos estamos conectando al puerto adecuado.

En una sesión SSH con el servidor, verificar que el puerto 3306 está a la escucha:

```
:~$ netstat -tln
```

```
ubuntu@ip-172-31-31-78:~$ netstat -tln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
tcp6       0      0 :::22                  :::*                     LISTEN
tcp6       0      0 :::33060                :::*                     LISTEN
tcp6       0      0 :::3306                 :::*                     LISTEN
tcp6       0      0 :::80                   :::*                     LISTEN
```

Si el comando anterior devuelve un error, es posible que haya que instalar el paquete “net-tools” para que el comando “netstat” esté disponible. Instalar con el siguiente comando de apt-get y volver a ejecutar

netstat:

```
:~$ sudo apt-get install net-tools
```

Acceder a la consola de MySQL en el servidor como el usuario “root” y obtener el puerto en el que está configurado:

```
mysql> show global variables like 'port';
```

```
ubuntu@ip-172-31-31-78:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.22-0ubuntu0.20.04.3 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show global variables like 'port';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| port          | 3306  |
+-----+-----+
1 row in set (0.01 sec)
```

Aparentemente, MySQL está escuchando en el puerto por defecto, pero el cliente no es capaz de conectarse. En este caso, el error sucede porque MySQL, por defecto, está configurado para aceptar conexiones locales en el servidor. El resto de conexiones se rechazan.

Para resolver este problema, volvemos a editar el fichero de configuración de *MySQL* y comentamos las líneas “*bind-address = 127.0.0.1*” y “*mysqlx-bind-address = 127.0.0.1*” utilizando el símbolo almohadilla (#). Tras guardar los cambios, reiniciamos el servicio de *MySQL* para que surtan efecto y que *MySQL* acepte conexiones remotas.

```

GNU nano 4.8 /etc/mysql/mysql.conf.d/mysqld.cnf
#
user            = mysql
# pid-file      = /var/run/mysqld/mysqld.pid
# socket       = /var/run/mysqld/mysqld.sock
# port         = 3306
# datadir      = /var/lib/mysql

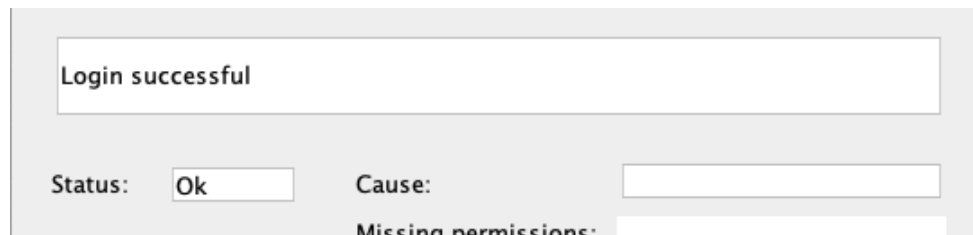
# If MySQL is running as a replication slave, this should be
# changed. Ref https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_tmpdir
# tmpdir       = /tmp
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
#bind-address  = 127.0.0.1
#mysqlx-bind-address = 127.0.0.1
#
# * Fine Tuning
#
key_buffer_size      = 16M
# max_allowed_packet = 64M
# thread_stack       = 256K

# thread_cache_size  = -1

^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line  M-E Redo

```

Una vez hecho este cambio, cliente debería hacer login correctamente con el usuario “dummy” y la contraseña “foobar2020”.

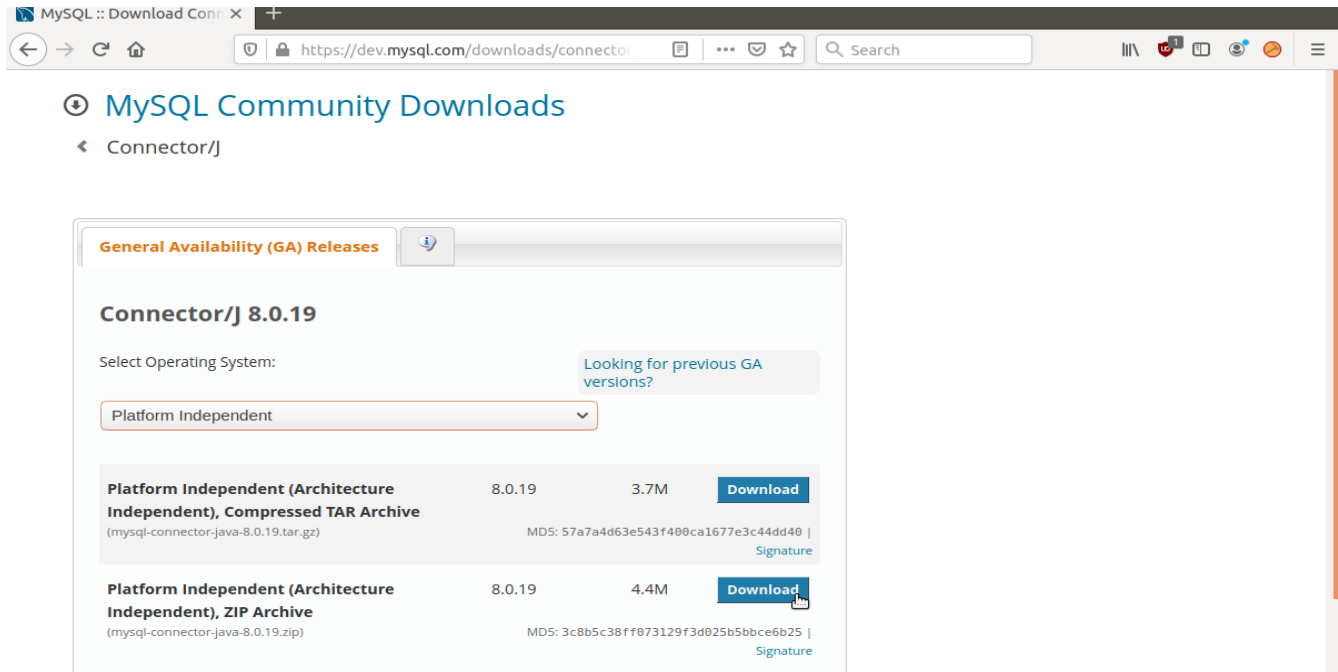


Con esta configuración es suficiente para completar el laboratorio utilizando el cliente Java. Sin embargo, si algún estudiante desea modificar el cliente Java para incorporar otras funcionalidades, el código fuente se incluye en la carpeta “src” del repositorio.

Para poder ejecutar el código fuente correctamente, será necesario utilizar un entorno de desarrollo para Java con el framework JDK y el conector de MySQL para Java. Si estás utilizando *Java EE (Java Enterprise Edition)*, el conector ya debería venir cargado por defecto. Sin embargo, si estás usando *Java*

SE (Java Standard Edition), tendrás que descargarlo de la siguiente dirección (elegir la distribución “Platform independent” en el desplegable):

<https://dev.mysql.com/downloads/connector/j/>



Se recomienda utilizar un editor de código como IntelliJ Idea o Eclipse para hacer modificaciones en el código fuente. Una vez descargado, el conector debe añadirse al proyecto del editor que estéis utilizando para modificar el código.

4 Ejercicio (ENTREGA)

A continuación, vamos a crear las bases de datos con las que trabajaremos en este laboratorio. Como trabajaremos en grupos de 2 personas, cada uno creará una base de datos distinta: “*DBCoin*” y “*DBet*”.

Una vez que nos hayamos repartido las bases de datos, cada uno creará su BD en el servidor de su compañero o compañera de grupo. Para esto, inicialmente cada administrador de bases de datos (usando su usuario “*root*”) creará un usuario (de nombre “*admDBCoin*” y “*admDBet*” respectivamente) que inicialmente carece de permisos.

Recordad, los permisos que un usuario tiene asignado actualmente se pueden mediante el comando *show grants*

Estos nuevos usuarios, a los que genéricamente llamaremos “*adm?*”, serán los que habrá que utilizar a partir de ahora para realizar todas las tareas en las bases de datos que crearemos.

No se permite usar *phpMyAdmin*: si lo utilizas, tendrás un 0 como calificación en este laboratorio

Para comenzar, inicia una sesión remota mediante *SSH* en el servidor de tu compañer@ y apunta cada sentencia *SQL* que utilices en una hoja nueva de un editor de texto plano (*gedit*, por ejemplo), indicando el usuario que ejecuta cada sentencia.

La forma de trabajar que utilizaremos se describe a continuación. Por cada operación a realizar:

1. Especifica la sentencia SQL que el administrador de la base de datos (es decir, “*adm?*”) tiene que utilizar y prueba a ejecutarla.
2. Si el usuario “*adm?*” ya tiene permisos para realizar la operación, continúa con la siguiente operación.
3. Si el usuario “*adm?*” no tiene los permisos necesarios, solicita a tu compañero o compañera que le asigne el permiso que le hace falta (pero ninguno más) a “*adm?*” y vuelve a ejecutar la sentencia SQL. Para esto, tu compañero o compañera tendrá que utilizar su usuario “*root*”, como antes. Este paso se repite hasta que el usuario “*adm?*” tenga los permisos necesarios.

Después de asignar un nuevo permiso, es conveniente ejecutar el comando:

```
mysql> flush privileges;
```

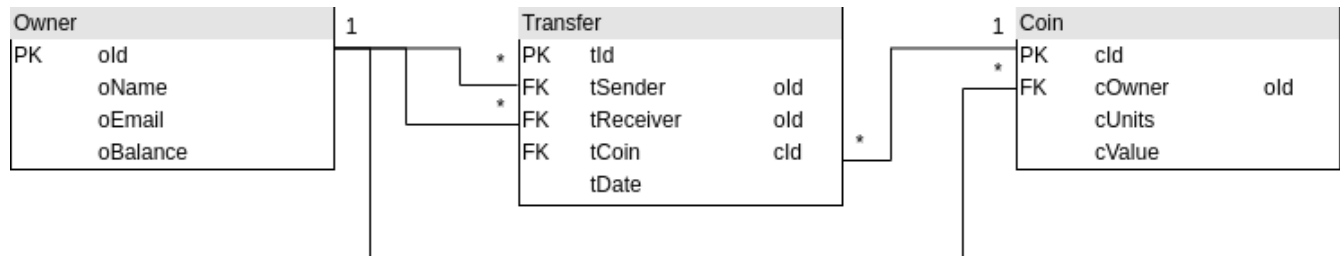
Para ejecutar este comando es necesario poseer el privilegio “*reload*”. Este comando lo puede ejecutar quien asigna el nuevo permiso (por ejemplo, “*root*”), pero también es conveniente que lo ejecute quien recibe el nuevo permiso (por ejemplo, “*adm?*”). Por lo tanto, es recomendable asignar el privilegio “*reload*” a “*adm?*” antes de comenzar a realizar las acciones. En cualquier caso, es posible que en alguna ocasión los permisos de un usuario no se actualicen hasta que no se inicie una nueva sesión de *MySQL*. Por este motivo, si después de asignar un nuevo permiso el usuario “*adm?*” sigue sin poder hacer uso de él, prueba a cerrar y abrir sesión de nuevo en el cliente de *MySQL*.

Para agilizar el trabajo se recomienda que trabajéis en paralelo utilizando 2 ventanas de terminal distintas. Una sugerencia para no confundirse con las ventanas de terminal es cambiarles el título, las indicaciones para ello se incluyen en forma de anexo al final de este documento.

Al finalizar el laboratorio, se debe subir a *eGela* la hoja de texto con las sentencias SQL ejecutadas.

A continuación, se enumeran las operaciones a realizar para crear cada base de datos.

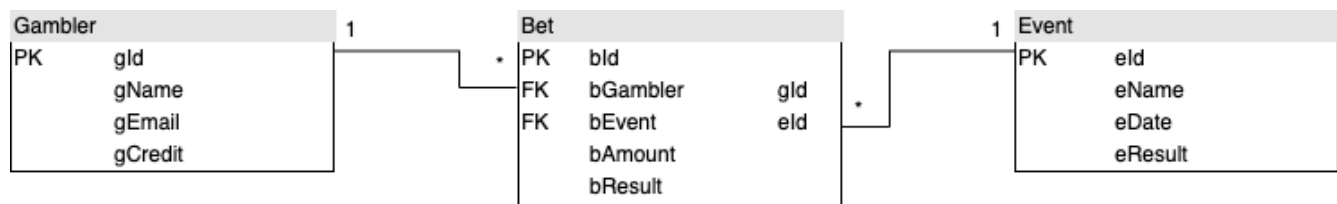
A) Base de datos *DBcoin*



Utilizando el usuario “*admDBcoin*”:

1. Crear la tabla “Transfer” (sin definir las claves extranjeras).
2. Crear las tablas “Owner” y “Coin”.
3. Crear las claves extranjeras de la tabla “Transfer”.
4. Introducir un dueño de nombre “Ripple” en la tabla “Owner”.
5. Introducir otro cliente de nombre “Ethereum” en la tabla “Owner”.
6. Introducir una moneda cuyo dueño sea “Ripple” en la tabla “Coin”.
7. Introducir otra moneda cuyo dueño sea “Ethereum” en la tabla “Coin”.
8. Introducir una transferencia de la primera moneda (creada en el punto 6) realizada el día 1 de febrero de 2021 donde el emisor sea “Ethereum” y el receptor sea “Ripple”.
9. Introducir otra transferencia de la segunda moneda (creada en el punto 7) realizada el día 2 de febrero de 2021 donde el emisor sea “Ripple” y el receptor sea “Ethereum”.
10. Modificar la fecha de la transferencia introducida en el punto anterior al 3 de febrero de 2021.
11. Crear los usuarios MySQL “MIser1” y “MIser2”.
12. Asignar a los nuevos usuarios los siguientes permisos:

DBCoin										
User	Object	Privilege	Attributes							
MIser1	Owner	Select	:	old	,	oName	,	oEmail		
		Insert	:	old	,			oEmail	,	oBalance
		Update	:	old	,			oEmail	,	oBalance
		Delete								
	Transfer	Select	:	tId	,			tReceiver	,	tCoin
		Insert	:	tId	,			tReceiver	,	tCoin
		Update	:			tSender	,	tReceiver	,	tDate
	Coin	Select	(all)							
		Insert	(all)							
		Update	:	cId	,	cOwner	,	cUnits		
		Delete								
	Unspecified	Consultar el identificador y el nombre de los dueños con balance negativo.								
		Consultar el identificador de una transferencia y los nombres del emisor y del receptor.								
		Dado el identificador de un dueño, modificar su balance con un límite del 10%.								
		Dado el identificador de un dueño, modificar su balance.								
MIser2	Owner	Select	:	old	,			oEmail	,	oBalance
		Insert	:	old	,	oName				
		Update	:	old	,	oName				
		Delete								
	Transfer	Select	:	tId	,	tSender	,	tReceiver	,	tDate
		Insert	:	tId	,	tSender	,	tReceiver		
		Update	(all)							
	Coin	Select	:					cUnits	,	cValue
		Insert	:			cOwner	,			cValue
		Update	:					cUnits	,	cValue
		Delete								
	Unspecified	Consultar el identificador, el dueño y el valor de una moneda.								
		Consultar el identificador, el emisor y el receptor de las transferencias del año 2020.								
		Dado el identificador de una moneda, modificar su número de unidades.								
		Dado el identificador de una moneda, modificar el número de unidades de aquellas cuyo valor es menor que 5.								

B) Base de datos *DBet*

Utilizando el usuario “*admDBet*”:

1. Crear la tabla “Bet” (sin definir las claves extranjeras).
2. Crear las tablas “Gambler” y “Event”.
3. Crear las claves extranjeras de la tabla “Bet”.
4. Introducir un apostante de nombre “MacBeth” en la tabla “Gambler”.
5. Introducir otro apostante de nombre “Betty” en la tabla “Gambler”.
6. Introducir un evento de nombre “SnailRace” y fecha 1 de febrero de 2021 en la tabla “Event”.
7. Introducir otro evento de nombre “CaterpillarFight” y fecha 2 de febrero de 2021 en la tabla “Event”.
8. Introducir una apuesta de “MacBeth” para el evento “SnailRace” donde el resultado sea “Turbo wins”.
9. Introducir una apuesta de “Betty” para el event “CaterpillarFight” donde el resultado sea “Butterfly loses”.
10. Modificar la fecha del evento “SnailRace” al 3 de febrero de 2021.
11. Crear los usuarios MySQL “LUser1” y “LUser2”.
12. Asignar a los nuevos usuarios los permisos que se indican en la siguiente tabla:

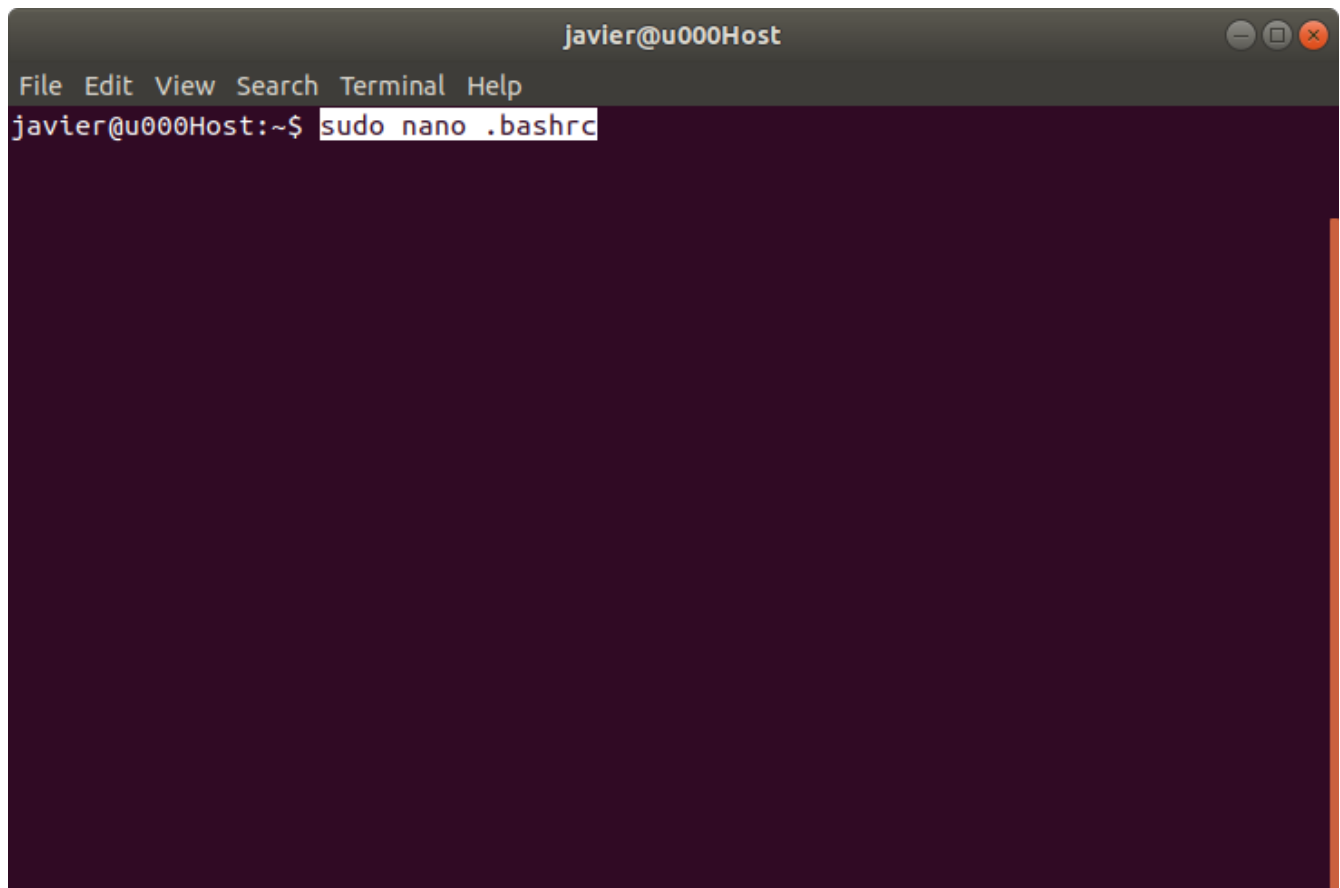
DBet									
User	Object	Privilege	Attributes						
LUser1	Gambler	Select	:			gEmail			
		Insert	:		gName	,		gCredit	
		Update	:		gName	,	gEmail	,	gCredit
		Delete							
	Bet	Select	:	bld	,	bGambler	,	bAmount	bResult
		Insert	(all)						
		Update	(all)						
		Delete							
	Event	Select	:	eld	,		eDate		
		Insert	:	eld	,	eName	,	eDate	
		Update	:	eld	,	eName			
		Delete							
	Unspecified	Consultar el identificador y el crédito de los apostadores							
		Consultar el identificador y el evento de las apuestas ganadoras.							
		Dado el identificador de un evento, actualizar la fecha de los eventos que aún no se hayan celebrado.							
		Dado el identificador de una apuesta, actualizar su importe si el evento aún no se han disputado.							
LUser2	Gambler	Select	:	gld	,			gCredit	
		Insert	(all)						
		Update	:	gld	,	gName	,	gCredit	
		Delete							
	Bet	Select	:		bGambler	,	bEvent	,	bResult
		Insert	:		bGambler	,	bEvent	,	bAmount
		Update	:	bld	,	bGambler	,	bEvent	
		Delete							
	Event	Select	:		eName	,	eDate		
		Insert	:	eld	,	eName	,	eDate	
		Update	(all)						
		Delete							
	Unspecified	Consultar el identificador y el nombre de los apostadores que hayan apostado durante el año 2021.							
		Consultar el identificador, el evento y el importe de las apuestas de hoy.							
		Dado el identificador de una apuesta, doblar su importe si el apostador tiene crédito suficiente.							
		Dado el identificador de una apuesta, actualizar el importe.							

5 ANEXO: Cambiar el título a la terminal

Para agilizar el trabajo, se recomienda que trabajéis en paralelo utilizando 2 ventanas de terminal distintas. Una sugerencia para no confundirse con las ventanas de terminal es cambiarles el título, de tal forma que el título de la ventana nos indique el usuario que estamos utilizando en el cliente de *MySQL*.

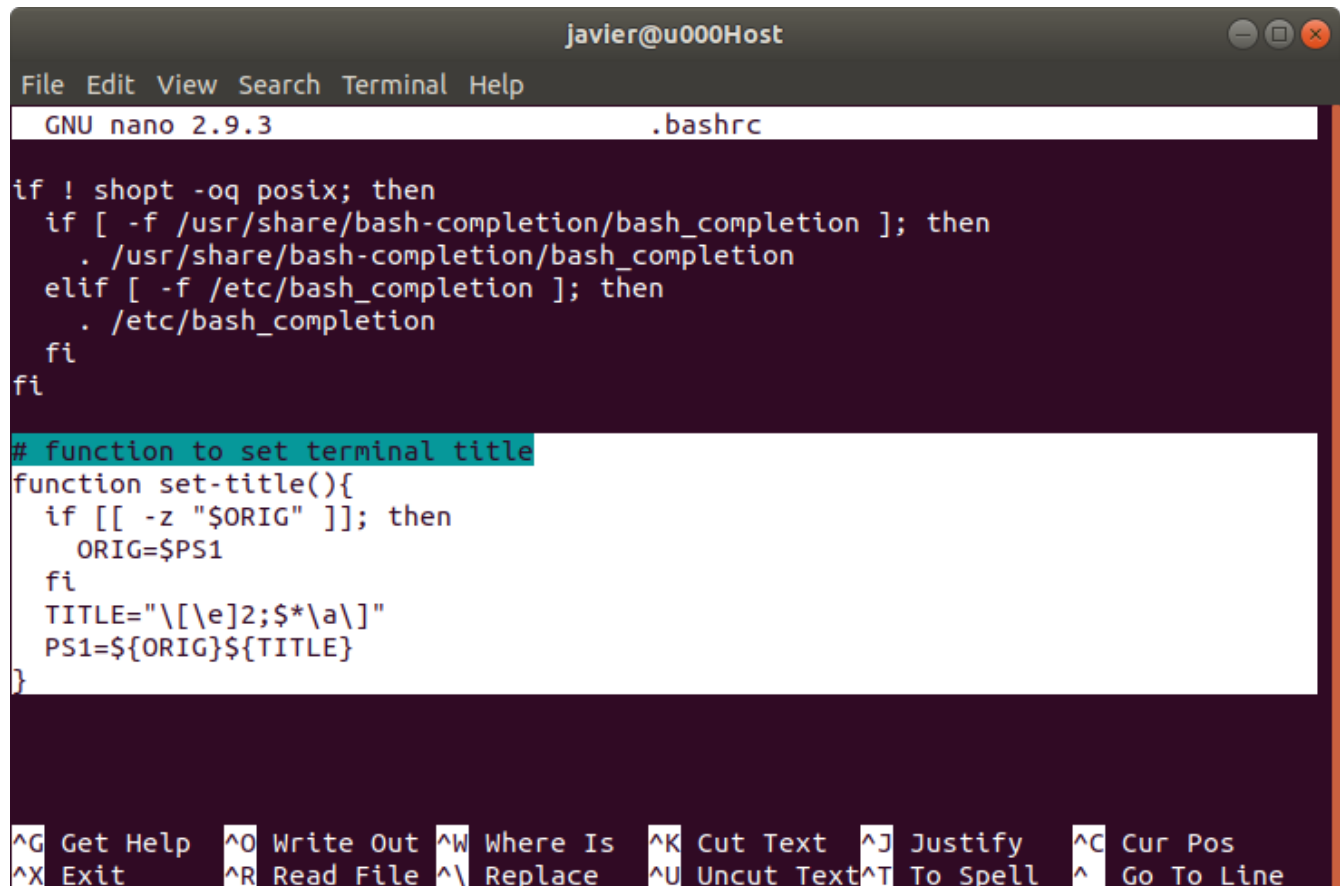
El primer paso es editar el archivo “*.bashrc*” de nuestra carpeta de usuario en el equipo local.

```
host:~$ sudo nano .bashrc
```



Al final del archivo, introduce la siguiente función:

```
# function to set terminal title
function set-title(){
    if [[ -z "$ORIG" ]]; then
        ORIG=$PS1
    fi
    TITLE="\[\e]2;$*\a\"
    PS1=${ORIG}${TITLE}
}
```



The screenshot shows a terminal window titled "javier@u000Host" with a menu bar (File, Edit, View, Search, Terminal, Help) and a status bar (GNU nano 2.9.3, .bashrc). The editor content shows the end of the .bashrc file with the following code:

```
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

# function to set terminal title
function set-title(){
  if [[ -z "$ORIG" ]]; then
    ORIG=$PS1
  fi
  TITLE="\[\e]2;$*\a\"
  PS1=${ORIG}${TITLE}
}
```

The status bar at the bottom displays various keyboard shortcuts: ^G Get Help, ^O Write Out, ^W Where Is, ^K Cut Text, ^J Justify, ^C Cur Pos, ^X Exit, ^R Read File, ^\ Replace, ^U Uncut Text, ^T To Spell, and ^_ Go To Line.

Para que los cambios tengan efecto, hay que cerrar la ventana de terminal y abrir una nueva. En la nueva ventana de terminal, ya podemos cambiar el título utilizando la función anteriormente definida de la siguiente forma (reemplazar <terminal-title> por el nombre que queráis en los comandos):

```
host:~$ set-title "<username>@<terminal-title>"
host:~$ PS1="\u@<terminal-title>:\w\$"
```

