Review the database operations

Welcome to this Lab activity

In this lab activity you will be reviewing the database operations you encountered in your previous lab activity. For the purpose of this lab you will be working with the Terminal panel inside Visual Studio Code.

Start the Lab environment application

It is simple to launch a lab exercise. You only need to click on the button “Start” below the activity title to enter a lab environment.

Let’s explore this lab activity. Go ahead and click on the “Start” button!

Review the database operations

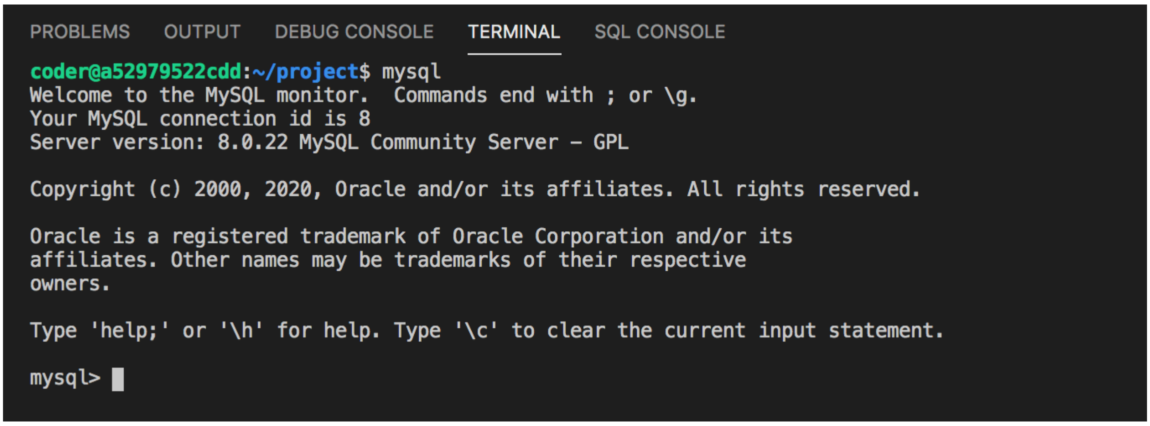
Task 1: Accessing the MySQL interactive shell

The folder structure has already been partially constructed for you and organised into different topics. For the purpose of this lab, you are not required to make any changes to the folder structure. You can see a folder called ‘topic6’ inside this lab environment; it is only there as a reference for you and you are not required to add any content to it at the moment. Let’s get started!

In order to access your mysql interactive shell use the Visual Studio Code Terminal and run the following command:

- **mysql**: type this command and press *Enter*. This command will log you into mysql shell as the root(default) user.

If you have successfully followed all the above steps you should now be logged in inside mysql and see the following result on the Terminal:



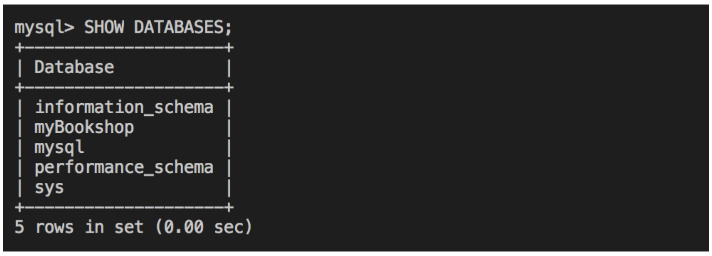
```
coder@a52979522cdd:~/project$ mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.22 MySQL Community Server - GPL

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> 
```

Task 2: Creating a new database

Before creating a new database, let's check which databases are already linked to your virtual server.

Use the Visual Studio Code Terminal and run the following command:

- **SHOW DATABASES;** type this command and press *Enter*. This command will show you all the databases that already exist in your virtual server.



```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| myBookshop         |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

Let's create a brand new database which will simulate a restaurant menu.

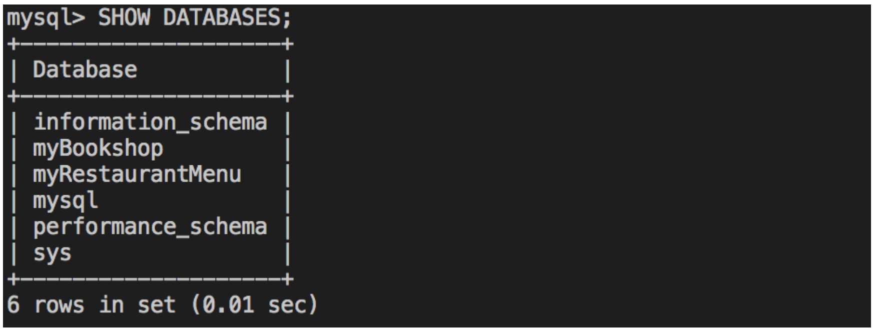
Use the Visual Studio Code Terminal and run the following command:

- **CREATE DATABASE myRestaurantMenu;** type this command and press *Enter*. This command will create a new database called "myRestaurantMenu" in your virtual server.

Please note: names are lower and upper case-sensitive; “myRestaurantMenu” is different from “myrestaurantmenu”! Please use the same names as instructed on the lab instructions to avoid any confusion. Commands, on the other hand, are not case-sensitive; “CREATE” is the same as “create”.

Now that you have created your new database, you can view it by running the following command once again:

- **SHOW DATABASES;**



```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| myBookshop         |
| myRestaurantMenu   |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
6 rows in set (0.01 sec)
```

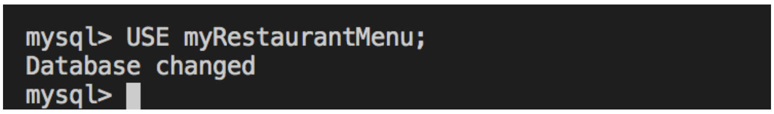
As you can see the “myRestaurantMenu” database is now part of your databases list.

Task 3: Creating a table on your database

It is now time to add a table to your newly created database. The very first thing you need to do is to select your “myRestaurantMenu” database. Use the Visual Studio Code Terminal and run the following command:

- **USE myRestaurantMenu;** type this command and press *Enter*. This command will switch and select the “myRestaurantMenu” database.

If you have successfully selected the database you will get the following confirmation:

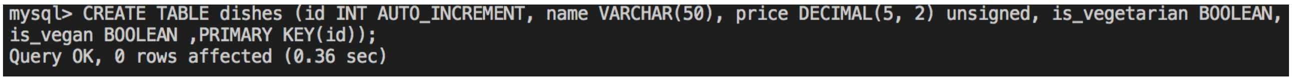


```
mysql> USE myRestaurantMenu;
Database changed
mysql>
```

Now that you have correctly selected your “myRestaurantMenu” database, go ahead and create a table name “dishes” by running the following command on the Terminal:

- **CREATE TABLE dishes (id INT AUTO_INCREMENT, name VARCHAR(50), price DECIMAL(5, 2) unsigned, is_vegetarian BOOLEAN, is_vegan BOOLEAN ,PRIMARY KEY(id));** type this command and press *Enter*. This command will create a new table called “dishes” in your currently selected database.

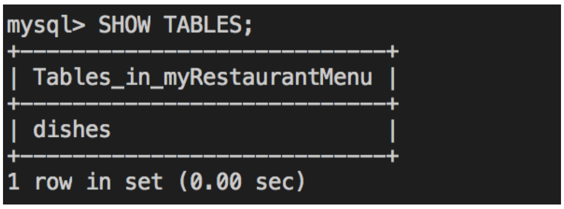
If you have successfully created a new table you will get the following confirmation:



```
mysql> CREATE TABLE dishes (id INT AUTO_INCREMENT, name VARCHAR(50), price DECIMAL(5, 2) unsigned, is_vegetarian BOOLEAN,
is_vegan BOOLEAN ,PRIMARY KEY(id));
Query OK, 0 rows affected (0.36 sec)
```

You can now view your newly created table by running the following command:

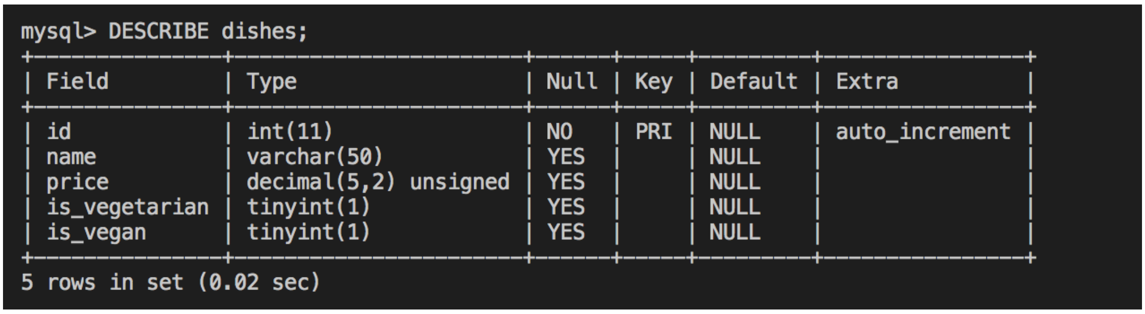
- **SHOW TABLES;** type this command and press *Enter*. This command will show you all the tables related to your currently selected database.



```
mysql> SHOW TABLES;
+----------------------------+
| Tables_in_myRestaurantMenu |
+----------------------------+
| dishes                     |
+----------------------------+
1 row in set (0.00 sec)
```

Finally let’s view the content of the “dishes” table. Run the following command:

- **DESCRIBE dishes;** type this command and press *Enter*. This command will show you all the tables related to your currently selected database.



```
mysql> DESCRIBE dishes;
+---------------+-----------------------+------+-----+---------+----------------+
| Field         | Type                  | Null | Key | Default | Extra          |
+---------------+-----------------------+------+-----+---------+----------------+
| id            | int(11)               | NO   | PRI | NULL    | auto_increment |
| name          | varchar(50)           | YES  |     | NULL    |                |
| price         | decimal(5,2) unsigned | YES  |     | NULL    |                |
| is_vegetarian | tinyint(1)            | YES  |     | NULL    |                |
| is_vegan      | tinyint(1)            | YES  |     | NULL    |                |
+---------------+-----------------------+------+-----+---------+----------------+
5 rows in set (0.02 sec)
```

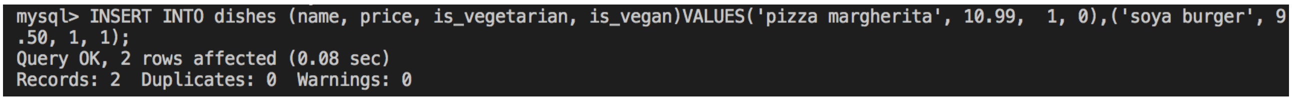
You can see from the above picture that you have successfully created a new table called “dishes” with 5 fields (id, name, price, is_vegetarian, is_vegan) in your “myRestaurantMenu” database.

Task 4: Adding dummy data to your database

Now that you have correctly created the “dishes” table in your database, it is time for you to add some data to it. Run the following command in the Terminal:

- **INSERT INTO TableName (name, price, is_vegetarian, is_vegan)VALUES('pizza margherita', 10.99, 1, 0),('soya burger', 9.50, 1, 1);** replace the TableName value with the name of your table, type this command and press *Enter*. This command will insert two new books in your database.

If you have successfully run the above command you will get the following confirmation message:



```
mysql> INSERT INTO dishes (name, price, is_vegetarian, is_vegan)VALUES('pizza margherita', 10.99,  1, 0),('soya burger', 9
.50, 1, 1);
Query OK, 2 rows affected (0.08 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

You have successfully added two dishes you your “myRestaurantMenu” database under the “dishes” table.

Task 5: Query the data in mysql shell

Now that you have successfully created a new database and corresponding table, you are able to perform SQL queries on it. In order to read data from a database with SQL, you need to use the SELECT statement.

The basic format of the SELECT statement looks like below:

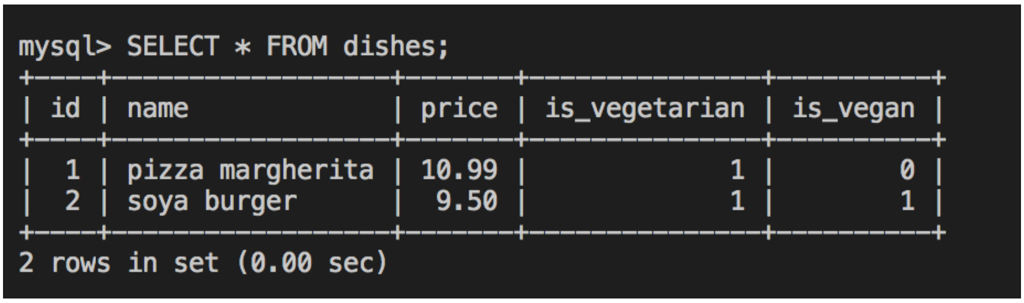
SELECT filedName1, fieldName2 FROM TableName;

filedName1 and fieldName2 are the field names that you want to retrieve and TableName is the name of the table where you want to retrieve the data from.

You may also use the wildcard (*) to return all the fields in a table:

SELECT * FROM TableName;

Try to run the above command in the Terminal, you should get the following result:



```
mysql> SELECT * FROM dishes;
+----+------------------+-------+---------------+----------+
| id | name             | price | is_vegetarian | is_vegan |
+----+------------------+-------+---------------+----------+
|  1 | pizza margherita | 10.99 |             1 |        0 |
|  2 | soya burger      |  9.50 |             1 |        1 |
+----+------------------+-------+---------------+----------+
2 rows in set (0.00 sec)
```

Remember to change TableName with the name of your table. Also the *is_vegetarian* and *is_vegan* fields are represented by a boolean value. 1 means "yes" and 0 mean "no".

Now try running the following command:

SELECT * FROM TableName LIMIT 1;

Here, the asterisk (*) is a wildcard, which means "all". Including the LIMIT clause restricts how many records (rows) are returned in the result-set.

SELECT can be combined with various clauses to restrict/filter the records returned in the result-set. For example, the WHERE clause can be used to return only records matching a specific criteria.

Try running the following query:

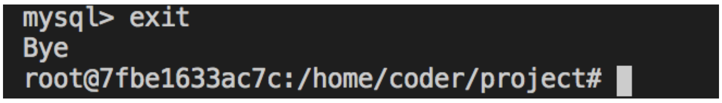
SELECT name, price FROM dishes WHERE id=2;

Task 4: Exit mysql shell

Exiting the mysql shell is very straight forward. In your Terminal panel type the following command:

- **exit**: type this command and press *Enter*. This command will log you out from your mysql virtual server.

If you have successfully exited the database you will get the following confirmation:



```
mysql> exit
Bye
root@7fbe1633ac7c:/home/coder/project#
```

End of Section

Congratulations for completing this section. As long as you have saved your work, your files will remain when you close this lab activity so do not worry about losing your data. This was just a review of the main SLQ operations that you have learned so far. In the next lab activity you will be exploring how to update and delete your database.