Hands-on Lab: Built-in Functions

Estimated time needed: 20 minutes

In SQL, you can access many built-in functions that may be used to get more variety in our data analysis. These functions include aggregation functions (like `MAX`, `MIN`, `SUM`, and `AVG`), string functions (like `LENGTH`, `UCASE`, and `LCASE`), scalar functions (like `ROUND`), and a variety of date functions as well. In this tab, you'll get hands-on practice on how to use all of them.

Software Used in this Lab

In this lab, you will use MySQL. MySQL is a Relational Database Management System (RDBMS) designed to store, manipulate, and retrieve data efficiently.

To complete this lab, you will use MySQL relational database service available as part of IBM Skills Network Labs (SN Labs) Cloud IDE. SN Labs is a virtual lab environment used in this course.

Objectives

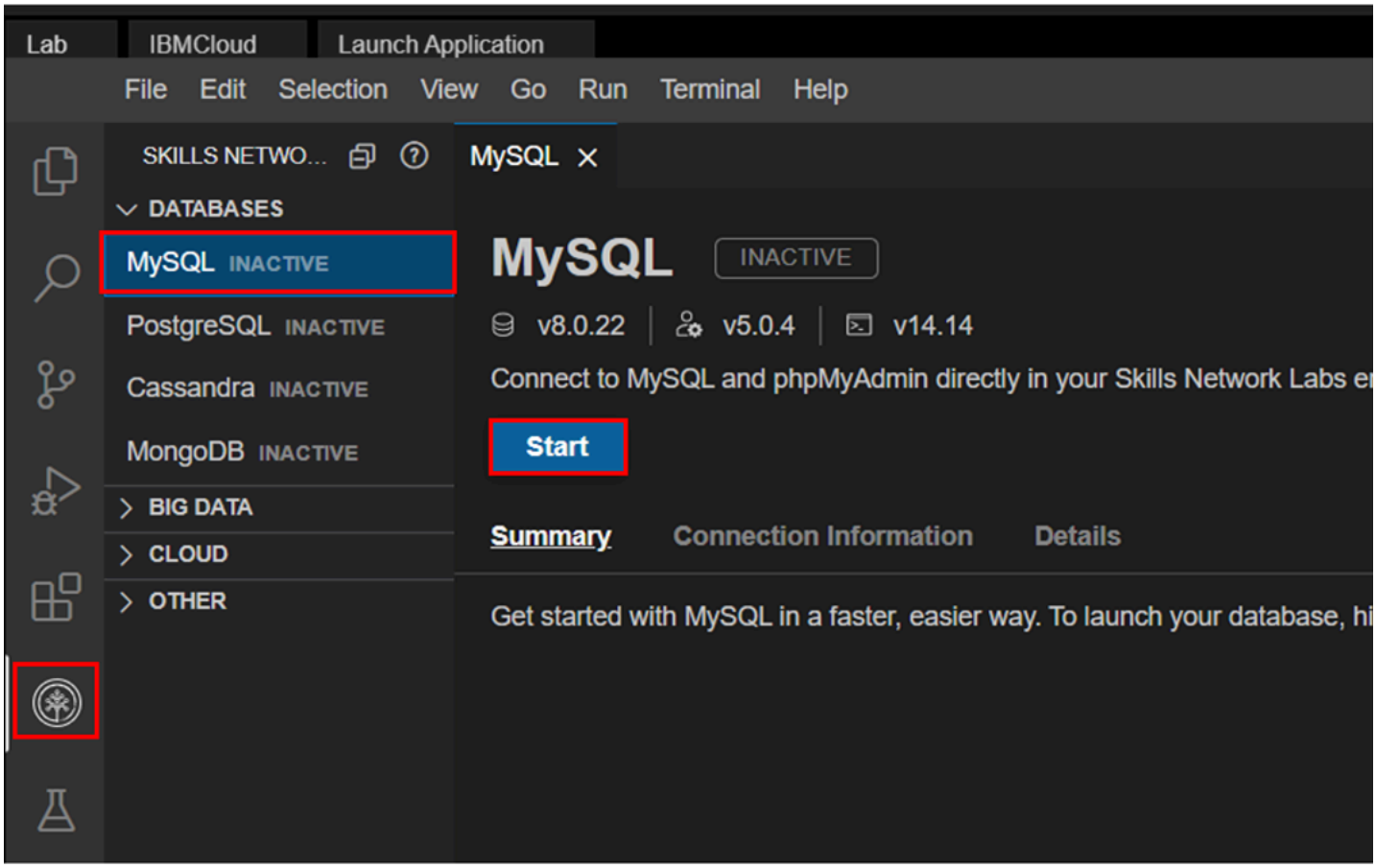
After completing this lab, you will be able to compose queries in phpMyAdmin with MySQL using:

- Aggregation Functions
- Scalar Functions
- String Functions
- Date Functions

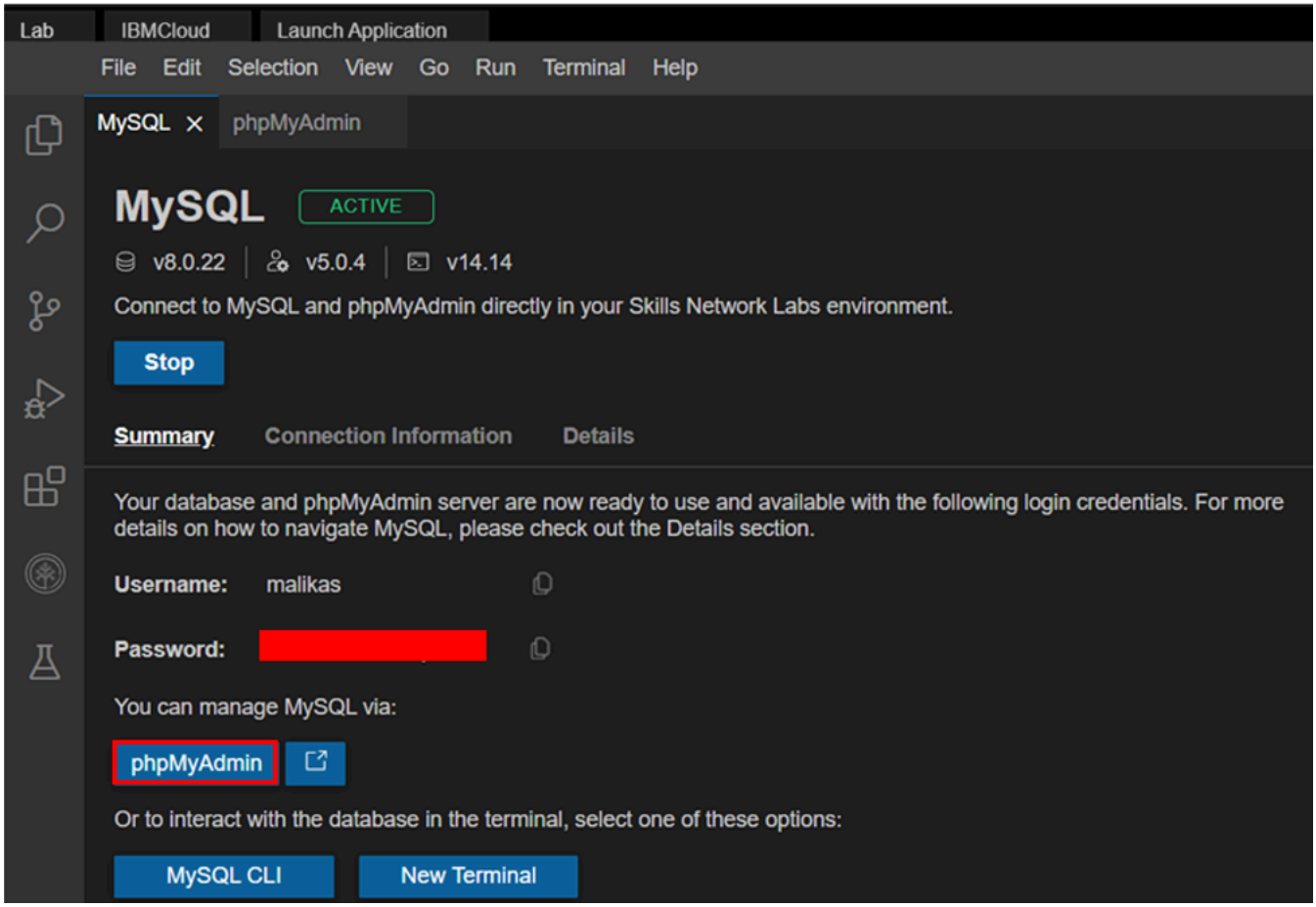
Create the database

Click on **Skills Network Toolbox** in the programming interface ribbon. In the **Database** section, click **MySQL**.

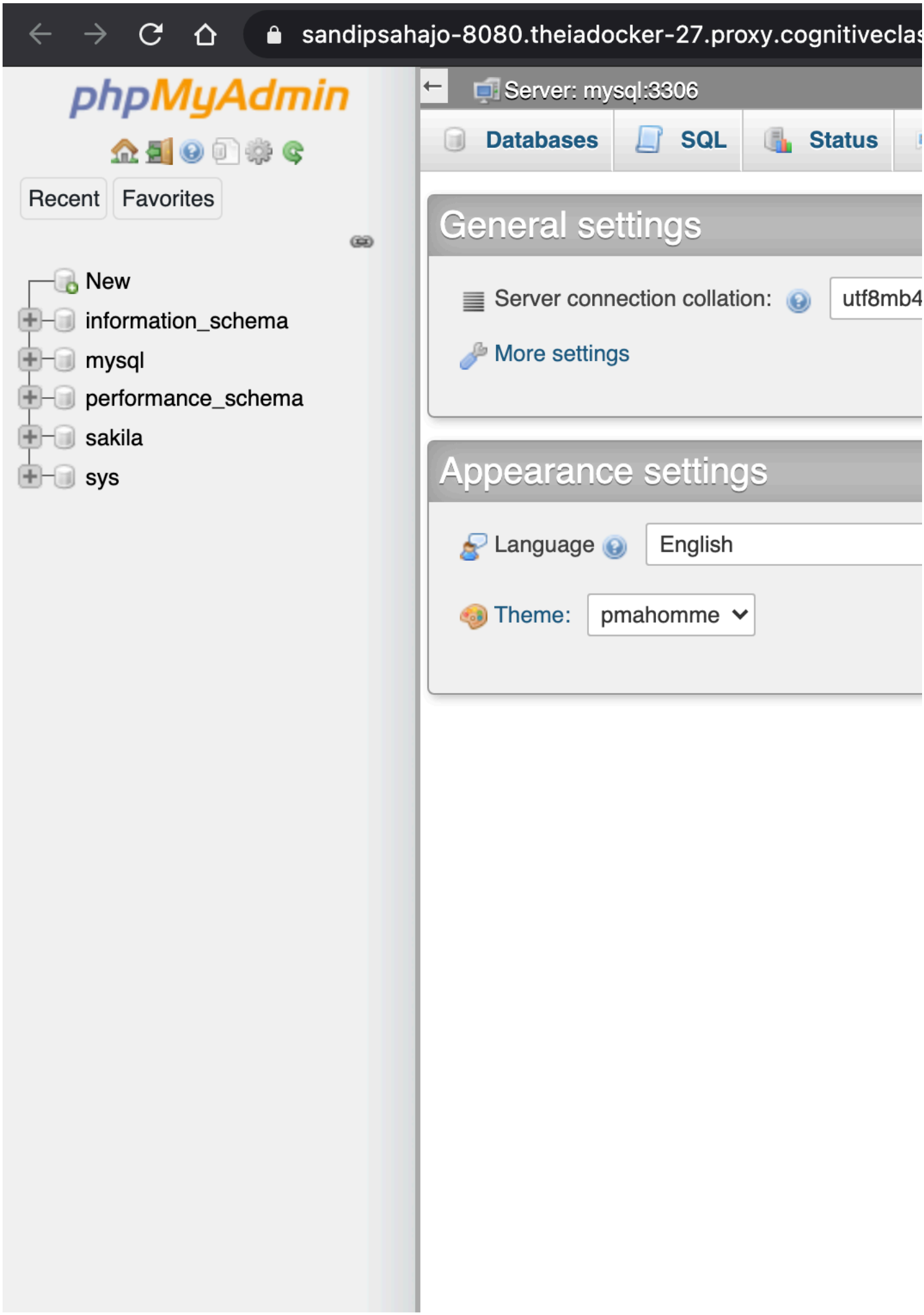
To start the MySQL engine, click **Start**.



Once **MySQL** has started, click the **phpMyAdmin button** to open **phpMyAdmin** in the same window.

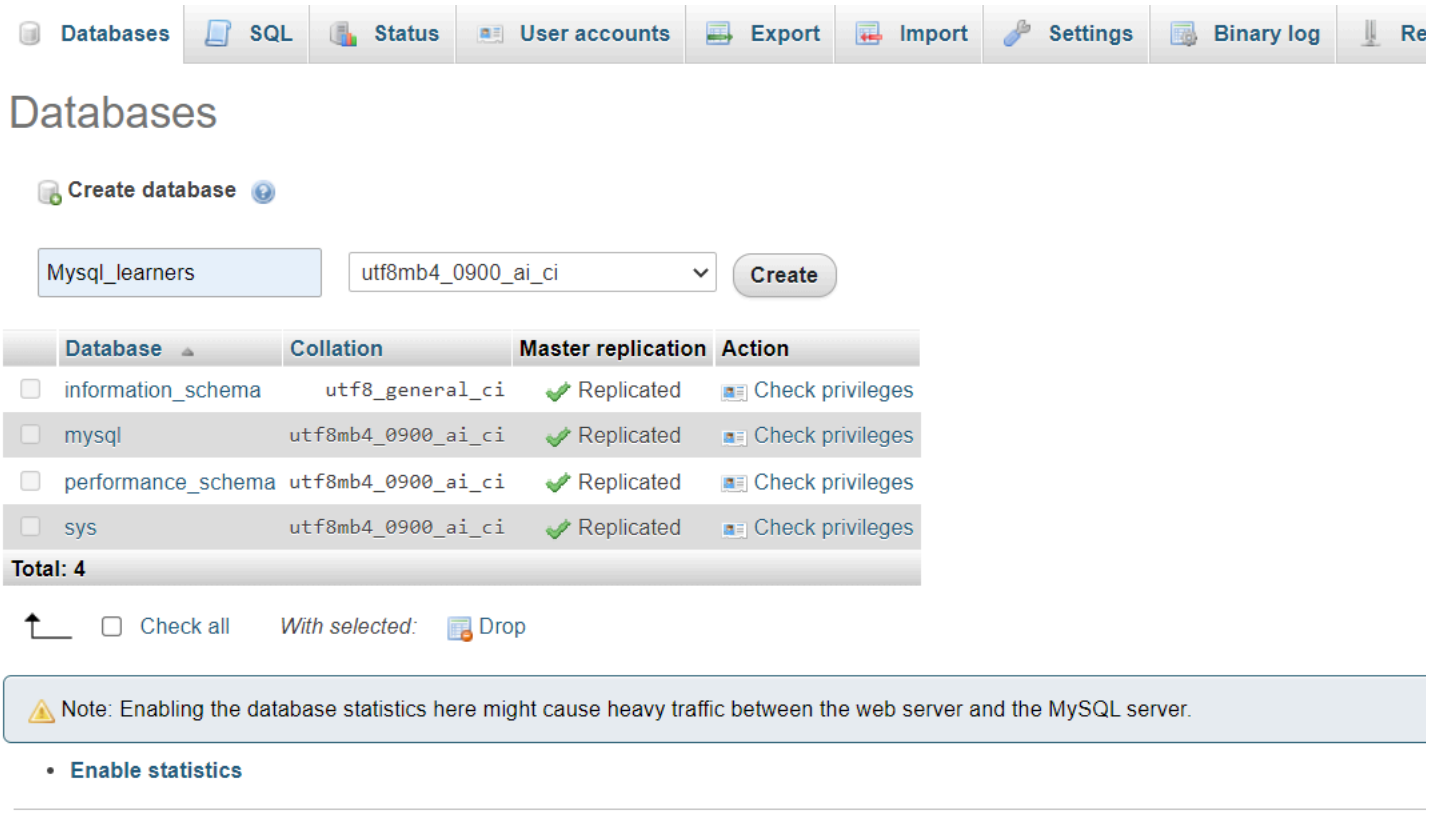


You will see the phpMyAdmin GUI tool.



In the tree view, click **New** to create a new empty database. Then, enter **Mysql_Learners** as the name of the database and click **Create**.

Leave the default **utf8** encoding. UTF-8 is the most commonly used character encoding for content or data.



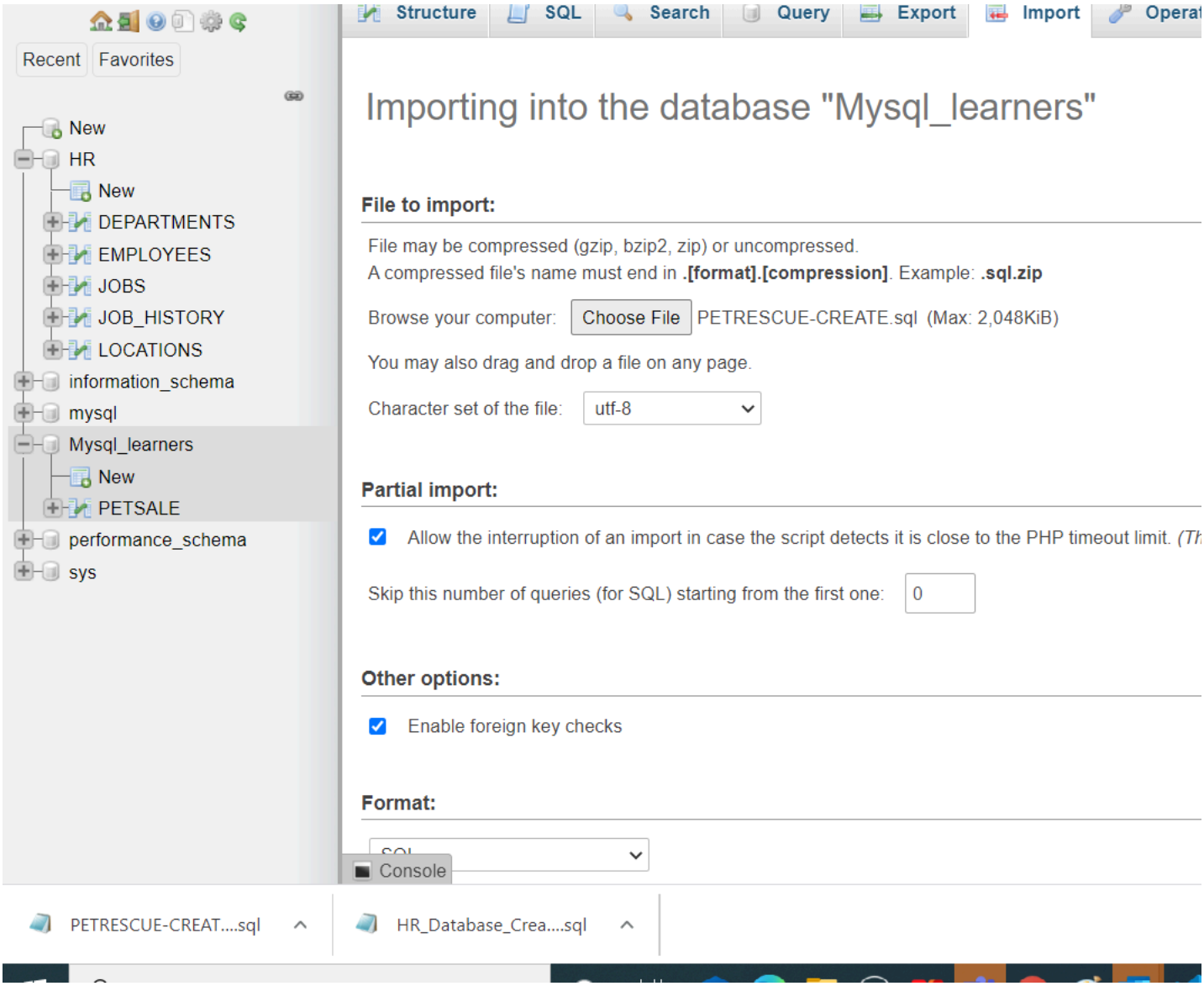
Create the PETRESCUE table

Rather than create the table manually by typing the DDL commands in the SQL editor, you will execute a script containing the create table command.

Download the script file PETRESCUE-CREATE.sql

> **Note**: To download, right-click on the link above and click on **Save As** or **Save Link As** depending on your browser. Remember to save the file as a .sql file and not HTML.

Next, load the `.sql` file to your database using the `Import` option as shown in the image below.



Upon execution, the table PETRESCUE will be created in the `Mysql_Learners` database and loaded with a set of values as well. The attributes of the PETRESCUE table are:

Column Name	Data Type	Description
ID	INTEGER	ID of the entry
ANIMAL	VARCHAR(20)	Type of animal
QUANTITY	INTEGER	Number of animals
COST	DECIMAL(6,2)	Cost incurred
RESCUEDATE	DATE	Date of Rescue

Once the table is loaded, you may open the sql editor to start executing the queries.

Aggregation Functions

1. Write a query that calculates the total cost of all animal rescues in the PETRESCUE table.

For this query, we will use the function `SUM(COLUMN_NAME)`. The output of this query will be the total value of all elements in the column. The query for this question can be written as:

```
SELECT SUM(COST) FROM PETRESCUE;
```

You can further assign a label to the query `SUM_OF_COST`.

```
SELECT SUM(COST) AS SUM_OF_COST FROM PETRESCUE;
```

2. Write a query that displays the maximum quantity of animals rescued (of any kind).

For this query, we will use the function `MAX(COLUMN_NAME)`. The output of this query will be the maximum value of all elements in the column. The query for this question can be written as:

```
SELECT MAX(QUANTITY) FROM PETRESCUE;
```

The query can easily be changed to display the minimum quantity using the `MIN` function instead.

```
SELECT MIN(QUANTITY) FROM PETRESCUE;
```

3. Write a query that displays the average cost of animals rescued.

For this query, we will use `AVG(COLUMN_NAME)`. The output of this query will be the average of all elements in the column. The query for this question can be written as

```
SELECT AVG(COST) FROM PETRESCUE;
```

Scalar Functions and String Functions

1. Write a query that displays the rounded integral cost of each rescue.

For this query, we will use the function `ROUND(COLUMN_NAME, NUMBER_OF_DECIMALS)`. The output of this query will be the value of each element in the column rounded to the specified number of decimal places. Note that the second argument is optional and, if omitted, results in rounding to an integer value. The query for this question can be written as:

```
SELECT ROUND(COST) FROM PETRESCUE;
```

The query could also be written as:

```
SELECT ROUND(COST, 0) FROM PETRESCUE;
```

In case the question was to round the value to 2 decimal places, the query would change to:

```
SELECT ROUND(COST, 2) FROM PETRESCUE;
```

2. Write a query that displays the length of each animal name.

For this query, we will use the function `LENGTH(COLUMN_NAME)`. The output of this query will be the length of each element in the column. The query for this question can be written as:

```
SELECT LENGTH(ANIMAL) FROM PETRESCUE;
```

3. Write a query that displays the animal name in each rescue in uppercase.

For this query, we will use the function `UCASE(COLUMN_NAME)`. The output of this query will be each element in the column in upper case letters. The query for this question can be written as:

```
SELECT UCASE(ANIMAL) FROM PETRESCUE;
```

Just as easily, the user could ask for a lower case representation, and the query would be changed to:

```
SELECT LCASE(ANIMAL) FROM PETRESCUE;
```

Date Functions

1. Write a query that displays the rescue date.

For this query, we will use the function `DAY(COLUMN_NAME)`. The output of this query will be only the `DAY` part of the date in the column. The query for this question can be written as:

```
SELECT DAY(RESCUEDATE) FROM PETRESCUE;
```

In case the query was asking for `MONTH` of rescue, the query would change to:

```
SELECT MONTH(RESCUEDATE) FROM PETRESCUE;
```

In case the query was asking for `YEAR` of rescue, the query would change to:

```
SELECT YEAR(RESCUEDATE) FROM PETRESCUE;
```

2. Animals rescued should see the vet within three days of arrival. Write a query that displays the third day of each rescue.

For this query, we will use the function `DATE_ADD(COLUMN_NAME, INTERVAL Number Date_element)`. Here, the quantity in `Number` and in `Date_element` will combine to form the interval to be added to the date in the column. For the given question, the query would be:

```
SELECT DATE_ADD(RESCUEDATE, INTERVAL 3 DAY) FROM PETRESCUE
```

If the question was to add 2 months to the date, the query would change to:

```
SELECT DATE_ADD(RESCUEDATE, INTERVAL 2 MONTH) FROM PETRESCUE
```

Similarly, we can retrieve a date before the one given in the column by a given number using the function `DATE_SUB`. By modifying the same example, the following query would provide the date 3 days before the rescue.

```
SELECT DATE_SUB(RESCUEDATE, INTERVAL 3 DAY) FROM PETRESCUE
```

3. Write a query that displays the length of time the animals have been rescued, for example, the difference between the current date and the rescue date.

For this query, we will use the function `DATEDIFF(Date_1, Date_2)`. This function calculates the difference between the two given dates and gives the output in number of days. For the given question, the query would be:

```
SELECT DATEDIFF(CURRENT_DATE, RESCUEDATE) FROM PETRESCUE
```

CURRENT_DATE is also an inbuilt function that returns the present date as known to the server.

To present the output in a YYYY-MM-DD format, another function `FROM_DAYS(number_of_days)`can be used. This function takes a number of days and returns the required formatted output. The query above would thus be modified to

```
SELECT FROM_DAYS(DATEDIFF(CURRENT_DATE, RESCUEDATE)) FROM PETRESCUE
```

Practice Problems

1. Write a query that displays the average cost of rescuing a single dog. Note that the cost per dog would not be the same in different instances.

▶ Click here for a hint
▼ Click here for the solution

```
SELECT AVG(COST/QUANTITY) FROM PETRESCUE WHERE ANIMAL = 'Dog';
```

2. Write a query that displays the animal name in each rescue in uppercase without duplications.

▶ Click here for a hint
▼ Click here for the solution

```
SELECT DISTINCT UCASE(ANIMAL) FROM PETRESCUE;
```

3. Write a query that displays all the columns from the PETRESCUE table where the animal(s) rescued are cats. Use **cat** in lowercase in the query.

▼ Click here for a hint
You have to use `LCASE` function for writing this query.
▼ Click here for the solution

```
SELECT * FROM PETRESCUE WHERE LCASE(ANIMAL)="cat";
```

4. Write a query that displays the number of rescues in the 5^{th} month.

▶ Click here for a hint
▼ Click here for the solution

```
SELECT SUM(QUANTITY) FROM PETRESCUE WHERE MONTH(RESCUEDATE)="05";
```

5. The rescue shelter is supposed to find good homes for all animals within 1 year of their rescue. Write a query that displays the ID and the target date.

▶ Click here for a hint
▼ Click here for Solution

```
SELECT ID, DATE_ADD(RESCUEDATE, INTERVAL 1 YEAR) FROM PETRESCUE;
```

Conclusion

Congratulations on completing this lab.

You are now able to:

- Use aggregation functions to calculate total, maximum, minimum, and average values of numerical attributes.
- Use scalar functions to round a floating value to the desired number of decimal places.
- Use string functions to convert text into upper or lower cases.
- Use date operations to manipulate data columns with the attribute as date.

Author(s)

Lakshmi Holla

Malika Singla

Abhishek Gagneja