Getting started with MySQL command line

Hands-on Lab : Getting started with MySQL command line

Estimated time needed: 20 minutes

In this lab, you will use the MySQL command line interface (CLI) to create a database and to restore the structure and contents of tables. Then you will learn how to explore and query tables. Finally, you will learn how to dump/backup tables from database.

Software Used in this Lab

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

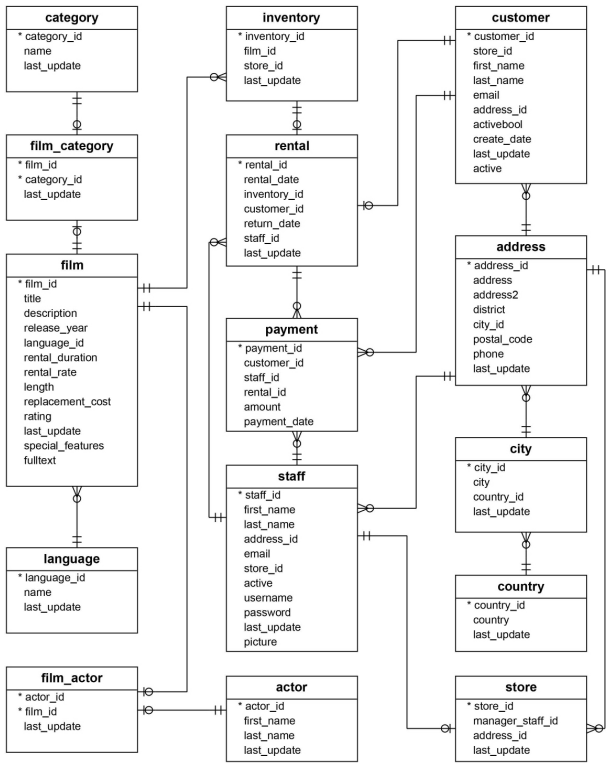
To complete this lab you will utilize the MySQL relational database service available as part of the IBM Skills Network Labs (SN Labs) Cloud IDE. SN Labs is a virtual lab environment used in this course.

Database Used in this Lab

The Sakila database used in this lab comes from the following source: https://dev.mysql.com/doc/sakila/en/ under New BSD license [Copyright 2021 - Oracle Corporation].

You will use a modified version of the database for the lab, so to follow the lab instructions successfully please use the database provided with the lab, rather than the database from the original source.

The following entity relationship diagram (ERD) shows the schema of the Sakila database:



Objectives

After completing this lab, you will be able to use the MySQL command line to:

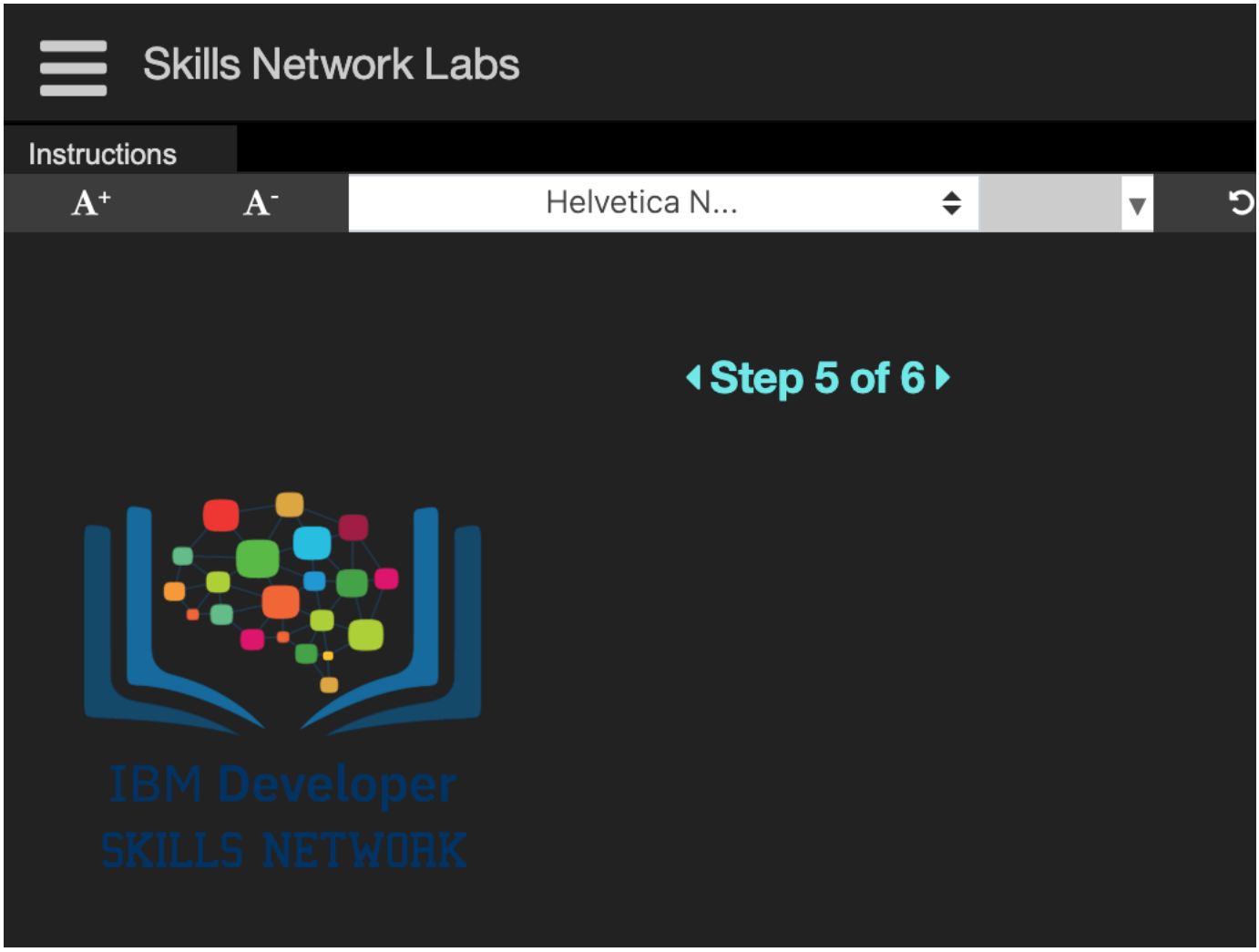
- Create a database.
- Restore the structure and data of a table.
- Explore and query tables.
- Dump/backup tables from a database.

Exercise

In this exercise through different tasks, you will use the MySQL command line interface (CLI) to create a database and to restore the structure and contents of tables. Then you will learn how to explore and query tables. Finally, you will learn how to dump/backup tables from database.

Task A: Create a database

1. Go to **Terminal > New Terminal** to open a terminal from the side by side launched Cloud IDE.



2. Copy the command below by clicking on the little copy button on the bottom right of the codeblock and then paste it into the terminal using **Ctrl + V** (Mac: ⌘ + V) to fetch the sakila_mysql_dump.sql file to the Cloud IDE.

```
1. 1
1. wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0110EN-SkillsNetwork/datasets/sakila/sakila_mysql_dump.sql
```

Copied!

```
Problems        theia@theiadocker-sandipsahajo: /home/project  ×

theia@theiadocker-sandipsahajo:/home/project$ wget https://cf-cour
BM-DB0110EN-SkillsNetwork/datasets/sakila/sakila_mysql_dump.sql
--2021-03-16 07:25:29--  https://cf-courses-data.s3.us.cloud-objec
datasets/sakila/sakila_mysql_dump.sql
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.clo
ain.cloud)... 169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain
pdomain.cloud)|169.63.118.104|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3625781 (3.5M) [application/x-sql]
Saving to: 'sakila_mysql_dump.sql'

sakila_mysql_dump.sql          100%[=============================

2021-03-16 07:25:31 (1.94 MB/s) - 'sakila_mysql_dump.sql' saved [3
```

3. Start the MySQL service session in the Cloud IDE using the command below in the terminal:

```
1. 1
1. start_mysql
```

Copied!

```
theia@theiadocker-sandipsahajo:/home/project$ start_mysql
Starting your MySQL database....
This process can take up to a minute.

MySQL database started, waiting for all services to be ready....

Your MySQL database is now ready to use and available with usernam

You can access your MySQL database via:
 • The browser at: https://sandipsahajo-8080.theiadocker-27.proxy.
 • CommandLine: mysql --host=127.0.0.1 --port=3306 --user=root --p
theia@theiadocker-sandipsahajo:/home/project$
```

4. Initiate the mysql command prompt session within MySQL service session using the command below in the terminal:

```
1. 1
1. mysql --host=127.0.0.1 --port=3306 --user=root --password
```

Copied!

When prompted, enter the password that was displayed when MySQL started up, as shown in the screenshot below.

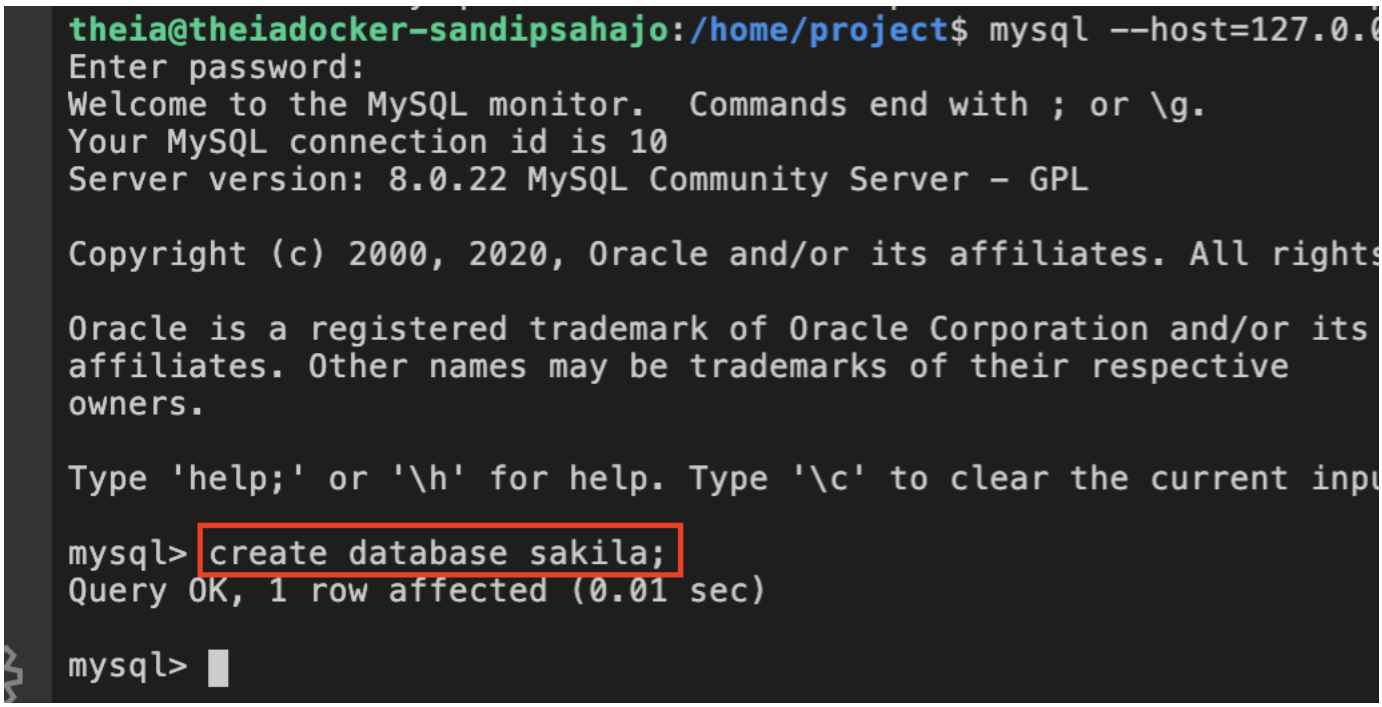
```
theia@theiadocker-sandipsahajo:/home/project$ start_mysql
Starting your MySQL database....
This process can take up to a minute.

MySQL database started, waiting for all services to be ready....

Your MySQL database is now ready to use and available with usernam

You can access your MySQL database via:
 • The browser at: https://sandipsahajo-8080.theiadocker-27.proxy.
 • CommandLine: mysql --host=127.0.0.1 --port=3306 --user=root --p
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0
Enter password:
```

Please note, you won't be able to see your password when typing it in. Not to worry, this is expected!

5. Enter your MySQL service session password from the highlighted location of the terminal shown in the image above. Note down your MySQL service session password because you may need to use it later in the lab.

6. Create a new database **sakila** using the command below in the terminal and proceed to Task B:

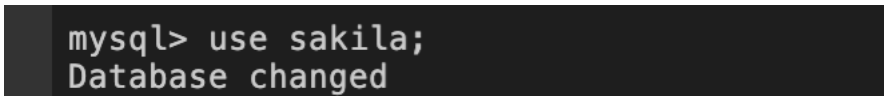
```
1. 1
1. create database sakila;
```
Copied!



Task B: Restore the structure and data of a table

1. To use the newly created empty sakila database, use the command below in the terminal:

```
1. 1
1. use sakila;
```
Copied!



2. Restore the sakila mysql dump file (containing the sakila database table definitions and data) to the newly created empty sakila database. A dump file is a text file that contains the data from a database in the form of SQL statements. This file can be imported using the command line with the following command:

```
1. 1
1. source sakila_mysql_dump.sql;
```
Copied!

> **Note:** You can use the **source** command to restore the database dump file within the mysql command prompt. To restore the database dump file outside of the mysql command prompt, you can use the `mysql --host=127.0.0.1 --port=3306 --user=root --password sakila < sakila_mysql_dump.sql` command after quitting the mysql command prompt session with command `\q`.

Task C: Explore and query tables

1. To list all the tables names from the sakila database, use the command below in the terminal:

```
1. 1
1. SHOW FULL TABLES WHERE table_type = 'BASE TABLE';
```
Copied!

```
mysql> SHOW FULL TABLES WHERE table_type = 'BASE TABLE';
+-----------------+------------+
| Tables_in_sakila | Table_type |
+-----------------+------------+
| actor           | BASE TABLE |
| address         | BASE TABLE |
| category        | BASE TABLE |
| city            | BASE TABLE |
| country         | BASE TABLE |
| customer        | BASE TABLE |
| film            | BASE TABLE |
| film_actor      | BASE TABLE |
| film_category   | BASE TABLE |
| inventory       | BASE TABLE |
| language        | BASE TABLE |
| payment         | BASE TABLE |
| rental          | BASE TABLE |
| staff           | BASE TABLE |
| store           | BASE TABLE |
+-----------------+------------+
15 rows in set (0.00 sec)

mysql>
```

The **Table_type** for these tables is **BASE TABLE**. **BASE TABLE** means that it is a table as opposed to a view (**VIEW**) or an `INFORMATION_SCHEMA` view (**SYSTEM VIEW**).

2. Explore the structure of the **staff** table using the command below in the terminal:

```
1. 1
1. DESCRIBE staff;
```

Copied!

```
mysql> DESCRIBE staff;
+-------------+-------------------+------+-----+-------------------
| Field       | Type              | Null | Key | Default
+-------------+-------------------+------+-----+-------------------
| staff_id    | tinyint unsigned  | NO   | PRI | NULL
| first_name  | varchar(45)       | NO   |     | NULL
| last_name   | varchar(45)       | NO   |     | NULL
| address_id  | smallint unsigned | NO   | MUL | NULL
| picture     | blob              | YES  |     | NULL
| email       | varchar(50)       | YES  |     | NULL
| store_id    | tinyint unsigned  | NO   | MUL | NULL
| active      | tinyint(1)        | NO   |     | 1
| username    | varchar(16)       | NO   |     | NULL
| password    | varchar(40)       | YES  |     | NULL
| last_update | timestamp         | NO   |     | CURRENT_TIMESTAMP
+-------------+-------------------+------+-----+-------------------
11 rows in set (0.00 sec)

mysql>
```

To understand the output, see the following table:

Column Name	Definition
Field	Name of the column.
Type	Data type of the column.
Null	Displays **YES** if column can contain NULL values and **NO** if not. Notice how the primary key displays **NO**.

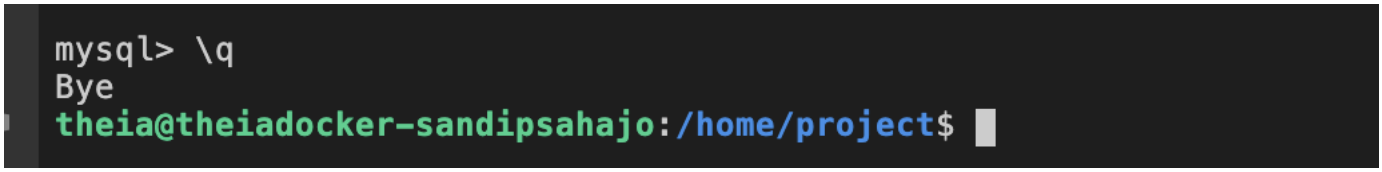
Column Name	Definition
Key	Displays the value **PRI** if the column is a primary key, **UNI** if the column is a unique key, and **MUL** if the column is a non-unique index in which one value can appear multiple times. If there is no value displayed, then the column isn't indexed or it's indexed as a secondary column. Please note, that if more than one of these values applies to the column, the value that appears will be displayed based on the following order: **PRI**, **UNI**, and **MUL**.
Default	The default value of the column. If the column's value has specifically been set as NULL, then the value that appears will be NULL.
Extra	Any additional information about a column.

3. Now retrieve all the records from the **staff** table using the command below in the terminal:

```
1. 1
1. SELECT * FROM staff;
```
Copied!

4. Quit the MySQL command prompt session using the command below in the terminal and proceed to Task D:

```
1. 1
1. \q
```
Copied!



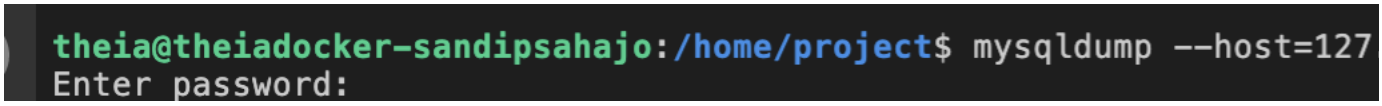
Task D: Dump/backup tables from a database

1. Finally, dump/backup the **staff** table from the database using the command below in the terminal:

```
1. 1
1. mysqldump --host=127.0.0.1 --port=3306 --user=root --password sakila staff > sakila_staff_mysql_dump.sql
```
Copied!

This command will backup the **staff** table from the **sakila** database into a file called **sakila_staff_mysql_dump.sql**.

2. Enter your MySQL service session password.



3. To view the contents of the dump file within the terminal, use the command below:

```
1. 1
1. cat sakila_staff_mysql_dump.sql
```
Copied!

```
theia@theiadocker-sandipsahajo:/home/project$ cat sakila_staff_mys
-- MySQL dump 10.13  Distrib 5.7.32, for Linux (x86_64)
--
-- Host: 127.0.0.1    Database: sakila
-- ------------------------------------------------------
-- Server version       8.0.22

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `staff`
--

DROP TABLE IF EXISTS `staff`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `staff` (
  `staff_id` tinyint unsigned NOT NULL AUTO_INCREMENT,
  `first_name` varchar(45) NOT NULL,
  `last_name` varchar(45) NOT NULL,
  `address_id` smallint unsigned NOT NULL,
  `picture` blob,
  `email` varchar(50) DEFAULT NULL,
  `store_id` tinyint unsigned NOT NULL,
  `active` tinyint(1) NOT NULL DEFAULT '1',
  `username` varchar(16) NOT NULL,
  `password` varchar(40) CHARACTER SET utf8 COLLATE utf8_bin DEFAU
  `last_update` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UP
  PRIMARY KEY (`staff_id`),
  KEY `idx_fk_store_id` (`store_id`),
  KEY `idx_fk_address_id` (`address_id`),
  CONSTRAINT `fk_staff_address` FOREIGN KEY (`address_id`) REFEREN
  CONSTRAINT `fk_staff_store` FOREIGN KEY (`store_id`) REFERENCES
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;
```

Congratulations! You have completed this lab, and you are ready for the next topic.

Author(s)

- Sandip Saha Joy

Other Contributor(s)

- Kathy An

Changelog

Date	Version	Changed by	Change Description
2021-03-15	1.0	Sandip Saha Joy	Created initial version
2021-10-18	1.1	Kathy An	Updated lab instructions