Hands-on Lab : Backup and Restore using MySQL

Estimated time needed: 25 minutes

In this lab, you will learn how to use the MySQL command line interface (CLI) to create different types of backups of a database and restore the structure and data of a database with your created backups when needed.

Objectives

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

- Perform a Logical Backup and Restore
- Perform Point-in-Time Backup and Restoration
- Perform a Physical Backup and Restore

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.

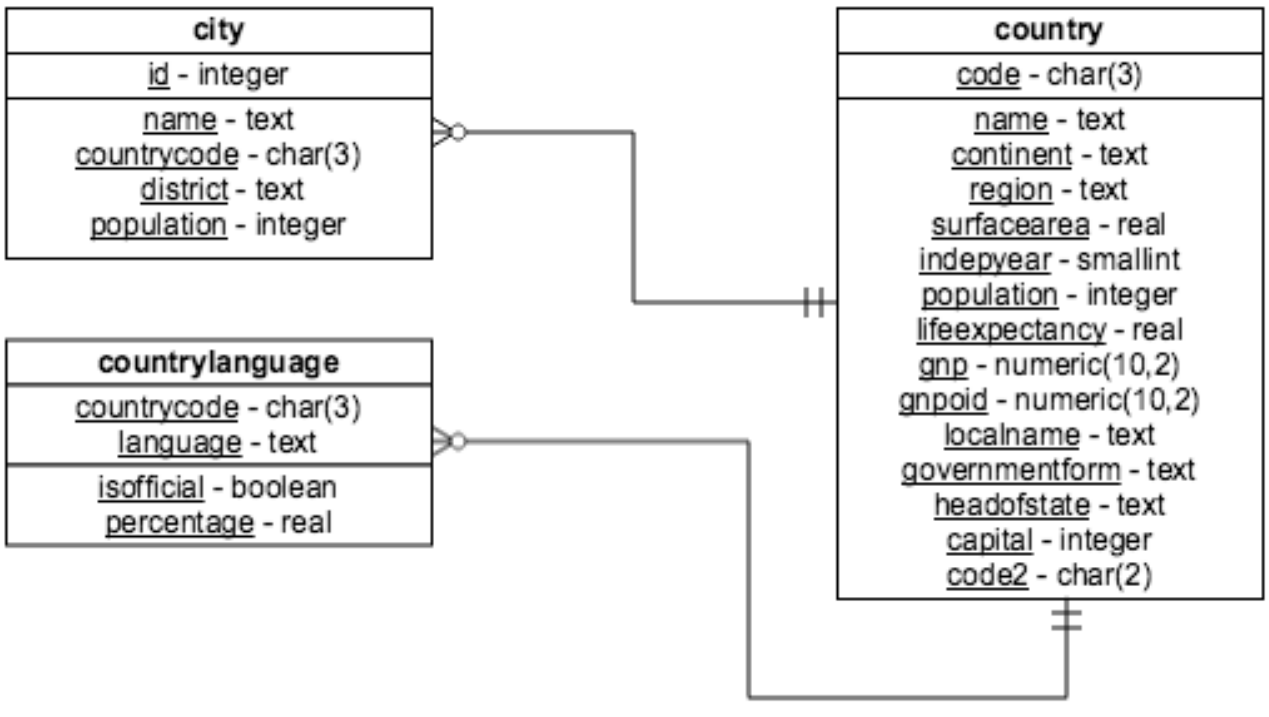
To complete this lab you will utilize the MySQL relational database service available as part of the IBM Skills Network Labs (SN Labs) Cloud IDE. SN Labs is a virtual lab environment used in this course.

Database Used in this Lab

The World database used in this lab comes from the following source: https://dev.mysql.com/doc/world-setup/en/ under CC BY 4.0 License with Copyright 2021 - Statistics Finland.

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 ERD diagram shows the schema of the World database:



The first row is the table name, the second is the primary key, and the remaining items are any additional attributes.

Exercises

This lab is divided into two exercises: an Example Exercise and Practice Exercise.

Example Exercise A: Perform a Logical Backup and Restore

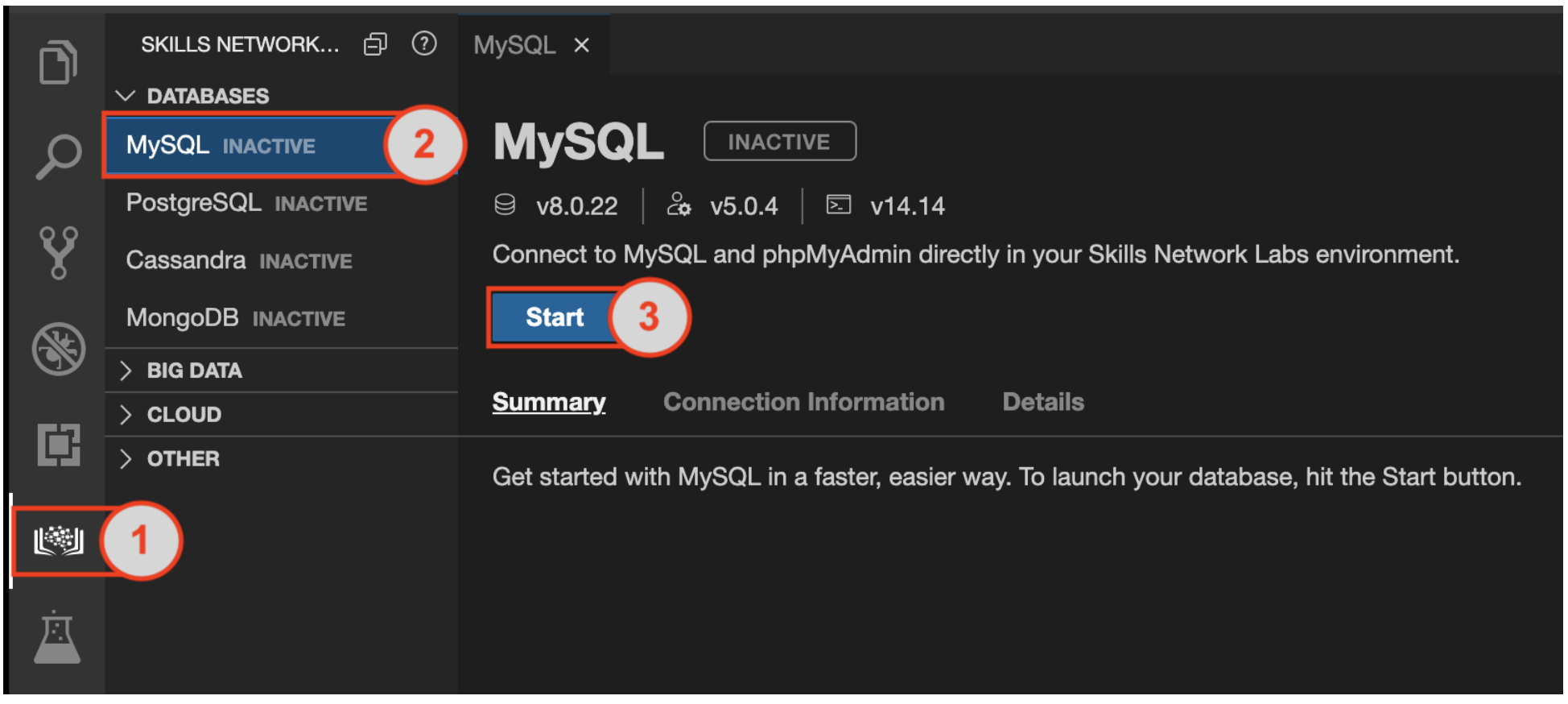
In this example exercise, you will go through an example covering how to perform a logical backup and restoration of a database table.

A logical backup creates a file containing DDL (such as create table) and DML commands (such as insert) that recreate the objects and data in the database. As such, you can use this file to recreate the database on the same or on another system. Generally, when you perform a logical backup and restore, you reclaim any wasted space from the original database since the restoration process creates a clean version of the tables. Logical backups enable you to backup granular objects. For example, you can back up an individual database table, however, you cannot use it to backup log files or database configuration settings. Suppose you are in a situation where you dropped one or more tables of a database accidentally. This is where you make use of the logical backup of a database table to restore the structure and data of the table.

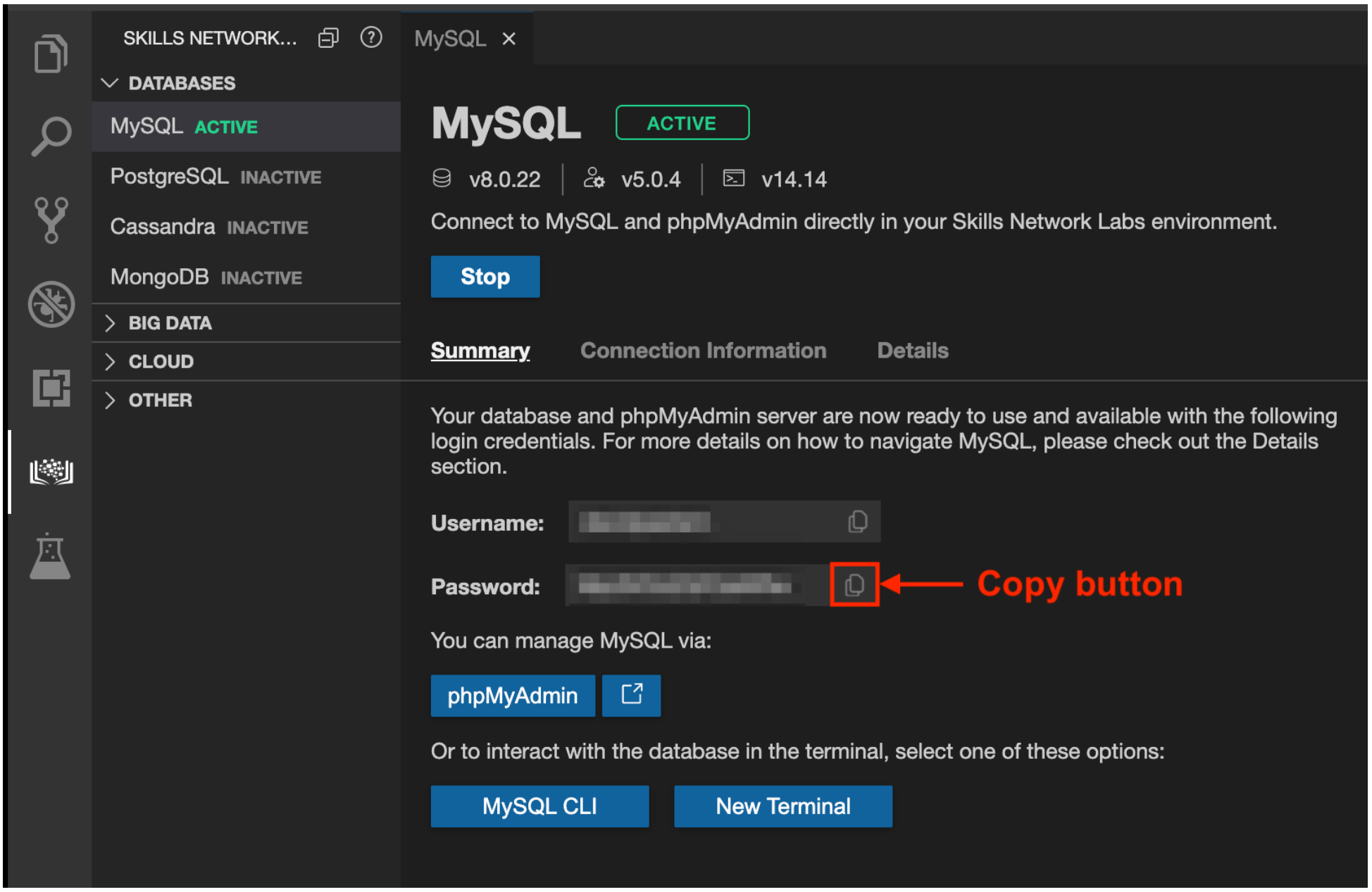
1. Go to **Skills Network Toolbox** by clicking the icon shown below from the side by side launched Cloud IDE.

2. From the **Databases** drop down menu, click **MySQL** to open the MySQL service session tab.

3. Click the **Start** button and wait until MySQL service session gets launched.



The MySQL server will take a few moments to start. Once it is ready, you will see the green "Active" label near the top of the window.



- **NOTE:** Whenever you are required to enter your MySQL service session password from the MySQL service session tab at any step of the lab, copy the password by clicking on the small copy button on the right of the password block. Paste the password into the terminal using **Ctrl + V** (Mac: ⌘ + V), and press **Enter** on the keyboard. For security reasons, you will not see the password as it is entered on the terminal.

4. Click **New Terminal** button from the mysql service session tab. Now you need to fetch two mysql script files to the Cloud IDE user session storage. Copy the command below by clicking on the little copy button on the bottom right of the codeblock. Then paste it into the terminal at the command line prompt using **Ctrl + V** (Mac: ⌘ + V), and **Enter** on keyboard. Do this for each of the commands below one at a time.

   - world_mysql_script.sql

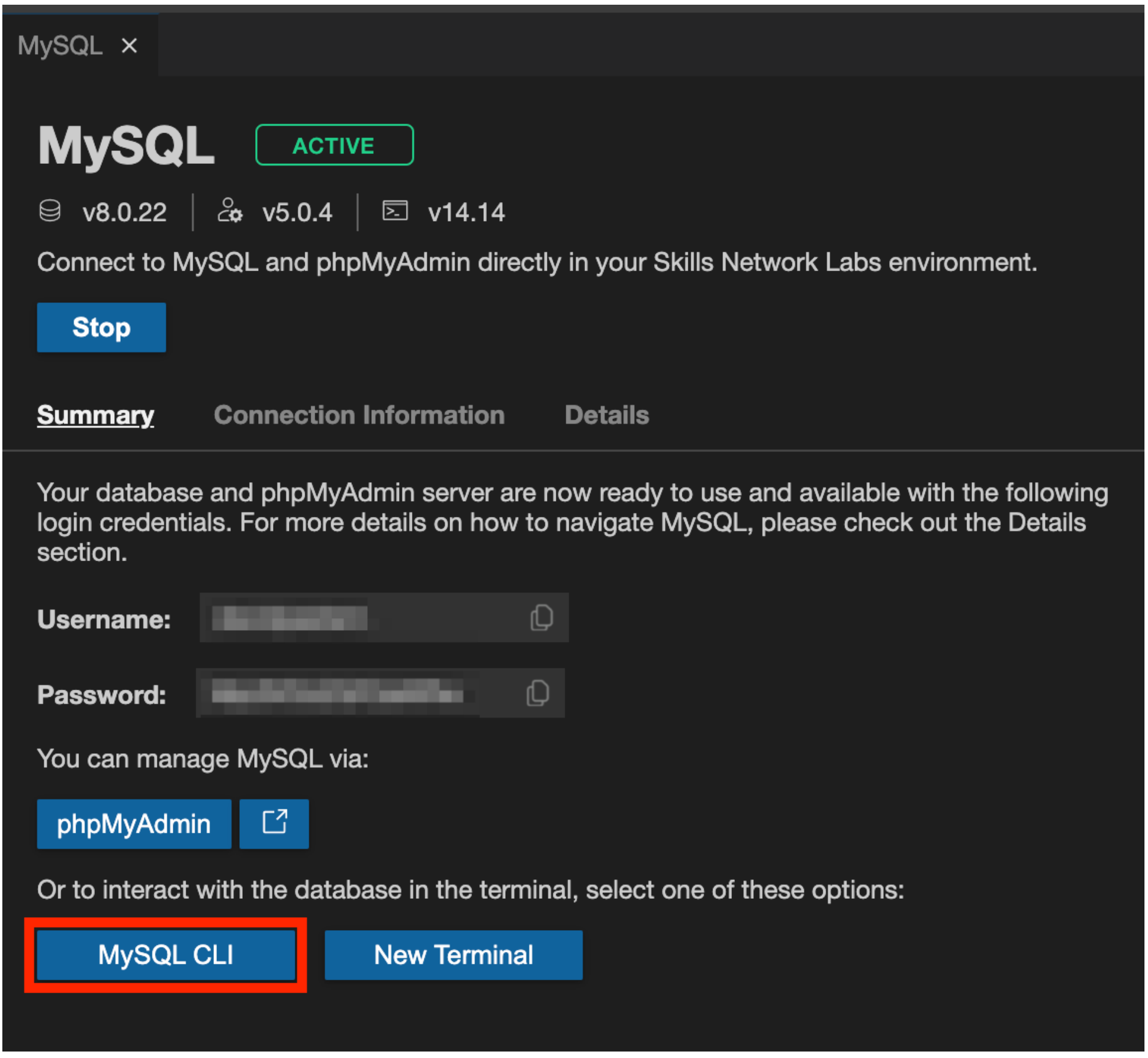
     ```
     wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0231EN-SkillsNetwork/datasets/World/world_mysql_script.sql
     ```

   - world_mysql_update_A.sql

     ```
     wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0231EN-SkillsNetwork/datasets/World/world_mysql_update_A.sql
     ```

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

theia@theiadocker-sandipsahajo:/home/project$ wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0
231EN-SkillsNetwork/datasets/World/world_mysql_script.sql
--2021-06-28 12:34:19--  https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0231EN-SkillsNetwork/datase
ts/World/world_mysql_script.sql
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-courses-data.s3.us.cloud-object-storage.appdomain.cl
oud)... 169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-courses-data.s3.us.cloud-object-storage.appdomai
n.cloud)|169.63.118.104|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 389649 (381K) [application/x-sql]
Saving to: 'world_mysql_script.sql'

world_mysql_script.sql           100%[=====================================================>] 380.52K  1.21MB/s    in 0.3s


2021-06-28 12:34:20 (1.21 MB/s) - 'world_mysql_script.sql' saved [389649/389649]

theia@theiadocker-sandipsahajo:/home/project$ wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0
231EN-SkillsNetwork/datasets/World/world_mysql_update_A.sql
--2021-06-28 12:34:58--  https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0231EN-SkillsNetwork/datase
ts/World/world_mysql_update_A.sql
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-courses-data.s3.us.cloud-object-storage.appdomain.cl
oud)... 169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-courses-data.s3.us.cloud-object-storage.appdomai
n.cloud)|169.63.118.104|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 254 [application/x-sql]
Saving to: 'world_mysql_update_A.sql'

world_mysql_update_A.sql         100%[=====================================================>]     254  --.-KB/s    in 0s


2021-06-28 12:34:58 (26.9 MB/s) - 'world_mysql_update_A.sql' saved [254/254]

theia@theiadocker-sandipsahajo:/home/project$
```

5. Initiate a mysql command prompt session by clicking the **MySQL CLI** button from the mysql service session tab.



6. Create a new database **world** using the command below in the terminal:

```
CREATE DATABASE world;
```

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3039
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> create database world;
Query OK, 1 row affected (0.01 sec)

mysql>
```

7. To use the newly created world database, use the command below in the terminal:

```
USE world;
```

```
mysql> use world;
Database changed
mysql>
```

8. Execute the world mysql script (world_mysql.sql) to complete the world database creation process using the command below in the terminal:

```
SOURCE world_mysql_script.sql;
```

```
Query OK, 1 row affected (0.01 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql>
```

9. To list all the table names from the world database, use the command below in the terminal:

```
SHOW TABLES;
```

```
mysql> SHOW TABLES;
+-----------------+
| Tables_in_world |
+-----------------+
| city            |
| country         |
| countrylanguage |
+-----------------+
3 rows in set (0.00 sec)

mysql>
```

10. Retrieve all the Canada (countrycode='CAN') related records from the **countrylanguage** table using the command below in the terminal:

```
SELECT * FROM countrylanguage WHERE countrycode='CAN';
```

```
mysql> SELECT * FROM countrylanguage WHERE countrycode='CAN';
Empty set (0.00 sec)

mysql>
```

11. You will observe the returned result set is empty set. This means Canada related records are currently absent from the table. Run the update script (world_mysql_update_A.sql) to insert the records you were looking for.

```
SOURCE world_mysql_update_A.sql;
```

12. Now redo step-9 to verify.

```
mysql> SELECT * FROM countrylanguage WHERE countrycode='CAN';
+-------------+-----------------+------------+------------+
| CountryCode | Language        | IsOfficial | Percentage |
+-------------+-----------------+------------+------------+
| CAN         | Chinese         | F          |        2.5 |
| CAN         | Dutch           | F          |        0.5 |
| CAN         | English         | T          |       60.4 |
| CAN         | Eskimo Languages| F          |        0.1 |
| CAN         | French          | T          |       23.4 |
| CAN         | German          | F          |        1.6 |
| CAN         | Italian         | F          |        1.7 |
| CAN         | Polish          | F          |        0.7 |
| CAN         | Portuguese      | F          |        0.7 |
| CAN         | Punjabi         | F          |        0.7 |
| CAN         | Spanish         | F          |        0.7 |
| CAN         | Ukrainian       | F          |        0.6 |
+-------------+-----------------+------------+------------+
12 rows in set (0.00 sec)

mysql>
```

13. Quit the MySQL command prompt session using the command below in the terminal:

```
\q
```

```
mysql> \q
Bye
theia@theiadocker-sandipsahajo:/home/project$
```

14. Now backup the **countrylanguage** table of the **world** database using the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysqldump --host=127.0.0.1 --port=3306 --user=root --password world countrylanguage >
world_countrylanguage_mysql_backup.sql
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysqldump --host=127.0.0.1 --port=3306 --user=root --password
world countrylanguage > world_countrylanguage_mysql_backup.sql
Enter password:
theia@theiadocker-sandipsahajo:/home/project$
```

15. To view the contents of the backup file within the terminal, use the command below:

```
cat world_countrylanguage_mysql_backup.sql
```

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

'T',47.5),('WSM','Samoan-English','F',52.0),('YEM','Arabic','T',99.6),('YEM','Soqutri','F',0.0),('YUG','Alba
niana','F',16.5),('YUG','Hungarian','F',3.4),('YUG','Macedonian','F',0.5),('YUG','Romani','F',1.4),('YUG','S
erbo-Croatian','T',75.2),('YUG','Slovak','F',0.7),('ZAF','Afrikaans','T',14.3),('ZAF','English','T',8.5),('Z
AF','Ndebele','F',1.5),('ZAF','Northsotho','F',9.1),('ZAF','Southsotho','F',7.6),('ZAF','Swazi','F',2.5),('Z
AF','Tsonga','F',4.3),('ZAF','Tswana','F',8.1),('ZAF','Venda','F',2.2),('ZAF','Xhosa','T',17.7),('ZAF','Zulu
','T',22.7),('ZMB','Bemba','F',29.7),('ZMB','Chewa','F',5.7),('ZMB','Lozi','F',6.4),('ZMB','Nsenga','F',4.3)
,('ZMB','Nyanja','F',7.8),('ZMB','Tongan','F',11.0),('ZWE','English','T',2.2),('ZWE','Ndebele','F',16.2),('Z
WE','Nyanja','F',2.2),('ZWE','Shona','F',72.1);
/*!40000 ALTER TABLE `countrylanguage` ENABLE KEYS */;
UNLOCK TABLES;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2021-06-28 13:19:45
theia@theiadocker-sandipsahajo:/home/project$
```

16. Run the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysql --host=127.0.0.1 --port=3306 --user=root --password --execute="DROP TABLE world.countrylanguage;"
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
 --execute="DROP TABLE world.countrylanguage;"
Enter password:
theia@theiadocker-sandipsahajo:/home/project$
```

17. To list all the table names from the world database, use the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysql --host=127.0.0.1 --port=3306 --user=root --password --execute="SHOW TABLES FROM world;"
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
 --execute="SHOW TABLES FROM world;"
Enter password:
+-----------------+
| Tables_in_world |
+-----------------+
| city            |
| country         |
+-----------------+
theia@theiadocker-sandipsahajo:/home/project$
```

18. You will observe the table **countrylanguage** is missing from the world database. Now you are in the situation where you dropped a table of a database accidentally. This is where you will make use of the backup of the database table (you created backup **world_countrylanguage_mysql_backup.sql**) to restore the structure and data of the table.

19. To restore the structure and data of the table **countrylanguage**, use the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysql --host=127.0.0.1 --port=3306 --user=root --password world < world_countrylanguage_mysql_backup.sql
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
 world < world_countrylanguage_mysql_backup.sql
Enter password:
theia@theiadocker-sandipsahajo:/home/project$
```

20. Now redo step-17 to verify.

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
 --execute="SHOW TABLES FROM world;"
Enter password:
+-----------------+
| Tables_in_world |
+-----------------+
| city            |
| country         |
| countrylanguage |
+-----------------+
theia@theiadocker-sandipsahajo:/home/project$
```

21. Again retrieve all the Canada (countrycode='CAN') related records from the **countrylanguage** table using the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysql --host=127.0.0.1 --port=3306 --user=root --password --execute="SELECT * FROM world.countrylanguage WHERE countrycode='CAN';"
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
 --execute="SELECT * FROM world.countrylanguage WHERE countrycode='CAN';"
Enter password:
+-------------+-----------------+------------+------------+
| CountryCode | Language        | IsOfficial | Percentage |
+-------------+-----------------+------------+------------+
| CAN         | Chinese         | F          |        2.5 |
| CAN         | Dutch           | F          |        0.5 |
| CAN         | English         | T          |       60.4 |
| CAN         | Eskimo Languages| F          |        0.1 |
| CAN         | French          | T          |       23.4 |
| CAN         | German          | F          |        1.6 |
| CAN         | Italian         | F          |        1.7 |
| CAN         | Polish          | F          |        0.7 |
| CAN         | Portuguese      | F          |        0.7 |
| CAN         | Punjabi         | F          |        0.7 |
| CAN         | Spanish         | F          |        0.7 |
| CAN         | Ukrainian       | F          |        0.6 |
+-------------+-----------------+------------+------------+
theia@theiadocker-sandipsahajo:/home/project$
```

Example Exercise B: Perform Point-in-Time Backup and Restore

In this example exercise, you will go through an example on how to perform a point-in-time backup and restore of a database.

Say you have a full logical backup of your whole database in your last mysqldump file as of yesterday evening. However, several changes may have been made (including data loss) since then. Using point-in-time backup and restore, you can get each and every change that occurred since then, so that even after your last logical backup you have a record of all new transactions. Point-in-time backup is the set of binary log files generated subsequent to a logical backup operation of a database. The binary log files contain events that describe database changes such as table creation operations or changes to table data. To restore a database to a point-in-time, you will be using binary log files containing changes of a database for a time interval along with the last logical backup of the database.

1. Click **New Terminal** button from the mysql service session tab.

2. Now you need to fetch a mysql script file to the Cloud IDE user session storage. Copy the command below by clicking on the little copy button on the bottom right of the codeblock. Then paste it into the terminal at the command line prompt using **Ctrl + V** (Mac: ⌘ + V), and **Enter** on keyboard.

    - world_mysql_update_B.sql

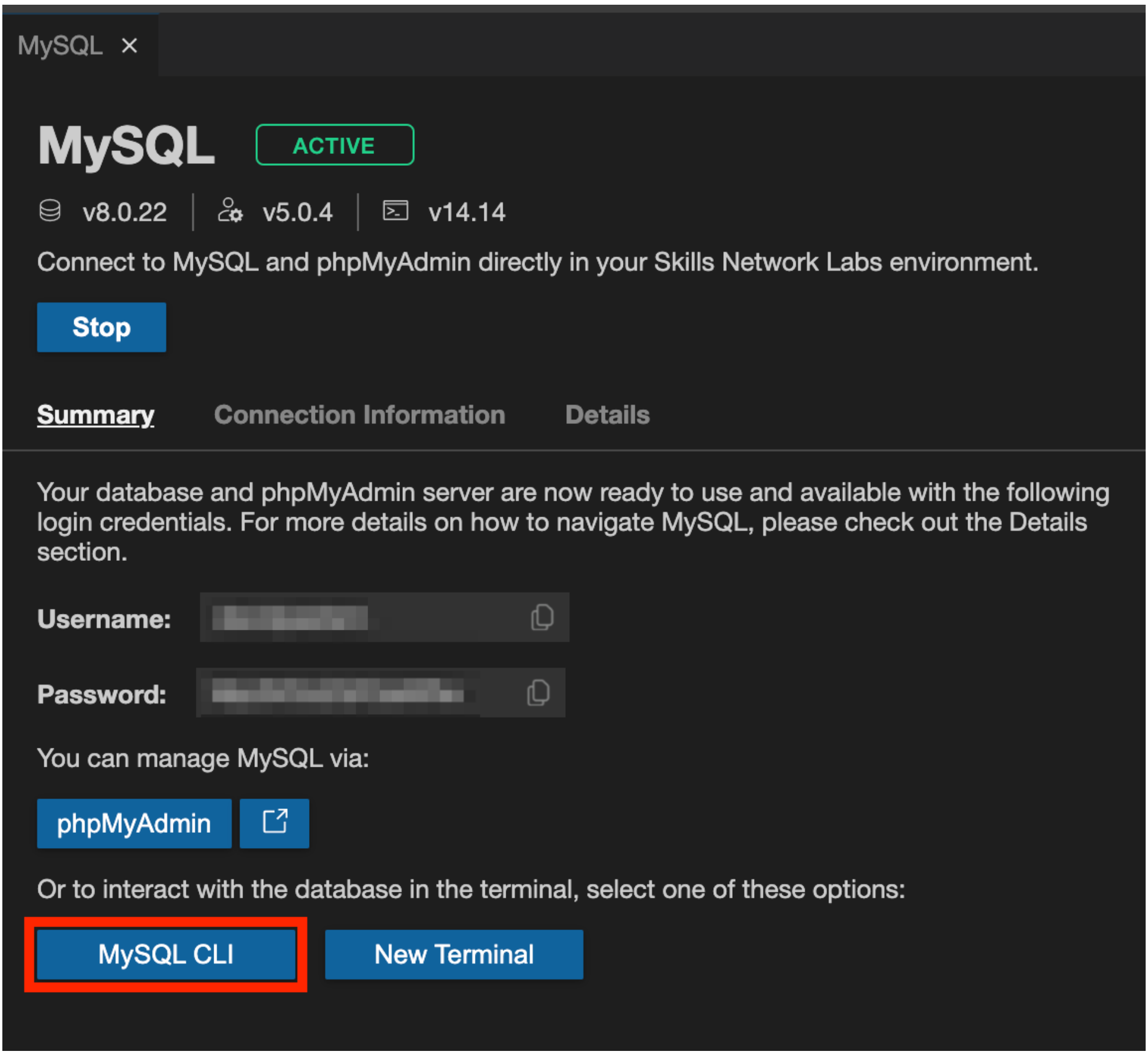
    ```
    wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0231EN-SkillsNetwork/datasets/World/world_mysql_update_B.sql
    ```

3. First create a full logical backup of the current state of your whole **world** database. Use the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

    ```
    mysqldump --host=127.0.0.1 --port=3306 --user=root --password --flush-logs --delete-master-logs  --databases world > world_mysql_full_backup.sql
    ```

    - **NOTE:** The two parameters in the command above, `--flush-logs` (starts writing to a new binlog file) and `--delete-master-logs` (removes old binlog files) ensures that there will be only binary log files created after a full logical backup.

4. Initiate a mysql command prompt session by clicking the **MySQL CLI** button from the mysql service session tab.



5. To use the already created world database of example exercise A, use the command below in the terminal:

    ```
    use world;
    ```

6. List all the table names from the world database using the command below in the terminal:

    ```
    SHOW TABLES;
    ```

7. Retrieve all the Canada (countrycode='CAN') related records from the **city** table using the command below in the terminal:

```
SELECT * FROM city WHERE countrycode='CAN';
```

```
mysql> SELECT * FROM city WHERE countrycode='CAN';
Empty set (0.00 sec)

mysql>
```

8. You will observe the returned result set is empty set. This means Canada related records are currently absent from the table. Run the update script (world_mysql_update_B.sql) to insert the records you were looking for.

```
source world_mysql_update_B.sql;
```

9. Now redo step-7 to verify.

```
mysql> SELECT * FROM city WHERE countrycode='CAN';
+------+-------------------+-------------+-------------------+------------+
| ID   | Name              | CountryCode | District          | Population |
+------+-------------------+-------------+-------------------+------------+
| 1810 | Montréal          | CAN         | Québec            |    1016376 |
| 1811 | Calgary           | CAN         | Alberta           |     768082 |
| 1812 | Toronto           | CAN         | Ontario           |     688275 |
| 1813 | North York        | CAN         | Ontario           |     622632 |
| 1814 | Winnipeg          | CAN         | Manitoba          |     618477 |
| 1815 | Edmonton          | CAN         | Alberta           |     616306 |
| 1816 | Mississauga       | CAN         | Ontario           |     608072 |
| 1817 | Scarborough       | CAN         | Ontario           |     594501 |
| 1818 | Vancouver         | CAN         | British Colombia  |     514008 |
| 1819 | Etobicoke         | CAN         | Ontario           |     348845 |
| 1820 | London            | CAN         | Ontario           |     339917 |
| 1821 | Hamilton          | CAN         | Ontario           |     335614 |
| 1822 | Ottawa            | CAN         | Ontario           |     335277 |
| 1823 | Laval             | CAN         | Québec            |     330393 |
| 1824 | Surrey            | CAN         | British Colombia  |     304477 |
| 1825 | Brampton          | CAN         | Ontario           |     296711 |
| 1826 | Windsor           | CAN         | Ontario           |     207588 |
| 1827 | Saskatoon         | CAN         | Saskatchewan      |     193647 |
| 1828 | Kitchener         | CAN         | Ontario           |     189959 |
| 1829 | Markham           | CAN         | Ontario           |     189098 |
| 1830 | Regina            | CAN         | Saskatchewan      |     180400 |
| 1831 | Burnaby           | CAN         | British Colombia  |     179209 |
| 1832 | Québec            | CAN         | Québec            |     167264 |
| 1833 | York              | CAN         | Ontario           |     154980 |
| 1834 | Richmond          | CAN         | British Colombia  |     148867 |
| 1835 | Vaughan           | CAN         | Ontario           |     147889 |
| 1836 | Burlington        | CAN         | Ontario           |     145150 |
| 1837 | Oshawa            | CAN         | Ontario           |     140173 |
| 1838 | Oakville          | CAN         | Ontario           |     139192 |
| 1839 | Saint Catharines  | CAN         | Ontario           |     136216 |
| 1840 | Longueuil         | CAN         | Québec            |     127977 |
| 1841 | Richmond Hill     | CAN         | Ontario           |     116428 |
| 1842 | Thunder Bay       | CAN         | Ontario           |     115913 |
| 1843 | Nepean            | CAN         | Ontario           |     115100 |
| 1844 | Cape Breton       | CAN         | Nova Scotia       |     114733 |
| 1845 | East York         | CAN         | Ontario           |     114034 |
| 1846 | Halifax           | CAN         | Nova Scotia       |     113910 |
| 1847 | Cambridge         | CAN         | Ontario           |     109186 |
| 1848 | Gloucester        | CAN         | Ontario           |     107314 |
| 1849 | Abbotsford        | CAN         | British Colombia  |     105403 |
| 1850 | Guelph            | CAN         | Ontario           |     103593 |
| 1851 | Saint John´s      | CAN         | Newfoundland      |     101936 |
| 1852 | Coquitlam         | CAN         | British Colombia  |     101820 |
| 1853 | Saanich           | CAN         | British Colombia  |     101388 |
| 1854 | Gatineau          | CAN         | Québec            |     100702 |
```

10. Quit the MySQL command prompt session using the command below in the terminal:

```
\q
```

11. Now you will create a scenario where a database crash will be conducted intentionally which will result a significant loss of your **world** database files. To create the scenario, copy the command below by clicking on the little copy button on the bottom right of the codeblock. Then paste it into the terminal at the command line prompt using **Ctrl + V** (Mac: ⌘ + V), and **Enter** on keyboard. Do this for each of the commands below one at a time.

```
docker exec mysql-mysql-1 rm -rf /var/lib/mysql/world
```

```
docker exec -it mysql-mysql-1 mysqladmin -p shutdown
```

```
theia@theiadocker-sandipsahajo:/home/project$ docker exec mysql_mysql_1 rm -rf /var/lib/mysql/world
theia@theiadocker-sandipsahajo:/home/project$ docker exec -it mysql_mysql_1 mysqladmin -p shutdown
Enter password:
```

12. Try to retrieve records from any table of the database using like the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysql --host=127.0.0.1 --port=3306 --user=root --password --execute="SELECT * FROM world.city;"
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
 --execute="SELECT * FROM world.city;"
Enter password:
ERROR 1812 (HY000) at line 1: Tablespace is missing for table `world`.`city`.
theia@theiadocker-sandipsahajo:/home/project$
```

13. You will face errors since a significant loss of your **world** database files happened. Now you have to restore the world database along with the updates you made earlier in this exercise running the update script (world_mysql_update_B.sql). Display the binary logs using the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysql --host=127.0.0.1 --port=3306 --user=root --password --execute="SHOW BINARY LOGS;"
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
--execute="SHOW BINARY LOGS;"
Enter password:
+---------------+-----------+-----------+
| Log_name      | File_size | Encrypted |
+---------------+-----------+-----------+
| binlog.000003 |     15827 | No        |
| binlog.000004 |       156 | No        |
+---------------+-----------+-----------+
theia@theiadocker-sandipsahajo:/home/project$
```

14. Write the contents of all binary log files listed above to a single file using the command below in the terminal:

```
docker exec mysql-mysql-1 mysqlbinlog /var/lib/mysql/binlog.000003 /var/lib/mysql/binlog.000004 > logfile.sql
```

```
theia@theiadocker-sandipsahajo:/home/project$ docker exec mysql_mysql_1 mysqlbinlog /var/lib/mysql/binlog.000003
/var/lib/mysql/binlog.000004 > logfile.sql
theia@theiadocker-sandipsahajo:/home/project$
```

15. You are ready to perform point-in-time restore. First restore the full logical backup of your whole **world** database you created earlier in this exercise using the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysql --host=127.0.0.1 --port=3306 --user=root --password < world_mysql_full_backup.sql
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
< world_mysql_full_backup.sql
Enter password:
theia@theiadocker-sandipsahajo:/home/project$
```

16. To verify if you have the updates from the update script (world_mysql_update_B.sql), retrieve all the Canada (countrycode='CAN') related records from the **city** table using the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysql --host=127.0.0.1 --port=3306 --user=root --password --execute="SELECT * FROM world.city WHERE
countrycode='CAN';"
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
--execute="SELECT * FROM world.city WHERE countrycode='CAN';"
Enter password:
theia@theiadocker-sandipsahajo:/home/project$
```

17. Now run the logfile you created in step-14 using the command below in the terminal (enter your MySQL service session password from the MySQL service session tab if necessary):

```
mysql --host=127.0.0.1 --port=3306 --user=root --password < logfile.sql
```

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
< logfile.sql
Enter password:
theia@theiadocker-sandipsahajo:/home/project$
```

18. Redo step-16 to verify if you have the updates from the update script (world_mysql_update_B.sql).

```
theia@theiadocker-sandipsahajo:/home/project$ mysql --host=127.0.0.1 --port=3306 --user=root --password
--execute="SELECT * FROM world.city WHERE countrycode='CAN';"
Enter password:
+------+------------------+-------------+------------------+------------+
| ID   | Name             | CountryCode | District         | Population |
+------+------------------+-------------+------------------+------------+
| 1810 | Montréal         | CAN         | Québec           |    1016376 |
| 1811 | Calgary          | CAN         | Alberta          |     768082 |
| 1812 | Toronto          | CAN         | Ontario          |     688275 |
| 1813 | North York       | CAN         | Ontario          |     622632 |
| 1814 | Winnipeg         | CAN         | Manitoba         |     618477 |
| 1815 | Edmonton         | CAN         | Alberta          |     616306 |
| 1816 | Mississauga      | CAN         | Ontario          |     608072 |
| 1817 | Scarborough      | CAN         | Ontario          |     594501 |
| 1818 | Vancouver        | CAN         | British Colombia |     514008 |
| 1819 | Etobicoke        | CAN         | Ontario          |     348845 |
| 1820 | London           | CAN         | Ontario          |     339917 |
| 1821 | Hamilton         | CAN         | Ontario          |     335614 |
| 1822 | Ottawa           | CAN         | Ontario          |     335277 |
| 1823 | Laval            | CAN         | Québec           |     330393 |
| 1824 | Surrey           | CAN         | British Colombia |     304477 |
| 1825 | Brampton         | CAN         | Ontario          |     296711 |
| 1826 | Windsor          | CAN         | Ontario          |     207588 |
| 1827 | Saskatoon        | CAN         | Saskatchewan     |     193647 |
| 1828 | Kitchener        | CAN         | Ontario          |     189959 |
| 1829 | Markham          | CAN         | Ontario          |     189098 |
| 1830 | Regina           | CAN         | Saskatchewan     |     180400 |
| 1831 | Burnaby          | CAN         | British Colombia |     179209 |
| 1832 | Québec           | CAN         | Québec           |     167264 |
| 1833 | York             | CAN         | Ontario          |     154980 |
| 1834 | Richmond         | CAN         | British Colombia |     148867 |
| 1835 | Vaughan          | CAN         | Ontario          |     147889 |
| 1836 | Burlington       | CAN         | Ontario          |     145150 |
| 1837 | Oshawa           | CAN         | Ontario          |     140173 |
| 1838 | Oakville         | CAN         | Ontario          |     139192 |
| 1839 | Saint Catharines | CAN         | Ontario          |     136216 |
| 1840 | Longueuil        | CAN         | Québec           |     127977 |
| 1841 | Richmond Hill    | CAN         | Ontario          |     116428 |
| 1842 | Thunder Bay      | CAN         | Ontario          |     115913 |
| 1843 | Nepean           | CAN         | Ontario          |     115100 |
| 1844 | Cape Breton      | CAN         | Nova Scotia      |     114733 |
| 1845 | East York        | CAN         | Ontario          |     114034 |
| 1846 | Halifax          | CAN         | Nova Scotia      |     113910 |
| 1847 | Cambridge        | CAN         | Ontario          |     109186 |
| 1848 | Gloucester       | CAN         | Ontario          |     107314 |
| 1849 | Abbotsford       | CAN         | British Colombia |     105403 |
| 1850 | Guelph           | CAN         | Ontario          |     103593 |
| 1851 | Saint John´s     | CAN         | Newfoundland     |     101936 |
| 1852 | Coquitlam        | CAN         | British Colombia |     101820 |
| 1853 | Saanich          | CAN         | British Colombia |     101388 |
```

19. Finally through the point-in-time recovery, you have the **world** database in the same state before you conducted the intentional crash scenario.

Example Exercise C: Perform Physical Backup and Restore

In this example exercise, you will go through an example on how to perform a physical backup and restore of a database.

A physical or raw backup creates a copy of all the physical storage files and directories that belong to a table, database, or other object, including the data files, configuration files, and log files. Physical backups are often smaller and quicker than logical backups, so are useful for large or important databases that require fast recovery times. You will be performing a storage level snapshots as physical backup. This method is common for databases utilizing specialized cloud storage systems like the one you are using for this lab provide by the Skills Network Labs.

1. Click **New Terminal** button from the mysql service session tab.

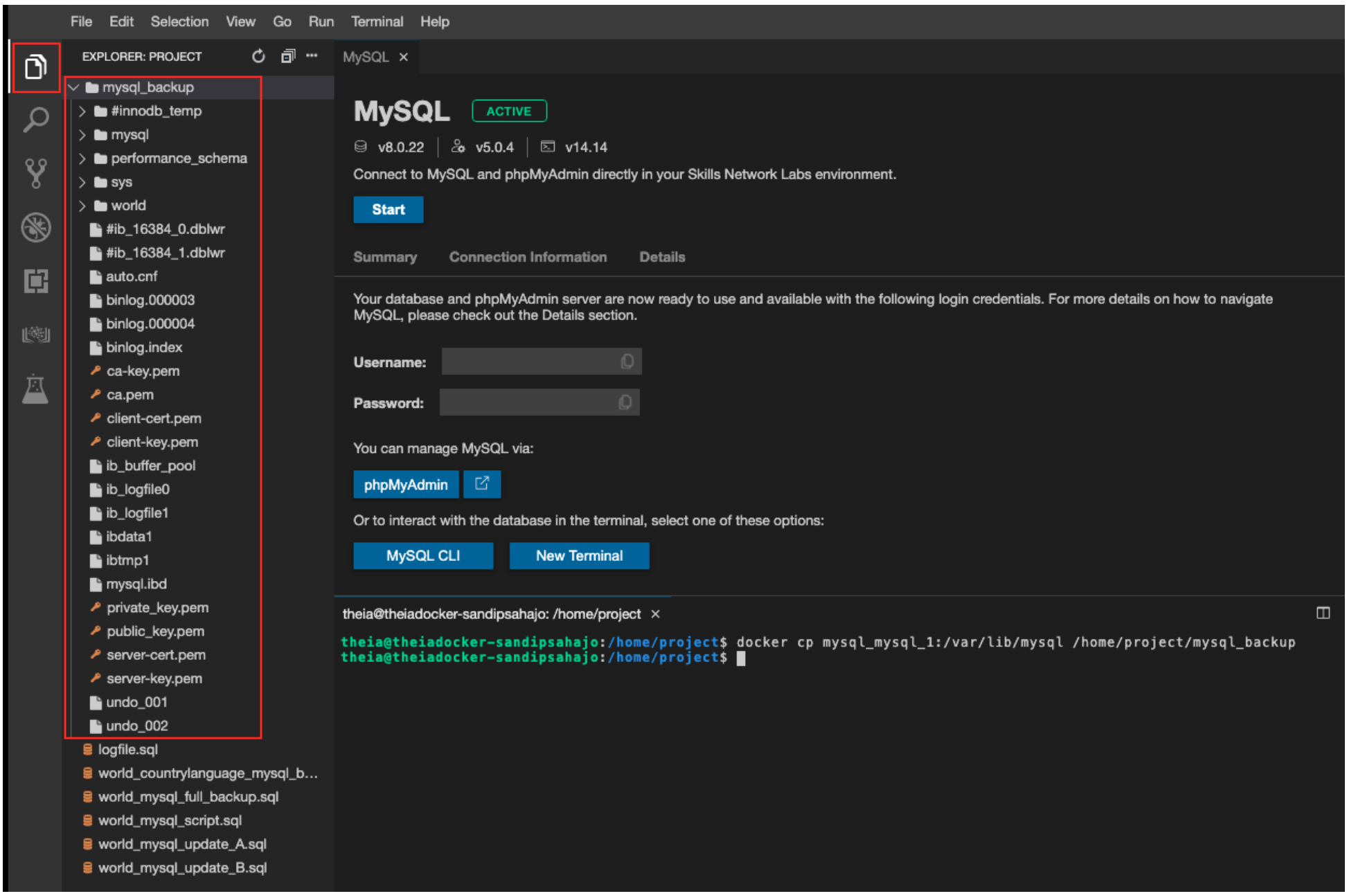
2. To perform physical backup, you will take a storage snapshot of your MySQL server data directory within the docker container of the Skills Network Labs specialized cloud system. Then copy that to your Cloud IDE user session storage. Use the command below in the terminal:

```
docker cp mysql-mysql-1:/var/lib/mysql /home/project/mysql_backup
```

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

theia@theiadocker-sandipsahajo:/home/project$ docker cp mysql_mysql_1:/var/lib/mysql /home/project/mysql_backup
theia@theiadocker-sandipsahajo:/home/project$
```

> **TIPS:** Say instead of taking snapshot of the whole MySQL server data directory which may contain several databases, you want to take snapshot of your specific **world** database for physical backup. The command for that should look like: `docker cp mysql-mysql-1:/var/lib/mysql/world /home/project/mysql_world_backup`

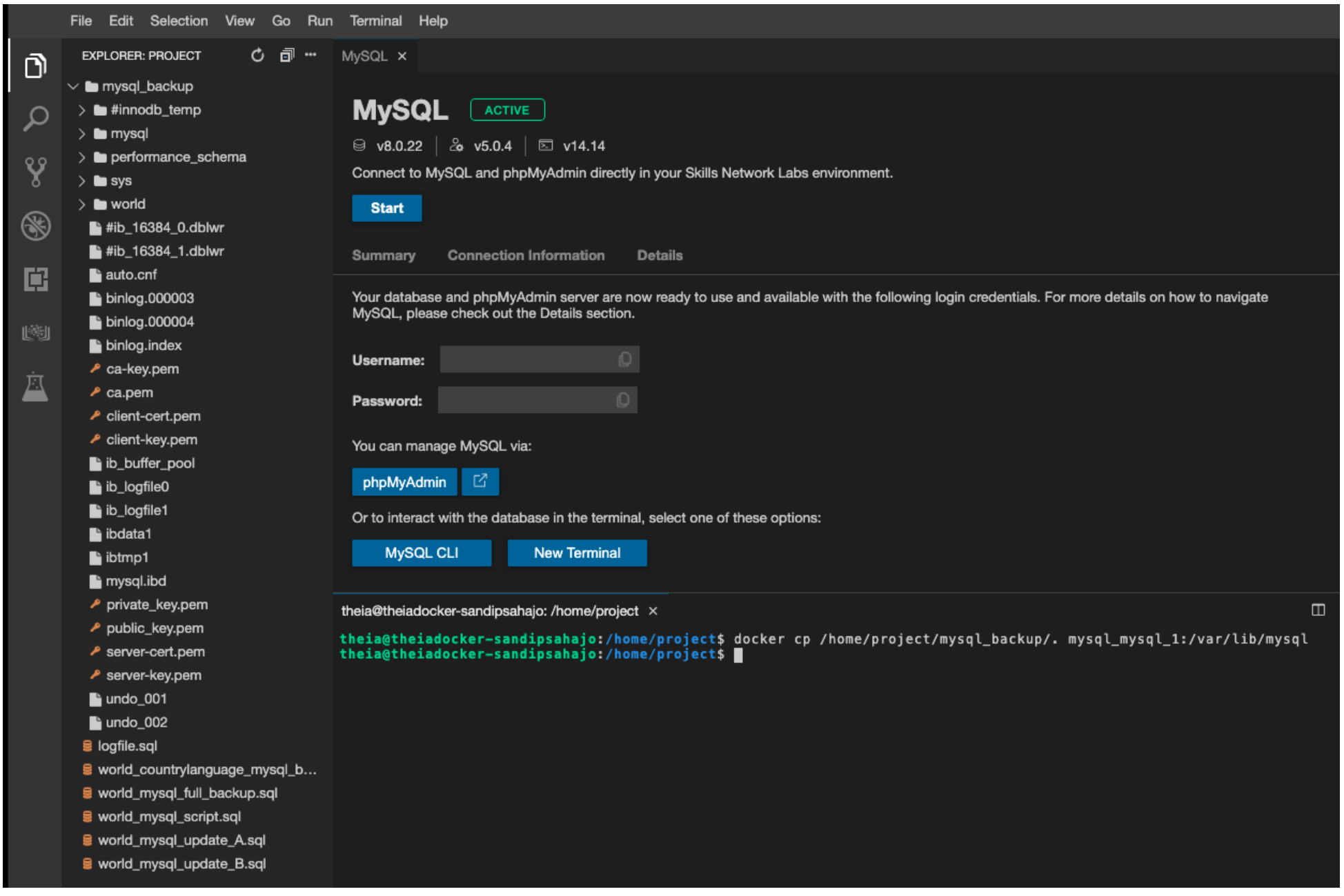
3. Click the **Explorer** icon as shown below in the Cloud IDE to access the user session storage. **mysql_backup** folder will appear which you created as physical backup in step-2. You can click the folder to explore the contents of the created physical backup.



4. When needed, you can restore the physical backup using the command below in the terminal:

```
docker cp /home/project/mysql_backup/. mysql-mysql-1:/var/lib/mysql
```

> **NOTE:** For this exercise, you don't need to run this command.



Practice Exercise 1: Perform Logical Backup and Restore

In this practice exercise, you will practice performing a logical backup and restore of a database table.

> Scenario: *You are planning to update and migrate one of the tables from your **world** database to a new MySQL server. Perform a logical backup of the table **city** from the database **world**. The backup table is expected to contain data of **Bangladesh**. Validate if your created backup is in working state.*

▶ Hint (Click Here)
▶ Solution (Click Here)

Practice Exercise 2: Perform Physical Backup and Restore

In this practice exercise, you will practice performing a physical backup and restore of a database table.

> Scenario: *Perform a physical backup of the database **world**. The backup database is expected to contain data of **Canada** as well as **Bangladesh**.*

▶ Hint (Click Here)
▶ Solution (Click Here)

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

Author(s)

- Sandip Saha Joy

Other Contributor(s)

- David Pasternak

Changelog

Date	Version	Changed by	Change Description
2021-06-15	1.0	Sandip Saha Joy	Created initial version
2021-10-04	1.1	David Pasternak	Updated screenshots
2022-07-12	1.2	Lakshmi Holla	Updated html tag