Laboratory 1

Title of the Laboratory Exercise: DDL and DML commands

1. Introduction and Purpose of Experiment

Structured Query Language (SQL) is used to pass the query to retrieve and manipulate the information from database. Depending upon the nature of query, SQL is divided into different components such as Data Definition Language (DDL) and Data Manipulation Language (DML). DDL statements create the database, maintain the structure of the database and remove database objects such as tables, indexes, and users. DML statements are used for managing data in database such as insert tuples into, delete tuples from, and modify tuples in the database. By doing this lab, students will be able to execute DDL and DML commands

2. Aim and Objectives

Aim

- To execute Data Definition Language (DDL) and Data Management Language (DML) commands

Objectives

At the end of this lab, the student will be able to

- Create a database and populate it with data using SQL commands
- Execute DDL and DML commands for the given database

3. Experimental Procedure

i. Analyse the problem statement
ii. Execute DDL and DML commands
iii. Create a database for the given schema
iv. Design SQL commands using DDL and DML commands
v. Test the executed commands
vi. Analyse and discuss the outcomes of your experiment
vii. Document the work

4. Questions

i. Practice DDL and DML commands
ii. Consider the following relational schema that keeps track of the employees in a company. Enter at least five tuples for the relation. Assume appropriate domain and data type for each field.

EMPLOYEE (EmpId, EmpName, EmpAddress)

Execute the following queries based on the above schema

i. Display the details of all the employees
ii. Display the name and address of the employee with id=101
iii. Insert a new employee<105, 'Bob', 'Bangalore'>
iv. Change the address of the employeeBob to 'Delhi'
v. Delete the details of an employee with employee id=105
vi. Add a column to the schema EMPLOYEE with appropriate data type

5. Calculations/Computations/Algorithms

Command to open mySQL(in Mac terminal) : `mysql -u root -p`

Connect to Database

1. To show Database

```
>> Show Databases;
```

2. To use Database

```
>> Use <Database_Name>;
```

3. To show tables

```
>> Show Tables;
```

DDL Commands

1. **Create** command:

Create command is used to create a table in the database or a database itself. It defines each column of the table uniquely. Each column has minimum of three attributes, a name, data type and size.

1.1. To create a database:

```
>> Create Database <name>;
```

1.2. To create a table:

```
>> CREATE TABLE <table_name>(<col1><datatype>(<size>),..);
```

Ex: `>> CREATE TABLE Employee(No int (2), id char (10));`

2. **Describe** Command (`DESC`):

To describe the structure (column and data types) of an existing database, table, index, or view.

```
>> DESC Table_Name;
```

3. **Rename** Command

Used to change the name of the table.

```
>> RENAME Table <Old_Table_Name> TO <New_Table_Name>
```

4. **Drop** Command

   If a table is dropped, all records held within it are lost and cannot be recovered

   ```
   >> Drop Table <Name>;
   ```

5. **Alter** Command

   To modify an existing database object. The structure of the table can be changed using this command like add new column, change the width of a datatype, change the datatype of a column.

   5.1. Adding New Columns

   ```
   >> ALTER TABLE <name> ADD(<col_name> <datatype> (size);
   ```

   Ex: `>> ALTER TABLE Emp ADD(Salery <int> (10);`

   5.2. Dropping a columns from a table

   ```
   >> ALTER TABLE <name> Drop(<col_name>);
   ```

   Ex: `>> ALTER TABLE <Emp> Drop(Salery);`

   5.3. Modifying Existing columns

   ```
   >> ALTER TABLE <name> Modify(<col_name> <DataType> <size>);
   ```

   Ex: `>> ALTER TABLE <Emp> Modify(Salery Char (10);`

6. **Truncate** Command

   To destroy the data in an existing database, table, index, or view. If a table is truncated all records held within it are lost and cannot be recovered but the table structure is maintained.

   ```
   >> Truncate Table <Name>;
   ```

DML Commands

1. **Insert** Command

   Insert statement adds one or more records to any single table in a relational database. It is used to insert data into a table.

   ```
   >> Insert into <table_name> values(expr1,expr2,..);
   ```

Ex: `>> Insert into Employee values(100,'Smith');`

2. **Update** Command

   Update statement that changes the data of one or more records in a table. It is used to update a row of a table. Either all the rows can be updated, or a subset may be chosen using a condition.

   ```
   >> Update <table_name> set <column_name> = <value>
          Where <condition>;
   ```

Ex: `>> Update Employee Set emp_id = 'John' Where Emp_no = 100;`

3. **Delete** Command

   Delete statement removes one or more records from a table. A subset may be defined for deletion using a condition, otherwise all records are removed.

   ```
   >> Delete from <table_name> Where <condition>;
   ```

Ex: `>> Delete from Employee Where Emp_no = 100;`

4. **Select** Command

   Select statement is used to query or retrieve data from a table in the database.

   4.1. To get all the elements

   ```
   >> Select *from <table_name>;
   ```

   Ex: `>> Select *from Employee;`

   4.2. To get particular column

   ```
   >> Select <column_name> from <table_name>;
   ```

   Ex: `>> Select Emp_id from Employees;`

6. Presentation of Results

```
Rajat-PC:~ RAJAT_SINGH$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.0.18 MySQL Community Server - GPL

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> Show Databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| Rajat              |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql> use Rajat;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----------------+
| Tables_in_rajat |
+-----------------+
| Employee        |
+-----------------+
1 row in set (0.00 sec)

mysql>
```

Fig. 1.1 Connecting to a database

```
mysql> /* Using 'create' command to create a database;
    /*> To create a table;
    /*> Table need a seperate declaration. */
mysql> Create Database RUAS;
Query OK, 1 row affected (0.00 sec)

mysql> -- Database is been created and named as RUAS --
mysql> show Databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| Rajat              |
| RUAS               |
| sys                |
+--------------------+
6 rows in set (0.00 sec)

mysql> -- Now we will create a Table in our Database --
mysql> use RUAS;
Database changed
mysql> Create Table Employees(No int (4), Emp_id char (11), Emp_name char (20), Emp_Addr char (20));
Query OK, 0 rows affected, 1 warning (0.02 sec)

mysql> show tables;
+----------------+
| Tables_in_ruas |
+----------------+
| Employees      |
+----------------+
1 row in set (0.00 sec)

mysql> Desc Employees;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| No       | int(4)   | YES  |     | NULL    |       |
| Emp_id   | char(11) | YES  |     | NULL    |       |
| Emp_name | char(20) | YES  |     | NULL    |       |
| Emp_Addr | char(20) | YES  |     | NULL    |       |
+----------+----------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Newly Created Database

Newly Created Table

Fig. 1.2 DDL Command – I

```
mysql> Desc Employees;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| No       | int(4)   | YES  |     | NULL    |       |
| Emp_id   | char(11) | YES  |     | NULL    |       |
| Emp_name | char(20) | YES  |     | NULL    |       |
| Emp_Addr | char(20) | YES  |     | NULL    |       |
+----------+----------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> Rename Table Employees to Emp;
Query OK, 0 rows affected (0.02 sec)

mysql> show tables;
+----------------+
| Tables_in_ruas |
+----------------+
| Emp            |
+----------------+
1 row in set (0.00 sec)
```

changes

Fig. 1.3 DDL Command – II

```
mysql> show tables;
+-----------------+
| Tables_in_ruas |
+-----------------+
| Emp             |
| Waste           |
+-----------------+
2 rows in set (0.00 sec)

mysql> Drop table Waste;
Query OK, 0 rows affected (0.01 sec)

mysql> show tables;
+-----------------+
| Tables_in_ruas |
+-----------------+
| Emp             |
+-----------------+
1 row in set (0.00 sec)
```

Fig. 1.4 Drop Command

```
mysql> Alter Table Emp add(Salary int (10));
Query OK, 0 rows affected, 1 warning (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 1

mysql> Desc Emp;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| No       | int(4)   | YES  |     | NULL    |       |
| Emp_id   | char(11) | YES  |     | NULL    |       |
| Emp_name | char(20) | YES  |     | NULL    |       |
| Emp_Addr | char(20) | YES  |     | NULL    |       |
| Salary   | int(10)  | YES  |     | NULL    |       |
+----------+----------+------+-----+---------+-------+
5 rows in set (0.00 sec)

mysql> -- Above we have shown how to add another column to it --
mysql> -- Now for same we need to drop that additional column --
mysql> Alter Table Emp Drop Salary;
Query OK, 0 rows affected (0.13 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> Desc Emp;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| No       | int(4)   | YES  |     | NULL    |       |
| Emp_id   | char(11) | YES  |     | NULL    |       |
| Emp_name | char(20) | YES  |     | NULL    |       |
| Emp_Addr | char(20) | YES  |     | NULL    |       |
+----------+----------+------+-----+---------+-------+
4 rows in set (0.01 sec)

mysql> -- now we'll modify the size of Emp_id form '11' to '12' --
mysql> Alter table Emp modify Emp_id char (12);
Query OK, 4 rows affected (0.03 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> Desc Emp;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| No       | int(4)   | YES  |     | NULL    |       |
| Emp_id   | char(12) | YES  |     | NULL    |       |
| Emp_name | char(20) | YES  |     | NULL    |       |
| Emp_Addr | char(20) | YES  |     | NULL    |       |
+----------+----------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

Fig. 1.5 Alter Command

```
mysql> insert into Emp(No, Emp_id, Emp_name, Emp_Addr) Values(1, '17ETCS2138', 'Rajat Kumar Singh','Bangalore' );
Query OK, 1 row affected (0.01 sec)

mysql> Select *from Emp;
+------+------------+------------------+-----------+
| No   | Emp_id     | Emp_name         | Emp_Addr  |
+------+------------+------------------+-----------+
|    1 | 17ETCS2138 | Rajat Kumar Singh | Bangalore |
+------+------------+------------------+-----------+
1 row in set (0.00 sec)
```

Fig. 1.6 DML command (Insert and Select)

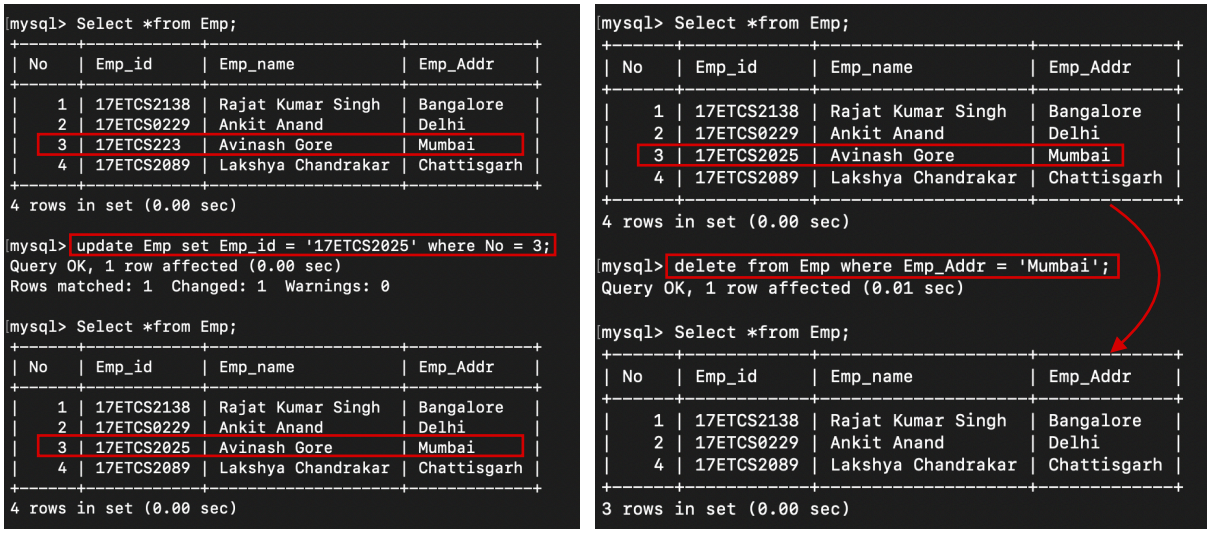

Fig. 1.7 DML command (Update and Delete)

7. Analysis and Discussions

From the following experiment we have analysed that the Data definition Languages(DDL) and Data manipulation Languages(DML) together form a Database language.

DDL is used to specify the database schema(representation) database structure, on the other hand, DML is used to access, modify or retrieve the data from the database.

Note: for syntax and all see the Algorithm section.

8. Conclusions

From the following experiment we have concluded that the basic difference between DDL and DML is that, the DML are further classified to procedural and non-procedural, where as the DLL is not further classified. For forming the database language both DDL and DML is necessary. As they both will be required to form and access the databases.

9. Comments

a. Limitations of Experiments

The experiment is limited to some extent of usage of DDL and DML command to operate with only databases and tables in the databases. We didn't required to operate with other functionalities of databases.

b. Limitations of Results

The results are limited to the formation of tables in the databases and their representation in Terminal.

c. Learning happened

From this we have learnt about the creation and management of databases using mysql terminal commands and acknowledged their flow accordingly.

d. Recommendations

For the databases operation I would recommend to operate with the limited range of data to be easily operable and if we select the complete data and suppose if the data consist millions of entry it would keep on loading and consumes extra CPU usage. therefore to operate with the databases we should choose limited range and field to operate at a time.