MIT World Peace University

Database Management Systems
Second Year B. Tech, Semester 4

Learning SQL DML Commands
Data Manipulation Language

Assignment No. 3

Prepared By

Krishnaraj Thadesar
Cyber Security and Forensics
Batch A1, PA 20

March 2, 2023

Contents

1 Aim

Write suitable DML and select command to manipulate and retrieve requested data from tables.

2 Objectives

1. DML (Insert, Update, Delete) commands,
2. SQL Select- Logical, IN, Negation, NULL, Comparison Operators.
3. Where Clause, Between AND, Exists, ALL, LIKE

3 Problem Statement

4 Theory

4.1 SQL Data Manipulation Language (DML)

4.1.1 What is Data Manipulation Language?

Data Manipulation Language (DML) is a computer language used to access and manipulate data stored in a database. It is used to retrieve, insert, update, and delete data in a database.

4.1.2 DML Commands

The following are the Commands that are used in DML:

1. SELECT - Retrieves data from a database.
2. INSERT - Inserts data into a table.
3. UPDATE - Updates existing data within a table.
4. DELETE - Deletes existing data within a table.

4.2 DML Command Syntax and Examples

1. SELECT - Retrieves data from a database.

```
SELECT column1, column2, ...
FROM table_name;
```

2. INSERT - Inserts data into a table.

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

3. UPDATE - Updates existing data within a table.

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

4. DELETE - Deletes existing data within a table.

   ```
   DELETE FROM table_name WHERE condition;
   ```

4.3 SELECT query

4.3.1 What is SELECT query?

The SELECT statement is used to select data from a database. The data returned is stored in a result table, called the result-set.

4.3.2 SELECT Syntax

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value;
```

4.3.3 SELECT Operators

The following are the Operators that are used in SELECT:

1. **AND** - Returns rows where both conditions are true.
2. **OR** - Returns rows where either condition is true.
3. **NOT** - Returns rows where the condition(s) is not true.
4. **BETWEEN** - Returns rows where the value is within a range of two values.
5. **LIKE** - Returns rows where the value matches a pattern.
6. **IN** - Returns rows where the value matches any value in a list.

4.3.4 Examples of the SELECT Query

1. `SELECT * FROM CUSTOMERS;`
2. `SELECT * FROM CUSTOMERS WHERE CUST_ID = 1;`
3. `SELECT * FROM CUSTOMERS WHERE CUST_ID = 1 AND CUST_NAME = 'Krishnaraj';`
4. `SELECT * FROM CUSTOMERS WHERE CUST_ID = 1 OR CUST_NAME = 'Krishnaraj';`
5. `SELECT * FROM CUSTOMERS WHERE NOT CUST_ID = 1;`
6. `SELECT * FROM CUSTOMERS WHERE CUST_ID BETWEEN 1 AND 5;`
7. `SELECT * FROM CUSTOMERS WHERE CUST_NAME LIKE 'Krish%';`
8. `SELECT * FROM CUSTOMERS WHERE CUST_ID IN (1, 2, 3);`

4.4 SQL Operators

4.4.1 What are SQL Operators?

Operators are special symbols in SQL that allow you to perform specific operations on data.

4.4.2 SQL Operators

The following are the Operators that are used in SQL:

1. **Arithmetic Operators** - Used to perform mathematical operations on numbers.
2. **Comparison Operators** - Used to compare values.
3. **Logical Operators** - Used to combine two or more conditions.
4. **Misc Operators** - Used to perform other operations.

4.4.3 Arithmetic Operators

The following are the Arithmetic Operators that are used in SQL:

1. **+** - Addition
2. **-** - Subtraction
3. * - Multiplication
4. / - Division
5. **MOD** - Modulus

4.4.4 Comparison Operators

The following are the Comparison Operators that are used in SQL:

1. **=** - Equal
2. **<>** - Not equal. Note: In some versions of SQL this operator may be written as !=
3. **>** - Greater than
4. **<** - Less than
5. **>=** - Greater than or equal
6. **<=** - Less than or equal
7. **BETWEEN** - Between an inclusive range
8. **LIKE** - Search for a pattern
9. **IN** - To specify multiple possible values for a column

4.4.5 Logical Operators

The following are the Logical Operators that are used in SQL:

1. **AND** - Logical AND
2. **OR** - Logical OR
3. **NOT** - Logical NOT

5 Platform

Operating System: Arch Linux x86-64
IDEs or Text Editors Used: Draw.io for Drawing the ER diagram.

6 Input

Given Database from the Problem Statement for the Assignment for our batch. (A1 PA 20)

7 Output

```
MariaDB [dbms_lab]> create database Company;
Query OK, 1 row affected (0.001 sec)

MariaDB [dbms_lab]> show databases;
+--------------------+
| Database           |
+--------------------+
| Company            |
| class              |
| class_stuff        |
| dbms_lab           |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test               |
| test_libreoffice   |
+--------------------+
10 rows in set (0.001 sec)

MariaDB [dbms_lab]> use Company;
Database changed
MariaDB [Company]> create table emp(empno int primary key, empname varchar(50) not
    null, job varchar(10), mgr int not null, hiredate date, sal int not null, comm
    int, deptno int not null);
Query OK, 0 rows affected (0.008 sec)

MariaDB [Company]> describe emp;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| empno    | int(11)     | NO   | PRI | NULL    |       |
| empname  | varchar(50) | NO   |     | NULL    |       |
```

```
| job      | varchar(10) | YES  |     | NULL    |       |
| mgr      | int(11)     | NO   |     | NULL    |       |
| hiredate | date        | YES  |     | NULL    |       |
| sal      | int(11)     | NO   |     | NULL    |       |
| comm     | int(11)     | YES  |     | NULL    |       |
| deptno   | int(11)     | NO   |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
8 rows in set (0.002 sec)

MariaDB [Company]> create table dept(deptno int primary key, dname varchar(50),
    loc varchar(50) not null);
Query OK, 0 rows affected (0.008 sec)

MariaDB [Company]> describe dept;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| deptno | int(11)     | NO   | PRI | NULL    |       |
| dname  | varchar(50) | YES  |     | NULL    |       |
| loc    | varchar(50) | NO   |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.002 sec)

MariaDB [Company]> insert into emp values (7369,  "Smith",  "Clerk",  7902,  "
    1980-12-17",  800,  300,  20);
Query OK, 1 row affected (0.001 sec)

MariaDB [Company]> select * from emp;
+-------+---------+-------+------+------------+-----+------+--------+
| empno | empname | job   | mgr  | hiredate   | sal | comm | deptno |
+-------+---------+-------+------+------------+-----+------+--------+
|  7369 | Smith   | Clerk | 7902 | 1980-12-17 | 800 |  300 |     20 |
+-------+---------+-------+------+------------+-----+------+--------+
1 row in set (0.001 sec)

MariaDB [Company]> insert into emp values (7499, "Allen", "Salesman", 7698, "
    1981-02-20", 1600, 300, 30);
Query OK, 1 row affected (0.001 sec)

MariaDB [Company]> select * from emp;
+-------+---------+----------+------+------------+------+------+--------+
| empno | empname | job      | mgr  | hiredate   | sal  | comm | deptno |
+-------+---------+----------+------+------------+------+------+--------+
|  7369 | Smith   | Clerk    | 7902 | 1980-12-17 |  800 |  300 |     20 |
|  7499 | Allen   | Salesman | 7698 | 1981-02-20 | 1600 |  300 |     30 |
+-------+---------+----------+------+------------+------+------+--------+
2 rows in set (0.000 sec)

MariaDB [Company]> insert into dept values(10, "Accounting", "New York");
Query OK, 1 row affected (0.001 sec)

MariaDB [Company]> insert into dept values(20, "Research", "Dallas");
Query OK, 1 row affected (0.001 sec)

MariaDB [Company]> insert into dept values
    -> (30, "Sales", "Chicago");
Query OK, 1 row affected (0.001 sec)

MariaDB [Company]> insert into dept values(40, "Operations", "Boston");
```

```
Query OK, 1 row affected (0.001 sec)

MariaDB [Company]> select * from dept;
+--------+------------+----------+
| deptno | dname      | loc      |
+--------+------------+----------+
|     10 | Accounting | New York |
|     20 | Research   | Dallas   |
|     30 | Sales      | Chicago  |
|     40 | Operations | Boston   |
+--------+------------+----------+
4 rows in set (0.001 sec)

MariaDB [Company]> select * from emp;
+-------+---------+----------+------+------------+------+------+--------+
| empno | empname | job      | mgr  | hiredate   | sal  | comm | deptno |
+-------+---------+----------+------+------------+------+------+--------+
|  7369 | Smith   | Clerk    | 7902 | 1980-12-17 |  800 |  300 |     20 |
|  7499 | Allen   | Salesman | 7698 | 1981-02-20 | 1600 |  300 |     30 |
+-------+---------+----------+------+------------+------+------+--------+
2 rows in set (0.001 sec)
```

8 Conclusion

Thus, we have learned SQL DML commands, SELECT Command with SQL operators thoroughly.

9 FAQ

1. **What is the difference between Truncate table and Drop table command?**

   (a) *Truncate table command deletes all the records from the table and resets the identity column to 1.*

   (b) *Drop table command deletes the table and all the records from the table.*

   (c) *Truncate table command is faster than Drop table command.*

   (d) *Truncate table command cannot be rolled back.*

   (e) *Drop table command can be rolled back.*

   Example:

   (a) *Truncate table command*

   ```
   Truncate table CUSTOMERS;
   ```

   (b) *Drop table command*

   ```
   Drop table CUSTOMERS;
   ```

2. **How is the pattern matching done in the SQL?**

   (a) *The pattern matching is done using the LIKE operator.*

   (b) *The pattern matching is done using the wildcard characters.*

   (c) *The wildcard characters are:*

   - *% - Represents zero or more characters.*
   - *_ - Represents a single character.*
   - *[charlist] - Represents any single character in charlist.*

   The Syntax of the command is:

   ```
   SELECT column_name(s) FROM table_name WHERE column_name LIKE pattern;
   ```

   Example:

   ```
   SELECT * FROM CUSTOMERS WHERE CUST_NAME LIKE 'Emp%';
   SELECT * FROM STUDENTS WHERE CUST_NAME LIKE 'AssignmentNumber_';
   ```

3. **Write a DELETE command to delete all the records from CUSTOMERS table.**

   ```
   DELETE FROM CUSTOMERS;
   ```