AROR UNIVERSITY
OF ART, ARCHITECTURE,
DESIGN & HERITAGE,
SUKKUR, SINDH

Department of Artificial Intelligence & Multimedia Gamming

CSC-207: Database Systems

Lab # 10: To Work with SQL DML Queries

Objectives

1. Introduction to DML Statements
2. INSERT
3. UPDATE
4. DELETE

Introduction of DML Statements

- The SQL commands that deal with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements.

Examples of DML:

INSERT – is used to insert data into a table.

UPDATE – is used to update existing data within a table.

DELETE – is used to delete records from a database table.

Insert Command

MySQL INSERT statement is used to insert record(s) or row(s) into a table. The insertion of records or rows in the table can be done in two ways, insert a single row at a time, and insert multiple rows at a time.

The INSERT INTO statement is used to insert new records in a table. It is possible to write the INSERT INTO statement in two ways.

The first way specifies both the column names and the values to be inserted:

Syntax:

```
INSERT INTO table_name ( field1, field2,...fieldN ) Values (value2, value2, …, valueN);
```

Another way is if you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query. However, make sure the order of the values is in the same order as the columns in the table. The INSERT INTO syntax would be as follows:

```
INSERT INTO table_name Values (value2, value2, …, valueN);
```

Example

```
CREATE TABLE Professors (
   ProfessorID INT PRIMARY KEY,
   FirstName VARCHAR(50),
   LastName VARCHAR(50),
   Email VARCHAR(100) -- Example additional column
);
INSERT INTO Professors (ProfessorID, FirstName, LastName, Email)
VALUES
   (1, 'John', 'Smith', 'john.smith@example.com'),
   (2, 'Emily', 'Johnson', 'emily.johnson@example.com'),
   (3, 'Sana', 'Ahmed', 'sana.ahmed@example.com');
```

- **Insert Data Only in Specified Columns Example**

```
INSERT INTO Professors (ProfessorID, FirstName)
VALUES (2, 'Jack', );
```

• INSERT INTO SELECT

To insert data from other tables into a table, you use the following SQL INSERT INTO SELECT statement:

```
INSERT INTO table_name(column_list)
SELECT
  select_list
FROM
  another_table
```

```
WHERE    condition;
```

Example:

```
-- Create the Employees table
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   FirstName VARCHAR(50),
   LastName VARCHAR(50),
   IsRetired BOOLEAN
);

-- Create the RetiredEmployees table
CREATE TABLE RetiredEmployees (
   EmployeeID INT PRIMARY KEY,
   FirstName VARCHAR(50),
   LastName VARCHAR(50),
   IsRetired BOOLEAN
);

-- Insert data into the Employees table
INSERT INTO Employees (EmployeeID, FirstName, LastName, IsRetired)
VALUES
   (1, 'James', 'Smith', TRUE),
   (2, 'Olivia', 'Johnson', FALSE),
   (3, 'Emma', 'Williams', TRUE),
   (4, 'Noah', 'Brown', FALSE),
   (5, 'Sophia', 'Davis', TRUE);

-- Use INSERT INTO SELECT to insert data from Employees to RetiredEmployees for employees who are retired
INSERT INTO RetiredEmployees (EmployeeID, FirstName, LastName)
SELECT EmployeeID, FirstName, LastName
FROM Employees
```

```
WHERE IsRetired = TRUE;

-- Verify the contents of the RetiredEmployees table
SELECT * FROM RetiredEmployees;
```

Note:

As long as the data types and order of the columns match, you can use INSERT INTO SELECT to insert data into tables with different column names.

Note:

To insert data from other tables into a table, you use the following SQL INSERT INTO SELECT statement:

```
INSERT INTO RetiredEmployees (EmployeeID, FirstName, LastName)
SELECT EmployeeID, FirstName, LastName
FROM Employees;
```

It will copy all records and insert it into RetiredEmployees.

This will copy DDL and DML of professors table for newfaculty table.

```
create table newfaculty
select * from professors;

select * from newfaculty ;
```

• Using SELECT statement in the VALUES list

Example

```
-- Step 1: Create the Orders table
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderAmount DECIMAL(10, 2),
    OrderDate DATE
);
```

```
-- Step 2: Create the OrderSummary table
CREATE TABLE OrderSummary (
    SummaryID INT PRIMARY KEY AUTO_INCREMENT,
    TotalOrders INT,
    AverageOrderAmount DECIMAL(10, 2),
    MostRecentOrderDate DATE
);

-- Step 3: Insert data into the Orders table
INSERT INTO Orders (OrderID, CustomerID, OrderAmount, OrderDate)
VALUES
    (1, 100, 150.00, '2024-04-01'),
    (2, 101, 200.00, '2024-04-02'),
    (3, 100, 300.00, '2024-04-03'),
    (4, 102, 250.00, '2024-04-04'),
    (5, 103, 350.00, '2024-04-05');

-- Step 4: Use INSERT INTO ... VALUES with SELECT subqueries in the VALUES list
-- Insert a new record into the OrderSummary table
INSERT INTO OrderSummary (TotalOrders, AverageOrderAmount, MostRecentOrderDate)
VALUES (
    -- Retrieve the count of orders from Orders table as TotalOrders
    (SELECT COUNT(*) FROM Orders),
    -- Retrieve the average order amount from Orders table as AverageOrderAmount
    (SELECT AVG(OrderAmount) FROM Orders),
    -- Retrieve the most recent order date from Orders table as MostRecentOrderDate
    (SELECT MAX(OrderDate) FROM Orders)
);

-- Step 5: Verify the contents of the OrderSummary table
SELECT * FROM OrderSummary;
```

UPDATE Command

The UPDATE statement modifies existing data in a table. You can also use the UPDATE statement change values in one or more columns of a single row or multiple rows.

The following illustrates the basic syntax of the UPDATE statement:

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition1 AND condition2 ... AND conditionN;
```

Note:

- You can update one or more field altogether.
- You can specify any condition using the WHERE clause.
- You can update the values in a single table at a time.
- The WHERE clause is very useful when you want to update the selected rows in a table.

Example:

```
-- Create the Employees table
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   Name VARCHAR(50),
   Salary DECIMAL(10, 2),
   Department VARCHAR(50)
);

-- Insert data into the Employees table
INSERT INTO Employees (EmployeeID, Name, Salary, Department)
VALUES
   (1, 'Alice', 45000.00, 'Sales'),
   (2, 'Bob', 52000.00, 'Marketing'),
   (3, 'Carol', 47000.00, 'Sales'),
   (4, 'David', 60000.00, 'HR');

-- Update the salary of employees in the Sales department
UPDATE Employees
SET Salary = Salary * 1.1 -- Increase salary by 10%
```

```
WHERE Department = 'Sales';
```

- ## UPDATE to replace string example

```
UPDATE table_name
SET column_name = REPLACE(column_name, old_string, new_string)
WHERE condition;
```

Example:

```
UPDATE employee
SET email = REPLACE(email,'@colorado.edu','@aror.edu.pk')
WHERE empNo='E100';
UPDATE professors
SET email = REPLACE(email,'@example.com','@aror.edu.pk')
WHERE professorid=6;
```

- ## Using MySQL UPDATE with SELECT statement

Syntax:

```
UPDATE target_table
SET column1 = (SELECT value1 FROM source_table WHERE condition),
   column2 = (SELECT value2 FROM source_table WHERE condition),
   ...
WHERE update_condition;
```

Example:

```
CREATE TABLE tasks (
   task_id INT PRIMARY KEY,
   description VARCHAR(255)
);

CREATE TABLE employee(
   empNo VARCHAR(255) PRIMARY KEY,
   department VARCHAR(255)
```

```
);

-- Insert data into the tasks table
INSERT INTO tasks (task_id, description)
VALUES
   (1, NULL),  -- task with no description
   (2, 'Task 2');  -- task with description

-- Insert data into the employee table
INSERT INTO employee (empNo, department)
VALUES
   ('E100', 'Sales'),  -- employee E100 in the Sales department
   ('E200', 'HR');  -- another employee in the HR department

-- Update the description in the tasks table based on the department from the employee table
UPDATE tasks
SET description = (
   SELECT department
   FROM employee
   WHERE empNo = 'E100'
)
WHERE description IS NULL;

select * from tasks;
```

DELETE Command

If you want to delete a record from any MySQL table, then you can use the SQL command DELETE FROM.

Syntax:

DELETE FROM table_name [WHERE Clause]

Note:

If the WHERE clause is not specified, then all the records will be deleted from the given MySQL table.

The WHERE clause is very useful when you want to delete selected rows in a table.

Example:

```
DELETE FROM employees
WHERE empNo = 'E101';
```

To delete all rows from the *employee* table, you use the DELETE statement without the WHERE clause as follows:

```
DELETE FROM employee;
```

Note:

The TRUNCATE TABLE statement is a faster way to empty a table than a DELETE statement with no WHERE clause.

```
TRUNCATE TABLE employee;
```

```
DROP TABLE employee;
```

MySQL DELETE with LIMIT and ORDER BY clause example:

```
DELETE FROM tasks
ORDER BY title DESC LIMIT 2;
```

MySQL DELETE Multiple tables with JOIN example:

```
DELETE t
FROM tasks t
JOIN employee e ON t.description = e.department;
```

Exercises (Class)

1. Add here all the tasks performed in lab.

Exercises (Weekly)

Run SQL statement to create a table projects and student.

CREATE TABLE projects(project_id INT AUTO_INCREMENT, name VARCHAR(100) NOT NULL, start_date DATE, end_date DATE,

PRIMARY KEY(project_id));

CREATE TABLE student(std_NO varchar(8) not null, std_Name varchar(30) not null, Department varchar(30) not null, email varchar(30) not null, phone varchar(30) , project_id int ,

FOREIGN KEY (project_id) REFERENCES projects (project_id),

PRIMARY KEY (std_No));

1. Write a SQL statement to insert 4 rows in project table and 4 rows in student table below by a single insert statement.

project_id	name	start_date	end_date
1	AI for Marketing	2019-08-01	2019-12-31
2	ML for Sales	2019-05-15	2019-11-20
3	CS for IT	2020-01-01	2020-05-20
4	SQL for input	2020-06-13	2020-11-20

std_NO	std_Name	Department	email	phone	project_id
S100	Ali Mehmood	Administration	ali@iba-suk.edu.pk	0333-895311	3
S101	Manisha Kataria	Computer Science	manisha@iba-suk.edu.pk	0345-111333444	2
S102	Sagar Sanjay	Engineering	sagar@iba-suk.edu.pk	0300-22224454	2
S103	Sara Shaikh	IT	sara@iba-suk.edu.pk	0300-111110000	3

Now Create duplicate of projects table named projects_copy with all structure and data

CREATE TABLE IF NOT EXISTS projects_copy

SELECT * FROM projects;

2. Write a SQL statement to delete all records from projects table.
3. Write a SQL statement to insert rows from projects_copy table to projects table.
4. Write a SQL statement to update start_date to '2023-02-01' of a project name CS for IT.

Now add a column in table project by following command.

ALTER TABLE projects

ADD cost INT;

ALTER TABLE student

ADD daysToComplete INT;

5. Write a SQL statement to update cost of project to 90000 where cost are null.

6. Write a SQL statement to update daysToComplete column from student by calculating difference from project start_date and end_date.