Labsheet-03

1. Insert One employee record using all Columns.

```
INSERT INTO employee (emp-id, name, department,
    Salary, hire-date)
  VALUES(101, 'John Smith', 'IT', 55000,'2024-01-15');
```

Output

1 row inserted

2. Insert multiple employee records in one Statement.

```
INSERT INTO employee (emp-id, name,
    department, Salary, hire-date)
VALUES
(102, 'Sarah Johnson', 'HR', 48000, '2024-02-10');
(103, 'Mike Davis', 'Finance', 52000, '2024-01-20');
```

Output

2 rows inserted

3. Insert a record using only selected Columns

```
Insert INTO employees(emp-id, name, department)
Values (104, 'Lisa Wilson', 'Marketing');
```

Output

```
1 row inserted
```

4. View table Contents.

```
SELECT * FROM Employees;
```

Output

```
emp-id   name           department   Salary   hire-date
101      John Smith     IT           55000    24-01-15
102      Sarah Johnson  HR           48000    24-02-10
103      Mike Davis     Finance      52000    24-01-20
104      Lisa Wilson    Marketing    NULL     NULL
```

2. Using Employee data (added extra rows as per question):

1. Display all employees

```
SELECT * FROM employees;
```

2. Names and Salaries

```
SELECT name, Salary FROM employees;
```

3. IT department

```
SELECT * FROM Employees WHERE department
    = 'IT';
```

4. Salary > 50000

```
SELECT * FROM employees WHERE Salary > 50000;
```

5. Hired in January 2024

```
SELECT * FROM employees WHERE hire_date
  LIKE '2024-01%';
```

6. Distinct departments

```
SELECT DISTINCT department FROM employees;
```

7. Count employees

```
SELECT COUNT (*) AS total_employees FROM employees;
```

8. Max, Min, Avg Salary

```
SELECT MAX(salary), MIN(salary), AVG(Salary) FROM
                                   employees;
```

3. Insert a record using only Selected Columns.

```
Insert INTO employees(emp-id, name, department)
VALUES (104, 'Lisa wilson', 'marketing');
```

Output

```
1 row inserted
```

4. Basic DELETE Operations.

1. Delete products with 0 stock

```
DELETE FROM products WHERE Stock-quality=0;
```

2. Delete Electronics >1000

```
DELETE FROM products WHERE Category =" Electron-
                                ico' And price >1000;
```

3. Delete Desk Chair

```
DELETE FROM products WHERE product_name =
                    'Disk Chair';
```

4. Delete items >₹100

```
DELETE FROM products WHERE price >100;
```

5. Advanced SELECT with Sorting and limiting

```
--1. Sort by marks desc
SELECT * FROM Students ORDER BY marks DESC;
---2. Top 5 scorers
SELECT * FROM Students ORDER BY marks
DESC LIMIT 5;
----3. CS Students alphabetically
SELECT * FROM Students WHERE Course = 'Computer
    Science' ORDER BY Student_name;
----4. Bottom 3 Scores
SELECT * FROM Students ORDER BY marks ASC
    LIMIT 3;
----5. marks between 80-80
SELECT * FROM Students WHERE marks BETWEEN 80
    AND 90;
------6. Names Start with A
SELECT * FROM Students WHERE Student_name
    LIKE 'A%';
----7. Students from Mumbai or Delhi
SELECT * FROM Students WHERE City
 IN ('Mumbai', 'Delhi');
```

6. Insert with Select

```
---1. High value sales >5000
CREATE TABLE high-value-Sales As
SELECT * FROM Sales_data WHERE Sale_amount>
                                  5000;

----2. Salesperson total sales
CREATE TABLE top-performancers As
SELECT salesperson, SUM(sale_amount)
        As total - Sales
  FROM Sales_data GROUP BY Salesperson;
```

7. Complex UPDATE

```
---- 1. Increase Price by 10% if Stack <
          reorder_level

    UPDATE inventory
    SET unit_price = unit_price * 1.10
    WHERE Current_Stock < reorder_level;
---- 2. Update last_updated
    UPDATE inventory
    SET last_updated = CURRENT_TIMESTAMP;
---- 3. Halve Stock if price > 30000
      UPDATE inventory
      SET Current_Stock = Current_Stock/2
      WHERE unit_price > 30000;
```

8. Advanced DELETE

```
----1. Delete Cancelled before 2024
DELETE FROM Customer_orders
WHERE Status='Cancelled' AND order_date
                    <'2024-01-01';

___2. Remove zero amount
DELETE FROM Customer_orders WHERE
          order_amount =0;

_____3. Delete lisa Wilson's Orders
DELETE FROM Customer_orders WHERE
     Customer_name ='lisa Wilson';
```

9. MERGE Operations

```
---- 1. Update prices of existing
     UPDATE master-products m
    JOIN update-Products u ON m product-Code=u
               product-Code.

   SET m.price=u.Price, m.Status=u.Status;

---- 2. Insert new
    INSERT INTO master-products (product-code,
           product-name, Price, Status)

   SELECT u. product-Code, u. product-name, u.price,
        u. Status.

   FROM update-products u
   WHERE u. product-Code NOT IN (SELECT product-
         Code FROM master-products);

---- 3. Mark missing as Discontinued
     UPDATE master-Products
     SET Status = "Discontinued"
   WHERE product-Code NOT IN (SELECT product-
             Code FROM update-products);
```

10. Library Management

```
----1. Books available
SELECT * FROM books WHERE copies_available >0;
----2. Mark Overdue
  UPDATE transactions
    SET Status='overdue'
 WHERE return_date < CURDATE() AND Status
     <>'Returned';
-----3. Insert new book
 INSERT INTO BOOKS VALUES (101,'AI Fundamentals',
                 John Doe', 'Computer Science',
               5, 10);
------4. Summary
  SELECT COUNT(*) AS total_issued FROM
  transactions WHERE Status ='Issued';

 SELECT COUNT (*) AS overdue_books FROM
 transactions WHERE Status ="Overdue";
```

[illegible]
8/10/25