

SQL Joins

INNER JOIN
LEFT JOIN
RIGHT JOIN
FULL OUTER JOIN

SUBMITTED BY:SHALINI T



1. Introduction

The objective of this task is to practice and understand different types of SQL joins using MySQL Workbench. We created two related tables: Customers and Orders, and applied INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL OUTER JOIN to analyze how data is combined from multiple tables.

2. Database Setup

I have created a database named 'practice_db' and defined two tables: Customers and Orders.

SQL Code:

```
CREATE DATABASE practice_db;  
USE practice_db;
```

-- Customers table

```
CREATE TABLE Customers (  
    CustomerID INT PRIMARY KEY,  
    CustomerName VARCHAR(50),  
    City VARCHAR(50)  
);
```

-- Orders table

```
CREATE TABLE Orders (  
    OrderID INT PRIMARY KEY,  
    CustomerID INT,  
    Product VARCHAR(50),  
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)  
);
```

3. Data Insertion

Sample data was inserted into both tables.

```
INSERT INTO Customers (CustomerID, CustomerName, City)  
VALUES  
(1, 'John', 'New York'),  
(2, 'Alice', 'London'),  
(3, 'David', 'Paris'),
```

```
(4, 'Sophia', 'Berlin');
```

```
INSERT INTO Orders (OrderID, CustomerID, Product)
VALUES
(101, 1, 'Laptop'),
(102, 2, 'Phone'),
(103, 1, 'Tablet'),
(104, 5, 'Camera');
```

4. SQL Joins and Results

4.1 INNER JOIN

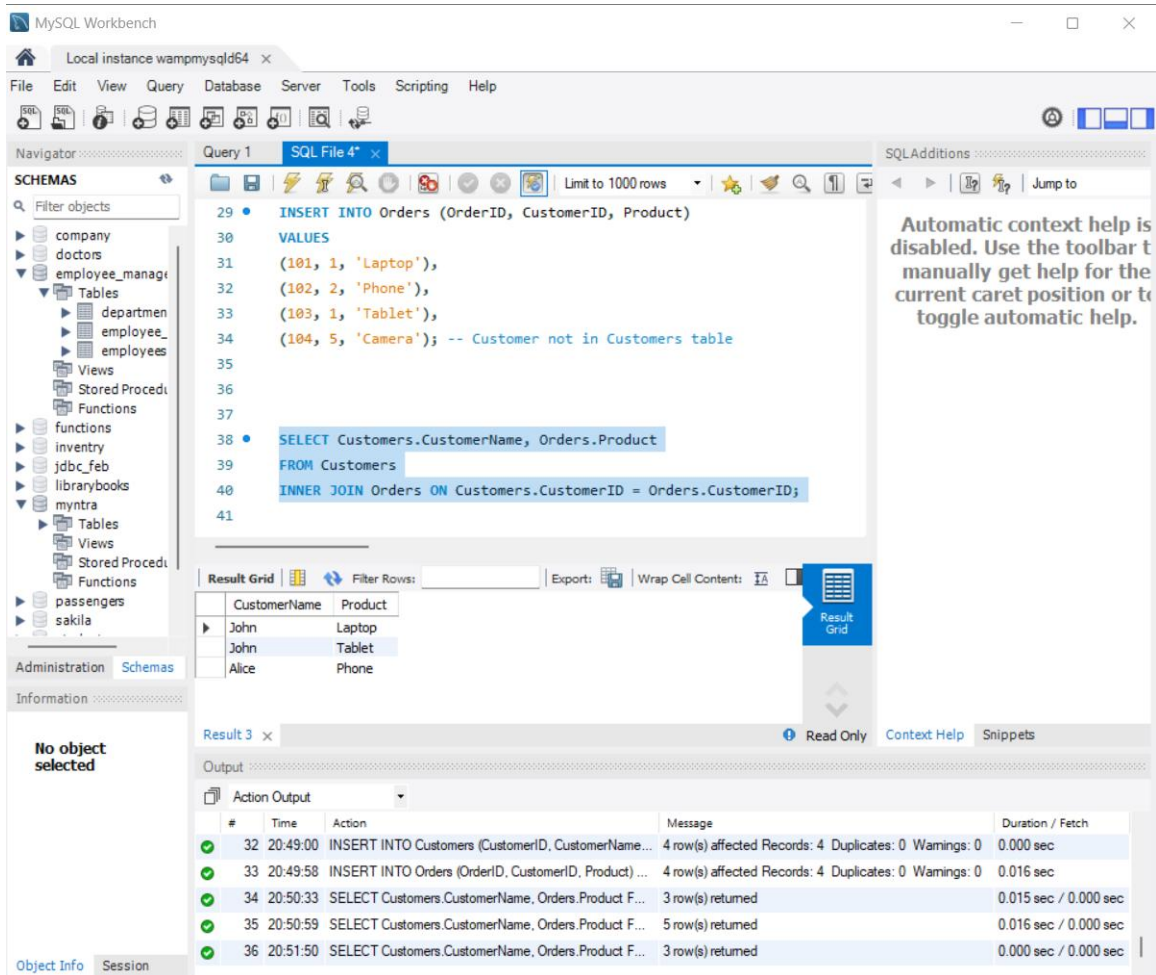
Returns only the rows that have matching values in both tables.

It is used when you want to see the common records between related tables.

Query:

```
SELECT Customers.CustomerName, Orders.Product
FROM Customers
INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

➡ This shows only customers who have placed orders.



4.2 LEFT JOIN

Returns all rows from the left table, even if there are no matches in the right table.
If no match is found, NULL values are returned for the right table's columns.

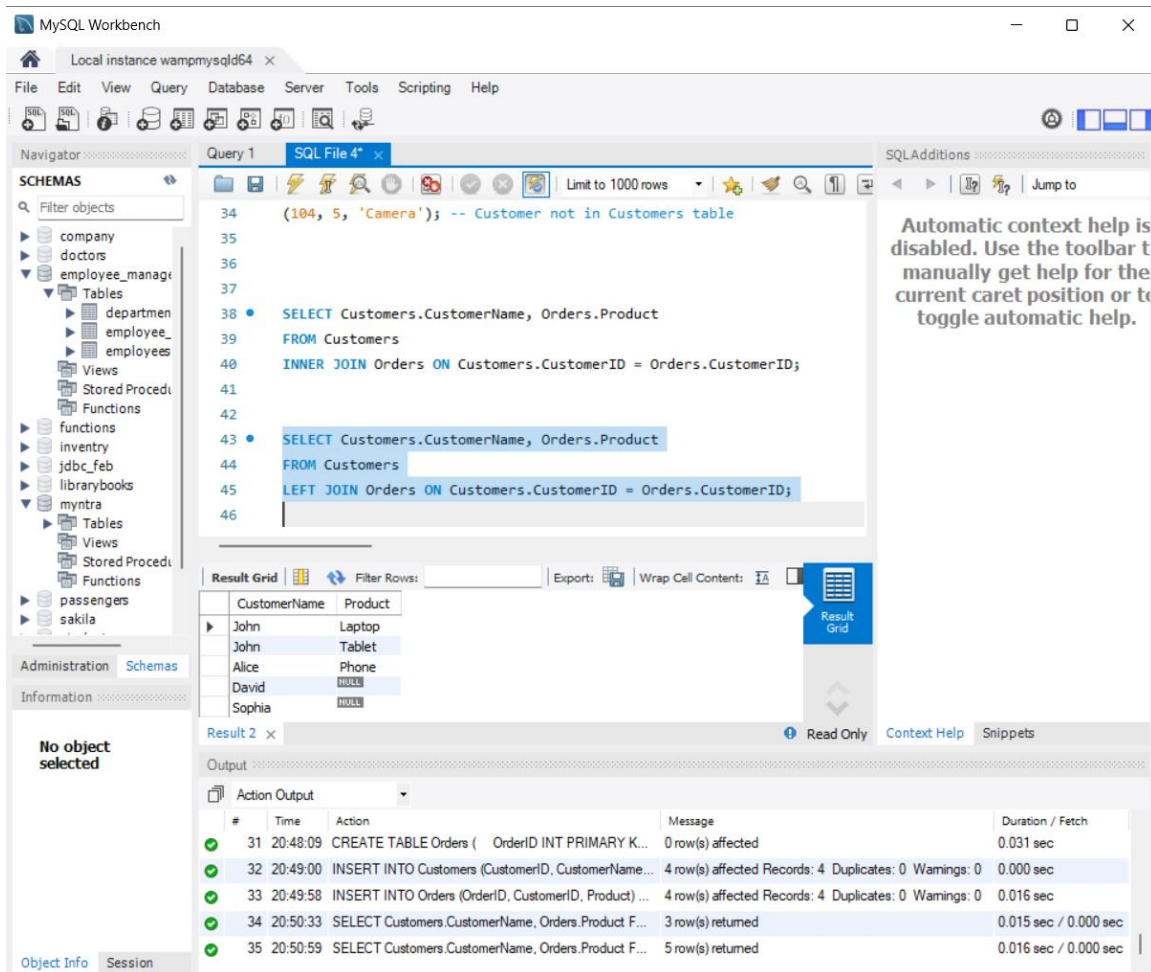
Query:

SELECT Customers.CustomerName, Orders.Product

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

➡ This shows all customers, including those without orders (NULL in Product).



4.3 RIGHT JOIN

Returns all rows from the right table, even if there are no matches in the left table.
If no match is found, NULL values are returned for the left table's columns.

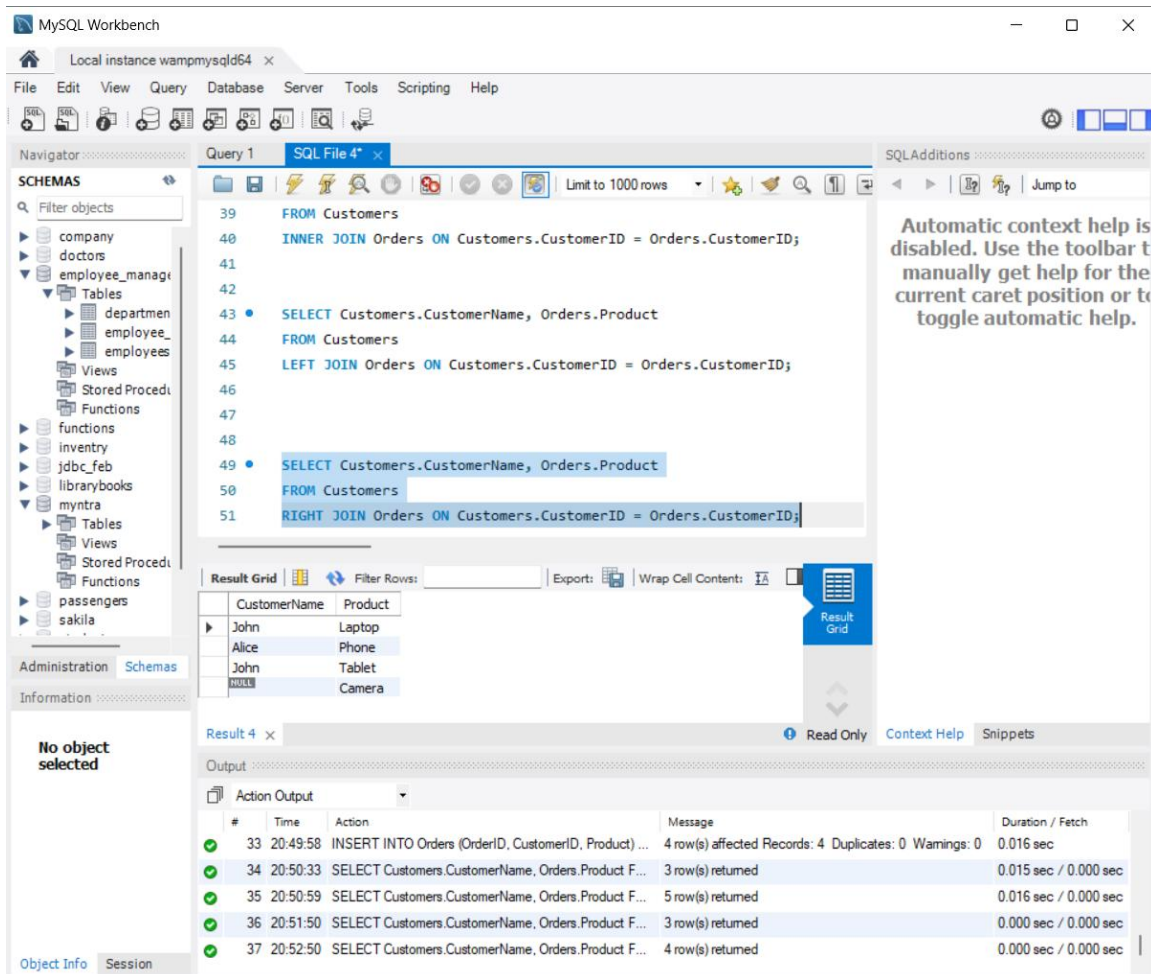
Query:

SELECT Customers.CustomerName, Orders.Product

FROM Customers

RIGHT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

➡ This shows all orders, including those without matching customers (NULL in CustomerName).



4.4 FULL OUTER JOIN (using UNION)

Returns all rows from both tables, with matches where possible.
Rows without a match in either table will show NULL values.

Query:

```

SELECT Customers.CustomerName, Orders.Product
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
UNION
SELECT Customers.CustomerName, Orders.Product
FROM Customers
RIGHT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

```

➡ This combines results of LEFT and RIGHT JOIN, showing all customers and all orders.

The screenshot displays the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a tree view of databases including 'company', 'doctors', 'employee_manage', 'mynta', and 'sakila'. The 'sakila' database is selected. The main query editor shows a SQL query (Query 1) that uses a UNION of two queries, each involving a LEFT JOIN and a RIGHT JOIN between 'Customers' and 'Orders' tables. The query is limited to 1000 rows. Below the query editor, the 'Result Grid' shows the output of the query, displaying columns 'CustomerName' and 'Product' with rows for John (Tablet), John (Laptop), Alice (Phone), David (NULL), Sophia (NULL), and Camera. To the right of the query editor, a 'SQLAdditions' pane displays a message: 'Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.' Below the result grid, the 'Output' pane shows the 'Action Output' log, which includes a table with columns '#', 'Time', 'Action', 'Message', and 'Duration / Fetch'. The log shows five successful queries (IDs 34-38) and their execution times.

SQL Query:

```

53
54
55
56 SELECT Customers.CustomerName, Orders.Product
57 FROM Customers
58 LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
59 UNION
60 SELECT Customers.CustomerName, Orders.Product
61 FROM Customers
62 RIGHT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
63
64

```

Result Grid:

CustomerName	Product
John	Tablet
John	Laptop
Alice	Phone
David	NULL
Sophia	NULL
Camera	

Action Output Log:

#	Time	Action	Message	Duration / Fetch
34	20:50:33	SELECT Customers.CustomerName, Orders.Product F...	3 row(s) returned	0.015 sec / 0.000 sec
35	20:50:59	SELECT Customers.CustomerName, Orders.Product F...	5 row(s) returned	0.016 sec / 0.000 sec
36	20:51:50	SELECT Customers.CustomerName, Orders.Product F...	3 row(s) returned	0.000 sec / 0.000 sec
37	20:52:50	SELECT Customers.CustomerName, Orders.Product F...	4 row(s) returned	0.000 sec / 0.000 sec
38	20:53:46	SELECT Customers.CustomerName, Orders.Product F...	6 row(s) returned	0.000 sec / 0.000 sec

5. Conclusion

Through this exercise, we understood how INNER, LEFT, RIGHT, and FULL OUTER JOIN work in MySQL. These joins are essential for merging and analyzing data across multiple related tables.