

✓ Query 1: Show all customers and their orders (LEFT JOIN)

- **Overview:** Display all customers' orders.
- **Data Used:** customers, orders
- **Analysis Technique:** LEFT JOIN to include customers even without orders
- **Tools Applied:** MySQL, SELECT, LEFT JOIN, ORDER BY

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
1 SELECT
2   c.customer_id,
3   c.name AS customer_name,
4   o.order_id,
5   o.order_date
6 FROM customers c
7 LEFT JOIN orders o ON c.customer_id = o.customer_id
8 ORDER BY c.customer_id;
```

The Results grid displays the output of the query, showing columns: customer_id, customer_name, order_id, and order_date. The data is as follows:

customer_id	customer_name	order_id	order_date
1	Amit Sharma	1	2024-01-05
1	Amit Sharma	4	2024-01-15
1	Amit Sharma	17	2024-02-01
2	Rina Das	2	2024-01-06
2	Rina Das	18	2024-02-02
3	Fazari Ali	3	2024-01-07
3	Fazari Ali	19	2024-02-03
4	Karan Mehra	5	2024-01-18
4	Karan Mehra	20	2024-02-04
5	Sneha Kapoor	6	2024-01-20
5	Sneha Kapoor	21	2024-02-05
6	Rohit Kumar	7	2024-01-21
6	Rohit Kumar	22	2024-02-06
7	Priya Sinha	8	2024-01-22
7	Priya Sinha	23	2024-02-07
8	Neha Singh	9	2024-01-23
8	Neha Singh	24	2024-02-08
9	Vikram Chauhan	10	2024-01-24
9	Vikram Chauhan	25	2024-02-09

✓ Query 2: Total orders by each customer

- **Overview:** Count total orders placed by each customer
- **Data Used:** customers, orders
- **Analysis Technique:** JOIN, COUNT aggregation, GROUP BY
- **Tools Applied:** MySQL, COUNT(), GROUP BY, ORDER BY

Query 1 SQL File 12 SQL File 13 SQL File 14

```

1 SELECT
2   c.name AS customer_name,
3   COUNT(o.order_id) AS total_orders
4 FROM customers c
5 JOIN orders o ON c.customer_id = o.customer_id
6 GROUP BY c.customer_id
7 ORDER BY total_orders DESC;
8

```

Result Grid | Filter Rows: | Exports: | Wrap Cell Content: |

customer_name	total_orders
Amit Sharma	3
Rina Das	2
Faizan Ali	2
Karan Mehra	2
Sneha Kapoor	2
Rohit Kumar	2
Priya Sinha	2
Neha Singh	2
Vikram Chaudhan	2
Tina Dsouza	1
Alok Tiwari	1
Shivam Rana	1
Divya Verma	1
Nikhil Joshi	1
Megha Jain	1

✓ Query 3: Customers with more than 2 orders

```

1 SELECT
2   c.name AS customer_name,
3   COUNT(o.order_id) AS total_orders
4 FROM customers c
5 JOIN orders o ON c.customer_id = o.customer_id
6 GROUP BY c.customer_id
7 HAVING total_orders > 2
8 ORDER BY total_orders DESC;
9

```

Result Grid | Filter Rows: | Exports: | Wrap Cell Content: |

customer_name	total_orders
Amit Sharma	3

Result Grid
Form Editor
Field Types
Query Stats

- **Overview:** Identify high-frequency customers with more than 2 orders
- **Data Used:** customers, orders
- **Analysis Technique:** GROUP BY with HAVING filter
- **Tools Applied:** MySQL, COUNT(), GROUP BY, HAVING

✓ Query 4: Latest order for each customer

Query 1 SQL File 12* SQL File 13* SQL File 14* SQL File 15* x

Limit to 1000 rows

```

1 • SELECT
2     c.name AS customer_name,
3     MAX(o.order_date) AS latest_order
4 FROM customers c
5 JOIN orders o ON c.customer_id = o.customer_id
6 GROUP BY c.customer_id;
7

```

Result Grid Filter Rows: Export: Wrap Cell Content:

customer_name	latest_order
Amit Sharma	2024-02-01
Rina Das	2024-02-02
Faizan Ali	2024-02-03
Karan Mehra	2024-02-04
Sneha Kapoor	2024-02-05
Rohit Kumar	2024-02-06
Priya Sinha	2024-02-07
Neha Singh	2024-02-08
Vikram Chauhan	2024-02-09
Tina Dsouza	2024-01-25
Alok Tiwari	2024-01-26
Shivam Rana	2024-01-27
Divya Verma	2024-01-28
Nikhil Joshi	2024-01-29
Megha Jain	2024-01-30

Result 1 x

- **Overview:** Find the most recent order date per customer
- **Data Used:** customers, orders
- **Analysis Technique:** Aggregation using MAX()
- **Tools Applied:** MySQL, MAX(), GROUP BY

✓ Query 5: Total revenue by customer

Limit to 1000 rows

```

4 FROM customers c
5 JOIN orders o ON c.customer_id = o.customer_id
6 JOIN order_items oi ON o.order_id = oi.order_id
7 JOIN products p ON oi.product_id = p.product_id
8 GROUP BY c.customer_id
9 ORDER BY total_spent DESC;
10

```

Result Grid Filter Rows: Export: Wrap Cell Content:

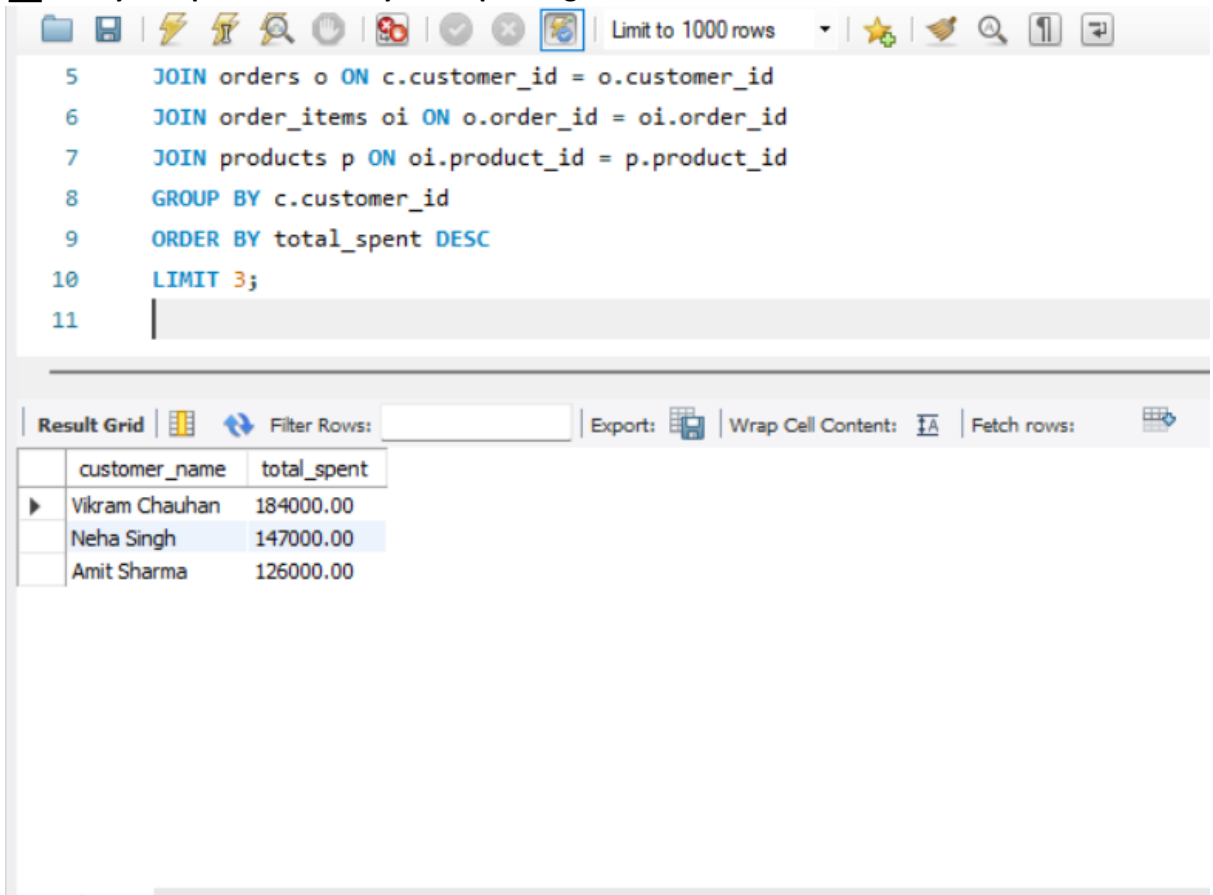
customer_name	total_spent
Vikram Chauhan	184000.00
Neha Singh	147000.00
Amit Sharma	126000.00
Sneha Kapoor	117500.00
Tina Dsouza	75000.00
Divya Verma	70000.00
Rina Das	57000.00
Karan Mehra	50000.00
Rohit Kumar	38000.00
Alok Tiwari	30500.00
Faizan Ali	28000.00
Nikhil Joshi	16500.00

Result 1 x

Output

- **Overview:** Calculate how much each customer has spent in total
- **Data Used:** customers, orders, order_items, products
- **Analysis Technique:** Revenue calculation using JOINS and SUM
- **Tools Applied:** MySQL, JOIN, SUM(), Arithmetic, GROUP BY

✓ Query 6: Top 3 customers by total spending

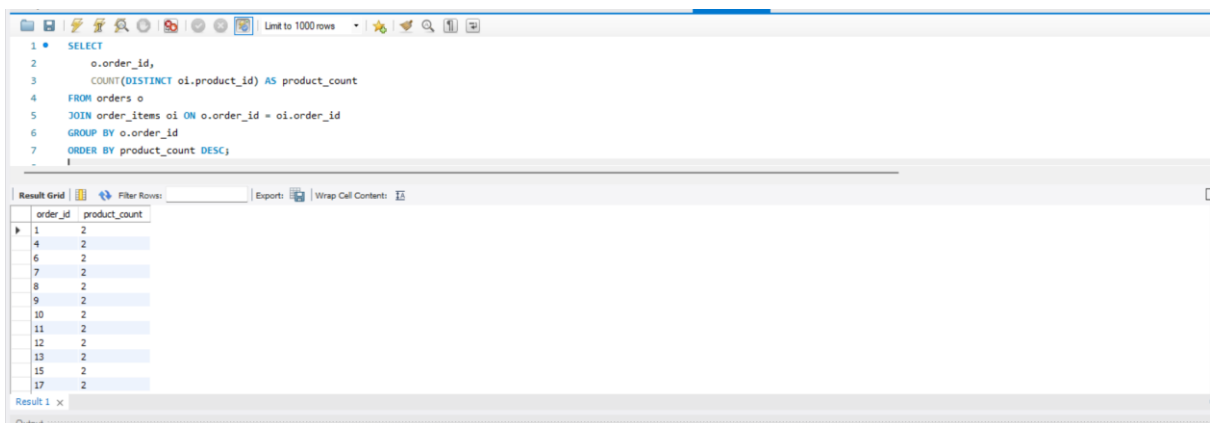


```
5 JOIN orders o ON c.customer_id = o.customer_id
6 JOIN order_items oi ON o.order_id = oi.order_id
7 JOIN products p ON oi.product_id = p.product_id
8 GROUP BY c.customer_id
9 ORDER BY total_spent DESC
10 LIMIT 3;
11
```

Result Grid

	customer_name	total_spent
▶	Vikram Chauhan	184000.00
	Neha Singh	147000.00
	Amit Sharma	126000.00

✓ Query 7: Number of products in each order



```
1 SELECT
2   o.order_id,
3   COUNT(DISTINCT oi.product_id) AS product_count
4 FROM orders o
5 JOIN order_items oi ON o.order_id = oi.order_id
6 GROUP BY o.order_id
7 ORDER BY product_count DESC;
```

Result Grid

order_id	product_count
1	2
4	2
6	2
7	2
8	2
9	2
10	2
11	2
12	2
13	2
15	2
17	2

- **Overview:** Count how many distinct products are in each order
- **Data Used:** orders, order_items
- **Analysis Technique:** COUNT DISTINCT with GROUP BY
- **Tools Applied:** MySQL, COUNT(DISTINCT), GROUP BY