DATABASE MANAGEMENT SYSTEM

Name: Sayali Bhaurao Gangurde

PRN: 2124UCEF1085

FY, CSE

Database Schema Design

Choose a unique and personal topic for your database (e.g., tracking daily expenses, managing a local sports club, or maintaining a personal book collection etc). Create at least two related tables, ensuring that one table has a Primary Key (PK) and another has a Foreign Key (FK).

Data Insertion

Manually insert at least 10 records per table. (At least one record should contain your Name). Ensure that the data is self-created and relevant to your chosen topic.

SQL Query Execution

Write and execute the following queries:

a) A JOIN query that retrieves data from both tables.

b) An aggregate query using functions such as COUNT, SUM, AVG, or GROUP BY.

Handwritten Explanation

Write a single handwritten paragraph explanation of the queries, describing their logic and expected results.

Database Schema Design for Order Management System

1] Customer Table:

```
CREATE TABLE Customers (
   customer_id INT PRIMARY KEY,
   customer_name VARCHAR(30) NOT NULL,
   customer_address VARCHAR(30)
);
```

2] Order Table:

```
CREATE TABLE Orders (
   order_id INT PRIMARY KEY,
   order_date DATE NOT NULL,
   customer_id INT,
   FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);
```

CREATING TWO TABLES, Customers and Order

```
CREATE TABLE Customers (
   customer_id INT PRIMARY KEY,
   customer_name VARCHAR(30) NOT NULL,
   customer_address VARCHAR(30)
);


CREATE TABLE Orders (
   order_id INT PRIMARY KEY,
   order_date DATE NOT NULL,
   customer_id INT,
   FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);
```

INSERTING DATA INTO Customer Table

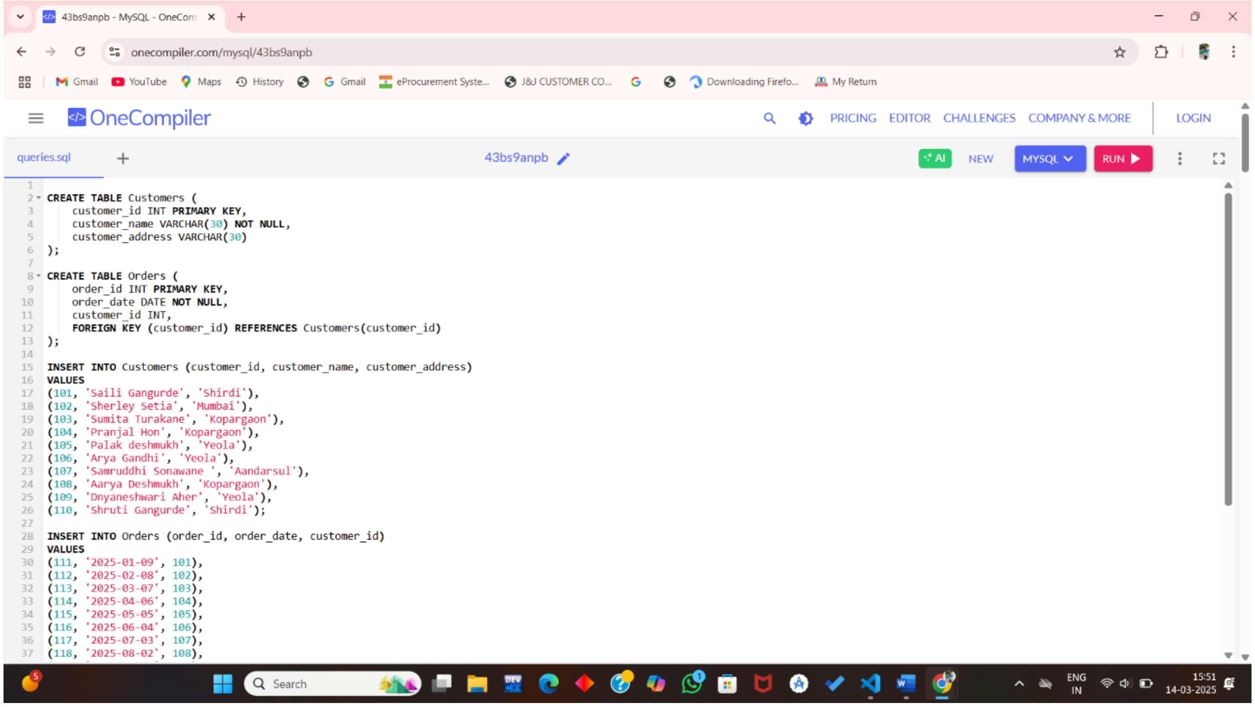
```
INSERT INTO Customers (customer_id, customer_name, customer_address)
VALUES
(101, 'Saili Gangurde', 'Shirdi'),
(102, 'Sherley Setia', 'Mumbai'),
(103, 'Sumita Turakane', 'Kopargaon'),
(104, 'Pranjal Hon', 'Kopargaon'),
(105, 'Palak deshmukh', 'Yeola'),
(106, 'Arya Gandhi', 'Yeola'),
(107, 'Samruddhi Sonawane ', 'Aandarsul'),
(108, 'Aarya Deshmukh', 'Kopargaon'),
(109, 'Dnyaneshwari Aher', 'Yeola'),

(110, 'Shruti Gangurde', 'Shirdi');
```

INSERTING DATA INTO Order TABLE

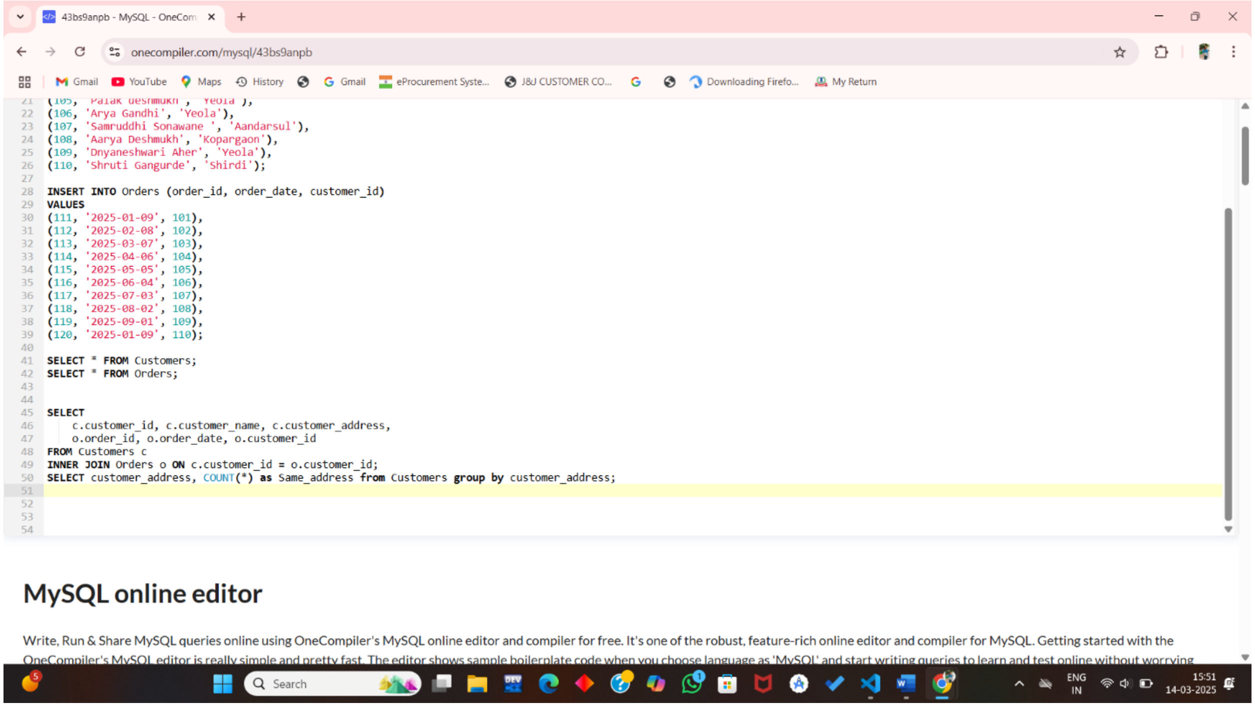
```
INSERT INTO Orders (order_id, order_date, customer_id)

VALUES

(111, '2025-01-09', 101),

(112, '2025-02-08', 102),

(113, '2025-03-07', 103),

(114, '2025-04-06', 104),

(115, '2025-05-05', 105),

(116, '2025-06-04', 106),

(117, '2025-07-03', 107),

(118, '2025-08-02', 108),

(119, '2025-09-01', 109),

(120, '2025-01-09', 110);
```

A JOIN QUERY THAT RETRIEVES DATA FROM BOTH TABLES.

```
  c.customer_id, c.customer_name, c.customer_address,

   o.order_id, o.order_date, o.customer_id

FROM Customers c

INNER JOIN Orders o ON c.customer_id = o.customer_id;

SELECT customer_address, COUNT(*) as Same_address from Customers group by
customer_address;
```



```

CREATE TABLE Customers (
    customer_id INT PRIMARY KEY,
    customer_name VARCHAR(30) NOT NULL,
    customer_address VARCHAR(30)
);

CREATE TABLE Orders (
    order_id INT PRIMARY KEY,
    order_date DATE NOT NULL,
    customer_id INT,
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);

INSERT INTO Customers (customer_id, customer_name, customer_address)
VALUES
(101, 'Saili Gangurde', 'Shirdi'),
(102, 'Sherley Setia', 'Mumbai'),
(103, 'Sumita Turakane', 'Kopargaon'),
(104, 'Pranjal Hon', 'Kopargaon'),
(105, 'Palak deshmukh', 'Yeola'),
(106, 'Arya Gandhi', 'Yeola'),
(107, 'Samruddhi Sonawane ', 'Aandarsul'),
(108, 'Aarya Deshmukh', 'Kopargaon'),
(109, 'Dnyaneshwari Aher', 'Yeola'),
(110, 'Shruti Gangurde', 'Shirdi');

INSERT INTO Orders (order_id, order_date, customer_id)
VALUES
(111, '2025-01-09', 101),
(112, '2025-02-08', 102),
(113, '2025-03-07', 103),
(114, '2025-04-06', 104),
(115, '2025-05-05', 105),
(116, '2025-06-04', 106),
(117, '2025-07-03', 107),
(118, '2025-08-02', 108),
```

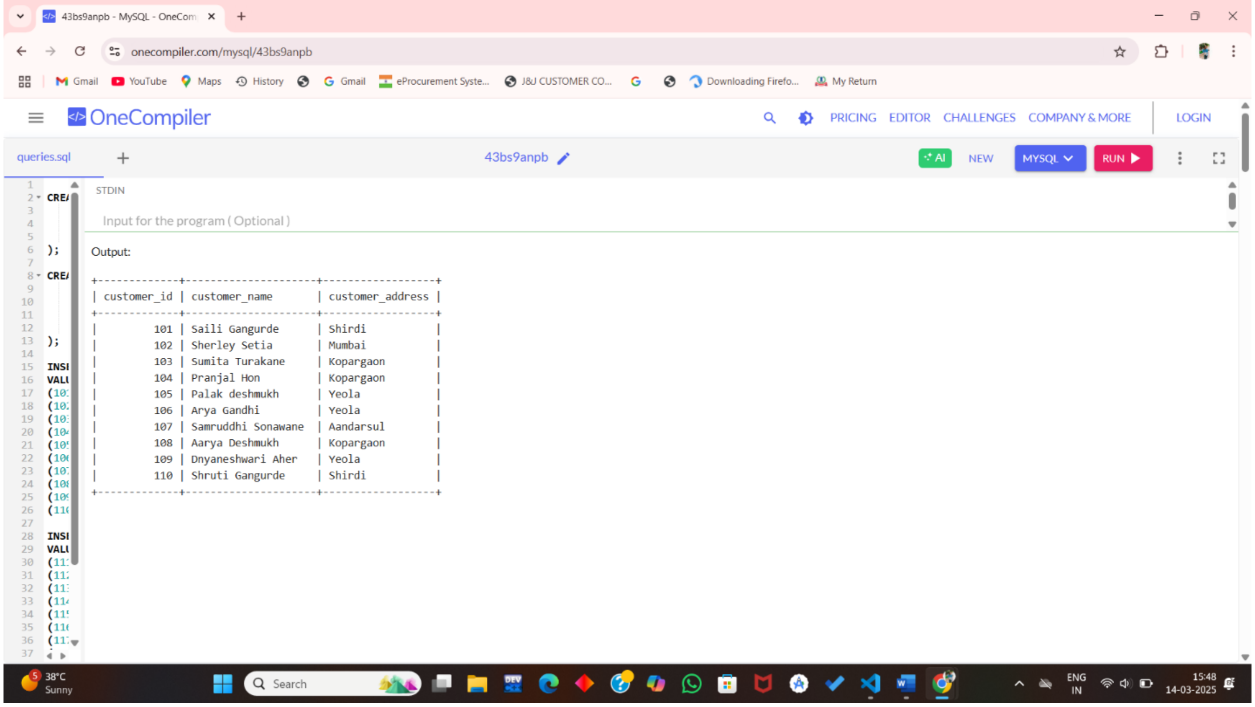


```
(105, 'Palak deshmukh', 'Yeola'),
(106, 'Arya Gandhi', 'Yeola'),
(107, 'Samruddhi Sonawane ', 'Aandarsul'),
(108, 'Aarya Deshmukh', 'Kopargaon'),
(109, 'Dnyaneshwari Aher', 'Yeola'),
(110, 'Shruti Gangurde', 'Shirdi');

INSERT INTO Orders (order_id, order_date, customer_id)
VALUES
(111, '2025-01-09', 101),
(112, '2025-02-08', 102),
(113, '2025-03-07', 103),
(114, '2025-04-06', 104),
(115, '2025-05-05', 105),
(116, '2025-06-04', 106),
(117, '2025-07-03', 107),
(118, '2025-08-02', 108),
(119, '2025-09-01', 109),
(120, '2025-01-09', 110);

SELECT * FROM Customers;
SELECT * FROM Orders;


SELECT
    c.customer_id, c.customer_name, c.customer_address,
    o.order_id, o.order_date, o.customer_id
FROM Customers c
INNER JOIN Orders o ON c.customer_id = o.customer_id;
SELECT customer_address, COUNT(*) as Same_address from Customers group by customer_address;
```

MySQL online editor

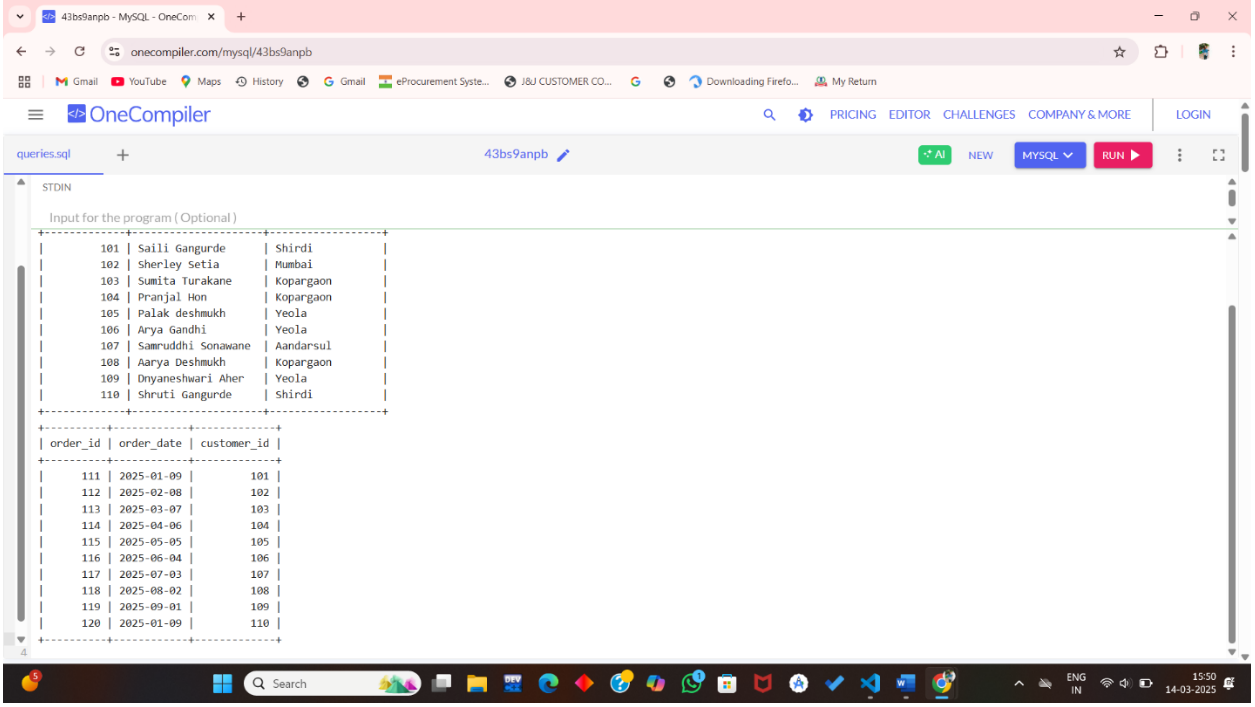
Write, Run & Share MySQL queries online using OneCompiler's MySQL online editor and compiler for free. It's one of the robust, feature-rich online editor and compiler for MySQL. Getting started with the OneCompiler's MySQL editor is really simple and pretty fast. The editor shows sample boilerplate code when you choose language as 'MySQL' and start writing queries to learn and test online without worrying

OUTPUT

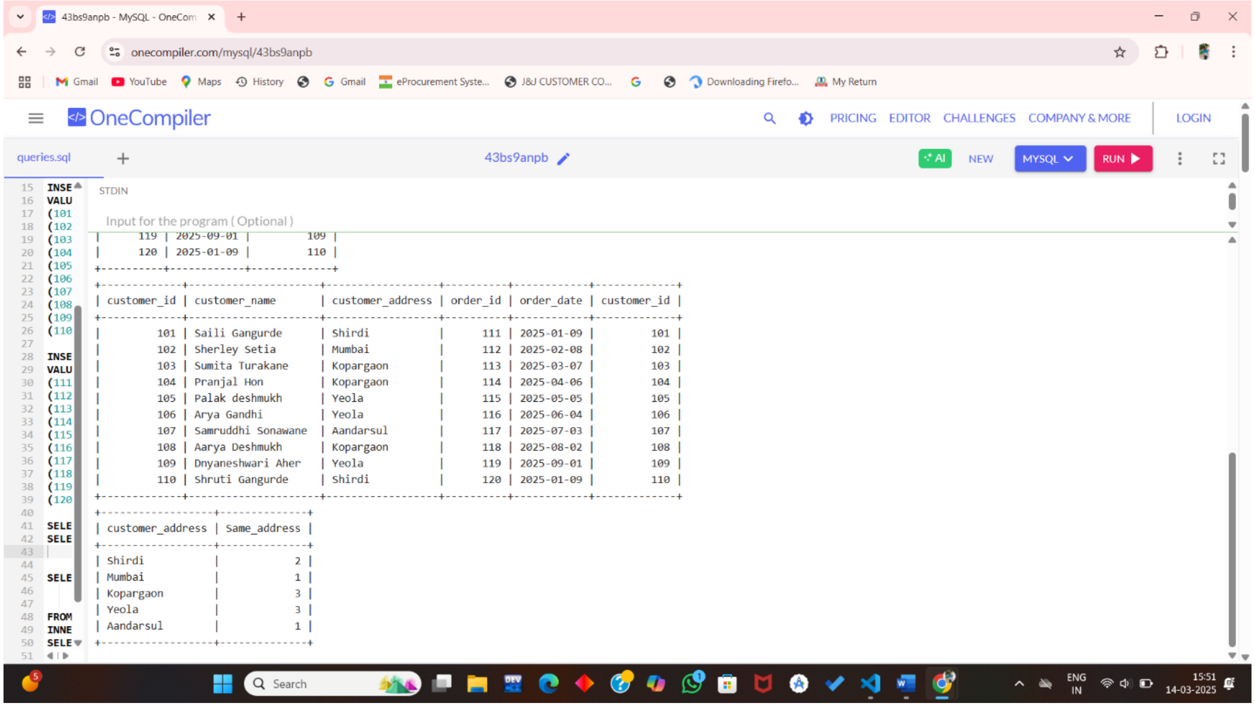


Output:

```
+-------------+--------------------+------------------+
| customer_id | customer_name      | customer_address |
+-------------+--------------------+------------------+
|         101 | Saili Gangurde     | Shirdi           |
|         102 | Sherley Setia      | Mumbai           |
|         103 | Sumita Turakane    | Kopargaon        |
|         104 | Pranjal Hon        | Kopargaon        |
|         105 | Palak deshmukh     | Yeola            |
|         106 | Arya Gandhi        | Yeola            |
|         107 | Samruddhi Sonawane | Aandarsul        |
|         108 | Aarya Deshmukh     | Kopargaon        |
|         109 | Dnyaneshwari Aher  | Yeola            |
|         110 | Shruti Gangurde    | Shirdi           |
+-------------+--------------------+------------------+
```



```
+-------------+--------------------+------------------+
|         101 | Saili Gangurde     | Shirdi           |
|         102 | Sherley Setia      | Mumbai           |
|         103 | Sumita Turakane    | Kopargaon        |
|         104 | Pranjal Hon        | Kopargaon        |
|         105 | Palak deshmukh     | Yeola            |
|         106 | Arya Gandhi        | Yeola            |
|         107 | Samruddhi Sonawane | Aandarsul        |
|         108 | Aarya Deshmukh     | Kopargaon        |
|         109 | Dnyaneshwari Aher  | Yeola            |
|         110 | Shruti Gangurde    | Shirdi           |
+-------------+--------------------+------------------+
+----------+------------+-------------+
| order_id | order_date | customer_id |
+----------+------------+-------------+
|      111 | 2025-01-09 |         101 |
|      112 | 2025-02-08 |         102 |
|      113 | 2025-03-07 |         103 |
|      114 | 2025-04-06 |         104 |
|      115 | 2025-05-05 |         105 |
|      116 | 2025-06-04 |         106 |
|      117 | 2025-07-03 |         107 |
|      118 | 2025-08-02 |         108 |
|      119 | 2025-09-01 |         109 |
|      120 | 2025-01-09 |         110 |
+----------+------------+-------------+
```



```
INSE
VALU
(101
(102
(103
(104
(105
(106
(107
(108
(109
(110

INSE
VALU
(111
(112
(113
(114
(115
(116
(117
(118
(119
(120

SELE
SELE


SELE


FROM
INNE
SELE
```

```
|     119 | 2025-09-01 |         109 |
|     120 | 2025-01-09 |         110 |
+----------+------------+-------------+

+-------------+--------------------+------------------+----------+------------+-------------+
| customer_id | customer_name      | customer_address | order_id | order_date | customer_id |
+-------------+--------------------+------------------+----------+------------+-------------+
|         101 | Saili Gangurde     | Shirdi           |      111 | 2025-01-09 |         101 |
|         102 | Sherley Setia      | Mumbai           |      112 | 2025-02-08 |         102 |
|         103 | Sumita Turakane    | Kopargaon        |      113 | 2025-03-07 |         103 |
|         104 | Pranjal Hon        | Kopargaon        |      114 | 2025-04-06 |         104 |
|         105 | Palak deshmukh     | Yeola            |      115 | 2025-05-05 |         105 |
|         106 | Arya Gandhi        | Yeola            |      116 | 2025-06-04 |         106 |
|         107 | Samruddhi Sonawane | Aandarsul        |      117 | 2025-07-03 |         107 |
|         108 | Aarya Deshmukh     | Kopargaon        |      118 | 2025-08-02 |         108 |
|         109 | Dnyaneshwari Aher  | Yeola            |      119 | 2025-09-01 |         109 |
|         110 | Shruti Gangurde    | Shirdi           |      120 | 2025-01-09 |         110 |
+-------------+--------------------+------------------+----------+------------+-------------+

+------------------+--------------+
| customer_address | Same_address |
+------------------+--------------+
| Shirdi           |            2 |
| Mumbai           |            1 |
| Kopargaon        |            3 |
| Yeola            |            3 |
| Aandarsul        |            1 |
+------------------+--------------+
```

Explanation of the queries, describing their logic and expected results

The SQL script first creates two tables: Customers and Orders. The Customers table has three columns: customer_id as the primary key, customer_name (a required field), and customer_address. The Orders table contains order_id as the primary key, order_date (a required field), and customer_id, which acts as a foreign key referencing Customers(customer_id), establishing a relationship between customers and their respective orders.

Next, data is inserted into both tables. The Customers table is populated with 10 customers, each having a unique customer_id, name, and address. The Orders table is also populated with 10 orders, each linked to a corresponding customer_id, ensuring every listed order belongs to an existing customer.

An INNER JOIN query is then executed to retrieve records of customers who have placed orders. This query joins both tables based on customer_id, selecting relevant details like customer_name, customer_address, order_id, and order_date. Since it's an INNER JOIN, only customers with associated orders are displayed.

Finally, a GROUP BY query is used to count the number of customers from each address. This groups customers by customer_address and calculates how many belong to the same location, providing insights into how customers are distributed across different areas. The overall queries efficiently manage customer and order data while ensuring meaningful retrieval of information.