

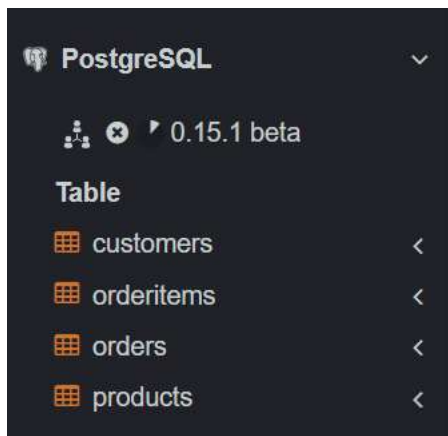
Saifertek SQL Assignment

2100031993

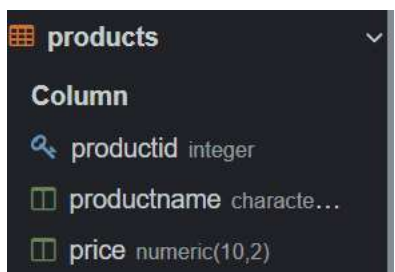
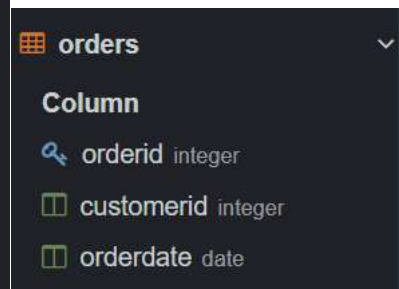
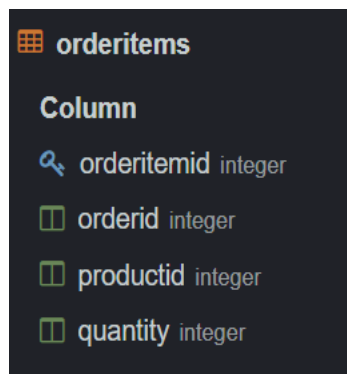
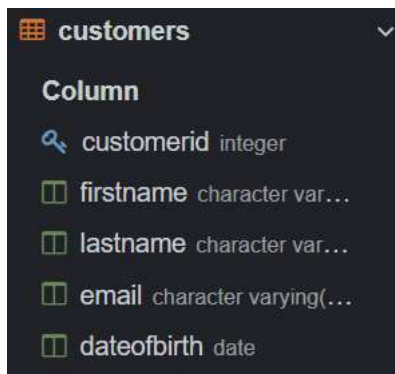
K.Harika Chinmay

Points to be noted:

- 1) To do the following task I have used online PostgreSQL compiler
- 2) I created all 4 tables and inserted the data into them



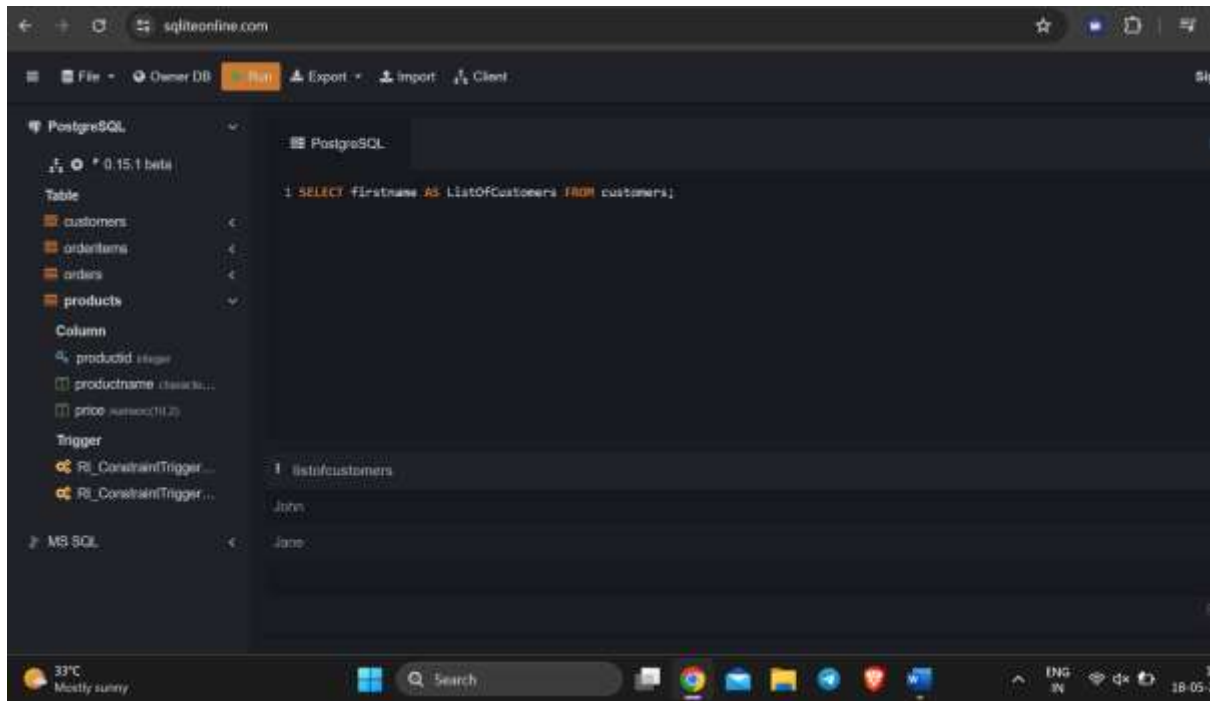
3) Following image displays the schema of the tables also:



Task

Question: . List all customers.

Query and output:



The screenshot shows the sqlliteonline.com interface. On the left, a sidebar lists database objects for a PostgreSQL database: 'customers', 'orderitems', 'orders', and 'products'. The 'customers' table is selected, showing its columns: 'productid' (integer), 'productname' (character), and 'price' (numeric(10,2)).

The main query editor contains the following SQL query:

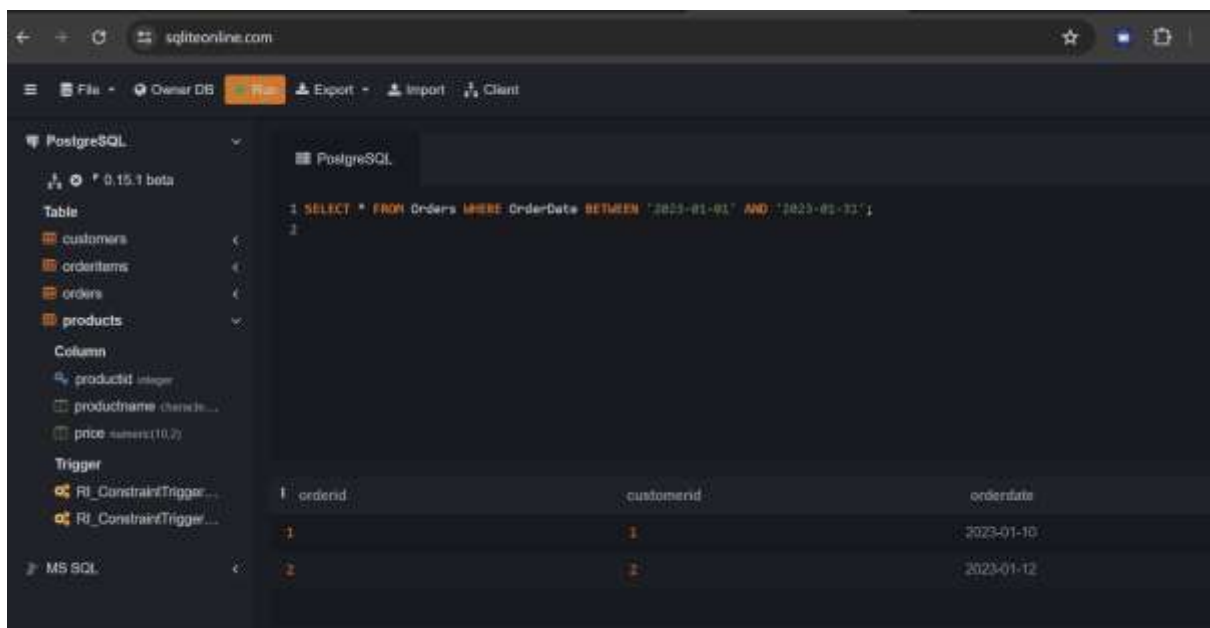
```
1 SELECT firstname AS ListOfCustomers FROM customers;
```

The output of the query is displayed in a table below the editor:

1	ListOfCustomers
1	John
2	Jane

2)question: Find all orders placed in January 2023.

Output and query:



The screenshot shows the sqlliteonline.com interface. On the left, a sidebar lists database objects for a PostgreSQL database: 'customers', 'orderitems', 'orders', and 'products'. The 'orders' table is selected, showing its columns: 'productid' (integer), 'productname' (character), and 'price' (numeric(10,2)).

The main query editor contains the following SQL query:

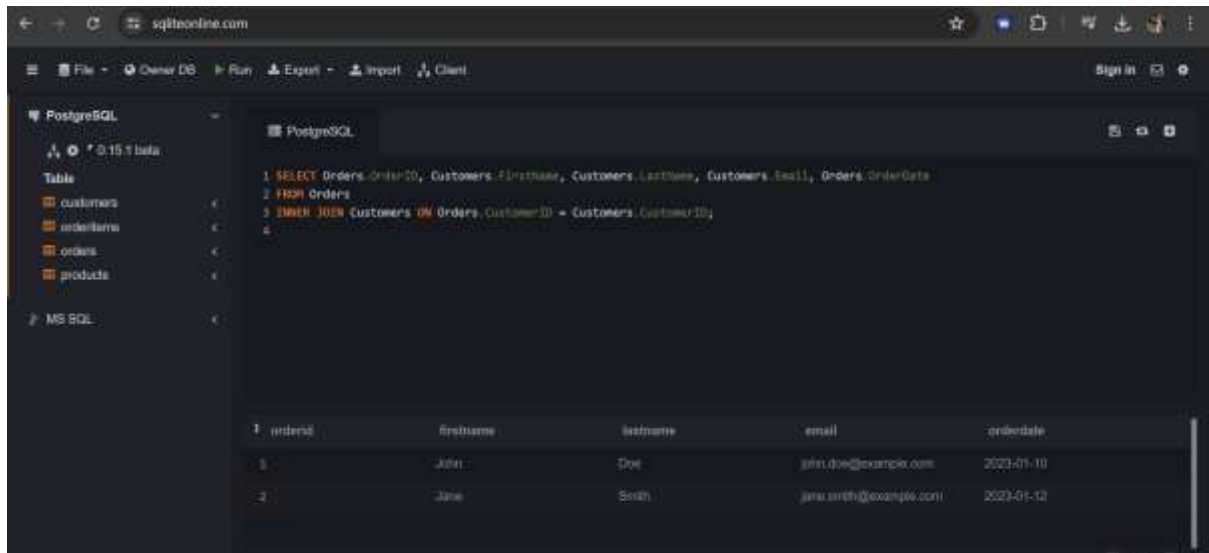
```
1 SELECT * FROM Orders WHERE OrderDate BETWEEN '2023-01-01' AND '2023-01-31';
```

The output of the query is displayed in a table below the editor:

1	orderid	customerid	orderdate
1	1	1	2023-01-10
2	2	2	2023-01-12

3)Question: . Get the details of each order, including the customer name and email.

Output and query:



The screenshot shows the sqlliteonline.com interface. On the left, a sidebar lists database tables: customers, orderitems, orders, and products. The main area displays a PostgreSQL query:

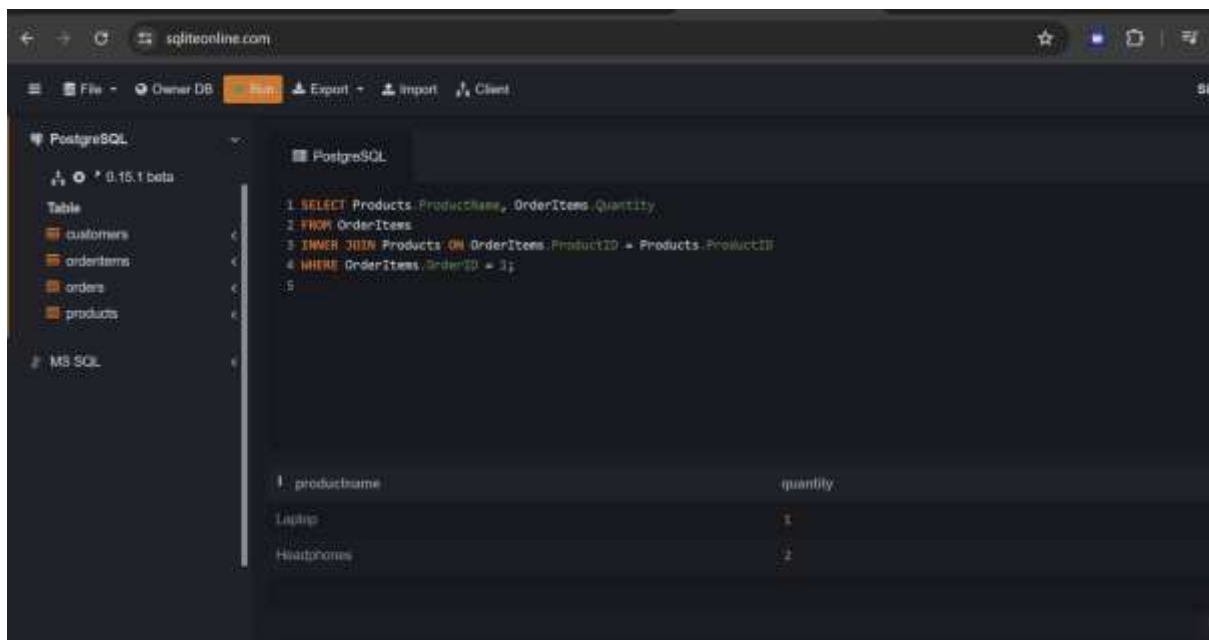
```
1 SELECT Orders.OrderID, Customers.Firstname, Customers.Lastname, Customers.Email, Orders.OrderDate
2 FROM Orders
3 INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
4
```

Below the query, the results are shown in a table with 5 columns: orderid, firstname, lastname, email, and orderdate. The results are as follows:

orderid	firstname	lastname	email	orderdate
1	John	Doe	john.doe@example.com	2023-01-10
2	Jane	Smith	jane.smith@example.com	2023-01-12

4) question: List the products purchased in a specific order (e.g., OrderID = 1).

Query and output:



The screenshot shows the sqlliteonline.com interface. On the left, a sidebar lists database tables: customers, orderitems, orders, and products. The main area displays a PostgreSQL query:

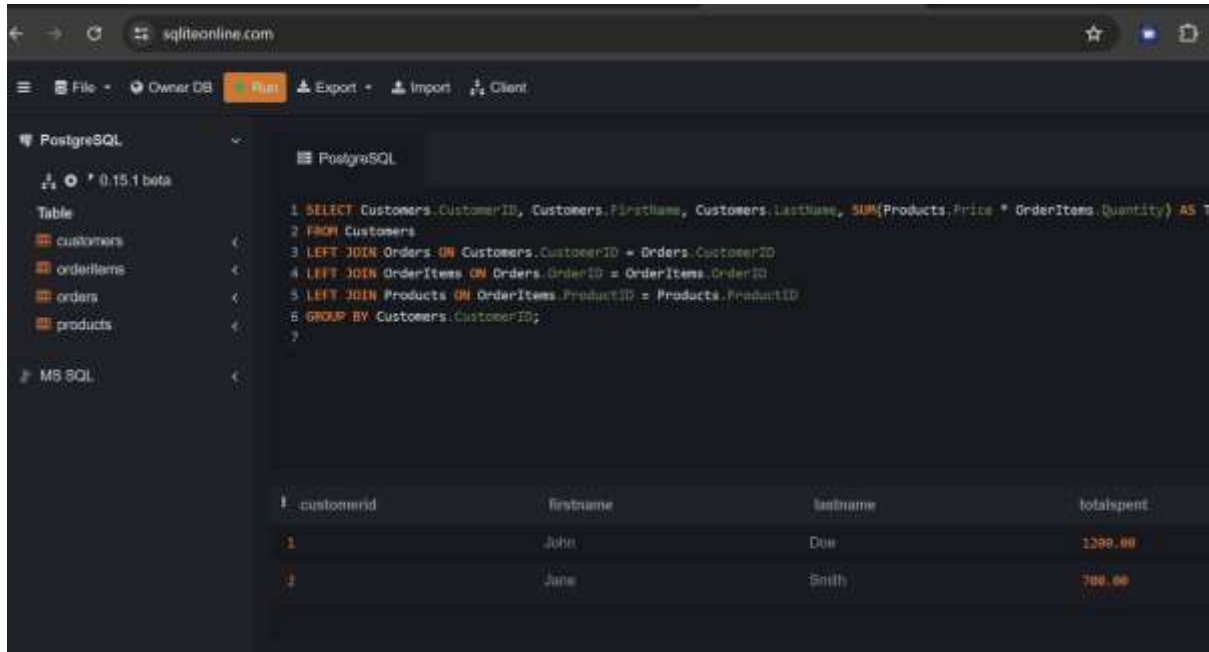
```
1 SELECT Products.ProductName, OrderItems.Quantity
2 FROM OrderItems
3 INNER JOIN Products ON OrderItems.ProductID = Products.ProductID
4 WHERE OrderItems.OrderID = 1;
5
```

Below the query, the results are shown in a table with 2 columns: productname and quantity. The results are as follows:

productname	quantity
Laptop	1
Headphones	2

5) . Calculate the total amount spent by each customer.

Query and output:



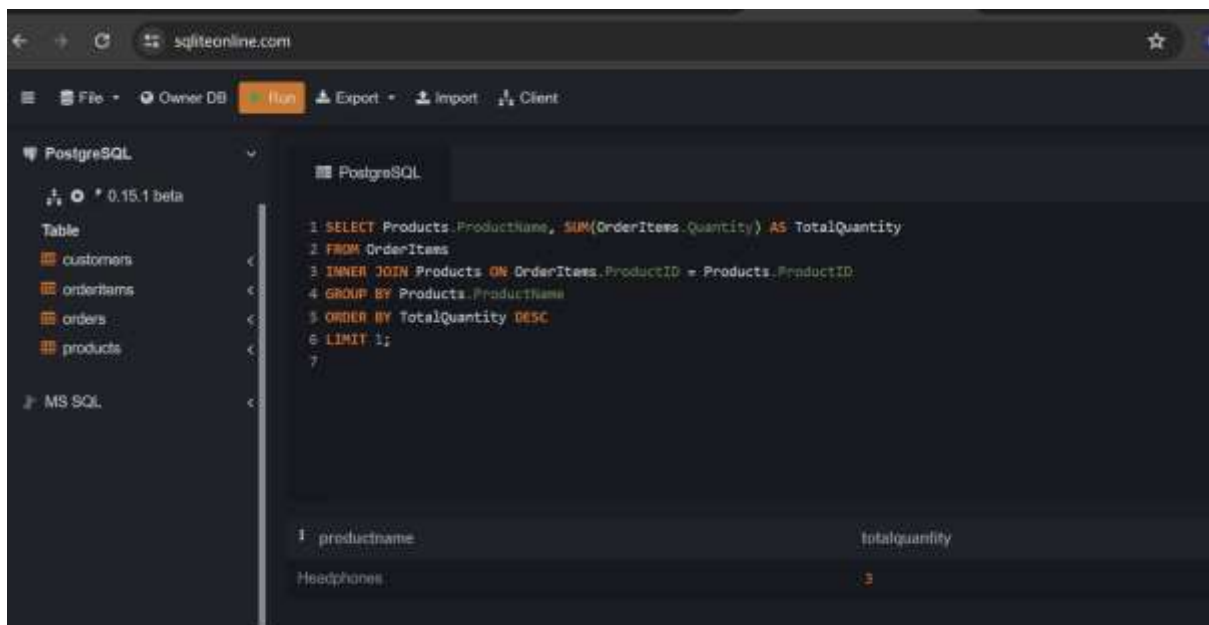
The screenshot shows the SQLiteonline.com web interface. On the left, a sidebar lists the database tables: customers, orderItems, orders, and products. The main area displays a PostgreSQL query that calculates the total amount spent by each customer. The query uses LEFT JOINs to combine data from the Customers, Orders, OrderItems, and Products tables, then groups the results by CustomerID. The output table shows two customers: John Doe with a total spent of 1399.00, and Jane Smith with a total spent of 700.00.

```
1 SELECT Customers.CustomerID, Customers.Firstname, Customers.Lastname, SUM(Products.Price * OrderItems.Quantity) AS TotalSpent
2 FROM Customers
3 LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
4 LEFT JOIN OrderItems ON Orders.OrderID = OrderItems.OrderID
5 LEFT JOIN Products ON OrderItems.ProductID = Products.ProductID
6 GROUP BY Customers.CustomerID;
```

customerid	firstname	lastname	totalspent
1	John	Doe	1399.00
2	Jane	Smith	700.00

6) Find the most popular product (the one that has been ordered the most).

query and output:



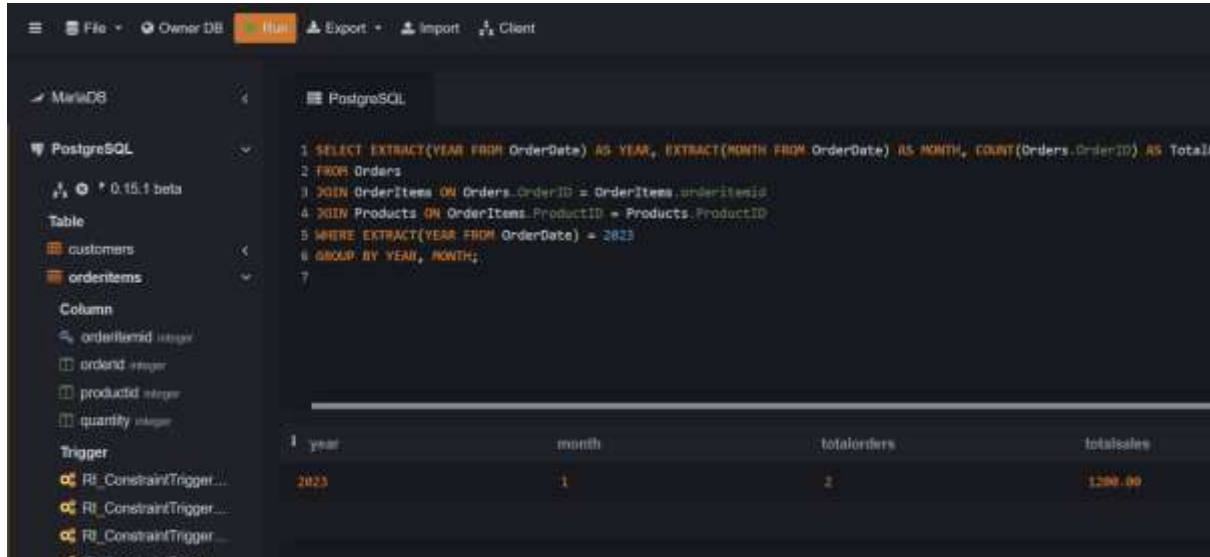
The screenshot shows the SQLiteonline.com web interface. On the left, a sidebar lists the database tables: customers, orderItems, orders, and products. The main area displays a PostgreSQL query that finds the most popular product by summing the quantities of each product ordered and then ordering the results in descending order of total quantity, limiting the output to one result. The output table shows that 'Headphones' is the most popular product with a total quantity of 3.

```
1 SELECT Products.ProductName, SUM(OrderItems.Quantity) AS TotalQuantity
2 FROM OrderItems
3 INNER JOIN Products ON OrderItems.ProductID = Products.ProductID
4 GROUP BY Products.ProductName
5 ORDER BY TotalQuantity DESC
6 LIMIT 1;
```

productname	totalquantity
Headphones	3

7) Get the total number of orders and the total sales amount for each month in 2023

query and output:



The screenshot shows a database client interface with a PostgreSQL query editor. The query is as follows:

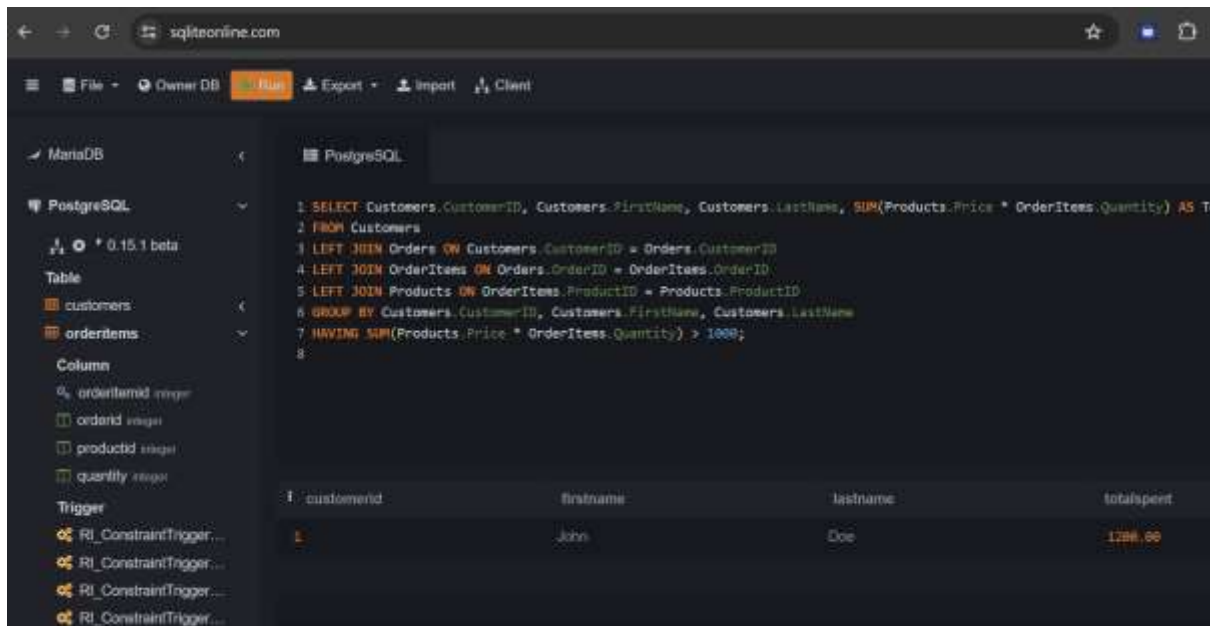
```
1 SELECT EXTRACT(YEAR FROM OrderDate) AS YEAR, EXTRACT(MONTH FROM OrderDate) AS MONTH, COUNT(Orders.OrderID) AS TotalOrders, SUM(Products.Price * OrderItems.Quantity) AS TotalSales
2 FROM Orders
3 JOIN OrderItems ON Orders.OrderID = OrderItems.orderitemid
4 JOIN Products ON OrderItems.ProductID = Products.ProductID
5 WHERE EXTRACT(YEAR FROM OrderDate) = 2023
6 GROUP BY YEAR, MONTH;
```

The output table has the following columns: year, month, totalorders, totalsales. The output shows 12 rows of data for the year 2023, with the first row being 2023, 1, 2, 1200.00.

year	month	totalorders	totalsales
2023	1	2	1200.00
2023	2	3	1800.00
2023	3	4	2400.00
2023	4	5	3000.00
2023	5	6	3600.00
2023	6	7	4200.00
2023	7	8	4800.00
2023	8	9	5400.00
2023	9	10	6000.00
2023	10	11	6600.00
2023	11	12	7200.00
2023	12	13	7800.00

8) Find customers who have spent more than \$1000.

Query and output:



The screenshot shows a database client interface with a PostgreSQL query editor. The query is as follows:

```
1 SELECT Customers.CustomerID, Customers.FirstName, Customers.LastName, SUM(Products.Price * OrderItems.Quantity) AS TotalSpent
2 FROM Customers
3 LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
4 LEFT JOIN OrderItems ON Orders.OrderID = OrderItems.OrderID
5 LEFT JOIN Products ON OrderItems.ProductID = Products.ProductID
6 GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName
7 HAVING SUM(Products.Price * OrderItems.Quantity) > 1000;
```

The output table has the following columns: customerid, firstname, lastname, totalspent. The output shows 1 row of data for the customer with ID 1, first name John, and last name Doe, with a total spent of 1200.00.

customerid	firstname	lastname	totalspent
1	John	Doe	1200.00

Python code and output for same task