SQL Data Analysis

December 2023

Dataset insight

For this project I used w3schools (https://www.w3schools.com/) dataset.

At first, I prepared a schema with a little help of https://app.diagrams.net/:

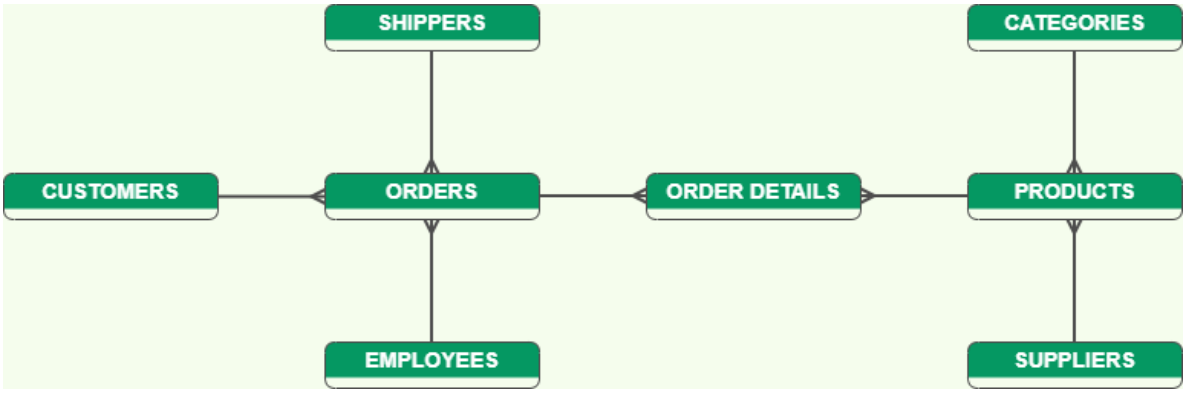


Figure 1: Database schema.

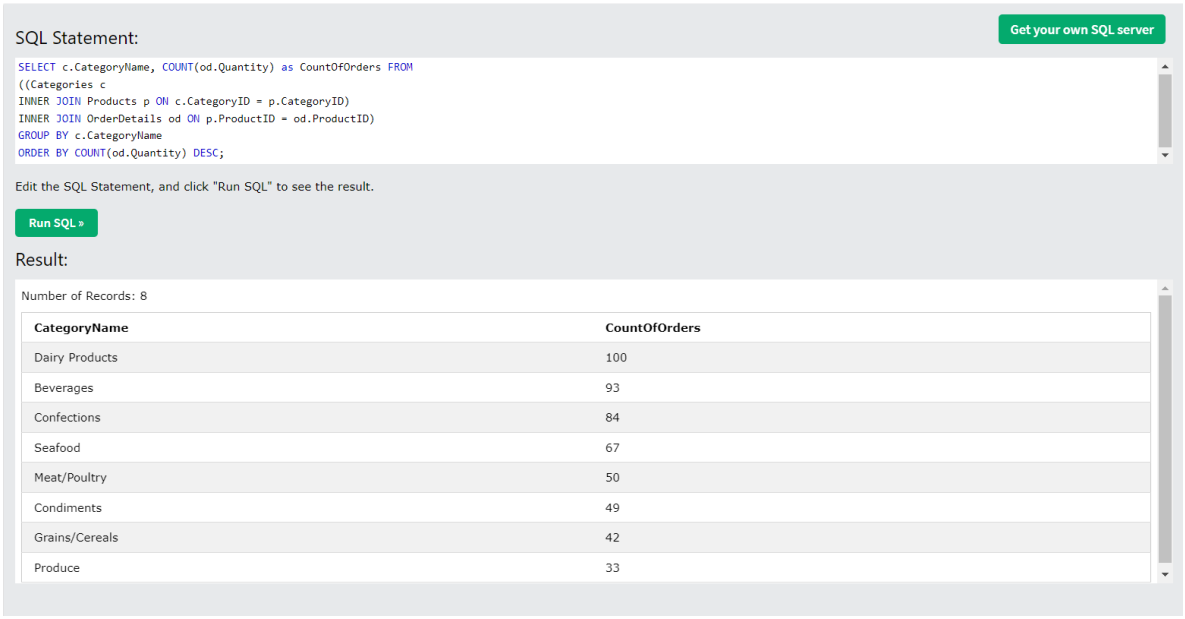
There are 8 tables in the database, in which we can find data describing orders of products and details like customers who did the orders, employees who supervised the orders, products' suppliers, orders' shippers and so on. Types of relationships between tables (one-to-many) are presented in the picture.

The main aim of this project was to conduct a data analysis using SQL queries. During the time I spent with the database, I wrote many queries, but below I will present only a few most, in my opinion, complex and insightful. In the screenshots shown below, you can find queries results as well.

Please enjoy!

O.

Let's start with presenting the products:



SQL Statement:

Get your own SQL server

```
SELECT c.CategoryName, COUNT(od.Quantity) as CountOfOrders FROM
((Categories c
INNER JOIN Products p ON c.CategoryID = p.CategoryID)
INNER JOIN OrderDetails od ON p.ProductID = od.ProductID)
GROUP BY c.CategoryName
ORDER BY COUNT(od.Quantity) DESC;
```

Edit the SQL Statement, and click "Run SQL" to see the result.

Run SQL »

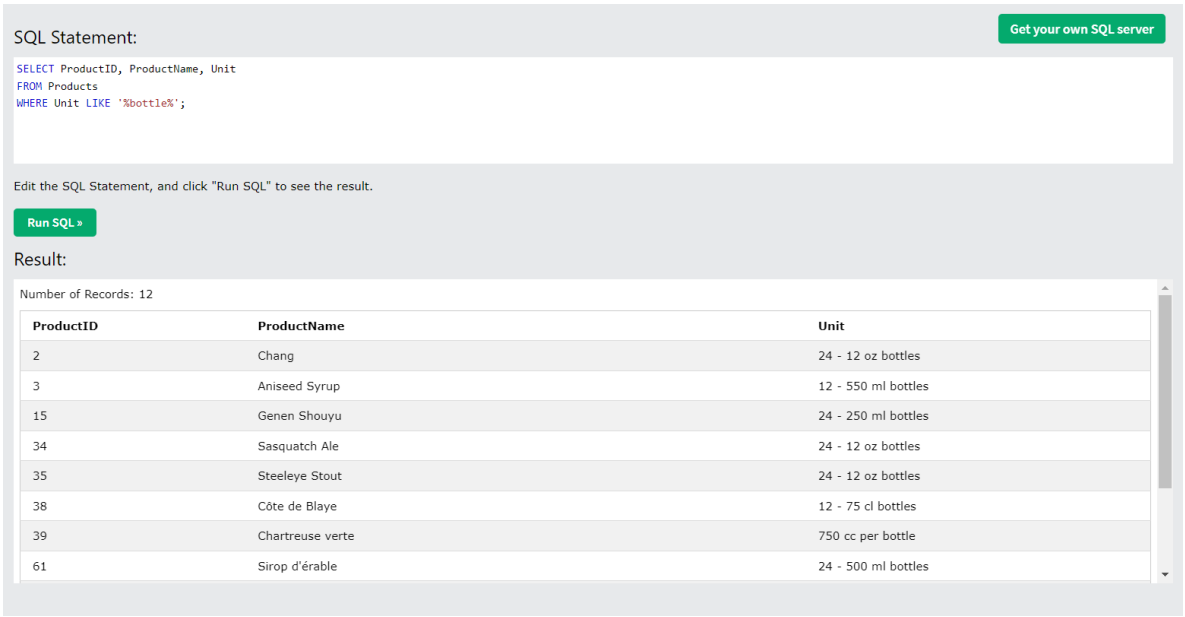
Result:

Number of Records: 8

CategoryName	CountOfOrders
Dairy Products	100
Beverages	93
Confections	84
Seafood	67
Meat/Poultry	50
Condiments	49
Grains/Cereals	42
Produce	33

Figure 2: Products in order of the most frequently ordered.

The products are divided into 8 categories and are sold by units - for example in bottles, which, as can be seen below, are also of different sizes.



SQL Statement:

Get your own SQL server

```
SELECT ProductID, ProductName, Unit
FROM Products
WHERE Unit LIKE '%bottle%';
```

Edit the SQL Statement, and click "Run SQL" to see the result.

Run SQL »

Result:

Number of Records: 12

ProductID	ProductName	Unit
2	Chang	24 - 12 oz bottles
3	Aniseed Syrup	12 - 550 ml bottles
15	Genen Shouyu	24 - 250 ml bottles
34	Sasquatch Ale	24 - 12 oz bottles
35	Steeleye Stout	24 - 12 oz bottles
38	Côte de Blaye	12 - 75 cl bottles
39	Chartreuse verte	750 cc per bottle
61	Sirop d'érable	24 - 500 ml bottles

Figure 3: Various sizes and volumes of products' bottles.

From all orders, I calculated the percentage of bottle products.

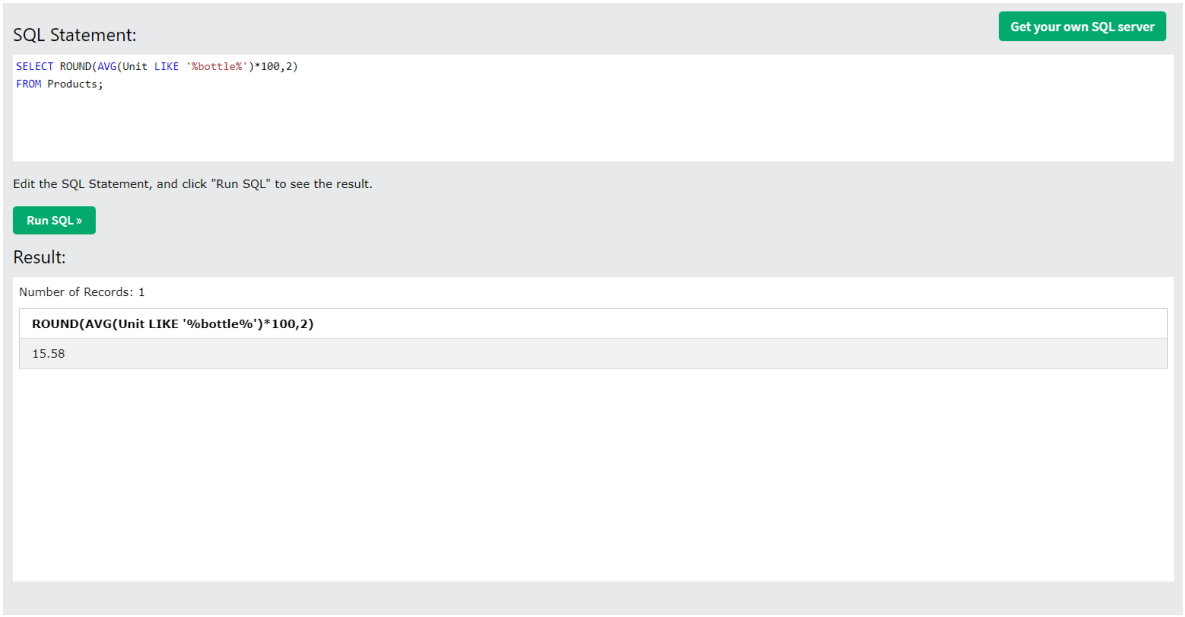


Figure 4: Share of products sold in bottles.

Based on number of sold products and their costs, I wrote a query that returned products which were the most profitable:

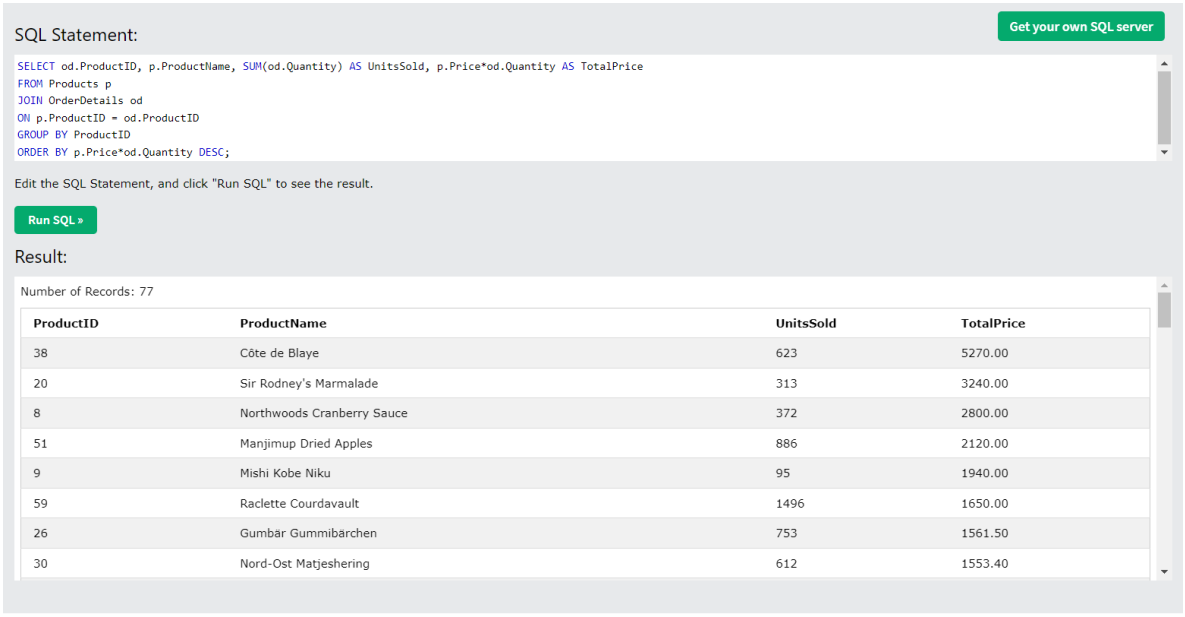
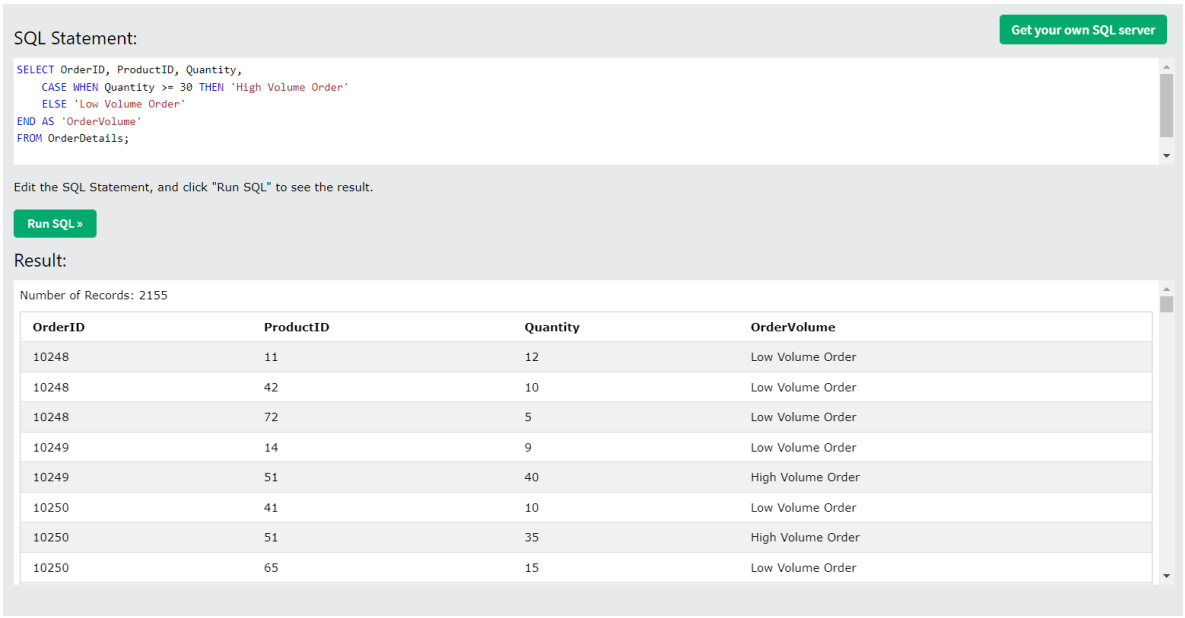


Figure 5: Most profitable products.

Products are usually ordered in bulk. The quantities vary from 1 to 120. Using case-when in SQL query, I assigned “Low Volume” and “High Volume” feature to each order.



SQL Statement:

Get your own SQL server

```
SELECT OrderID, ProductID, Quantity,
    CASE WHEN Quantity >= 30 THEN 'High Volume Order'
    ELSE 'Low Volume Order'
END AS 'OrderVolume'
FROM OrderDetails;
```

Edit the SQL Statement, and click "Run SQL" to see the result.

Run SQL »

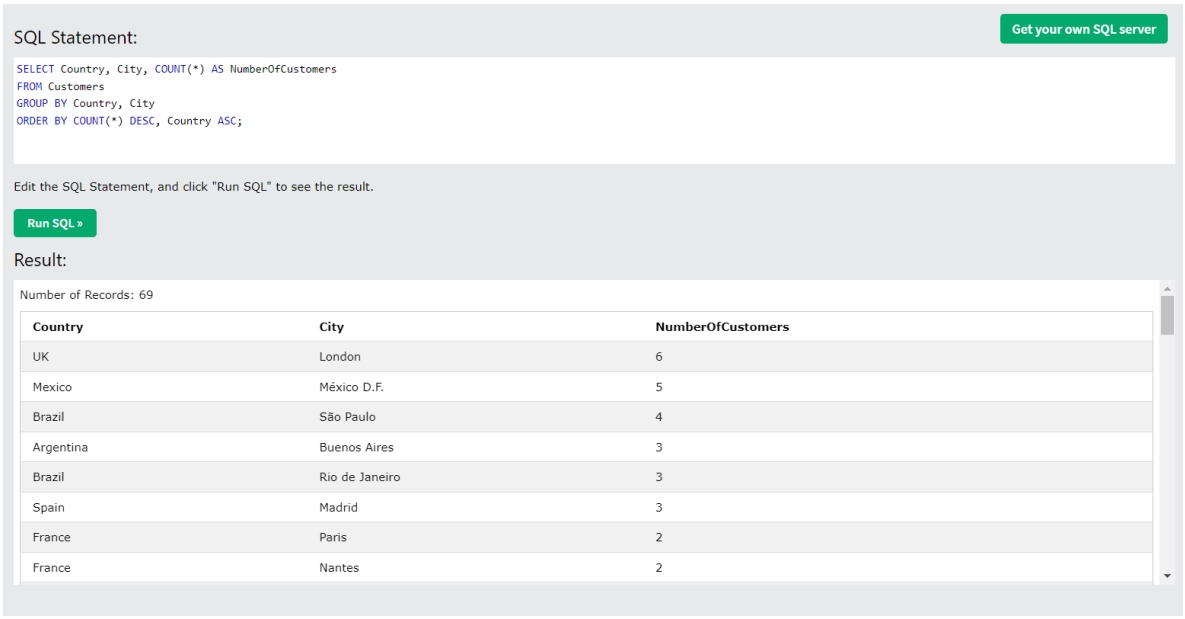
Result:

Number of Records: 2155

OrderID	ProductID	Quantity	OrderVolume
10248	11	12	Low Volume Order
10248	42	10	Low Volume Order
10248	72	5	Low Volume Order
10249	14	9	Low Volume Order
10249	51	40	High Volume Order
10250	41	10	Low Volume Order
10250	51	35	High Volume Order
10250	65	15	Low Volume Order

Figure 6: New feature of each ordered product - volume.

Now, let’s focus on customers. In screenshot below, you can see results of query which presents where the customers are from.



SQL Statement:

Get your own SQL server

```
SELECT Country, City, COUNT(*) AS NumberOfCustomers
FROM Customers
GROUP BY Country, City
ORDER BY COUNT(*) DESC, Country ASC;
```

Edit the SQL Statement, and click "Run SQL" to see the result.

Run SQL »

Result:

Number of Records: 69

Country	City	NumberOfCustomers
UK	London	6
Mexico	México D.F.	5
Brazil	São Paulo	4
Argentina	Buenos Aires	3
Brazil	Rio de Janeiro	3
Spain	Madrid	3
France	Paris	2
France	Nantes	2

Figure 7: Countries and cities of customers.

Using *union* I created a combined list of all contacts - to customers and suppliers.

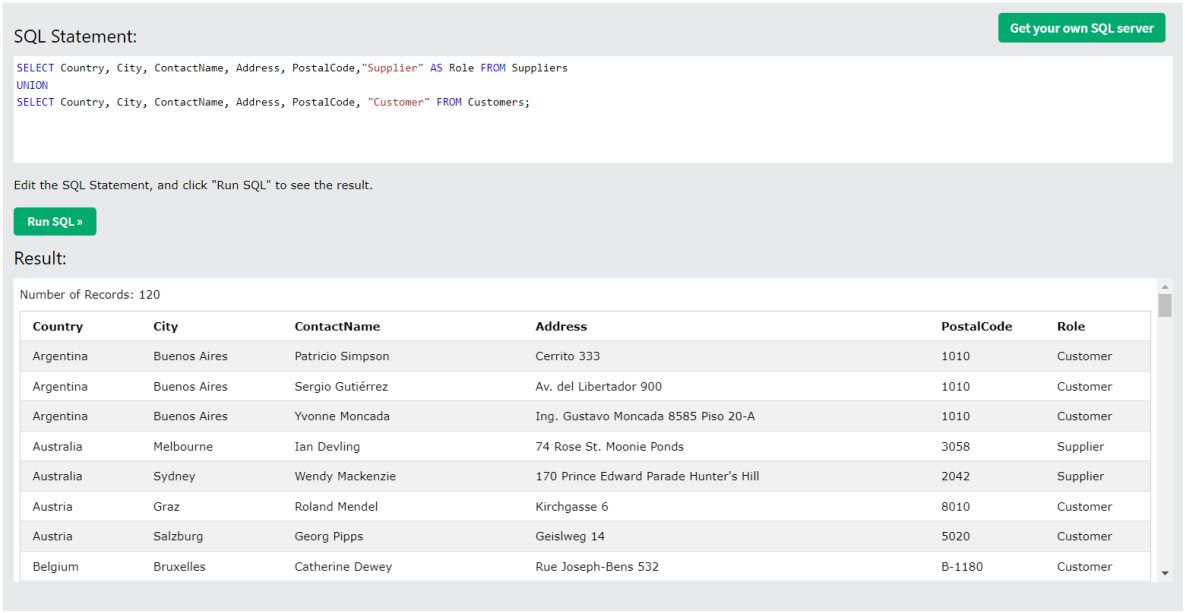


Figure 8: Customers and suppliers - contact information.

In screenshot below, you can see how the number of orders changes over time for each year and its month.

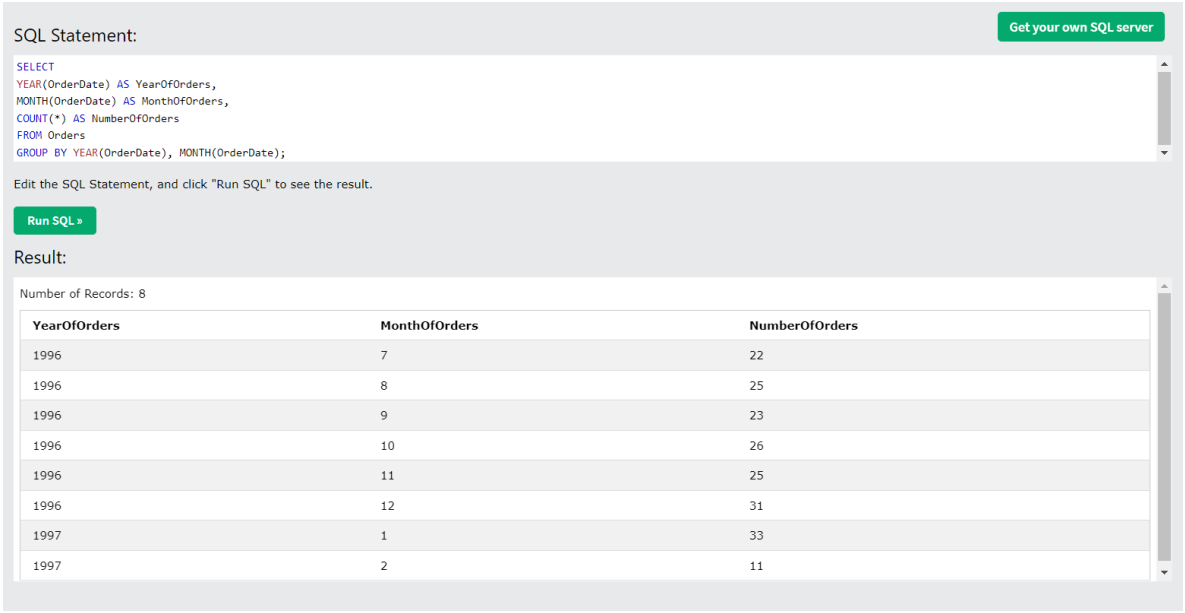
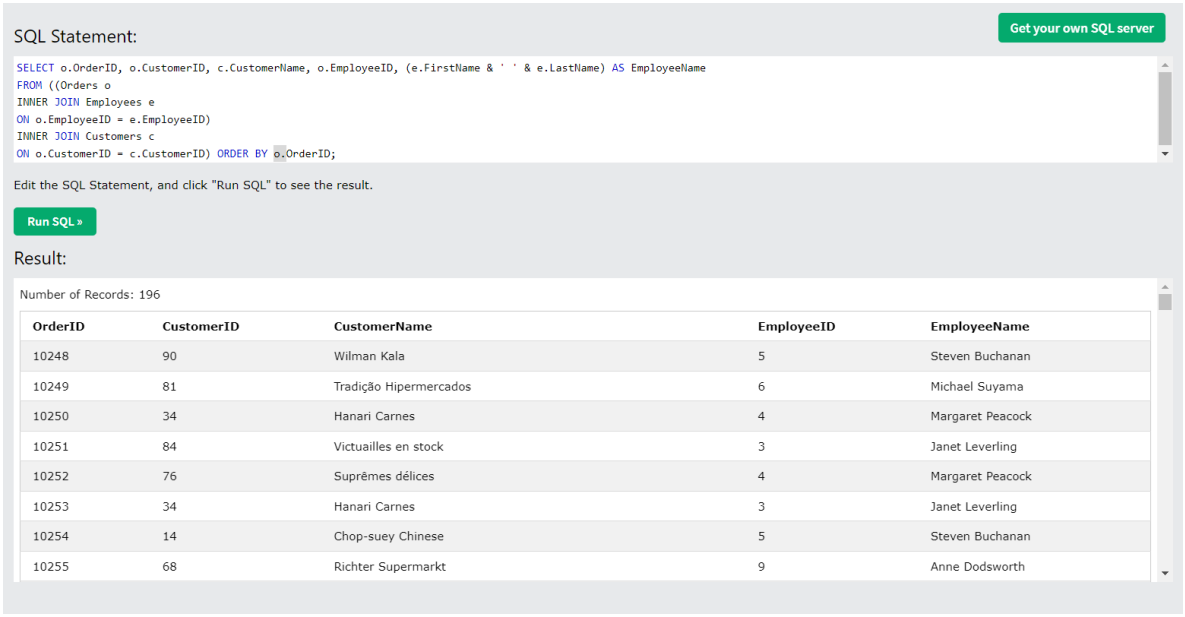


Figure 9: Number of orders over time.

For each order, we can see the list of its customer and employee responsible for the order.



SQL Statement:

Get your own SQL server

```
SELECT o.OrderID, o.CustomerID, c.CustomerName, o.EmployeeID, (e.FirstName & ' ' & e.LastName) AS EmployeeName
FROM ((Orders o
INNER JOIN Employees e
ON o.EmployeeID = e.EmployeeID)
INNER JOIN Customers c
ON o.CustomerID = c.CustomerID) ORDER BY o.OrderID;
```

Edit the SQL Statement, and click "Run SQL" to see the result.

Run SQL »

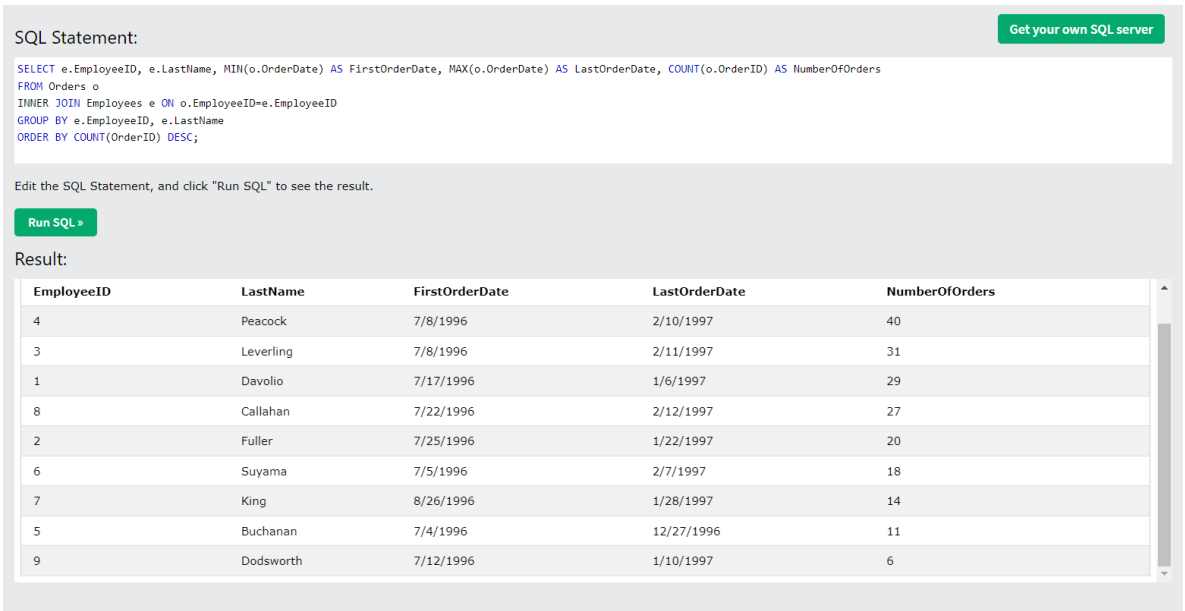
Result:

Number of Records: 196

OrderID	CustomerID	CustomerName	EmployeeID	EmployeeName
10248	90	Wilman Kala	5	Steven Buchanan
10249	81	Tradição Hipermercados	6	Michael Suyama
10250	34	Hanari Carnes	4	Margaret Peacock
10251	84	Victuailles en stock	3	Janet Leverling
10252	76	Suprêmes délices	4	Margaret Peacock
10253	34	Hanari Carnes	3	Janet Leverling
10254	14	Chop-suey Chinese	5	Steven Buchanan
10255	68	Richter Supermarkt	9	Anne Dodsworth

Figure 10: Customers and employees of each order.

With that, statistics for employees can be easily obtained.



SQL Statement:

Get your own SQL server

```
SELECT e.EmployeeID, e.LastName, MIN(o.OrderDate) AS FirstOrderDate, MAX(o.OrderDate) AS LastOrderDate, COUNT(o.OrderID) AS NumberOfOrders
FROM Orders o
INNER JOIN Employees e ON o.EmployeeID=e.EmployeeID
GROUP BY e.EmployeeID, e.LastName
ORDER BY COUNT(OrderID) DESC;
```

Edit the SQL Statement, and click "Run SQL" to see the result.

Run SQL »

Result:

EmployeeID	LastName	FirstOrderDate	LastOrderDate	NumberOfOrders
4	Peacock	7/8/1996	2/10/1997	40
3	Leverling	7/8/1996	2/11/1997	31
1	Davolio	7/17/1996	1/6/1997	29
8	Callahan	7/22/1996	2/12/1997	27
2	Fuller	7/25/1996	1/22/1997	20
6	Suyama	7/5/1996	2/7/1997	18
7	King	8/26/1996	1/28/1997	14
5	Buchanan	7/4/1996	12/27/1996	11
9	Dodsworth	7/12/1996	1/10/1997	6

Figure 11: Employees in sequence of number of conducted orders, from the most orders.