# 

# High Impact Skills Development Program

# in Artificial Intelligence, Data Science, and Blockchain

**Project Title**: Online Retail Segmentation. **Module 06:** Data Mining

**Name : Muhammad Aslam Roll-no :SK23008**

**Training center: Skardu Section :02**

**email :** [**maslambaig009@gmail.com**](mailto:maslambaig009@gmail.com)

**github link :** **https://github.com/Aslam009/ORS-Data-mining**

**Beginner Queries:**

**1- Define meta data in mysql workbench**

1. InvoiceNo:

- Data Type: VARCHAR

2. StockCode:

- Data Type: VARCHAR (to store alphanumeric stock codes)

- Constraints: PRIMARY KEY ( StockCode is unique)

3. Description:

- Data Type: TEXT

- Constraints: None

4. Quantity:

- Data Type: INT

- Constraints: (constraint to ensure positive values)

5. InvoiceDate:

- Data Type: DATETIME or TIMESTAMP

- Constraints: None (not null)

6. UnitPrice:

- Data Type: DECIMAL

- Constraints: None (non-negative values)

7. CustomerID:

- Data Type: INT (assuming customer IDs are represented as integers)

- Constraints: FOREIGN KEY (if linked to a Customers table) or None

8. Country:

- Data Type: VARCHAR

- Constraints: None

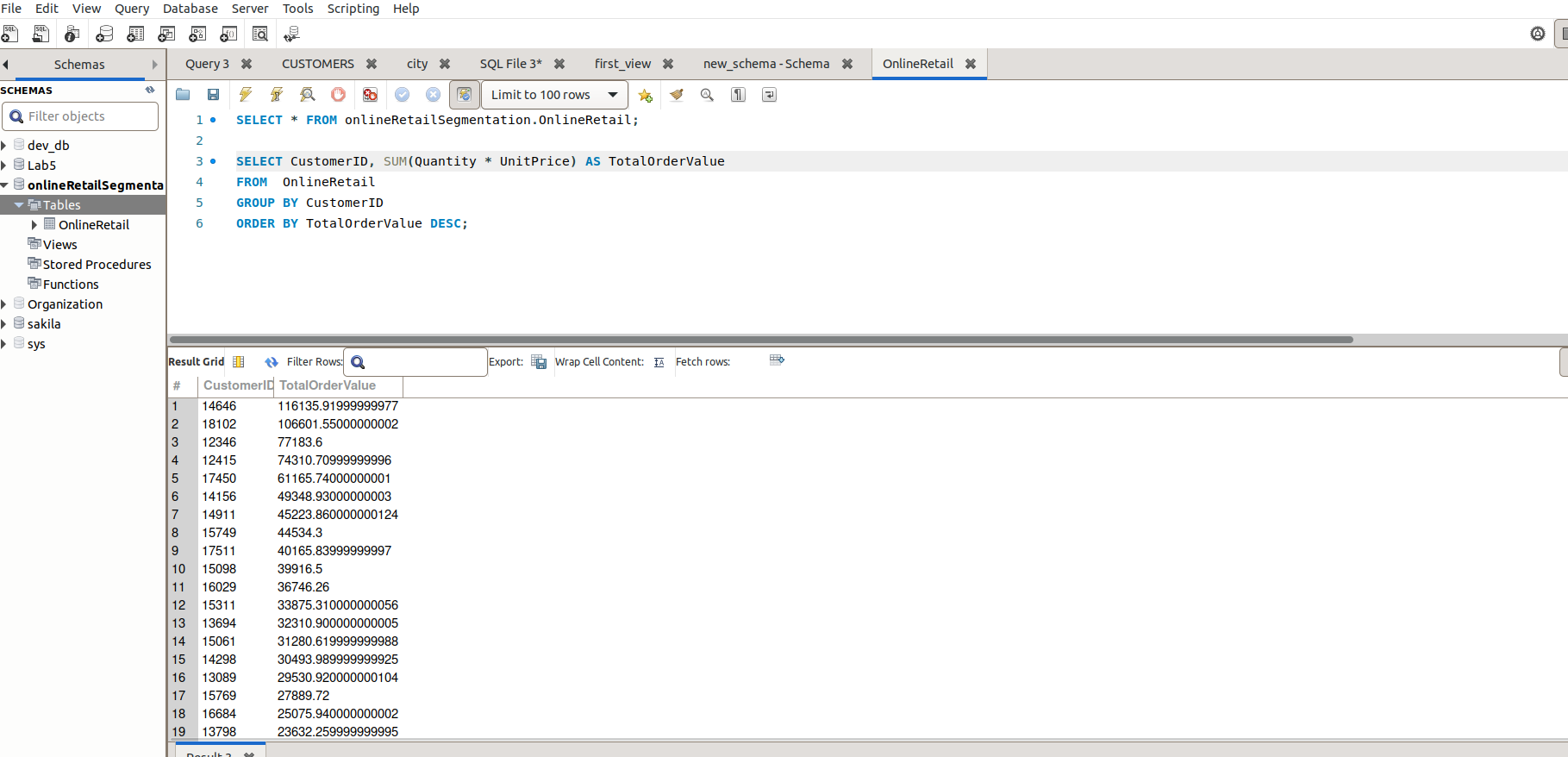
2:**What is the distribution of order values across all customers in the dataset?**

SELECT CustomerID, SUM(Quantity \* UnitPrice) AS TotalOrderValue

FROM OnlineRetail

GROUP BY CustomerID

ORDER BY TotalOrderValue DESC;



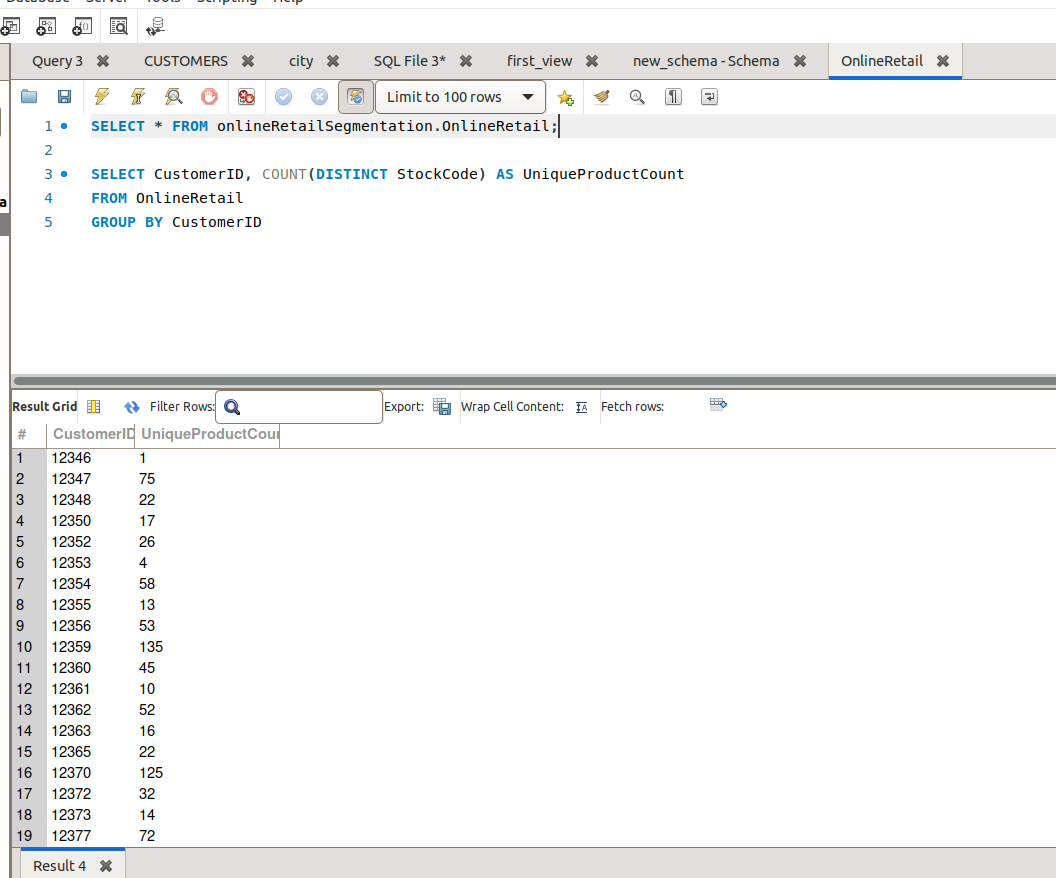
This SQL query get information from customers' total order values from the "OnlineRetail" table. The query calculates the sum of the product of quantity and unit price for each customer's transactions. The results are then grouped by customer and ordered in descending order based on the total order value.

**3:How many unique products has each customer purchased?**

SELECT CustomerID, COUNT(DISTINCT StockCode) AS UniqueProductCount

FROM OnlineRetail

GROUP BY CustomerID

This SQL query retrieves information about the count of unique products that each customer has purchased from the "OnlineRetail" table. The query calculates the number of distinct product codes (StockCodes) associated with each customer's transactions. The results are grouped by customer.

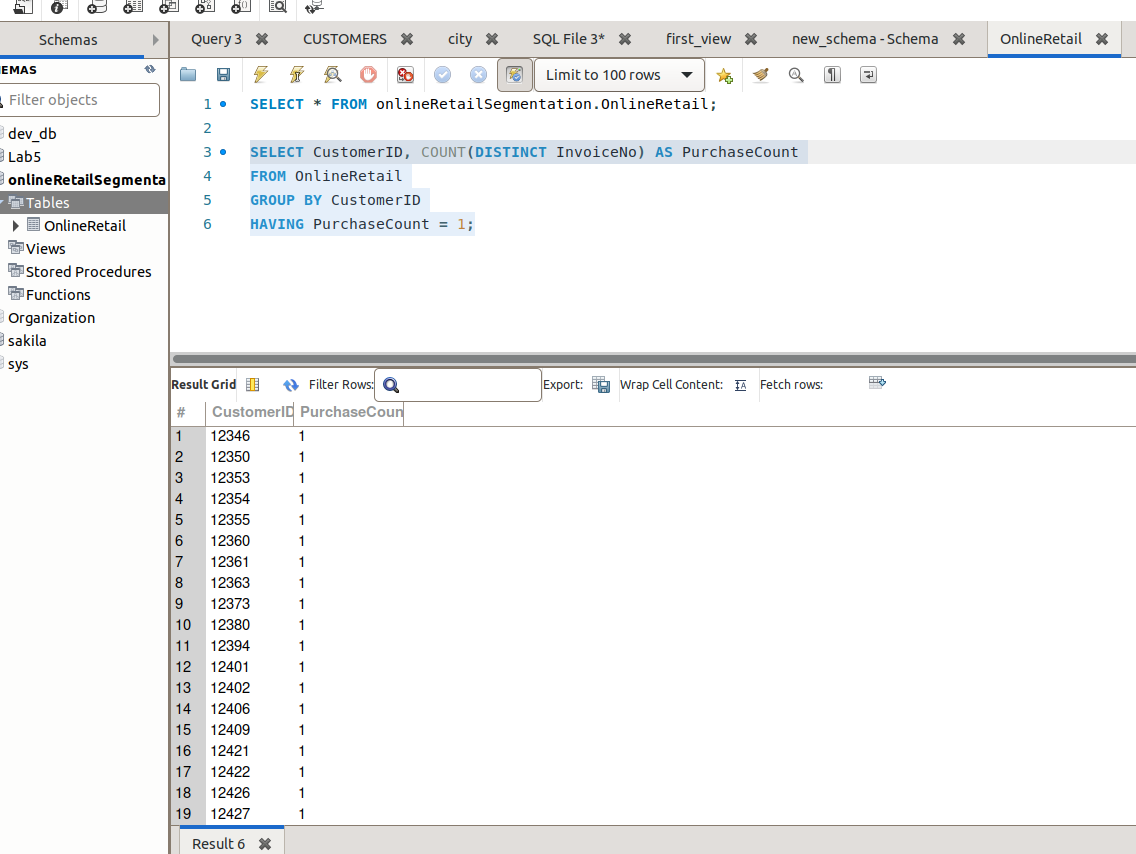
**4:Which customers have only made a single purchase from the company?**

**SELECT CustomerID, COUNT(DISTINCT InvoiceNo) AS PurchaseCount**

**FROM OnlineRetail**

**GROUP BY CustomerID**

**HAVING PurchaseCount = 1;**



This SQL query get information from customers who have a single purchase from the "OnlineRetail" table. The query calculates the count of distinct invoice numbers (InvoiceNos) associated with each customer's transactions and filters the results to include only customers with a purchase count of 1.

**5:Which products are most commonly purchased together by customers in the dataset?**

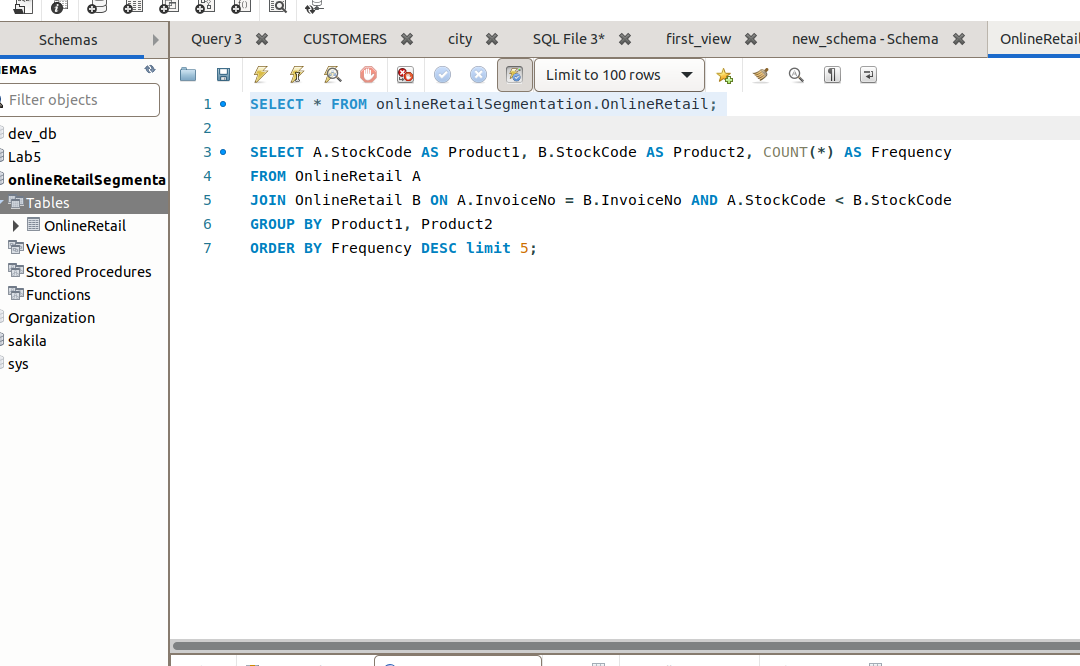
**SELECT A.StockCode AS Product1, B.StockCode AS Product2, COUNT(\*) AS Frequency**

**FROM OnlineRetail A**

**JOIN OnlineRetail B ON A.InvoiceNo = B.InvoiceNo AND A.StockCode < B.StockCode**

**GROUP BY Product1, Product2**

**ORDER BY Frequency DESC limit 5;**



This SQL query is to find out the frequency of pairs of products that are commonly purchased together in the "OnlineRetail" table. The query performs a self-join on the "OnlineRetail" table to compare different products within the same invoices.

## **Advance Queries :**

**1: Customer Segmentation by Purchase Frequency**

SELECT CustomerID,

CASE

WHEN PurchaseCount > 5 THEN 'High Frequency'

WHEN PurchaseCount > 2 THEN 'Medium Frequency'

ELSE 'Low Frequency'

END AS PurchaseFrequencySegment

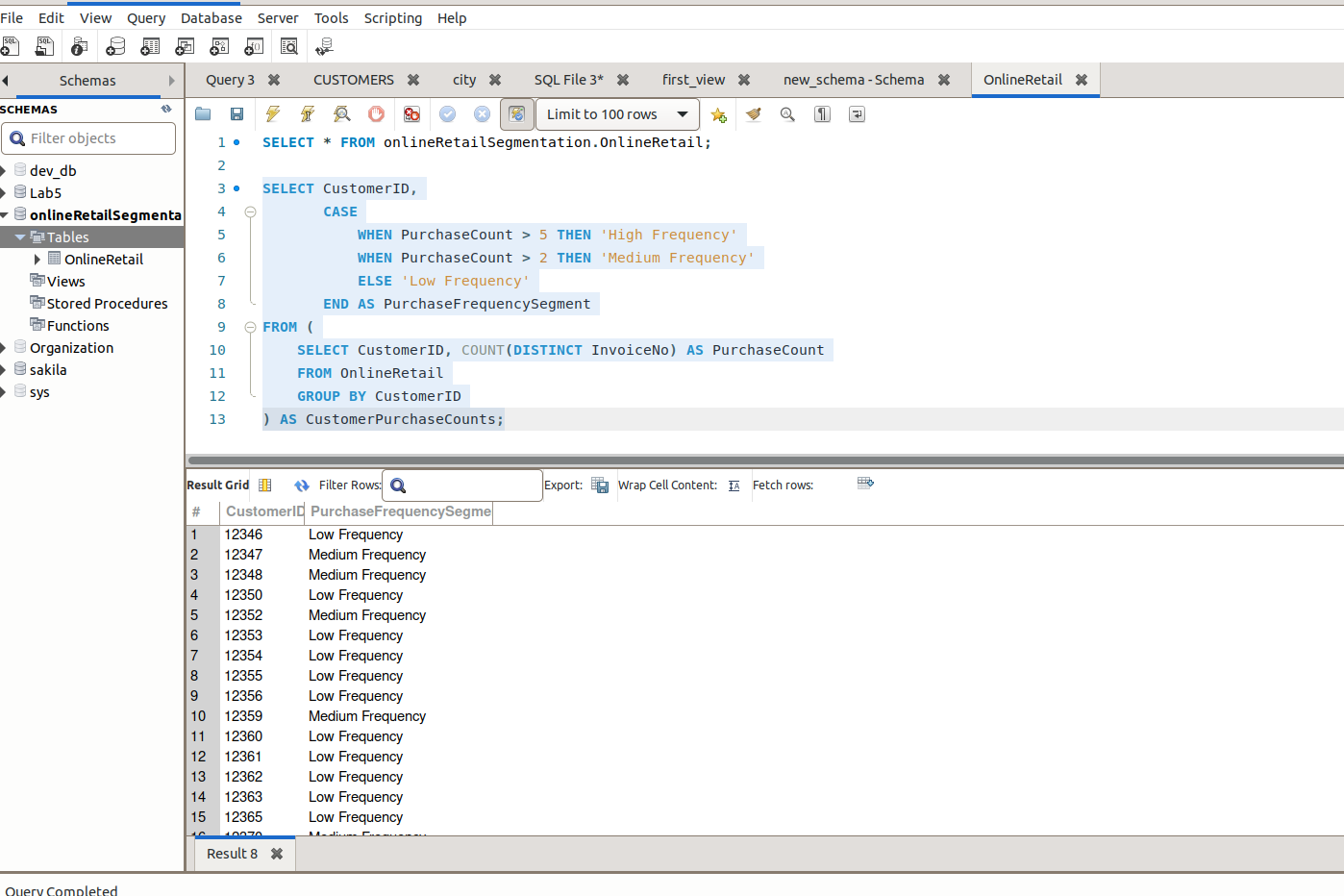
FROM (

SELECT CustomerID, COUNT(DISTINCT InvoiceNo) AS PurchaseCount

FROM OnlineRetail

GROUP BY CustomerID

) AS CustomerPurchaseCounts;



This query selects data from the subquery and applies a CASE statement to categorize customers based on their purchase frequency.

**2:Average Order Value by Country**

SELECT Country, AVG(TotalOrderValue) AS AverageOrderValue

FROM (

SELECT Country, InvoiceNo, SUM(Quantity \* UnitPrice) AS TotalOrderValue

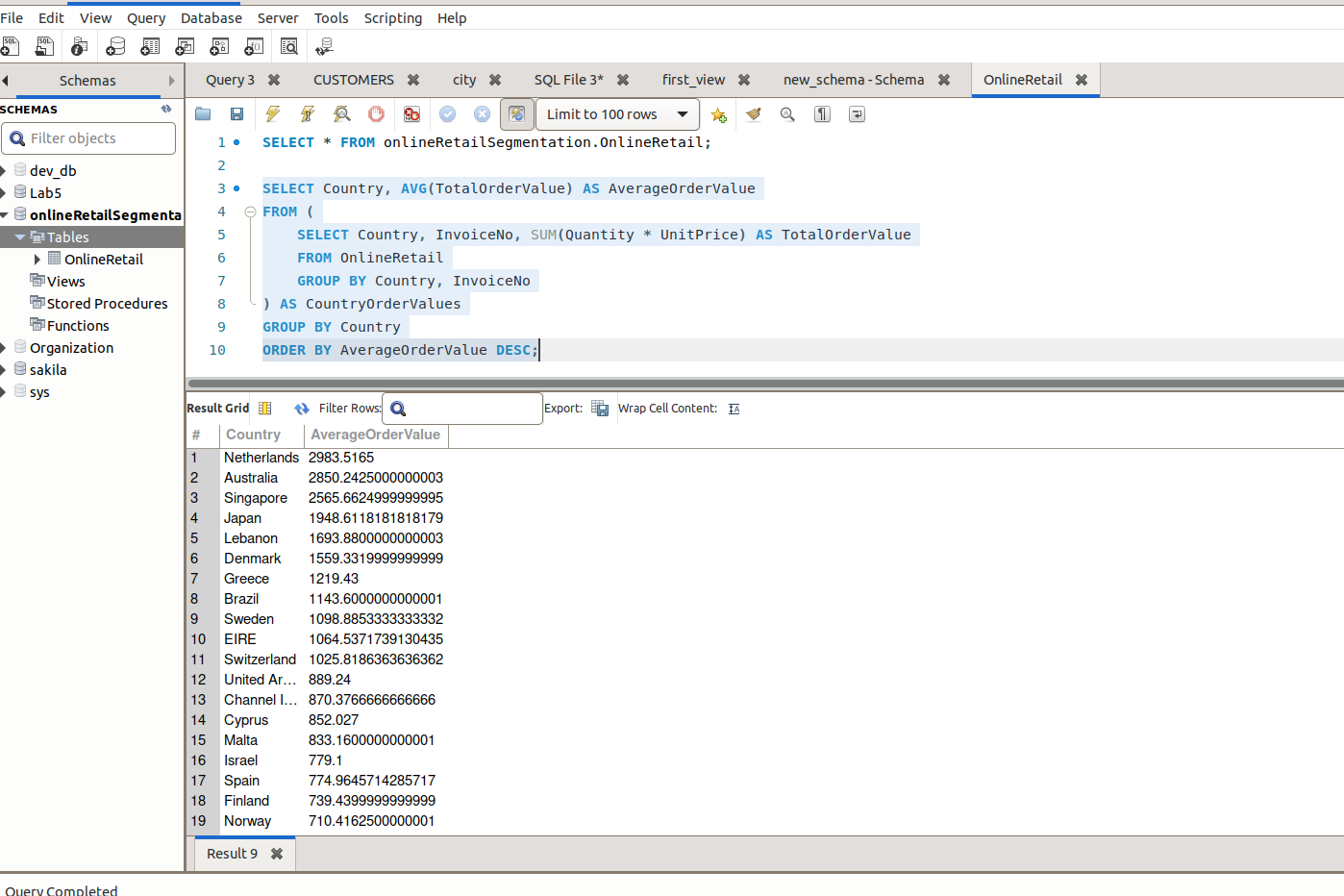
FROM OnlineRetail

GROUP BY Country, InvoiceNo

) AS CountryOrderValues

GROUP BY Country

**ORDER BY AverageOrderValue DESC;**



This SQL query calculates the average order value for each country in the "OnlineRetail" dataset. It involves nested subqueries to first calculate the total order value for each invoice within each country, and then it calculates the average order value for each country.

**3:Customer Churn Analysis**

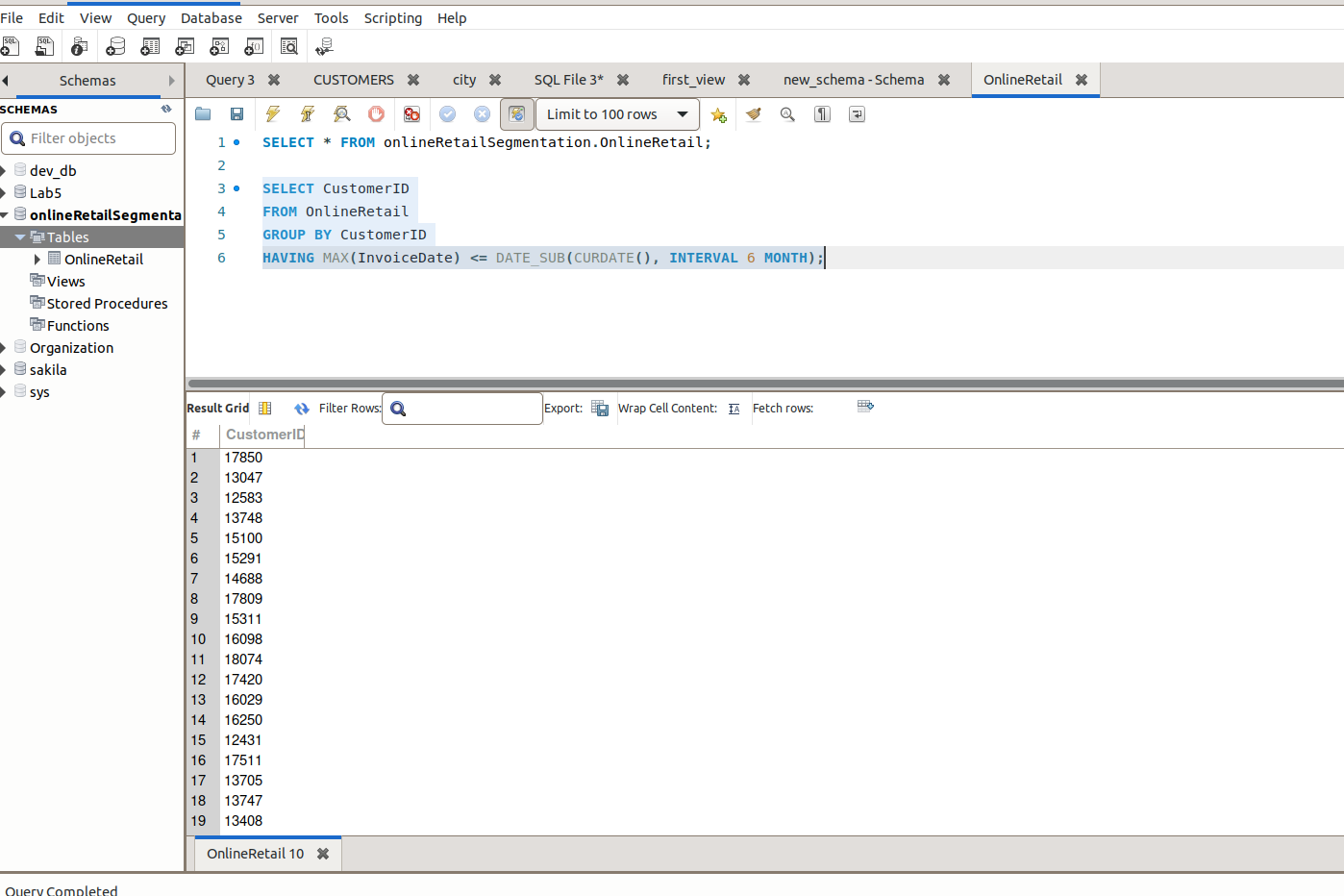
**Identify customers who haven't made a purchase in a specific period (e.g., last 6 months) to assess churn.**

SELECT CustomerID

FROM OnlineRetail

GROUP BY CustomerID

**HAVING MAX(InvoiceDate) <= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH);**



4:**Product Affinity Analysis**

**Determine which products are often purchased together by calculating the correlation between product purchases.**

SELECT

DATE\_FORMAT(InvoiceDate, '%Y-%m') AS Month,

SUM(TotalOrderValue) AS TotalSales

FROM (

SELECT

InvoiceDate,

SUM(Quantity \* UnitPrice) AS TotalOrderValue

FROM

OnlineRetail

GROUP BY

InvoiceNo, InvoiceDate

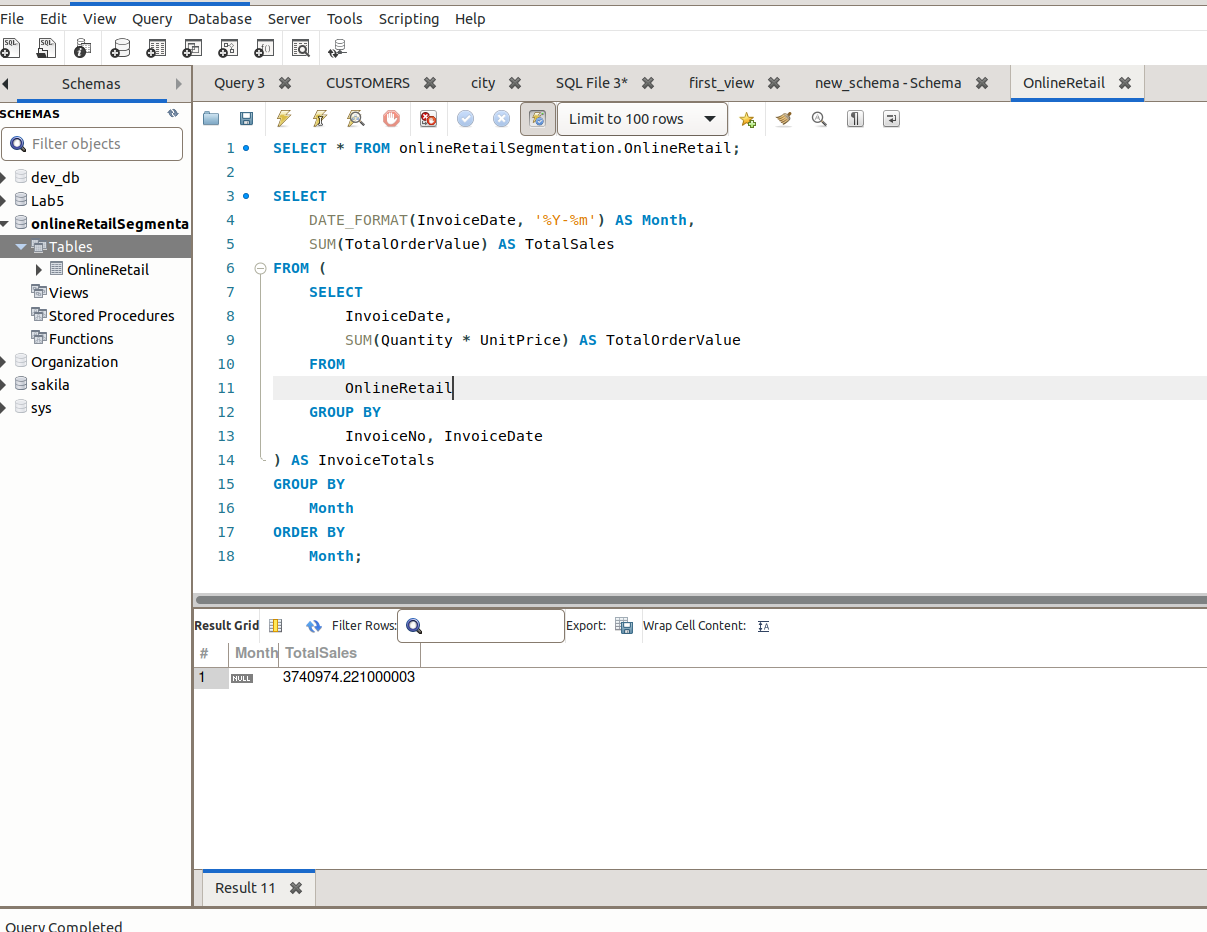
) AS InvoiceTotals

GROUP BY

Month

ORDER BY

Month;



This SQL query is used to summarize the total sales for each month in the "OnlineRetail" table. The query involves nested subqueries to first calculate the total order value for each invoice and then aggregates those values on a monthly basis. Here's a breakdown of each part of the query:

5:**Time-based Analysis**

**Explore trends in customer behavior over time, such as monthly or quarterly sales patterns.**

SELECT

p1.StockCode AS Product1,

p2.StockCode AS Product2,

COUNT(DISTINCT p1.InvoiceNo) AS CoPurchaseCount

FROM

OnlineRetail p1

JOIN

OnlineRetail p2 ON p1.InvoiceNo = p2.InvoiceNo AND p1.StockCode < p2.StockCode

GROUP BY

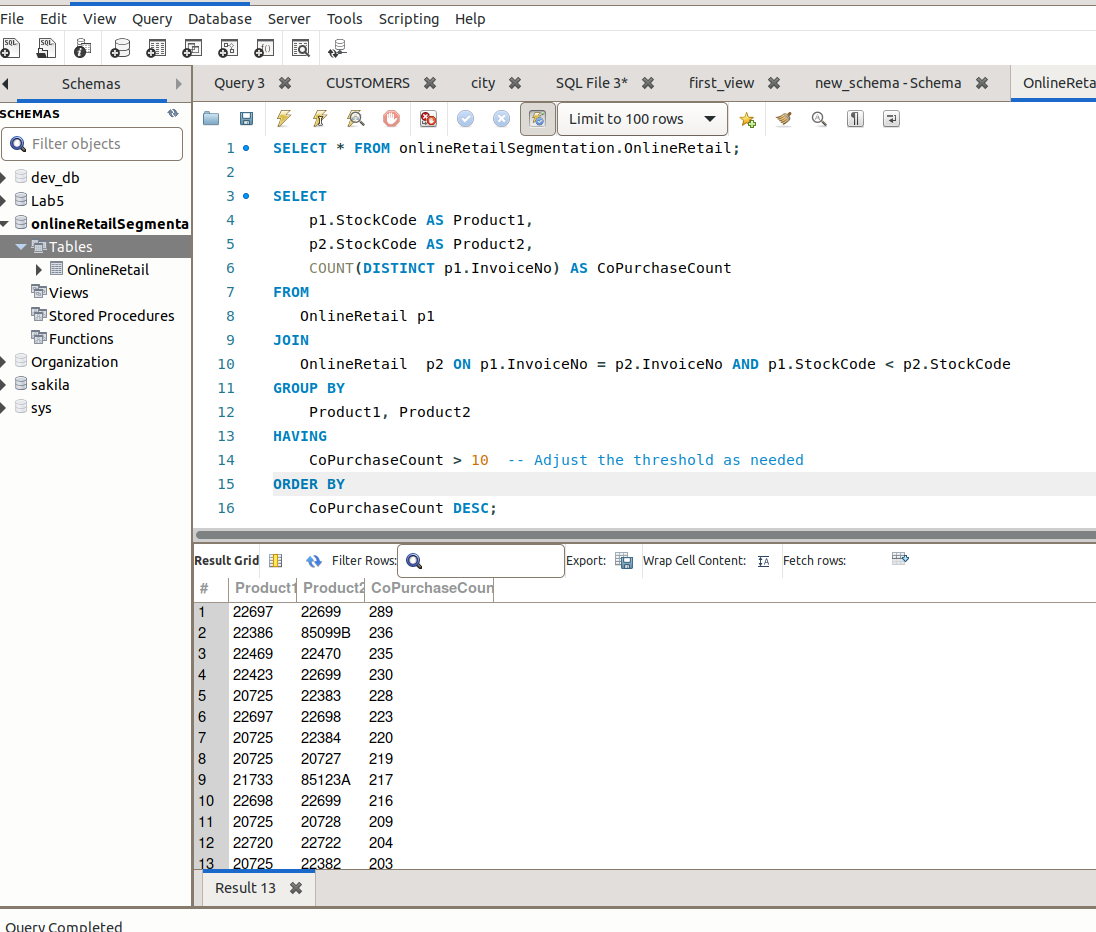
Product1, Product2

HAVING

CoPurchaseCount > 10 -- Adjust the threshold as needed

ORDER BY

CoPurchaseCount DESC



This SQL query is used to identify pairs of products that are commonly purchased together in the "OnlineRetail" table. The query performs a self-join on the "OnlineRetail" table to compare different products within the same invoices and calculates the count of distinct invoices where each pair of products was co-purchased.

The End