|  |  |
| --- | --- |
|  |  |
|  |
|  |

**High Impact Skills Development Program for Gilgit Baltistan**

**Computer Vision Module Project**

**Project Title**: Online Retail Segmentation.

**Name: Maqsood**

**Email Address:** [**qmrmqsd420@gmail.com**](mailto:qmrmqsd420@gmail.com)

**GitHub Link:** [**https://github.com/maqsood947/-SQLproject/blob/main/my%20sql%20project.sql**](https://github.com/maqsood947/-SQLproject/blob/main/my%20sql%20project.sql)

**Summary**

In this project, we explore how data can help businesses understand their customers and make smarter decisions. We'll use a tool called MySQL Workbench to analyze data from an online store. The data includes things like what customers bought, when, and how much they spent.

At the start, we'll set up our data in MySQL Workbench. Then, we'll use SQL to answer different questions and discover interesting things about customers.

For example, we can find out:

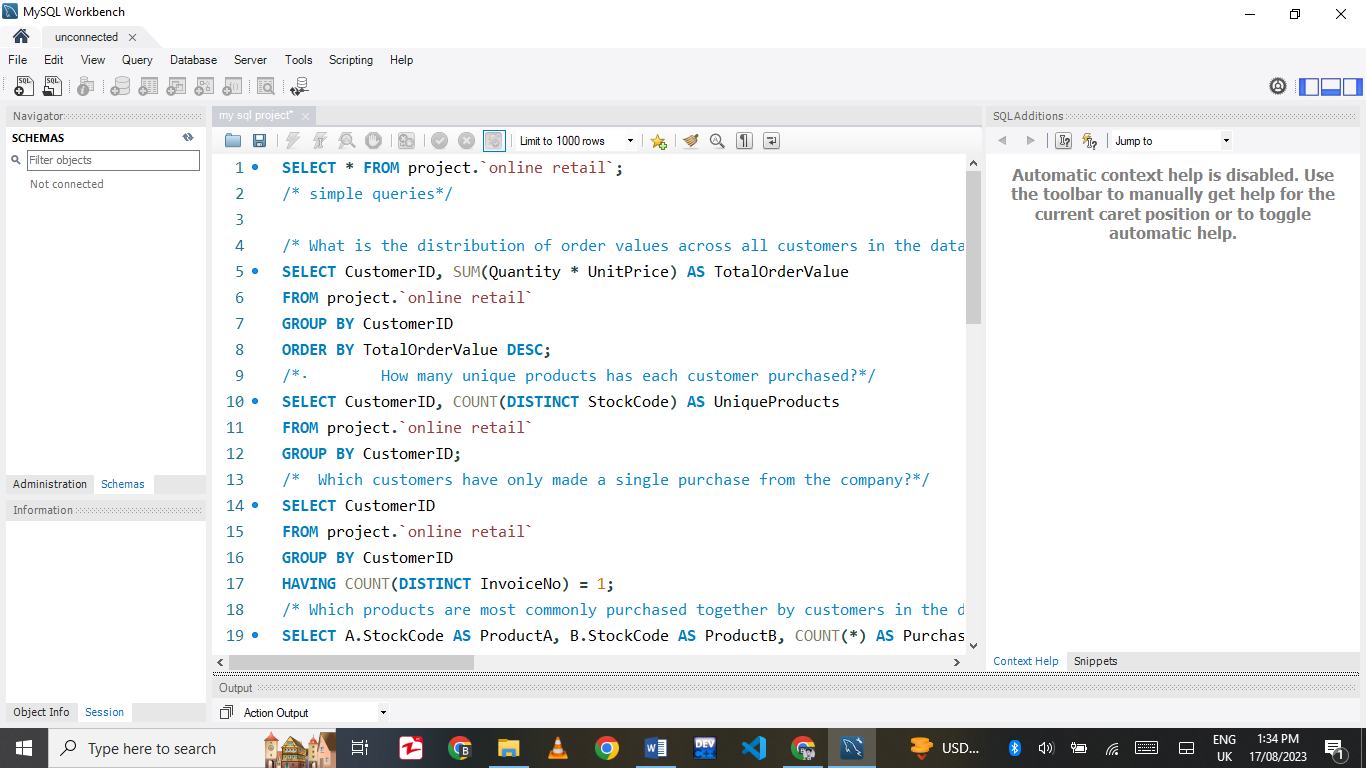
* How much money each customer spent.
* How many different products each customer bought.
* Which customers bought only once?
* Which products are often bought together?

In the advanced part, we'll do even cooler things:

* Group customers by how often they shop (like loyal, medium, or rare shoppers).
* Find out which countries have valuable customers.
* Identify customers who stopped buying recently.
* See which products are often bought together.
* Look at how customer behavior changes over time.

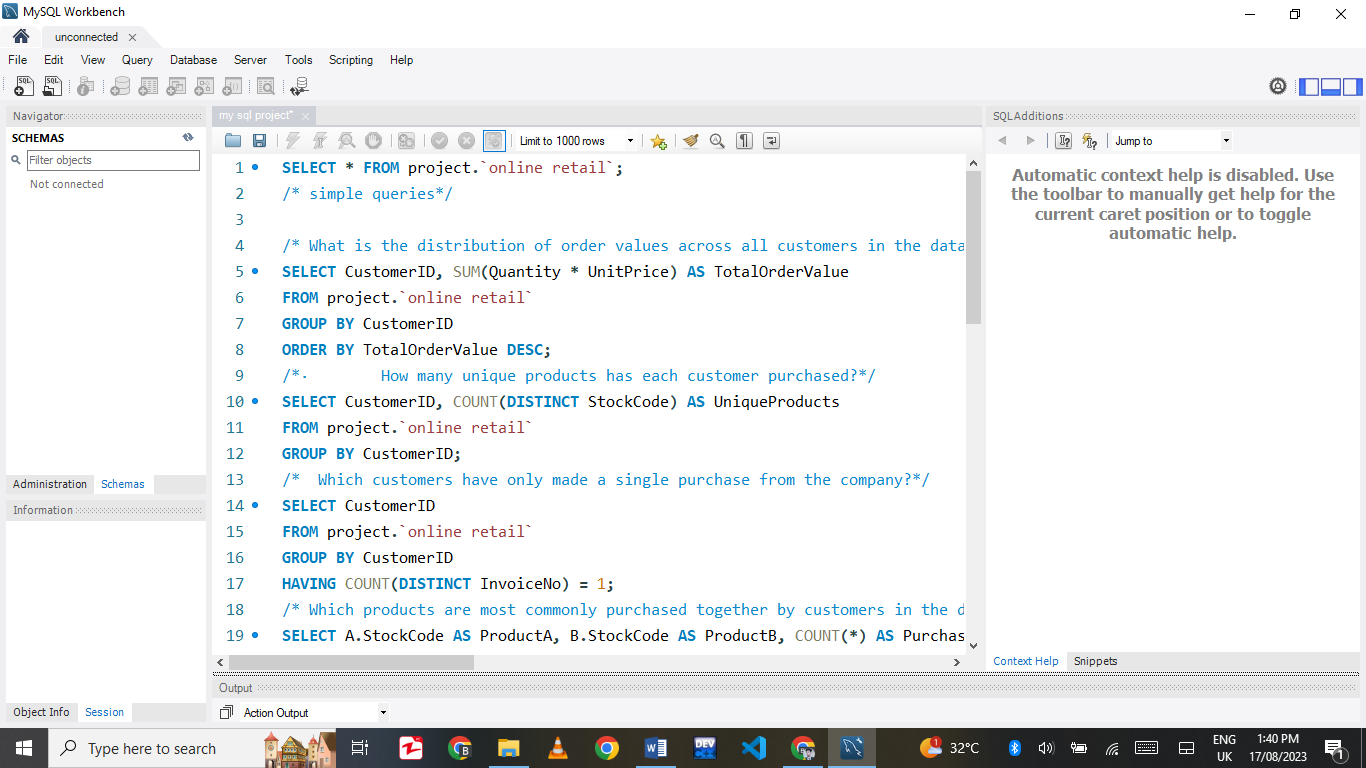
**Following are some of the queries with screenshots that will help us understand how we performed SQL queries to extract important information from given data:**

1. **What is the distribution of order values across all customers in the dataset?**



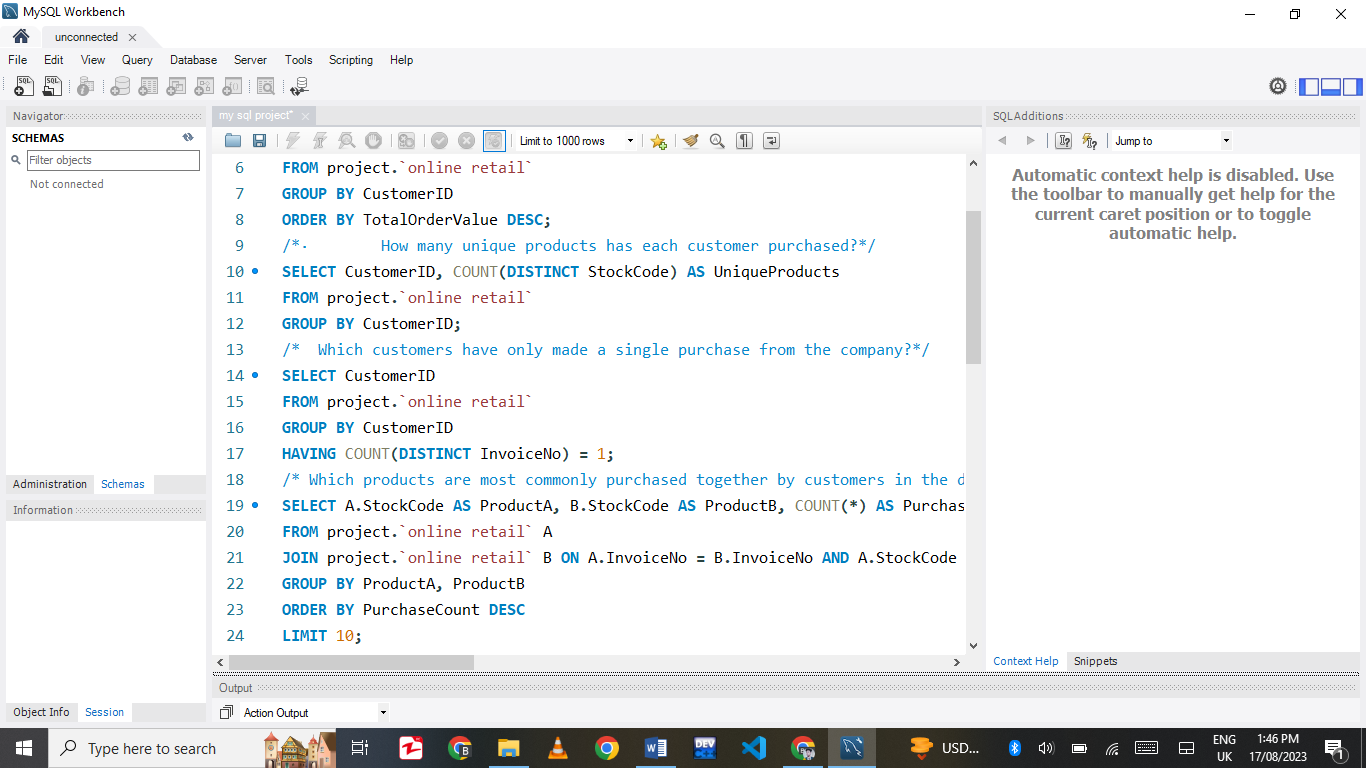
In this step, we're using SQL to find how much each customer spent in the online store. We multiply the quantity of each item they bought by its price to get its cost. Then, we add these costs for all their items to find the total spent. The `GROUP BY Customer ID` groups this calculation for each customer. This helps identify high-spending customers, contributing to store revenue insights.Top of Form

1. **How many unique products has each customer purchased?**



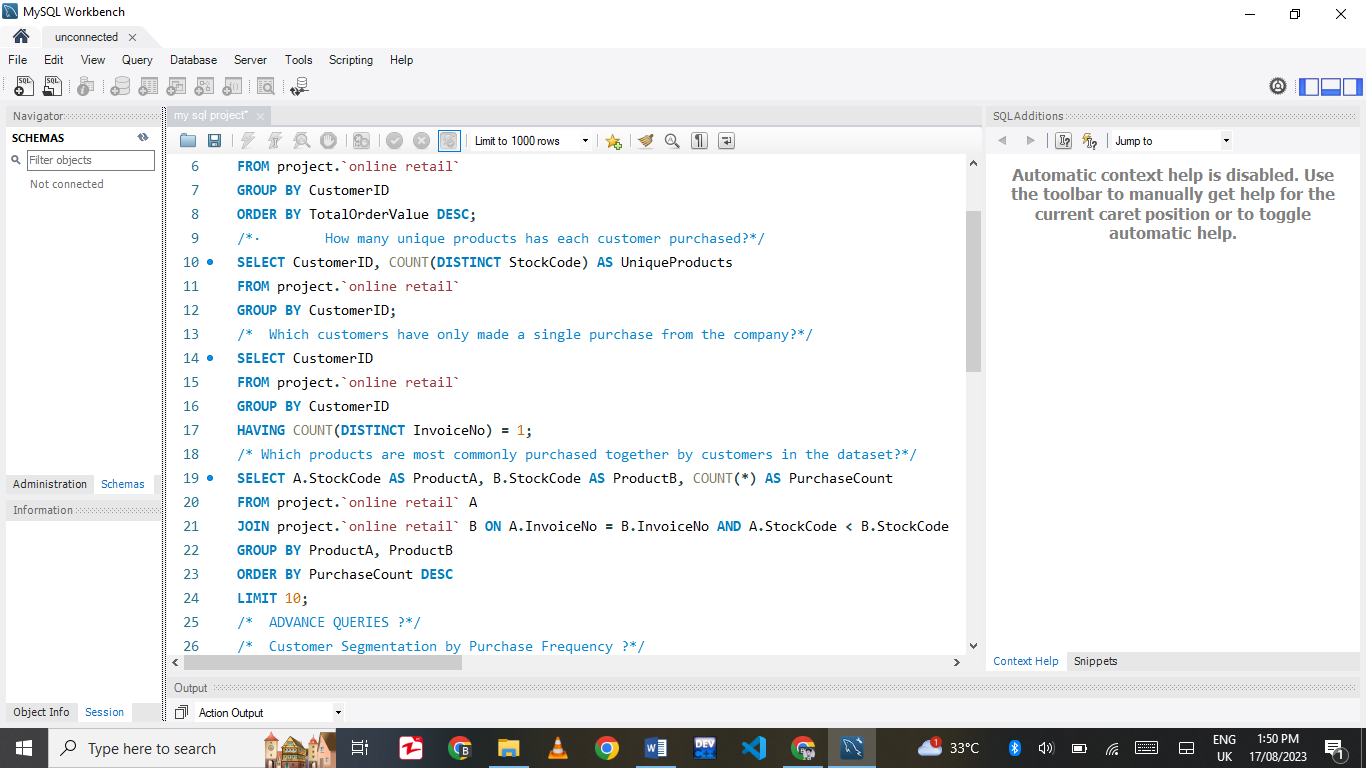
This SQL query counts the number of unique products each customer has purchased from the online store. For each customer, it tallies the different product codes. By using `COUNT (DISTINCT Stock Code)`, we determine the unique products. The query groups this count for each customer using `Customer ID`. This analysis helps understand customers' varied product preferences and aids in personalized marketing strategies.

1. **Which customers have only made a single purchase from the company?**



This query extracts CustomerIDs from the OnlineRetailData table. It groups the data by CustomerID. The "HAVING COUNT(DISTINCT InvoiceNo) = 1" condition filters and shows only those customers who made just one unique purchase (having only one InvoiceNo). This helps identify customers who have made only a single transaction with the company.

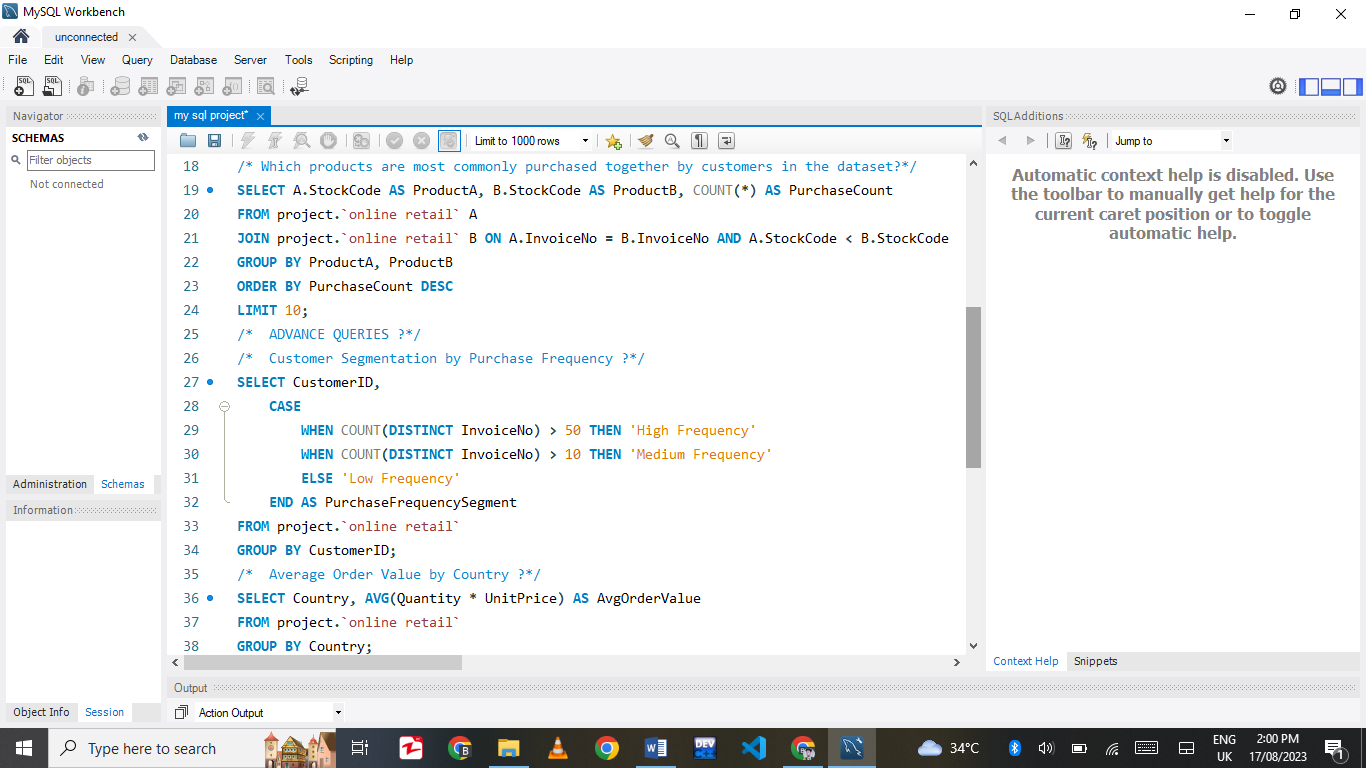
1. **Which products are most commonly purchased together by customers in the dataset?**



This SQL query finds products frequently purchased together by customers in the dataset. It connects two instances of the "online retail" table. It matches items with the same InvoiceNo and different StockCodes. By grouping the results by ProductA and ProductB, it calculates the count of these purchases (PurchaseCount). The query then orders the results by PurchaseCount in descending order and shows the top 10 pairs. This analysis helps understand which products tend to be bought together, aiding in product recommendations and marketing strategies.

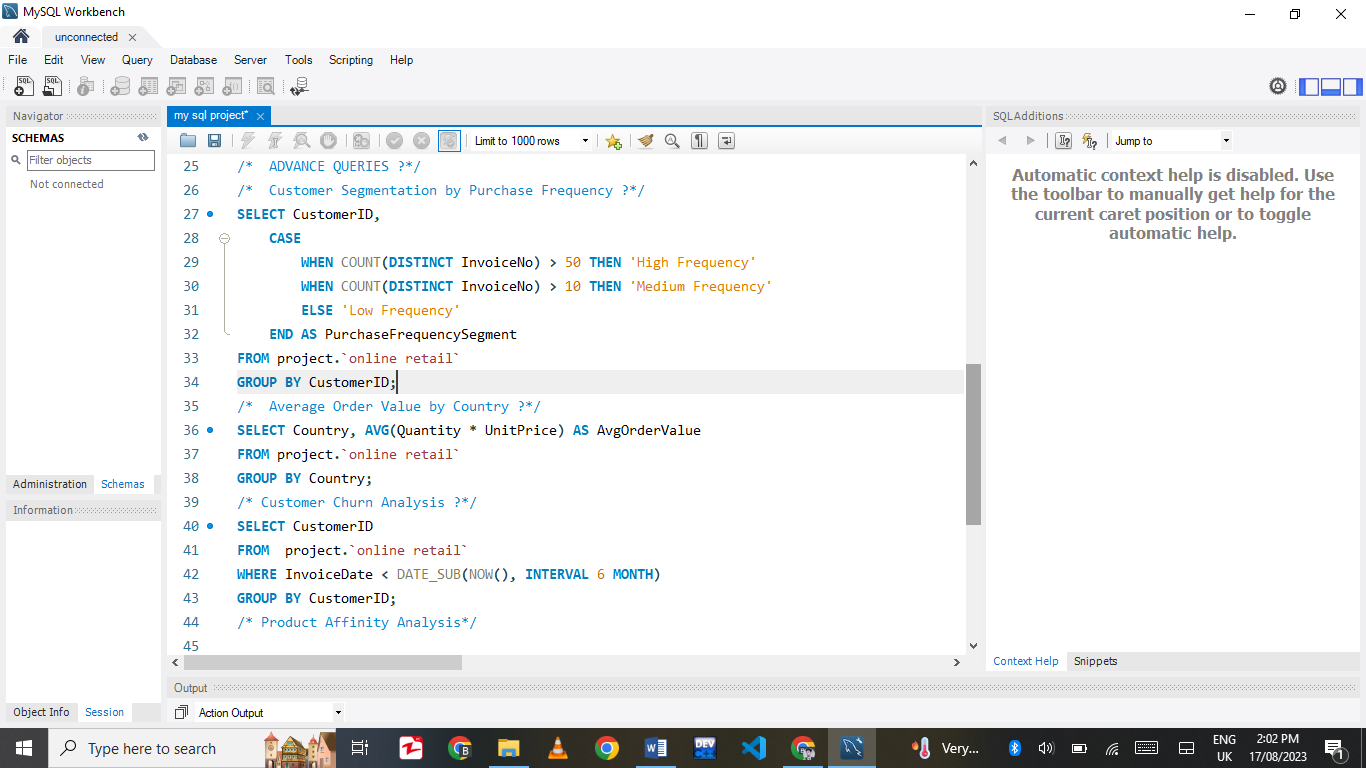
## **Advance Queries**

* 1. **Customer Segmentation by Purchase Frequency**



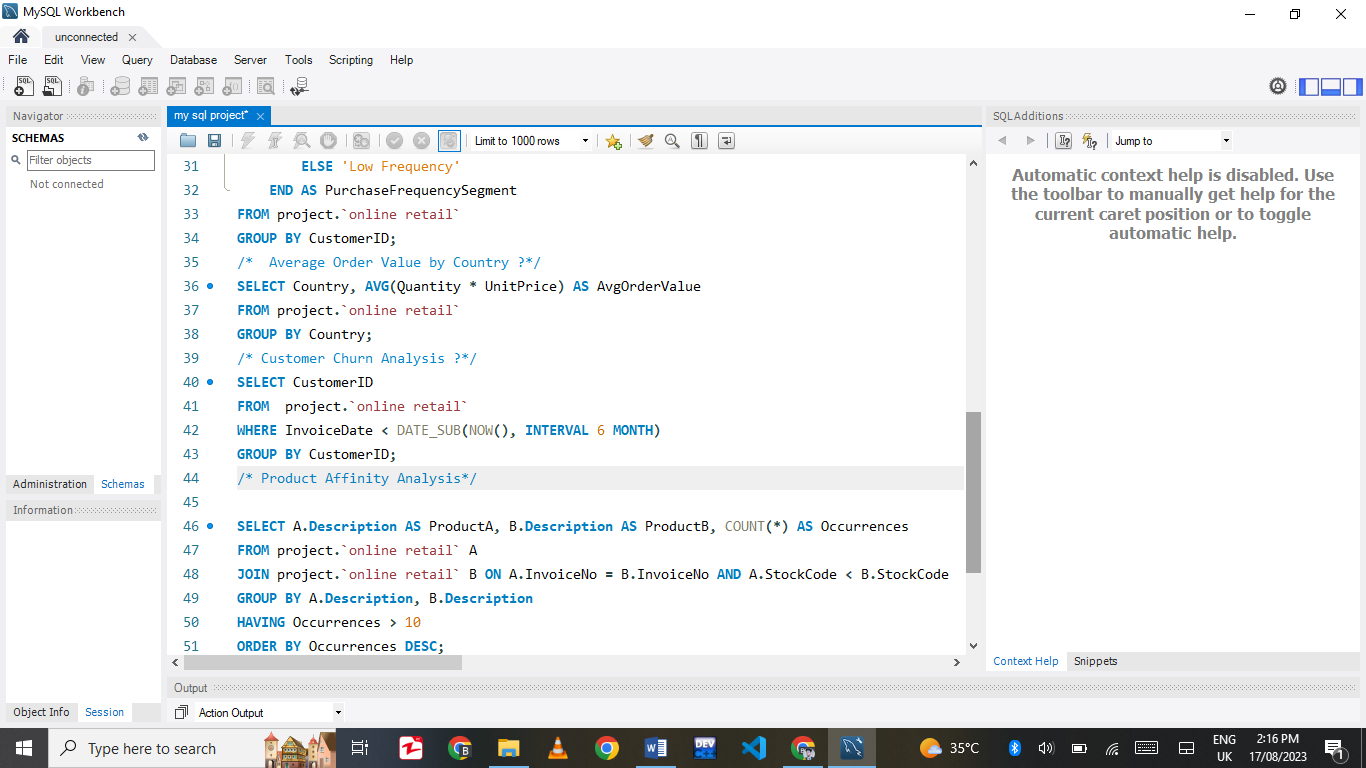
This SQL query groups customers from the "online retail" data by their shopping frequency. It assigns labels: "High Frequency" for more than 50 transactions, "Medium Frequency" for 11 to 50 transactions, and "Low Frequency" for 10 or fewer transactions. This helps understand how often customers shop.

* 1. **Average Order Value by Country**



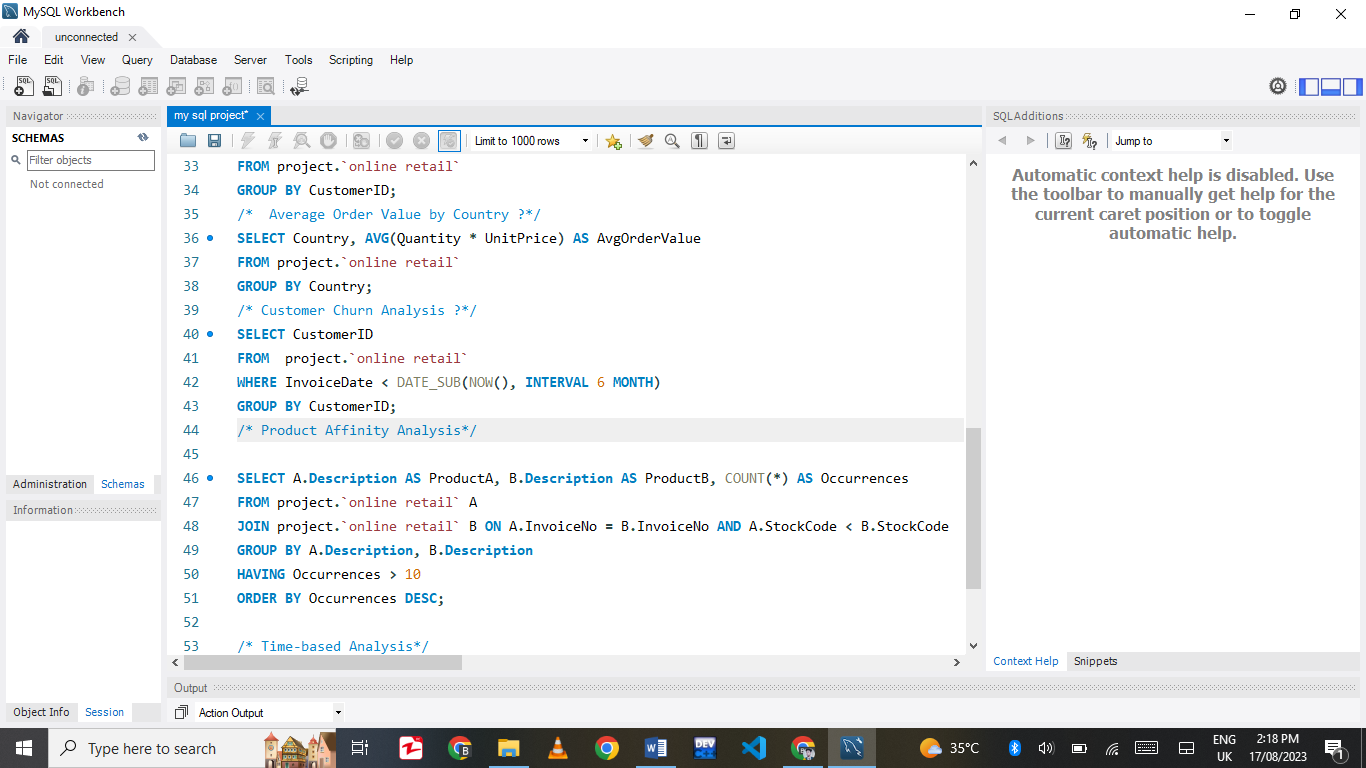
This SQL query analyzes the "online retail" data. It groups purchases by country and calculates the average order value for each. The quantity of items bought is multiplied by their price to find the order value. The results show the average spending in each country. This helps identify valuable markets based on customers' average spending habits.

* 1. **Customer Churn Analysis**



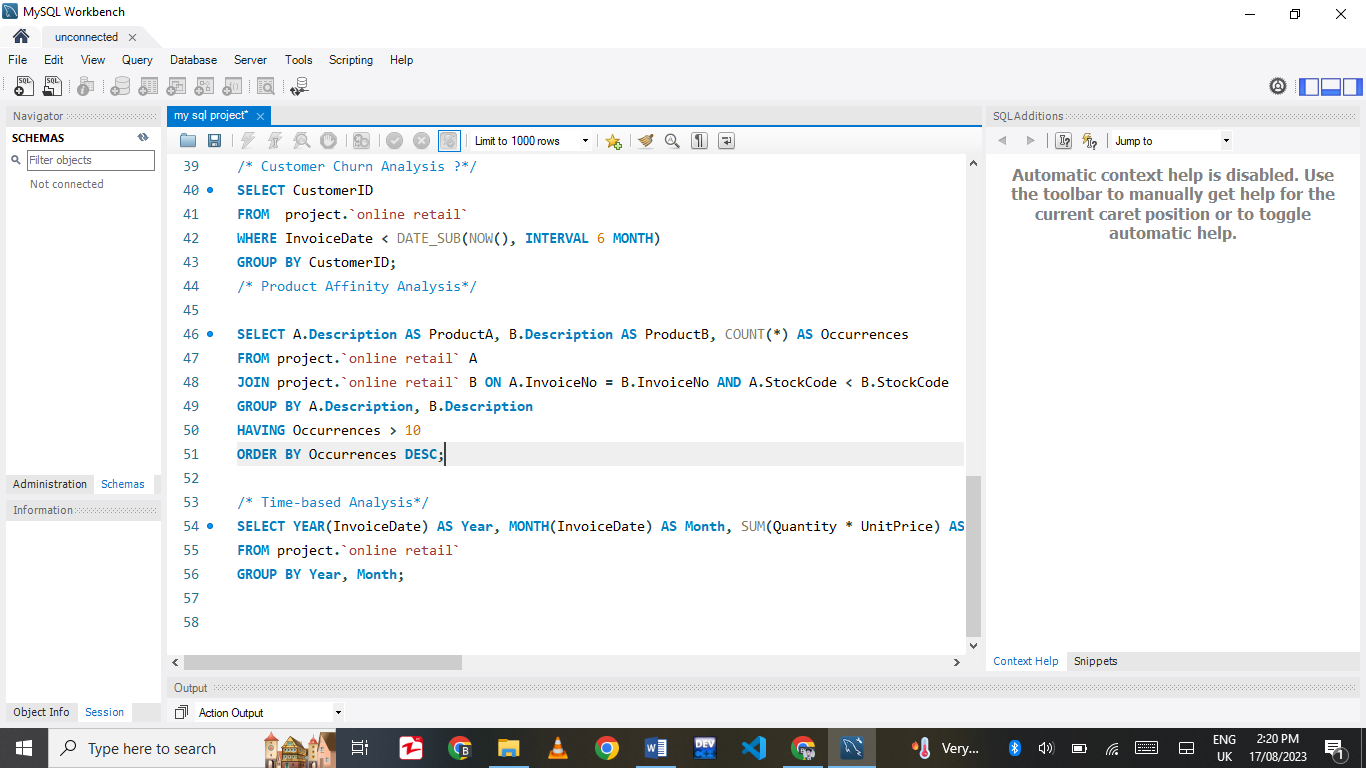
This SQL query performs a customer churn analysis on the "online retail" data. It selects CustomerIDs from those who haven't made a purchase in the last 6 months. The `WHERE` clause filters transactions with `InvoiceDate` older than 6 months from the current date. The query groups the results by CustomerID, helping identify customers who might be disengaging from the business.

* 1. **Product Affinity Analysis**



This SQL query analyzes product affinity in the "online retail" data. It pairs products often bought together. It matches products using InvoiceNo and StockCode, then groups them by ProductA and ProductB descriptions. The `HAVING` clause filters pairs occurring more than 10 times. Results are sorted by occurrences, revealing popular product pairs. This analysis helps understand product relationships and can guide cross-selling efforts.

* 1. **Time-based Analysis**



This SQL query conducts time-based analysis on the "online retail" data. It groups transactions by Year and Month, calculating the total sales by multiplying Quantity and UnitPrice for each item. Results show the TotalSales for each month in different years, allowing the observation of sales trends over time. This analysis provides insights into seasonal patterns and helps with planning for future business strategies.