**Project Report:** Online Retail Segmentation

**Project Title:** Online Retail Segmentation

**Learning Objectives:**

1. Understanding the fundamentals of Data Mining.

2. Learning how to use SQL in data mining.

3. Learning how to implement mining concepts.

**Overview:**

Customer segmentation is a strategic approach used by organizations to categorize clients based on their demographics, shopping patterns, or other traits. By analyzing factors such as demographics, shopping behaviors, and discernible traits, businesses can tailor their offerings, communications, and marketing efforts to enhance customer satisfaction, engagement, and overall business performance.

**Dataset:**

The success of any data analysis project hinges upon the quality and relevance of the dataset being utilized. In the context of our "Online Retail Segmentation" project, we leveraged a comprehensive and rich dataset that captures the essence of online retail transactions. This dataset serves as the foundation upon which we applied various data mining and SQL techniques to gain insights into customer behaviors, purchasing patterns, and market trends.

The dataset comprises a diverse range of attributes, each of which plays a vital role in understanding the nuances of customer interactions and preferences. Here is an overview of the key attributes present in the dataset:

1. InvoiceNo: A unique identifier for each transaction or invoice.

2. StockCode: A code representing the product in the inventory.

3. Description: A brief description of the product.

4. Quantity: The quantity of each product purchased in a transaction.

5. InvoiceDate: The date and time of the transaction.

6. UnitPrice: The price per unit of the product.

7. CustomerID: An identifier for each customer.

8. Country: The country where the transaction was conducted.Project Execution:

To accomplish the objectives of the project, we performed various SQL queries on the provided dataset. The queries were aimed at understanding customer behaviors and patterns within the online retail domain.

USE onlineretail\_info;

**-- Defined meta data in mysql workbench**.

CREATE TABLE Retail(

InvoiceNo INTEGER NOT NULL,

StockCode VARCHAR(16) NOT NULL,

Description VARCHAR(35),

Quantity INTEGER,

InvoiceDate VARCHAR(35),

UnitPrice NUMERIC(4,2),

CustomerID NUMERIC,

Country VARCHAR(14)

);

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

**STEPS:**

- We calculated the total order value for each customer.

- Excluded records with missing customer IDs.

- Grouped the results by CustomerID and ordered them by total order value.

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

FROM Retail

WHERE CustomerID IS NOT NULL

GROUP BY CustomerID

ORDER BY TotalOrderValue DESC;

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

**Queries**

**STEPS**

- We calculated the count of unique products purchased by each customer.

- Excluded records with missing customer IDs.

- Grouped the results by CustomerID and ordered them by the count of unique products.

**Query**

SELECT

CustomerID, COUNT(DISTINCT StockCode) AS UniqueProductsPurchased

FROM Retail

WHERE CustomerID IS NOT NULL

GROUP BY CustomerID

ORDER BY UniqueProductsPurchased DESC;

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

**STEPS**

- We identified customers who made exactly two purchases.

- Grouped the results by CustomerID and filtered results with a purchase count of 2.

**Query:**

SELECT CustomerID, COUNT(InvoiceNo) AS PurchaseCount

FROM Retail

GROUP BY CustomerID

HAVING PurchaseCount = 2;

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

**STEPS:**

- We determined which products are frequently purchased together by customers.

- Joined the Retail table with itself based on InvoiceNo and StockCode to find pairs of products.

- Grouped the results by product pairs and ordered them by purchase count.

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

FROM Retail A

JOIN Retail B ON A.InvoiceNo = B.InvoiceNo AND A.StockCode < B.StockCode

GROUP BY A.StockCode, B.StockCode

ORDER BY PurchaseCount DESC;

**Advanced Queries:**

1. **Group customers into segments based on their purchase frequency, such as high, medium, and low frequency customers. This can help you identify your most loyal customers and those who need more attention.**

**STEPS**

- We segmented customers into high, medium, and low purchase frequency categories.

- Used a CASE statement to categorize customers based on the count of invoices.

**Query**

SELECT

CustomerID,

CASE

WHEN COUNT(InvoiceNo) > 10 THEN 'High'

WHEN COUNT(InvoiceNo) > 5 THEN 'Medium'

ELSE 'Low'

END AS PurchaseFrequencySegment

FROM Retail

GROUP BY CustomerID;

1. **Calculate the average order value for each country to identify where your most valuable customers are located.**

**STEPS:**

- We calculated the average order value for each country.

- Grouped the results by Country and rounded the average order value.

**QUERY:**

SELECT

Country,

ROUND(AVG(Quantity \* UnitPrice)) AS AverageOrderValue

FROM Retail

GROUP BY Country;

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

**STEPS**

- We identified churned customers based on their last purchase date.

- Used a CASE statement to categorize customers as churned or active.

**QUERY**

SELECT

CustomerID,

MAX(InvoiceDate) AS LastPurchaseDate,

CASE

WHEN MAX(InvoiceDate) < DATE\_SUB(CURRENT\_DATE, INTERVAL 6 MONTH) THEN 'Churned'

ELSE 'Active'

END AS ChurnStatus

FROM Retail

GROUP BY CustomerID;

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

**STEPS**

Product Affinity Analysis:

- We identified product pairs that are commonly purchased together.

- Joined the Retail table with itself based on InvoiceNo and StockCode for product pairs

**QUERY:**

SELECT

r1.StockCode AS Product1,

r2.StockCode AS Product2,

COUNT(\*) AS PurchaseCount

FROM Retail r1

JOIN Retail r2 ON r1.InvoiceNo = r2.InvoiceNo AND r1.StockCode < r2.StockCode

GROUP BY r1.StockCode, r2.StockCode

ORDER BY PurchaseCount DESC;

.

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

**STEPS:**

Time-based Analysis:

- We analyzed sales trends over time by year and month.

- Grouped the results by SalesYear and SalesMonth to observe patterns.

**QUERY:**

SELECT

YEAR(InvoiceDate) AS SalesYear,

MONTH(InvoiceDate) AS SalesMonth,

SUM(Quantity \* UnitPrice) AS TotalSales

FROM Retail

GROUP BY SalesYear, SalesMonth

ORDER BY SalesYear, SalesMonth;

Conclusion:

The Online Retail Segmentation project provided valuable insights into customer behaviors and purchasing patterns. By applying SQL queries to the dataset, we were able to segment customers based on purchase frequency, analyze average order values by country, identify churned customers, explore product affinities, and examine time-based sales trends. These analyses contribute to a deeper understanding of customer preferences and can inform strategic decisions to enhance business performance.

**.**