Name : Muhammad Ali

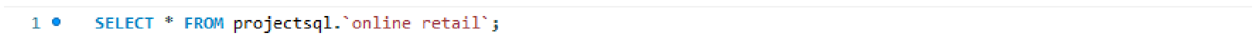
Father Name: Abdullah

CNIC NO: 71103-4462102-1

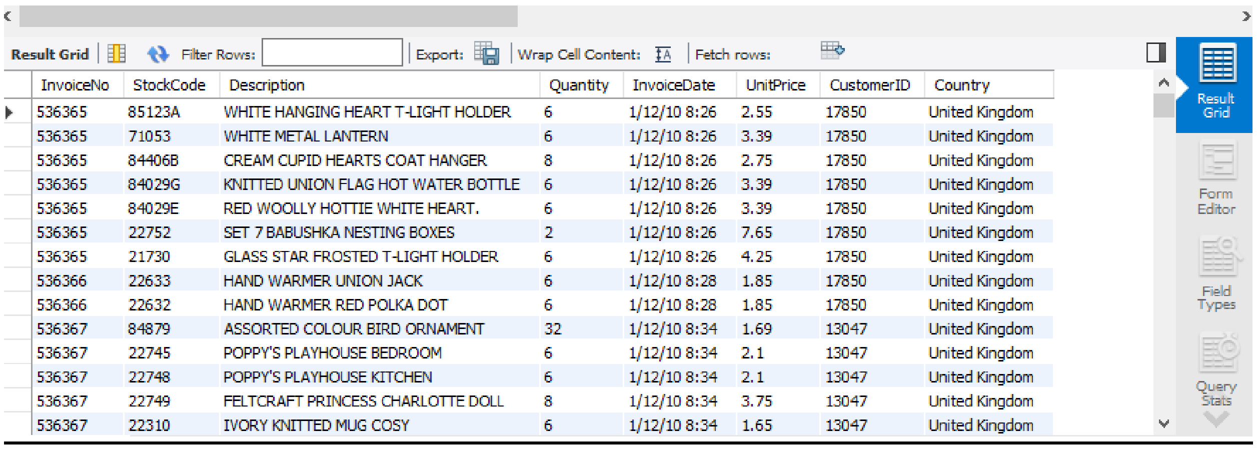
Roll No: Sk23144

Dataset: Online_retail:

Loading Data Set In Workbench



```
SELECT * FROM projectsql.`online retail`;
```



InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	1/12/10 8:26	2.55	17850	United Kingdom
536365	71053	WHITE METAL LANTERN	6	1/12/10 8:26	3.39	17850	United Kingdom
536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	1/12/10 8:26	2.75	17850	United Kingdom
536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	1/12/10 8:26	3.39	17850	United Kingdom
536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	1/12/10 8:26	3.39	17850	United Kingdom
536365	22752	SET 7 BABUSHKA NESTING BOXES	2	1/12/10 8:26	7.65	17850	United Kingdom
536365	21730	GLASS STAR FROSTED T-LIGHT HOLDER	6	1/12/10 8:26	4.25	17850	United Kingdom
536366	22633	HAND WARMER UNION JACK	6	1/12/10 8:28	1.85	17850	United Kingdom
536366	22632	HAND WARMER RED POLKA DOT	6	1/12/10 8:28	1.85	17850	United Kingdom
536367	84879	ASSORTED COLOUR BIRD ORNAMENT	32	1/12/10 8:34	1.69	13047	United Kingdom
536367	22745	POPPY'S PLAYHOUSE BEDROOM	6	1/12/10 8:34	2.1	13047	United Kingdom
536367	22748	POPPY'S PLAYHOUSE KITCHEN	6	1/12/10 8:34	2.1	13047	United Kingdom
536367	22749	FELTCRAFT PRINCESS CHARLOTTE DOLL	8	1/12/10 8:34	3.75	13047	United Kingdom
536367	22310	IVORY KNITTED MUG COSY	6	1/12/10 8:34	1.65	13047	United Kingdom

Customer Information Retrieval

The first query retrieves all columns for rows in the "online_retail" table where the CustomerID is equal to 7850. This query aims to provide a detailed view of the transactions associated with a specific customer with ID 7850.

Code:

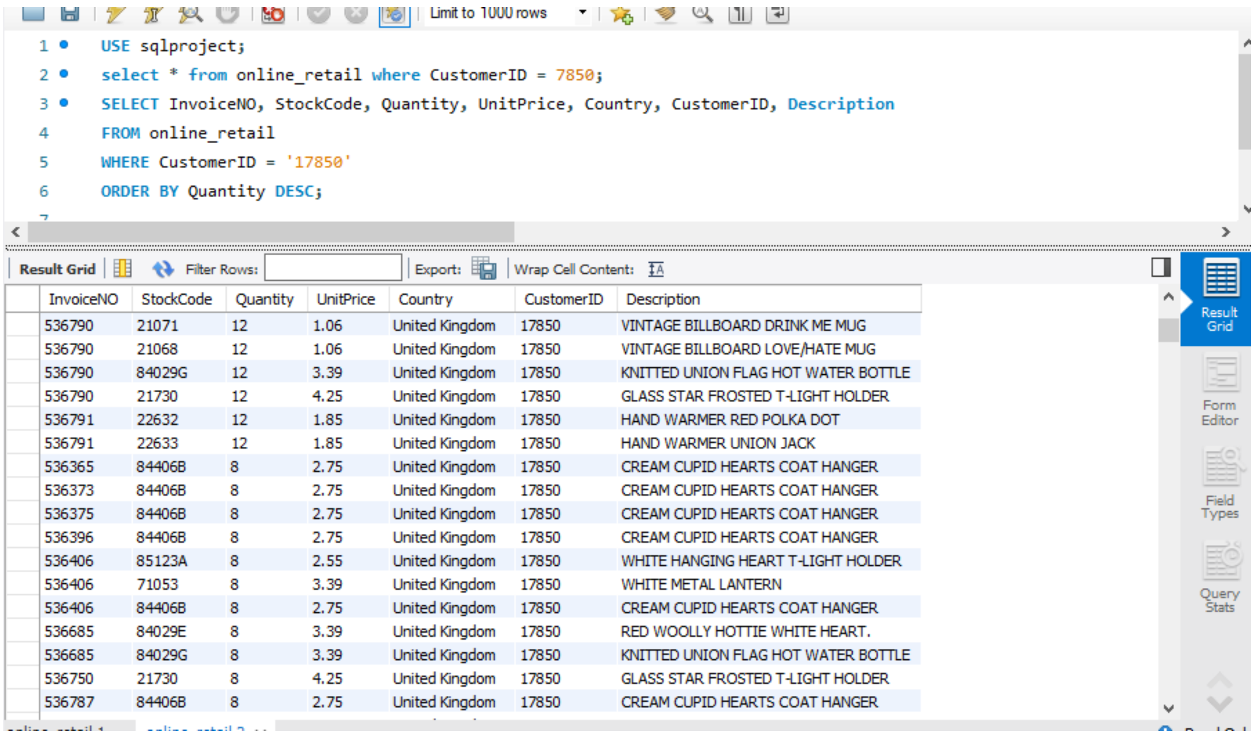
SELECT CustomerID, SUM(Quantity * UnitPrice) AS TotalOrderValue

FROM `online retail.`.`online retail.`

GROUP BY CustomerID

ORDER BY TotalOrderValue DESC;

Output:



```sql
USE sqlproject;
select * from online_retail where CustomerID = 7850;
SELECT InvoiceNO, StockCode, Quantity, UnitPrice, Country, CustomerID, Description
FROM online_retail
WHERE CustomerID = '17850'
ORDER BY Quantity DESC;
```

InvoiceNO	StockCode	Quantity	UnitPrice	Country	CustomerID	Description
536790	21071	12	1.06	United Kingdom	17850	VINTAGE BILLBOARD DRINK ME MUG
536790	21068	12	1.06	United Kingdom	17850	VINTAGE BILLBOARD LOVE/HATE MUG
536790	84029G	12	3.39	United Kingdom	17850	KNITTED UNION FLAG HOT WATER BOTTLE
536790	21730	12	4.25	United Kingdom	17850	GLASS STAR FROSTED T-LIGHT HOLDER
536791	22632	12	1.85	United Kingdom	17850	HAND WARMER RED POLKA DOT
536791	22633	12	1.85	United Kingdom	17850	HAND WARMER UNION JACK
536365	84406B	8	2.75	United Kingdom	17850	CREAM CUPID HEARTS COAT HANGER
536373	84406B	8	2.75	United Kingdom	17850	CREAM CUPID HEARTS COAT HANGER
536375	84406B	8	2.75	United Kingdom	17850	CREAM CUPID HEARTS COAT HANGER
536396	84406B	8	2.75	United Kingdom	17850	CREAM CUPID HEARTS COAT HANGER
536406	85123A	8	2.55	United Kingdom	17850	WHITE HANGING HEART T-LIGHT HOLDER
536406	71053	8	3.39	United Kingdom	17850	WHITE METAL LANTERN
536406	84406B	8	2.75	United Kingdom	17850	CREAM CUPID HEARTS COAT HANGER
536685	84029E	8	3.39	United Kingdom	17850	RED WOOLLY HOTTIE WHITE HEART.
536685	84029G	8	3.39	United Kingdom	17850	KNITTED UNION FLAG HOT WATER BOTTLE
536750	21730	8	4.25	United Kingdom	17850	GLASS STAR FROSTED T-LIGHT HOLDER
536787	84406B	8	2.75	United Kingdom	17850	CREAM CUPID HEARTS COAT HANGER

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

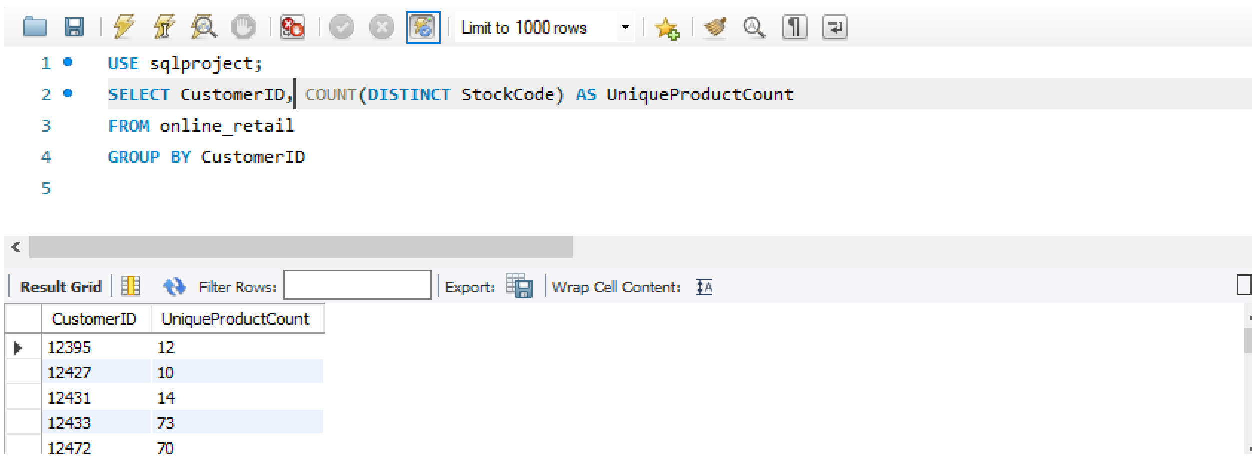
Code:

SELECT CustomerID, COUNT(DISTINCT StockCode) AS UniqueProductCount

FROM `online retail.`.`online retail.`

GROUP BY CustomerID

Output:



```
USE sqlproject;
SELECT CustomerID, COUNT(DISTINCT StockCode) AS UniqueProductCount
FROM online_retail
GROUP BY CustomerID
```

Limit to 1000 rows

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

CustomerID	UniqueProductCount
12395	12
12427	10
12431	14
12433	73
12472	70

How many unique products has each customer purchased?

Code:

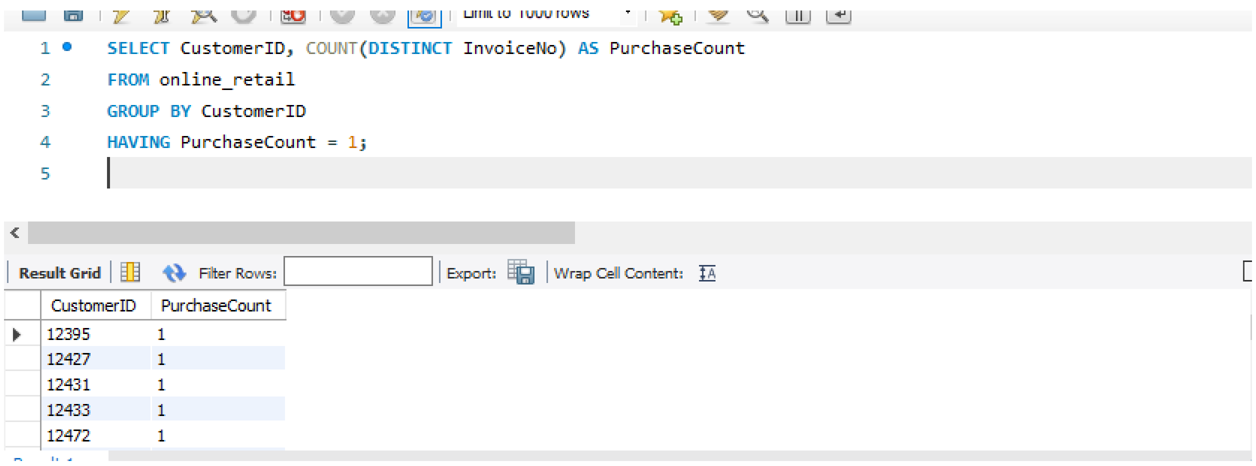
SELECT CustomerID, COUNT(DISTINCT InvoiceNo) AS PurchaseCount

FROM `online retail.`.`online retail.`

GROUP BY CustomerID

HAVING PurchaseCount = 1;

Output:



```
SELECT CustomerID, COUNT(DISTINCT InvoiceNo) AS PurchaseCount
FROM online_retail
GROUP BY CustomerID
HAVING PurchaseCount = 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

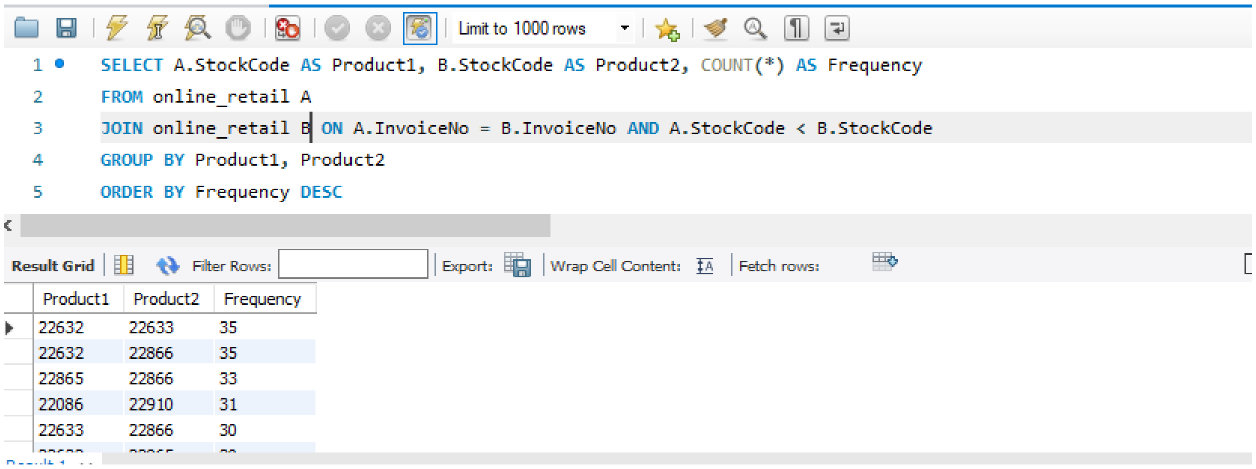
CustomerID	PurchaseCount
12395	1
12427	1
12431	1
12433	1
12472	1

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

Code:

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

FROM `online retail.`.`online retail.` A

JOIN `online retail.`.`online retail.`B ON A.InvoiceNo = B.InvoiceNo AND A.StockCode < B.StockCode

GROUP BY Product1, Product2

ORDER BY Frequency DESC

LIMIT 10;
```

Output:



```
SELECT A.StockCode AS Product1, B.StockCode AS Product2, COUNT(*) AS Frequency
FROM online_retail A
JOIN online_retail B ON A.InvoiceNo = B.InvoiceNo AND A.StockCode < B.StockCode
GROUP BY Product1, Product2
ORDER BY Frequency DESC
```

Product1	Product2	Frequency
22632	22633	35
22632	22866	35
22865	22866	33
22086	22910	31
22633	22866	30

Customer Segmentation by Purchase Frequency

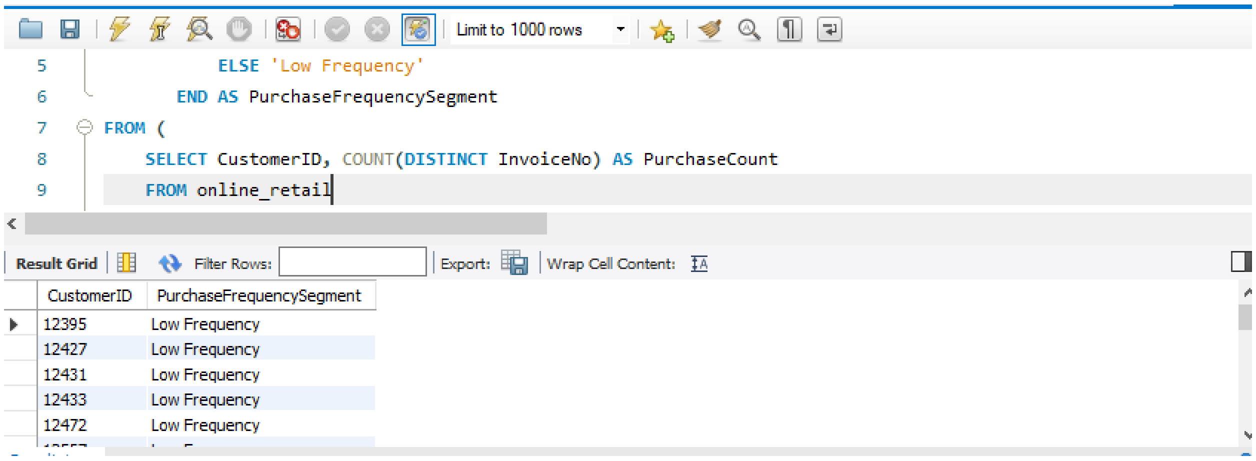
Group customers into segments based on their purchase frequency, such as high, medium, and low frequency customers. This can help you identify your most loyal customers and those who need more attention.

Code:

```
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 `online retail.`.`online retail.`
    GROUP BY CustomerID
) AS CustomerPurchaseCounts;
```

Output



```
            ELSE 'Low Frequency'
        END AS PurchaseFrequencySegment
FROM (
    SELECT CustomerID, COUNT(DISTINCT InvoiceNo) AS PurchaseCount
    FROM online_retail
```

	CustomerID	PurchaseFrequencySegment
▶	12395	Low Frequency
	12427	Low Frequency
	12431	Low Frequency
	12433	Low Frequency
	12472	Low Frequency

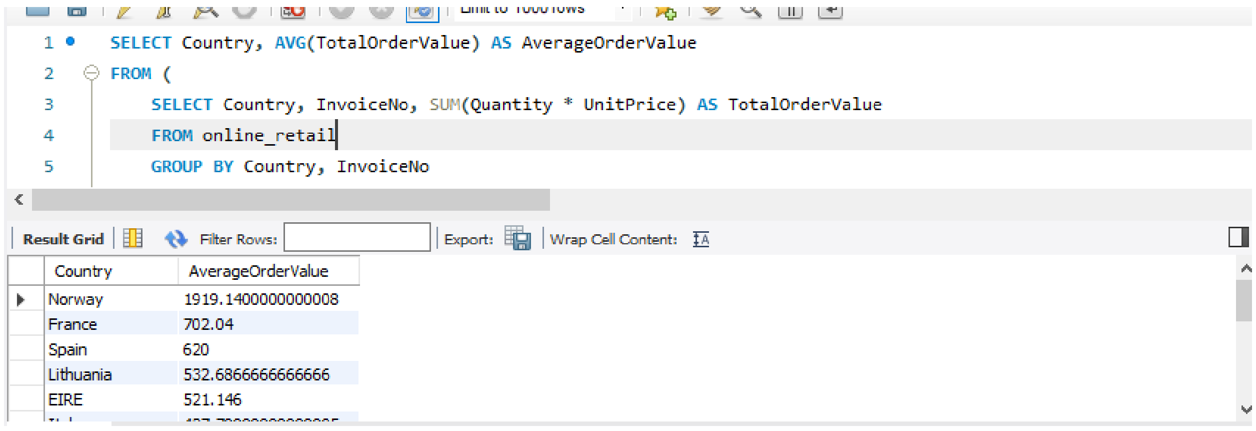
2. Average Order Value by Country

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

Code:

```
SELECT Country, AVG(TotalOrderValue) AS AverageOrderValue
FROM (
    SELECT Country, InvoiceNo, SUM(Quantity * UnitPrice) AS TotalOrderValue
    FROM `online retail.`.`online retail.`
    GROUP BY Country, InvoiceNo
) AS CountryOrderValues
GROUP BY Country
ORDER BY AverageOrderValue DESC;
```

Output:



```sql
SELECT Country, AVG(TotalOrderValue) AS AverageOrderValue
FROM (
    SELECT Country, InvoiceNo, SUM(Quantity * UnitPrice) AS TotalOrderValue
    FROM online_retail
    GROUP BY Country, InvoiceNo
```

Country	AverageOrderValue
Norway	1919.1400000000008
France	702.04
Spain	620
Lithuania	532.6866666666666
EIRE	521.146

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

Code:

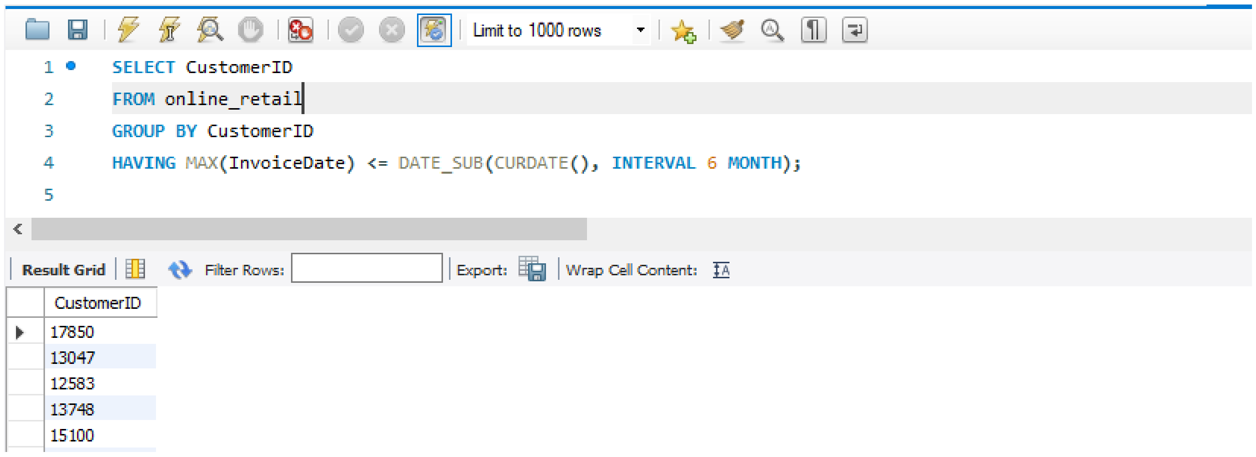
SELECT CustomerID

FROM `online retail.`.`online retail.`

GROUP BY CustomerID

HAVING MAX(InvoiceDate) <= DATE_SUB(CURDATE(), INTERVAL 6 MONTH);

Output:



```sql
SELECT CustomerID
FROM online_retail
GROUP BY CustomerID
HAVING MAX(InvoiceDate) <= DATE_SUB(CURDATE(), INTERVAL 6 MONTH);
```

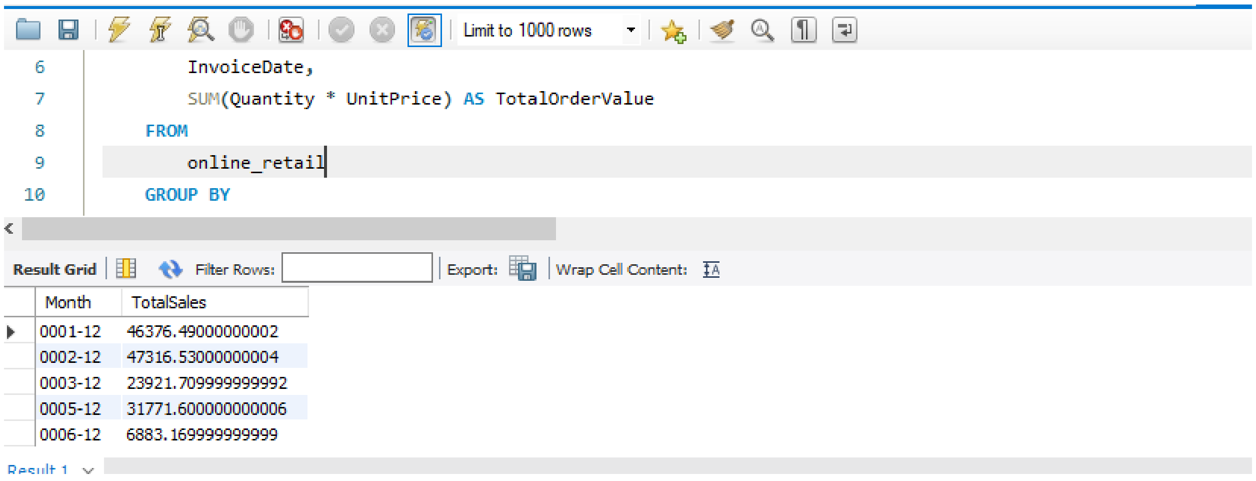
CustomerID
17850
13047
12583
13748
15100

. Time-based Analysis
Explore trends in customer behavior over time, such as monthly or quarterly sales patterns.

Code:

```
SELECT
    DATE_FORMAT(InvoiceDate, '%Y-%m') AS Month,
    SUM(TotalOrderValue) AS TotalSales
FROM (
    SELECT
        InvoiceDate,
        SUM(Quantity * UnitPrice) AS TotalOrderValue
    FROM
        `online retail.`.`online retail.`
    GROUP BY
        InvoiceNo, InvoiceDate
) AS InvoiceTotals
GROUP BY
    Month
ORDER BY
    Month;
```

Output:



```
        InvoiceDate,
        SUM(Quantity * UnitPrice) AS TotalOrderValue
    FROM
        online_retail
    GROUP BY
```

Month	TotalSales
0001-12	46376.49000000002
0002-12	47316.53000000004
0003-12	23921.709999999992
0005-12	31771.600000000006
0006-12	6883.169999999999

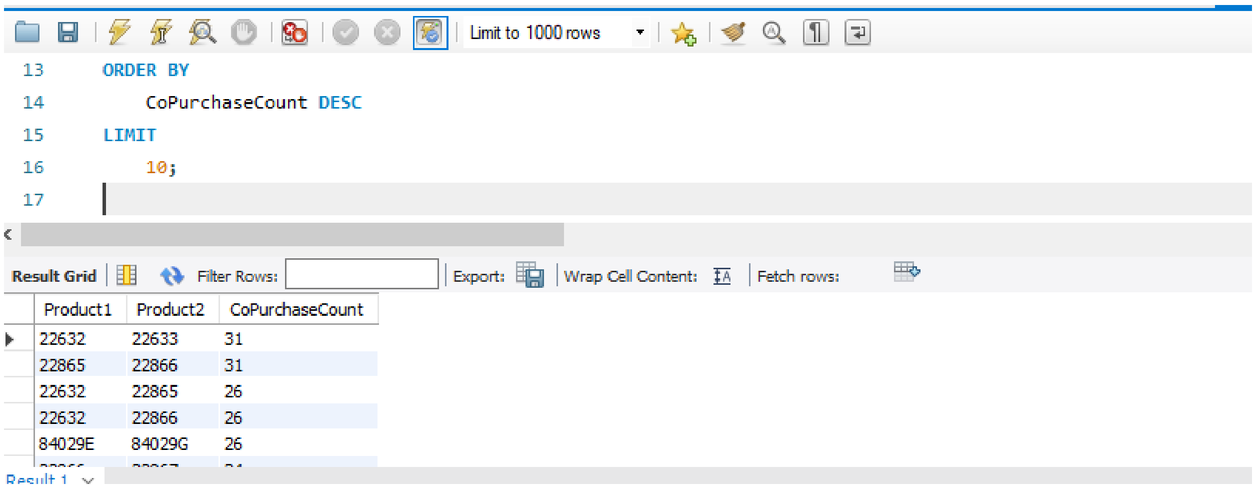
4. Product Affinity Analysis

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

Code:

```
SELECT
    p1.StockCode AS Product1,
    p2.StockCode AS Product2,
    COUNT(DISTINCT p1.InvoiceNo) AS CoPurchaseCount
FROM
    `online retail.`.`online retail.` p1
JOIN
   `online retail.`.`online retail.` 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
LIMIT
    10;
```

Output:



Limit to 1000 rows

```
ORDER BY
    CoPurchaseCount DESC
LIMIT
    10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Product1	Product2	CoPurchaseCount
22632	22633	31
22865	22866	31
22632	22865	26
22632	22866	26
84029E	84029G	26
[illegible]	[illegible]	[illegible]

Result 1

The End