



## High Impact Skills Development Program in Artificial Intelligence, Data Science, and Blockchain

**Project Title:** Online Retail Segmentation.

**Name:** Mureed Hussain Drotty

**Id:** sk23201

**Gmail:** [drottynew3492@gmail.com](mailto:drottynew3492@gmail.com)

### Final Project Data Mining

#### 1. Define meta data in mysql workbench:

Query 1 retail retail x

Limit to 1000 rows

1 • SELECT \* FROM project.retail;

Result Grid

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850	United Kingdom
536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850	United Kingdom
536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850	United Kingdom
536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850	United Kingdom
536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850	United Kingdom
536365	22752	SET 7 BABUSHKA NESTING BOXES	2	12/1/2010 8:26	7.65	17850	United Kingdom
536365	21730	GLASS STAR FROSTED T-LIGHT HOLDER	6	12/1/2010 8:26	4.25	17850	United Kingdom
536366	22633	HAND WARMER UNION JACK	6	12/1/2010 8:28	1.85	17850	United Kingdom
536366	22632	HAND WARMER RED POLKA DOT	6	12/1/2010 8:28	1.85	17850	United Kingdom
536367	84879	ASSORTED COLOUR BIRD ORNAMENT	32	12/1/2010 8:34	1.69	13047	United Kingdom
536367	22745	POPPY'S PLAYHOUSE BEDROOM	6	12/1/2010 8:34	2.1	13047	United Kingdom
536367	22748	POPPY'S PLAYHOUSE KITCHEN	6	12/1/2010 8:34	2.1	13047	United Kingdom
536367	22749	FELTCRAFT PRINCESS CHARLOTTE DOLL	8	12/1/2010 8:34	3.75	13047	United Kingdom
536367	22310	IVORY KNITTED MUG COSY	6	12/1/2010 8:34	1.65	13047	United Kingdom
536367	84969	BOX OF 6 ASSORTED COLOUR TEASPOONS	6	12/1/2010 8:34	4.25	13047	United Kingdom
536367	22623	BOX OF VINTAGE JIGSAW BLOCKS	3	12/1/2010 8:34	4.95	13047	United Kingdom

Output

Action Output

# Time Action Message

1 08:22:03 SELECT \* FROM project.retail LIMIT 0, 1000 1000 row(s) returned

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

Limit to 1000 rows

```

1  SELECT
2      InvoiceNo,
3      InvoiceDate,
4      CustomerID,
5      Country,
6      SUM(Quantity * UnitPrice) AS OrderValue
7  FROM
8      project.retail
9  GROUP BY
10     InvoiceNo, InvoiceDate, CustomerID, Country
11 ORDER BY
12     OrderValue DESC;

```

InvoiceNo	InvoiceDate	CustomerID	Country	OrderValue
541431	1/18/2011 10:01	12346	United Kingdom	77183.6
541220	1/14/2011 14:11	14156	EIRE	16774.719999999998
537659	12/7/2010 16:43	18102	United Kingdom	15885.489999999998
540815	1/11/2011 12:55	15749	United Kingdom	15160.900000000001
540689	1/11/2011 8:43	17450	United Kingdom	12797.52
541206	1/14/2011 12:24	14646	Netherlands	10389.060000000003
537657	12/7/2010 16:42	18102	United Kingdom	9639.119999999999
539731	12/21/2010 15:05	14646	Netherlands	8520.919999999998

Result 2 x

- How many unique products has each customer purchased?

Limit to 1000 rows

```

1  SELECT
2      CustomerID,
3      COUNT(DISTINCT StockCode) AS UniqueProductsPurchased
4  FROM
5      project.retail
6  GROUP BY
7      CustomerID
8  ORDER BY
9      UniqueProductsPurchased asc;

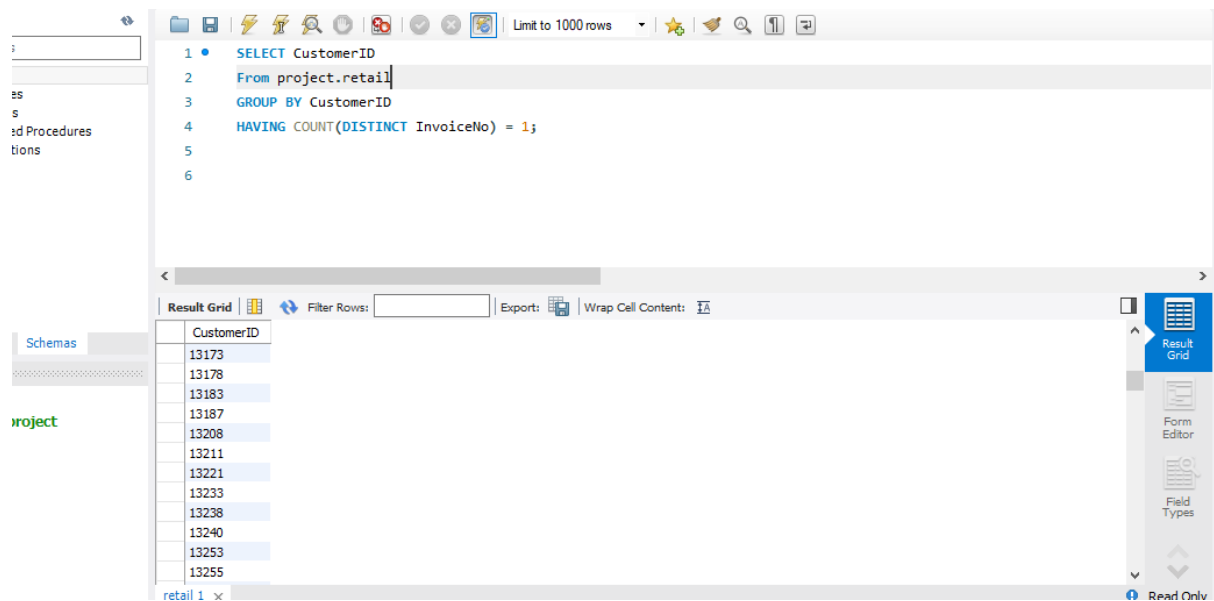
```

CustomerID	UniqueProductsPurchased
16949	5
17025	5
17430	5
17446	5
17940	5
18032	5
18050	5
18062	5
12464	6
12939	6
13027	6
13255	6

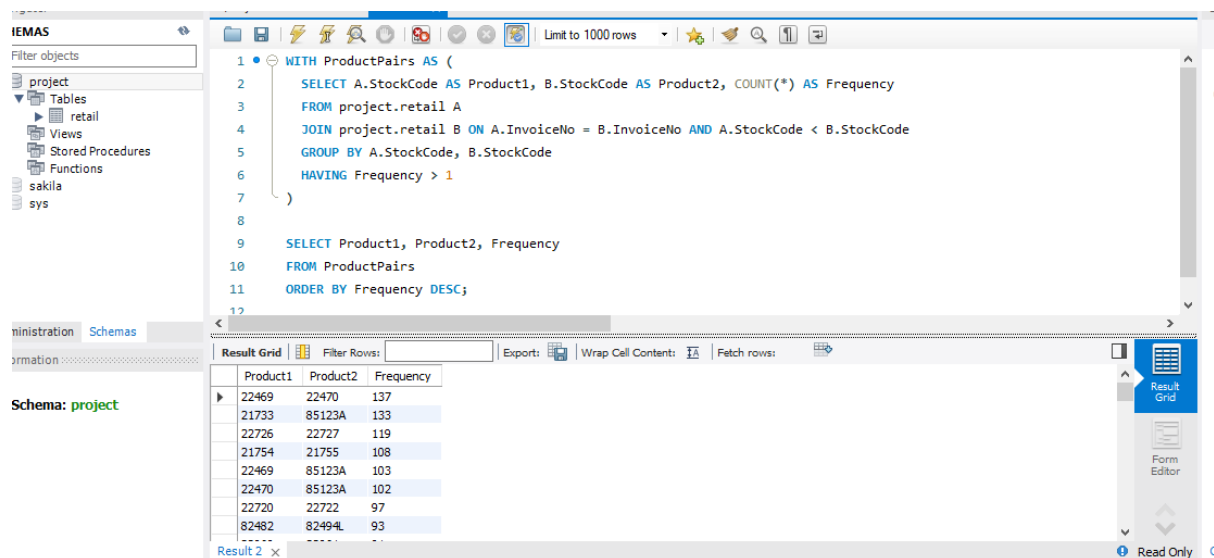
Result 3 x

Output

- Which customers have only made a single purchase from the company?



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



## Advance Queries

### 1. Customer Segmentation by Purchase Frequency:

SQL Query Editor showing a query to calculate Customer Purchase Frequency and its corresponding frequency segment.

```

1 WITH CustomerPurchaseFrequency AS (
2     SELECT
3         CustomerID,
4         COUNT(DISTINCT InvoiceNo) AS PurchaseFrequency
5     FROM
6         project.retail
7     GROUP BY
8         CustomerID
9 )
10
11 SELECT
12     CustomerID,
13     CASE
14         WHEN PurchaseFrequency >= 137 THEN 'High Frequency'
15         WHEN PurchaseFrequency >= 93 THEN 'Medium Frequency'
16         ELSE 'Low Frequency'
17     END AS PurchaseFrequencySegment
18 FROM
19     CustomerPurchaseFrequency
20 
```

Result Grid:

CustomerID	PurchaseFrequencySegment
13629	Low Frequency
13634	Low Frequency
13649	Low Frequency
13652	Low Frequency
13656	Low Frequency

## 2. Average Order Value by Country

SQL Query Editor showing a query to calculate Average Order Value by Country.

```

1 SELECT
2     Country,
3     AVG(TotalOrderValue) AS AverageOrderValue
4 FROM (
5     SELECT
6         Country,
7         InvoiceNo,
8         SUM(UnitPrice * Quantity) AS TotalOrderValue
9     FROM
10         project.retail -- Replace with your actual table name
11     GROUP BY
12         Country, InvoiceNo
13 ) AS OrderValues
14 GROUP BY
15     Country
16 ORDER BY
17     AverageOrderValue DESC;
18 
```

```

1 • SELECT
2     Country,
3     AVG(TotalOrderValue) AS AverageOrderValue
4 FROM (
5     SELECT

```

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

Country	AverageOrderValue
Netherlands	3691.6369999999997
Japan	2697.176
Greece	2661.24
Singapore	2053.07
Lebanon	1693.8800000000003
Sweden	1566.024
Norway	1373.8400000000001
EIRE	1301.34
Denmark	1281.5000000000002
Cyprus	1033.0700000000002
Spain	906.5440000000001
Switzerland	889.9257142857142
Australia	687.3673333333332
Portugal	649.568
Italy	642.8019999999999
Finland	593.94
Iceland	593.5899999999999
France	539.0147368421054
Channel Isl...	519.5550000000001
Germany	503.9799999999999

### 3. Customer Churn Analysis

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

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The SQL editor contains the following query:

```
1 • SELECT
2     CustomerID,
3     MAX(InvoiceDate) AS LastPurchaseDate
4 FROM
5     project.retail
6 GROUP BY
7     CustomerID
8 HAVING
9     LastPurchaseDate <= DATE_SUB(NOW(), INTERVAL 6 MONTH);
```

Below the editor is the 'Result Grid' tab, which displays the query results in a table. The table has two columns: 'CustomerID' and 'LastPurchaseDate'. The results are as follows:

CustomerID	LastPurchaseDate
17850	12/2/2010 9:44
13047	2/3/2011 13:06
12583	2/10/2011 15:38
13748	12/1/2010 9:00
15100	12/8/2010 12:09
15291	12/17/2010 10:32
14688	2/8/2011 11:27
17809	2/11/2011 9:11
15311	2/8/2011 14:06
16098	12/1/2010 9:45
18074	12/1/2010 9:53
17420	12/1/2010 9:56
16029	2/4/2011 14:25
16250	12/1/2010 9:59
12431	12/17/2010 14:10

The interface also includes a 'Filter Rows' input field, an 'Export' button, a 'Wrap Cell Content' checkbox, and a 'Fetch rows' button. The bottom status bar shows 'Output'.

#### 4. Product Affinity Analysis

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

```

WITH ProductPairs AS (
    SELECT
        A.StockCode AS Product1,
        B.StockCode AS Product2,
        COUNT(*) AS Frequency
    FROM
        project.retail AS A
        JOIN project.retail AS B ON A.InvoiceNo = B.InvoiceNo AND A.StockCode < B.StockCode
    GROUP BY
        A.StockCode, B.StockCode
),

```

```

ProductPurchaseCounts AS (
    SELECT
        StockCode,
        COUNT(*) AS TotalPurchases
    FROM
        project.retail
    GROUP BY
        StockCode
)

```

```

ProductPurchaseCounts AS (
    SELECT
        StockCode,
        COUNT(*) AS TotalPurchases
    FROM
        project.retail
    GROUP BY
        StockCode
)

SELECT
    PP.Product1,
    PP.Product2,
    PP.Frequency,
    PP.Frequency / (PP1.TotalPurchases * PP2.TotalPurchases) AS Correlation
FROM
    ProductPairs AS PP
    JOIN ProductPurchaseCounts AS PP1 ON PP.Product1 = PP1.StockCode
    JOIN ProductPurchaseCounts AS PP2 ON PP.Product2 = PP2.StockCode
ORDER BY
    Correlation DESC;

```

## 5. Time-based Analysis

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

1	•	SELECT DATE_FORMAT(InvoiceDate, '%Y-%m')
2		AS YearMonth, COUNT(DISTINCT CustomerID)
3		AS UniqueCustomers, COUNT(InvoiceNo)
4		AS TotalTransactions, SUM(UnitPrice * Quantity) AS TotalSales
5		FROM project.retail
6		GROUP BY YearMonth
7		ORDER BY YearMonth;
8		

Result Grid		Filter Rows: <input type="text"/>	Export:	Wrap Cell Content:
	YearMonth	UniqueCustomers	TotalTransactions	TotalSales
	NULL	1447	54199	1295384.439999882

Best of luck