

Distribution of Order Values:

Count Unique Product of Each Customer:

Limit to 1000 rows

1

-- To count the number of unique products each customer has purchased:

2

Select CustomerId, count(DISTINCT StockCode) as uniqueproduct

3

from onlineretail

4

group by CustomerID;

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

CustomerId	uniqueproduct
12431	14
12433	73
12557	5
12583	20
12600	2
12662	15
12682	16
12686	7
12717	6

Result 1

Output

Action Output

#	Time	Action	Message	Duration / Fetch
43	16:57:13	SELECT YEAR(str_to_date(InvoiceDate, '%m/%d/%y')) AS Year, MONTH(str_to_date(In...	4 row(s) returned	0.094 sec / 0.000 sec
44	17:02:28	SELECT InvoiceNo FROM 'project1':onlineretail LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec

Customer with Single Purchase:

Limit to 1000 rows

1

-- To find customers who have made only a single purchase:

2

select customerid

3

from onlineretail

4

group by CustomerID

5

having count(DISTINCT InvoiceNo)=1;

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

Result Grid

Form Editor

Read Only

customerid
12395
12427
12431
12433
12557
12583
12600
12662

onlineretail 1

Output

Action Output

#	Time	Action	Message	Duration / Fetch
43	16:57:13	SELECT YEAR(str_to_date(InvoiceDate,"%m/%d/%y")) AS Year, MONTH(str_to_date(In...	4 row(s) returned	0.094 sec / 0.000 sec
44	17:02:28	SELECT InvoiceNo FROM project1: onlineretail LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec

Commonly Purchased Products:

The screenshot shows a SQL query editor with the following query:

```
1  -- To find which products are most commonly purchased together:
2  • SELECT a.StockCode AS Product1, b.StockCode AS Product2, COUNT(*) AS CustomerID
3  FROM onlineretail a
4  JOIN onlineretail b ON a.InvoiceNo = b.InvoiceNo AND a.StockCode < b.StockCode
5  GROUP BY Product1, Product2
6  ORDER BY CustomerID DESC;
7
```

Below the query editor is a results grid showing the output of the query. The grid has three columns: Product1, Product2, and CustomerID. The results are as follows:

Product1	Product2	CustomerID
22632	22633	26
84029E	84029G	22
84029E	85123A	20
84029G	85123A	20
21730	85123A	18
71053	85123A	18
22865	22866	18
22752	85123A	17
71730	71053	17

Below the results grid is an output window showing the execution of the query. The output is as follows:

#	Time	Action	Message	Duration / Fetch
43	16:57:13	SELECT YEAR(str_to_date(InvoiceDate, '%m/%d/%y')) AS Year, MONTH(str_to_date(In...	4 row(s) returned	0.094 sec / 0.000 sec
44	17:02:28	SELECT InvoiceNo FROM 'project1' onlineretail LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec

Purchased Frequency Segmentation:

Limit to 1000 rows

1

-- Purchase Frequency Segmentation:

2

• SELECT CustomerID,

3

CASE

4

WHEN COUNT(DISTINCT InvoiceNo) > 4 THEN 'High Frequency'

5

WHEN COUNT(DISTINCT InvoiceNo) > 2 and COUNT(DISTINCT InvoiceNo) <=4 THEN 'Medium Frequency'

6

ELSE 'Low Frequency'

7

END AS PurchaseFrequencySegment

8

FROM onlinetail

9

GROUP BY CustomerID;

10

Result Grid

Filter Rows:

Export:

Wrap Cell Contents

Result Grid

Form Editor

Read Only

CustomerID	PurchaseFrequencySegment
14898	Low Frequency
14901	Low Frequency
14911	Medium Frequency
15012	Low Frequency
15061	High Frequency
15070	Low Frequency
15093	Low Frequency
15100	Low Frequency
15107	Low Frequency

Result 5

Output

Action Output

#	Time	Action	Message	Duration / Fetch
43	16:57:13	SELECT YEAR(atr_to_date(InvoiceDate,"mm/dd/yyyy")) AS Year, MONTH(atr_to_date(n...	4 row(s) returned	0.094 sec / 0.000 sec
44	17:02:28	SELECT InvoiceNo FROM 'project1'.onlinetail LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec

5

Average Order Value By Country:

The screenshot shows a SQL query editor with the following query:

```
1  -- Average Order Value by Country:
2  • SELECT Country,
3      AVG(Quantity * UnitPrice) AS AvgOrderValue
4  FROM onlineretall
5  GROUP BY Country;
6
```

Below the query editor, the 'Result Grid' displays the following data:

Country	AvgOrderValue
United Kingdom	22.919383305647568
France	29.796279069767447
Australia	25.589285714285715
Netherlands	96.30000000000001
Germany	20.68851063829787
Norway	26.2895890410959
EIRE	27.143020833333335
Switzerland	50.566666666666666
Spain	174

The 'Output' section shows the execution log:

#	Time	Action	Message	Duration / Fetch
43	16:57:13	SELECT YEAR(str_to_date(InvoiceDate, '%m/%d/%y')) AS Year, MONTH(str_to_date(In...	4 row(s) returned	0.094 sec / 0.000 sec
44	17:02:28	SELECT InvoiceNo FROM 'project1'.onlineretall LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec

Churn Analysis:

The screenshot shows the SQL IDE interface. The top toolbar includes icons for file operations, search, and execution. The query editor contains the following SQL code:

```
-- Customer Churn Analysis(Last 6 months):
SELECT CustomerID
FROM onlineretail
WHERE InvoiceDate < DATE_SUB(NOW(), INTERVAL 6 MONTH);
```

Below the query editor, the 'Result Grid' is displayed, showing a table with one column, 'CustomerID'. The table contains 10 rows of data, with the first row highlighted. The 'Output' pane at the bottom shows the execution log with two entries:

#	Time	Action	Message	Duration / Fetch
43	16:57:13	SELECT YEAR(str_to_date(InvoiceDate,'%m/%d/%y')) AS Year, MONTH(str_to_date(In...	4 row(s) returned	0.094 sec / 0.000 sec
44	17:02:28	SELECT InvoiceNo FROM 'project1'.onlineretail' LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec

Product Affinity Analysis:

Limit to 1000 rows

1

-- Product Affinity Analysis:

2

• SELECT a.StockCode AS Product1, b.StockCode AS Product2, COUNT(*) AS Quantity

3

FROM onlineretail a

4

JOIN onlineretail b ON a.InvoiceNo = b.InvoiceNo AND a.StockCode < b.StockCode

5

GROUP BY Product1, Product2

6

ORDER BY Quantity DESC;

7

Result Grid

Filter Rows:

Export:

Wrap Cell Contents

Fetch rows:

	Product1	Product2	Quantity
▶	22632	22633	26
	84029E	84029G	22
	84029E	85123A	20
	84029G	85123A	20
	21730	85123A	18
	71053	85123A	18
	22865	22866	18
	22752	85123A	17
	71790	71053	17

Result 1

Read Only

Form Editor

Output

Action Output

#	Time	Action	Message	Duration / Fetch
43	16:57:13	SELECT YEAR(str_to_date(InvoiceDate, '%m/%d/%y')) AS Year, MONTH(str_to_date(In...	4 row(s) returned	0.094 sec / 0.000 sec
44	17:02:28	SELECT InvoiceNo FROM 'project1':onlineretail' LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec

Timed Based Analysis(Monthly Sales):

Limit to 1000 rows

1

-- Time-based Analysis (Monthly Sales):

2

• SELECT YEAR(str_to_date(InvoiceDate,'%m/%d/%y')) AS Year,

3

MONTH(str_to_date(InvoiceDate,'%m/%d/%y')) AS Month,

4

SUM(Quantity * UnitPrice) AS TotalSales,

5

COUNT(DISTINCT CustomerID) AS UniqueCustomers

6

FROM onlineretail

7

GROUP BY Year, Month

8

ORDER BY Year, Month;

9

Result Grid

Filter Rows:

Export:

Wrap Cell Contents

Result Grid

Year	Month	TotalSales	UniqueCustomers
2020	1	46376.490000000003	95
2020	2	47316.529999999987	99
2020	3	23921.7100000000097	50
2020	5	2171.6000000000017	6

Result 2

Output

Action Output

#	Time	Action	Message	Duration / Fetch
43	16:57:13	SELECT YEAR(str_to_date(InvoiceDate,'%m/%d/%y')) AS Year, MONTH(str_to_date(In...	4 row(s) returned	0.094 sec / 0.000 sec
44	17:02:28	SELECT InvoiceNo FROM 'project1' onlineretail LIMIT 0, 1000	1000 row(s) returned	0.000 sec / 0.000 sec

THE END