EXPLORATORY DATA ANALYSIS AND CUSTOMER SEGMENTATION USING RFM ANALYSIS OF RETAIL STORES

Problem statement:

A chain of retail stores wants to launch a marketing campaign. So to target the customers as a group, the stores want to segment the customers into different categories using RFM analysis. Also, the stores want to make actionable insights out the data.

Action plan:

1. Exploratory data analysis
2. Customer segmentation using RFM analysis

Exploratory data analysis

Schema:

Field name	Type	Mode
InvoiceNo	INTEGER	NULLABLE
StockCode	STRING	NULLABLE
Description	STRING	NULLABLE
Quantity	INTEGER	NULLABLE
InvoiceDate	TIMESTAMP	NULLABLE
UnitPrice	FLOAT	NULLABLE
CustomerID	FLOAT	NULLABLE
Country	STRING	NULLABLE

Query 1:

```
SELECT
  *
FROM
  `CSM.sales`
LIMIT 5
```

Output 1:

Row	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
1	571035	21238	RED RETROSPOT CUP	8	2011-10-13 12:50:00 UTC	0.85	12446.0	RSA
2	571035	21243	PINK POLKADOT PLATE	8	2011-10-13 12:50:00 UTC	1.69	12446.0	RSA
3	571035	23240	SET OF 4 KNICK KNACK TINS DOILY	6	2011-10-13 12:50:00 UTC	4.15	12446.0	RSA
4	571035	23209	LUNCH BAG VINTAGE DOILY	10	2011-10-13 12:50:00 UTC	1.65	12446.0	RSA
5	571035	23201	JUMBO BAG ALPHABET	10	2011-10-13 12:50:00 UTC	2.08	12446.0	RSA

Attribute Information :

- InvoiceNo: Invoice number. A 6-digit integral number uniquely assigned to each transaction. If this code starts with letter 'c', it indicates a cancellation.
- StockCode: Product (item) code. A 5-digit integral number uniquely assigned to each distinct product.
- Description: Product (item) name.
- Quantity: The quantities of each product (item) per transaction. Numeric.
- InvoiceDate: Invoice Date and time. The date and time when each transaction was generated.
- UnitPrice: Unit price. Product price per unit in sterling.
- CustomerID: Customer number. A 5-digit integral number uniquely assigned to each customer.
- Country: Country name. The name of the country where each customer resides.

Query 2:

```
SELECT
  COUNT(DISTINCT StockCode)  no_of_distinct_products
FROM
  `CSM.sales`
```

Output 2:

Row	no_of_distinct_products
1	3479

There are 3479 different products that are sold.

Query 3:

```
SELECT
  DATE(MIN(InvoiceDate)) AS first_date,
  DATE(MAX(InvoiceDate)) AS last_date
FROM
  `CSM.sales`
```

Output 3:

Row	first_date	last_date
1	2010-12-01	2011-12-09

The date is collected between 1st December 2010 and 9th December 2011 which is almost equal to an year.

Query 3:

```
SELECT
  COUNT(DISTINCT InvoiceNo) AS total_no_of_transactions
FROM
  `CSM.sales`
```

Output 3:

Row	total_no_of_transacti
1	18224

There are total 18224 distinct Invoice numbers in the dataset.

Query 4:

```
SELECT
  CustomerID,
  COUNT(DISTINCT InvoiceNo) AS no_of_transactions
FROM
  `CSM.sales`
GROUP BY
  CustomerID
ORDER BY
  no_of_transactions DESC
LIMIT 5
```

Output 4:

Row	CustomerID	no_of_transactions
1	12748.0	204
2	14911.0	197
3	17841.0	124
4	13089.0	91
5	15311.0	91

Customers who involved in highest number of transactions during this time period are as shown above.

Query 5:

```
SELECT
  COUNT(DISTINCT CustomerID) AS no_of_customers,
  COUNT(DISTINCT Country) AS no_of_countries,

FROM
  `CSM.sales`
```

Output 5:

Row	no_of_customers	no_of_countries
1	4311	37

There are 4311 customers from 37 different countries in this dataset.

Objective: To find 5 top selling products in the retail stores.

Query 6:

```
SELECT
  Description,
  SUM(Quantity) AS total_quantity_sold,
  ROUND(SUM(Quantity*UnitPrice)) AS total_revenue
FROM
  `CSM.sales`
GROUP BY
  Description
ORDER BY
  total_quantity_sold DESC
LIMIT 5
```

Output 6:

Row	Description	total_quantity_sold	total_revenue
1	PAPER CRAFT , LITTLE BIRDIE	80995	168470.0
2	MEDIUM CERAMIC TOP STORA...	77916	81417.0
3	WORLD WAR 2 GLIDERS ASST...	54415	13586.0
4	JUMBO BAG RED RETROSPOT	46181	85221.0
5	WHITE HANGING HEART T-LIG...	36725	100448.0

Above 5 are the top selling products in the retail stores.

Objective: To find the top 5 products in terms of revenue made by the retail store.

Query 7:

```
SELECT
  Description,
  SUM(Quantity) AS total_quantity_sold,
  ROUND(SUM(Quantity*UnitPrice)) AS total_revenue
FROM
  `CSM.sales`
GROUP BY
  Description
ORDER BY
  total_revenue DESC
LIMIT 5
```

Output 7:

Row	Description	total_quantity_sold	total_revenue
1	PAPER CRAFT , LITTLE BIRDIE	80995	168470.0
2	WHITE HANGING HEART T-LIG...	36725	100448.0
3	JUMBO BAG RED RETROSPOT	46181	85221.0
4	MEDIUM CERAMIC TOP STORA...	77916	81417.0
5	PARTY BUNTING	15279	68715.0

Even though it is the same product that is sold in highest quantity and in terms of revenue as well, it's not the same order that is followed further.

Objective: To find the percentage of number of users from each countries

Query 8:

```
WITH cte1 AS
  (SELECT
    COUNT(DISTINCT CustomerID) AS total_no_of_customers
  FROM
    `CSM.sales`
  ),
cte2 AS
  (SELECT
    Country,
    COUNT(DISTINCT CustomerID) AS no_of_customers
  FROM
    `CSM.sales`
  GROUP BY
    Country
    )
SELECT
  Country,ROUND(100*(b.no_of_customers/a.total_no_of_customers),2) AS percentage_of_customers
FROM
  cte1 a, cte2 b
ORDER BY
  percentage_of_customers DESC
LIMIT 5
```

Output 8:

Row	Country	percentage_of_customers
1	United Kingdom	90.35
2	Germany	2.16
3	France	2.02
4	Spain	0.7
5	Belgium	0.58

Majority users in this dataset is from United Kingdom followed by Germany and France which are really low in numbers.

Objective: To understand peak hours of sales

Query 9:

```
WITH cte1 AS
  (SELECT
    COUNT(*) AS total_orders
  FROM
    `CSM.sales`),
cte2 AS
  (SELECT
    EXTRACT(HOUR FROM InvoiceDate) AS hour,
    COUNT(*) AS no_of_orders
  FROM
    `CSM.sales`
  GROUP BY
    hour)
SELECT
  hour, b.no_of_orders ,ROUND(100*(b.no_of_orders/a.total_orders),2) AS percentage_orders
FROM
  cte1 a,cte2 b
ORDER BY
  percentage_orders DESC
LIMIT 10
```

Output 9:

Row	hour	no_of_orders	percentage_orders
1	12	68181	18.17
2	13	60558	16.14
3	14	51136	13.63
4	11	46514	12.4
5	15	42799	11.41
6	10	35506	9.46
7	16	22790	6.07
8	9	20435	5.45
9	17	12237	3.26
10	8	8066	2.15

Highest percentage of orders are observed in noon and afternoon hours that is from 11 to 14.

RFM analysis for customer segmentation

Objective: Using quantity and unit price we can find amount associated with each invoice number.

Query 1:

```
SELECT
  InvoiceNo,
  SUM(Quantity*UnitPrice) AS amount
FROM
  `CSM.sales`
GROUP BY
  InvoiceNo
```

Output 1:

Row	InvoiceNo	amount
1	571035	783.8599999999...
2	580158	269.9600000000...
3	572215	653.64
4	580553	615.2799999999...
5	570467	1562.56
6	550644	307.45
7	539421	550.8400000000...
8	546569	939.3599999999...
9	553210	860.1800000000...

This output is saved as a new table named bill for future use.

Objective: Compute recency, frequency, and monetary values for each customer.

Recency: Reference date –last purchase date of each customer (Reference date : Max (last purchase date in days of all the customers) + 1)

Frequency: No of purchases / (difference between first purchase date and last purchase date in months for each customer)

Monetary: Sum of the total purchase amount

Query 2:

```
WITH cte AS
  (SELECT
    CustomerID,
    DATE(MIN(InvoiceDate)) AS first_purchase_date,
    DATE(MAX(InvoiceDate)) AS last_purchase_date,
    COUNT(DISTINCT a.InvoiceNo) AS no_of_purchases,
    SUM(amount) AS monetary
  FROM
    `CSM.sales` a
  LEFT JOIN
    `CSM.bill` b
```

```
  ON
    a.InvoiceNo = b.InvoiceNo
  GROUP BY
    CustomerID
  )
  SELECT
    *,
    DATE_DIFF(ref_date,last_purchase_date, day) AS recency,
    no_of_purchases/cus_months AS freq
  FROM
    (SELECT
      *,
      MAX(cte.last_purchase_date) OVER()+1 AS ref_date,
      DATE_DIFF(cte.last_purchase_date, cte.first_purchase_date, month)+1 AS cus_months
    FROM cte
    )
```

Output 2:

Row	CustomerID	first_purchase_date	last_purchase_date	no_of_purchases	monetary	ref_date	cus_months	recency	freq
17	15860.0	2010-12-06	2011-10-30	8	14714.93000000...	2011-12-10	11	41	0.727272727272...
18	15866.0	2011-08-30	2011-08-30	1	7546.560000000...	2011-12-10	1	102	1.0
19	17705.0	2011-02-11	2011-12-06	9	18247.51999999...	2011-12-10	11	4	0.818181818181...
20	16837.0	2011-04-08	2011-11-08	9	55462.23999999...	2011-12-10	8	32	1.125
21	18061.0	2010-12-08	2011-11-18	10	9842.43	2011-12-10	12	22	0.833333333333...

This dataset is saved as RFM.

Objective: We will assign quintiles to each recency, frequency, and monetary values based on the table given below.

Percentile	Recency_Score r_score	Frequency_Score f_score	Monetary_Score m_score
0 – 20	5	1	1
20 - 40	4	2	2
40 - 60	3	3	3
60 - 80	2	4	4
80 - 100	1	5	5

Query 3:

```
SELECT
  a.*,
  CAST(ROUND((f_score+m_score)/2) AS INT64) AS fm_score
FROM
  (SELECT
    CustomerID,
    recency, freq,monetary,
    (CASE WHEN recency<=r20 THEN 5
          WHEN recency<=r40 THEN 4
          WHEN recency<=r60 THEN 3
          WHEN recency<=r80 THEN 2
          WHEN recency<=r100 THEN 1 END) AS r_score,
```

```
    (CASE WHEN freq<=f20 THEN 1
          WHEN freq<=f40 THEN 2
          WHEN freq<=f60 THEN 3
          WHEN freq<=f80 THEN 4
          WHEN freq<=f100 THEN 5 END) AS f_score,
    (CASE WHEN monetary<=m20 THEN 1
          WHEN monetary<=m40 THEN 2
          WHEN monetary<=m60 THEN 3
          WHEN monetary<=m80 THEN 4
          WHEN monetary<=m100 THEN 5 END) AS m_score
  FROM
    `CSM.quintiles`
  ) a
```

OUTPUT 3:

Row	CustomerID	recency	freq	monetary	r_score	f_score	m_score	fm_score
1	17528.0	2	2.0	103900.9700000...	5	5	5	5
2	17243.0	2	2.0	114422.3000000...	5	5	5	5
3	18109.0	2	2.0	83905.08999999...	5	5	5	5
4	16525.0	3	2.0	83881.11999999...	5	5	5	5
5	16000.0	3	2.0	60462.39999999...	5	5	5	5
6	15498.0	3	2.0	387068.9900000...	5	5	5	5
7	16672.0	3	2.0	49984.91999999...	5	5	5	5
8	14216.0	4	2.0	23498.97999999...	5	5	4	5
9	13078.0	4	2.0	154211.9899999...	5	5	5	5
10	15621.0	5	2.0	5513.619999999...	5	5	2	4

r_score, f_score, m_score are calculated based on the quintiles, and fm_score is calculated as the mean of f_score and m_score. These scores will be further used to segment customers. Hence this data is stored as a new table named scores.

Objective: Based on the RFM scores that was previously calculated, we want to group the customers into 11 personas (reference from UK Data & Marketing Association (DMA))

Customer Segment	Activity	Actionable Tip
Champions	Bought recently, buy often and spend the most!	Reward them. Can be early adopters for new products. Will promote your brand.
Loyal Customers	Spend good money with us often. Responsive to promotions.	Upsell higher value products. Ask for reviews. Engage them.
Potential Loyalist	Recent customers, but spent a good amount and bought more than once.	Offer membership / loyalty program, recommend other products.
Recent Customers	Bought most recently, but not often.	Provide on-boarding support, give them early success, start building relationship.
Promising	Recent shoppers, but haven't spent much.	Create brand awareness, offer free trials
Customers Needing Attention	Above average recency, frequency and monetary values. May not have bought very recently though.	Make limited time offers, Recommend based on past purchases. Reactivate them.
About To Sleep	Below average recency, frequency and monetary values. Will lose them if not reactivated.	Share valuable resources, recommend popular products / renewals at discount, reconnect with them.
At Risk	Spent big money and purchased often. But long time ago. Need to bring them back!	Send personalized emails to reconnect, offer renewals, provide helpful resources.
Can't Lose Them	Made biggest purchases, and often. But haven't returned for a long time.	Win them back via renewals or newer products, don't lose them to competition, talk to them.
Hibernating	Last purchase was long back, low spenders and low number of orders.	Offer other relevant products and special discounts. Recreate brand value.
Lost	Lowest recency, frequency and monetary scores.	Revive interest with reach out campaign, ignore otherwise.

Customer Segment	Recency Score Range	Frequency & Monetary Combined Score Range
Champions	4-5	4-5
Loyal Customers	2-5	3-5
Potential Loyalist	3-5	1-3
Recent Customers	4-5	0-1
Promising	3-4	0-1
Customers Needing Attention	2-3	2-3
About To Sleep	2-3	0-2
At Risk	0-2	2-5
Can't Lose Them	0-1	4-5
Hibernating	1-2	1-2
Lost	0-2	0-2

Query 4:

```
SELECT
  CustomerID,
  recency,freq, monetary,
  r_score, f_score, m_score,
  fm_score,
CASE WHEN(r_score= 5 AND fm_score= 5)
        OR(r_score= 5 AND fm_score= 4)
        OR(r_score= 4 AND fm_score= 5)
      THEN 'Champions'
      WHEN (r_score= 5 AND fm_score=3)
        OR(r_score= 4 AND fm_score= 4)
        OR(r_score= 3 AND fm_score= 5)
        OR(r_score= 3 AND fm_score= 4)
      THEN 'Loyal Customers'
      WHEN (r_score= 5 AND fm_score= 2)
        OR(r_score= 4 AND fm_score= 2)
        OR(r_score= 3 AND fm_score= 3)
        OR(r_score= 4 AND fm_score= 3)
      THEN 'Potential Loyalists'
      WHEN r_score= 5 AND fm_score= 1 THEN 'Recent Customers'
      WHEN(r_score= 4 AND fm_score= 1)
        OR(r_score= 3 AND fm_score= 1)
      THEN 'Promising'
      WHEN(r_score= 3 AND fm_score= 2)
        OR(r_score= 2 AND fm_score= 3)
        OR(r_score= 2 AND fm_score= 2)
      THEN'Customers Needing Attention'
      WHEN r_score= 2 AND fm_score= 1 THEN 'About to Sleep'
      WHEN(r_score= 2 AND fm_score= 5)
        OR(r_score= 2 AND fm_score= 4)
        OR(r_score= 1 AND fm_score= 3)
      THEN'At Risk'
      WHEN(r_score= 1 AND fm_score= 5)
        OR(r_score= 1 AND fm_score= 4)
      THEN'Cant Lose Them'
      WHEN r_score= 1 AND fm_score= 2 THEN 'Hibernating'
      WHEN r_score= 1 AND fm_score= 1 THEN'Lost'
      END AS rfm_segment
FROM `CSM.scores`
```

Output 4:

Row	CustomerID	recency	freq	monetary	r_score	f_score	m_score	fm_score	rfm_segment
1	13404.0	2	0.25	2029.649...	5	1	1	1	Recent Customers
2	13525.0	15	0.25	1531.76	5	1	1	1	Recent Customers
3	18080.0	19	0.25	1231.5	4	1	1	1	Promising
4	13962.0	22	0.25	246.2999...	4	1	1	1	Promising
5	18246.0	24	0.25	669.8	4	1	1	1	Promising

This output is stored as new table named segmented data 2.

Objective: To find the percentage of customers belonging to each rfm_segment

Query 5:

```
WITH cte1 AS
  (SELECT
    COUNT(DISTINCT CustomerID) AS total_customers
  FROM
    `CSM.segmented data 2`),
  cte2 AS
    (SELECT
      rfm_segment,
      COUNT(CustomerID) AS no_of_customers
    FROM
      `CSM.segmented data 2`
    GROUP BY
      rfm_segment
    )
SELECT
  *,
  ROUND(100*(cte2.no_of_customers/cte1.total_customers),2) AS percentage_of_customers
FROM cte2, cte1
```

Output 5:

Row	rfm_segment	no_of_customers	total_customers	percentage_of_custo
1	Lost	13	4311	0.3
2	About to Sleep	39	4311	0.9
3	Promising	51	4311	1.18
4	Recent Customers	24	4311	0.56
5	Hibernating	383	4311	8.88
6	Customers Needing Attention	898	4311	20.83
7	Potential Loyalists	1077	4311	24.98
8	At Risk	513	4311	11.9
9	Loyal Customers	652	4311	15.12
10	Cant Lose Them	113	4311	2.62
11	Champions	548	4311	12.71

Highest percentage of customers belong to Potential Loyalists and then Customers needing attention and then Loyal customers.