Marketing Analytics – Lab 4

DATETIME

- The MIN and MAX functions are used to find the minimum and maximum values in a column, respectively.

Example: The marketing team wants to review the performance of their latest campaign and need to know the exact date range to analyze the data. We want to know the date range for our marketing campaign.

```
SELECT
  MIN(campaign_date) AS start_date,
  MAX(campaign_date) AS end_date
FROM `marketing_data.campaign_performance`;
```

DATETIME

- The DATE_DIFF function is used to calculate the difference between two dates in days, months, or years.
- The CURRENT_DATE function is used to get the current date.

Example: The marketing team wants to review the performance of their latest campaign and need to calculate the number of days between the order date and today's date for each order.

```sql
SELECT
  customer_id,
  DATE_DIFF(CURRENT_DATE(), MAX(purchase_date), DAY) AS recency_score
FROM customer_purchases
GROUP BY customer_id;
```

Partition: NTILE()

- **The NTILE function** in SQL is used to divide a result set into a specified number of groups or buckets, each containing an equal number of rows.
- It assigns a unique value to each row, indicating which group or bucket it belongs to.
- For example, if we use NTILE(4), the result set will be divided into 4 groups, with each group having roughly an equal number of rows.
- The rows in the first group will be assigned the value 1, the rows in the second group will be assigned the value 2, and so on.
- **The NTILE function can be used with the OVER clause to apply the function to a specific column in a table or result set**: to specify the ordering of the rows and the window or partition over which the function operates.

Partition: NTILE()

- If we want to divide the sales data into 3 buckets based on the total sales for each product, we can use the NTILE function.

Date	Product	Sales
2021-01-01	A	100
2021-01-01	B	50
2021-01-02	A	75
2021-01-02	B	90
2021-01-03	A	150
2021-01-03	B	75
2021-01-04	A	120
2021-01-04	B	100

```
SELECT
  Product,
  SUM(Sales) AS TotalSales,
  NTILE(3) OVER (ORDER BY SUM(Sales) DESC) AS SalesBucket
FROM SalesData
```

Product	TotalSales	SalesBucket
A	445	1
B	315	2

Measure of frequency

Frequency refers to the number of times a customer makes a purchase within a specific period.

We can calculate it based on the **total number of purchases made by the customer** or the **total number of days on which the customer made a purchase**.

For example, in the case of measuring the frequency of customer purchases, you could calculate it as the total number of purchases made by the customer within a certain period (e.g., a month, a quarter, or a year).

Alternatively, calculate it as the total number of unique days on which the customer made a purchase during that same period.

Measure of frequency

Both methods have their advantages and disadvantages, depending on the specific business goals and objectives.

- Calculating frequency based on the total number of purchases: more precise measurement of the customer's engagement and purchasing behavior.
- Calculating the total number of unique days: more comprehensive picture of the customer's overall engagement with the brand over time.

RFM Analysis – Segmentation – Solution 1 – 125 segments in total

```sql
WITH rfm_scores AS (
  SELECT
  customer_id,
  DATE_DIFF(CURRENT_DATE(), MAX(trans_date), DAY) AS recency_score,
  COUNT(DISTINCT trans_date) AS frequency_score,
  SUM(tran_amount) AS monetary_score
  FROM `mis784-lab2.lab4.retail_date_transactions`
  GROUP BY customer_id
),
rfm_quintiles AS (
  SELECT
    customer_id,
    NTILE(5) OVER (ORDER BY recency_score ASC) AS recency_quintile,
    NTILE(5) OVER (ORDER BY frequency_score DESC) AS frequency_quintile,
    NTILE(5) OVER (ORDER BY monetary_score DESC) AS monetary_quintile
  FROM rfm_scores
)
SELECT
  customer_id,
  CONCAT(recency_quintile, frequency_quintile, monetary_quintile) AS rfm_cell
FROM rfm_quintiles
ORDER BY rfm_cell DESC;
```

Row	customer_id	recency_score	frequency_score	monetary_score
1	CS3884	394	24	1859
2	CS1503	385	27	1927
3	CS3749	396	23	1658
4	CS4123	389	27	1701
5	CS2368	391	27	1859
6	CS3373	384	25	1597
7	CS3417	390	29	1954
8	CS1313	395	29	2072

RFM Analysis – Segmentation – Solution 2 (add them to single score)

```sql
WITH rfm_scores AS (
  SELECT
  customer_id,
  DATE_DIFF(CURRENT_DATE(), MAX(trans_date), DAY) AS recency_score,
  COUNT(DISTINCT trans_date) AS frequency_score,
  SUM(tran_amount) AS monetary_score
  FROM `mis784-lab2.lab4.retail_date_transactions`
  GROUP BY customer_id
),
rfm_quintiles AS (
  SELECT
    customer_id,
    NTILE(5) OVER (ORDER BY recency_score ASC) AS recency_quintile,
    NTILE(5) OVER (ORDER BY frequency_score DESC) AS frequency_quintile,
    NTILE(5) OVER (ORDER BY monetary_score DESC) AS monetary_quintile
  FROM rfm_scores
)
SELECT
  customer_id, recency_score, frequency_score, monetary_score,
  recency_quintile + frequency_quintile + monetary_quintile AS rfm_score
FROM rfm_quintiles
ORDER BY rfm_score DESC;
```

Row	customer_id	recency_score	frequency_score	monetary_score	rfm_score
1	CS4374	385	25	1596	15
2	CS5727	384	22	1666	15
3	CS5630	398	25	1671	15
4	CS6024	384	25	1745	15
5	CS5346	388	25	1801	15
6	CS3557	390	24	1872	15
7	CS2113	390	27	1892	15
8	CS1508	398	28	1933	15

RFM Analysis – Draw the lines between segments depends on then nature of your business

- If you want to reward your best purchasers, you may want to focus on customers with an RFM cell of 555.
- If you want to open things up a bit to include recent and frequent customers who may have spent a little less, you could create
- The key when creatin
  Marketing will lose the
  (5*5*5) — but small e
  several potential segr

	Cannot lose	Active fans	Promising newbies	Potential churners
RFM cell values	355, 255	543, 542, 453, 452	525, 524, 515, 514	335, 334, 325, 324
Conditions for inclusion	Low R High F&M	High R&F Low M	High R&M Low F (so far)	Mid R Low F High M
Description	Big spenders who haven't purchased lately	Customers who buy often & recently, but at low price points	New customers with large orders	High spending customers who haven't purchased often or lately

SQL CASE Expression

The CASE expression goes through conditions and returns a value when the first condition is met (like an if-then-else statement). So, once a condition is true, it will stop reading and return the result. If no conditions are true, it returns the value in the ELSE clause.

```
CASE
    WHEN condition1 THEN result1
    WHEN condition2 THEN result2
    WHEN conditionN THEN resultN
    ELSE result
END;
```

RFM Analysis – Segmentation – Solution 3 – 5 segments based on the business context

```sql
WITH rfm_scores AS (
  SELECT
  customer_id,
  DATE_DIFF(CURRENT_DATE(), MAX(trans_date), DAY) AS recency_score,
  COUNT(DISTINCT trans_date) AS frequency_score,
  SUM(tran_amount) AS monetary_score
  FROM `mis784-lab2.lab4.retail_date_transactions`
  GROUP BY customer_id
),
rfm_quintiles AS (
  SELECT
    customer_id,
    NTILE(5) OVER (ORDER BY recency_score ASC) AS recency_quintile,
    NTILE(5) OVER (ORDER BY frequency_score DESC) AS frequency_quintile,
    NTILE(5) OVER (ORDER BY monetary_score DESC) AS monetary_quintile
  FROM rfm_scores
)
SELECT
  customer_id, recency_score, frequency_score, monetary_score,
  CONCAT(recency_quintile, frequency_quintile, monetary_quintile) AS rfm_cell,
  CASE
    WHEN recency_quintile IN (3,5) AND frequency_quintile IN (5) AND monetary_quintile IN (5) THEN 'Cannot lose'
    WHEN recency_quintile IN (5) AND frequency_quintile IN (4,5) AND monetary_quintile IN (3,4,5) THEN 'Active fans'
    WHEN recency_quintile IN (5) AND frequency_quintile IN (2,3) AND monetary_quintile IN (4,5) THEN 'Promising newbies'
    WHEN recency_quintile IN (3,4,5) AND frequency_quintile IN (2,3,4) AND monetary_quintile IN (2,3,4) THEN 'Potential churners'
    ELSE 'Other'
  END AS rfm_group
FROM rfm_quintiles
ORDER BY rfm_group DESC, customer_id;
```

customer_id	recency_score	frequency_score	monetary_score	rfm_cell	rfm_group
CS1289	387	19	1355	534	Promising newbies
CS1320	396	18	1382	534	Promising newbies
CS1358	395	19	1408	534	Promising newbies
CS1361	401	19	1470	534	Promising newbies
CS1432	387	19	1362	534	Promising newbies
CS1616	394	19	1434	534	Promising newbies
CS1622	384	19	1360	534	Promising newbies
CS1698	389	19	1397	534	Promising newbies

customer_id

Performance Analysis

To determine which type of customers are most likely to respond to the promotion, we can calculate the response rate for each segment.

How to calculate the response rate for each group?

Step-by-step Solutions

Here's the steps to combine your query in BigQuery to calculate the response rate for each segment:

- RFM Analysis – RFM cell – Week 4
- Segments based on your requirements
- Join tables if necessary
- Calculate response rate
- Show the results

Row	rfm_segment	avg_response_ra
1	Other	0.10135018...
2	Promising newbies	0.09090909...
3	Potential churners	0.04444444...
4	Active fans	0.04117647...
5	Cannot lose	0.00875395...

Read and explain the results

Based on the analysis of transaction patterns, we can see that the “Active fans” and “Cannot lose” segments have the higher repones rate, indicating that these customers are more likely to respond positively to promotions and are more loyal to the brand.

While “”Promising newbies” and “potential churners” segments have the lower response rates to the promotional campaign, suggesting that these customers may be at risk of leaving and need to be engaged with targeted promotions to prevent customer missing.

Therefore, for future promotion campaigns, it is recommended to focus on the “Active fans” and “Cannot lose” segments, as they are more likely to respond positively to promotions and have a higher lifetime value. At the same time, it is also important to target the “Promising newbies” and “potential churners” segment with personalized and relevant promotions to retain them and prevent churn.

Please refer to the Week 4 Tutorial solution, Question 6, SQL queries

```sql
),
rfm_segments AS (
  SELECT
    customer_id,
    CONCAT(recency_quintile, frequency_quintile, monetary_quintile) AS rfm_cell,
    CASE
      WHEN CONCAT(recency_quintile, frequency_quintile, monetary_quintile) IN ('355', '255') THEN 'Cannot lose'
      WHEN CONCAT(recency_quintile, frequency_quintile, monetary_quintile) IN ('543', '542', '453', '452') THEN 'Active fans'
      WHEN CONCAT(recency_quintile, frequency_quintile, monetary_quintile) IN ('525', '524', '515', '514') THEN 'Promising newbie
      WHEN CONCAT(recency_quintile, frequency_quintile, monetary_quintile) IN ('335', '334', '325', '324') THEN 'Potential churne
      ELSE 'Other'
    END AS rfm_segment
  FROM
    rfm_quintiles
),
response_rates AS (
  SELECT
    s.rfm_segment,
    s.rfm_cell,
    COUNTIF(r.response =1) AS response_count,
    COUNT(r.customer_id) AS total_customers,
    COUNTIF(r.response =1)/COUNT(r.customer_id) AS response_rate
  FROM
    `mis384-lab2.lab4.response` r
  JOIN
    rfm_segments s
  ON
    r.customer_id = s.customer_id
  GROUP BY
    s.rfm_cell,
    s.rfm_segment
)
SELECT
  rfm_segment,
  AVG(response_rate) as avg_response_rate
FROM
  response_rates
GROUP BY
  rfm_segment
ORDER BY
  avg_response_rate DESC;
```

```sql
rfm_segment AS (
  SELECT
    customer_id,
    CONCAT(recency_quintile, frequency_quintile, monetary_quintile) AS rfm_cell
  FROM
    rfm_quintiles
),
response_rates AS (
  SELECT
    s.rfm_cell,
    COUNTIF(r.response =1) AS response_count,
    COUNT(r.customer_id) AS total_customers,
    COUNTIF(r.response =1)/CAST(COUNT(r.customer_id) AS FLOAT64) AS response_r
  FROM
    `mis384-lab2.lab4.response` r
  JOIN
    rfm_segment s
  ON
    r.customer_id = s.customer_id
  GROUP BY
    s.rfm_cell
)
SELECT
  s.rfm_cell,
  rr.response_count,
  rr.total_customers,
  rr.response_rate
FROM
  rfm_segment s
JOIN
  response_rates rr
ON
  s.rfm_cell = rr.rfm_cell
GROUP BY
  s.rfm_cell,
  rr.response_count,
  rr.total_customers,
  rr.response_rate
ORDER BY
  rr.response_rate DESC;
```


Please refer to the Week 4 Tutorial solution, Question 6, SQL queries