**Customer segmentation by RFM Model with Google BigQuery**

# Problem Statement:

A company in the e-commerce sector wants to segment its customers and determine their marketing strategies according to these segments. For that need to perform a RFM analysis for a chain of retail stores that sells a lot of different items and categories.

# Various Terminology Explanation:

1. **Customer Segmentation**:

Customer segmentation involves grouping of existing and potential customer based on shared characteristics.

Shared characteristics used for segmentation such as: Demographics, Behavioral, Psycho-graphic, Geographic, Firmographic segmentation.

1. **RFM Clustering Model:**

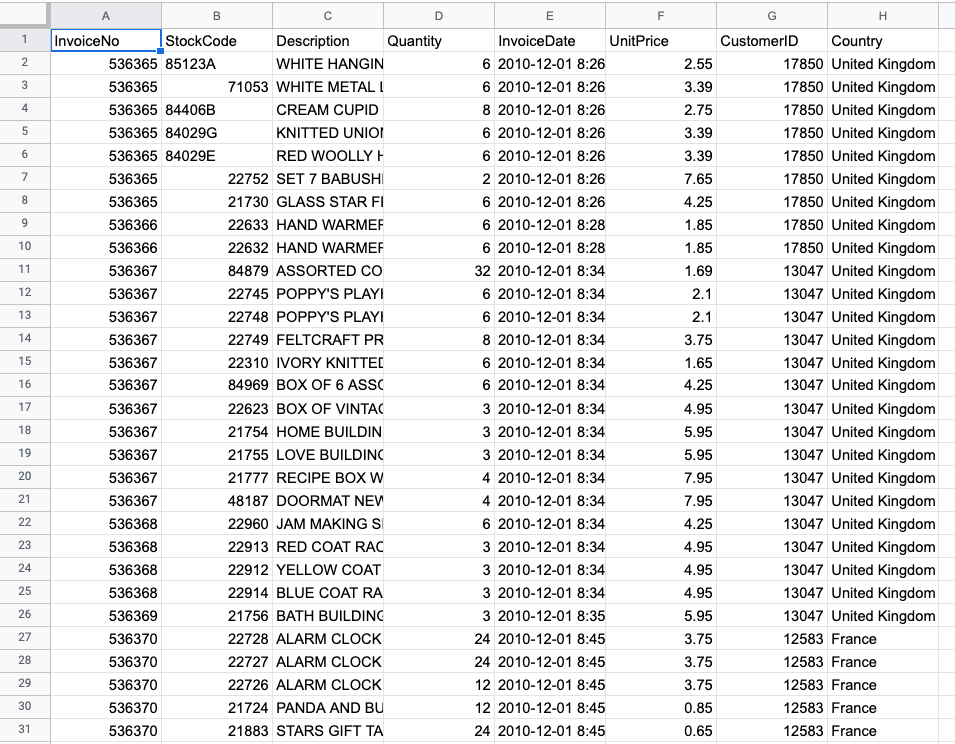
* It mainly built from understanding the customer purchase behavior.
* RFM stands for Recency, Frequency, and Monetary Value.
* RFM helps the company understand the customer’s characteristics based on their historical transactions.
* To extract information about their buying behavior, we need to analyze the RFM factor.
* **Recency:** How recently a customer has made a purchase, either in months, days or weeks depending on your market’s typical purchase cycle?
* **Frequency:** How often a customer makes a purchase, typically measured over the twelve months leading up to each customer’s last purchase?
* **Monetary Value:** How much money a customer spends on purchase, either in-total or on-average over the same twelve month period?
* Methodology:

Calculate RFM score for each customer by grouping the score for R, F & M (1 for lowest & 5 Highest score)

# Column profiling of Retail store Data-Set:

* **InvoiceNo:** Invoice number. Nominal, 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. Nominal, a 5-digit integral number uniquely assigned to each distinct product.
* **Description:** Product (item) name. Nominal.
* **Quantity:** The quantities of each product (item) per transaction. Numeric.
* **InvoiceDate**: Invoice Date and time. Numeric, the day and time when each transaction was generated.
* **UnitPrice:** Unit price. Numeric, Product price per unit in sterling.
* **CustomerID:** Customer number. Nominal, a 5-digit integral number uniquely assigned to each customer.
* **Country:** Country name. Nominal, the name of the country where each customer resides.

***Column profiling of Retail store Data-Set***



The RFM Segmentation can be executed using these five steps:

1. Data processing
2. Compute for recency, frequency, and monetary values per customer
3. Determine quantiles for each RFM metric
4. Assign scores for each RFM metric
5. Define the RFM segments using the scores in step 4

## **Data Processing**

**Adding the data to BigQuery:**

Create a new dataset and upload ‘sales.csv’ as a new table.

We created a dataset named `retail` in a project customer segmentation and the table name is `sales`.

Now if we look at the data we can see that there are products that have been bought in quantities more than one and we have unit price for those products but we do not have the total cost of that product.

So the first thing we’re gonna do is find the total cost for that product i.e., quantity \* unit price -

**Total cost of that product as** Amount**:**

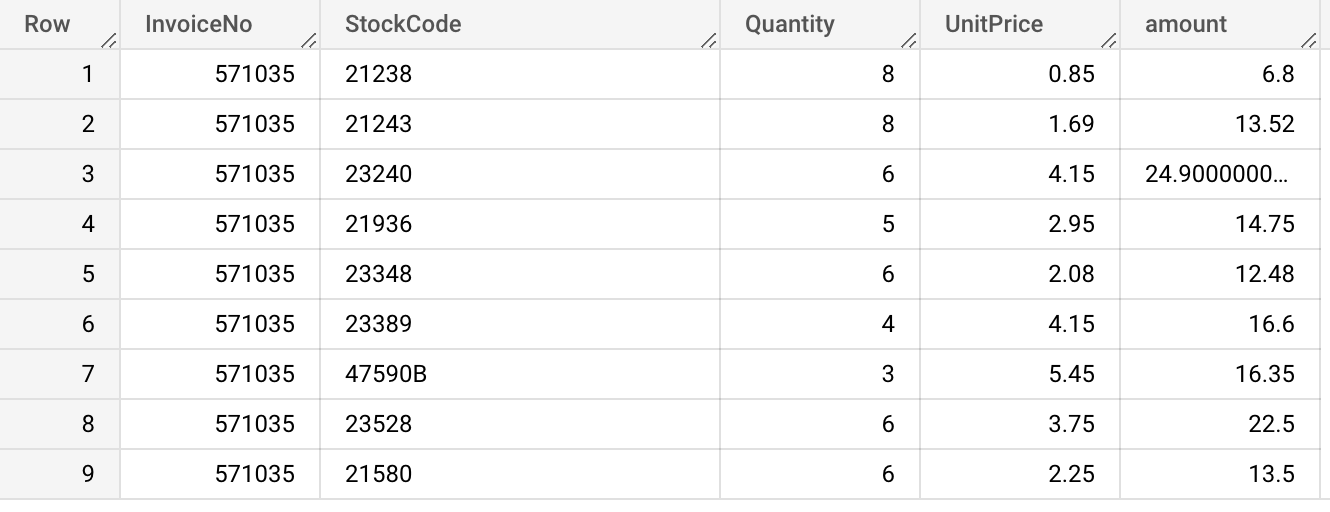
SELECT

  InvoiceNo,StockCode,Quantity,UnitPrice,

  (Quantity\*UnitPrice) AS amount

FROM

  `customer-segmentation.retail.sales`



Amount spent on each visit: For each invoice id there may be different products, and till now we have calculated the total for each product, but we do not have the total bill amount for individual invoice ids.

For this we use the above query and create a CTE. Then group it by invoice id and sum the total cost, getting the actual bill amount.

WITH

 bills AS (

 SELECT

   InvoiceNo,

   (Quantity\*UnitPrice) AS amount

 FROM

   `customer-segmentation.retail.sales`)

SELECT

 InvoiceNo,

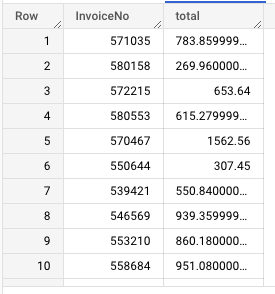
 SUM(amount) AS total

FROM

 bills

GROUP BY

 InvoiceNo



Save this data as a `bill` table.

## **Compute for recency, frequency and monetary values per customer:**

Because we will be joining the bill and sales table we will get multiple rows on the key that we are going to join ie: InvoiceNo, so we'll just take one row per InvoiceNo.

For that we will use the row number and get the data from the sales table that we need.

monetary value: We will join the `bill` table that we saved with the `sales` table and add the total cost on the customer level for monetary value.

SELECT

CustomerID,

DATE(MAX(InvoiceDate)) AS last\_purchase\_date,

DATE(MIN(InvoiceDate)) AS first\_purchase\_date,

COUNT(DISTINCT InvoiceNo) AS num\_purchases,

SUM(total) AS monetary,

FROM(

Select s.CustomerID, s.InvoiceDate, s.InvoiceNo, b.total

,ROW\_NUMBER() OVER(PARTITION BY s.InvoiceNo ORDER BY s.InvoiceNo) AS RN

From

`customer-segmentation-373712.retail.sales` s

LEFT JOIN

`customer-segmentation-373712.retail.bill` b

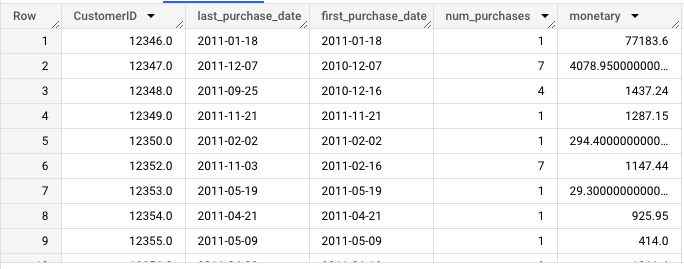
ON

s.InvoiceNo=b.InvoiceNo

) A

WHERE A.RN = 1

GROUP BY CustomerID



We can save this table as monetary

Recency: For recency, we chose a reference date, which is the most recent purchase in the dataset. In other situations, one may select the date when the data was analyzed instead.

After choosing the reference date, we get the date difference between the reference date and the last purchase date of each customer. This is the recency value for that particular customer.

Frequency: For frequency we calculate the months the person has been a customer by difference in first and last purchase +1 ( for when first and last month are same and the customer should be considered a customer for at least 1 month)

SELECT

\*,

DATE\_DIFF(reference\_date, last\_purchase\_date, DAY) AS recency,

num\_purchases/ (months\_cust) AS frequency,

FROM

(

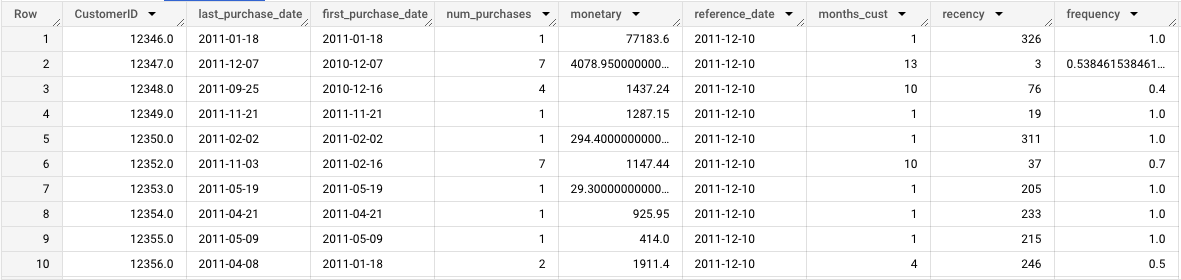
SELECT \*,

MAX(last\_purchase\_date) OVER () + 1 AS reference\_date,

DATE\_DIFF(last\_purchase\_date, first\_purchase\_date, month)+1 AS months\_cust

FROM `customer-segmentation.retail.monetary` )

ORDER BY CustomerID ;



Now that we have the RFM data we can save it as another table named `RFM`.

## **Determine quintiles for each RFM metric**

The next step would be to group the customers into quintiles in terms of their RFM values — we divide the customers into 5 equal groups, according to how high and low they scored in the RFM metrics.

The main advantage of using percentile is we do not have to change or set the values. It will be automatically calculated.

What is a Quintile?

* A quintile is a 1/5th (20 percent) portion of the whole. In statistics, it’s a population or sample divided into five equal groups, according to values of a particular variable. Quintiles are like percentiles, but instead of dividing the data into 100 parts, you divide it in 5 equal parts. Quintiles work with any industry since the data itself defines the ranges; they distribute customers evenly.

We do this for each of recency, frequency and monetary values per customer.

I used BigQuery’s APPROX\_QUANTILES() to achieve this.

How does APPROX\_QUANTILES() work?

* Returns the approximate boundaries for a group of expression values, where number represents the number of quantiles to create.
* This function returns an array of number+1 elements, where the first element is the approximate minimum and the last element is the approximate maximum.

NOTE : Approximate aggregate functions are scalable in terms of memory usage and time, but produce approximate results instead of exact results.

* OFFSET() accesses an ARRAY element by position and returns the element. The approximate\_quantiles will return an array for each percentile and for creating quintiles out of it we will need values at 20, 40 and so on. We save those values as m20, m40 for monetary and f, r for frequency and recency respectively.

SELECT

   a.\*,

   --All percentiles for MONETARY

   b.percentiles[offset(20)] AS m20,

   b.percentiles[offset(40)] AS m40,

   b.percentiles[offset(60)] AS m60,

   b.percentiles[offset(80)] AS m80,

   b.percentiles[offset(100)] AS m100,

   --All percentiles for FREQUENCY

   c.percentiles[offset(20)] AS f20,

   c.percentiles[offset(40)] AS f40,

   c.percentiles[offset(60)] AS f60,

   c.percentiles[offset(80)] AS f80,

   c.percentiles[offset(100)] AS f100,

   --All percentiles for RECENCY

   d.percentiles[offset(20)] AS r20,

   d.percentiles[offset(40)] AS r40,

   d.percentiles[offset(60)] AS r60,

   d.percentiles[offset(80)] AS r80,

   d.percentiles[offset(100)] AS r100

FROM

   `customer-segmentation-373712.retail.RFM` a,

   (SELECT APPROX\_QUANTILES(monetary, 100) percentiles FROM

   `customer-segmentation-373712.retail.RFM`) b,

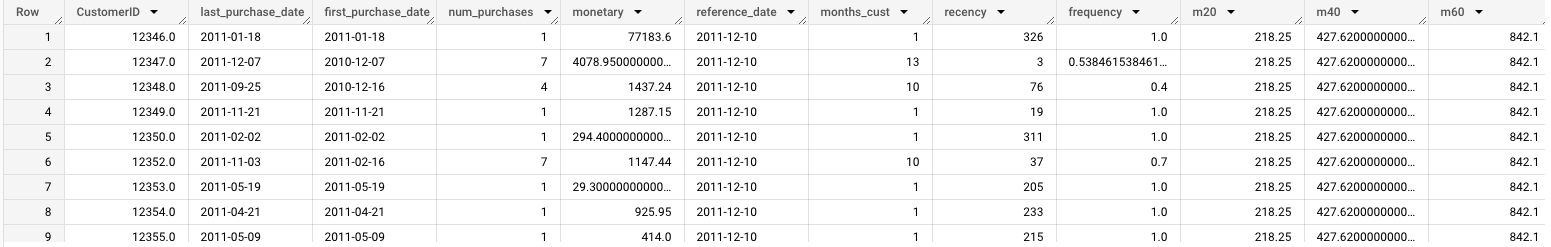
   (SELECT APPROX\_QUANTILES(frequency, 100) percentiles FROM

   `customer-segmentation-373712.retail.RFM`) c,

   (SELECT APPROX\_QUANTILES(recency, 100) percentiles FROM

   `customer-segmentation-373712.retail.RFM`) d

ORDER BY CustomerI



Again, we save these as a new table named `quantile`.

## **Assign scores for each RFM metric :**

Now that we know how each customer fares relative to other customers in terms of RFM values, we can now assign scores from 1 to 5.

Just keep in mind that while with F and M, we give higher scores for higher quintiles, R should be reversed as more recent customers should be scored higher in this metric.

Frequency and Monetary value are combined (as both of them are indicative to purchase volume anyway) to reduce the possible options from 125 to 50.

We will use CASE to get values and assign scores accordingly, so we just get the data from the `quintiles` table that we stored assign scores.

SELECT CustomerID,

m\_score,f\_score,r\_score,

recency, frequency,monetary,

 CAST(ROUND((f\_score + m\_score) / 2, 0) AS INT64) AS fm\_score

 FROM (

     SELECT \*,

     CASE WHEN monetary <= m20 THEN 1

         WHEN monetary <= m40 AND monetary > m20 THEN 2

         WHEN monetary <= m60 AND monetary > m40 THEN 3

         WHEN monetary <= m80 AND monetary > m60 THEN 4

         WHEN monetary <= m100 AND monetary > m80 THEN 5

     END AS m\_score,

     CASE WHEN frequency <= f20 THEN 1

         WHEN frequency <= f40 AND frequency > f20 THEN 2

         WHEN frequency <= f60 AND frequency > f40 THEN 3

         WHEN frequency <= f80 AND frequency > f60 THEN 4

         WHEN frequency <= f100 AND frequency > f80 THEN 5

     END AS f\_score,

     --Recency scoring is reversed

     CASE WHEN recency <= r20 THEN 5

         WHEN recency <= r40 AND recency > r20 THEN 4

         WHEN recency <= r60 AND recency > r40 THEN 3

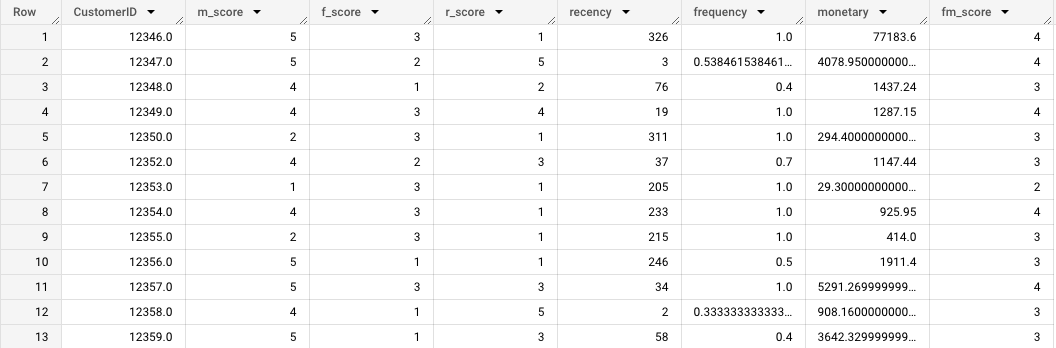
         WHEN recency <= r80 AND recency > r60 THEN 2

         WHEN recency <= r100 AND recency > r80 THEN 1

     END AS r\_score,

     FROM `customer-segmentation-373712.retail.Quintiles`

     )



Now you can save this as another table or create a CTE named score for this and use it for further calculations.

## **Define the RFM segments using these scores :**

The next step is to combine the scores we obtained to define the RFM segment each customer will belong to.

As there are 5 groups for each of the R, F, and M metrics, there are 125 potential permutations.

We will be using the 11 personas in the DMA as a guide and define the R vs. FM scores accordingly.



* For example, in the Champions segment, customers should have bought recently, bought often, and spent the most. Therefore, their R score should be 5 and their combined FM score should be 4 or 5.
* On the other hand, Can’t Lose Them customers made the biggest purchases, and often, but haven’t returned for a long time. Hence their R score should be 1, and FM score should be 4 or 5

SELECT

      CustomerID,

      recency,frequency,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'

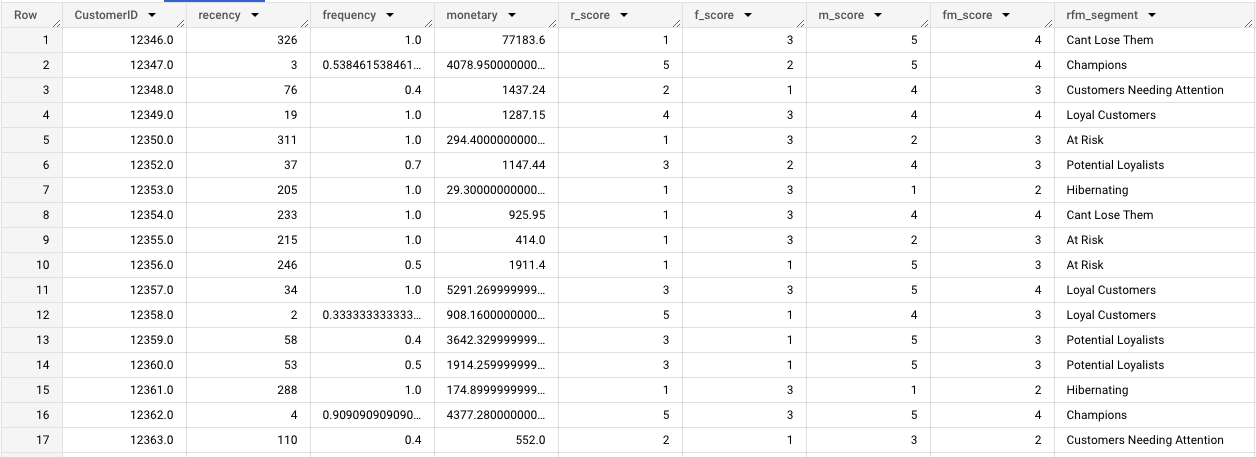
      WHEN r\_score = 1 AND fm\_score = 2 THEN 'Hibernating'

      WHEN r\_score = 1 AND fm\_score = 1 THEN 'Lost'

      END AS rfm\_segment

  FROM `customer-segmentation.retail.score`

  ORDER BY CustomerID



After this step, each customer should have an RFM segment assignment like this.

This type of segmentation focuses on the actual buying behavior and ignores the differences in motivations, intentions, and lifestyles of consumers.

RFM is nonetheless a useful start-off point, and because of its simplicity can be executed fast and in an automated way, giving companies the power to act and decide on business strategies swiftly.