About Business:

In this business case, we will perform Recency-Frequency-Monetary Analysis (RFM) for a chain of retail stores that sells many different items in different categories.

Business Problem:

They need to adjust their marketing budget and have better targeting of customers so they need to know which customers to focus on and how important they are for the business.

Solution:

Start with adding the data to bigquery. Create a new dataset and upload 'sales.csv' as a new table. We created a dataset named `main` 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 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.

```
select
  *
from main.sales;
```

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
571035	21238	RED RETROSPOT CUP	8	2011-10-13 12:50:00.000000 UTC	0.85	12446	RSA
571035	21243	PINK POLKADOT PLATE	8	2011-10-13 12:50:00.000000 UTC	1.69	12446	RSA
571035	23240	SET OF 4 KNICK KNACK TINS DOILY	6	2011-10-13 12:50:00.000000 UTC	4.15	12446	RSA
571035	23209	LUNCH BAG VINTAGE DOILY	10	2011-10-13 12:50:00.000000 UTC	1.65	12446	RSA

So we will calculate the total cost (amount) for that product i.e., quantity * unit price.

Since our study focuses on customer level granularity. We will fetch below details for each customer record:

- **days_since_last_purchase**: to get recency score
- **avg_per_month_purchase_since_first_pruchase**: to get frequency score
- **total_spending**: to get monetary score

SQL Query:

```
with customer_summary as
(
  select
    cast(CustomerID as int) as cid,
    min(extract(date from InvoiceDate)) as first_date,
    max(extract(date from InvoiceDate)) as last_date,
    count(distinct InvoiceNo) as num_purchases,
    round(sum(Quantity * UnitPrice)) as amount
  from main.sales
  group by 1
)
select
  cid,
  timestamp_diff(max(last_date) over(), last_date, day) as days_since_last_purchase,
  round(num_purchases / (timestamp_diff(max(last_date) over (), first_date, month) + 1), 2) as avg_per_month_purchase_s
ince_first_pruchase,
  amount as total_spending
from customer_summary;
```

We saved the above query as view "**vCustomerSummary**".

Output:

cid	days_since_last_purchase	avg_per_month_purchase_since_first_pruchase	total_spending
14156	8	4	98696
12584	2	0.8	1567
14912	15	0.22	868
14911	0	15.15	119715
12514	266	0.1	827
12515	352	0.08	191
12591	316	0.08	339
14016	160	0.31	3301
12586	16	0.5	187
12558	6	1	270

Compute for recency, frequency and monetary values per customer and assign scores for each RFM metric.

SQL Query:

```
with approx_q as
(
  select
    approx  quantiles(days  since  last  purchase, 100) as r,
    approx_quantiles(avg_per_month_purchase_since_first_pruchase, 100) as f,
    approx_quantiles(total_spending, 100) as m
  from main.vCustomerSummary
), rfm_scores as
(
  select
    cid,
    days_since_last_purchase as recency,
    avg_per_month_purchase_since_first_pruchase as frequency,
    total_spending as monetory,
    case
      when days_since_last_purchase <= r[offset(20)] then 5
      when days_since_last_purchase <= r[offset(40)] then 4
      when days_since_last_purchase <= r[offset(60)] then 3
      when days_since_last_purchase <= r[offset(80)] then 2
      else 1
    end as r_score,
    case
      when avg_per_month_purchase_since_first_pruchase <= f[offset(20)] then 1
      when avg_per_month_purchase_since_first_pruchase <= f[offset(40)] then 2
      when avg_per_month_purchase_since_first_pruchase <= f[offset(60)] then 3
      when avg_per_month_purchase_since_first_pruchase <= f[offset(80)] then 4
      else 5
    end as f_score,
    case
      when total_spending <= m[offset(20)] then 1
      when total_spending <= m[offset(40)] then 2
      when total_spending <= m[offset(60)] then 3
      when total_spending <= m[offset(80)] then 4
      else 5
    end as m_score
  from main.vCustomerSummary as c, approx_q as a
)
select
  cid as customer_id,
  recency,
  frequency,
  monetory,
  r_score,
  f_score,
  m_score,
  cast(round((f_score+m_score)/2) as int) as fm_score,
  concat(r_score, f_score, m_score) as rfm_score
from rfm_scores;
```

We saved the above query as view "**vCustomerRFM**".

Output:

customer_id	r_score	f_score	m_score	fm_score	rfm_score
12618	3	3	1	2	331
12584	5	4	4	4	544
12607	2	2	4	3	224
12552	2	2	1	2	221
12514	1	1	3	2	113
14016	2	2	5	4	225
12586	4	3	1	2	431
14912	4	2	3	3	423
12558	5	5	2	4	552
14156	4	5	5	5	455
14911	5	5	5	5	555
12578	3	2	4	3	324
12591	1	1	2	2	112
12733	1	1	2	2	112
12594	3	4	5	5	345
12646	5	4	3	4	543
12515	1	1	1	1	111
12610	3	4	4	4	344
12446	2	2	2	2	222
12349	4	3	3	3	433

We finally 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. As there are five 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.

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.

- For example, in the **Champions** segment, customers should have bought recently, bought often, and spent the most. Therefore, their R score should be five 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 have not returned for a long time. Hence, their R score should be 1, and FM score should be 4 or 5.

Defining the RFM segments

SQL Query:

```
SELECT
   customer_id,
   recency,
   frequency,
   monetory,
   r_score, f_score, m_score,
   fm_score, rfm_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 main.vCustomerRFM;
```

Output:

customer_id	recency	frequency	monetory	r_score	f_score	m_score	fm_score	rfm_score	rfm_segment
12578	20	0.17	2330	3	2	4	3	324	Potential Loyalists
14156	8	4	98696	4	5	5	5	455	Champions
12584	2	0.8	1567	5	4	4	4	544	Champions
12514	266	0.1	827	1	1	3	2	113	Hibernating
12552	38	0.33	215	2	2	1	2	221	Customers Needing Attention
12607	59	0.33	1563	2	2	4	3	224	Customers Needing Attention

Now each customer have a RFM segment assigned. Let us now understand the composition of these RFM segments.

SQL Query:

```
with segment_composition as
(
  select
    rfm_segment,
    max(r_score) as r_score,
    max(f_score) as f_score,
    max(m  score) as m  score,
    count(*) as number  of  customers
  from main.vCustomerSegmentation
  group by 1
)
select
  rfm_segment,
  number_of_customers,
  round(number_of_customers / sum(number_of_customers) over(rows between unbounded preceding and unbounded foll
owing) * 100, 2) || ' %' as pct_customers
from segment_composition
order by m_score desc, f_score desc, r_score desc;
```

Output:

rfm_segment	number_of_customers	pct_customers
Potential Loyalists	691	16.03%
Loyal Customers	798	18.51%
Champions	933	21.64%
At Risk	300	6.96%
Can’t Lose Them	28	0.65%
Customers Needing Attention	770	17.86%
Hibernating	374	8.68%
Recent Customers	1	0.02%
Promising	4	0.09%
About to Sleep	68	1.58%
Lost	344	7.98%