Query Appendix

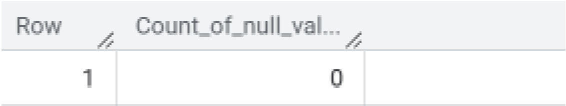
Data Cleaning and Inspection

Customer dataset

- Query a:

```
-- Check Null Values --
SELECT COUNT(*) AS Count_of_null_values
FROM `mis784t22025-466123.MIS784_A2.customer_A2`
WHERE Customer_ID IS NULL;
# No null values
```

- Result:

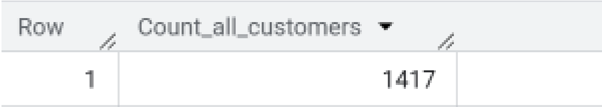


Row	Count_of_null_val...
1	0

- Query b:

```
-- Count of ALL Customers --
SELECT COUNT(Customer_ID) AS Count_all_customers
FROM `mis784t22025-466123.MIS784_A2.customer_A2`
WHERE Customer_ID IS NOT NULL;
# Count of ALL customers: 1417
```

- Result:

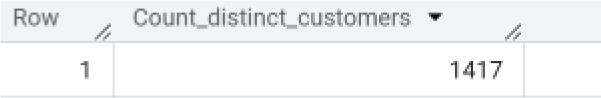


Row	Count_all_customers ▾
1	1417

- Query c:

```
-- COUNT of DISTINCT Customers --
SELECT COUNT(Distinct Customer_ID) AS Count_distinct_customers
FROM `mis784t22025-466123.MIS784_A2.customer_A2`
WHERE Customer_ID IS NOT NULL;
# Count of DISTINCT customers: 1417. Hence, no duplicates exist with every
row's Customer_ID is distinct
```

- Result:

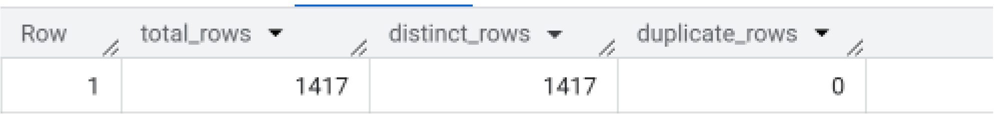


Row	Count_distinct_customers
1	1417

- Query d - Double check:

```
-- Check if any rows are exact duplicates across all columns --
SELECT
  COUNT(*) AS total_rows,
  COUNT(DISTINCT TO_JSON_STRING(t)) AS distinct_rows,
  COUNT(*) - COUNT(DISTINCT TO_JSON_STRING(t)) AS duplicate_rows
FROM `mis784t22025-466123.MIS784_A2.customer_A2` AS t;
# There are no redundant duplicates
```

- Result:

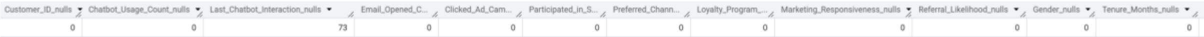


Row	total_rows	distinct_rows	duplicate_rows
1	1417	1417	0

- Query e:

```
-- Check all null values across all columns --
SELECT
  COUNTIF(Customer_ID IS NULL)                AS Customer_ID_nulls,
  COUNTIF(Chatbot_Usage_Count IS NULL)        AS Chatbot_Usage_Count_nulls,
  COUNTIF(Last_Chatbot_Interaction IS NULL) AS
Last_Chatbot_Interaction_nulls,
  COUNTIF(Email_Opened_Count IS NULL)         AS Email_Opened_Count_nulls,
  COUNTIF(Clicked_Ad_Campaigns IS NULL)       AS Clicked_Ad_Campaigns_nulls,
  COUNTIF(Participated_in_Survey IS NULL)     AS Participated_in_Survey_nulls,
  COUNTIF(Preferred_Channel IS NULL)          AS Preferred_Channel_nulls,
  COUNTIF(Loyalty_Program_Status IS NULL)     AS Loyalty_Program_Status_nulls,
  COUNTIF(Marketing_Responsiveness IS NULL) AS
Marketing_Responsiveness_nulls,
  COUNTIF(Referral_Likelihood IS NULL)        AS Referral_Likelihood_nulls,
  COUNTIF(Gender IS NULL)                     AS Gender_nulls,
  COUNTIF(Tenure_Months IS NULL)              AS Tenure_Months_nulls
FROM `mis784t22025-466123.MIS784_A2.customer_A2`;
# There are 73 null values in the Last_Chatbot_Interaction column
```

- Result:

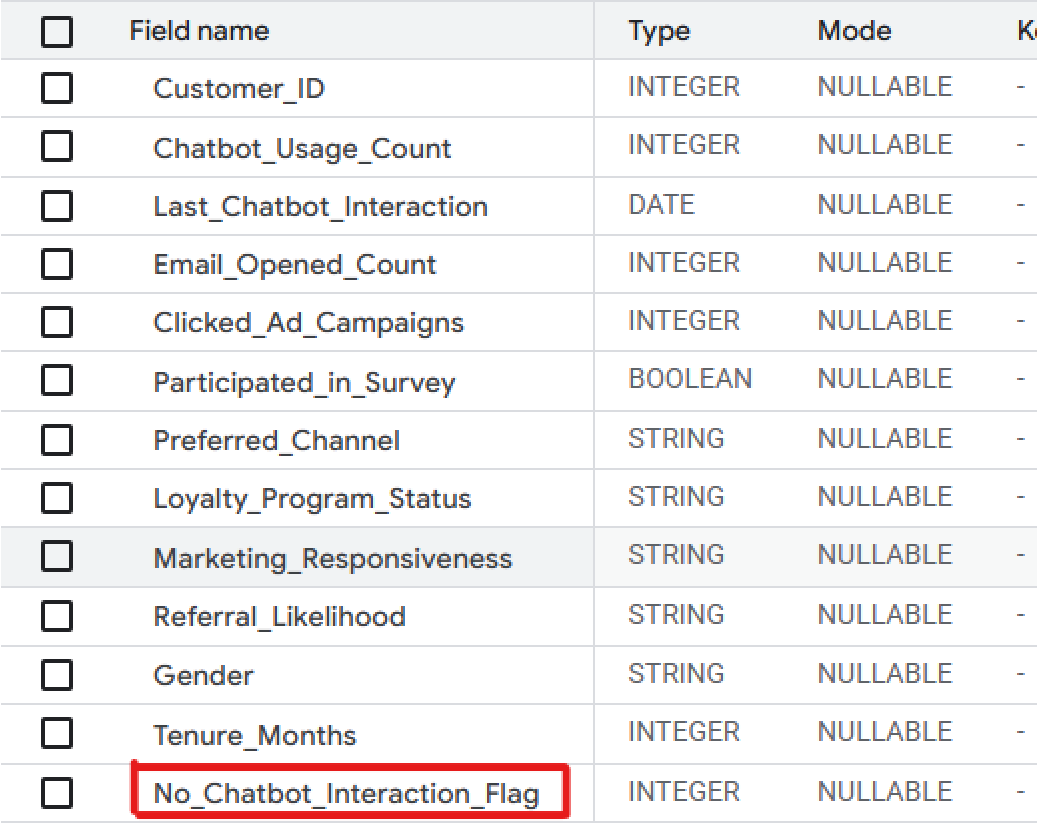


Customer_ID_nulls	Chatbot_Usage_Count_nulls	Last_Chatbot_Interaction_nulls	Email_Opened_C...	Clicked_Ad_Cam...	Participated_in_S...	Preferred_Chann...	Loyalty_Program_...	Marketing_Responsiveness_nulls	Referral_Likelihood_nulls	Gender_nulls	Tenure_Months_nulls
0	0	73	0	0	0	0	0	0	0	0	0

- Query f:

```
-- Create a new table that keeps the original data but adds a flag column
(No_Chatbot_Interaction_Flag) to indicate whether the customer has never
interacted with the chatbot --
CREATE OR REPLACE TABLE `mis784t22025-
466123.MIS784_A2.customer_A2_with_flag` AS
SELECT *,
CASE
  WHEN Last_Chatbot_Interaction IS NULL THEN 1
  ELSE 0
END AS No_Chatbot_Interaction_Flag
FROM `mis784t22025-466123.MIS784_A2.customer_A2`;
```

- Result:



□	Field name	Type	Mode	K
□	Customer_ID	INTEGER	NULLABLE	-
□	Chatbot_Usage_Count	INTEGER	NULLABLE	-
□	Last_Chatbot_Interaction	DATE	NULLABLE	-
□	Email_Opened_Count	INTEGER	NULLABLE	-
□	Clicked_Ad_Campaigns	INTEGER	NULLABLE	-
□	Participated_in_Survey	BOOLEAN	NULLABLE	-
□	Preferred_Channel	STRING	NULLABLE	-
□	Loyalty_Program_Status	STRING	NULLABLE	-
□	Marketing_Responsiveness	STRING	NULLABLE	-
□	Referral_Likelihood	STRING	NULLABLE	-
□	Gender	STRING	NULLABLE	-
□	Tenure_Months	INTEGER	NULLABLE	-
□	No_Chatbot_Interaction_Flag	INTEGER	NULLABLE	-

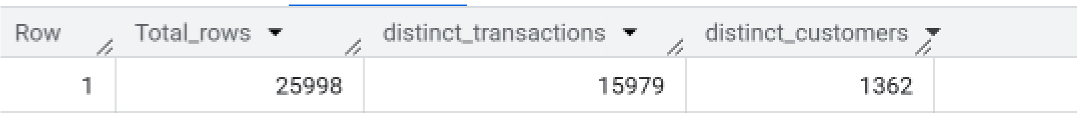
Transaction dataset

- Query a:

```
-- Check transaction records --
SELECT
```

```
  COUNT(*) AS Total_rows,
  COUNT(DISTINCT Transaction_ID) as distinct_transactions,
  COUNT(DISTINCT Customer_ID) as distinct_customers
FROM `mis784t22025-466123.MIS784_A2.transaction_A2`;

# There are 25,998 records in the transaction table with 15979 unique
transaction_id. That means around 10,019 rows are duplicates or multiple
entries per transaction ID. Transactions belong to 1362 unique customers.
However, there are 1417 unique customers in the Customer table, so around 55
customers in the customer table have no matching transactions (or vice
versa)
```

- Result:

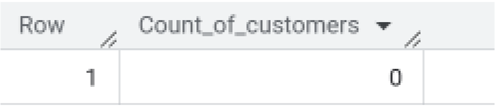


Row	Total_rows	distinct_transactions	distinct_customers
1	25998	15979	1362

- Query b:

```
-- Check if there are any transaction records for customers not present in
the customer dataset --
SELECT COUNT(DISTINCT Customer_ID) AS Count_of_customers
FROM `mis784t22025-466123.MIS784_A2.transaction_A2`
WHERE Customer_ID NOT IN (
  SELECT Customer_ID
  FROM `mis784t22025-466123.MIS784_A2.customer_A2_with_flag`
);
# There are no customer ids in the transaction table that are missing from
the customer table. In other words, every transaction is linked to a valid
customer. This means referential integrity between the transaction and
customer tables is solid. Some customers exist in the customer table but
simply have no transactions.
```

- Result:

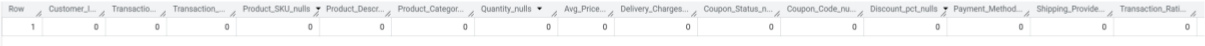


Row	Count_of_customers
1	0

- Query c:

```
-- Check all columns with null values --
SELECT
  COUNTIF(Customer_ID IS NULL)          AS Customer_ID_nulls,
  COUNTIF(Transaction_ID IS NULL)       AS Transaction_ID_nulls,
  COUNTIF(Transaction_Date IS NULL)      AS Transaction_Date_nulls,
  COUNTIF(Product_SKU IS NULL)           AS Product_SKU_nulls,
  COUNTIF(Product_Description IS NULL)   AS Product_Description_nulls,
  COUNTIF(Product_Category IS NULL)      AS Product_Category_nulls,
  COUNTIF(Quantity IS NULL)              AS Quantity_nulls,
  COUNTIF(Avg_Price IS NULL)             AS Avg_Price_nulls,
  COUNTIF(Delivery_Charges IS NULL)      AS Delivery_Charges_nulls,
  COUNTIF(Coupon_Status IS NULL)         AS Coupon_Status_nulls,
  COUNTIF(Coupon_Code IS NULL)           AS Coupon_Code_nulls,
  COUNTIF(Discount_pct IS NULL)          AS Discount_pct_nulls,
  COUNTIF(Payment_Method IS NULL)        AS Payment_Method_nulls,
  COUNTIF(Shipping_Provider IS NULL)     AS Shipping_Provider_nulls,
  COUNTIF(Transaction_Rating IS NULL)    AS Transaction_Rating_nulls
FROM `mis784t22025-466123.MIS784_A2.transaction_A2`;
# There are  no null values in all columns in the transaction table
```

- Result:



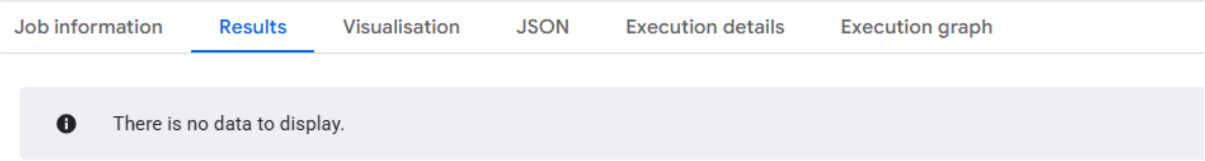
Row	Customer_I...	Transactio...	Transaction_...	Product_SKU_nulls	Product_Descr...	Product_Categor...	Quantity_nulls	Avg_Price...	Delivery_Charges...	Coupon_Status_n...	Coupon_Code_nu...	Discount_pct_nulls	Payment_Method...	Shipping_Provide...	Transaction_Rati...
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Question 1

- Query a:

```
-- Check if delivery fees were uniform within each transaction --
SELECT
  Transaction_ID,
  COUNT(DISTINCT Delivery_Charges) AS num_unique_delivery_charges,
  ARRAY_AGG(DISTINCT Delivery_Charges ORDER BY Delivery_Charges) AS
delivery_charge_values
FROM `mis784t22025-466123.MIS784_A2.transaction_A2`
GROUP BY Transaction_ID
HAVING COUNT(DISTINCT Delivery_Charges) > 1
ORDER BY num_unique_delivery_charges DESC;
# Each transaction uses only one delivery charge value
```

- Result:



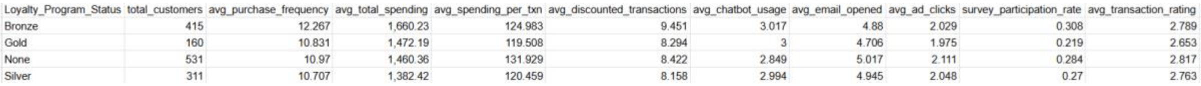
There is no data to display.

- Query b:

```
-- STEP 1: Aggregate product-line records to transaction-level summary --
```

```
WITH txn_level AS (
  SELECT
    Transaction_ID,
    Customer_ID,

    -- Total spend on products after discount per transaction
    SUM(IFNULL(Quantity, 0) * IFNULL(Avg_Price, 0) * (1 -
IFNULL(Discount_pct, 0) / 100)) AS net_product_spend,

    -- Delivery fee (same across all lines, keep only one value)
    MAX(IFNULL(Delivery_Charges, 0)) AS delivery_fee,

    -- Flag if discount used (either a discount percentage > 0 or coupon was
used)
    MAX(CASE
          WHEN IFNULL(Discount_pct, 0) > 0 OR Coupon_Status = 'Used' THEN 1
          ELSE 0
        END) AS used_discount_flag,

    -- Average rating across all items in a transaction
    AVG(Transaction_Rating) AS transaction_rating

  FROM `mis784t22025-466123.MIS784_A2.transaction_A2`
  GROUP BY Transaction_ID, Customer_ID
),

-- STEP 2: Aggregate transaction-level data to customer-level summary --
customer_level_txn AS (
  SELECT
    Customer_ID,

    -- Number of transactions made by customer
    COUNT(*) AS purchase_frequency,

    -- Total amount spent including delivery across all transactions
    SUM(net_product_spend + delivery_fee) AS total_spending,

    -- Average spend per transaction
    AVG(net_product_spend + delivery_fee) AS avg_spending_per_txn,

    -- Count of transactions where a discount was used
    SUM(used_discount_flag) AS transactions_with_discount,

    -- Average satisfaction rating across all transactions
    AVG(transaction_rating) AS avg_transaction_rating

  FROM txn_level
  GROUP BY Customer_ID
)

-- STEP 3: Join customer-level data with demographic info and summarise by
loyalty tier --
SELECT
  c.Loyalty_Program_Status,  -- Grouping variable: loyalty tier (e.g.,
Bronze, Silver)

  -- Number of customers in each loyalty tier
  COUNT(DISTINCT c.Customer_ID) AS total_customers,

  -- Behavioural and engagement metrics (rounded to 3 decimal places)
  ROUND(AVG(IFNULL(t.purchase_frequency, 0)), 3) AS avg_purchase_frequency,
  ROUND(AVG(IFNULL(t.total_spending, 0)), 3) AS avg_total_spending,
  ROUND(AVG(IFNULL(t.avg_spending_per_txn, 0)), 3) AS avg_spending_per_txn,
  ROUND(AVG(IFNULL(t.transactions_with_discount, 0)), 3) AS
avg_discounted_transactions,
  ROUND(AVG(c.Chatbot_Usage_Count), 3) AS avg_chatbot_usage,
  ROUND(AVG(c.Email_Opened_Count), 3) AS avg_email_opened,
  ROUND(AVG(c.Clicked_Ad_Campaigns), 3) AS avg_ad_clicks,
  ROUND(AVG(CAST(c.Participated_in_Survey AS INT64)), 3) AS
survey_participation_rate,
  ROUND(AVG(IFNULL(t.avg_transaction_rating, 0)), 3) AS
avg_transaction_rating

FROM `mis784t22025-466123.MIS784_A2.customer_A2_with_flag` AS c
LEFT JOIN customer_level_txn AS t
USING (Customer_ID)

GROUP BY c.Loyalty_Program_Status
ORDER BY c.Loyalty_Program_Status;
```

- Result (result exported to Google Sheet for better view)



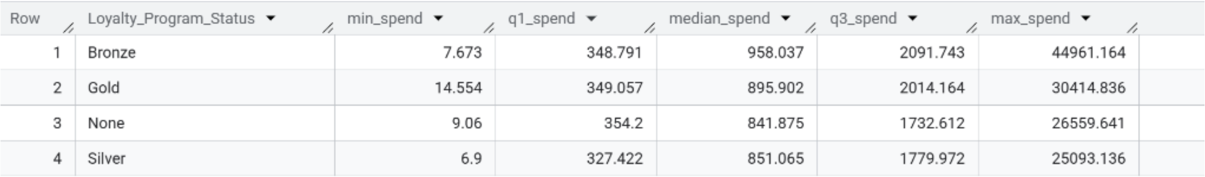
Loyalty_Program_Status	total_customers	avg_purchase_frequency	avg_total_spending	avg_spending_per_txn	avg_discounted_transactions	avg_chatbot_usage	avg_email_opened	avg_ad_clicks	survey_participation_rate	avg_transaction_rating
Bronze	415	12.267	1,660.23	124.983	9.451	3.017	4.88	2.029	0.308	2.789
Gold	160	10.831	1,472.19	119.508	8.294	3	4.706	1.975	0.219	2.653
None	531	10.97	1,460.36	131.929	8.422	2.849	5.017	2.111	0.284	2.817
Silver	311	10.707	1,382.42	120.459	8.158	2.994	4.945	2.048	0.27	2.763

- Query c: Distribution of total spend per customer by loyalty group

```
-- STEP 1: Aggregate product-line records to transaction-level summary --
```

```
WITH txn_level AS (
  SELECT
    Transaction_ID,
    Customer_ID,
    SUM(IFNULL(Quantity, 0) * IFNULL(Avg_Price, 0) * (1 -
IFNULL(Discount_pct, 0) / 100)) AS net_product_spend,
    MAX(IFNULL(Delivery_Charges, 0)) AS delivery_fee,
    MAX(CASE
          WHEN IFNULL(Discount_pct, 0) > 0 OR Coupon_Status = 'Used' THEN 1
          ELSE 0
        END) AS used_discount_flag,
    AVG(Transaction_Rating) AS transaction_rating
  FROM `mis784t22025-466123.MIS784_A2.transaction_A2`
  GROUP BY Transaction_ID, Customer_ID
),

-- STEP 2: Aggregate transaction-level data to customer-level summary --
customer_level_txn AS (
  SELECT
    Customer_ID,
    COUNT(*) AS purchase_frequency,
    SUM(net_product_spend + delivery_fee) AS total_spending,
    AVG(net_product_spend + delivery_fee) AS avg_spending_per_txn,
    SUM(used_discount_flag) AS transactions_with_discount,
    AVG(transaction_rating) AS avg_transaction_rating
  FROM txn_level
  GROUP BY Customer_ID
),

-- STEP 3: Join customer-level with loyalty data --
customer_with_loyalty AS (
  SELECT
    c.Loyalty_Program_Status,
    c.Customer_ID,
    t.total_spending
  FROM `mis784t22025-466123.MIS784_A2.customer_A2_with_flag` AS c
  LEFT JOIN customer_level_txn AS t
  USING (Customer_ID)
)

-- STEP 4: Explore quartiles of spending --
SELECT
  Loyalty_Program_Status,
  ROUND(APPROX_QUANTILES(total_spending, 4)[OFFSET(0)], 3) AS min_spend,
  ROUND(APPROX_QUANTILES(total_spending, 4)[OFFSET(1)], 3) AS q1_spend,
  ROUND(APPROX_QUANTILES(total_spending, 4)[OFFSET(2)], 3) AS median_spend,
  ROUND(APPROX_QUANTILES(total_spending, 4)[OFFSET(3)], 3) AS q3_spend,
  ROUND(APPROX_QUANTILES(total_spending, 4)[OFFSET(4)], 3) AS max_spend
FROM customer_with_loyalty
GROUP BY Loyalty_Program_Status
ORDER BY Loyalty_Program_Status;
```

- Result:



Row	Loyalty_Program_Status	min_spend	q1_spend	median_spend	q3_spend	max_spend
1	Bronze	7.673	348.791	958.037	2091.743	44961.164
2	Gold	14.554	349.057	895.902	2014.164	30414.836
3	None	9.06	354.2	841.875	1732.612	26559.641
4	Silver	6.9	327.422	851.065	1779.972	25093.136

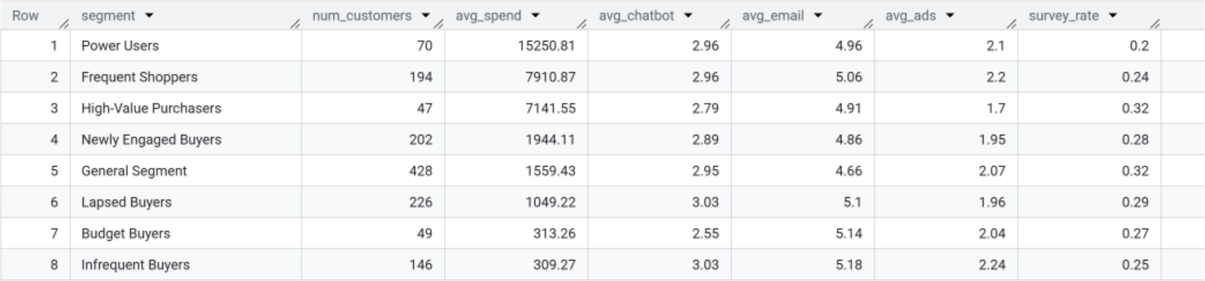
Question 2

- Query a:

```
-- In terms of RFM Segmentation --
-- STEP 1: Transaction-level aggregation --
WITH txn_level AS (
  SELECT
    Transaction_ID,
    Customer_ID,
    SUM(IFNULL(Quantity, 0) * IFNULL(Avg_Price, 0) * (1 -
IFNULL(Discount_pct, 0) / 100)) AS net_product_spend,
    MAX(IFNULL(Delivery_Charges, 0)) AS delivery_fee
  FROM `mis784t22025-466123.MIS784_A2.transaction_A2`
  GROUP BY Transaction_ID, Customer_ID
),

-- STEP 2: Customer-level aggregation --
customer_level_txn AS (
  SELECT
    t.Customer_ID,
    COUNT(*) AS frequency,
    SUM(net_product_spend + delivery_fee) AS monetary,
    MAX(a.Transaction_Date) AS last_purchase_date
  FROM txn_level t
  LEFT JOIN `mis784t22025-466123.MIS784_A2.transaction_A2` a
    ON t.Transaction_ID = a.Transaction_ID
  GROUP BY t.Customer_ID
),
-- STEP 3: Base RFM table --
rfm_base AS (
  SELECT
    Customer_ID,
    DATE_DIFF(DATE '2025-09-07', last_purchase_date, DAY) AS recency,
    frequency,
    ROUND(monetary, 2) AS monetary
  FROM customer_level_txn
),

-- STEP 4: RFM quintiles --
rfm_quintiles AS (
  SELECT
    Customer_ID,
    NTILE(5) OVER (ORDER BY recency DESC) AS r_quintile,
    NTILE(5) OVER (ORDER BY frequency ASC) AS f_quintile,
    NTILE(5) OVER (ORDER BY monetary ASC) AS m_quintile
  FROM rfm_base
),

-- STEP 5: Segment labeling --
rfm_segments AS (
  SELECT
    q.Customer_ID,
    r_quintile,
    f_quintile,
    m_quintile,
    b.monetary,
    CASE
      WHEN r_quintile = 5 AND f_quintile = 5 AND m_quintile = 5 THEN 'Power
Users'
      WHEN r_quintile = 5 THEN 'Newly Engaged Buyers'
      WHEN f_quintile = 5 THEN 'Frequent Shoppers'
      WHEN m_quintile = 5 THEN 'High-Value Purchasers'
      WHEN r_quintile = 1 THEN 'Lapsed Buyers'
      WHEN f_quintile = 1 THEN 'Infrequent Buyers'
      WHEN m_quintile = 1 THEN 'Budget Buyers'
      ELSE 'General Segment'
    END AS segment
  FROM rfm_quintiles q
  JOIN rfm_base b USING (Customer_ID)
),

-- STEP 6: Join with customer metadata --
```

```
rfm_full AS (
  SELECT
    r.Customer_ID,
    r.segment,
    r.monetary,
    c.Loyalty_Program_Status,
    c.Chatbot_Usage_Count,
    c.Email_Opened_Count,
    c.Clicked_Ad_Campaigns,
    c.Participated_in_Survey,
    c.Preferred_Channel
  FROM rfm_segments r
  LEFT JOIN `mis784t22025-466123.MIS784_A2.customer_A2_with_flag` c
  USING (Customer_ID)
)

-- STEP 7: Final summary by RFM segment --
SELECT
  segment,
  COUNT(*) AS num_customers,
  ROUND(AVG(monetary), 2) AS avg_spend,
  ROUND(AVG(Chatbot_Usage_Count), 2) AS avg_chatbot,
  ROUND(AVG(Email_Opened_Count), 2) AS avg_email,
  ROUND(AVG(Clicked_Ad_Campaigns), 2) AS avg_ads,
  ROUND(AVG(CAST(Participated_in_Survey AS INT64)), 2) AS survey_rate
FROM rfm_full
GROUP BY segment
ORDER BY avg_spend DESC;
```

- Result



Row	segment	num_customers	avg_spend	avg_chatbot	avg_email	avg_ads	survey_rate
1	Power Users	70	15250.81	2.96	4.96	2.1	0.2
2	Frequent Shoppers	194	7910.87	2.96	5.06	2.2	0.24
3	High-Value Purchasers	47	7141.55	2.79	4.91	1.7	0.32
4	Newly Engaged Buyers	202	1944.11	2.89	4.86	1.95	0.28
5	General Segment	428	1559.43	2.95	4.66	2.07	0.32
6	Lapsed Buyers	226	1049.22	3.03	5.1	1.96	0.29
7	Budget Buyers	49	313.26	2.55	5.14	2.04	0.27
8	Infrequent Buyers	146	309.27	3.03	5.18	2.24	0.25

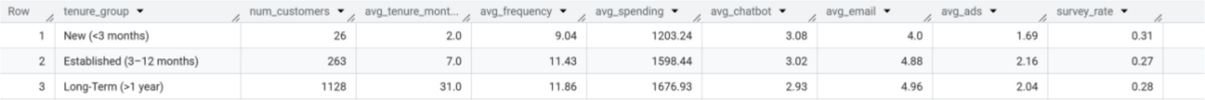
- Query b:

```
-- In terms of Tenure months --
-- STEP 1: Customer behaviour + tenure groups --
WITH customer_with_tenure_group AS (
  SELECT
```

```
    Customer_ID,
    CASE
      WHEN Tenure_Months < 3 THEN 'New (<3 months)'
      WHEN Tenure_Months < 12 THEN 'Established (3–12 months)'
      ELSE 'Long-Term (>1 year)'
    END AS tenure_group,
    Tenure_Months,
    Chatbot_Usage_Count,
    Email_Opened_Count,
    Clicked_Ad_Campaigns,
    Participated_in_Survey,
    Preferred_Channel
  FROM `mis784t22025-466123.MIS784_A2.customer_A2_with_flag`
),

-- STEP 2: Add purchase behaviour from transaction data --
txn_level AS (
  SELECT
    Customer_ID,
    COUNT(DISTINCT Transaction_ID) AS purchase_frequency,
    ROUND(SUM(IFNULL(Quantity, 0) * IFNULL(Avg_Price, 0) * (1 - IFNULL(Discount_pct, 0) / 100) + IFNULL(Delivery_Charges, 0)), 2) AS total_spending
  FROM `mis784t22025-466123.MIS784_A2.transaction_A2`
  GROUP BY Customer_ID
),

-- STEP 3: Combine both --
combined AS (
  SELECT
    c.tenure_group,
    c.Tenure_Months,
    t.purchase_frequency,
    t.total_spending,
    c.Chatbot_Usage_Count,
    c.Email_Opened_Count,
    c.Clicked_Ad_Campaigns,
    c.Participated_in_Survey
  FROM customer_with_tenure_group c
  LEFT JOIN txn_level t USING (Customer_ID)
)

-- FINAL STEP: Summary by tenure group --
SELECT
```

```
  tenure_group,
  COUNT(*) AS num_customers,
  ROUND(AVG(Tenure_Months), 1) AS avg_tenure_months,
  ROUND(AVG(purchase_frequency), 2) AS avg_frequency,
  ROUND(AVG(total_spending), 2) AS avg_spending,
  ROUND(AVG(Chatbot_Usage_Count), 2) AS avg_chatbot,
  ROUND(AVG(Email_Opened_Count), 2) AS avg_email,
  ROUND(AVG(Clicked_Ad_Campaigns), 2) AS avg_ads,
  ROUND(AVG(CAST(Participated_in_Survey AS INT64)), 2) AS survey_rate
FROM combined
GROUP BY tenure_group
ORDER BY avg_tenure_months;
```

- Result:



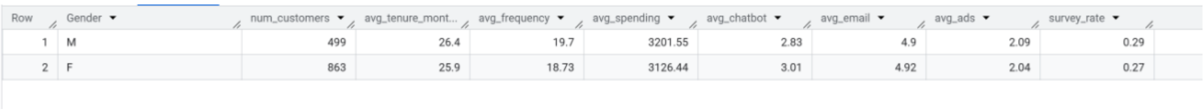
Row	tenure_group	num_customers	avg_tenure_mont...	avg_frequency	avg_spending	avg_chatbot	avg_email	avg_ads	survey_rate
1	New (<3 months)	26	2.0	9.04	1203.24	3.08	4.0	1.69	0.31
2	Established (3–12 months)	263	7.0	11.43	1598.44	3.02	4.88	2.16	0.27
3	Long-Term (>1 year)	1128	31.0	11.86	1676.93	2.93	4.96	2.04	0.28

- Query c:

```
-- In terms of gender --

WITH txn_level AS (
  SELECT
    Transaction_ID,
    Customer_ID,
    SUM(IFNULL(Quantity, 0) * IFNULL(Avg_Price, 0) * (1 -
IFNULL(Discount_pct, 0) / 100)) AS net_product_spend,
    MAX(IFNULL(Delivery_Charges, 0)) AS delivery_fee
  FROM `mis784t22025-466123.MIS784_A2.transaction_A2`
  GROUP BY Transaction_ID, Customer_ID
),

customer_level_txn AS (
  SELECT
    t.Customer_ID,
    COUNT(*) AS purchase_frequency,
    SUM(net_product_spend + delivery_fee) AS total_spending
  FROM txn_level t
  LEFT JOIN `mis784t22025-466123.MIS784_A2.transaction_A2` a
    ON t.Transaction_ID = a.Transaction_ID
  GROUP BY t.Customer_ID
)
```

```
SELECT
  c.Gender,
  COUNT(*) AS num_customers,
  ROUND(AVG(c.Tenure_Months), 1) AS avg_tenure_months,
  ROUND(AVG(purchase_frequency), 2) AS avg_frequency,
  ROUND(AVG(total_spending), 2) AS avg_spending,
  ROUND(AVG(Chatbot_Usage_Count), 2) AS avg_chatbot,
  ROUND(AVG(Email_Opened_Count), 2) AS avg_email,
  ROUND(AVG(Clicked_Ad_Campaigns), 2) AS avg_ads,
  ROUND(AVG(CAST(Participated_in_Survey AS INT64)), 2) AS survey_rate
FROM customer_level_txn t
JOIN `mis784t22025-466123.MIS784_A2.customer_A2_with_flag` c
  ON t.Customer_ID = c.Customer_ID
GROUP BY Gender
ORDER BY avg_spending DESC;
```

- Result:

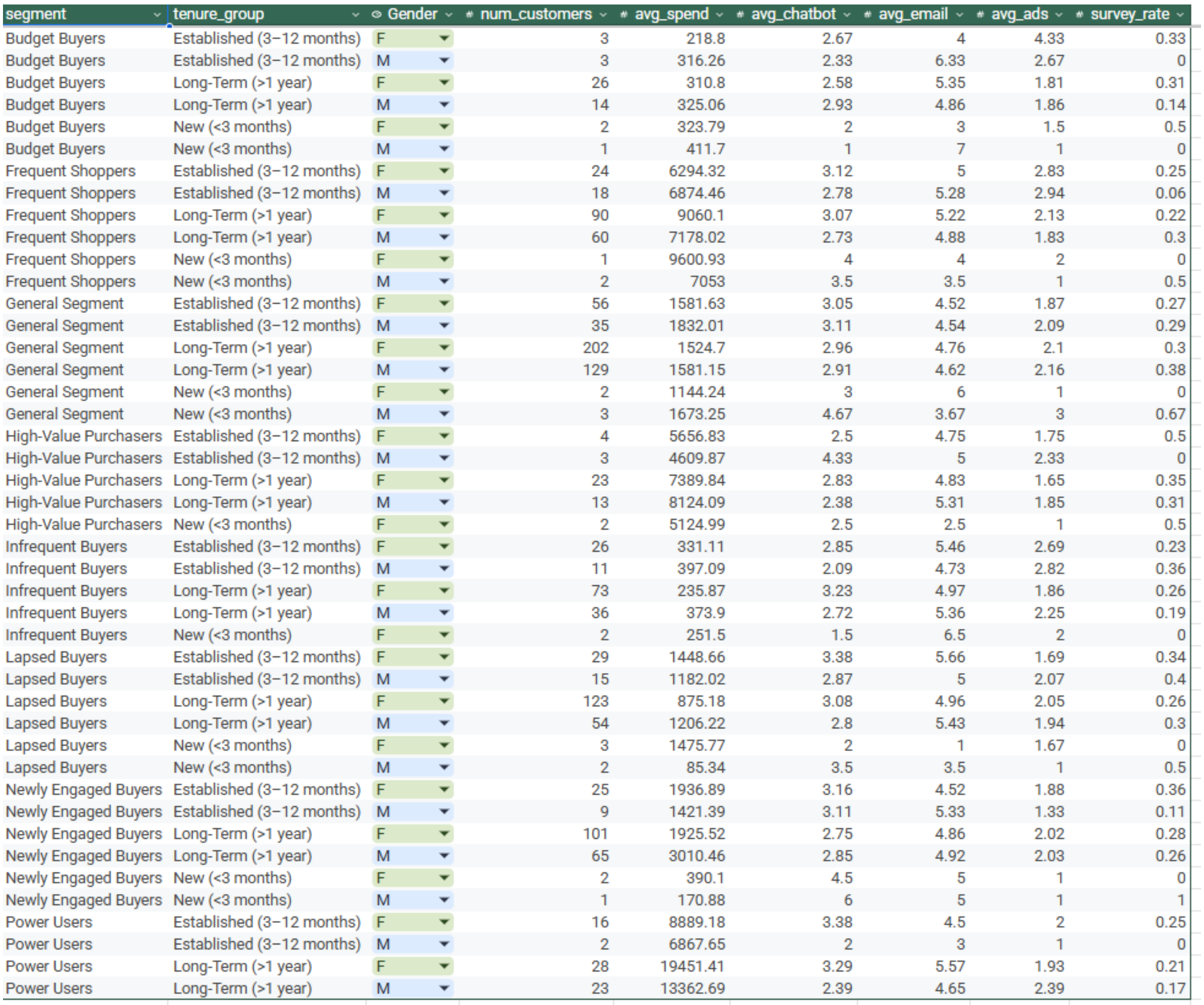


Row	Gender	num_customers	avg_tenure_mont...	avg_frequency	avg_spending	avg_chatbot	avg_email	avg_ads	survey_rate
1	M	499	26.4	19.7	3201.55	2.83	4.9	2.09	0.29
2	F	863	25.9	18.73	3126.44	3.01	4.92	2.04	0.27

- Query d:

```
-- Combining RFM, tenure, and gender --
-- STEP 1: Transaction-level aggregation --
WITH txn_level AS (
  SELECT
    Transaction_ID,
    Customer_ID,
    SUM(IFNULL(Quantity, 0) * IFNULL(Avg_Price, 0) * (1 -
IFNULL(Discount_pct, 0) / 100)) AS net_product_spend,
    MAX(IFNULL(Delivery_Charges, 0)) AS delivery_fee
  FROM `mis784t22025-466123.MIS784_A2.transaction_A2`
  GROUP BY Transaction_ID, Customer_ID
),

-- STEP 2: Customer-level aggregation --
customer_level_txn AS (
  SELECT
    t.Customer_ID,
    COUNT(*) AS frequency,
    SUM(net_product_spend + delivery_fee) AS monetary,
    MAX(a.Transaction_Date) AS last_purchase_date
  FROM txn_level t
  LEFT JOIN `mis784t22025-466123.MIS784_A2.transaction_A2` a
    ON t.Transaction_ID = a.Transaction_ID
  GROUP BY t.Customer_ID
),

-- STEP 3: RFM base table --
rfm_base AS (
  SELECT
    Customer_ID,
    DATE_DIFF(DATE '2025-09-07', last_purchase_date, DAY) AS recency,
    frequency,
    ROUND(monetary, 2) AS monetary
  FROM customer_level_txn
),

-- STEP 4: Calculate quintiles --
rfm_quintiles AS (
  SELECT
    Customer_ID,
    NTILE(5) OVER (ORDER BY recency DESC) AS r_quintile,
    NTILE(5) OVER (ORDER BY frequency ASC) AS f_quintile,
    NTILE(5) OVER (ORDER BY monetary ASC) AS m_quintile
  FROM rfm_base
),

-- STEP 5: Assign RFM segments --
rfm_segments AS (
  SELECT
    q.Customer_ID,
    r_quintile,
    f_quintile,
    m_quintile,
    b.monetary,
    CASE
      WHEN r_quintile = 5 AND f_quintile = 5 AND m_quintile = 5 THEN 'Power Users'
      WHEN r_quintile = 5 THEN 'Newly Engaged Buyers'
      WHEN f_quintile = 5 THEN 'Frequent Shoppers'
      WHEN m_quintile = 5 THEN 'High-Value Purchasers'
      WHEN r_quintile = 1 THEN 'Lapsed Buyers'
      WHEN f_quintile = 1 THEN 'Infrequent Buyers'
      WHEN m_quintile = 1 THEN 'Budget Buyers'
      ELSE 'General Segment'
    END AS segment
  FROM rfm_quintiles q
  JOIN rfm_base b USING (Customer_ID)
),

-- STEP 6: Join with tenure group and gender --
rfm_full AS (
  SELECT
    r.Customer_ID,
    r.segment,
    r.monetary,
    c.Tenure_Months,
    CASE
      WHEN c.Tenure_Months < 3 THEN 'New (<3 months)'
      WHEN c.Tenure_Months BETWEEN 3 AND 12 THEN 'Established (3–12 months)'
      ELSE 'Long-Term (>1 year)'
    END AS tenure_group,
    c.Gender,
    c.Loyalty_Program_Status,
    c.Chatbot_Usage_Count,
    c.Email_Opened_Count,
    c.Clicked_Ad_Campaigns,
    c.Participated_in_Survey,
    c.Preferred_Channel
  FROM rfm_segments r
  LEFT JOIN `mis784t22025-466123.MIS784_A2.customer_A2_with_flag` c
  USING (Customer_ID)
)

-- STEP 7: Final summary: RFM × Tenure × Gender --
SELECT
  segment,
  tenure_group,
  Gender,
  COUNT(*) AS num_customers,
  ROUND(AVG(monetary), 2) AS avg_spend,
  ROUND(AVG(Chatbot_Usage_Count), 2) AS avg_chatbot,
  ROUND(AVG(Email_Opened_Count), 2) AS avg_email,
  ROUND(AVG(Clicked_Ad_Campaigns), 2) AS avg_ads,
  ROUND(AVG(CAST(Participated_in_Survey AS INT64)), 2) AS survey_rate
FROM rfm_full
GROUP BY segment, tenure_group, Gender
ORDER BY segment, tenure_group, Gender;
```

- Result (exported to Google Sheet for better view)



segment	tenure_group	Gender	num_customers	avg_spend	avg_chatbot	avg_email	avg_ads	survey_rate
Budget Buyers	Established (3–12 months)	F	3	218.8	2.67	4	4.33	0.33
Budget Buyers	Established (3–12 months)	M	3	316.26	2.33	6.33	2.67	0
Budget Buyers	Long-Term (>1 year)	F	26	310.8	2.58	5.35	1.81	0.31
Budget Buyers	Long-Term (>1 year)	M	14	325.06	2.93	4.86	1.86	0.14
Budget Buyers	New (<3 months)	F	2	323.79	2	3	1.5	0.5
Budget Buyers	New (<3 months)	M	1	411.7	1	7	1	0
Frequent Shoppers	Established (3–12 months)	F	24	6294.32	3.12	5	2.83	0.25
Frequent Shoppers	Established (3–12 months)	M	18	6874.46	2.78	5.28	2.94	0.06
Frequent Shoppers	Long-Term (>1 year)	F	90	9060.1	3.07	5.22	2.13	0.22
Frequent Shoppers	Long-Term (>1 year)	M	60	7178.02	2.73	4.88	1.83	0.3
Frequent Shoppers	New (<3 months)	F	1	9600.93	4	4	2	0
Frequent Shoppers	New (<3 months)	M	2	7053	3.5	3.5	1	0.5
General Segment	Established (3–12 months)	F	56	1581.63	3.05	4.52	1.87	0.27
General Segment	Established (3–12 months)	M	35	1832.01	3.11	4.54	2.09	0.29
General Segment	Long-Term (>1 year)	F	202	1524.7	2.96	4.76	2.1	0.3
General Segment	Long-Term (>1 year)	M	129	1581.15	2.91	4.62	2.16	0.38
General Segment	New (<3 months)	F	2	1144.24	3	6	1	0
General Segment	New (<3 months)	M	3	1673.25	4.67	3.67	3	0.67
High-Value Purchasers	Established (3–12 months)	F	4	5656.83	2.5	4.75	1.75	0.5
High-Value Purchasers	Established (3–12 months)	M	3	4609.87	4.33	5	2.33	0
High-Value Purchasers	Long-Term (>1 year)	F	23	7389.84	2.83	4.83	1.65	0.35
High-Value Purchasers	Long-Term (>1 year)	M	13	8124.09	2.38	5.31	1.85	0.31
High-Value Purchasers	New (<3 months)	F	2	5124.99	2.5	2.5	1	0.5
Infrequent Buyers	Established (3–12 months)	F	26	331.11	2.85	5.46	2.69	0.23
Infrequent Buyers	Established (3–12 months)	M	11	397.09	2.09	4.73	2.82	0.36
Infrequent Buyers	Long-Term (>1 year)	F	73	235.87	3.23	4.97	1.86	0.26
Infrequent Buyers	Long-Term (>1 year)	M	36	373.9	2.72	5.36	2.25	0.19
Infrequent Buyers	New (<3 months)	F	2	251.5	1.5	6.5	2	0
Lapsed Buyers	Established (3–12 months)	F	29	1448.66	3.38	5.66	1.69	0.34
Lapsed Buyers	Established (3–12 months)	M	15	1182.02	2.87	5	2.07	0.4
Lapsed Buyers	Long-Term (>1 year)	F	123	875.18	3.08	4.96	2.05	0.26
Lapsed Buyers	Long-Term (>1 year)	M	54	1206.22	2.8	5.43	1.94	0.3
Lapsed Buyers	New (<3 months)	F	3	1475.77	2	1	1.67	0
Lapsed Buyers	New (<3 months)	M	2	85.34	3.5	3.5	1	0.5
Newly Engaged Buyers	Established (3–12 months)	F	25	1936.89	3.16	4.52	1.88	0.36
Newly Engaged Buyers	Established (3–12 months)	M	9	1421.39	3.11	5.33	1.33	0.11
Newly Engaged Buyers	Long-Term (>1 year)	F	101	1925.52	2.75	4.86	2.02	0.28
Newly Engaged Buyers	Long-Term (>1 year)	M	65	3010.46	2.85	4.92	2.03	0.26
Newly Engaged Buyers	New (<3 months)	F	2	390.1	4.5	5	1	0
Newly Engaged Buyers	New (<3 months)	M	1	170.88	6	5	1	1
Power Users	Established (3–12 months)	F	16	8889.18	3.38	4.5	2	0.25
Power Users	Established (3–12 months)	M	2	6867.65	2	3	1	0
Power Users	Long-Term (>1 year)	F	28	19451.41	3.29	5.57	1.93	0.21
Power Users	Long-Term (>1 year)	M	23	13362.69	2.39	4.65	2.39	0.17

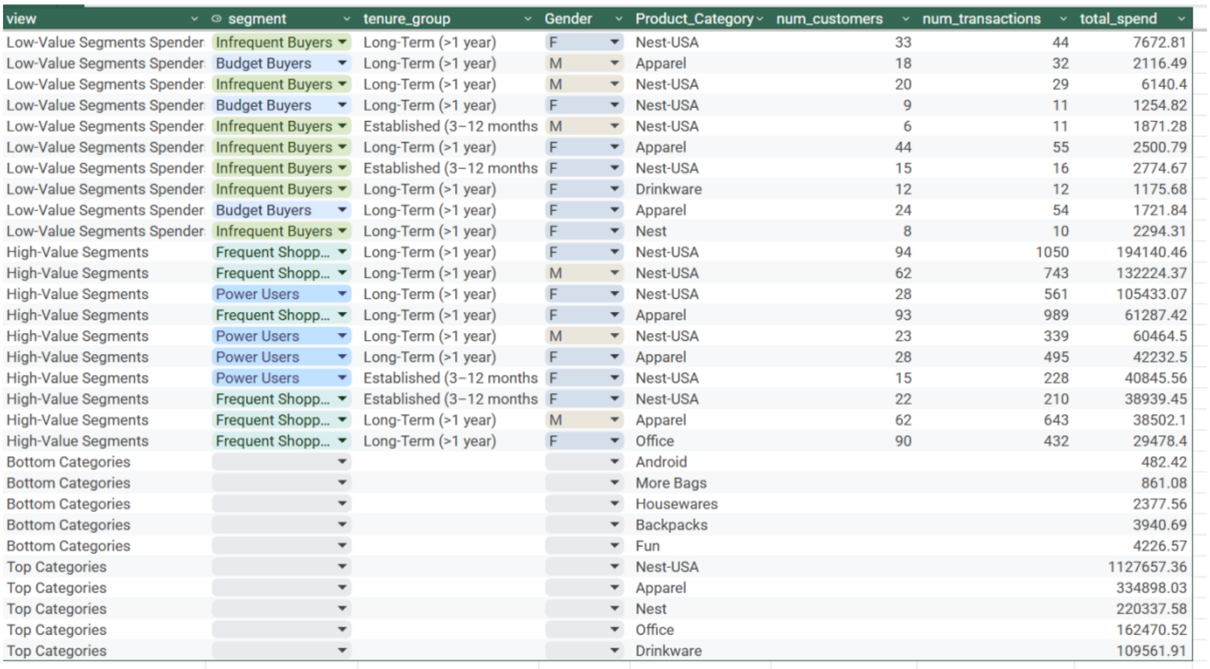
- Query e:

```
-- Investigating Product category w.r.t customer behaviour based on RFM,
tenure, and age --
WITH
-- STEP 1: Categorise customers
customer_segments AS (
  SELECT
    c.Customer_ID,
    CASE
      WHEN Tenure_Months < 3 THEN 'New (<3 months)'
      WHEN Tenure_Months < 12 THEN 'Established (3–12 months)'
      ELSE 'Long-Term (>1 year)'
    END AS tenure_group,
    Gender,
    CASE
```

```
      WHEN r_quintile = 5 AND f_quintile = 5 AND m_quintile = 5 THEN 'Power
Users'
      WHEN r_quintile = 5 THEN 'Newly Engaged Buyers'
      WHEN f_quintile = 5 THEN 'Frequent Shoppers'
      WHEN m_quintile = 5 THEN 'High-Value Purchasers'
      WHEN r_quintile = 1 THEN 'Lapsed Buyers'
      WHEN f_quintile = 1 THEN 'Infrequent Buyers'
      WHEN m_quintile = 1 THEN 'Budget Buyers'
      ELSE 'General Segment'
    END AS segment
  FROM `mis784t22025-466123.MIS784_A2.customer_A2_with_flag` c
  LEFT JOIN (
    SELECT
      Customer_ID,
      NTILE(5) OVER (ORDER BY DATE_DIFF(DATE '2025-09-07',
MAX(Transaction_Date), DAY) DESC) AS r_quintile,
      NTILE(5) OVER (ORDER BY COUNT(*) ASC) AS f_quintile,
      NTILE(5) OVER (ORDER BY SUM(
        IFNULL(Quantity, 0) * IFNULL(Avg_Price, 0) * (1 -
IFNULL(Discount_pct, 0) / 100) + IFNULL(Delivery_Charges, 0)
      ) ASC) AS m_quintile
    FROM `mis784t22025-466123.MIS784_A2.transaction_A2`
    GROUP BY Customer_ID
  ) rfm USING (Customer_ID)
),

-- STEP 2: Enrich transactions
txn_enriched AS (
  SELECT
    t.Customer_ID,
    t.Product_Category,
    t.Transaction_ID,
    IFNULL(Quantity, 0) * IFNULL(Avg_Price, 0) * (1 - IFNULL(Discount_pct,
0) / 100) + IFNULL(Delivery_Charges, 0) AS net_spend
  FROM `mis784t22025-466123.MIS784_A2.transaction_A2` t
),

-- STEP 3: Join segments and summarise
final_summary AS (
  SELECT
    s.segment,
    s.tenure_group,
    s.Gender,
    t.Product_Category,
    COUNT(DISTINCT t.Customer_ID) AS num_customers,
    COUNT(DISTINCT t.Transaction_ID) AS num_transactions,
    ROUND(SUM(t.net_spend), 2) AS total_spend
  FROM txn_enriched t
  JOIN customer_segments s USING (Customer_ID)
  GROUP BY s.segment, s.tenure_group, s.Gender, t.Product_Category
),

-- Top 5 categories overall
top_categories AS (
  SELECT
    'Top Categories' AS view,
    CAST(NULL AS STRING) AS segment,
    CAST(NULL AS STRING) AS tenure_group,
    CAST(NULL AS STRING) AS Gender,
    Product_Category,
    CAST(NULL AS INT64) AS num_customers,
    CAST(NULL AS INT64) AS num_transactions,
    ROUND(SUM(total_spend), 2) AS total_spend
  FROM final_summary
  GROUP BY Product_Category
  ORDER BY total_spend DESC
  LIMIT 5
),

-- Bottom 5 categories overall
bottom_categories AS (
  SELECT
    'Bottom Categories' AS view,
    CAST(NULL AS STRING) AS segment,
    CAST(NULL AS STRING) AS tenure_group,
    CAST(NULL AS STRING) AS Gender,
    Product_Category,
    CAST(NULL AS INT64) AS num_customers,
    CAST(NULL AS INT64) AS num_transactions,
    ROUND(SUM(total_spend), 2) AS total_spend
  FROM final_summary
  GROUP BY Product_Category
  ORDER BY total_spend ASC
  LIMIT 5
),

-- Top 10 transactions by Power Users / Frequent Shoppers in Established or
Long-Term tenure
```

```
high_value_segments AS (
  SELECT
    'High-Value Segments' AS view,
    segment,
    tenure_group,
    Gender,
    Product_Category,
    num_customers,
    num_transactions,
    total_spend
  FROM final_summary
  WHERE segment IN ('Power Users', 'Frequent Shoppers')
    AND tenure_group IN ('Established (3–12 months)', 'Long-Term (>1 year)')
  ORDER BY total_spend DESC
  LIMIT 10
),

-- Budget / Infrequent Buyers spending a lot (possible anomalies or upsell success)
low_value_segments_high_spend AS (
  SELECT
    'Low-Value Segments Spenders' AS view,
    segment,
    tenure_group,
    Gender,
    Product_Category,
    num_customers,
    num_transactions,
    total_spend
  FROM final_summary
  WHERE segment IN ('Budget Buyers', 'Infrequent Buyers')
    AND total_spend > 1000
  ORDER BY total_spend DESC
)

-- FINAL UNION OUTPUT
SELECT * FROM top_categories
UNION ALL
SELECT * FROM bottom_categories
UNION ALL
SELECT * FROM high_value_segments
UNION ALL
SELECT * FROM low_value_segments_high_spend;
```

- Result (exported to Google Sheet for better view)



view	segment	tenure_group	Gender	Product_Category	num_customers	num_transactions	total_spend
Low-Value Segments Spender	Infrequent Buyers	Long-Term (>1 year)	F	Nest-USA	33	44	7672.81
Low-Value Segments Spender	Budget Buyers	Long-Term (>1 year)	M	Apparel	18	32	2116.49
Low-Value Segments Spender	Infrequent Buyers	Long-Term (>1 year)	M	Nest-USA	20	29	6140.4
Low-Value Segments Spender	Budget Buyers	Long-Term (>1 year)	F	Nest-USA	9	11	1254.82
Low-Value Segments Spender	Infrequent Buyers	Established (3–12 months	M	Nest-USA	6	11	1871.28
Low-Value Segments Spender	Infrequent Buyers	Long-Term (>1 year)	F	Apparel	44	55	2500.79
Low-Value Segments Spender	Infrequent Buyers	Established (3–12 months	F	Nest-USA	15	16	2774.67
Low-Value Segments Spender	Infrequent Buyers	Long-Term (>1 year)	F	Drinkware	12	12	1175.68
Low-Value Segments Spender	Budget Buyers	Long-Term (>1 year)	F	Apparel	24	54	1721.84
Low-Value Segments Spender	Infrequent Buyers	Long-Term (>1 year)	F	Nest	8	10	2294.31
High-Value Segments	Frequent Shopp...	Long-Term (>1 year)	F	Nest-USA	94	1050	194140.46
High-Value Segments	Frequent Shopp...	Long-Term (>1 year)	M	Nest-USA	62	743	132224.37
High-Value Segments	Power Users	Long-Term (>1 year)	F	Nest-USA	28	561	105433.07
High-Value Segments	Frequent Shopp...	Long-Term (>1 year)	F	Apparel	93	989	61287.42
High-Value Segments	Power Users	Long-Term (>1 year)	M	Nest-USA	23	339	60464.5
High-Value Segments	Power Users	Long-Term (>1 year)	F	Apparel	28	495	42232.5
High-Value Segments	Power Users	Established (3–12 months	F	Nest-USA	15	228	40845.56
High-Value Segments	Frequent Shopp...	Established (3–12 months	F	Nest-USA	22	210	38939.45
High-Value Segments	Frequent Shopp...	Long-Term (>1 year)	M	Apparel	62	643	38502.1
High-Value Segments	Frequent Shopp...	Long-Term (>1 year)	F	Office	90	432	29478.4
Bottom Categories				Android			482.42
Bottom Categories				More Bags			861.08
Bottom Categories				Housewares			2377.56
Bottom Categories				Backpacks			3940.69
Bottom Categories				Fun			4226.57
Top Categories				Nest-USA			1127657.36
Top Categories				Apparel			334898.03
Top Categories				Nest			220337.58
Top Categories				Office			162470.52
Top Categories				Drinkware			109561.91