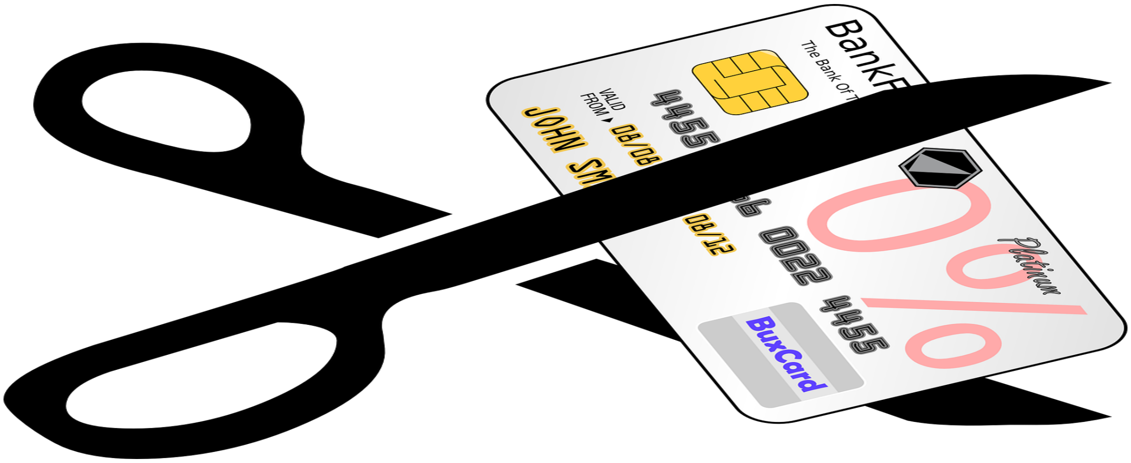


The Bank Project

May, 13th 2023

By Craig Schlachter

Key Findings

I analyzed data on 9,480 customers to identify users that are about to churn. My findings suggest that the following customers are at a higher risk of churning:

- Customers who spend less
- Customers who transact infrequently
- Customers who are dormant
- Female customers
- Middle-aged customers (41-55)
- Customers with low card utilization (0 - 0.333)
- Customers making less than $40,000 a year

These findings have implications for our bank that wants to reduce customer churn. The bank should focus on retaining customers who are at risk of churning by offering them discounts, rewards, or other incentives to keep them engaged.

The limitations of this study includes the fact that it is third party data and we noticed an unusually high amount of customers have been on the books for 36 months. We can't assess if this is a strange outlier or maybe a product promotion the bank ran for a card product.

For future research, I would recommend conducting a study with a larger sample size and collecting more data on the customers who are at risk of churning. This would help to better understand the factors that contribute to customer churn and develop more effective strategies for retaining customers.

. . .

Step 1: Ask

Here, we clarify the problem we are trying to solve, and the objectives we are trying to meet.

1.1 Background

A local bank has contacted us to help find solutions for their customer relations department. The bank manager is concerned about customer churn and wants to predict who is most likely to leave so they can take action to retain them.

The bank manager is confident that an in-depth analysis of their 10,000 banking customers would reveal more opportunities for customer service initiatives.

1.2 Business Task

Analyze a banking dataset consisting of 10,000 unique records to gain insights into how customer demographics, spending habits, and risk affects customer churn rates. Create a customer segmentation model that can be used to predict future churn rates and develop strategies to retain them.

1.3 Business Objectives

- What are the average churn rates for customers with RFM scores from 1-5?
- Who are the most common customer segments?
- What is each customer segments' average monthly RFM values?
- What segments are most likely to churn, and what are their characterizations?

1.4 Deliverables

- A clear summary of the business task
- A description of all data sources used
- Documentation of any cleaning or manipulation of data
- A summary of analysis
- Supporting visualizations and key findings
- High-level content recommendations based on the analysis

1.5 Key Stakeholders

- The bank's CEO, who is the leader of the bank and in charge of making financial lives better for their customers.
- The bank manager, who oversees the day-to-day operations of the branch, supervises staff and works to keep and attract new customers.
- The customer relations department, who are in charge of building and maintaining customer relationships.

. . .

Step 2: Prepare

In the Prepare phase, we assess the data and its limitations.

2.1 Information on Data Source

1. Data is publicly available on Kaggle: Credit Card Customers and stored in 1 csv file.
2. This dataset was collected from a website Leaps Analyttica, a website dedicated to learning data science.
3. This dataset consists of 10,000 unique records.
4. Data collected includes age, gender, marital status, income, education, credit limit, average utilization ratio, and churn.

2.2 Limitations of Data Set

- The data was collected at least 2 years ago in 2021. The data covers customers' demographics, spending habits, credit limits, average utilization ratio, revolving balance, average accounts open, and etc.
- It is a small sample size. There are only 10,000 customers in the dataset. This is a small number compared to the millions of credit card customers in the United States.
- This data is from a third party so we cannot ascertain its integrity or accuracy.

2.3 Is Data ROCCC?

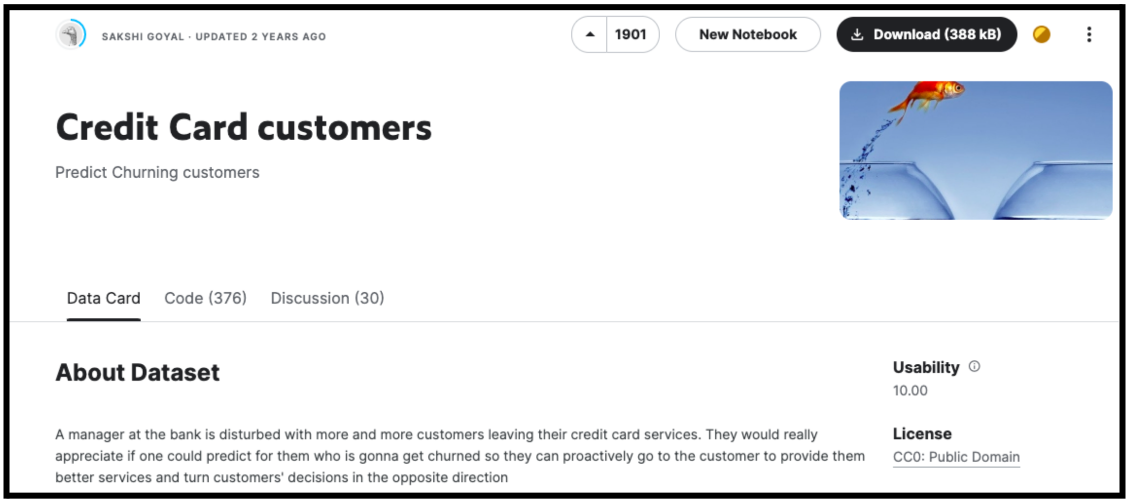
A good data source is ROCCC which stands for Reliable, Original, Comprehensive, Current, and Cited.

- Reliable - MED - Not highly reliable as it only has 10,000 records
- Original - LOW - Third Party Provider (Kaggle: Credit Card Customers)
- Comprehensive - MED - Parameters are sufficient to solve business task
- Current - MED - The data is 2 years old and may not be as relevant
- Cited - LOW - The data was collected from a third party, hence unknown

Overall, due to the limitations of this dataset, it is not recommended to use it to produce business intelligence recommendations.

2.4 Data Selection

The following file is downloaded and then imported into our created SQL table 'bank_cleaning'.



Credit Card customers

Predict Churning customers

About Dataset

A manager at the bank is disturbed with more and more customers leaving their credit card services. They would really appreciate if one could predict for them who is gonna get churned so they can proactively go to the customer to provide them better services and turn customers' decisions in the opposite direction

Usability
10.00

License
CC0: Public Domain

2.5 Tools

We are using SQL for data-wrangling and exploratory data analysis. Finally, we are using Tableau for visualizations.

. . .

Step 3: Process

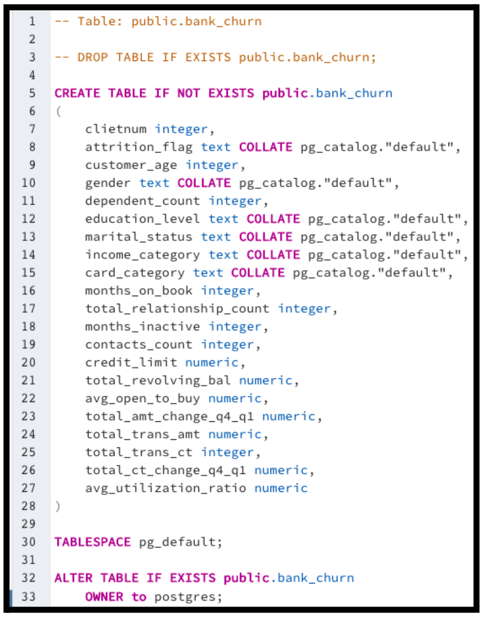
Here, we will process the data to ensure it is clean, correct, relevant, complete and free of errors and outliers by performing:

- Explore and observe the dataset
- Check for and handle any missing values
- Check for and remove any duplicate rows

- Ensure data is input and formatted correctly
- Check for and handle any outlier values
- Save cleaned data to a new file

3.1 Preparing the environment

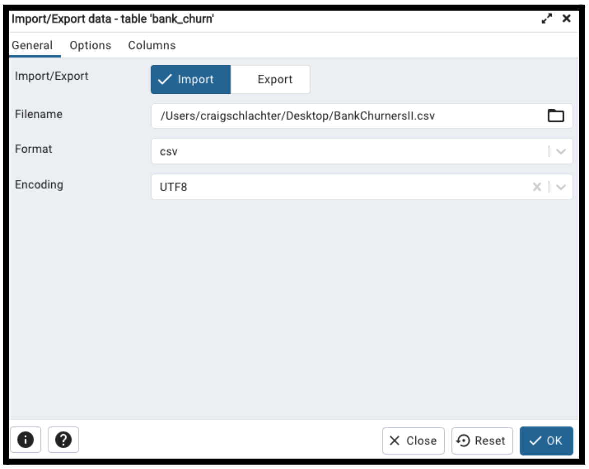
The SQL table is created, columns are named, and data types are set.



```
-- Table: public.bank_churn

-- DROP TABLE IF EXISTS public.bank_churn;

CREATE TABLE IF NOT EXISTS public.bank_churn
(
    clietnum integer,
    attrition_flag text COLLATE pg_catalog."default",
    customer_age integer,
    gender text COLLATE pg_catalog."default",
    dependent_count integer,
    education_level text COLLATE pg_catalog."default",
    marital_status text COLLATE pg_catalog."default",
    income_category text COLLATE pg_catalog."default",
    card_category text COLLATE pg_catalog."default",
    months_on_book integer,
    total_relationship_count integer,
    months_inactive integer,
    contacts_count integer,
    credit_limit numeric,
    total_revolving_bal numeric,
    avg_open_to_buy numeric,
    total_amt_change_q4_q1 numeric,
    total_trans_amt numeric,
    total_trans_ct integer,
    total_ct_change_q4_q1 numeric,
    avg_utilization_ratio numeric
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.bank_churn
    OWNER to postgres;
```

3.2 Importing Dataset

Reading in the selected file.



Import/Export data - table 'bank_churn'

Import/Export: ✓ Import | Export

Filename: /Users/craigschlachter/Desktop/BankChurnersII.csv

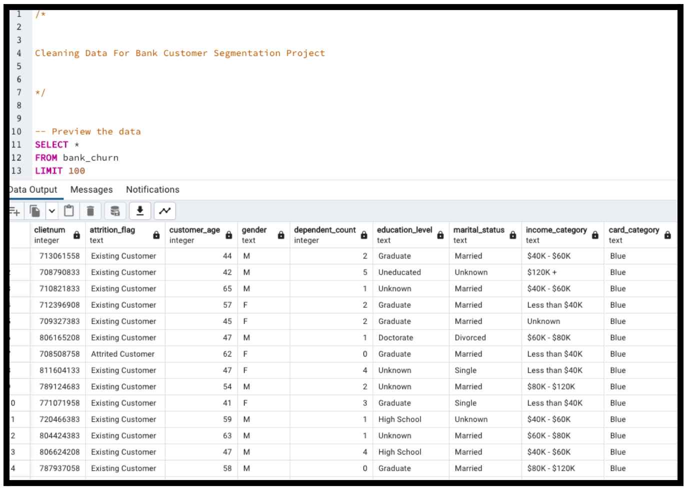
Format: csv

Encoding: UTF8

3.2 Data Cleaning and Manipulation

1. Observe and familiarize with data
2. Check for nulls or missing values
3. Check for and remove duplicates
4. Check for and remove outliers
5. Perform validation checks of data

Previewing the first 100 rows to familiarize with the data.

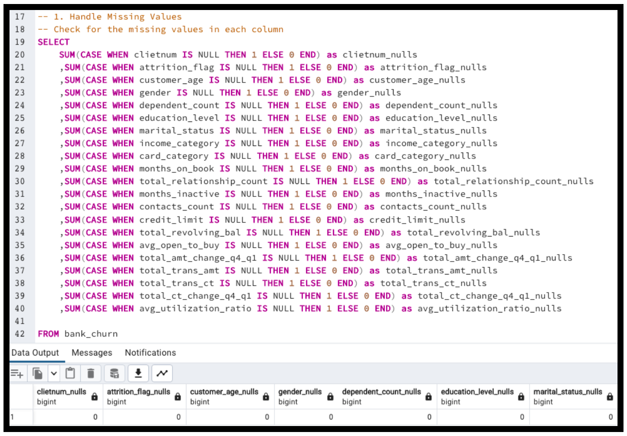


```sql
/*


Cleaning Data For Bank Customer Segmentation Project


*/


-- Preview the data
SELECT *
FROM bank_churn
LIMIT 100
```

clietnum integer	attrition_flag text	customer_age integer	gender text	dependent_count integer	education_level text	marital_status text	income_category text	card_category text
713061558	Existing Customer	44	M	2	Graduate	Married	$40K - $60K	Blue
708790833	Existing Customer	42	M	5	Uneducated	Unknown	$120K +	Blue
710821833	Existing Customer	65	M	1	Unknown	Married	$40K - $60K	Blue
712396908	Existing Customer	57	F	2	Graduate	Married	Less than $40K	Blue
709327383	Existing Customer	45	F	2	Graduate	Married	Unknown	Blue
806165208	Existing Customer	47	M	1	Doctorate	Divorced	$60K - $80K	Blue
708508758	Attrited Customer	62	F	0	Graduate	Married	Less than $40K	Blue
811604133	Existing Customer	47	F	4	Unknown	Single	Less than $40K	Blue
789124683	Existing Customer	54	M	2	Unknown	Married	$80K - $120K	Blue
771071958	Existing Customer	41	F	3	Graduate	Single	Less than $40K	Blue
720466383	Existing Customer	59	M	1	High School	Unknown	$40K - $60K	Blue
804424383	Existing Customer	63	M	1	Unknown	Married	$60K - $80K	Blue
806624208	Existing Customer	47	M	4	High School	Married	$40K - $60K	Blue
787937058	Existing Customer	58	M	0	Graduate	Married	$80K - $120K	Blue

Our initial impression of the dataset is that it is well-structured, organized, and formatted in a way that is suitable for the business task at hand.

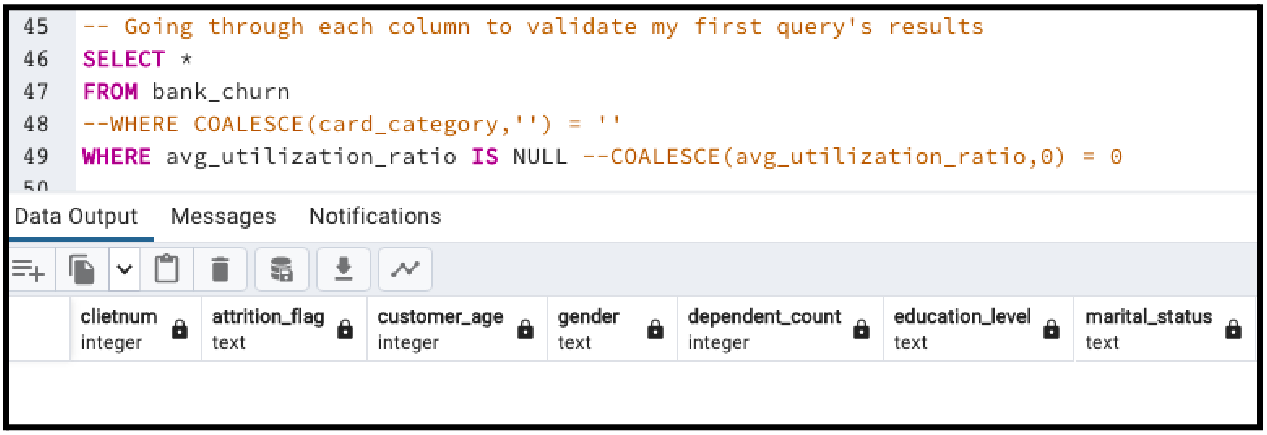
Now, we will check for nulls or missing values.



```sql
-- 1. Handle Missing Values
-- Check for the missing values in each column
SELECT
    SUM(CASE WHEN clietnum IS NULL THEN 1 ELSE 0 END) as clietnum_nulls
    ,SUM(CASE WHEN attrition_flag IS NULL THEN 1 ELSE 0 END) as attrition_flag_nulls
    ,SUM(CASE WHEN customer_age IS NULL THEN 1 ELSE 0 END) as customer_age_nulls
    ,SUM(CASE WHEN gender IS NULL THEN 1 ELSE 0 END) as gender_nulls
    ,SUM(CASE WHEN dependent_count IS NULL THEN 1 ELSE 0 END) as dependent_count_nulls
    ,SUM(CASE WHEN education_level IS NULL THEN 1 ELSE 0 END) as education_level_nulls
    ,SUM(CASE WHEN marital_status IS NULL THEN 1 ELSE 0 END) as marital_status_nulls
    ,SUM(CASE WHEN income_category IS NULL THEN 1 ELSE 0 END) as income_category_nulls
    ,SUM(CASE WHEN card_category IS NULL THEN 1 ELSE 0 END) as card_category_nulls
    ,SUM(CASE WHEN months_on_book IS NULL THEN 1 ELSE 0 END) as months_on_book_nulls
    ,SUM(CASE WHEN total_relationship_count IS NULL THEN 1 ELSE 0 END) as total_relationship_count_nulls
    ,SUM(CASE WHEN months_inactive IS NULL THEN 1 ELSE 0 END) as months_inactive_nulls
    ,SUM(CASE WHEN contacts_count IS NULL THEN 1 ELSE 0 END) as contacts_count_nulls
    ,SUM(CASE WHEN credit_limit IS NULL THEN 1 ELSE 0 END) as credit_limit_nulls
    ,SUM(CASE WHEN total_revolving_bal IS NULL THEN 1 ELSE 0 END) as total_revolving_bal_nulls
    ,SUM(CASE WHEN avg_open_to_buy IS NULL THEN 1 ELSE 0 END) as avg_open_to_buy_nulls
    ,SUM(CASE WHEN total_amt_change_q4_q1 IS NULL THEN 1 ELSE 0 END) as total_amt_change_q4_q1_nulls
    ,SUM(CASE WHEN total_trans_amt IS NULL THEN 1 ELSE 0 END) as total_trans_amt_nulls
    ,SUM(CASE WHEN total_trans_ct IS NULL THEN 1 ELSE 0 END) as total_trans_ct_nulls
    ,SUM(CASE WHEN total_ct_change_q4_q1 IS NULL THEN 1 ELSE 0 END) as total_ct_change_q4_q1_nulls
    ,SUM(CASE WHEN avg_utilization_ratio IS NULL THEN 1 ELSE 0 END) as avg_utilization_ratio_nulls

FROM bank_churn
```

	clietnum_nulls bigint	attrition_flag_nulls bigint	customer_age_nulls bigint	gender_nulls bigint	dependent_count_nulls bigint	education_level_nulls bigint	marital_status_nulls bigint
1	0	0	0	0	0	0	0

All rows were absent of nulls or missing values. Now, we will validate each individual column for missing values or nulls to validate our above query.

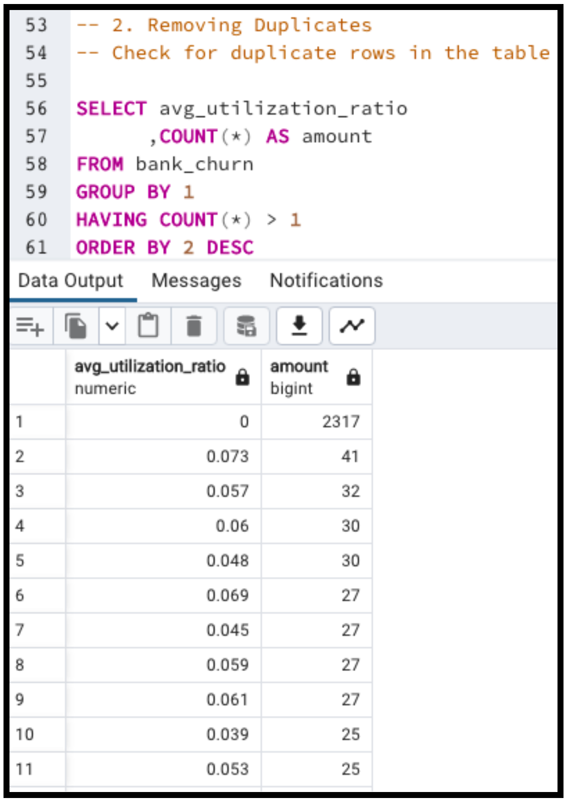


```sql
-- Going through each column to validate my first query's results
SELECT *
FROM bank_churn
--WHERE COALESCE(card_category,'') = ''
WHERE avg_utilization_ratio IS NULL --COALESCE(avg_utilization_ratio,0) = 0
```

clietnum integer	attrition_flag text	customer_age integer	gender text	dependent_count integer	education_level text	marital_status text

Our validation process confirms our initial query's findings.

Next, I am checking for and removing any duplicates column by column.

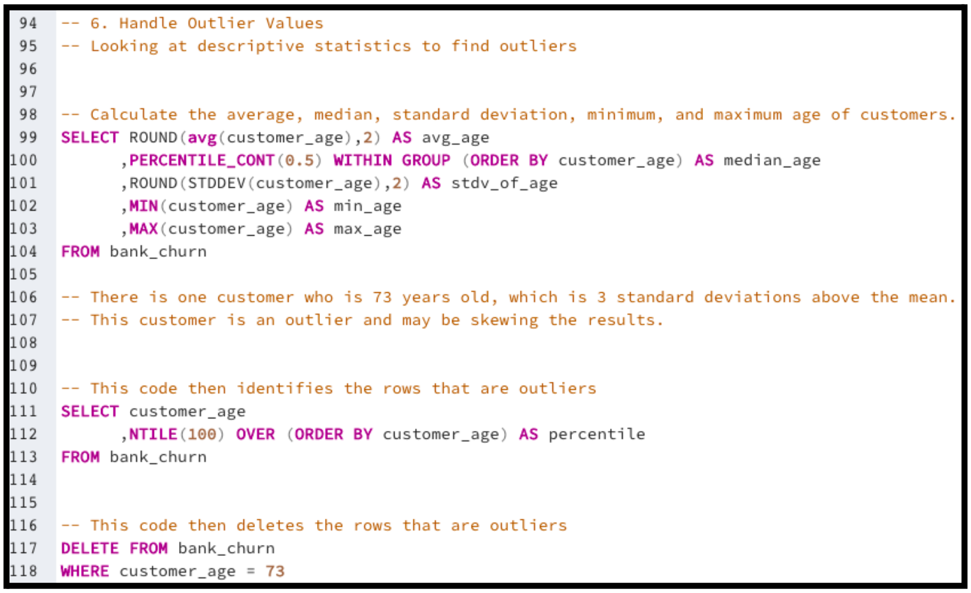


```sql
-- 2. Removing Duplicates
-- Check for duplicate rows in the table

SELECT avg_utilization_ratio
      ,COUNT(*) AS amount
FROM bank_churn
GROUP BY 1
HAVING COUNT(*) > 1
ORDER BY 2 DESC
```

	avg_utilization_ratio numeric	amount bigint
1	0	2317
2	0.073	41
3	0.057	32
4	0.06	30
5	0.048	30
6	0.069	27
7	0.045	27
8	0.059	27
9	0.061	27
10	0.039	25
11	0.053	25

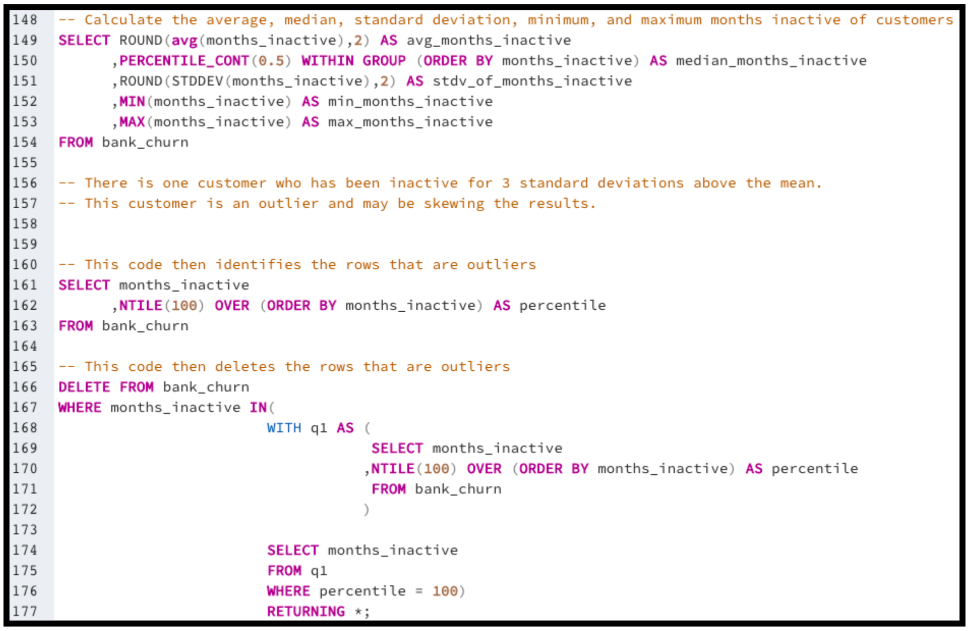
After reviewing the results for each column, I did not find any duplicates that appeared to be unnatural for this type of dataset.

Now, I am checking for and removing outliers.



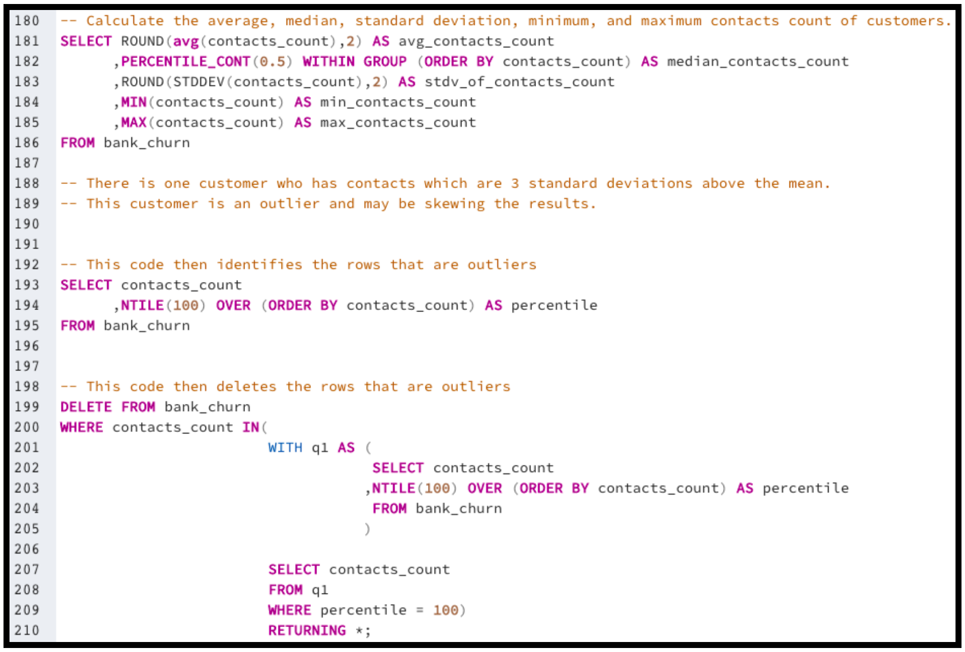
```sql
-- 6. Handle Outlier Values
-- Looking at descriptive statistics to find outliers


-- Calculate the average, median, standard deviation, minimum, and maximum age of customers.
SELECT ROUND(avg(customer_age),2) AS avg_age
      ,PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY customer_age) AS median_age
      ,ROUND(STDDEV(customer_age),2) AS stdv_of_age
      ,MIN(customer_age) AS min_age
      ,MAX(customer_age) AS max_age
FROM bank_churn

-- There is one customer who is 73 years old, which is 3 standard deviations above the mean.
-- This customer is an outlier and may be skewing the results.


-- This code then identifies the rows that are outliers
SELECT customer_age
      ,NTILE(100) OVER (ORDER BY customer_age) AS percentile
FROM bank_churn


-- This code then deletes the rows that are outliers
DELETE FROM bank_churn
WHERE customer_age = 73
```

We identified an outlier in the “customer_age” column and removed it.



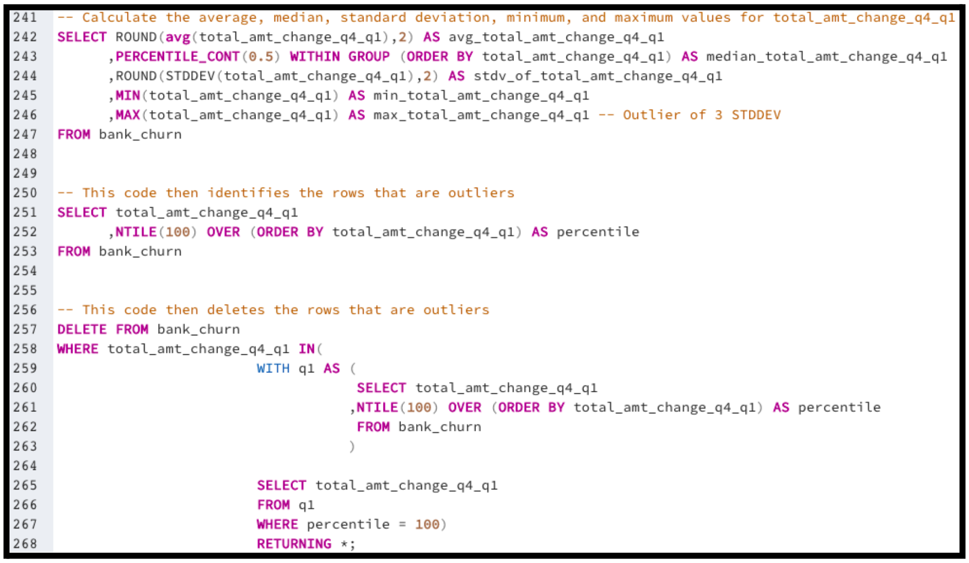
```sql
-- Calculate the average, median, standard deviation, minimum, and maximum months inactive of customers
SELECT ROUND(avg(months_inactive),2) AS avg_months_inactive
      ,PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY months_inactive) AS median_months_inactive
      ,ROUND(STDDEV(months_inactive),2) AS stdv_of_months_inactive
      ,MIN(months_inactive) AS min_months_inactive
      ,MAX(months_inactive) AS max_months_inactive
FROM bank_churn

-- There is one customer who has been inactive for 3 standard deviations above the mean.
-- This customer is an outlier and may be skewing the results.


-- This code then identifies the rows that are outliers
SELECT months_inactive
      ,NTILE(100) OVER (ORDER BY months_inactive) AS percentile
FROM bank_churn

-- This code then deletes the rows that are outliers
DELETE FROM bank_churn
WHERE months_inactive IN(
                        WITH q1 AS (
                                    SELECT months_inactive
                                   ,NTILE(100) OVER (ORDER BY months_inactive) AS percentile
                                    FROM bank_churn
                                   )

                        SELECT months_inactive
                        FROM q1
                        WHERE percentile = 100)
                        RETURNING *;
```

We identified an outlier in the “months_inactive” column and removed it.



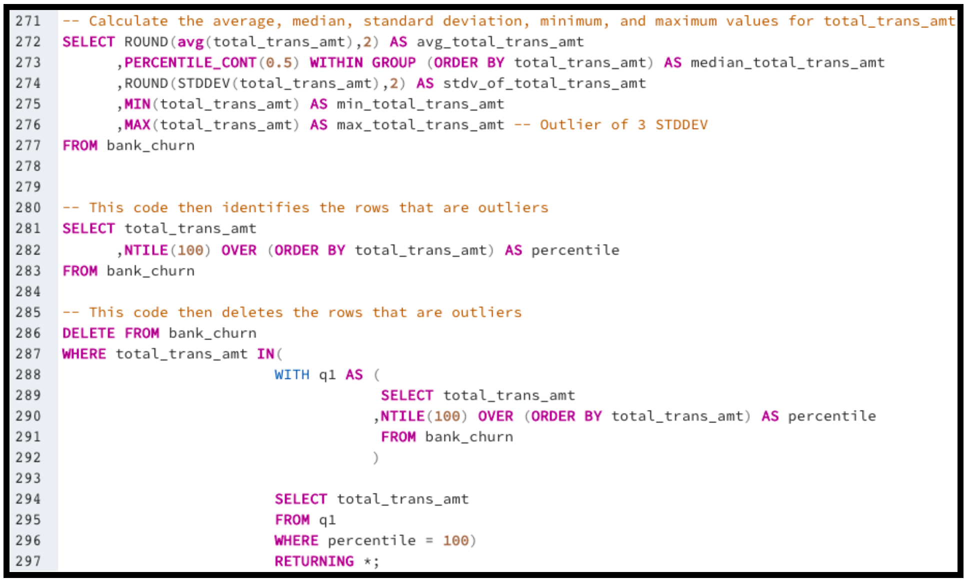
```sql
-- Calculate the average, median, standard deviation, minimum, and maximum contacts count of customers.
SELECT ROUND(avg(contacts_count),2) AS avg_contacts_count
      ,PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY contacts_count) AS median_contacts_count
      ,ROUND(STDDEV(contacts_count),2) AS stdv_of_contacts_count
      ,MIN(contacts_count) AS min_contacts_count
      ,MAX(contacts_count) AS max_contacts_count
FROM bank_churn

-- There is one customer who has contacts which are 3 standard deviations above the mean.
-- This customer is an outlier and may be skewing the results.


-- This code then identifies the rows that are outliers
SELECT contacts_count
      ,NTILE(100) OVER (ORDER BY contacts_count) AS percentile
FROM bank_churn


-- This code then deletes the rows that are outliers
DELETE FROM bank_churn
WHERE contacts_count IN(
                        WITH q1 AS (
                                    SELECT contacts_count
                                   ,NTILE(100) OVER (ORDER BY contacts_count) AS percentile
                                    FROM bank_churn
                                   )

                        SELECT contacts_count
                        FROM q1
                        WHERE percentile = 100)
                        RETURNING *;
```

We identified an outlier in the “contacts_count” column and removed it.



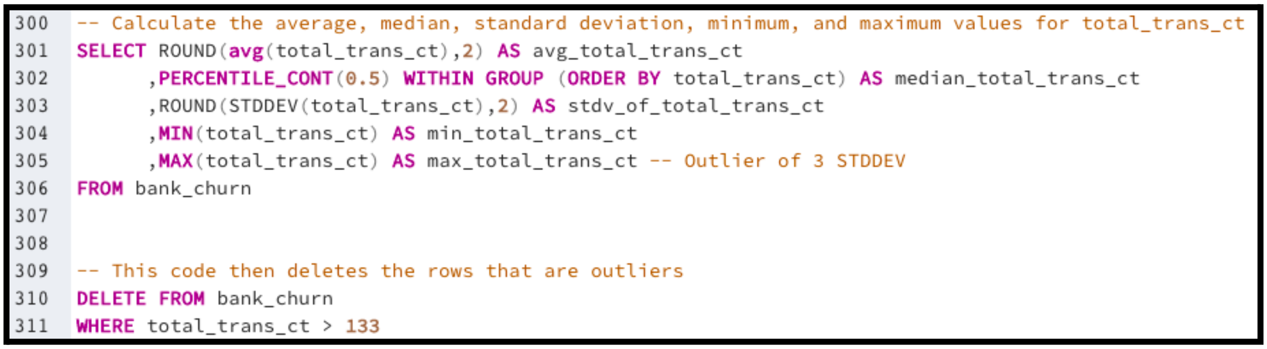
```sql
-- Calculate the average, median, standard deviation, minimum, and maximum values for total_amt_change_q4_q1
SELECT ROUND(avg(total_amt_change_q4_q1),2) AS avg_total_amt_change_q4_q1
      ,PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY total_amt_change_q4_q1) AS median_total_amt_change_q4_q1
      ,ROUND(STDDEV(total_amt_change_q4_q1),2) AS stdv_of_total_amt_change_q4_q1
      ,MIN(total_amt_change_q4_q1) AS min_total_amt_change_q4_q1
      ,MAX(total_amt_change_q4_q1) AS max_total_amt_change_q4_q1 -- Outlier of 3 STDDEV
FROM bank_churn


-- This code then identifies the rows that are outliers
SELECT total_amt_change_q4_q1
      ,NTILE(100) OVER (ORDER BY total_amt_change_q4_q1) AS percentile
FROM bank_churn


-- This code then deletes the rows that are outliers
DELETE FROM bank_churn
WHERE total_amt_change_q4_q1 IN(
                        WITH q1 AS (
                                    SELECT total_amt_change_q4_q1
                                   ,NTILE(100) OVER (ORDER BY total_amt_change_q4_q1) AS percentile
                                    FROM bank_churn
                                   )

                        SELECT total_amt_change_q4_q1
                        FROM q1
                        WHERE percentile = 100)
                        RETURNING *;
```

We identified an outlier in the “total_amt_change_q4_q1” column and removed it.



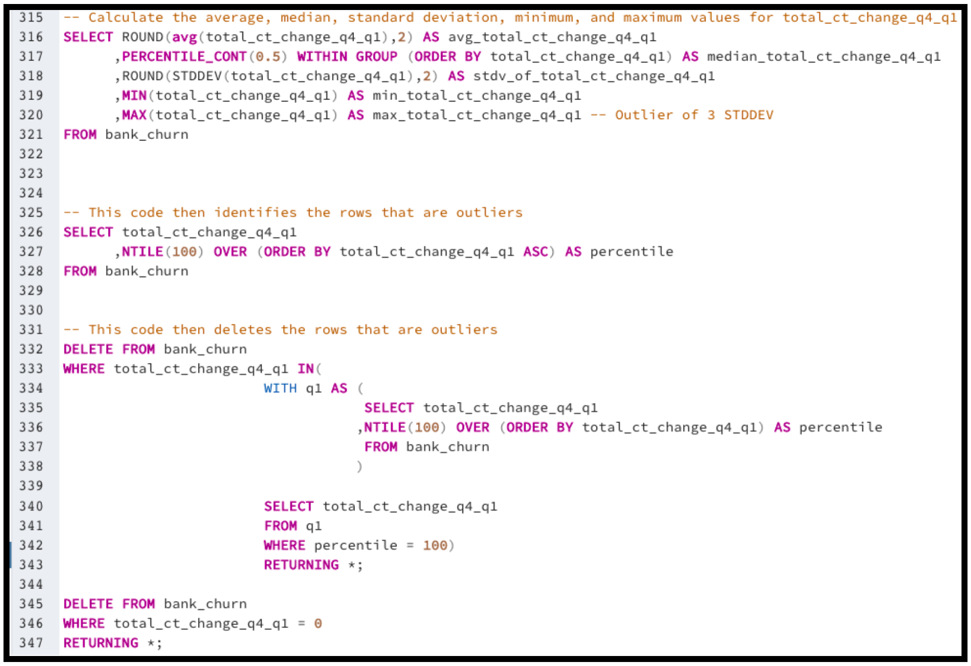
```
-- Calculate the average, median, standard deviation, minimum, and maximum values for total_trans_amt
SELECT ROUND(avg(total_trans_amt),2) AS avg_total_trans_amt
      ,PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY total_trans_amt) AS median_total_trans_amt
      ,ROUND(STDDEV(total_trans_amt),2) AS stdv_of_total_trans_amt
      ,MIN(total_trans_amt) AS min_total_trans_amt
      ,MAX(total_trans_amt) AS max_total_trans_amt -- Outlier of 3 STDDEV
FROM bank_churn


-- This code then identifies the rows that are outliers
SELECT total_trans_amt
      ,NTILE(100) OVER (ORDER BY total_trans_amt) AS percentile
FROM bank_churn

-- This code then deletes the rows that are outliers
DELETE FROM bank_churn
WHERE total_trans_amt IN(
                        WITH q1 AS (
                                    SELECT total_trans_amt
                                   ,NTILE(100) OVER (ORDER BY total_trans_amt) AS percentile
                                    FROM bank_churn
                                   )

                        SELECT total_trans_amt
                        FROM q1
                        WHERE percentile = 100)
                        RETURNING *;
```

We identified an outlier in the “total_trans_amt” column and removed it.



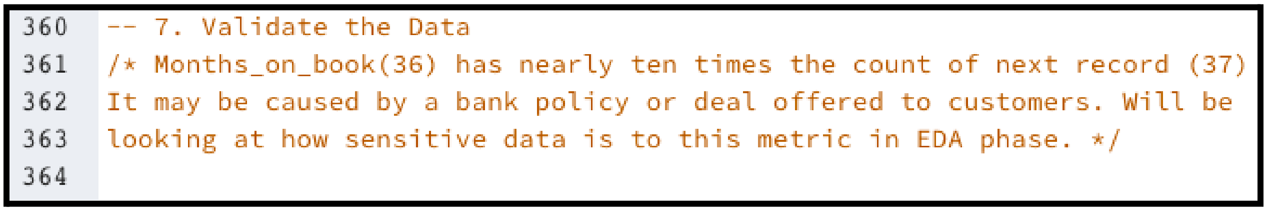
```
-- Calculate the average, median, standard deviation, minimum, and maximum values for total_trans_ct
SELECT ROUND(avg(total_trans_ct),2) AS avg_total_trans_ct
      ,PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY total_trans_ct) AS median_total_trans_ct
      ,ROUND(STDDEV(total_trans_ct),2) AS stdv_of_total_trans_ct
      ,MIN(total_trans_ct) AS min_total_trans_ct
      ,MAX(total_trans_ct) AS max_total_trans_ct -- Outlier of 3 STDDEV
FROM bank_churn


-- This code then deletes the rows that are outliers
DELETE FROM bank_churn
WHERE total_trans_ct > 133
```

We identified an outlier in the “total_trans_ct” column and removed it.



```sql
-- Calculate the average, median, standard deviation, minimum, and maximum values for total_ct_change_q4_q1
SELECT ROUND(avg(total_ct_change_q4_q1),2) AS avg_total_ct_change_q4_q1
      ,PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY total_ct_change_q4_q1) AS median_total_ct_change_q4_q1
      ,ROUND(STDDEV(total_ct_change_q4_q1),2) AS stdv_of_total_ct_change_q4_q1
      ,MIN(total_ct_change_q4_q1) AS min_total_ct_change_q4_q1
      ,MAX(total_ct_change_q4_q1) AS max_total_ct_change_q4_q1 -- Outlier of 3 STDDEV
FROM bank_churn



-- This code then identifies the rows that are outliers
SELECT total_ct_change_q4_q1
      ,NTILE(100) OVER (ORDER BY total_ct_change_q4_q1 ASC) AS percentile
FROM bank_churn


-- This code then deletes the rows that are outliers
DELETE FROM bank_churn
WHERE total_ct_change_q4_q1 IN(
                        WITH q1 AS (
                                    SELECT total_ct_change_q4_q1
                                   ,NTILE(100) OVER (ORDER BY total_ct_change_q4_q1) AS percentile
                                    FROM bank_churn
                                   )

                        SELECT total_ct_change_q4_q1
                        FROM q1
                        WHERE percentile = 100)
                        RETURNING *;

DELETE FROM bank_churn
WHERE total_ct_change_q4_q1 = 0
RETURNING *;
```

We identified an outlier in the “total__ct_change_q4_q1” column and removed it.



```sql
-- 7. Validate the Data
/* Months_on_book(36) has nearly ten times the count of next record (37)
It may be caused by a bank policy or deal offered to customers. Will be
looking at how sensitive data is to this metric in EDA phase. */

```

We identified another outlier in the “months_on_book” column while we were validating the data and will determine its sensitivity during the EDA phase.

Now that our data is cleaned and saved, we will move onto the next phase to explore and analyze the data.

. . .

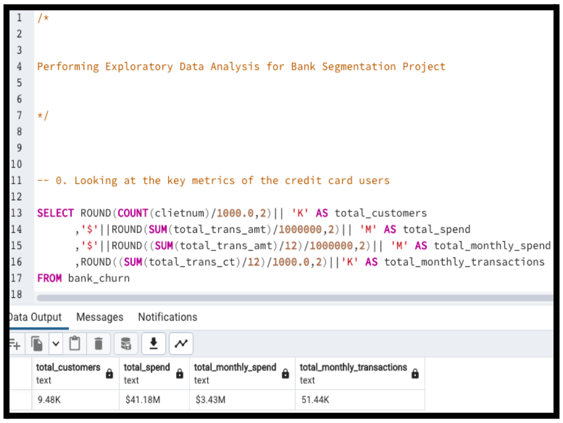
Step 4: Analyze

4.1 Performing Calculations

Pulling statistics for analysis:

1. Examine and select the most significant metrics for bank customers
2. Create customer segments to identify customers at risk of churning
3. Build an RFM model to improve customer retention
4. Compute the average recency, frequency, and monetary value for each customer segment by month.
5. Find the percentage of customers who fall into each demographic segment and have certain spending habits within each RFM customer segment.
6. Calculate the average churn rate for each unique combination of recency and frequency/monetary.

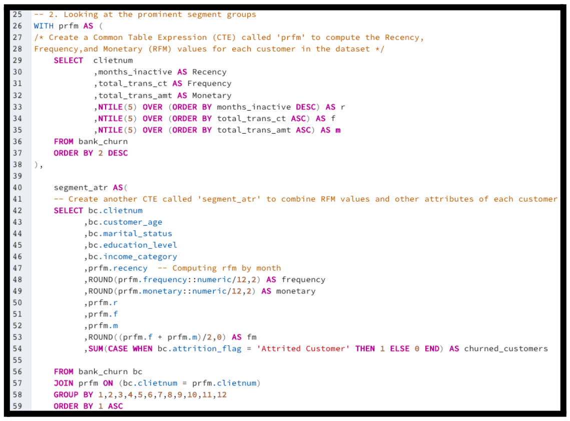
[Step 1's Results]



```
/*


Performing Exploratory Data Analysis for Bank Segmentation Project


*/



-- 0. Looking at the key metrics of the credit card users

SELECT ROUND(COUNT(clietnum)/1000.0,2)|| 'K' AS total_customers
      ,'$'||ROUND(SUM(total_trans_amt)/1000000,2)|| 'M' AS total_spend
      ,'$'||ROUND((SUM(total_trans_amt)/12)/1000000,2)|| 'M' AS total_monthly_spend
      ,ROUND((SUM(total_trans_ct)/12)/1000.0,2)||'K' AS total_monthly_transactions
FROM bank_churn

```

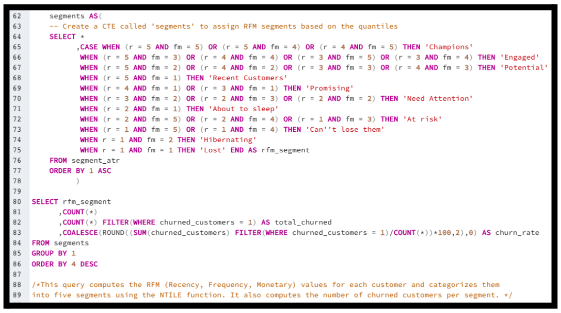
	total_customers	total_spend	total_monthly_spend	total_monthly_transactions
	text	text	text	text
	9.48K	$41.18M	$3.43M	51.44K

[Step 2's Results]



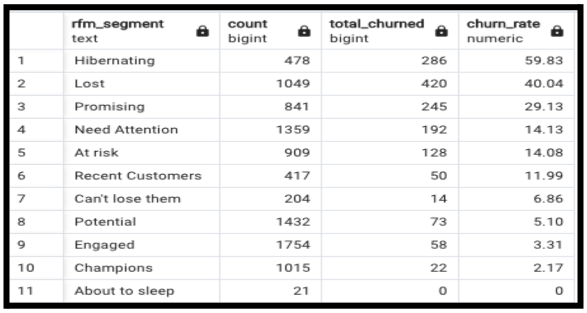
```
-- 2. Looking at the prominent segment groups
WITH prfm AS (
/* Create a Common Table Expression (CTE) called 'prfm' to compute the Recency,
Frequency,and Monetary (RFM) values for each customer in the dataset */
    SELECT  clietnum
            ,months_inactive AS Recency
            ,total_trans_ct AS Frequency
            ,total_trans_amt AS Monetary
            ,NTILE(5) OVER (ORDER BY months_inactive DESC) AS r
            ,NTILE(5) OVER (ORDER BY total_trans_ct ASC) AS f
            ,NTILE(5) OVER (ORDER BY total_trans_amt ASC) AS m
    FROM bank_churn
    ORDER BY 2 DESC
),

    segment_atr AS(
    -- Create another CTE called 'segment_atr' to combine RFM values and other attributes of each customer
    SELECT bc.clietnum
          ,bc.customer_age
          ,bc.marital_status
          ,bc.education_level
          ,bc.income_category
          ,prfm.recency  -- Computing rfm by month
          ,ROUND(prfm.frequency::numeric/12,2) AS frequency
          ,ROUND(prfm.monetary::numeric/12,2) AS monetary
          ,prfm.r
          ,prfm.f
          ,prfm.m
          ,ROUND((prfm.f + prfm.m)/2,0) AS fm
          ,SUM(CASE WHEN bc.attrition_flag = 'Attrited Customer' THEN 1 ELSE 0 END) AS churned_customers

    FROM bank_churn bc
    JOIN prfm ON (bc.clietnum = prfm.clietnum)
    GROUP BY 1,2,3,4,5,6,7,8,9,10,11,12
    ORDER BY 1 ASC
```

[Step 2's Results]



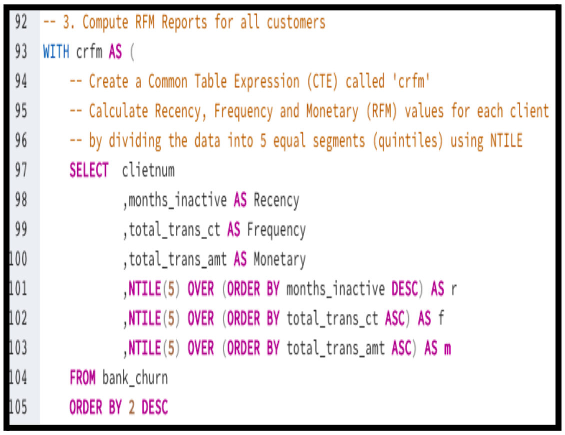
```
    segments AS(
    -- Create a CTE called 'segments' to assign RFM segments based on the quantiles
    SELECT *
          ,CASE WHEN (r = 5 AND fm = 5) OR (r = 5 AND fm = 4) OR (r = 4 AND fm = 5) THEN 'Champions'
           WHEN (r = 5 AND fm = 3) OR (r = 4 AND fm = 4) OR (r = 3 AND fm = 5) OR (r = 3 AND fm = 4) THEN 'Engaged'
           WHEN (r = 5 AND fm = 2) OR (r = 4 AND fm = 2) OR (r = 3 AND fm = 3) OR (r = 4 AND fm = 3) THEN 'Potential'
           WHEN (r = 5 AND fm = 1) THEN 'Recent Customers'
           WHEN (r = 4 AND fm = 1) OR (r = 3 AND fm = 1) THEN 'Promising'
           WHEN (r = 3 AND fm = 2) OR (r = 2 AND fm = 3) OR (r = 2 AND fm = 2) THEN 'Need Attention'
           WHEN (r = 2 AND fm = 1) THEN 'About to sleep'
           WHEN (r = 2 AND fm = 5) OR (r = 2 AND fm = 4) OR (r = 1 AND fm = 3) THEN 'At risk'
           WHEN (r = 1 AND fm = 5) OR (r = 1 AND fm = 4) THEN 'Can''t lose them'
           WHEN r = 1 AND fm = 2 THEN 'Hibernating'
           WHEN r = 1 AND fm = 1 THEN 'Lost' END AS rfm_segment
    FROM segment_atr
    ORDER BY 1 ASC
          )

SELECT rfm_segment
      ,COUNT(*)
      ,COUNT(*) FILTER(WHERE churned_customers = 1) AS total_churned
      ,COALESCE(ROUND((SUM(churned_customers) FILTER(WHERE churned_customers = 1)/COUNT(*))*100,2),0) AS churn_rate
FROM segments
GROUP BY 1
ORDER BY 4 DESC

/*This query computes the RFM (Recency, Frequency, Monetary) values for each customer and categorizes them
into five segments using the NTILE function. It also computes the number of churned customers per segment. */
```

[Step 2's Results]



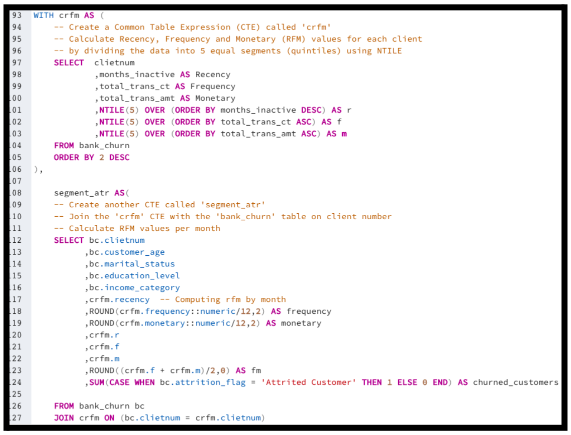
	rfm_segment	count	total_churned	churn_rate
	text	bigint	bigint	numeric
1	Hibernating	478	286	59.83
2	Lost	1049	420	40.04
3	Promising	841	245	29.13
4	Need Attention	1359	192	14.13
5	At risk	909	128	14.08
6	Recent Customers	417	50	11.99
7	Can't lose them	204	14	6.86
8	Potential	1432	73	5.10
9	Engaged	1754	58	3.31
10	Champions	1015	22	2.17
11	About to sleep	21	0	0

[Step 3’s Results]



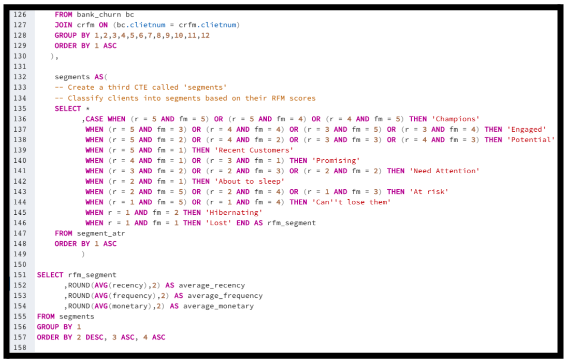
```sql
-- 3. Compute RFM Reports for all customers
WITH crfm AS (
    -- Create a Common Table Expression (CTE) called 'crfm'
    -- Calculate Recency, Frequency and Monetary (RFM) values for each client
    -- by dividing the data into 5 equal segments (quintiles) using NTILE
    SELECT  clietnum
            ,months_inactive AS Recency
            ,total_trans_ct AS Frequency
            ,total_trans_amt AS Monetary
            ,NTILE(5) OVER (ORDER BY months_inactive DESC) AS r
            ,NTILE(5) OVER (ORDER BY total_trans_ct ASC) AS f
            ,NTILE(5) OVER (ORDER BY total_trans_amt ASC) AS m
    FROM bank_churn
    ORDER BY 2 DESC
```

[Step 4’s Results]



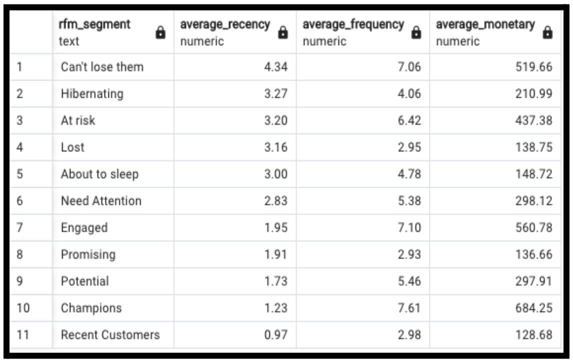
```sql
WITH crfm AS (
    -- Create a Common Table Expression (CTE) called 'crfm'
    -- Calculate Recency, Frequency and Monetary (RFM) values for each client
    -- by dividing the data into 5 equal segments (quintiles) using NTILE
    SELECT  clietnum
            ,months_inactive AS Recency
            ,total_trans_ct AS Frequency
            ,total_trans_amt AS Monetary
            ,NTILE(5) OVER (ORDER BY months_inactive DESC) AS r
            ,NTILE(5) OVER (ORDER BY total_trans_ct ASC) AS f
            ,NTILE(5) OVER (ORDER BY total_trans_amt ASC) AS m
    FROM bank_churn
    ORDER BY 2 DESC
),

    segment_atr AS(
    -- Create another CTE called 'segment_atr'
    -- Join the 'crfm' CTE with the 'bank_churn' table on client number
    -- Calculate RFM values per month
    SELECT bc.clietnum
          ,bc.customer_age
          ,bc.marital_status
          ,bc.education_level
          ,bc.income_category
          ,crfm.recency  -- Computing rfm by month
          ,ROUND(crfm.frequency::numeric/12,2) AS frequency
          ,ROUND(crfm.monetary::numeric/12,2) AS monetary
          ,crfm.r
          ,crfm.f
          ,crfm.m
          ,ROUND((crfm.f + crfm.m)/2,0) AS fm
          ,SUM(CASE WHEN bc.attrition_flag = 'Attrited Customer' THEN 1 ELSE 0 END) AS churned_customers

    FROM bank_churn bc
    JOIN crfm ON (bc.clietnum = crfm.clietnum)
```

[Step 4’s Results]



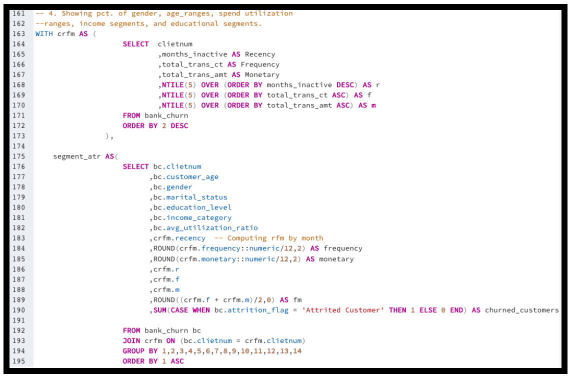
```sql
    FROM bank_churn bc
    JOIN crfm ON (bc.clietnum = crfm.clietnum)
    GROUP BY 1,2,3,4,5,6,7,8,9,10,11,12
    ORDER BY 1 ASC
   ),

    segments AS(
    -- Create a third CTE called 'segments'
    -- Classify clients into segments based on their RFM scores
    SELECT *
          ,CASE WHEN (r = 5 AND fm = 5) OR (r = 5 AND fm = 4) OR (r = 4 AND fm = 5) THEN 'Champions'
           WHEN (r = 5 AND fm = 3) OR (r = 4 AND fm = 4) OR (r = 3 AND fm = 5) OR (r = 3 AND fm = 4) THEN 'Engaged'
           WHEN (r = 5 AND fm = 2) OR (r = 4 AND fm = 2) OR (r = 3 AND fm = 3) OR (r = 4 AND fm = 3) THEN 'Potential'
           WHEN (r = 5 AND fm = 1) THEN 'Recent Customers'
           WHEN (r = 4 AND fm = 1) OR (r = 3 AND fm = 1) THEN 'Promising'
           WHEN (r = 3 AND fm = 2) OR (r = 2 AND fm = 3) OR (r = 2 AND fm = 2) THEN 'Need Attention'
           WHEN (r = 2 AND fm = 1) THEN 'About to sleep'
           WHEN (r = 2 AND fm = 5) OR (r = 2 AND fm = 4) OR (r = 1 AND fm = 3) THEN 'At risk'
           WHEN (r = 1 AND fm = 5) OR (r = 1 AND fm = 4) THEN 'Can''t lose them'
           WHEN r = 1 AND fm = 2 THEN 'Hibernating'
           WHEN r = 1 AND fm = 1 THEN 'Lost' END AS rfm_segment
    FROM segment_atr
    ORDER BY 1 ASC
          )

SELECT rfm_segment
      ,ROUND(AVG(recency),2) AS average_recency
      ,ROUND(AVG(frequency),2) AS average_frequency
      ,ROUND(AVG(monetary),2) AS average_monetary
FROM segments
GROUP BY 1
ORDER BY 2 DESC, 3 ASC, 4 ASC
```

[Step 4’s Results]



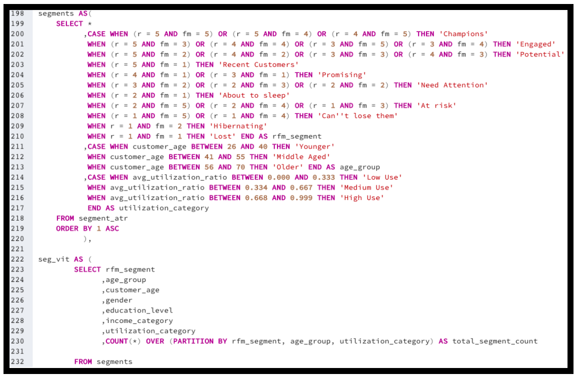
	rfm_segment text	average_recency numeric	average_frequency numeric	average_monetary numeric
1	Can't lose them	4.34	7.06	519.66
2	Hibernating	3.27	4.06	210.99
3	At risk	3.20	6.42	437.38
4	Lost	3.16	2.95	138.75
5	About to sleep	3.00	4.78	148.72
6	Need Attention	2.83	5.38	298.12
7	Engaged	1.95	7.10	560.78
8	Promising	1.91	2.93	136.66
9	Potential	1.73	5.46	297.91
10	Champions	1.23	7.61	684.25
11	Recent Customers	0.97	2.98	128.68

[Step 5’s Results]



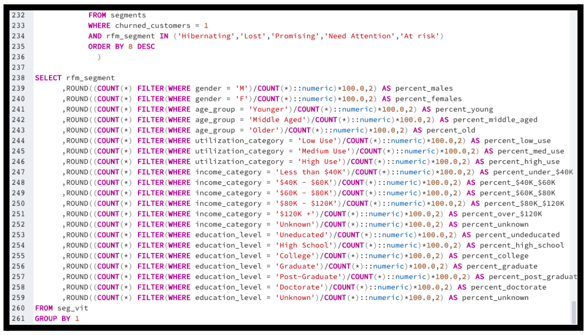
```sql
-- 4. Showing pct. of gender, age_ranges, spend utilization
--ranges, income segments, and educational segments.
WITH crfm AS (
                    SELECT  clietnum
                            ,months_inactive AS Recency
                            ,total_trans_ct AS Frequency
                            ,total_trans_amt AS Monetary
                            ,NTILE(5) OVER (ORDER BY months_inactive DESC) AS r
                            ,NTILE(5) OVER (ORDER BY total_trans_ct ASC) AS f
                            ,NTILE(5) OVER (ORDER BY total_trans_amt ASC) AS m
                    FROM bank_churn
                    ORDER BY 2 DESC
                ),

    segment_atr AS(
                    SELECT bc.clietnum
                          ,bc.customer_age
                          ,bc.gender
                          ,bc.marital_status
                          ,bc.education_level
                          ,bc.income_category
                          ,bc.avg_utilization_ratio
                          ,crfm.recency  -- Computing rfm by month
                          ,ROUND(crfm.frequency::numeric/12,2) AS frequency
                          ,ROUND(crfm.monetary::numeric/12,2) AS monetary
                          ,crfm.r
                          ,crfm.f
                          ,crfm.m
                          ,ROUND((crfm.f + crfm.m)/2,0) AS fm
                          ,SUM(CASE WHEN bc.attrition_flag = 'Attrited Customer' THEN 1 ELSE 0 END) AS churned_customers

                    FROM bank_churn bc
                    JOIN crfm ON (bc.clietnum = crfm.clietnum)
                    GROUP BY 1,2,3,4,5,6,7,8,9,10,11,12,13,14
                    ORDER BY 1 ASC
```

[Step 5’s Results]



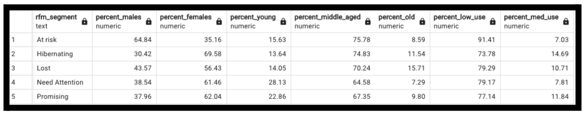
```sql
segments AS(
    SELECT *
          ,CASE WHEN (r = 5 AND fm = 5) OR (r = 5 AND fm = 4) OR (r = 4 AND fm = 5) THEN 'Champions'
           WHEN (r = 5 AND fm = 3) OR (r = 4 AND fm = 4) OR (r = 3 AND fm = 5) OR (r = 3 AND fm = 4) THEN 'Engaged'
           WHEN (r = 5 AND fm = 2) OR (r = 4 AND fm = 2) OR (r = 3 AND fm = 3) OR (r = 4 AND fm = 3) THEN 'Potential'
           WHEN (r = 5 AND fm = 1) THEN 'Recent Customers'
           WHEN (r = 4 AND fm = 1) OR (r = 3 AND fm = 1) THEN 'Promising'
           WHEN (r = 3 AND fm = 2) OR (r = 2 AND fm = 3) OR (r = 2 AND fm = 2) THEN 'Need Attention'
           WHEN (r = 2 AND fm = 1) THEN 'About to sleep'
           WHEN (r = 2 AND fm = 5) OR (r = 2 AND fm = 4) OR (r = 1 AND fm = 3) THEN 'At risk'
           WHEN (r = 1 AND fm = 5) OR (r = 1 AND fm = 4) THEN 'Can''t lose them'
           WHEN r = 1 AND fm = 2 THEN 'Hibernating'
           WHEN r = 1 AND fm = 1 THEN 'Lost' END AS rfm_segment
          ,CASE WHEN customer_age BETWEEN 26 AND 40 THEN 'Younger'
           WHEN customer_age BETWEEN 41 AND 55 THEN 'Middle Aged'
           WHEN customer_age BETWEEN 56 AND 70 THEN 'Older' END AS age_group
          ,CASE WHEN avg_utilization_ratio BETWEEN 0.000 AND 0.333 THEN 'Low Use'
           WHEN avg_utilization_ratio BETWEEN 0.334 AND 0.667 THEN 'Medium Use'
           WHEN avg_utilization_ratio BETWEEN 0.668 AND 0.999 THEN 'High Use'
           END AS utilization_category
    FROM segment_atr
    ORDER BY 1 ASC
          ),

seg_vit AS (
        SELECT rfm_segment
              ,age_group
              ,customer_age
              ,gender
              ,education_level
              ,income_category
              ,utilization_category
              ,COUNT(*) OVER (PARTITION BY rfm_segment, age_group, utilization_category) AS total_segment_count

        FROM segments
```

[Step 5’s Results]



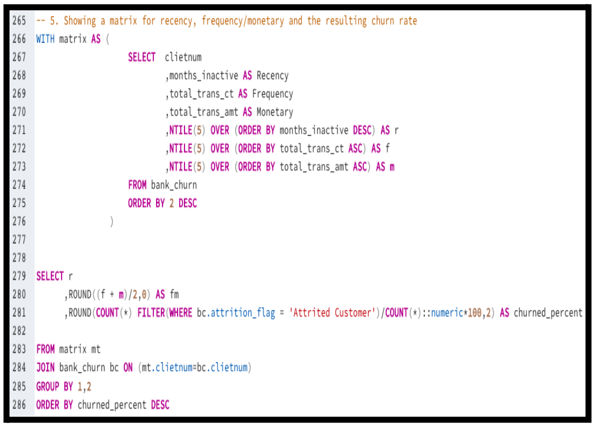
```sql
                FROM segments
                WHERE churned_customers = 1
                AND rfm_segment IN ('Hibernating','Lost','Promising','Need Attention','At risk')
                ORDER BY 8 DESC
                  )

SELECT rfm_segment
        ,ROUND((COUNT(*) FILTER(WHERE gender = 'M')/COUNT(*)::numeric)*100.0,2) AS percent_males
        ,ROUND((COUNT(*) FILTER(WHERE gender = 'F')/COUNT(*)::numeric)*100.0,2) AS percent_females
        ,ROUND((COUNT(*) FILTER(WHERE age_group = 'Younger')/COUNT(*)::numeric)*100.0,2) AS percent_young
        ,ROUND((COUNT(*) FILTER(WHERE age_group = 'Middle Aged')/COUNT(*)::numeric)*100.0,2) AS percent_middle_aged
        ,ROUND((COUNT(*) FILTER(WHERE age_group = 'Older')/COUNT(*)::numeric)*100.0,2) AS percent_old
        ,ROUND((COUNT(*) FILTER(WHERE utilization_category = 'Low Use')/COUNT(*)::numeric)*100.0,2) AS percent_low_use
        ,ROUND((COUNT(*) FILTER(WHERE utilization_category = 'Medium Use')/COUNT(*)::numeric)*100.0,2) AS percent_med_use
        ,ROUND((COUNT(*) FILTER(WHERE utilization_category = 'High Use')/COUNT(*)::numeric)*100.0,2) AS percent_high_use
        ,ROUND((COUNT(*) FILTER(WHERE income_category = 'Less than $40K')/COUNT(*)::numeric)*100.0,2) AS percent_under_$40K
        ,ROUND((COUNT(*) FILTER(WHERE income_category = '$40K - $60K')/COUNT(*)::numeric)*100.0,2) AS percent_$40K_$60K
        ,ROUND((COUNT(*) FILTER(WHERE income_category = '$60K - $80K')/COUNT(*)::numeric)*100.0,2) AS percent_$60K_$80K
        ,ROUND((COUNT(*) FILTER(WHERE income_category = '$80K - $120K')/COUNT(*)::numeric)*100.0,2) AS percent_$80K_$120K
        ,ROUND((COUNT(*) FILTER(WHERE income_category = '$120K +')/COUNT(*)::numeric)*100.0,2) AS percent_over_$120K
        ,ROUND((COUNT(*) FILTER(WHERE income_category = 'Unknown')/COUNT(*)::numeric)*100.0,2) AS percent_unknown
        ,ROUND((COUNT(*) FILTER(WHERE education_level = 'Uneducated')/COUNT(*)::numeric)*100.0,2) AS percent_undeducated
        ,ROUND((COUNT(*) FILTER(WHERE education_level = 'High School')/COUNT(*)::numeric)*100.0,2) AS percent_high_school
        ,ROUND((COUNT(*) FILTER(WHERE education_level = 'College')/COUNT(*)::numeric)*100.0,2) AS percent_college
        ,ROUND((COUNT(*) FILTER(WHERE education_level = 'Graduate')/COUNT(*)::numeric)*100.0,2) AS percent_graduate
        ,ROUND((COUNT(*) FILTER(WHERE education_level = 'Post-Graduate')/COUNT(*)::numeric)*100.0,2) AS percent_post_graduat
        ,ROUND((COUNT(*) FILTER(WHERE education_level = 'Doctorate')/COUNT(*)::numeric)*100.0,2) AS percent_doctorate
        ,ROUND((COUNT(*) FILTER(WHERE education_level = 'Unknown')/COUNT(*)::numeric)*100.0,2) AS percent_unknown
FROM seg_vit
GROUP BY 1
```

[Step 5’s Results(Summary of Results)]



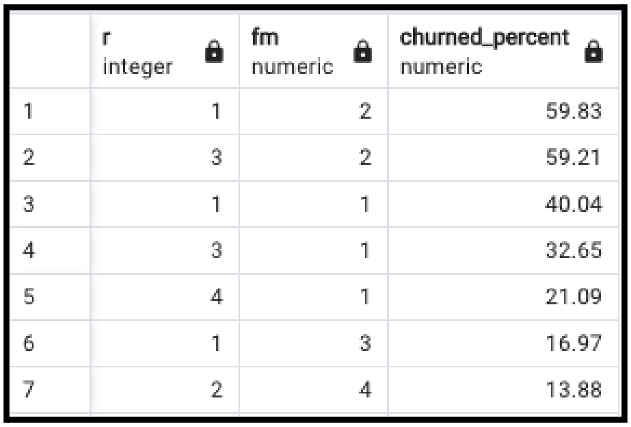
	rfm_segment text	percent_males numeric	percent_females numeric	percent_young numeric	percent_middle_aged numeric	percent_old numeric	percent_low_use numeric	percent_med_use numeric
1	At risk	64.84	35.16	15.63	75.78	8.59	91.41	7.03
2	Hibernating	30.42	69.58	13.64	74.83	11.54	73.78	14.69
3	Lost	43.57	56.43	14.05	70.24	15.71	79.29	10.71
4	Need Attention	38.54	61.46	28.13	64.58	7.29	79.17	7.81
5	Promising	37.96	62.04	22.86	67.35	9.80	77.14	11.84

[Step 6’s Results]



```sql
-- 5. Showing a matrix for recency, frequency/monetary and the resulting churn rate
WITH matrix AS (
                SELECT  clietnum
                        ,months_inactive AS Recency
                        ,total_trans_ct AS Frequency
                        ,total_trans_amt AS Monetary
                        ,NTILE(5) OVER (ORDER BY months_inactive DESC) AS r
                        ,NTILE(5) OVER (ORDER BY total_trans_ct ASC) AS f
                        ,NTILE(5) OVER (ORDER BY total_trans_amt ASC) AS m
                FROM bank_churn
                ORDER BY 2 DESC
            )


SELECT r
      ,ROUND((f + m)/2,0) AS fm
      ,ROUND(COUNT(*) FILTER(WHERE bc.attrition_flag = 'Attrited Customer')/COUNT(*)::numeric*100,2) AS churned_percent

FROM matrix mt
JOIN bank_churn bc ON (mt.clietnum=bc.clietnum)
GROUP BY 1,2
ORDER BY churned_percent DESC
```

[Step 6’s Results(Summary of Results)]



	r integer	fm numeric	churned_percent numeric
1	1	2	59.83
2	3	2	59.21
3	1	1	40.04
4	3	1	32.65
5	4	1	21.09
6	1	3	16.97
7	2	4	13.88

Interpreting statistical findings:

1. Our data analysis showed that our total customer base was **9,480 customers**. They **spent $41.18M** over the **year**, with an average of **51.44K transactions** per **month**. Unfortunately, **15.70%** of them **churned**.
2. We created **11 different segments** for our customers based on their RFM scores. This allowed us to **identify customers who are at risk of churning** and target them with specific marketing campaigns.
3. We analyzed our data and found that customers who were **inactive for 3 months** or more were more likely to churn. We also found that customers who **spent less than $300 per month** and who **transacted less than 5.5 times per month** were at a higher risk of churning.
4. Our data analysis shows that the majority of customers who are **most likely to churn** are **female**, **middle-aged**, with l**ow card utilization**, **low income**, and a

graduate level of education. These customers are in the Hibernating, Lost, and Promising segments.

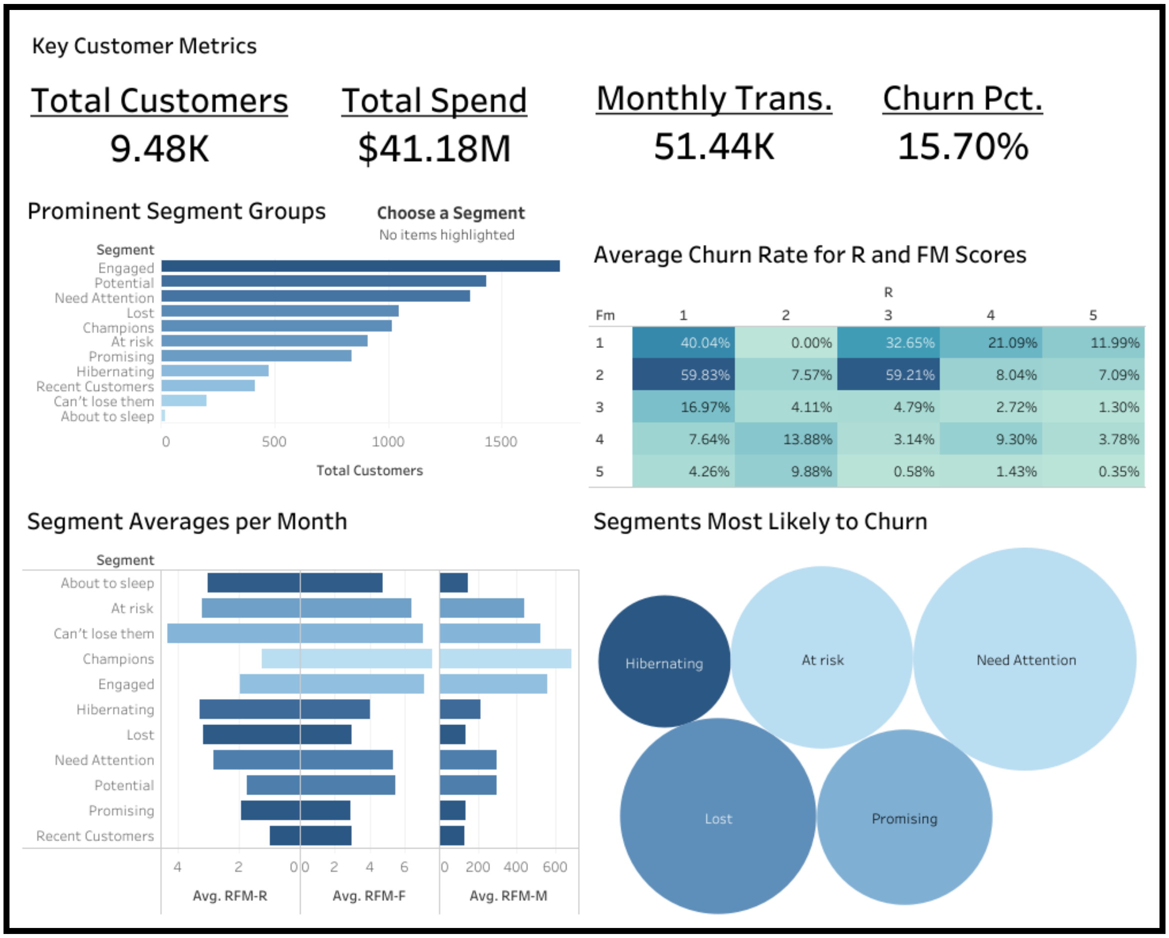
5. We found that customers who had recent activity and **low frequency and monetary scores of 1 or 2** were more likely to churn. This was also true for customers who had a **recency score of 3 or 4** and low frequency and monetary scores.

. . .

Step 5: Share

In this step, we are creating visualizations and communicating our findings based on our analysis.

5.1 Data Visualizations and Findings



Key Customer Metrics

Total Customers	Total Spend	Monthly Trans.	Churn Pct.
9.48K	$41.18M	51.44K	15.70%

Average Churn Rate for R and FM Scores

	R				
Fm	1	2	3	4	5
1	40.04%	0.00%	32.65%	21.09%	11.99%
2	59.83%	7.57%	59.21%	8.04%	7.09%
3	16.97%	4.11%	4.79%	2.72%	1.30%
4	7.64%	13.88%	3.14%	9.30%	3.78%
5	4.26%	9.88%	0.58%	1.43%	0.35%

Customer Segmentation by RFM, Demographics, and Spending
(Visit my tableau profile for interactivity with this dashboard)

This dashboard analyzes the most prominent segment groups, their RFM averages, and identifies trends about segments that are most likely to churn.

1. We have identified three customer segments that are most likely to churn: **Hibernating**, **Lost**, and **Promising**. These segments represent **24.97% of our customer base** and their behavior and demographics can help us develop strategies to limit churn.
2. The Hibernating, Lost, and Promising segments had a higher churn rate, **40.16%**, than other segments. This is likely due to their behavior, which is characterized by less recent, less frequent, and lower spending.
3. We believe that there is a **correlation** between **frequency** of purchase and **retention rate**. Customers who purchase more frequently are more likely to be satisfied with our products and services, and they are less likely to switch to a competitor.

. . .

Step 6: Act

In the final step, we will be delivering our insights and providing recommendations based on our analysis.

Here, we revisit our business questions and share with you our high-level business recommendations.

1. **What are the average churn rates for customers with RFM scores from 1-5?**

    - Our analysis found that customers with low recency, frequency, and monetary scores had up to a **59.83% churn rate**. Customers with a recency score of 3 but low frequency and monetary scores had up to a **59.21% churn rate**. We should focus on increasing customer engagement by encouraging the use of our products and services more often. We can accomplish this by offering special promotions and discounts, or by providing more personalized customer service.

2. **Who are our most common customer segments?**
    - Our analysis found that the majority of our customers are happy with our service, but two segments, **Need Attention** and **Lost**, are at risk of churning. These segments **represent 25.40% of our customer base**. Based on the data we see between frequency and customer retention, I recommend that we focus on offering special promotions and discounts to help these customers engage and enjoy their card service more.

3. **What is each customer segments' average monthly RFM values?**
    - Our analysis of monthly averages revealed patterns among the different segment groups. **The segments most likely to churn** were inactive for an average of 2.87 months, transacted 4.35 times per month, and spent $244.40 per month. **In contrast**, the rest of the group had an average of 2.20 months of inactivity, 5.83 transactions per month, and $390.02 in spending. I recommend personalized customer interactions which can lead to increased customer satisfaction, loyalty, and repeat business.

4. **What segments are most likely to churn, and what are their characterizations?**
    - In summary, our analysis revealed that the following demographic and spending habits are common among customers who belong to the **Hibernating, Lost, Promising, Need Attention, and At risk** segments:

- **Female**: 56.93%
- **Middle-aged**: 70.56%
- **Low card utilization**: 80.16%
- **Income under $40,000**: 37.71%
- **Graduate-level education**: 30.34%

These findings can be used to develop targeted interventions to reduce customer churn.