

CUSTOMER SENTIMENT ANALYSIS (END CAPSTONE)

SUBMITTED BY: T.SASHIDHAR

COHORT: 20

INTEGRATED DATA SCIENCE

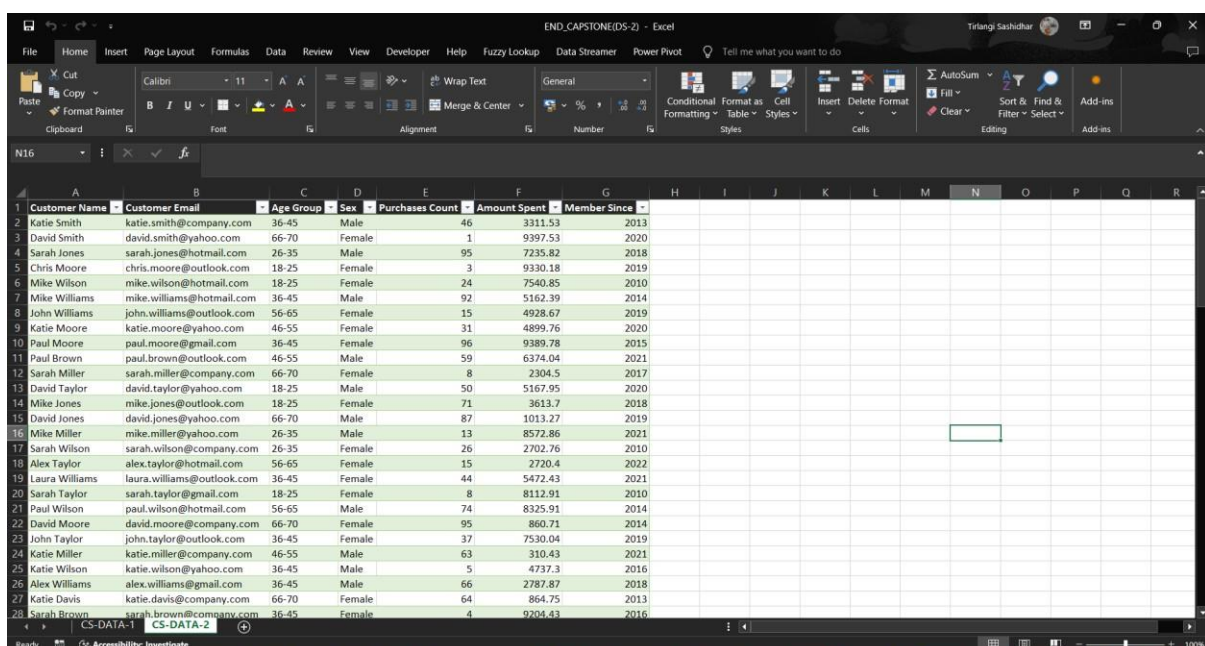
1. Introduction

1.1 Project Overview

This project aims to build an automated system to analyze customer data, calculate Customer Lifetime Value (CLV), and segment customers based on their purchasing behavior. The system also includes interactive visualizations for exploring relationships between variables such as customer segments, purchase history, and revenue.

The system is designed to handle large datasets efficiently, using Python-based tools and libraries such as Pandas, Matplotlib, Plotly, and IPyWidgets. Key features include:

- **Customer Lifetime Value (CLV) Calculation:** Calculate CLV to identify high-value customers.
- **Customer Segmentation:** Customers are segmented based on CLV into High, Medium, and Low-value segments.
- **Memory-Efficient Data Processing:** A generator function is used to handle large datasets without consuming excessive memory.
- **Interactive Visualizations:** Visual analysis with dynamic, interactive charts that allow users to filter and explore customer segments and trends over time.



Customer Name	Customer Email	Age Group	Sex	Purchases Count	Amount Spent	Member Since
Katie Smith	katie.smith@company.com	36-45	Male	46	3311.53	2013
David Smith	david.smith@yahoo.com	66-70	Female	1	9397.53	2020
Sarah Jones	sarah.jones@hotmail.com	26-35	Male	95	7235.82	2018
Chris Moore	chris.moore@outlook.com	18-25	Female	3	9330.18	2019
Mike Wilson	mike.wilson@hotmail.com	18-25	Female	24	7540.85	2010
Mike Williams	mike.williams@hotmail.com	36-45	Male	92	5162.39	2014
John Williams	john.williams@outlook.com	56-65	Female	15	4928.67	2019
Katie Moore	katie.moore@yahoo.com	46-55	Female	31	4899.76	2020
Paul Moore	paul.moore@gmail.com	36-45	Female	96	9389.78	2015
Paul Brown	paul.brown@outlook.com	46-55	Male	59	6374.04	2021
Sarah Miller	sarah.miller@company.com	66-70	Female	8	2304.5	2017
David Taylor	david.taylor@yahoo.com	18-25	Male	50	5167.95	2020
Mike Jones	mike.jones@outlook.com	18-25	Female	71	3613.7	2018
David Jones	david.jones@yahoo.com	66-70	Male	87	1013.27	2019
Mike Miller	mike.miller@yahoo.com	26-35	Male	13	8572.86	2021
Sarah Wilson	sarah.wilson@company.com	26-35	Female	26	2702.76	2010
Alex Taylor	alex.taylor@hotmail.com	56-65	Female	15	2720.4	2022
Laura Williams	laura.williams@outlook.com	36-45	Female	44	5472.43	2021
Sarah Taylor	sarah.taylor@gmail.com	18-25	Female	8	8112.91	2010
Paul Wilson	paul.wilson@hotmail.com	56-65	Male	74	8325.91	2014
David Moore	david.moore@company.com	66-70	Female	95	860.71	2014
John Taylor	john.taylor@outlook.com	36-45	Female	37	7530.04	2019
Katie Miller	katie.miller@company.com	46-55	Male	63	310.43	2021
Katie Wilson	katie.wilson@yahoo.com	36-45	Male	5	4737.3	2016
Alex Williams	alex.williams@gmail.com	36-45	Male	66	2787.87	2018
Katie Davis	katie.davis@company.com	66-70	Female	64	864.75	2013
Sarah Brown	sarah.brown@company.com	36-45	Female	4	9204.43	2016

2.1. Data Preprocessing

- **Loading Data:** The dataset was loaded from an Excel file.
- **Date Conversion:** The 'Customer Since' column was converted to a datetime format for easier manipulation and filtering.
- **Handling Missing Data:** Any invalid date entries were identified and handled appropriately.

The screenshot displays the Power Query Editor interface. The main area shows a table with the following columns: Name, Email, Age, Gender, Total Purchases, Total Spent, and Customer Since. The table contains 28 rows of customer data. The 'Query Settings' pane on the right shows the 'APPLIED STEPS' list, which includes 'Removed Duplicates1'.

	Name	Email	Age	Gender	Total Purchases	Total Spent	Customer Since
1	Katie Smith	katie.smith@hotmail.com	40	Male	38	5423.18	2017
2	David Smith	david.smith@hotmail.com	40	Female	54	8292.73	2019
3	Sarah Jones	sarah.jones@yahoo.com	36	Female	90	8270.97	2014
4	Chris Moore	chris.moore@yahoo.com	22	Male	22	4891.91	2019
5	Mike Wilson	mike.wilson@company.com	30	Female	27	1276.33	2017
6	Mike Williams	mike.williams@company.com	18	Male	33	5163	2014
7	John Williams	john.williams@yahoo.com	25	Male	71	4552.67	2017
8	Katie Moore	katie.moore@outlook.com	68	Male	15	1362.9	2018
9	Paul Moore	paul.moore@company.com	59	Male	30	1493.74	2021
10	Paul Brown	paul.brown@outlook.com	30	Female	29	9293.06	2022
11	Sarah Miller	sarah.miller@outlook.com	51	Female	78	6791.9	2023
12	David Taylor	david.taylor@hotmail.com	49	Male	12	3473.45	2017
13	Mike Jones	mike.jones@hotmail.com	49	Male	82	141.93	2021
14	David Jones	david.jones@gmail.com	24	Male	99	2020.54	2020
15	Mike Miller	mike.miller@yahoo.com	18	Female	25	620.27	2020
16	Sarah Wilson	sarah.wilson@hotmail.com	19	Male	81	1366.3	2012
17	Alex Taylor	alex.taylor@yahoo.com	31	Female	38	2197.21	2023
18	Laura Williams	laura.williams@gmail.com	34	Male	24	8645.22	2019
19	Sarah Taylor	sarah.taylor@gmail.com	26	Female	84	1266.02	2019
20	Paul Wilson	paul.wilson@hotmail.com	26	Female	57	5613.42	2020
21	David Moore	david.moore@yahoo.com	39	Female	47	3553.52	2020
22	John Taylor	john.taylor@company.com	48	Female	49	7135.55	2012
23	Katie Miller	katie.miller@gmail.com	41	Male	39	8998.09	2017
24	Katie Wilson	katie.wilson@yahoo.com	21	Female	65	8311.87	2018
25	Alex Williams	alex.williams@outlook.com	61	Male	97	7728.88	2018
26	Katie Davis	katie.davis@hotmail.com	27	Male	90	6420.23	2013
27	Sarah Brown	sarah.brown@hotmail.com	42	Female	21	9228.89	2017
28	Paul Johnson	paul.johnson@hotmail.com	57	Male	42	6342.83	2017

2. System Architecture

The system architecture revolves around a single Python class called Customer, which encapsulates the main functionalities for processing customer data. The key methods and components are described below.

2.1 Class and Methods

- **Customer Class:** The Customer class is responsible for managing customer data and performing the core analysis.
 - **Attributes:**
 - **customers_df:** A Pandas DataFrame containing customer data such as Customer ID, Total Spent, Total Purchases, and CLV.

Methods:

- `calculate_clv`: Calculates CLV using the formula $CLV = \text{Total Spent} / \text{Total Purchases}$. This method is wrapped with a timing decorator to log execution time.
- `segment_customers`: Segments customers based on their calculated CLV into three categories: High Value, Medium Value, and Low Value.
- `customer_generator`: A generator function that yields one customer record at a time, useful for memory-efficient data processing.
- `process_large_dataset`: Uses the `customer_generator` to process customer records in a memory-efficient manner.
- `plot_summary_statistics`: Generates summary statistics and visualizations for customer data, including histograms for total spending and bar charts for revenue.
- `interactive_charts`: Creates interactive charts using IPyWidgets and Plotly to dynamically explore customer segments.

PYTHON CLASS CUSTOMER THAT REPRESENTS A CUSTOMER AND INCLUDES
✓ METHODS TO CALCULATE CLV AND SEGMENT THE CUSTOMER BASED ON THEIR
PURCHASING BEHAVIOR

```
import pandas as pd

# DEFINE CUSTOMER CLASS
class Customer:
    def __init__(self, name, email, age, gender, total_purchases, total_spent, customer_since):
        self.name = name
        self.email = email
        self.age = age
        self.gender = gender
        self.total_purchases = total_purchases
        self.total_spent = total_spent
        self.customer_since = customer_since

    def calculate_clv(self):
        """CALCULATE CUSTOMER LIFETIME VALUE (CLV)."""
        current_year = pd.Timestamp.now().year
        years_as_customer = current_year - self.customer_since
```

```
Name: Katie Smith
Email: katie.smith@hotmail.com
Age: 40
Gender: Male
Total Purchases: 38
Total Spent: $5423.18
Customer Since: 2017
CLV: $774.74
Segment: Loyal
```

IMPLEMENT A TIMING_DECORATOR AND APPLY IT TO THE CALCULATE_CLVMETHOD OF THE CUSTOMER CLASS TO LOG EXECUTION TIME.

```
import time

# DEFINE THE TIMING DECORATOR
def timing_decorator(func):
    def wrapper(*args, **kwargs):
        start_time = time.time()
        result = func(*args, **kwargs)
        end_time = time.time()
        execution_time = end_time - start_time
        print(f"Execution time for {func.__name__}: {execution_time:.6f} seconds")
        return result
    return wrapper

# SETTING UP THE FILE PATH
file_path = "/content/drive/MyDrive/END_CAPSTONE(DS-2).xlsx"
data_1 = pd.read_excel(file_path, sheet_name='CS-DATA-1')

# CREATING A CUSTOMER INSTANCE FROM THE FIRST ROW OF DATA IN CS-DATA-1
customer_example = Customer(
    name=data_1.loc[0, 'Name'],
    email=data_1.loc[0, 'Email'],
    age=data_1.loc[0, 'Age'],
    gender=data_1.loc[0, 'Gender'],
    total_purchases=data_1.loc[0, 'Total Purchases'],
```

```
Execution time for calculate_clv: 0.000082 seconds
Name: Katie Smith
Email: katie.smith@hotmail.com
Age: 40
Gender: Male
Total Purchases: 38
Total Spent: $5423.18
Customer Since: 2017
CLV: $774.74
Segment: Loyal
```

WRITE A GENERATOR FUNCTION TO ITERATE OVER LARGE DATASETS AND YIELD

- CUSTOMER RECORDS ONE AT A TIME. INTEGRATE THIS GENERATOR INTO THE CUSTOMER CLASS TO PROCESS DATA MEMORY-EFFICIENTLY.

```
[ ] import pandas as pd
import time

# DEFINE THE TIMING DECORATOR
def timing_decorator(func):
    def wrapper(*args, **kwargs):
        start_time = time.time()
        result = func(*args, **kwargs)
        end_time = time.time()
        execution_time = end_time - start_time
        print(f"Execution time for {func.__name__}: {execution_time:.6f} seconds")
        return result
    return wrapper

# DEFINE A GENERATOR FUNCTION TO YIELD CUSTOMER RECORDS FROM A LARGE EXCEL DATASET
def customer_record_generator(file_path, sheet_name, chunk_size=1000):
    """Generator function to read and yield customer records from a large dataset."""
    start_row = 0
    while True:
        chunk = pd.read_excel(file_path, sheet_name=sheet_name, skiprows=start_row, nrows=chunk_size)
```

```
CLV: $277.92
Segment: Regular
-----
Execution time for calculate_clv: 0.000020 seconds
Name: Alex Davis
Email: alex.davis@hotmail.com
Age: 21
Gender: Female
Total Purchases: 78
Total Spent: $6271.78
Customer Since: 2010
CLV: $447.98
Segment: Loyal
-----
Execution time for calculate_clv: 0.000021 seconds
Name: Jane Johnson
Email: jane.johnson@gmail.com
Age: 35
Gender: Female
Total Purchases: 98
Total Spent: $8400.11
Customer Since: 2012
CLV: $700.01
Segment: Loyal
```

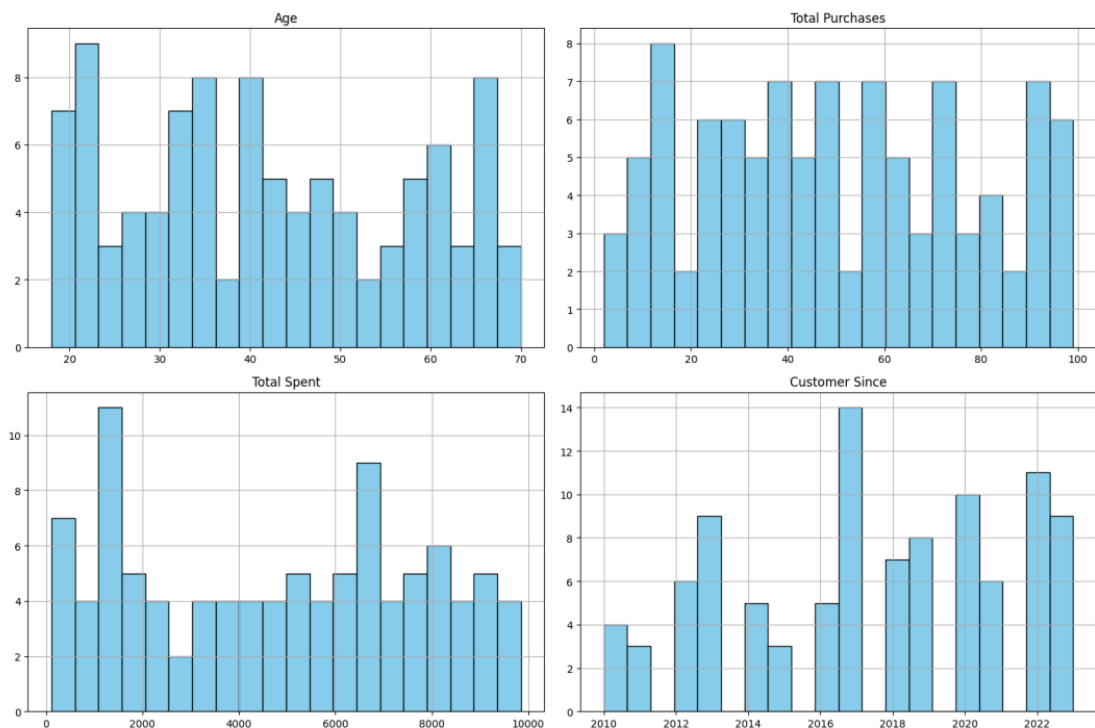
SUMMARY STATISTICS, VISUALIZATIONS, AND CORRELATION MATRICES TO EXPLORE RELATIONSHIPS BETWEEN VARIABLES

```
[ ] import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

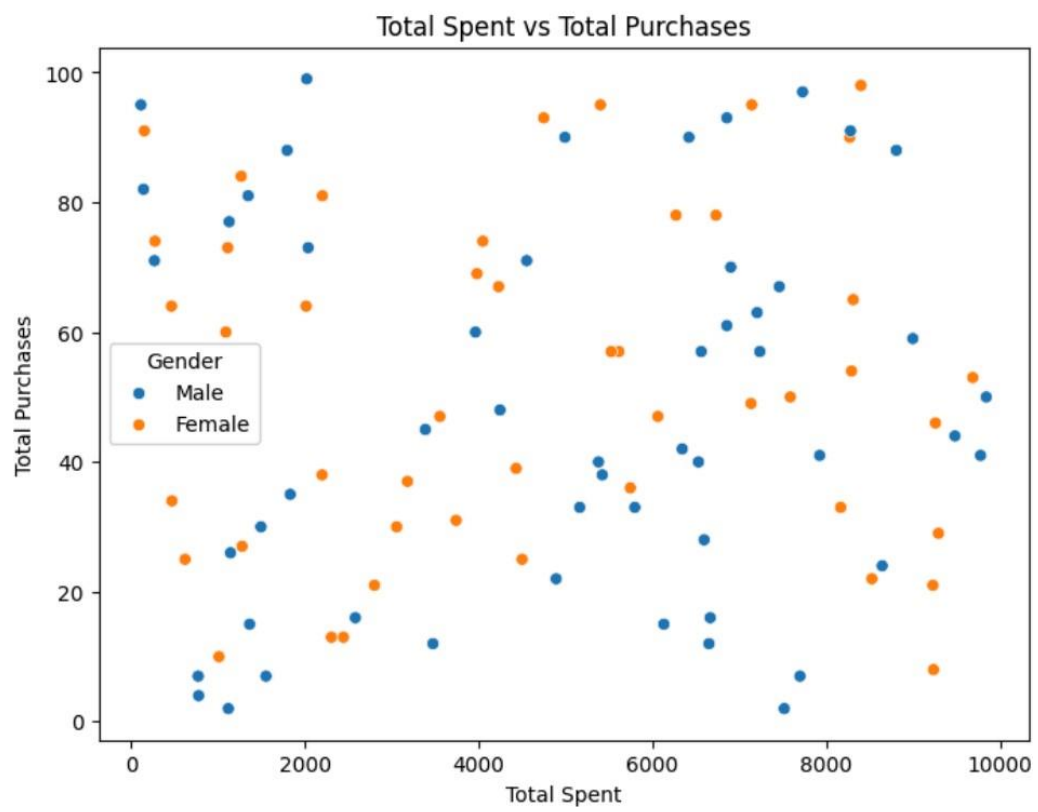
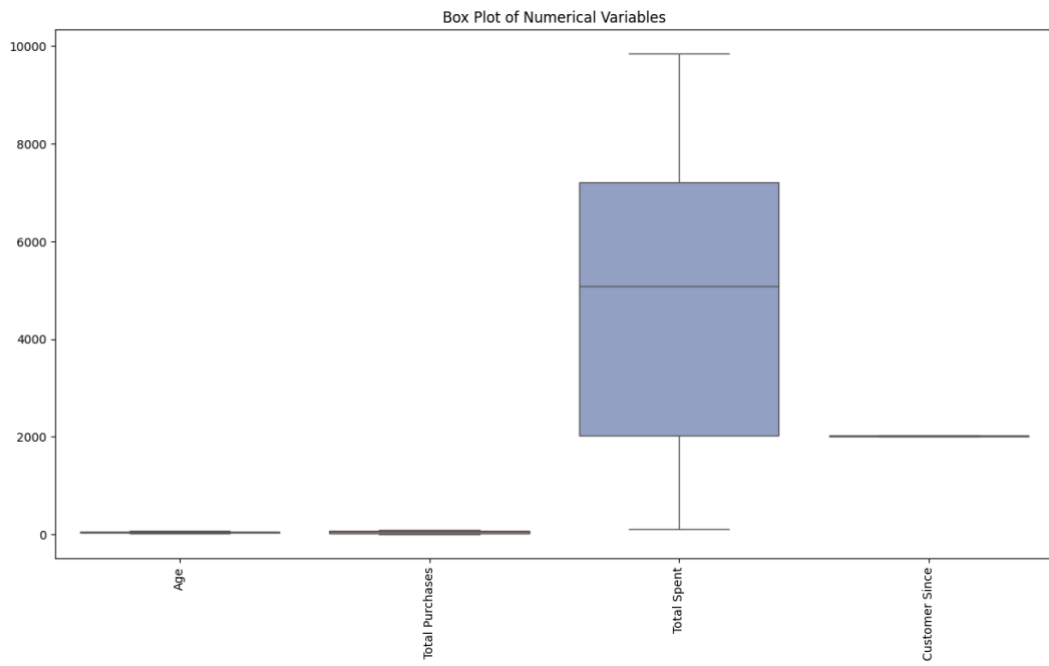
```
# LOAD THE DATASET FROM EXCEL
file_path = "/content/drive/MyDrive/END_CAPSTONE(DS-2).xlsx"
df = pd.read_excel(file_path, sheet_name='CS-DATA-1')

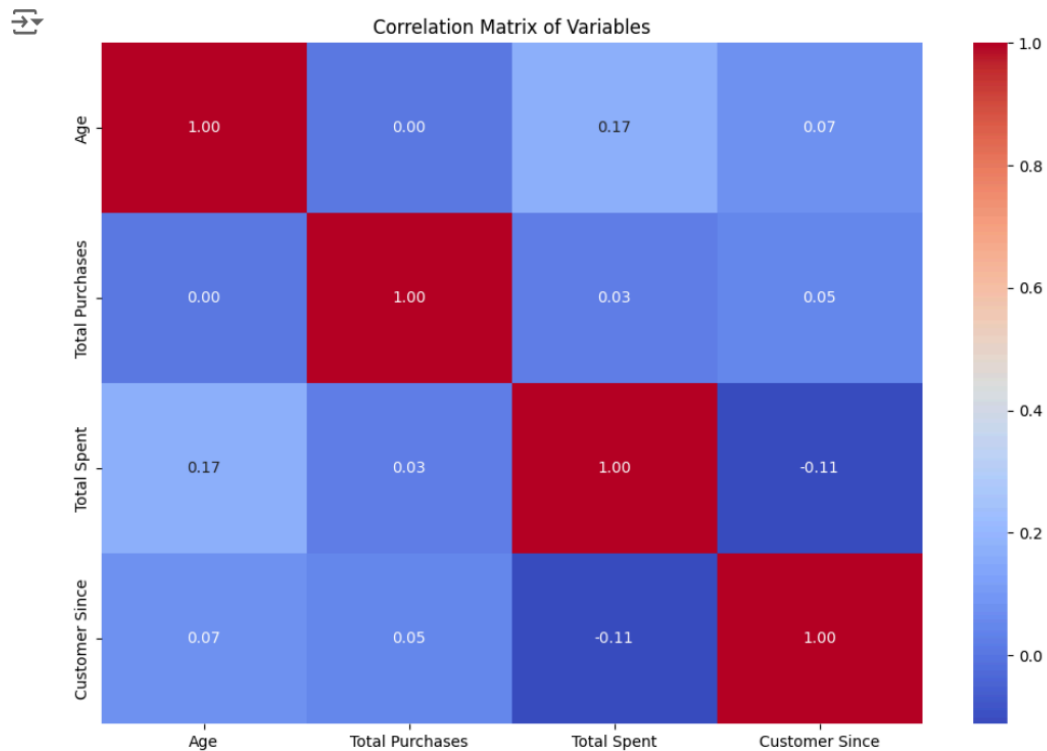
# GENERATE SUMMARY STATISTICS
summary_statistics = df.describe()
print(summary_statistics)
```

	Age	Total Purchases	Total Spent	Customer Since
count	100.000000	100.000000	100.000000	100.000000
mean	42.180000	50.000000	4829.193000	2017.480000
std	15.582547	27.891203	2958.068726	3.841506
min	18.000000	2.000000	112.120000	2010.000000
25%	30.000000	27.750000	2018.805000	2014.000000
50%	41.000000	47.500000	5078.085000	2018.000000
75%	56.250000	73.000000	7213.442500	2021.000000
max	70.000000	99.000000	9843.330000	2023.000000



```
# BOX PLOT FOR ALL NUMERICAL VALUES
plt.figure(figsize=(15, 8))
sns.boxplot(data=df.select_dtypes(include=np.number), palette='Set2')
plt.xticks(rotation=90)
plt.title('Box Plot of Numerical Variables')
plt.show()
```



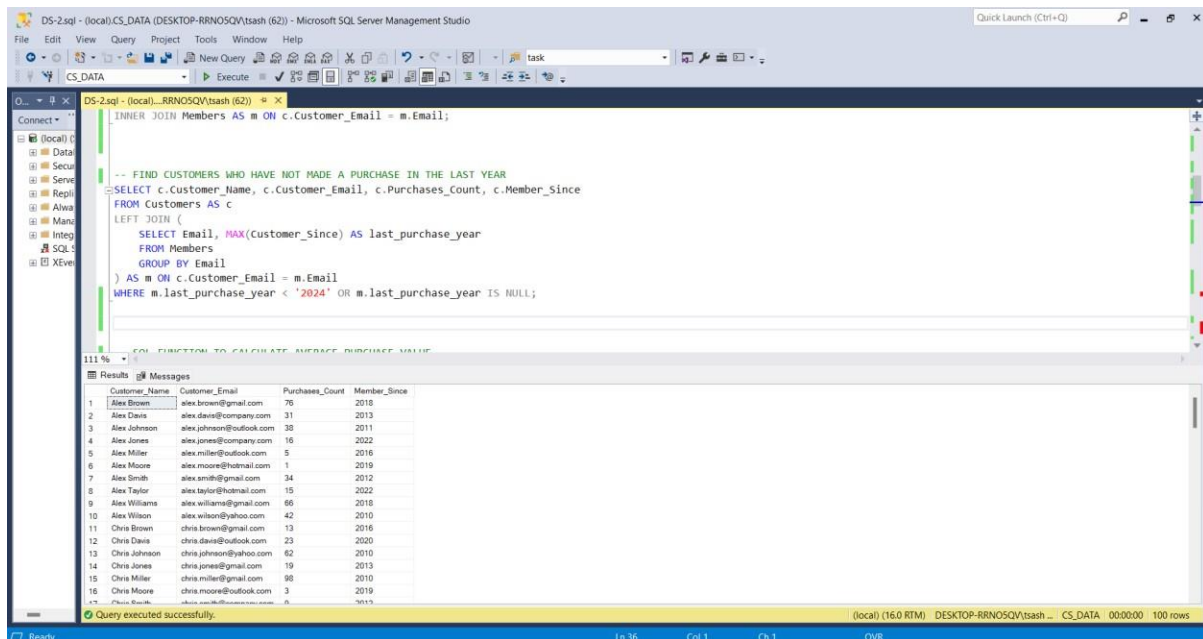


After that, we load the table in SQL SMS to perform further analysis

Query executed successfully.

	Name	Email	Age	Gender	Total_Purchases	Total_Spent	Customer_Since
1	Alex Brown	alex.brown@company.com	35	Male	15	6131.8100559375	2023
2	Alex Davis	alex.davis@hotmail.com	21	Female	78	6271.77978515625	2010
3	Alex Johnson	alex.johnson@gmail.com	42	Female	64	2013.59997558594	2019
4	Alex Jones	alex.jones@yahoo.com	07	Male	70	8901.97021494375	2015
5	Alex Miller	alex.miller@yahoo.com	21	Female	47	6062.83984375	2012
6	Alex Moore	alex.moore@outlook.com	65	Female	95	7144	2023
7	Alex Smith	alex.smith@outlook.com	43	Female	50	7587.5400390625	2018
8	Alex Taylor	alex.taylor@yahoo.com	31	Female	38	2197.209600375	2023
9	Alex Williams	alex.williams@outlook.com	61	Male	07	7728.876020125	2018
10	Alex Wilson	alex.wilson@yahoo.com	44	Female	57	5524.740234375	2016
11	Chris Brown	chris.brown@company.com	20	Female	74	4048.1368257813	2017
12	Chris Davis	chris.davis@hotmail.com	33	Female	31	3740.02001903125	2020
13	Chris Johnson	chris.johnson@outlook.com	06	Female	13	2442.8004852813	2016
14	Chris Jones	chris.jones@hotmail.com	46	Female	46	6295.330078125	2018
15	Chris Miller	chris.miller@gmail.com	30	Female	21	2880.1298828125	2015
16	Chris Moore	chris.moore@yahoo.com	22	Male	22	4891.91015625	2019
17	Chris Smith	chris.smith@company.com	54	Male	87	7754.8008838125	2015

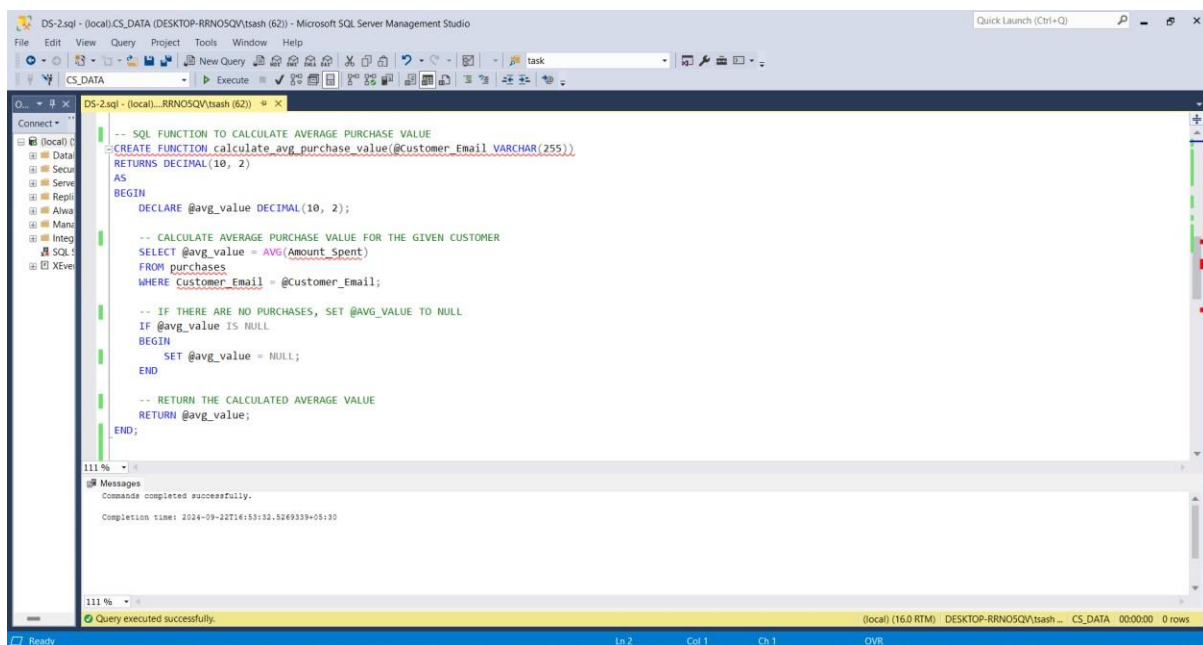
FIND CUSTOMERS WHO HAVE NOT MADE A PURCHASE IN THE LAST YEAR



```
-- FIND CUSTOMERS WHO HAVE NOT MADE A PURCHASE IN THE LAST YEAR
SELECT c.Customer_Name, c.Customer_Email, c.Purchases_Count, c.Member_Since
FROM Customers AS c
LEFT JOIN (
    SELECT Email, MAX(Customer_Since) AS last_purchase_year
    FROM Members
    GROUP BY Email
) AS m ON c.Customer_Email = m.Email
WHERE m.last_purchase_year < '2024' OR m.last_purchase_year IS NULL;
```

Customer_Name	Customer_Email	Purchases_Count	Member_Since
Alex Brown	alex.brown@gmail.com	76	2018
Alex Davis	alex.davis@company.com	31	2013
Alex Johnson	alex.johnson@outlook.com	30	2011
Alex Jones	alex.jones@company.com	16	2022
Alex Miller	alex.miller@outlook.com	5	2016
Alex Moore	alex.moore@hotmail.com	1	2019
Alex Smith	alex.smith@gmail.com	34	2012
Alex Taylor	alex.taylor@hotmail.com	15	2022
Alex Williams	alex.williams@gmail.com	66	2018
Alex Wilson	alex.wilson@yahoo.com	42	2010
Chris Brown	chris.brown@gmail.com	13	2016
Chris Davis	chris.davis@outlook.com	23	2020
Chris Johnson	chris.johnson@yahoo.com	62	2010
Chris Jones	chris.jones@gmail.com	19	2013
Chris Miller	chris.miller@gmail.com	98	2010
Chris Moore	chris.moore@outlook.com	3	2019
Chris Smith	chris.smith@company.com	8	2015

SQL FUNCTION TO CALCULATE AVERAGE PURCHASE VALUE



```
-- SQL FUNCTION TO CALCULATE AVERAGE PURCHASE VALUE
CREATE FUNCTION calculate_avg_purchase_value(@Customer_Email VARCHAR(255))
RETURNS DECIMAL(10, 2)
AS
BEGIN
    DECLARE @avg_value DECIMAL(10, 2);

    -- CALCULATE AVERAGE PURCHASE VALUE FOR THE GIVEN CUSTOMER
    SELECT @avg_value = AVG(Amount_Spent)
    FROM purchases
    WHERE Customer_Email = @Customer_Email;

    -- IF THERE ARE NO PURCHASES, SET @AVG_VALUE TO NULL
    IF @avg_value IS NULL
    BEGIN
        SET @avg_value = NULL;
    END

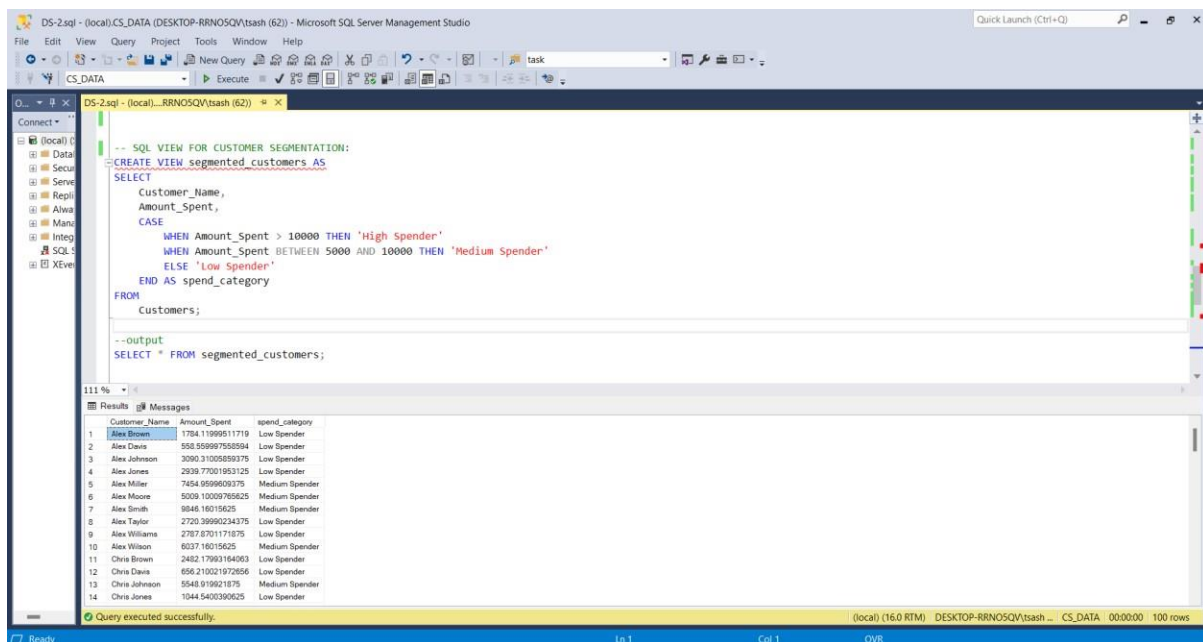
    -- RETURN THE CALCULATED AVERAGE VALUE
    RETURN @avg_value;
END;
```

Messages

Command completed successfully.

Completion time: 2024-09-22T16:53:32.5269339+05:30

SQL VIEW FOR CUSTOMER SEGMENTATION:



POWER BI (DATA VISUALIZATION AND DASHBOARD ANALYSIS)

1. Data Model Design:

In Power BI, the customer dataset should be modeled to reflect key relationships and attributes:

- **Customer Table:** Contains details like Customer NAME, Total Spent, Total Purchases, and Segment (High, Medium, Low Value).
- **CLV Calculation:** Use calculated columns or measures to compute Customer Lifetime Value (CLV) within Power BI, similar to the Python logic.
- **Segmentation:** Apply DAX formulas to segment customers based on CLV thresholds.

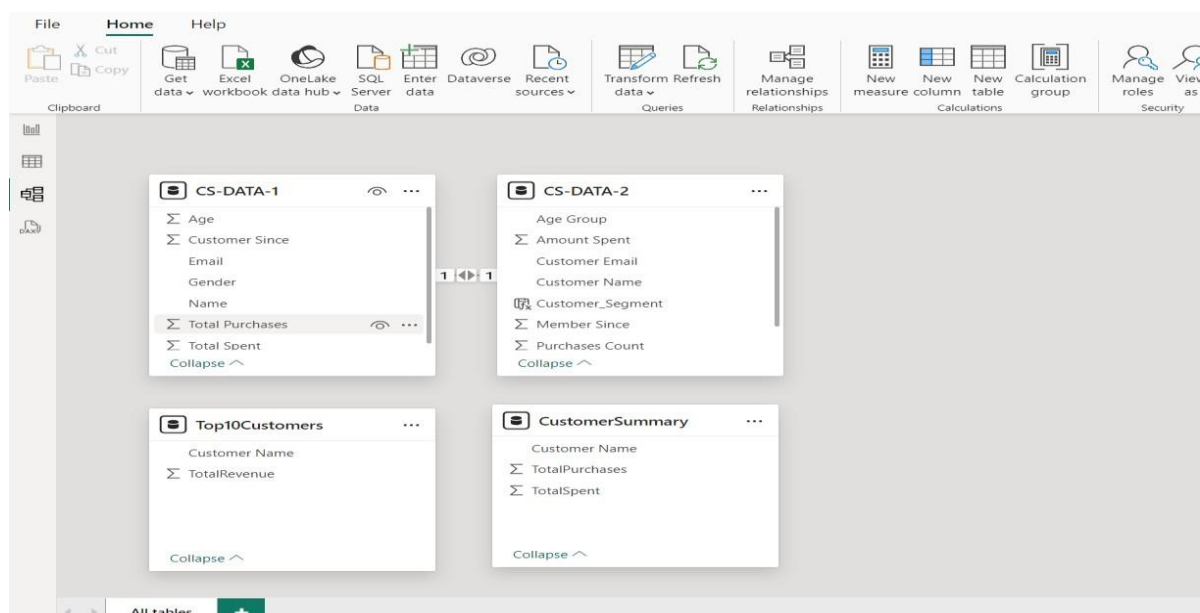
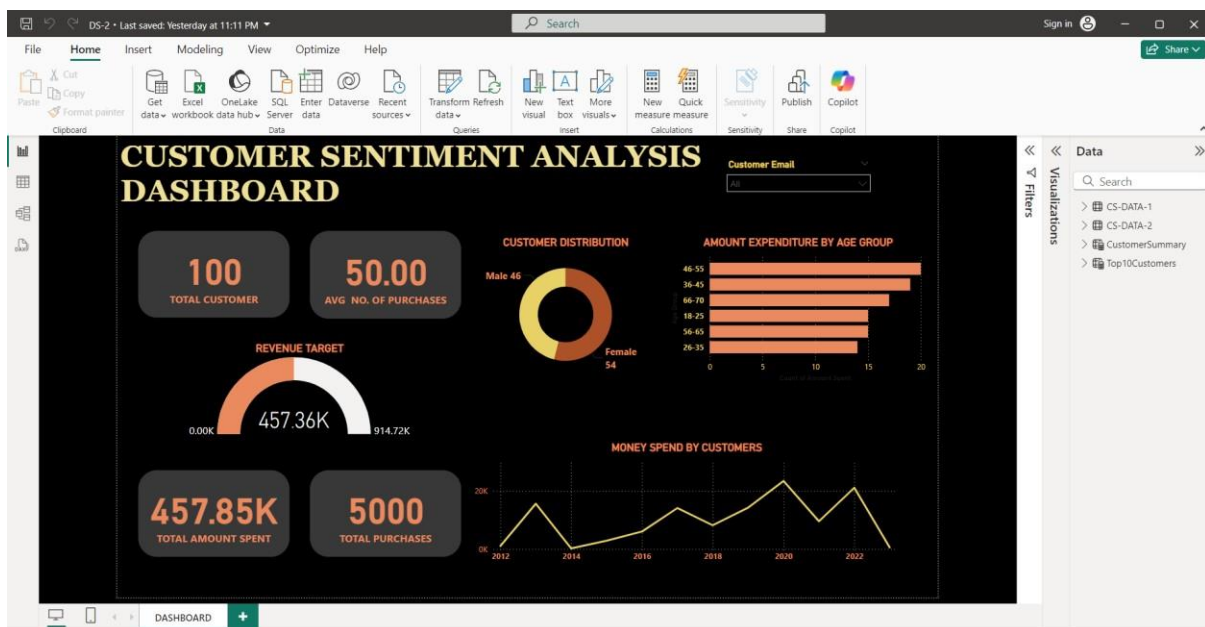
2. Key Measures:

- **Total CLV:** Aggregates CLV across the customer base.
- **Revenue by Segment:** A measure to Total Spent by each segment.
- **Average CLV:** Calculated as total CLV divided by the number of customers.
- **Customer Count by Segment:** Counts customers in each segment (High, Medium, Low).

3. Visualizations:

Power BI will create dynamic visualizations to explore the relationships between customer segments and behavior:

- **Segmented Bar Charts:** Display revenue or CLV by customer segment.
- **Interactive Slicers:** Enable users to filter data by customer segment or date ranges.
- **Line Charts:** Show CLV trends or Total Spent over time, helping analyze the evolution of customer value.
- **Distribution Histograms:** Visualize the distribution of total spending or CLV across customers.



This project has successfully developed an automated system for calculating Customer Lifetime Value (CLV) and segmenting customers. By leveraging Python for data processing, memory-efficient techniques, and interactive visualizations, the system provides scalable solutions for businesses to analyze and categorize customers based on purchasing behavior. The inclusion of a generator function ensures efficient processing of large datasets, while the interactive charts enable real-time exploration of customer segments and trends. Furthermore, the solution's modular and expandable design lays a solid foundation for future enhancements, such as advanced machine learning models and integration with real-time data sources, offering valuable insights for informed business decisions.