CUSTOMER SEGMENTATION ANALYSIS

Table of Contents:

Introduction:

In today's highly competitive market, understanding customer behavior and maximizing the lifetime value of each customer is crucial for businesses. This project focuses on segmenting customers based on their purchasing patterns and demographics, allowing for targeted marketing strategies to improve customer retention. The main objective is to calculate Customer Lifetime Value (CLV) for each customer, segment the customer base into distinct groups (high, medium, and low-value), and provide actionable insights through advanced data analytics techniques, so businesses can identify high-value customers, optimize marketing strategies, and enhance overall profitability.

Objective of the Project:

To analyze customer data from an e-commerce platform to segment customers based on purchasing behavior and demographics. Calculate Customer Lifetime Value (CLV) to inform targeted marketing strategies and enhance customer retention, by using Excel for initial data cleaning, SQL Server for data storage and management, Power BI for interactive dashboards, and Python for advanced analytics.

Scope of the Project:

The scope of the Resume Parser Project encompasses the following key areas:

1. **Data Cleaning and Preprocessing:**
   - Initial customer data, which included inconsistencies and multiple entries across different sheets, was cleaned using Excel, and a unique customer ID was created.
   - VBA Automation is used to automate this process.
2. **Database Integration:**
   - The cleaned customer data was stored in SQL Server using VBA automation. This ensures that any future entries in the Excel sheets are automatically reflected in the SQL Server database, providing an ongoing mechanism for data updates.

3. **Customer Segmentation:**
   - The combined customer data was used to calculate Customer Lifetime Value (CLV) for each customer in SQL. Based on the CLV values, customers were segmented into high-value, medium-value, and low-value customer groups.
4. **Data Visualization:**
   - Power BI was used to connect directly to the SQL Server database via Direct Query. This ensured that any changes in the database, such as new customer entries or updates to existing records, were automatically reflected in the Power BI dashboard.
   - The dashboard provided real-time visual insights into customer segments, purchasing trends, and CLV distribution.
5. **Advanced Analytics with Python:**
   - The cleaned customer data was also imported into Python for further analysis. Python's advanced analytical capabilities were leveraged to perform deeper segmentation analysis and create interactive visualizations.
6. **Limitations:**
   - This project does not cover predictive modelling of future CLV, nor does it delve into customer demographics beyond the basic purchase behavior and historical spending.
   - Additionally, more advanced machine learning techniques for customer segmentation or predictive analytics are beyond the scope of the current work.

Executive Summary:

This project focuses on analyzing customer data from an e-commerce platform to enhance decision-making for targeted marketing. Key tasks included:

- Cleaning inconsistent customer data from Excel sheets.
- Automating data updates to SQL Server.
- Calculating CLV and segmenting customers using SQL.
- Creating dynamic dashboards in Power BI that auto-refresh upon changes in the database.
- Conducting Python analysis to reinforce customer segmentation insights.

Research Methodology:

The research methodology involved a detailed review of customer data analytics, including:

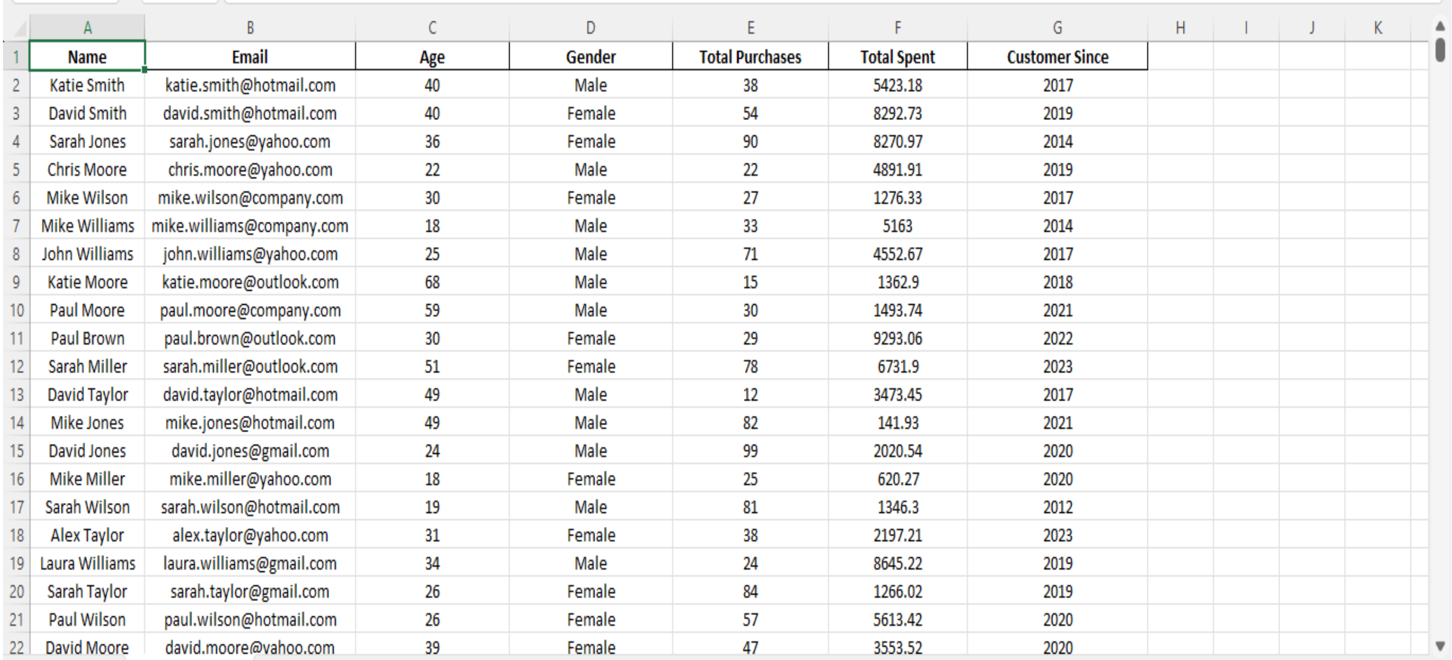
- Best practices for data cleaning.
- Methods for calculating CLV.
- Techniques for customer segmentation.
- Integrating data pipelines across Excel, SQL, Power BI, and Python to create an end-to-end solution.

Code Explanation:

- The raw customer data was cleaned using VBA in Excel. The following steps were automated:
  - **Removing Inconsistent Columns**: Columns like Age and Gender were removed due to inconsistencies such as multiple records with the same name and email but different ages and genders.
  - **Customer ID Creation**: A unique customer ID was created by concatenating the customer's name and email with a hyphen ("-").
  - **Adding Total Purchase and Total Spent**: Duplicate records were identified, and total purchases and total spent were aggregated for each customer.
  - **Keeping Oldest Year**: The 'Member Since' and 'Customer Since' columns were checked, and the oldest year for each customer was retained

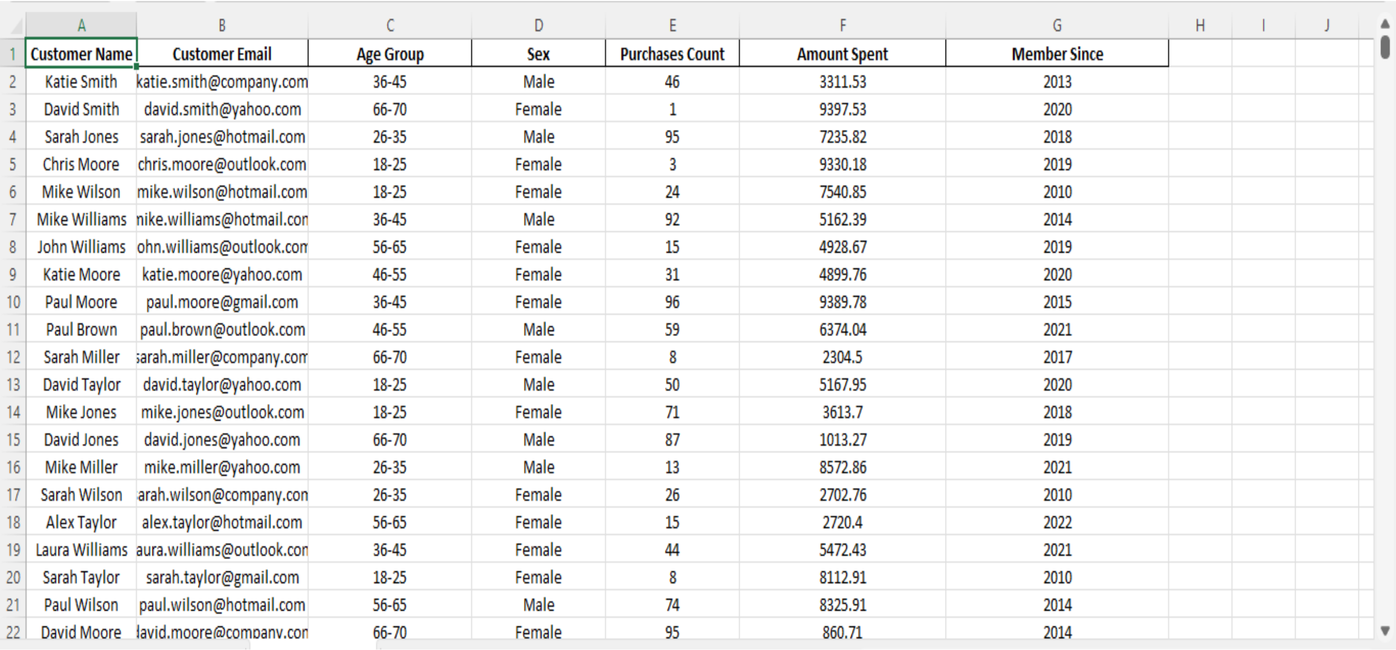
Raw Customer Data –

Customers Sheet:



	A	B	C	D	E	F	G
1	Name	Email	Age	Gender	Total Purchases	Total Spent	Customer Since
2	Katie Smith	katie.smith@hotmail.com	40	Male	38	5423.18	2017
3	David Smith	david.smith@hotmail.com	40	Female	54	8292.73	2019
4	Sarah Jones	sarah.jones@yahoo.com	36	Female	90	8270.97	2014
5	Chris Moore	chris.moore@yahoo.com	22	Male	22	4891.91	2019
6	Mike Wilson	mike.wilson@company.com	30	Female	27	1276.33	2017
7	Mike Williams	mike.williams@company.com	18	Male	33	5163	2014
8	John Williams	john.williams@yahoo.com	25	Male	71	4552.67	2017
9	Katie Moore	katie.moore@outlook.com	68	Male	15	1362.9	2018
10	Paul Moore	paul.moore@company.com	59	Male	30	1493.74	2021
11	Paul Brown	paul.brown@outlook.com	30	Female	29	9293.06	2022
12	Sarah Miller	sarah.miller@outlook.com	51	Female	78	6731.9	2023
13	David Taylor	david.taylor@hotmail.com	49	Male	12	3473.45	2017
14	Mike Jones	mike.jones@hotmail.com	49	Male	82	141.93	2021
15	David Jones	david.jones@gmail.com	24	Male	99	2020.54	2020
16	Mike Miller	mike.miller@yahoo.com	18	Female	25	620.27	2020
17	Sarah Wilson	sarah.wilson@hotmail.com	19	Male	81	1346.3	2012
18	Alex Taylor	alex.taylor@yahoo.com	31	Female	38	2197.21	2023
19	Laura Williams	laura.williams@gmail.com	34	Male	24	8645.22	2019
20	Sarah Taylor	sarah.taylor@gmail.com	26	Female	84	1266.02	2019
21	Paul Wilson	paul.wilson@hotmail.com	26	Female	57	5613.42	2020
22	David Moore	david.moore@yahoo.com	39	Female	47	3553.52	2020

Members Sheet:



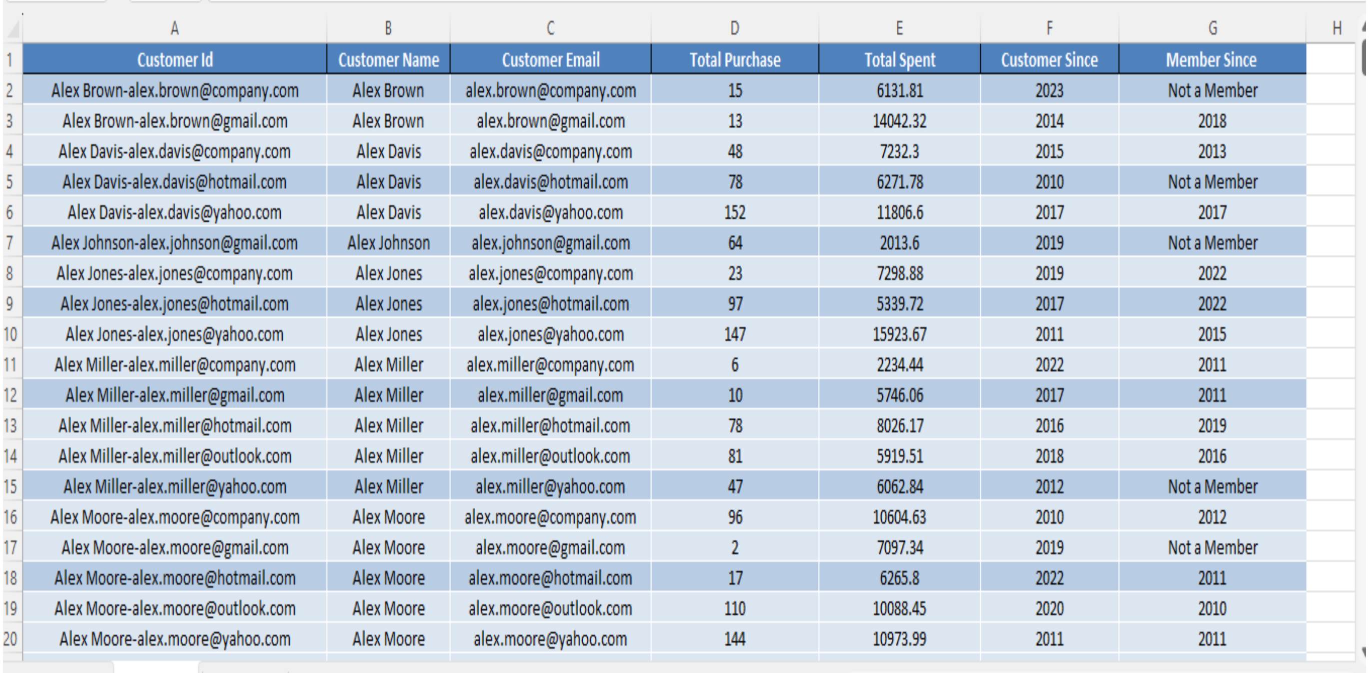
	A	B	C	D	E	F	G	H	I	J
1	Customer Name	Customer Email	Age Group	Sex	Purchases Count	Amount Spent	Member Since			
2	Katie Smith	katie.smith@company.com	36-45	Male	46	3311.53	2013			
3	David Smith	david.smith@yahoo.com	66-70	Female	1	9397.53	2020			
4	Sarah Jones	sarah.jones@hotmail.com	26-35	Male	95	7235.82	2018			
5	Chris Moore	chris.moore@outlook.com	18-25	Female	3	9330.18	2019			
6	Mike Wilson	mike.wilson@hotmail.com	18-25	Female	24	7540.85	2010			
7	Mike Williams	nike.williams@hotmail.con	36-45	Male	92	5162.39	2014			
8	John Williams	ohn.williams@outlook.con	56-65	Female	15	4928.67	2019			
9	Katie Moore	katie.moore@yahoo.com	46-55	Female	31	4899.76	2020			
10	Paul Moore	paul.moore@gmail.com	36-45	Female	96	9389.78	2015			
11	Paul Brown	paul.brown@outlook.com	46-55	Male	59	6374.04	2021			
12	Sarah Miller	sarah.miller@company.con	66-70	Female	8	2304.5	2017			
13	David Taylor	david.taylor@yahoo.com	18-25	Male	50	5167.95	2020			
14	Mike Jones	mike.jones@outlook.com	18-25	Female	71	3613.7	2018			
15	David Jones	david.jones@yahoo.com	66-70	Male	87	1013.27	2019			
16	Mike Miller	mike.miller@yahoo.com	26-35	Male	13	8572.86	2021			
17	Sarah Wilson	arah.wilson@company.con	26-35	Female	26	2702.76	2010			
18	Alex Taylor	alex.taylor@hotmail.com	56-65	Female	15	2720.4	2022			
19	Laura Williams	aura.williams@outlook.con	36-45	Female	44	5472.43	2021			
20	Sarah Taylor	sarah.taylor@gmail.com	18-25	Female	8	8112.91	2010			
21	Paul Wilson	paul.wilson@hotmail.com	56-65	Male	74	8325.91	2014			
22	David Moore	lavid.moore@company.con	66-70	Female	95	860.71	2014			

VBA Code to automate data cleaning process:

```
Sub CleanData()
    Dim wsCustomers As Worksheet
    Dim wsMembers As Worksheet
    Dim lastRowCustomers As Long
    Dim lastRowMembers As Long
    Dim customerID As String
    Dim i As Long, j As Long
    Dim foundID As Range

    ' Set the worksheets
    Set wsCustomers = ThisWorkbook.Sheets("Customers")
    Set wsMembers = ThisWorkbook.Sheets("Members")

    ' Delete the Gender and Age columns in the "Customers" sheet (assuming Gender in column 3 and Age in column 4)
    wsCustomers.Columns(3).Delete
    wsCustomers.Columns(3).Delete ' Age moves to column 3 after Gender is deleted

    ' Delete the Sex and Age Range columns in the "Members" sheet (assuming Sex in column 3 and Age Range in column 4)
    wsMembers.Columns(3).Delete
    wsMembers.Columns(3).Delete ' Age Range moves to column 3 after Sex is deleted

    ' Find the last row in both sheets
    lastRowCustomers = wsCustomers.Cells(wsCustomers.Rows.Count, 1).End(xlUp).Row
    lastRowMembers = wsMembers.Cells(wsMembers.Rows.Count, 1).End(xlUp).Row

    ' Creating Customer-ID for both sheets

    ' Creating Customer-ID in the "customers" sheet
    For i = 2 To lastRowCustomers
        wsCustomers.Cells(i, 9).Value = wsCustomers.Cells(i, 1).Value & "-" & wsCustomers.Cells(i, 2).Value ' Name-Email
    Next i
```

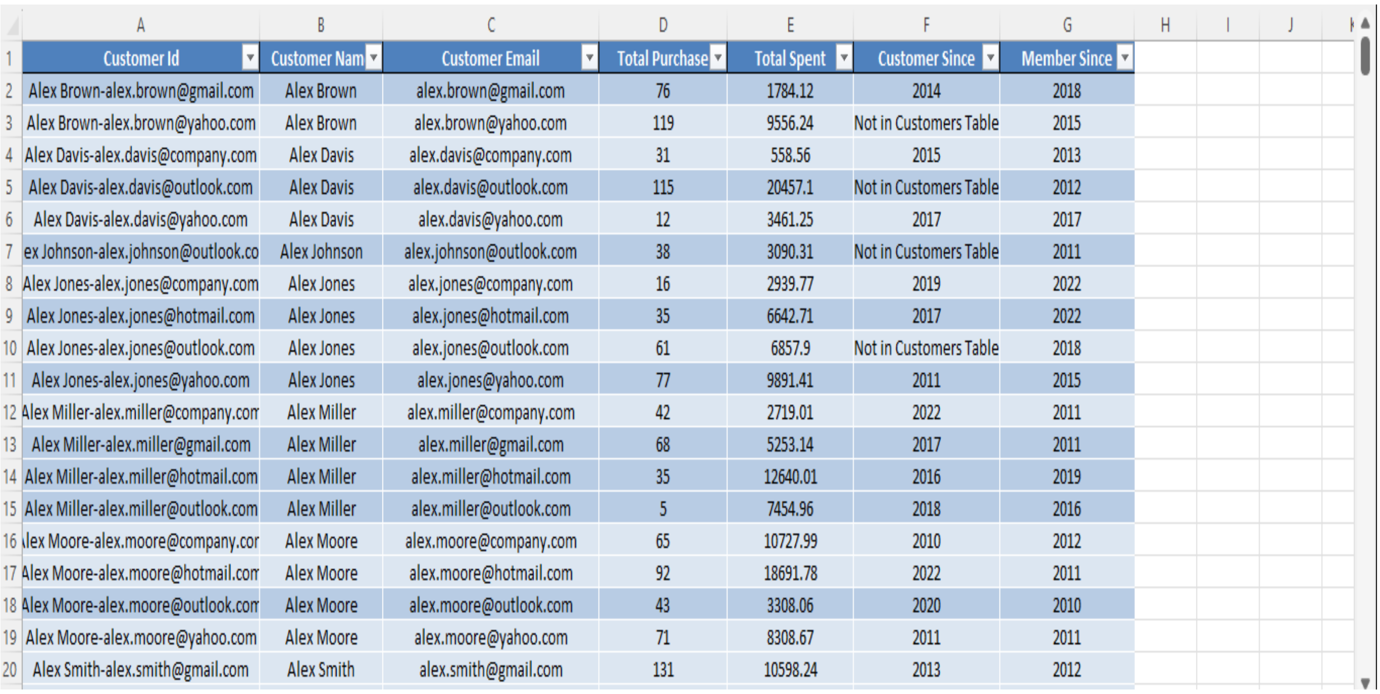
```
' Creating Customer-ID in the "members" sheet
For i = 2 To lastRowMembers
    wsMembers.Cells(i, 9).Value = wsMembers.Cells(i, 1).Value & "-" & wsMembers.Cells(i, 2).Value ' Name-Email
Next i

' Checking and merging duplicates within the same sheet

' Checking for duplicates in the "customers" sheet
For i = 2 To lastRowCustomers
    customerID = wsCustomers.Cells(i, 9).Value ' Customer-ID

    ' Checking for duplicates further down in the "customers" sheet
    For j = i + 1 To lastRowCustomers
        If wsCustomers.Cells(j, 9).Value = customerID Then
            ' Adding Total Purchases and Total Spent
            wsCustomers.Cells(i, 5).Value = wsCustomers.Cells(i, 5).Value + wsCustomers.Cells(j, 5).Value
            wsCustomers.Cells(i, 6).Value = wsCustomers.Cells(i, 6).Value + wsCustomers.Cells(j, 6).Value

            ' Keeping the oldest Customer Since and Member Since year
            If wsCustomers.Cells(i, 7).Value > wsCustomers.Cells(j, 7).Value Then
                wsCustomers.Cells(i, 7).Value = wsCustomers.Cells(j, 7).Value ' Update Customer Since
            End If
            If wsCustomers.Cells(i, 8).Value > wsCustomers.Cells(j, 8).Value Then
                wsCustomers.Cells(i, 8).Value = wsCustomers.Cells(j, 8).Value ' Update Member Since
            End If

            ' Deleting the duplicate row
            wsCustomers.Rows(j).Delete
            lastRowCustomers = lastRowCustomers - 1
            j = j - 1 ' Adjusting loop index after deletion
        End If
    Next j
Next i


    MsgBox "Customer data cleaned and merged within each sheet!"
End Sub
```

- After data cleaning a cross-reference between the sheets is created using XLOOKUP on the 'Customer Since' and 'Member Since' columns.
  So final processed sheets are-
  Customers Table:



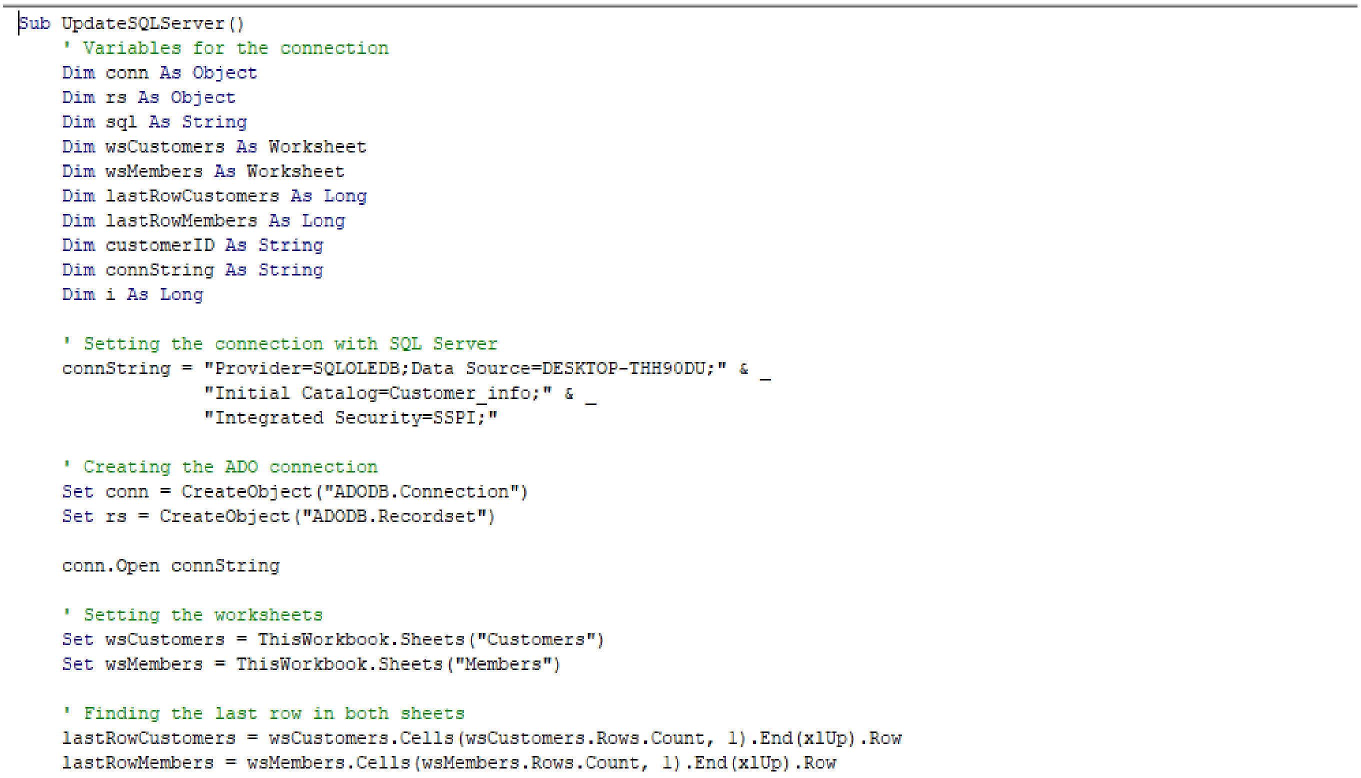
Customer Id	Customer Name	Customer Email	Total Purchase	Total Spent	Customer Since	Member Since
Alex Brown-alex.brown@company.com	Alex Brown	alex.brown@company.com	15	6131.81	2023	Not a Member
Alex Brown-alex.brown@gmail.com	Alex Brown	alex.brown@gmail.com	13	14042.32	2014	2018
Alex Davis-alex.davis@company.com	Alex Davis	alex.davis@company.com	48	7232.3	2015	2013
Alex Davis-alex.davis@hotmail.com	Alex Davis	alex.davis@hotmail.com	78	6271.78	2010	Not a Member
Alex Davis-alex.davis@yahoo.com	Alex Davis	alex.davis@yahoo.com	152	11806.6	2017	2017
Alex Johnson-alex.johnson@gmail.com	Alex Johnson	alex.johnson@gmail.com	64	2013.6	2019	Not a Member
Alex Jones-alex.jones@company.com	Alex Jones	alex.jones@company.com	23	7298.88	2019	2022
Alex Jones-alex.jones@hotmail.com	Alex Jones	alex.jones@hotmail.com	97	5339.72	2017	2022
Alex Jones-alex.jones@yahoo.com	Alex Jones	alex.jones@yahoo.com	147	15923.67	2011	2015
Alex Miller-alex.miller@company.com	Alex Miller	alex.miller@company.com	6	2234.44	2022	2011
Alex Miller-alex.miller@gmail.com	Alex Miller	alex.miller@gmail.com	10	5746.06	2017	2011
Alex Miller-alex.miller@hotmail.com	Alex Miller	alex.miller@hotmail.com	78	8026.17	2016	2019
Alex Miller-alex.miller@outlook.com	Alex Miller	alex.miller@outlook.com	81	5919.51	2018	2016
Alex Miller-alex.miller@yahoo.com	Alex Miller	alex.miller@yahoo.com	47	6062.84	2012	Not a Member
Alex Moore-alex.moore@company.com	Alex Moore	alex.moore@company.com	96	10604.63	2010	2012
Alex Moore-alex.moore@gmail.com	Alex Moore	alex.moore@gmail.com	2	7097.34	2019	Not a Member
Alex Moore-alex.moore@hotmail.com	Alex Moore	alex.moore@hotmail.com	17	6265.8	2022	2011
Alex Moore-alex.moore@outlook.com	Alex Moore	alex.moore@outlook.com	110	10088.45	2020	2010
Alex Moore-alex.moore@yahoo.com	Alex Moore	alex.moore@yahoo.com	144	10973.99	2011	2011

Members Table:

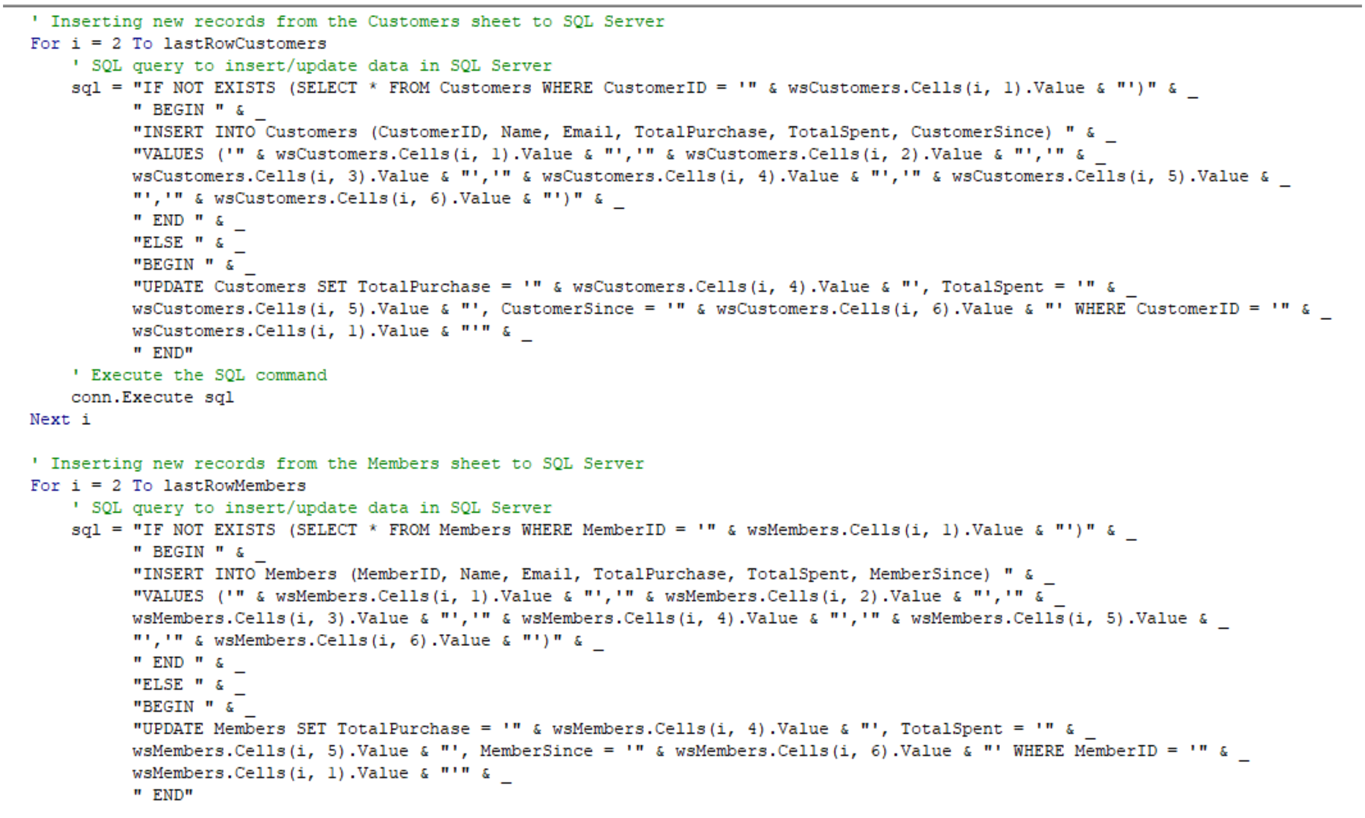


	Customer Id	Customer Nam	Customer Email	Total Purchase	Total Spent	Customer Since	Member Since
2	Alex Brown-alex.brown@gmail.com	Alex Brown	alex.brown@gmail.com	76	1784.12	2014	2018
3	Alex Brown-alex.brown@yahoo.com	Alex Brown	alex.brown@yahoo.com	119	9556.24	Not in Customers Table	2015
4	Alex Davis-alex.davis@company.com	Alex Davis	alex.davis@company.com	31	558.56	2015	2013
5	Alex Davis-alex.davis@outlook.com	Alex Davis	alex.davis@outlook.com	115	20457.1	Not in Customers Table	2012
6	Alex Davis-alex.davis@yahoo.com	Alex Davis	alex.davis@yahoo.com	12	3461.25	2017	2017
7	ex Johnson-alex.johnson@outlook.co	Alex Johnson	alex.johnson@outlook.com	38	3090.31	Not in Customers Table	2011
8	Alex Jones-alex.jones@company.com	Alex Jones	alex.jones@company.com	16	2939.77	2019	2022
9	Alex Jones-alex.jones@hotmail.com	Alex Jones	alex.jones@hotmail.com	35	6642.71	2017	2022
10	Alex Jones-alex.jones@outlook.com	Alex Jones	alex.jones@outlook.com	61	6857.9	Not in Customers Table	2018
11	Alex Jones-alex.jones@yahoo.com	Alex Jones	alex.jones@yahoo.com	77	9891.41	2011	2015
12	Alex Miller-alex.miller@company.com	Alex Miller	alex.miller@company.com	42	2719.01	2022	2011
13	Alex Miller-alex.miller@gmail.com	Alex Miller	alex.miller@gmail.com	68	5253.14	2017	2011
14	Alex Miller-alex.miller@hotmail.com	Alex Miller	alex.miller@hotmail.com	35	12640.01	2016	2019
15	Alex Miller-alex.miller@outlook.com	Alex Miller	alex.miller@outlook.com	5	7454.96	2018	2016
16	lex Moore-alex.moore@company.cor	Alex Moore	alex.moore@company.com	65	10727.99	2010	2012
17	Alex Moore-alex.moore@hotmail.com	Alex Moore	alex.moore@hotmail.com	92	18691.78	2022	2011
18	Alex Moore-alex.moore@outlook.com	Alex Moore	alex.moore@outlook.com	43	3308.06	2020	2010
19	Alex Moore-alex.moore@yahoo.com	Alex Moore	alex.moore@yahoo.com	71	8308.67	2011	2011
20	Alex Smith-alex.smith@gmail.com	Alex Smith	alex.smith@gmail.com	131	10598.24	2013	2012

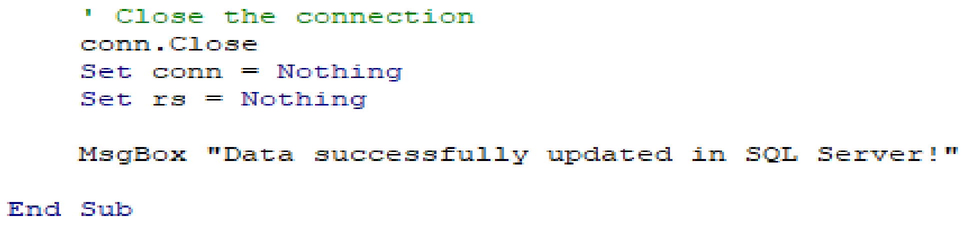
- The cleaned Excel data from the Customers and Members sheets was then connected to SQL Server using VBA. This automated the process of updating the SQL database whenever new customer records were added to the Excel sheets.
  VBA Code is –



```
Sub UpdateSQLServer()
    ' Variables for the connection
    Dim conn As Object
    Dim rs As Object
    Dim sql As String
    Dim wsCustomers As Worksheet
    Dim wsMembers As Worksheet
    Dim lastRowCustomers As Long
    Dim lastRowMembers As Long
    Dim customerID As String
    Dim connString As String
    Dim i As Long

    ' Setting the connection with SQL Server
    connString = "Provider=SQLOLEDB;Data Source=DESKTOP-THH90DU;" & _
                 "Initial Catalog=Customer_info;" & _
                 "Integrated Security=SSPI;"

    ' Creating the ADO connection
    Set conn = CreateObject("ADODB.Connection")
    Set rs = CreateObject("ADODB.Recordset")

    conn.Open connString

    ' Setting the worksheets
    Set wsCustomers = ThisWorkbook.Sheets("Customers")
    Set wsMembers = ThisWorkbook.Sheets("Members")

    ' Finding the last row in both sheets
    lastRowCustomers = wsCustomers.Cells(wsCustomers.Rows.Count, 1).End(xlUp).Row
    lastRowMembers = wsMembers.Cells(wsMembers.Rows.Count, 1).End(xlUp).Row
```

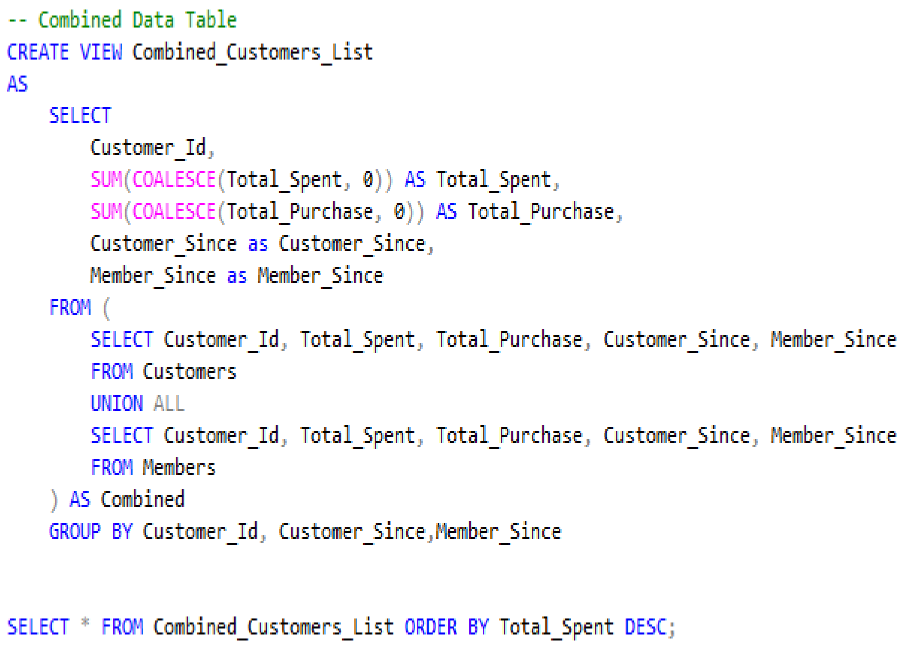


```
' Inserting new records from the Customers sheet to SQL Server
For i = 2 To lastRowCustomers
    ' SQL query to insert/update data in SQL Server
    sql = "IF NOT EXISTS (SELECT * FROM Customers WHERE CustomerID = '" & wsCustomers.Cells(i, 1).Value & "')" & _
          " BEGIN " & _
          "INSERT INTO Customers (CustomerID, Name, Email, TotalPurchase, TotalSpent, CustomerSince) " & _
          "VALUES ('" & wsCustomers.Cells(i, 1).Value & "','" & wsCustomers.Cells(i, 2).Value & "','" & _
          wsCustomers.Cells(i, 3).Value & "','" & wsCustomers.Cells(i, 4).Value & "','" & wsCustomers.Cells(i, 5).Value & _
          "','" & wsCustomers.Cells(i, 6).Value & "')" & _
          " END " & _
          "ELSE " & _
          "BEGIN " & _
          "UPDATE Customers SET TotalPurchase = '" & wsCustomers.Cells(i, 4).Value & "', TotalSpent = '" & _
          wsCustomers.Cells(i, 5).Value & "', CustomerSince = '" & wsCustomers.Cells(i, 6).Value & "' WHERE CustomerID = '" & _
          wsCustomers.Cells(i, 1).Value & "'" & _
          " END"
    ' Execute the SQL command
    conn.Execute sql
Next i

' Inserting new records from the Members sheet to SQL Server
For i = 2 To lastRowMembers
    ' SQL query to insert/update data in SQL Server
    sql = "IF NOT EXISTS (SELECT * FROM Members WHERE MemberID = '" & wsMembers.Cells(i, 1).Value & "')" & _
          " BEGIN " & _
          "INSERT INTO Members (MemberID, Name, Email, TotalPurchase, TotalSpent, MemberSince) " & _
          "VALUES ('" & wsMembers.Cells(i, 1).Value & "','" & wsMembers.Cells(i, 2).Value & "','" & _
          wsMembers.Cells(i, 3).Value & "','" & wsMembers.Cells(i, 4).Value & "','" & wsMembers.Cells(i, 5).Value & _
          "','" & wsMembers.Cells(i, 6).Value & "')" & _
          " END " & _
          "ELSE " & _
          "BEGIN " & _
          "UPDATE Members SET TotalPurchase = '" & wsMembers.Cells(i, 4).Value & "', TotalSpent = '" & _
          wsMembers.Cells(i, 5).Value & "', MemberSince = '" & wsMembers.Cells(i, 6).Value & "' WHERE MemberID = '" & _
          wsMembers.Cells(i, 1).Value & "'" & _
          " END"
```

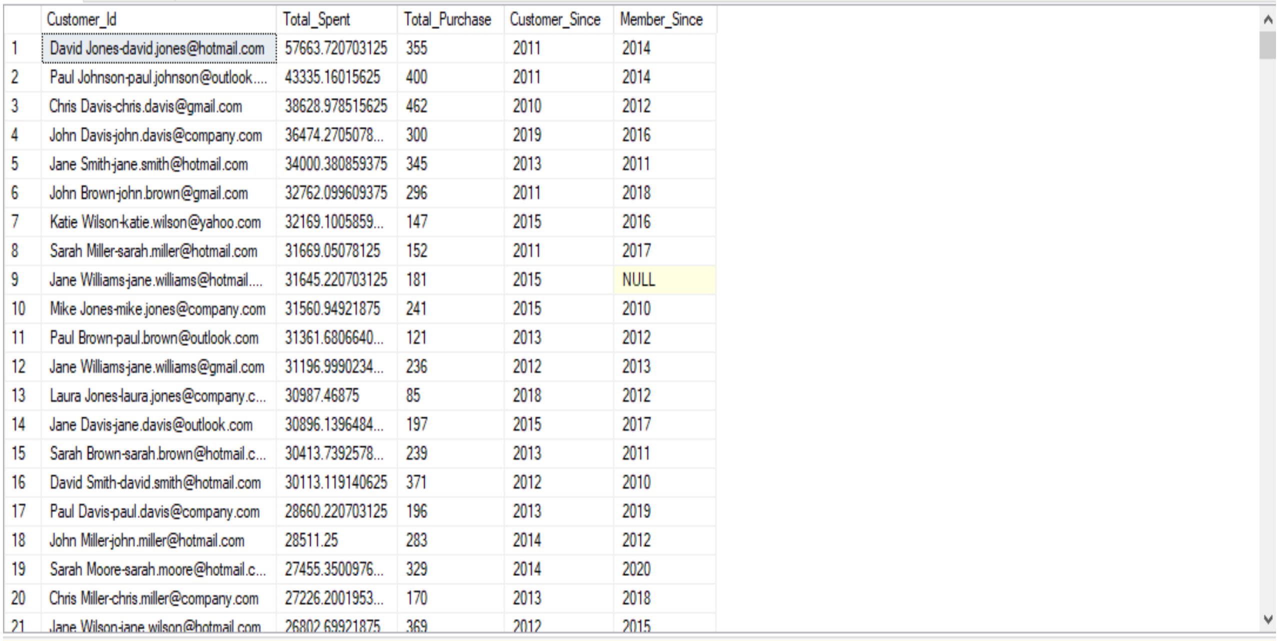


```
    ' Close the connection
    conn.Close
    Set conn = Nothing
    Set rs = Nothing

    MsgBox "Data successfully updated in SQL Server!"

End Sub
```

- In SQL Server, the following analysis were performed:
  - **Combined Customer Table View(‘Combined_Customers_List’)**: A combined table was created to merge records from both Customers and Members tables, resolving inconsistencies in duplicated customer data, also the total purchase and total spent was added for the duplicate customer ID.

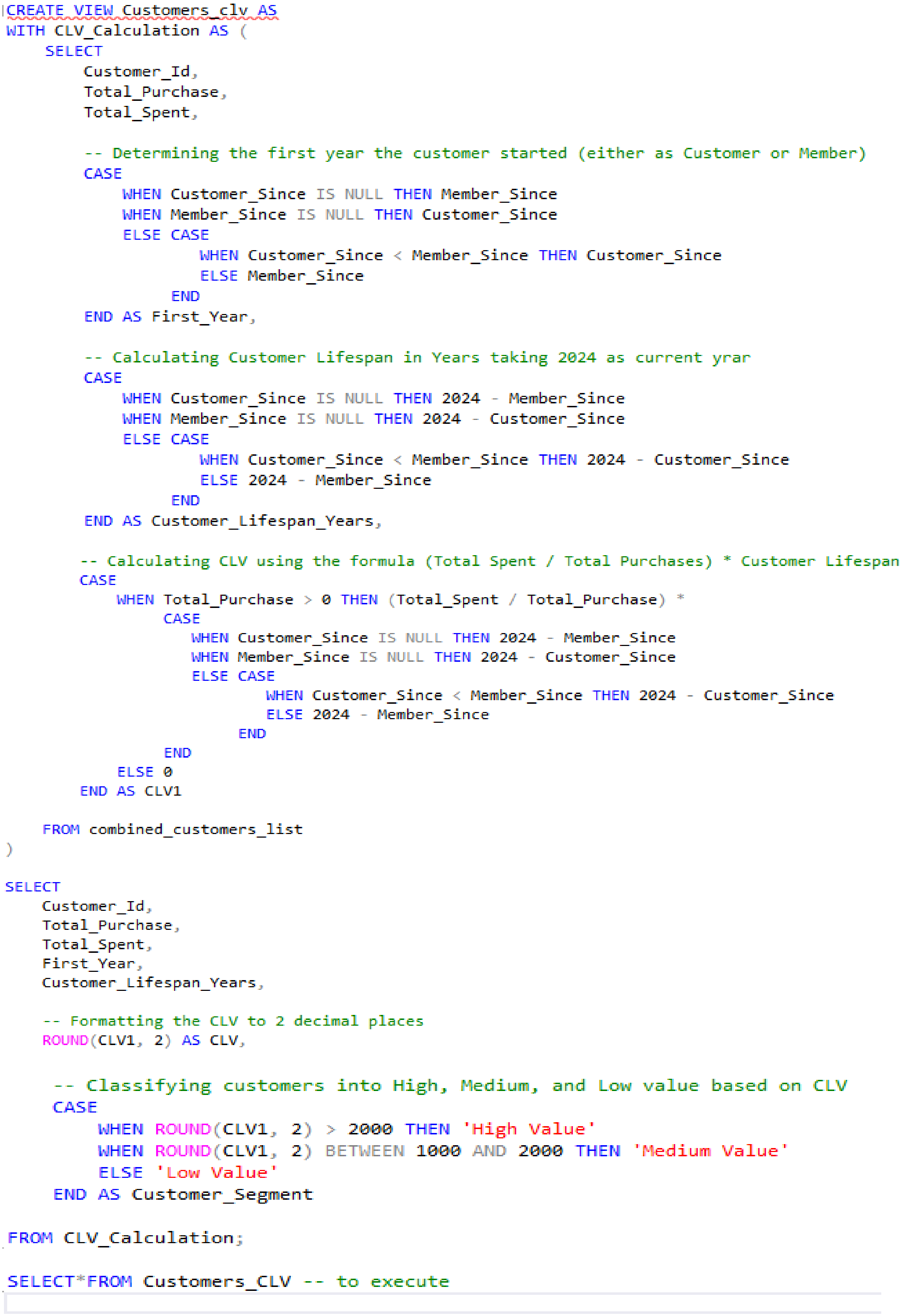


```
-- Combined Data Table
CREATE VIEW Combined_Customers_List
AS
    SELECT
        Customer_Id,
        SUM(COALESCE(Total_Spent, 0)) AS Total_Spent,
        SUM(COALESCE(Total_Purchase, 0)) AS Total_Purchase,
        Customer_Since as Customer_Since,
        Member_Since as Member_Since
    FROM (
        SELECT Customer_Id, Total_Spent, Total_Purchase, Customer_Since, Member_Since
        FROM Customers
        UNION ALL
        SELECT Customer_Id, Total_Spent, Total_Purchase, Customer_Since, Member_Since
        FROM Members
    ) AS Combined
    GROUP BY Customer_Id, Customer_Since,Member_Since


SELECT * FROM Combined_Customers_List ORDER BY Total_Spent DESC;
```

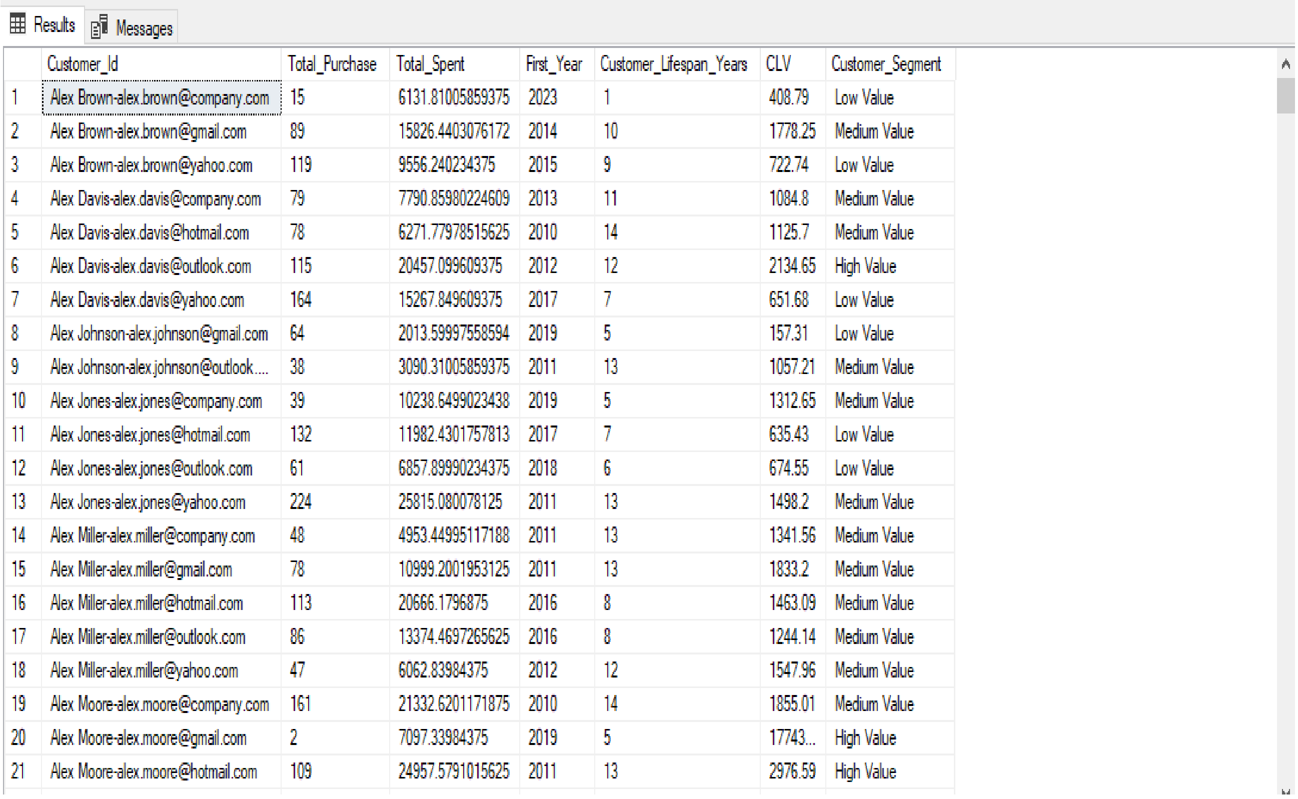


	Customer_Id	Total_Spent	Total_Purchase	Customer_Since	Member_Since
1	David Jones-david.jones@hotmail.com	57663.720703125	355	2011	2014
2	Paul Johnson-paul.johnson@outlook....	43335.16015625	400	2011	2014
3	Chris Davis-chris.davis@gmail.com	38628.978515625	462	2010	2012
4	John Davis-john.davis@company.com	36474.2705078...	300	2019	2016
5	Jane Smith-jane.smith@hotmail.com	34000.380859375	345	2013	2011
6	John Brown-john.brown@gmail.com	32762.099609375	296	2011	2018
7	Katie Wilson-katie.wilson@yahoo.com	32169.1005859...	147	2015	2016
8	Sarah Miller-sarah.miller@hotmail.com	31669.05078125	152	2011	2017
9	Jane Williams-jane.williams@hotmail....	31645.220703125	181	2015	NULL
10	Mike Jones-mike.jones@company.com	31560.94921875	241	2015	2010
11	Paul Brown-paul.brown@outlook.com	31361.6806640...	121	2013	2012
12	Jane Williams-jane.williams@gmail.com	31196.9990234...	236	2012	2013
13	Laura Jones-laura.jones@company.c...	30987.46875	85	2018	2012
14	Jane Davis-jane.davis@outlook.com	30896.1396484...	197	2015	2017
15	Sarah Brown-sarah.brown@hotmail.c...	30413.7392578...	239	2013	2011
16	David Smith-david.smith@hotmail.com	30113.119140625	371	2012	2010
17	Paul Davis-paul.davis@company.com	28660.220703125	196	2013	2019
18	John Miller-john.miller@hotmail.com	28511.25	283	2014	2012
19	Sarah Moore-sarah.moore@hotmail.c...	27455.3500976...	329	2014	2020
20	Chris Miller-chris.miller@company.com	27226.2001953...	170	2013	2018
21	Jane Wilson-jane.wilson@hotmail.com	26802.69921875	369	2012	2015

- **Customer Lifetime Value (CLV) Table View('Customers_clv'):** To Calculate each customer's CLV and segment them based on CLV score into high-value, medium-value, and low-value customers.

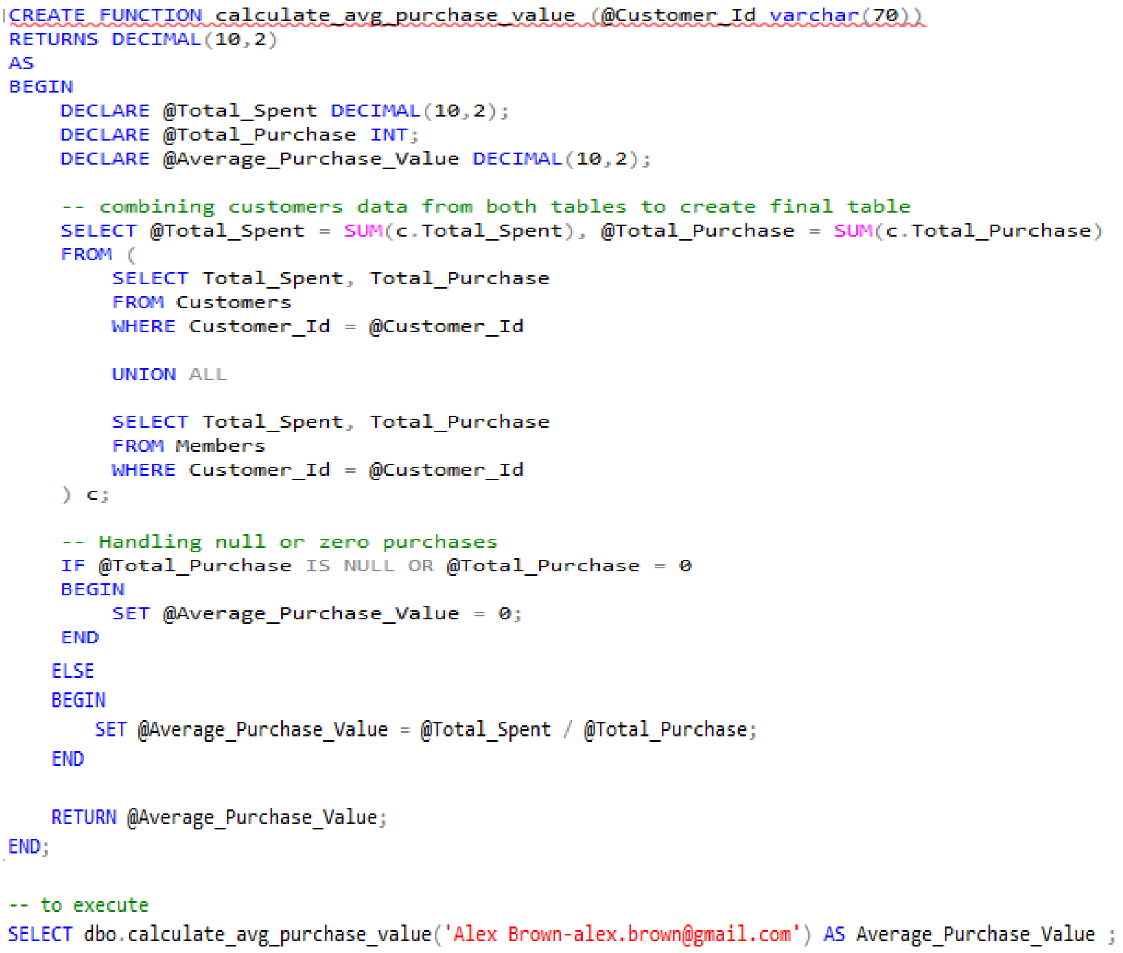


```
CREATE VIEW Customers_clv AS
WITH CLV_Calculation AS (
    SELECT
        Customer_Id,
        Total_Purchase,
        Total_Spent,

        -- Determining the first year the customer started (either as Customer or Member)
        CASE
            WHEN Customer_Since IS NULL THEN Member_Since
            WHEN Member_Since IS NULL THEN Customer_Since
            ELSE CASE
                    WHEN Customer_Since < Member_Since THEN Customer_Since
                    ELSE Member_Since
                 END
        END AS First_Year,

        -- Calculating Customer Lifespan in Years taking 2024 as current yrar
        CASE
            WHEN Customer_Since IS NULL THEN 2024 - Member_Since
            WHEN Member_Since IS NULL THEN 2024 - Customer_Since
            ELSE CASE
                    WHEN Customer_Since < Member_Since THEN 2024 - Customer_Since
                    ELSE 2024 - Member_Since
                 END
        END AS Customer_Lifespan_Years,

        -- Calculating CLV using the formula (Total Spent / Total Purchases) * Customer Lifespan
        CASE
            WHEN Total_Purchase > 0 THEN (Total_Spent / Total_Purchase) *
                 CASE
                    WHEN Customer_Since IS NULL THEN 2024 - Member_Since
                    WHEN Member_Since IS NULL THEN 2024 - Customer_Since
                    ELSE CASE
                            WHEN Customer_Since < Member_Since THEN 2024 - Customer_Since
                            ELSE 2024 - Member_Since
                         END
                 END
            ELSE 0
        END AS CLV1

    FROM combined_customers_list
)

SELECT
    Customer_Id,
    Total_Purchase,
    Total_Spent,
    First_Year,
    Customer_Lifespan_Years,

    -- Formatting the CLV to 2 decimal places
    ROUND(CLV1, 2) AS CLV,

    -- Classifying customers into High, Medium, and Low value based on CLV
    CASE
        WHEN ROUND(CLV1, 2) > 2000 THEN 'High Value'
        WHEN ROUND(CLV1, 2) BETWEEN 1000 AND 2000 THEN 'Medium Value'
        ELSE 'Low Value'
    END AS Customer_Segment

FROM CLV_Calculation;

SELECT*FROM Customers_CLV -- to execute
```

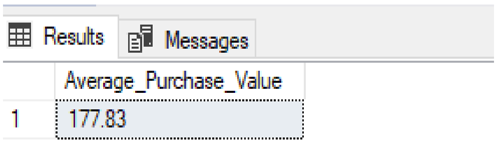


	Customer_Id	Total_Purchase	Total_Spent	First_Year	Customer_Lifespan_Years	CLV	Customer_Segment
1	Alex Brown-alex.brown@company.com	15	6131.81005859375	2023	1	408.79	Low Value
2	Alex Brown-alex.brown@gmail.com	89	15826.4403076172	2014	10	1778.25	Medium Value
3	Alex Brown-alex.brown@yahoo.com	119	9556.240234375	2015	9	722.74	Low Value
4	Alex Davis-alex.davis@company.com	79	7790.85980224609	2013	11	1084.8	Medium Value
5	Alex Davis-alex.davis@hotmail.com	78	6271.77978515625	2010	14	1125.7	Medium Value
6	Alex Davis-alex.davis@outlook.com	115	20457.099609375	2012	12	2134.65	High Value
7	Alex Davis-alex.davis@yahoo.com	164	15267.849609375	2017	7	651.68	Low Value
8	Alex Johnson-alex.johnson@gmail.com	64	2013.59997558594	2019	5	157.31	Low Value
9	Alex Johnson-alex.johnson@outlook....	38	3090.31005859375	2011	13	1057.21	Medium Value
10	Alex Jones-alex.jones@company.com	39	10238.6499023438	2019	5	1312.65	Medium Value
11	Alex Jones-alex.jones@hotmail.com	132	11982.4301757813	2017	7	635.43	Low Value
12	Alex Jones-alex.jones@outlook.com	61	6857.89990234375	2018	6	674.55	Low Value
13	Alex Jones-alex.jones@yahoo.com	224	25815.080078125	2011	13	1498.2	Medium Value
14	Alex Miller-alex.miller@company.com	48	4953.44995117188	2011	13	1341.56	Medium Value
15	Alex Miller-alex.miller@gmail.com	78	10999.2001953125	2011	13	1833.2	Medium Value
16	Alex Miller-alex.miller@hotmail.com	113	20666.1796875	2016	8	1463.09	Medium Value
17	Alex Miller-alex.miller@outlook.com	86	13374.4697265625	2016	8	1244.14	Medium Value
18	Alex Miller-alex.miller@yahoo.com	47	6062.83984375	2012	12	1547.96	Medium Value
19	Alex Moore-alex.moore@company.com	161	21332.6201171875	2010	14	1855.01	Medium Value
20	Alex Moore-alex.moore@gmail.com	2	7097.33984375	2019	5	17743...	High Value
21	Alex Moore-alex.moore@hotmail.com	109	24957.5791015625	2011	13	2976.59	High Value

- ### Function to calculate the Average Purchase Value of each customer ('calculate_avg_purchase_value'):

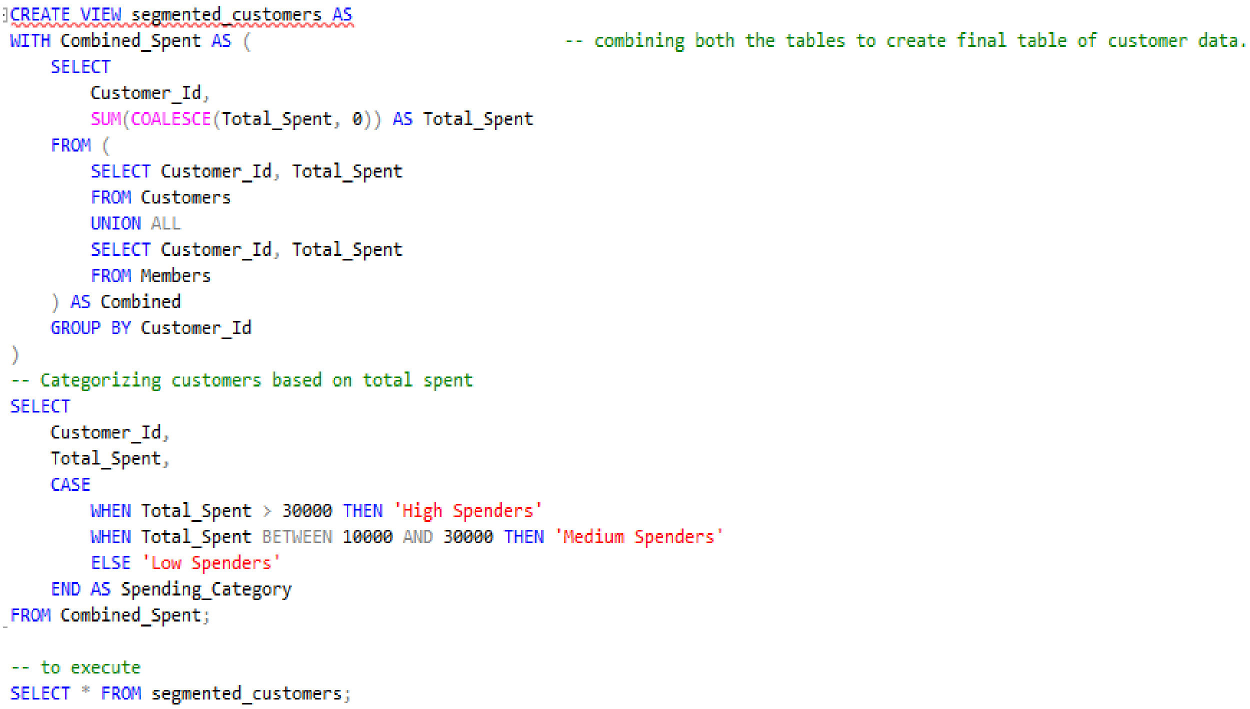


```
CREATE FUNCTION calculate_avg_purchase_value (@Customer_Id varchar(70))
RETURNS DECIMAL(10,2)
AS
BEGIN
    DECLARE @Total_Spent DECIMAL(10,2);
    DECLARE @Total_Purchase INT;
    DECLARE @Average_Purchase_Value DECIMAL(10,2);

    -- combining customers data from both tables to create final table
    SELECT @Total_Spent = SUM(c.Total_Spent), @Total_Purchase = SUM(c.Total_Purchase)
    FROM (
        SELECT Total_Spent, Total_Purchase
        FROM Customers
        WHERE Customer_Id = @Customer_Id

        UNION ALL

        SELECT Total_Spent, Total_Purchase
        FROM Members
        WHERE Customer_Id = @Customer_Id
    ) c;

    -- Handling null or zero purchases
    IF @Total_Purchase IS NULL OR @Total_Purchase = 0
    BEGIN
        SET @Average_Purchase_Value = 0;
    END
    ELSE
    BEGIN
        SET @Average_Purchase_Value = @Total_Spent / @Total_Purchase;
    END

    RETURN @Average_Purchase_Value;
END;

-- to execute
SELECT dbo.calculate_avg_purchase_value('Alex Brown-alex.brown@gmail.com') AS Average_Purchase_Value ;
```

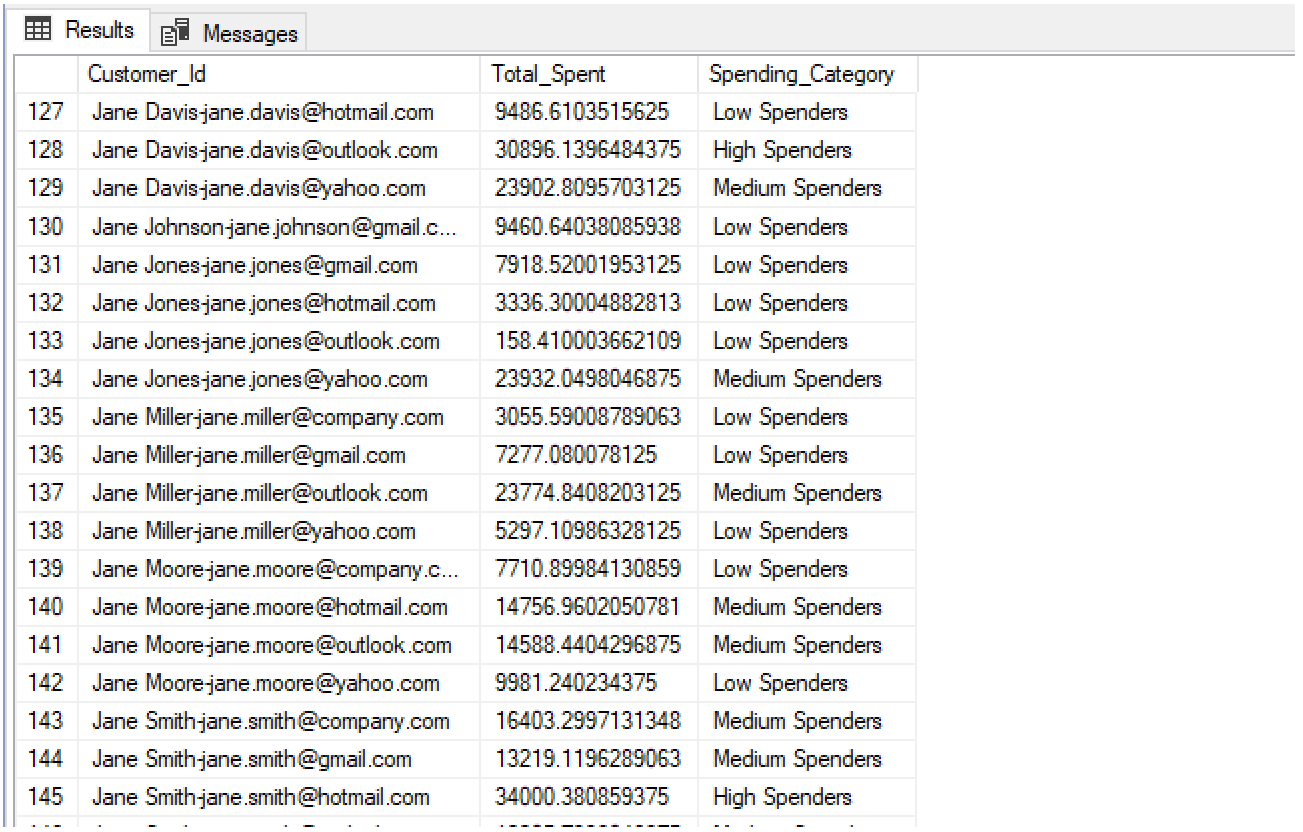


	Average_Purchase_Value
1	177.83

- **View to segment customers based on their spending**: Segmenting customers into high spenders, medium spenders, and low spenders based on their total spent.



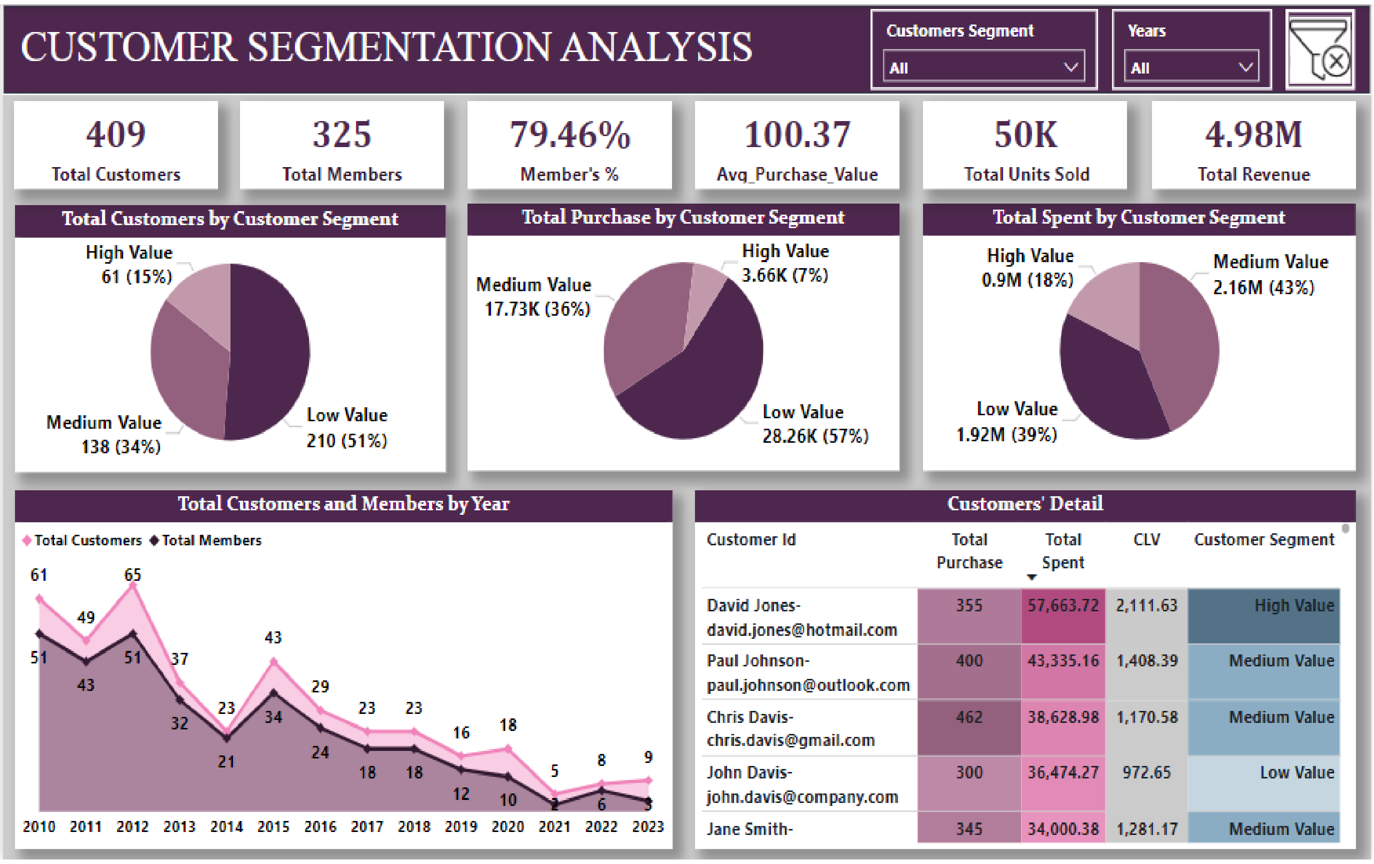
```
CREATE VIEW segmented_customers AS
WITH Combined_Spent AS (                    -- combining both the tables to create final table of customer data.
    SELECT
        Customer_Id,
        SUM(COALESCE(Total_Spent, 0)) AS Total_Spent
    FROM (
        SELECT Customer_Id, Total_Spent
        FROM Customers
        UNION ALL
        SELECT Customer_Id, Total_Spent
        FROM Members
    ) AS Combined
    GROUP BY Customer_Id
)
-- Categorizing customers based on total spent
SELECT
    Customer_Id,
    Total_Spent,
    CASE
        WHEN Total_Spent > 30000 THEN 'High Spenders'
        WHEN Total_Spent BETWEEN 10000 AND 30000 THEN 'Medium Spenders'
        ELSE 'Low Spenders'
    END AS Spending_Category
FROM Combined_Spent;

-- to execute
SELECT * FROM segmented_customers;
```



	Customer_Id	Total_Spent	Spending_Category
127	Jane Davis-jane.davis@hotmail.com	9486.6103515625	Low Spenders
128	Jane Davis-jane.davis@outlook.com	30896.1396484375	High Spenders
129	Jane Davis-jane.davis@yahoo.com	23902.8095703125	Medium Spenders
130	Jane Johnson-jane.johnson@gmail.c...	9460.64038085938	Low Spenders
131	Jane Jones-jane.jones@gmail.com	7918.52001953125	Low Spenders
132	Jane Jones-jane.jones@hotmail.com	3336.30004882813	Low Spenders
133	Jane Jones-jane.jones@outlook.com	158.410003662109	Low Spenders
134	Jane Jones-jane.jones@yahoo.com	23932.0498046875	Medium Spenders
135	Jane Miller-jane.miller@company.com	3055.59008789063	Low Spenders
136	Jane Miller-jane.miller@gmail.com	7277.080078125	Low Spenders
137	Jane Miller-jane.miller@outlook.com	23774.8408203125	Medium Spenders
138	Jane Miller-jane.miller@yahoo.com	5297.10986328125	Low Spenders
139	Jane Moore-jane.moore@company.c...	7710.89984130859	Low Spenders
140	Jane Moore-jane.moore@hotmail.com	14756.9602050781	Medium Spenders
141	Jane Moore-jane.moore@outlook.com	14588.4404296875	Medium Spenders
142	Jane Moore-jane.moore@yahoo.com	9981.240234375	Low Spenders
143	Jane Smith-jane.smith@company.com	16403.2997131348	Medium Spenders
144	Jane Smith-jane.smith@gmail.com	13219.1196289063	Medium Spenders
145	Jane Smith-jane.smith@hotmail.com	34000.380859375	High Spenders

- The SQL Server tables were connected to Power BI via Direct Query mode. This ensures that any updates in the SQL database(‘Customer_info’) are automatically reflected in Power BI.
    - Various measures were created using DAX formulas such as COUNTA, DIVIDE, SUM, and CALCULATE.

Dashboard Showcase:



CUSTOMER SEGMENTATION ANALYSIS

Customers Segment: All

Years: All

409	325	79.46%	100.37	50K	4.98M
Total Customers	Total Members	Member's %	Avg_Purchase_Value	Total Units Sold	Total Revenue

Total Customers by Customer Segment

Total Purchase by Customer Segment

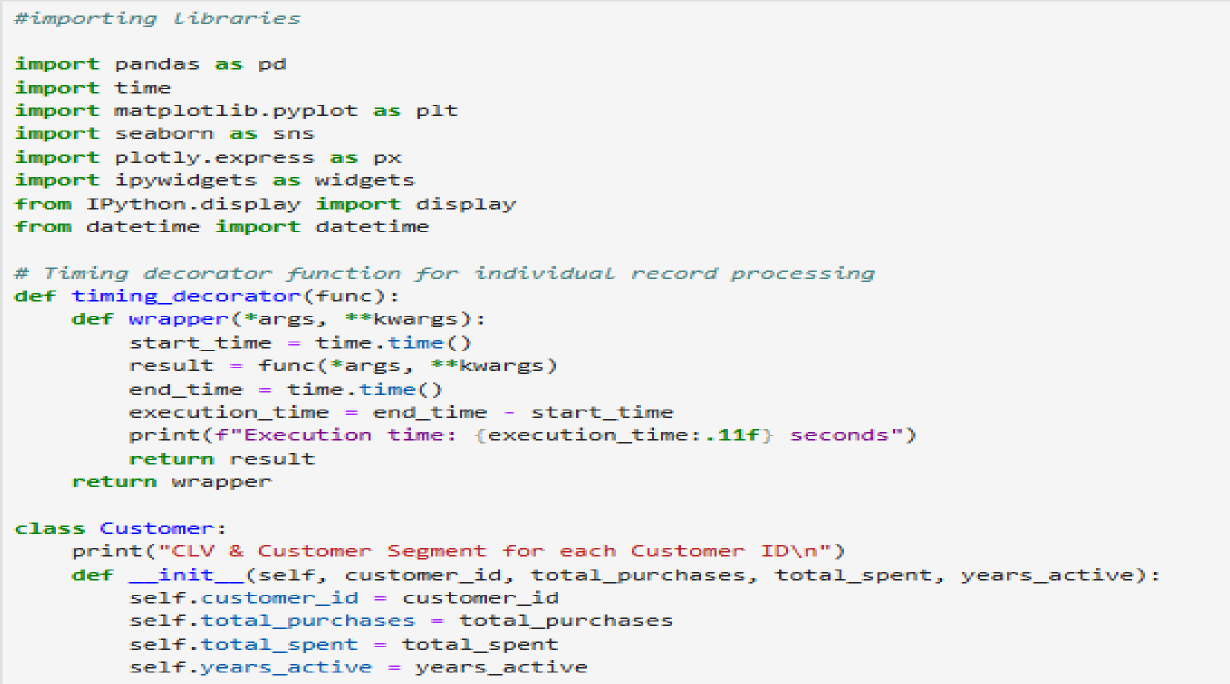
Total Spent by Customer Segment

Total Customers and Members by Year

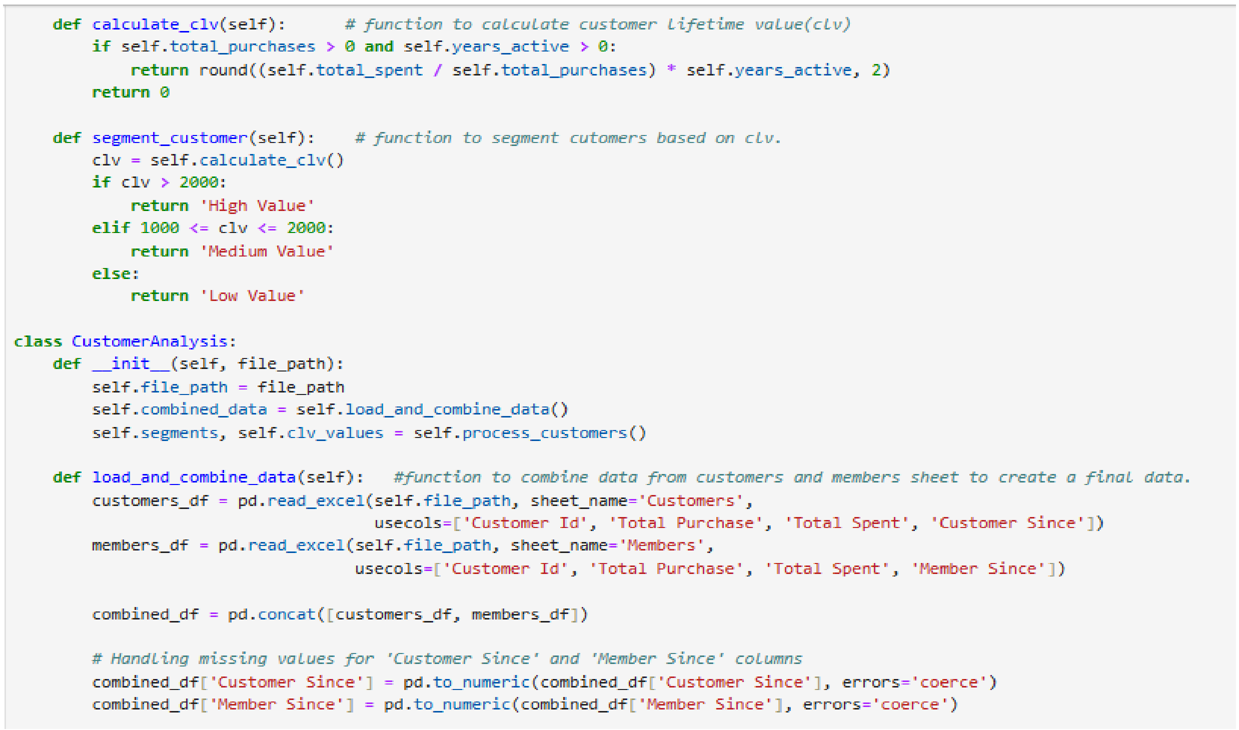
Customers' Detail

Customer Id	Total Purchase	Total Spent	CLV	Customer Segment
David Jones-david.jones@hotmail.com	355	57,663.72	2,111.63	High Value
Paul Johnson-paul.johnson@outlook.com	400	43,335.16	1,408.39	Medium Value
Chris Davis-chris.davis@gmail.com	462	38,628.98	1,170.58	Medium Value
John Davis-john.davis@company.com	300	36,474.27	972.65	Low Value
Jane Smith-	345	34,000.38	1,281.17	Medium Value

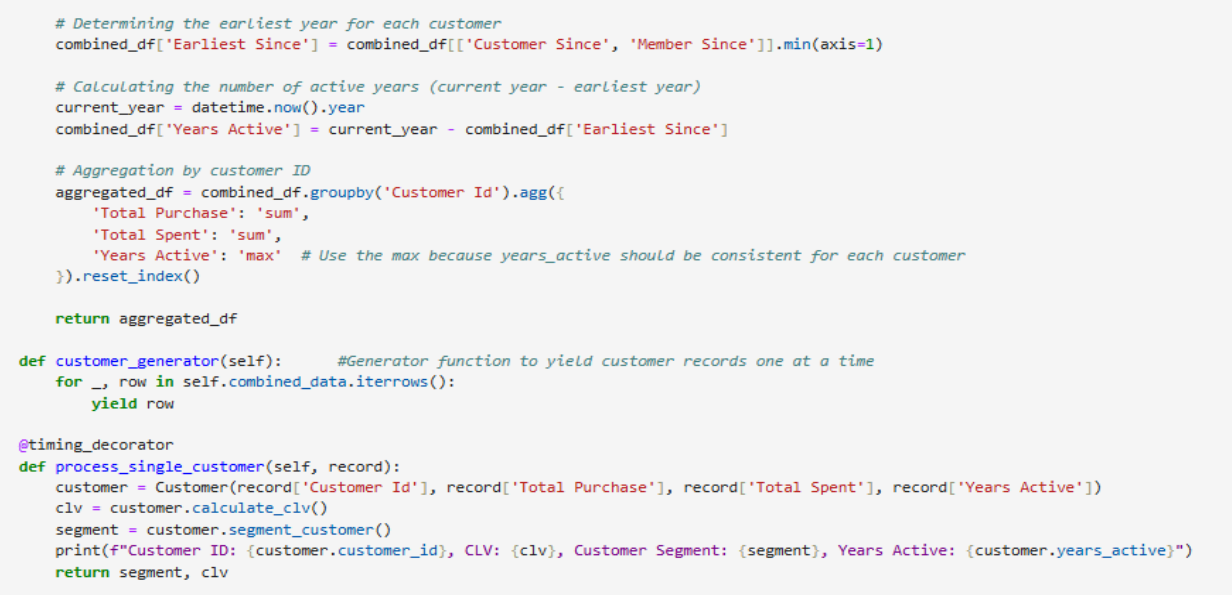
- The cleaned Excel data was also imported into Python, where additional segmentation and analysis were performed. Using Python libraries, interactive visualizations were created to support the segmentation based on CLV.
    - A statistical summary for numerical columns, such as total purchase, total spent, and CLV score, was generated.
    - The correlation matrix for total purchase and total spent was calculated.
    - The relation between purchase and spent was also analyzed and visualized via scatter plot.



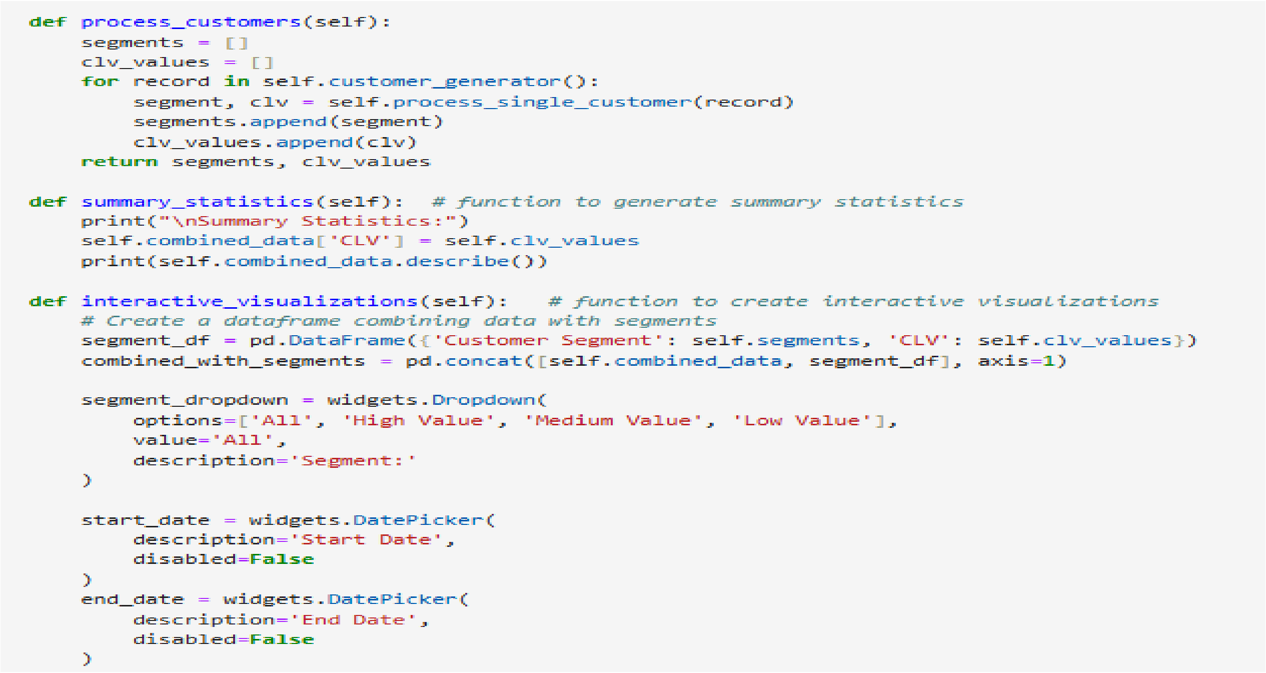
```python
#importing libraries

import pandas as pd
import time
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import ipywidgets as widgets
from IPython.display import display
from datetime import datetime

# Timing decorator function for individual record processing
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: {execution_time:.11f} seconds")
        return result
    return wrapper

class Customer:
    print("CLV & Customer Segment for each Customer ID\n")
    def __init__(self, customer_id, total_purchases, total_spent, years_active):
        self.customer_id = customer_id
        self.total_purchases = total_purchases
        self.total_spent = total_spent
        self.years_active = years_active
```



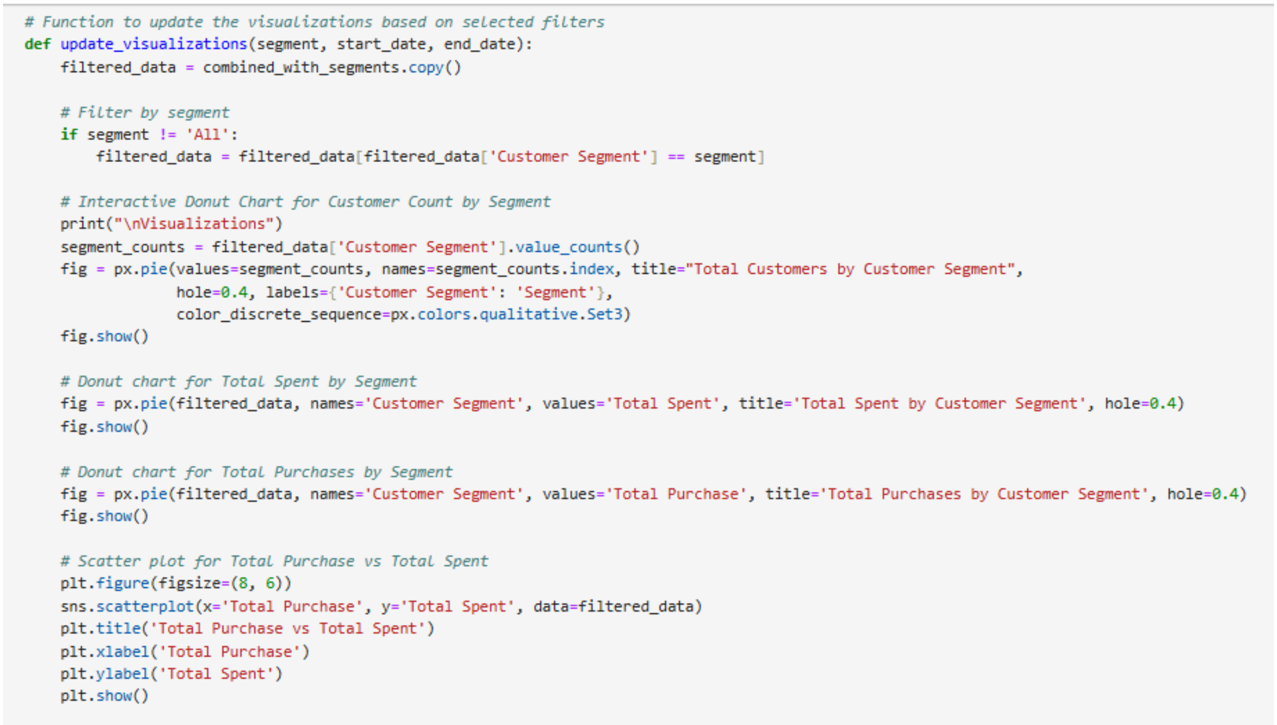
```python
    def calculate_clv(self):      # function to calculate customer lifetime value(clv)
        if self.total_purchases > 0 and self.years_active > 0:
            return round((self.total_spent / self.total_purchases) * self.years_active, 2)
        return 0

    def segment_customer(self):    # function to segment cutomers based on clv.
        clv = self.calculate_clv()
        if clv > 2000:
            return 'High Value'
        elif 1000 <= clv <= 2000:
            return 'Medium Value'
        else:
            return 'Low Value'

class CustomerAnalysis:
    def __init__(self, file_path):
        self.file_path = file_path
        self.combined_data = self.load_and_combine_data()
        self.segments, self.clv_values = self.process_customers()

    def load_and_combine_data(self):   #function to combine data from customers and members sheet to create a final data.
        customers_df = pd.read_excel(self.file_path, sheet_name='Customers',
                                     usecols=['Customer Id', 'Total Purchase', 'Total Spent', 'Customer Since'])
        members_df = pd.read_excel(self.file_path, sheet_name='Members',
                                   usecols=['Customer Id', 'Total Purchase', 'Total Spent', 'Member Since'])

        combined_df = pd.concat([customers_df, members_df])

        # Handling missing values for 'Customer Since' and 'Member Since' columns
        combined_df['Customer Since'] = pd.to_numeric(combined_df['Customer Since'], errors='coerce')
        combined_df['Member Since'] = pd.to_numeric(combined_df['Member Since'], errors='coerce')
```



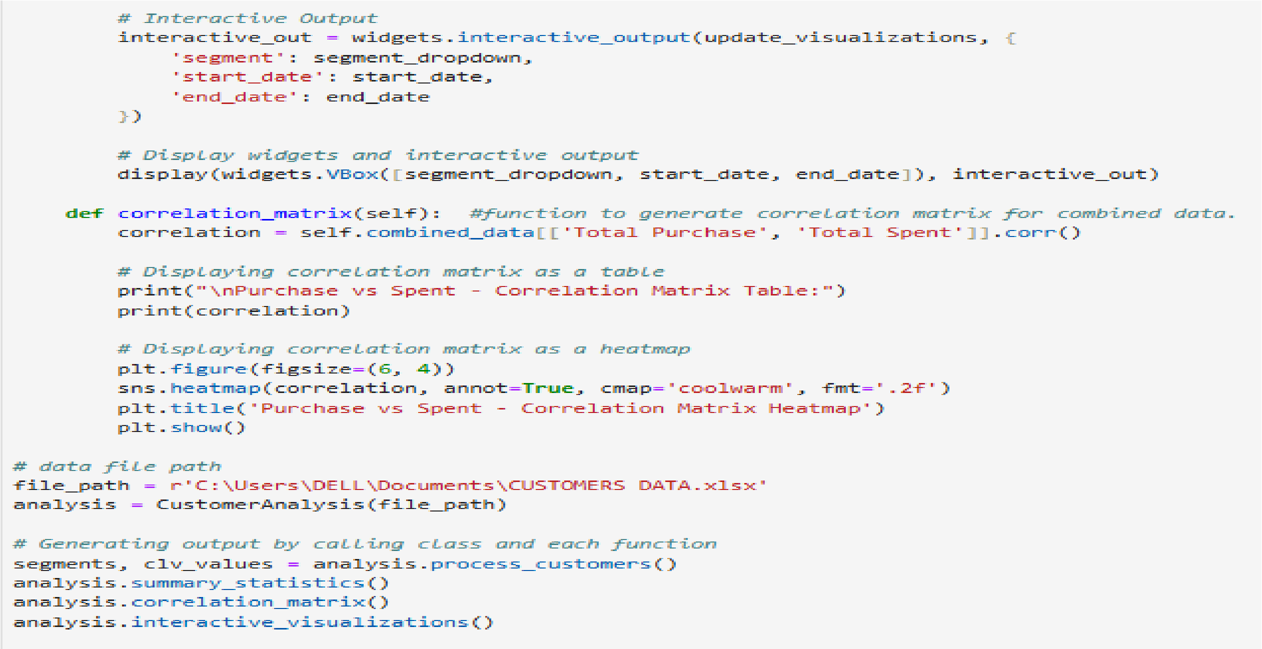
```python
        # Determining the earliest year for each customer
        combined_df['Earliest Since'] = combined_df[['Customer Since', 'Member Since']].min(axis=1)

        # Calculating the number of active years (current year - earliest year)
        current_year = datetime.now().year
        combined_df['Years Active'] = current_year - combined_df['Earliest Since']

        # Aggregation by customer ID
        aggregated_df = combined_df.groupby('Customer Id').agg({
            'Total Purchase': 'sum',
            'Total Spent': 'sum',
            'Years Active': 'max'  # Use the max because years_active should be consistent for each customer
        }).reset_index()

        return aggregated_df

    def customer_generator(self):      #Generator function to yield customer records one at a time
        for _, row in self.combined_data.iterrows():
            yield row

    @timing_decorator
    def process_single_customer(self, record):
        customer = Customer(record['Customer Id'], record['Total Purchase'], record['Total Spent'], record['Years Active'])
        clv = customer.calculate_clv()
        segment = customer.segment_customer()
        print(f"Customer ID: {customer.customer_id}, CLV: {clv}, Customer Segment: {segment}, Years Active: {customer.years_active}")
        return segment, clv
```



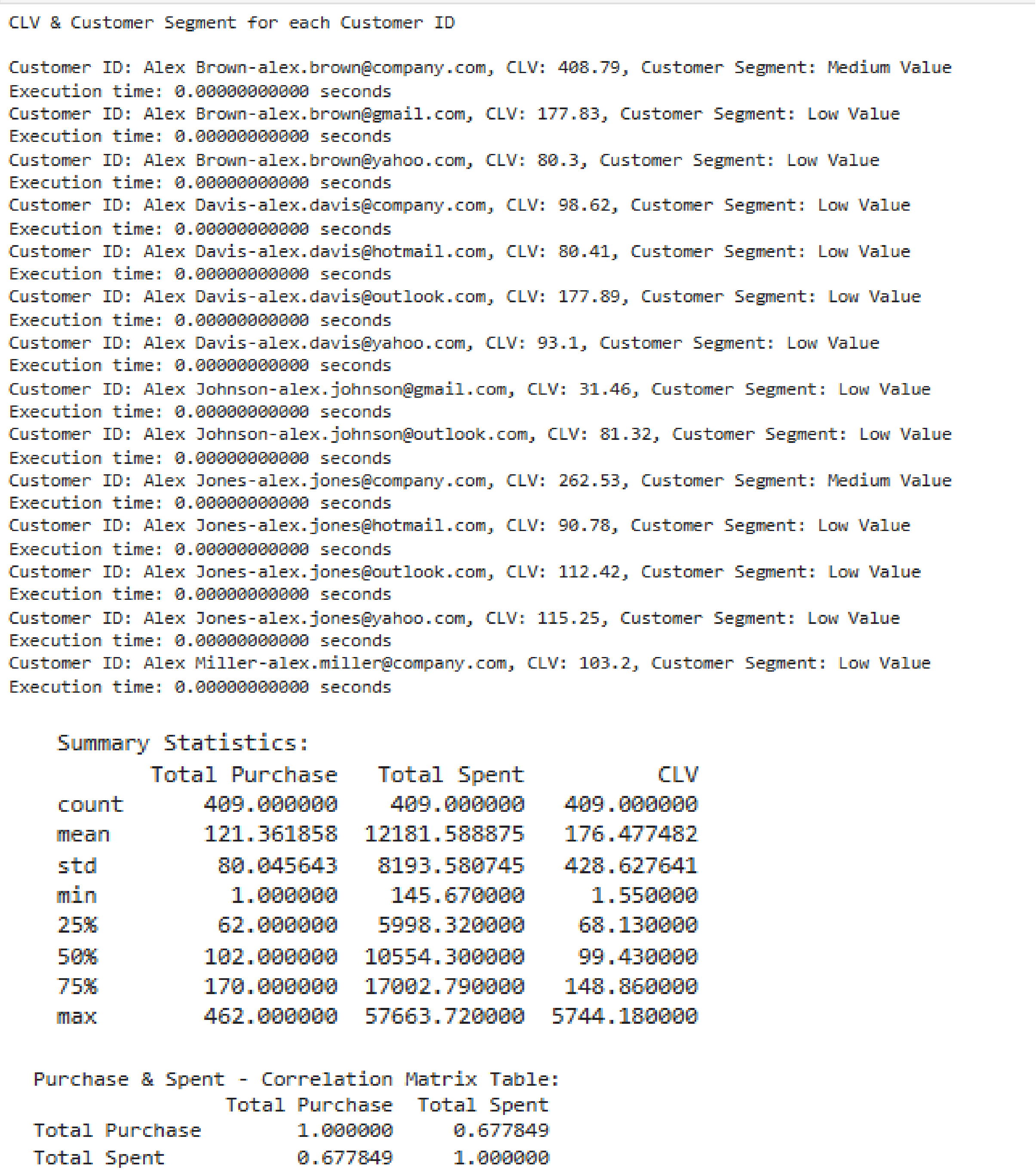
```
def process_customers(self):
    segments = []
    clv_values = []
    for record in self.customer_generator():
        segment, clv = self.process_single_customer(record)
        segments.append(segment)
        clv_values.append(clv)
    return segments, clv_values

def summary_statistics(self):   # function to generate summary statistics
    print("\nSummary Statistics:")
    self.combined_data['CLV'] = self.clv_values
    print(self.combined_data.describe())

def interactive_visualizations(self):    # function to create interactive visualizations
    # Create a dataframe combining data with segments
    segment_df = pd.DataFrame({'Customer Segment': self.segments, 'CLV': self.clv_values})
    combined_with_segments = pd.concat([self.combined_data, segment_df], axis=1)

    segment_dropdown = widgets.Dropdown(
        options=['All', 'High Value', 'Medium Value', 'Low Value'],
        value='All',
        description='Segment:'
    )

    start_date = widgets.DatePicker(
        description='Start Date',
        disabled=False
    )
    end_date = widgets.DatePicker(
        description='End Date',
        disabled=False
    )
```



```
# Function to update the visualizations based on selected filters
def update_visualizations(segment, start_date, end_date):
    filtered_data = combined_with_segments.copy()

    # Filter by segment
    if segment != 'All':
        filtered_data = filtered_data[filtered_data['Customer Segment'] == segment]

    # Interactive Donut Chart for Customer Count by Segment
    print("\nVisualizations")
    segment_counts = filtered_data['Customer Segment'].value_counts()
    fig = px.pie(values=segment_counts, names=segment_counts.index, title="Total Customers by Customer Segment",
                 hole=0.4, labels={'Customer Segment': 'Segment'},
                 color_discrete_sequence=px.colors.qualitative.Set3)
    fig.show()

    # Donut chart for Total Spent by Segment
    fig = px.pie(filtered_data, names='Customer Segment', values='Total Spent', title='Total Spent by Customer Segment', hole=0.4)
    fig.show()

    # Donut chart for Total Purchases by Segment
    fig = px.pie(filtered_data, names='Customer Segment', values='Total Purchase', title='Total Purchases by Customer Segment', hole=0.4)
    fig.show()

    # Scatter plot for Total Purchase vs Total Spent
    plt.figure(figsize=(8, 6))
    sns.scatterplot(x='Total Purchase', y='Total Spent', data=filtered_data)
    plt.title('Total Purchase vs Total Spent')
    plt.xlabel('Total Purchase')
    plt.ylabel('Total Spent')
    plt.show()
```



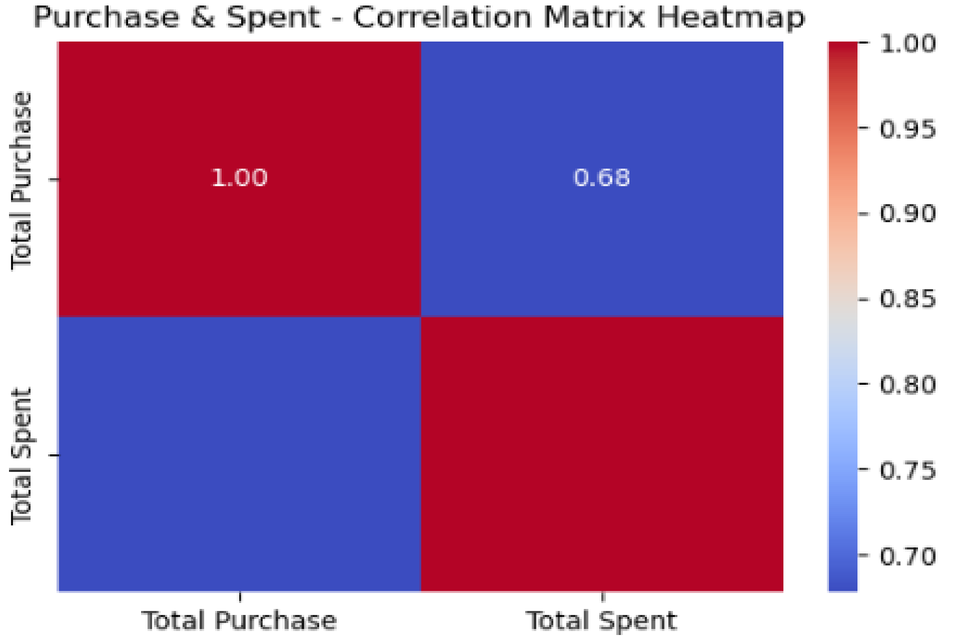
```
        # Interactive Output
        interactive_out = widgets.interactive_output(update_visualizations, {
            'segment': segment_dropdown,
            'start_date': start_date,
            'end_date': end_date
        })

        # Display widgets and interactive output
        display(widgets.VBox([segment_dropdown, start_date, end_date]), interactive_out)

    def correlation_matrix(self):  #function to generate correlation matrix for combined data.
        correlation = self.combined_data[['Total Purchase', 'Total Spent']].corr()

        # Displaying correlation matrix as a table
        print("\nPurchase vs Spent - Correlation Matrix Table:")
        print(correlation)

        # Displaying correlation matrix as a heatmap
        plt.figure(figsize=(6, 4))
        sns.heatmap(correlation, annot=True, cmap='coolwarm', fmt='.2f')
        plt.title('Purchase vs Spent - Correlation Matrix Heatmap')
        plt.show()

# data file path
file_path = r'C:\Users\DELL\Documents\CUSTOMERS DATA.xlsx'
analysis = CustomerAnalysis(file_path)

# Generating output by calling class and each function
segments, clv_values = analysis.process_customers()
analysis.summary_statistics()
analysis.correlation_matrix()
analysis.interactive_visualizations()
```



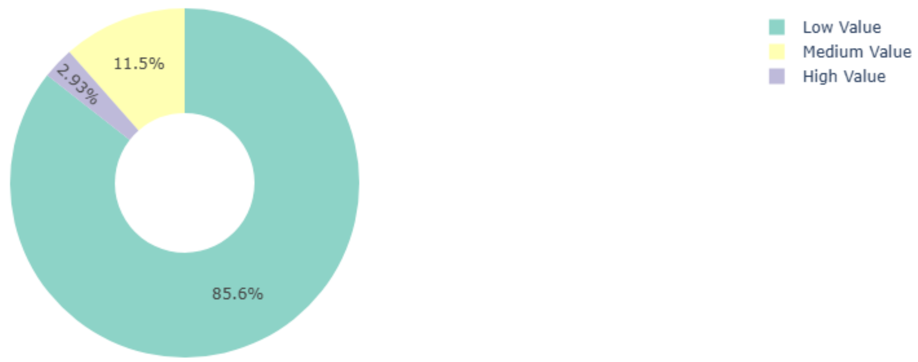
```
CLV & Customer Segment for each Customer ID

Customer ID: Alex Brown-alex.brown@company.com, CLV: 408.79, Customer Segment: Medium Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Brown-alex.brown@gmail.com, CLV: 177.83, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Brown-alex.brown@yahoo.com, CLV: 80.3, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Davis-alex.davis@company.com, CLV: 98.62, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Davis-alex.davis@hotmail.com, CLV: 80.41, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Davis-alex.davis@outlook.com, CLV: 177.89, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Davis-alex.davis@yahoo.com, CLV: 93.1, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Johnson-alex.johnson@gmail.com, CLV: 31.46, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Johnson-alex.johnson@outlook.com, CLV: 81.32, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Jones-alex.jones@company.com, CLV: 262.53, Customer Segment: Medium Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Jones-alex.jones@hotmail.com, CLV: 90.78, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Jones-alex.jones@outlook.com, CLV: 112.42, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Jones-alex.jones@yahoo.com, CLV: 115.25, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
Customer ID: Alex Miller-alex.miller@company.com, CLV: 103.2, Customer Segment: Low Value
Execution time: 0.00000000000 seconds
```

```
Summary Statistics:
       Total Purchase   Total Spent          CLV
count      409.000000    409.000000   409.000000
mean       121.361858  12181.588875   176.477482
std         80.045643   8193.580745   428.627641
min          1.000000    145.670000     1.550000
25%         62.000000   5998.320000    68.130000
50%        102.000000  10554.300000    99.430000
75%        170.000000  17002.790000   148.860000
max        462.000000  57663.720000  5744.180000
```

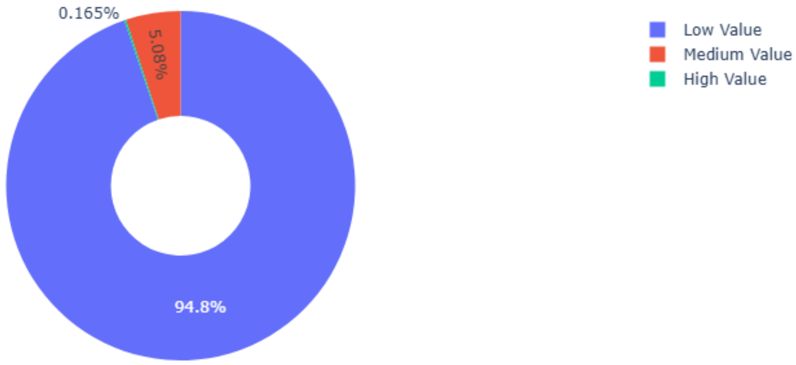
```
Purchase & Spent - Correlation Matrix Table:
                Total Purchase  Total Spent
Total Purchase        1.000000     0.677849
Total Spent           0.677849     1.000000
```



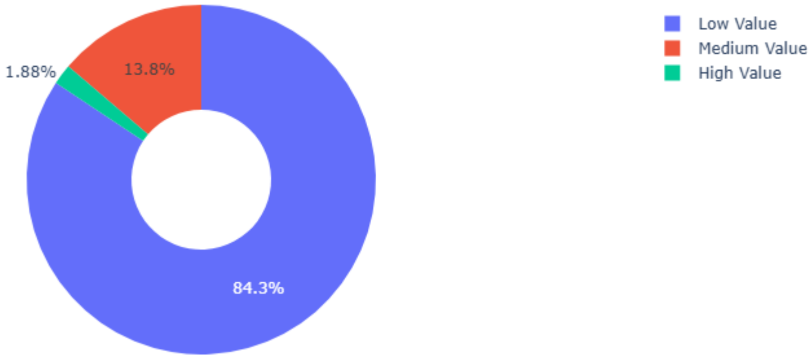
Total Customers by Customer Segment



Total Purchase by Customer Segment



Total Spent by Customer Segment



Insights and Recommendations:

Insights -

- **Customer-to-Member Conversion Rate**:
  Out of the total 409 customers, 325 are also members, resulting in a strong customer-to-member conversion rate of **79.5%**. This indicates that the majority of customers recognize the value of becoming members, suggesting the success of loyalty or membership programs.
- **High-Value Customers**:
  Only **15%** of customers are classified as high-value based on their Customer Lifetime Value (CLV). These high-value customers are responsible for purchasing **7%** of the total products sold, but they generate **18%** of the total revenue. Although they represent a smaller portion of the customer base, their contribution to revenue is significant.
- **Medium-Value Customers**:
  Medium-value customers account for **34%** of the total customer base, purchasing **36%** of the products sold and generating **43%** of the total revenue. This segment represents a substantial portion of both sales and revenue, making them critical to the business's stability.
- **Low-Value Customers**:
  The majority of customers (**51%**) fall into the low-value segment. They purchased **57%** of the total products sold but generated only **39%** of the total revenue. This indicates that while low-value customers drive product volume, their spending per transaction is relatively low.
- **Top 5 High Spenders**:
  The top 5 highest spenders account for **42%** of the total revenue (**4.98M**), despite only purchasing **3.7%** of the total products sold (**49.64K**). This underscores the importance of these high-value customers and suggests opportunities for premium offerings or tailored loyalty programs to further increase their engagement.
- **Bottom 10 Spenders**:
  The bottom 10 spenders contributed only **0.9%** of the total revenue, purchasing **1.3%** of the total products sold. This group represents a small, less-engaged segment that could be targeted with retention strategies or low-cost engagement campaigns.
- **Average Purchase Value**:
  The overall average purchase value is **$100.37**. High-value customers have an average purchase value of **$245.16**, medium-value customers average **$122**, and low-value customers average only **$68.07**. This shows a clear disparity in spending patterns across different customer segments, with high-value customers contributing significantly more per transaction.
- **Decline in Customer Acquisition**:
  Since 2012, there has been a consistent decline in the acquisition of new customers and members, with the exception of 2015, which saw a slight increase. This trend may indicate that current marketing and acquisition strategies are becoming less effective over time.

Recommendations -

- **Enhance Loyalty Programs for High-Value Customers**:
  Given that 15% of high-value customers contribute disproportionately to revenue, consider developing premium loyalty programs or exclusive offers tailored to this group. This could include early access to new products, personalized discounts, or tiered membership benefits to deepen their engagement.
- **Upsell and Cross-Sell to Medium-Value Customers**:
  Medium-value customers already contribute a large share of revenue. By identifying opportunities for upselling or cross-selling, such as personalized product recommendations or bundle offers, their average transaction value can be increased. This group has the potential to transition into high-value customers.
- **Re-engage Low-Value Customers**:
  With over half the customer base classified as low-value, focus on re-engagement strategies for this segment. Targeted marketing campaigns, such as email or SMS promotions offering discounts on products they've shown interest in, could encourage more frequent purchases and improve their lifetime value.
- **Customer Retention Focus**:
  The decline in customer acquisition since 2012, with a small uptick in 2015, suggests that customer retention efforts should be prioritized over new acquisitions. Focus on retaining existing customers through personalized marketing, loyalty rewards, and re-engagement campaigns to ensure long-term business sustainability.
- **New Customer Acquisition Strategy**:
  Address the decline in new customer acquisition by revisiting and refreshing marketing strategies. A renewed focus on digital marketing, social media outreach, and referral programs can help reverse this trend. Additionally, focusing on 2015's successful tactics may provide insights into strategies that could work in the current market.

Conclusion:

This project successfully implemented a comprehensive customer segmentation and Customer Lifetime Value (CLV) analysis pipeline. By automating data cleaning in Excel, ensuring real-time updates in SQL Server, and providing dynamic visualizations in Power BI, it streamlined the process of understanding customer behavior and making data-driven decisions. The segmentation of customers into high, medium, and low-value groups based on CLV allowed for a more focused approach to customer relationship management and strategic decision-making.

The detailed analysis revealed that while high-value customers make up a small portion of the customer base, they generate a disproportionate share of revenue, underscoring their critical role in business profitability. Medium-value customers, representing a significant percentage of sales and revenue, present promising opportunities for targeted marketing and loyalty initiatives. Low-value customers, though driving product volume, contribute less to overall revenue, highlighting the need for strategic re-engagement efforts.

In addition, the analysis identified areas for improvement, such as the steady decline in new customer acquisition since 2012, emphasizing the importance of refreshing marketing strategies to attract and retain new customers. The customer-to-member conversion rate of 79.5% shows the effectiveness of loyalty programs, though there is potential for further enhancing member engagement.

The integration of Python added advanced visualization and segmentation capabilities, facilitating deeper insights into customer trends. The combined efforts across Excel, SQL Server, Power BI, and Python offer a scalable, efficient framework for businesses to analyze their customer base, improve customer engagement, boost retention, and maximize CLV.

In conclusion, this project offers actionable insights into customer behavior and provides a roadmap for improving long-term business growth and profitability through personalized strategies, retention efforts, and data-driven marketing initiatives.

.