Import Library

```
!pip install gdown
!pip install matplotlib
!pip install wget
import gdown
import pandas as pd
import numpy as np
import os
import math
import scipy.stats as stat
import datetime as dt
import seaborn as sb
import re
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: gdown in /usr/local/lib/python3.8/dist-packages (4.4.0)
Requirement already satisfied: requests[socks] in /usr/local/lib/python3.8/dist-packages (from gdown) (2.25.1)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.8/dist-packages (from gdown) (4.6.3)
Requirement already satisfied: six in /usr/local/lib/python3.8/dist-packages (from gdown) (1.15.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.8/dist-packages (from gdown) (4.64.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.8/dist-packages (from gdown) (3.9.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.8/dist-packages (from requests[socks]->gdown) (1.24
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-packages (from requests[socks]->gdown) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/dist-packages (from requests[socks]->gdown) (2022.12
Requirement already satisfied: chardet<5,>=3.0.2 in /usr/local/lib/python3.8/dist-packages (from requests[socks]->gdown) (4.0.0)
Requirement already satisfied: PySocks!=1.5.7,>=1.5.6 in /usr/local/lib/python3.8/dist-packages (from requests[socks]->gdown) (1.7
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: matplotlib in /usr/local/lib/python3.8/dist-packages (3.2.2)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.8/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.8/dist-packages (from matplotlib) (1.4.4)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.8/dist-packages (from matplotlib) (0.11.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.8/dist-packages (from matplotlib)
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.8/dist-packages (from matplotlib) (1.21.6)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.8/dist-packages (from python-dateutil>=2.1->matplotlib) (1.15.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: wget in /usr/local/lib/python3.8/dist-packages (3.2)
```

Import Data

```
url1 = "https://raw.githubusercontent.com/hendraswntr/MiniProject-DataAnalyst-RFM_Analysis/main/data/Products.csv"
url2 = "https://raw.githubusercontent.com/hendraswntr/MiniProject-DataAnalyst-RFM_Analysis/main/data/OrderDetails.csv"
url3 = "https://raw.githubusercontent.com/hendraswntr/MiniProject-DataAnalyst-RFM_Analysis/main/data/Orders.csv"
url4 = "https://raw.githubusercontent.com/hendraswntr/MiniProject-DataAnalyst-RFM_Analysis/main/data/Customers.csv"

products = pd.read_csv(url1, sep=';')
orderdetails = pd.read_csv(url2, sep=';')
orders = pd.read_csv(url3, sep=';')
customers = pd.read_csv(url4, sep=';')

```

```
products.head()
```

	ProductID	ProductName	SupplierID	CategoryID	QuantityPerUnit	UnitPrice	Uni
0	1	Chai	1	1	10 boxes x 20 bags	18	
1	2	Chang	1	1	24 - 12 oz bottles	19	
2	3	Aniseed Syrup	1	2	12 - 550 ml bottles	10	
3	4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22	
4	5	Chef Anton's Gumbo Mix	2	2	36 boxes	21,35	

```
orderdetails.head()
```

	OrderID	ProductID	UnitPrice	Quantity	Discount
0	10248	11	14	12	0
1	10248	42	9,8	10	0
2	10248	72	34,8	5	0
3	10249	14	18,6	9	0
4	10249	51	42,4	40	0

```
orders.head()
```

	OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate	ShippedDate	ShipVia
0	10248	VINET	5	1996-07-04 00:00:00	1996-08-01 00:00:00	1996-07-16 00:00:00	3
1	10249	TOMSP	6	1996-07-05 00:00:00	1996-08-16 00:00:00	1996-07-10 00:00:00	1
2	10250	HANAR	4	1996-07-08 00:00:00	1996-08-05 00:00:00	1996-07-12 00:00:00	2
3	10251	VICTE	3	1996-07-08 00:00:00	1996-08-05 00:00:00	1996-07-15 00:00:00	1
4	10252	SUPRD	4	1996-07-09 00:00:00	1996-08-06 00:00:00	1996-07-11 00:00:00	2

```
customers.head()
```

	CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Regio
0	ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	Na
1	ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner	Avda. de la Constitucion 2222	Moxico D.F.	Na
2	ANTON	Antonio Moreno Taqueroa	Antonio Moreno	Owner	Mataderos 2312	Moxico D.F.	Na
3	AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London	Na
4	BERGS	Berglunds snabbkop	Christina Berglund	Order Administrator	Berguvsvogen 8	Luleo	Na

Data Merge

```
data1 = pd.merge(customers, orders, on=['CustomerID'])
data2 = pd.merge(data1, orderdetails, on=['OrderID'])
data = pd.merge(data2, products, on=['ProductID'])
```

```
data.head()
```

	CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Reg
0	ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	I
1	ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	I
2	BERGS	Berglunds snabbkop	Christina Berglund	Order Administrator	Berguvsvogen 8	Luleo	I
3	BERGS	Berglunds snabbkop	Christina Berglund	Order Administrator	Berguvsvogen 8	Luleo	I

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2155 entries, 0 to 2154
Data columns (total 37 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   CustomerID       2155 non-null   object
 1   CompanyName      2155 non-null   object
 2   ContactName      2155 non-null   object
 3   ContactTitle     2155 non-null   object
 4   Address          2155 non-null   object
 5   City             2155 non-null   object
 6   Region           826 non-null    object
 7   PostalCode       2100 non-null   object
 8   Country          2155 non-null   object
 9   Phone            2155 non-null   object
 10  Fax              1506 non-null   object
 11  OrderID          2155 non-null   int64
 12  EmployeeID       2155 non-null   int64
 13  OrderDate        2155 non-null   object
 14  RequiredDate     2155 non-null   object
 15  ShippedDate      2082 non-null   object
 16  ShipVia          2155 non-null   int64
 17  Freight          2155 non-null   object
 18  ShipName         2155 non-null   object
 19  ShipAddress      2155 non-null   object
 20  ShipCity         2155 non-null   object
 21  ShipRegion       856 non-null    object
 22  ShipPostalCode   2100 non-null   object
 23  ShipCountry      2155 non-null   object
 24  ProductID        2155 non-null   int64
 25  UnitPrice_x      2155 non-null   object
 26  Quantity         2155 non-null   int64
 27  Discount         2155 non-null   object
 28  ProductName      2155 non-null   object
 29  SupplierID       2155 non-null   int64
 30  CategoryID       2155 non-null   int64
 31  QuantityPerUnit  2155 non-null   object
 32  UnitPrice_y      2155 non-null   object
 33  UnitsInStock     2155 non-null   int64
 34  UnitsOnOrder     2155 non-null   int64
 35  ReorderLevel     2155 non-null   int64
 36  Discontinued     2155 non-null   bool
dtypes: bool(1), int64(10), object(26)
memory usage: 625.0+ KB
```

Data Cleansing & Processing

```
data.isnull().sum()
```

```
CustomerID            0
CompanyName           0
ContactName           0
ContactTitle          0
Address               0
City                  0
Region             1329
PostalCode           55
Country               0
Phone                 0
Fax                 649
OrderID               0
EmployeeID            0
OrderDate             0
RequiredDate          0
ShippedDate          73
ShipVia               0
Freight               0
ShipName              0
ShipAddress           0
```

```
ShipCity              0
ShipRegion         1299
ShipPostalCode       55
ShipCountry           0
ProductID             0
UnitPrice_x           0
Quantity              0
Discount              0
ProductName           0
SupplierID            0
CategoryID            0
QuantityPerUnit       0
UnitPrice_y           0
UnitsInStock          0
UnitsOnOrder          0
ReorderLevel          0
Discontinued          0
dtype: int64
```

```
del data['Region']
del data['PostalCode']
del data['Fax']
del data['ShippedDate']
del data['ShipRegion']
del data['ShipPostalCode']
```

```
data.duplicated().sum()
```

```
0
```

```
data.nunique()
```

```
CustomerID          89
CompanyName         89
ContactName         89
ContactTitle        12
Address             89
City                69
Country             21
Phone               89
OrderID            830
EmployeeID           9
OrderDate          480
RequiredDate       454
ShipVia              3
Freight            799
ShipName            90
ShipAddress         89
ShipCity            70
ShipCountry         21
ProductID           77
UnitPrice_x        116
Quantity            55
Discount            11
ProductName         77
SupplierID          29
CategoryID           8
QuantityPerUnit     70
UnitPrice_y         62
UnitsInStock        51
UnitsOnOrder        10
ReorderLevel         7
Discontinued         2
dtype: int64
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2155 entries, 0 to 2154
Data columns (total 31 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   CustomerID       2155 non-null   object
 1   CompanyName      2155 non-null   object
 2   ContactName      2155 non-null   object
 3   ContactTitle     2155 non-null   object
 4   Address          2155 non-null   object
 5   City             2155 non-null   object
 6   Country          2155 non-null   object
 7   Phone            2155 non-null   object
 8   OrderID          2155 non-null   int64
 9   EmployeeID       2155 non-null   int64
 10  OrderDate        2155 non-null   object
 11  RequiredDate     2155 non-null   object
 12  ShipVia          2155 non-null   int64
```

```
 13  Freight          2155 non-null   object
 14  ShipName         2155 non-null   object
 15  ShipAddress      2155 non-null   object
 16  ShipCity         2155 non-null   object
 17  ShipCountry      2155 non-null   object
 18  ProductID        2155 non-null   int64
 19  UnitPrice_x      2155 non-null   object
 20  Quantity         2155 non-null   int64
 21  Discount         2155 non-null   object
 22  ProductName      2155 non-null   object
 23  SupplierID       2155 non-null   int64
 24  CategoryID       2155 non-null   int64
 25  QuantityPerUnit  2155 non-null   object
 26  UnitPrice_y      2155 non-null   object
 27  UnitsInStock     2155 non-null   int64
 28  UnitsOnOrder     2155 non-null   int64
 29  ReorderLevel     2155 non-null   int64
 30  Discontinued     2155 non-null   bool
dtypes: bool(1), int64(10), object(20)
memory usage: 524.0+ KB
```

```
data['UnitPrice_x'] = pd.to_numeric(data['UnitPrice_x'].str.replace(',','.'), errors='coerce', downcast='float')
data['UnitPrice_y'] = pd.to_numeric(data['UnitPrice_y'].str.replace(',','.'), errors='coerce', downcast='float')
data['Discount'] = pd.to_numeric(data['Discount'].str.replace(',','.'), errors='coerce', downcast='float')
```

Data Analysis

```
data['OrderDate'].head()
```

```
0    1997-08-25 00:00:00
1    1998-03-16 00:00:00
2    1997-11-07 00:00:00
3    1998-03-04 00:00:00
4    1997-04-17 00:00:00
Name: OrderDate, dtype: object
```

```
data['recentpurchasedate'] = pd.DatetimeIndex(data['OrderDate']).date
```

```
data['recentpurchasedate'].head()
```

```
0    1997-08-25
1    1998-03-16
2    1997-11-07
3    1998-03-04
4    1997-04-17
Name: recentpurchasedate, dtype: object
```

```
recency_data = data.groupby(by='CustomerID', as_index=False)['recentpurchasedate'].max()
recency_data.columns = ['CustomerID','lastpurshacedate']
recency_data.head()
```

	CustomerID	lastpurshacedate
0	ALFKI	1998-04-09
1	ANATR	1998-03-04
2	ANTON	1998-01-28
3	AROUT	1998-04-10
4	BERGS	1998-03-04

```
data['OrderDate'].max()
```

```
'1998-05-06 00:00:00'
```

```
now = dt.date(1998,5,6)
print(now)
```

```
1998-05-06
```

```
recency_data['Recency'] = recency_data['lastpurshacedate'].apply(lambda x: (now - x).days)
recency_data.head()
```

	CustomerID	lastpurshacedate	Recency
0	ALFKI	1998-04-09	27
1	ANATR	1998-03-04	63
2	ANTON	1998-01-28	98

```
recency_data.drop('lastpurshacedate',axis=1,inplace=True)
recency_data.head()
```

	CustomerID	Recency
0	ALFKI	27
1	ANATR	63
2	ANTON	98
3	AROUT	26
4	BERGS	63

```
frequency_data = data.groupby(by=['CustomerID'], as_index=False)['OrderID'].count()
frequency_data.columns = ['CustomerID','Frequency']
frequency_data.head()
```

	CustomerID	Frequency
0	ALFKI	12
1	ANATR	10
2	ANTON	17
3	AROUT	30
4	BERGS	52

```
data['TotalSales'] = (data['UnitPrice_x'] - data['Discount'])*data['Quantity']
data['TotalSales']
```

```
0        680.249977
1         91.199997
2        911.999969
3       1365.000000
4        136.799995
            ...
2150     185.399998
2151     386.249995
2152     155.000000
2153     247.999992
2154     775.000000
Name: TotalSales, Length: 2155, dtype: float64
```

```
monetary_data = data.groupby(by='CustomerID',as_index=False).agg({'TotalSales': 'sum'})
monetary_data.columns = ['CustomerID','Monetary']
monetary_data.head()
```

	CustomerID	Monetary
0	ALFKI	4583.500010
1	ANATR	1402.949990
2	ANTON	7497.149937
3	AROUT	13788.899798
4	BERGS	26900.049850

```
temp_data = recency_data.merge(frequency_data,on='CustomerID')
rfm_data = temp_data.merge(monetary_data,on='CustomerID')
rfm_data.set_index('CustomerID',inplace=True)
rfm_data.head()
```

	Recency	Frequency	Monetary
CustomerID			

```
rfm_data.shape
```

```
(89, 3)
```

```
quantiles = rfm_data.quantile(q=[0.25,0.5,0.75])
quantiles
```

	Recency	Frequency	Monetary
0.25	8.0	11.0	3361.000004
0.50	23.0	20.0	7531.800077
0.75	58.0	31.0	18086.900101

```
quantiles.to_dict()
```

```
{'Recency': {0.25: 8.0, 0.5: 23.0, 0.75: 58.0},
 'Frequency': {0.25: 11.0, 0.5: 20.0, 0.75: 31.0},
 'Monetary': {0.25: 3361.0000038146973,
  0.5: 7531.8000774383545,
  0.75: 18086.900100708008}}
```

```
def RScore(x,p,d):
    if x <= d[p][0.25]:
        return 4
    elif x <= d[p][0.50]:
        return 3
    elif x <= d[p][0.75]:
        return 2
    else:
        return 1

def FMScore(x,p,d):
    if x <= d[p][0.25]:
        return 1
    elif x <= d[p][0.50]:
        return 2
    elif x <= d[p][0.75]:
        return 3
    else:
        return 4
```

```
rfm_segmentation = rfm_data
rfm_segmentation['R_Score'] = rfm_segmentation['Recency'].apply(RScore, args=('Recency',quantiles,))
rfm_segmentation['F_Score'] = rfm_segmentation['Frequency'].apply(FMScore, args=('Frequency',quantiles,))
rfm_segmentation['M_Score'] = rfm_segmentation['Monetary'].apply(FMScore, args=('Monetary',quantiles,))
rfm_segmentation.head()
```

	Recency	Frequency	Monetary	R_Score	F_Score	M_Score
CustomerID						
ALFKI	27	12	4583.500010	2	2	2
ANATR	63	10	1402.949990	1	1	1
ANTON	98	17	7497.149937	1	2	2
AROUT	26	30	13788.899798	2	3	3
BERGS	63	52	26900.049850	1	4	4

```
rfm_segmentation['RFMScore'] = rfm_segmentation.R_Score.map(str) \
                            + rfm_segmentation.F_Score.map(str) \
                            + rfm_segmentation.M_Score.map(str)
rfm_segmentation.head()
```

	Recency	Frequency	Monetary	R_Score	F_Score	M_Score	RFMScore
CustomerID							

```
rfm_segmentation[rfm_segmentation['RFMScore']=='444'].sort_values('Monetary', ascending=False).head(10)
```

	Recency	Frequency	Monetary	R_Score	F_Score	M_Score	RFMScore
CustomerID							
SAVEA	5	116	115253.339402	4	4	4	444
ERNSH	1	102	112903.829578	4	4	4	444
HUNGO	6	55	57109.540001	4	4	4	444
RATTC	0	71	52188.890416	4	4	4	444
QUEEN	2	40	30112.550216	4	4	4	444
WHITC	5	40	28990.799867	4	4	4	444
BONAP	0	44	23776.400169	4	4	4	444
HILAA	8	45	23577.579861	4	4	4	444
LEHMS	1	39	21209.819996	4	4	4	444

```
rfm_segmentation['Customer_Segmentation'] = np.where(rfm_segmentation['RFMScore'] ==444, "Best Customers",
                                (np.where(rfm_segmentation['F_Score'] == 4,"Loyal Customers",
                                (np.where(rfm_segmentation['M_Score'] == 4, "Big Spenders",
                                (np.where(rfm_segmentation['F_Score'] == 3, "Gold Customers",
                                (np.where(rfm_segmentation['F_Score'] == 2, "Silver Customers",
                                (np.where(rfm_segmentation['RFMScore'] == 244,"Almost Lost",
                               (np.where(rfm_segmentation['RFMScore'] == 144, "Lost Customers",
                               np.where(rfm_segmentation['RFMScore'] == 111,'Lost Cheap Customers', 'Low Value Custor
                                        ))))))))))))))
rfm_segmentation
```

	Recency	Frequency	Monetary	R_Score	F_Score	M_Score	RFMScore
CustomerID							
ALFKI	27	12	4583.500010	2	2	2	222
ANATR	63	10	1402.949990	1	1	1	111
ANTON	98	17	7497.149937	1	2	2	122
AROUT	26	30	13788.899798	2	3	3	233
BERGS	63	52	26900.049850	1	4	4	144
...	...	...	...	...	...	...	...
WARTH	21	37	16580.599975	3	4	3	343
WELLI	58	19	6465.750010	2	2	2	222
WHITC	5	40	28990.799867	4	4	4	444
WILMK	29	17	3161.349983	2	2	1	221
WOLZA	13	16	3531.949996	3	2	2	322

89 rows × 8 columns

```
rfsegment = pd.merge(rfm_segmentation, data, on=['CustomerID'])
data = rfsegment.sort_values(by=['CustomerID'])
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2155 entries, 0 to 2154
Data columns (total 41 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   CustomerID             2155 non-null   object
 1   Recency                2155 non-null   int64
 2   Frequency              2155 non-null   int64
 3   Monetary               2155 non-null   float64
 4   R_Score                2155 non-null   int64
 5   F_Score                2155 non-null   int64
 6   M_Score                2155 non-null   int64
 7   RFMScore               2155 non-null   object
 8   Customer_Segmentation  2155 non-null   object
 9   CompanyName            2155 non-null   object
 10  ContactName            2155 non-null   object
 11  ContactTitle           2155 non-null   object
 12  Address                2155 non-null   object
```

```
 13  City                  2155 non-null   object
 14  Country               2155 non-null   object
 15  Phone                 2155 non-null   object
 16  OrderID               2155 non-null   int64
 17  EmployeeID            2155 non-null   int64
 18  OrderDate             2155 non-null   object
 19  RequiredDate          2155 non-null   object
 20  ShipVia               2155 non-null   int64
 21  Freight               2155 non-null   object
 22  ShipName              2155 non-null   object
 23  ShipAddress           2155 non-null   object
 24  ShipCity              2155 non-null   object
 25  ShipCountry           2155 non-null   object
 26  ProductID             2155 non-null   int64
 27  UnitPrice_x           2155 non-null   float32
 28  Quantity              2155 non-null   int64
 29  Discount              2155 non-null   float32
 30  ProductName           2155 non-null   object
 31  SupplierID            2155 non-null   int64
 32  CategoryID            2155 non-null   int64
 33  QuantityPerUnit       2155 non-null   object
 34  UnitPrice_y           2155 non-null   float32
 35  UnitsInStock          2155 non-null   int64
 36  UnitsOnOrder          2155 non-null   int64
 37  ReorderLevel          2155 non-null   int64
 38  Discontinued          2155 non-null   bool
 39  recentpurchasedate    2155 non-null   object
 40  TotalSales            2155 non-null   float64
dtypes: bool(1), float32(3), float64(2), int64(15), object(20)
memory usage: 667.1+ KB
```

```
data.head()
```

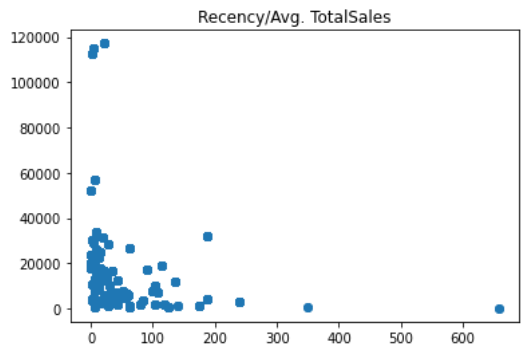
	CustomerID	Recency	Frequency	Monetary	R_Score	F_Score	M_Score	RFMScore
0	ALFKI	27	12	4583.50001	2	2	2	222
11	ALFKI	27	12	4583.50001	2	2	2	222
10	ALFKI	27	12	4583.50001	2	2	2	222
9	ALFKI	27	12	4583.50001	2	2	2	222
7	ALFKI	27	12	4583.50001	2	2	2	222

5 rows × 41 columns

```
data['Customer_Segmentation'].value_counts()
```

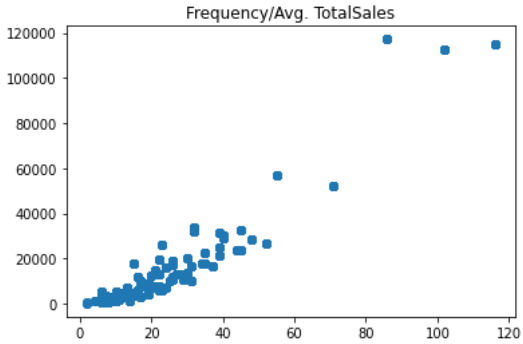
```
Loyal Customers        1066
Gold Customers          459
Silver Customers        318
Low Value Customers     211
Big Spenders            101
Name: Customer_Segmentation, dtype: int64
```

```
plt.scatter(data['Recency'], data['Monetary'])
plt.title('Recency/Avg. TotalSales')
plt.savefig("RecedncyAvgTotalSales.png")
plt.show()
```




```
plt.scatter(data['Frequency'], data['Monetary'])
```

```
plt.title('Frequency/Avg. TotalSales')
plt.savefig("FrequencyAvgTotalSales.png")
plt.show()
```




```
data.to_csv('./RFM_Analysis.csv', index=False)
```

✓ 0s completed at 15:02