PROBLEM STATEMENT : The transactions made by a UK-based, registered, non-store online retailer between December 1, 2010, and December 9,2011, are all included in the transnational data set known as online retail. The company primarily offers one-of-a-kind gifts for every occasion. The company has a large number of wholesalers as clients.Company Objective Using the global online retail dataset, we will design a clustering model and select the ideal group of clients for the business to target.

Importing Libraries

In [1]:

```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
```

Reading the data

In [3]:

```
df=pd.read_csv(r"C:\Users\G S R  KARTHIK\Documents\OnlineRetailData[1]\OnlineRetail.csv"
df
```

Out[3]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	C
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850.0	K
1	536365	71053	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850.0	K
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850.0	K
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850.0	K
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850.0	K
...	...	...	...	...	...	...	...	
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	09-12-2011 12:50	0.85	12680.0	
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	09-12-2011 12:50	2.10	12680.0	
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	09-12-2011 12:50	4.15	12680.0	
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	09-12-2011 12:50	4.15	12680.0	
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	09-12-2011 12:50	4.95	12680.0	

541909 rows × 8 columns

Data Cleaning and Preprocessing

In [4]:

```
df.head()
```

Out[4]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850.0	United Kingdom

In [5]:

```
df.tail()
```

Out[5]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	C
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	09-12-2011 12:50	0.85	12680.0	
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	09-12-2011 12:50	2.10	12680.0	
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	09-12-2011 12:50	4.15	12680.0	
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	09-12-2011 12:50	4.15	12680.0	
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	09-12-2011 12:50	4.95	12680.0	

In [6]:

```
df.describe()
```

Out[6]:

	Quantity	UnitPrice	CustomerID
count	541909.000000	541909.000000	406829.000000
mean	9.552250	4.611114	15287.690570
std	218.081158	96.759853	1713.600303
min	-80995.000000	-11062.060000	12346.000000
25%	1.000000	1.250000	13953.000000
50%	3.000000	2.080000	15152.000000
75%	10.000000	4.130000	16791.000000
max	80995.000000	38970.000000	18287.000000

In [7]:

```
df.shape
```

Out[7]:

```
(541909, 8)
```

In [8]:

```
df.columns
```

Out[8]:

```
Index(['InvoiceNo', 'StockCode', 'Description', 'Quantity', 'InvoiceDate',
       'UnitPrice', 'CustomerID', 'Country'],
      dtype='object')
```

In [9]:

```
df.isnull().sum()
```

Out[9]:

```
InvoiceNo          0
StockCode          0
Description     1454
Quantity           0
InvoiceDate        0
UnitPrice          0
CustomerID     135080
Country            0
dtype: int64
```

In [13]:

```
df.fillna(method="ffill",inplace=True)
```

In [14]:

```
df.isnull().sum()
```

Out[14]:

```
InvoiceNo      0
StockCode      0
Description    0
Quantity       0
InvoiceDate    0
UnitPrice      0
CustomerID     0
Country        0
dtype: int64
```

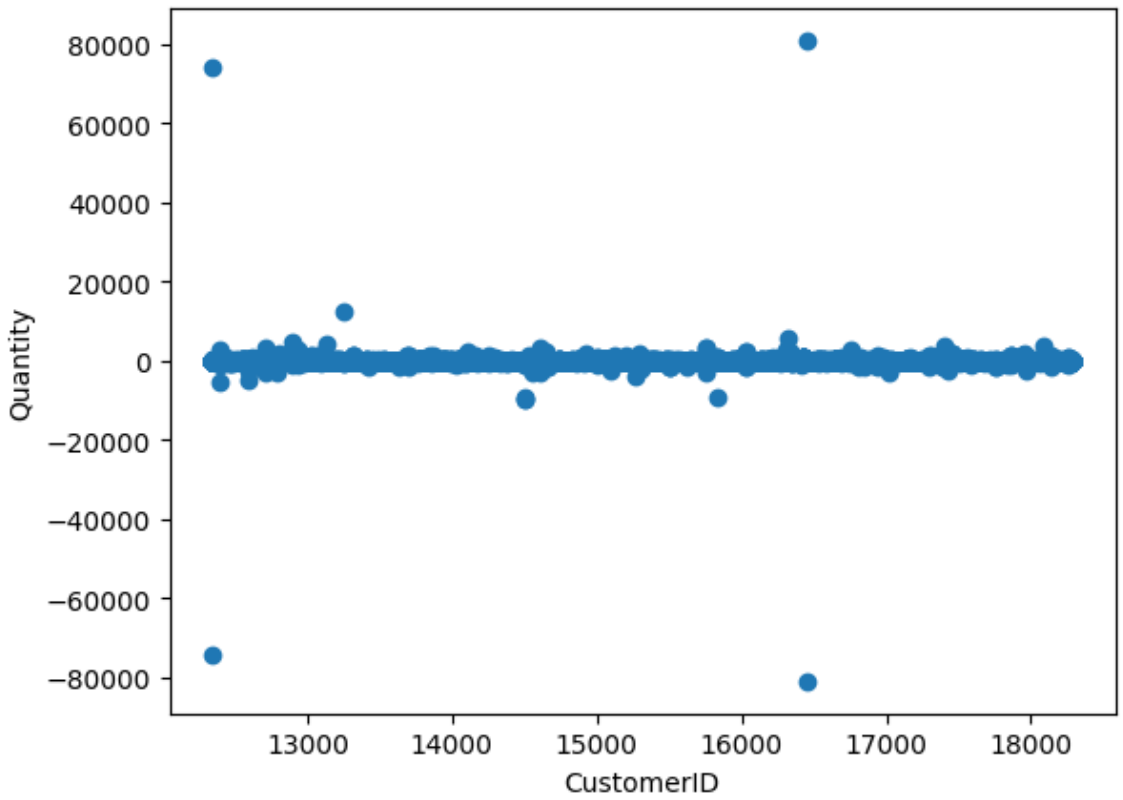
Data Visualization

In [17]:

```
plt.scatter(df["CustomerID"],df["Quantity"])
plt.xlabel("CustomerID")
plt.ylabel("Quantity")
```

Out[17]:

```
Text(0, 0.5, 'Quantity')
```



Importing KMeans

In [18]:

```
from sklearn.cluster import KMeans
km=KMeans()
km
```

Out[18]:

KMeans()

In [20]:

```
y_predicted=km.fit_predict(df[["CustomerID","Quantity"]])
y_predicted
```

```
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
```

Out[20]:

```
array([1, 1, 1, ..., 5, 5, 5])
```

In [21]:

```
df["cluster"]=y_predicted
df.head()
```

Out[21]:

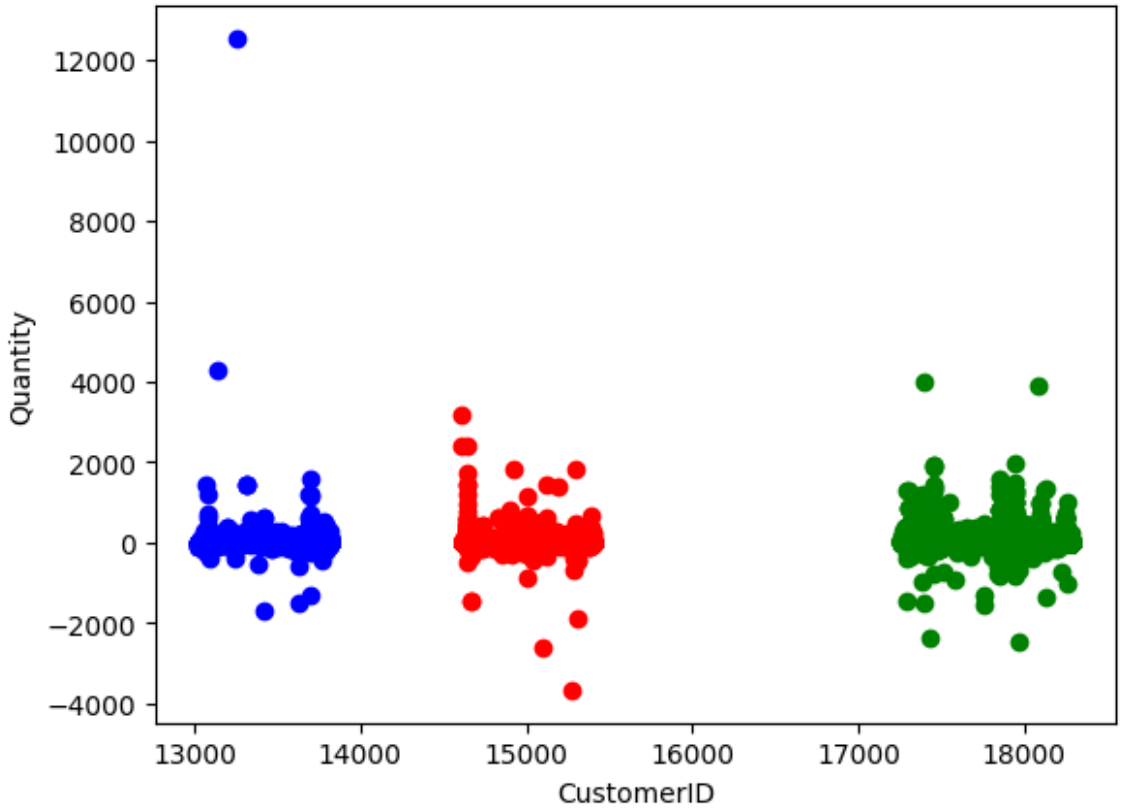
	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850.0	United Kingdom

In [22]:

```
df1=df[df.cluster==0]
df2=df[df.cluster==1]
df3=df[df.cluster==2]
plt.scatter(df1["CustomerID"],df1["Quantity"],color="red")
plt.scatter(df2["CustomerID"],df2["Quantity"],color="green")
plt.scatter(df3["CustomerID"],df3["Quantity"],color="blue")
plt.xlabel("CustomerID")
plt.ylabel("Quantity")
```

Out[22]:

```
Text(0, 0.5, 'Quantity')
```



In [23]:

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(df[["Quantity"]])
df["Quantity"]=scaler.transform(df[["Quantity"]])
df.head()
```

Out[23]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	0.500037	01-12-2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	0.500037	01-12-2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	0.500049	01-12-2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	0.500037	01-12-2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	0.500037	01-12-2010 08:26	3.39	17850.0	United Kingdom

In [24]:

```
scaler.fit(df[["CustomerID"]])
df["CustomerID"]=scaler.transform(df[["CustomerID"]])
df.head()
```

Out[24]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	0.500037	01-12-2010 08:26	2.55	0.926443	United Kingdom
1	536365	71053	WHITE METAL LANTERN	0.500037	01-12-2010 08:26	3.39	0.926443	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	0.500049	01-12-2010 08:26	2.75	0.926443	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	0.500037	01-12-2010 08:26	3.39	0.926443	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	0.500037	01-12-2010 08:26	3.39	0.926443	United Kingdom

In [25]:

```
y_predicted=km.fit_predict(df[["CustomerID","Quantity"]])
y_predicted
```

```
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
```

Out[25]:

```
array([5, 5, 5, ..., 3, 3, 3])
```

In [26]:

```
df["New Cluster"]=y_predicted
df.head()
```

Out[26]:

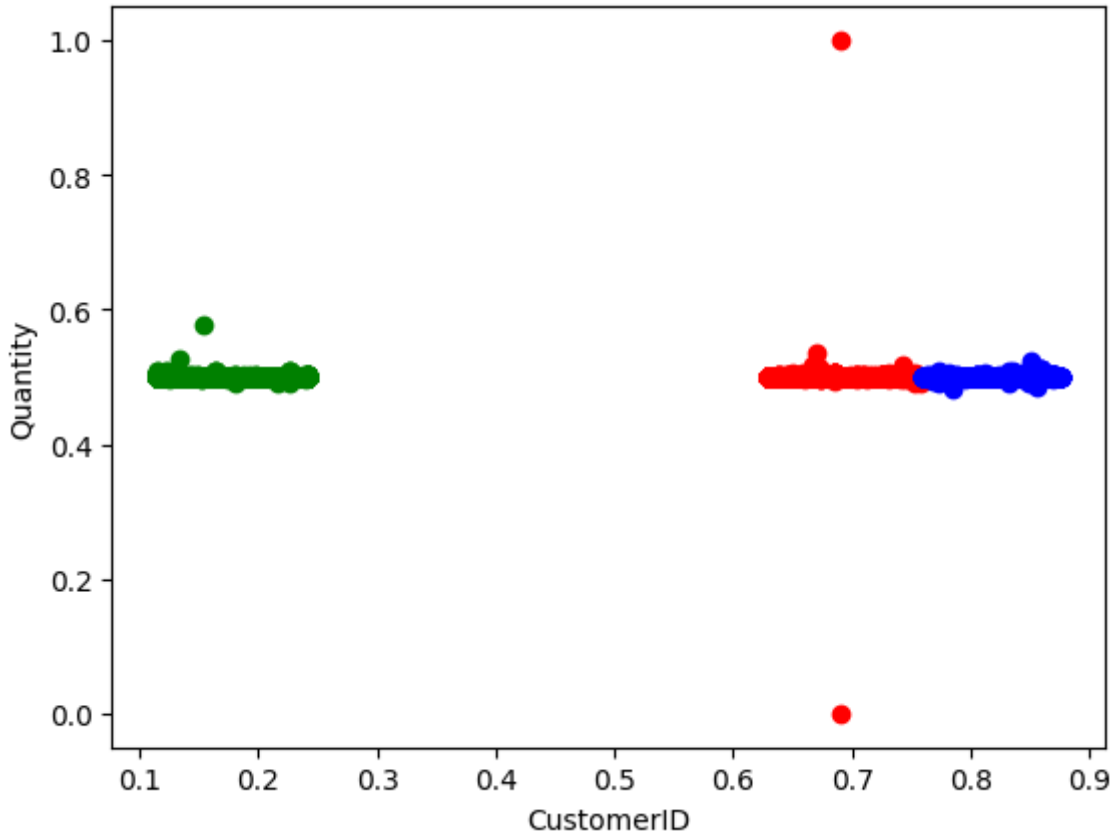
	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	0.500037	01-12-2010 08:26	2.55	0.926443	United Kingdom
1	536365	71053	WHITE METAL LANTERN	0.500037	01-12-2010 08:26	3.39	0.926443	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	0.500049	01-12-2010 08:26	2.75	0.926443	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	0.500037	01-12-2010 08:26	3.39	0.926443	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	0.500037	01-12-2010 08:26	3.39	0.926443	United Kingdom

In [27]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==1]
df3=df[df["New Cluster"]==2]
plt.scatter(df1["CustomerID"],df1["Quantity"],color="red")
plt.scatter(df2["CustomerID"],df2["Quantity"],color="green")
plt.scatter(df3["CustomerID"],df3["Quantity"],color="blue")
plt.xlabel("CustomerID")
plt.ylabel("Quantity")
```

Out[27]:

```
Text(0, 0.5, 'Quantity')
```



In [28]:

```
km.cluster_centers_
```

Out[28]:

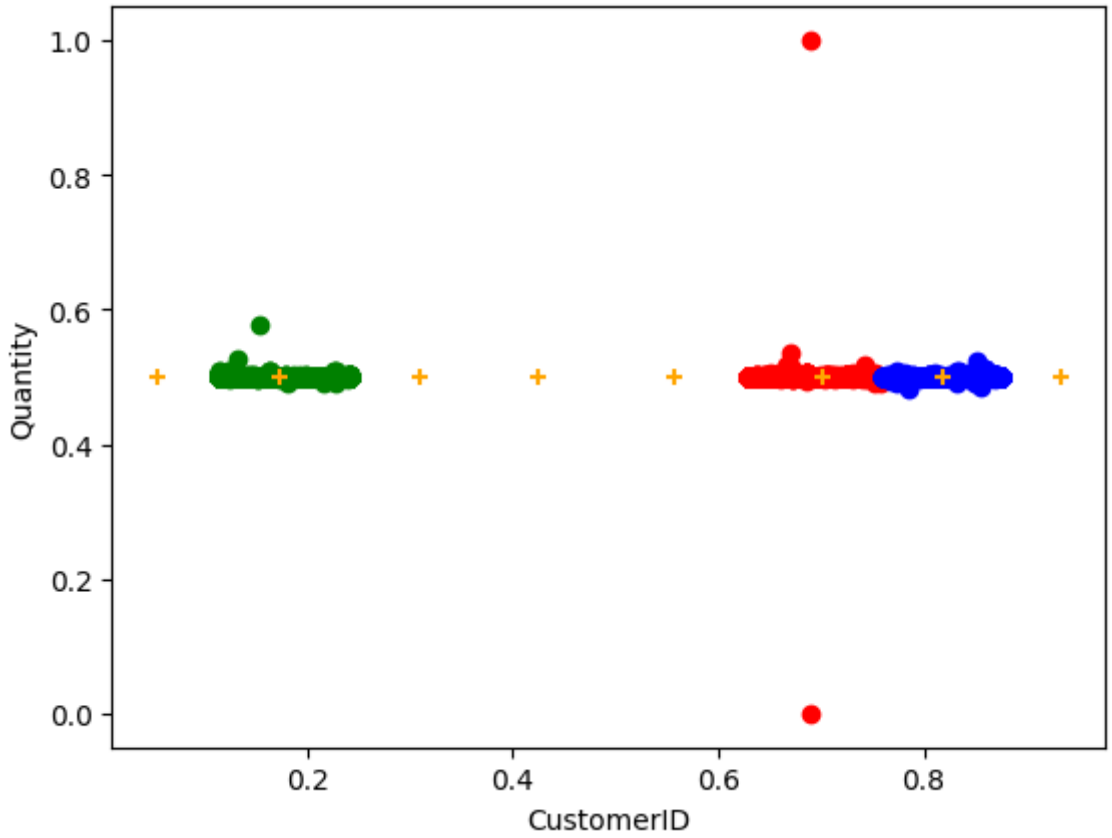
```
array([[0.70128655, 0.50005787],
       [0.17396373, 0.50005875],
       [0.81845621, 0.50006031],
       [0.05399113, 0.50006863],
       [0.42569425, 0.50006643],
       [0.93301334, 0.50005098],
       [0.31038999, 0.50005491],
       [0.55731105, 0.50005339]])
```

In [29]:

```
df1=df[df["New Cluster"]==0]
df2=df[df["New Cluster"]==1]
df3=df[df["New Cluster"]==2]
plt.scatter(df1["CustomerID"],df1["Quantity"],color="red")
plt.scatter(df2["CustomerID"],df2["Quantity"],color="green")
plt.scatter(df3["CustomerID"],df3["Quantity"],color="blue")
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color="orange",marker="+")
plt.xlabel("CustomerID")
plt.ylabel("Quantity")
```

Out[29]:

```
Text(0, 0.5, 'Quantity')
```



In [30]:

```
k_rng=range(1,10)
sse=[]
```

In [31]:

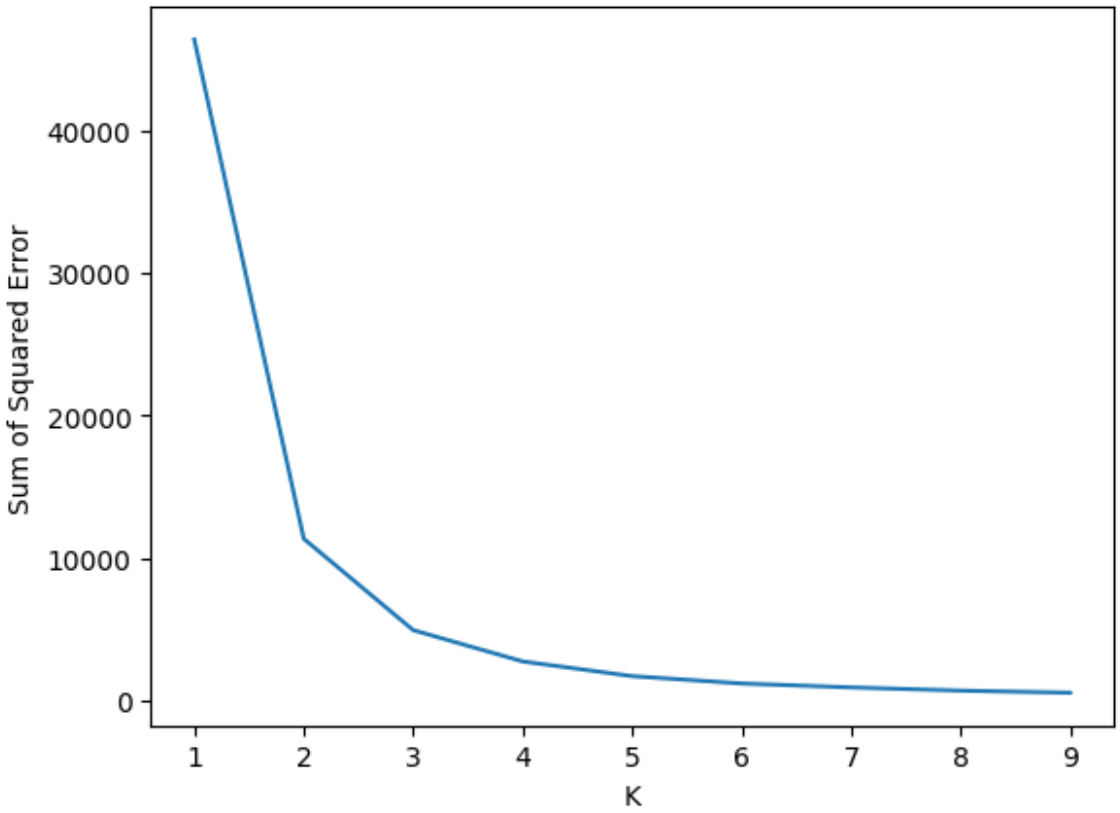
```
for k in k_rng:
    km=KMeans(n_clusters=k)
    km.fit(df[["CustomerID","Quantity"]])
    sse.append(km.inertia_)
#km.inertia_ will give you the value of sum of square error
print(sse)
plt.plot(k_rng,sse)
plt.xlabel("K")
plt.ylabel("Sum of Squared Error")
```

```
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
C:\Users\G S R  KARTHIK\AppData\Local\Programs\Python\Python310\lib\site-p
ackages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value o
f `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  warnings.warn(
```

```
[46374.84553398474, 11336.064585311016, 4921.068384045962, 2723.5191051895
285, 1695.0392229313325, 1178.5808580112473, 902.7121607671178, 677.321701
2227778, 529.4339099645493]
```

Out[31]:

```
Text(0, 0.5, 'Sum of Squared Error')
```



CONCLUSION : The KMeans model is the best for the given dataset

In [ ]: