# 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 offer 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.

# Data Collection

In [32]:

```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
```

In [33]:

```
df=pd.read_csv(r"C:\Users\chila\Downloads\Retail.csv")
df
```

Out[33]:

| | 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

In [34]:

```
df.head()
```

Out[34]:

| | 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 [35]:

```
df.tail()
```

Out[35]:

| | 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 [36]:

```
df['CustomerID'].value_counts()
```

Out[36]:

```
CustomerID
17841.0    7983
14911.0    5903
14096.0    5128
12748.0    4642
14606.0    2782
           ...
15070.0       1
15753.0       1
17065.0       1
16881.0       1
16995.0       1
Name: count, Length: 4372, dtype: int64
```

In [37]:

```
df['Quantity'].value_counts()
```

Out[37]:

```
Quantity
 1        148227
 2         81829
 12        61063
 6         40868
 4         38484
           ...
-472           1
-161           1
-1206          1
-272           1
-80995         1
Name: count, Length: 722, dtype: int64
```

In [38]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    541909 non-null  object
 1   StockCode    541909 non-null  object
 2   Description  540455 non-null  object
 3   Quantity     541909 non-null  int64
 4   InvoiceDate  541909 non-null  object
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
```

In [39]:

```
df.isnull().any()
```

Out[39]:

```
InvoiceNo      False
StockCode      False
Description     True
Quantity       False
InvoiceDate    False
UnitPrice      False
CustomerID      True
Country        False
dtype: bool
```

In [40]:

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

In [41]:

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

Out[41]:

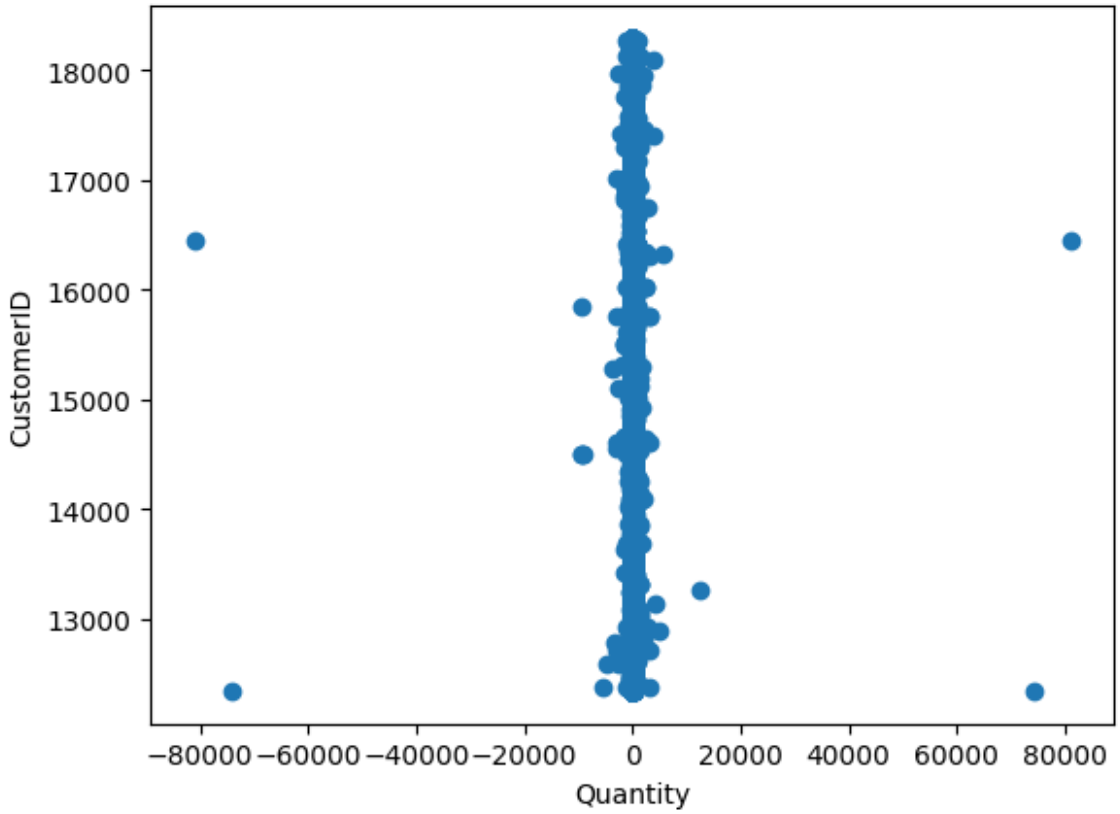
```
InvoiceNo      0
StockCode      0
Description    0
Quantity       0
InvoiceDate    0
UnitPrice      0
CustomerID     0
Country        0
dtype: int64
```

In [42]:

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

Out[42]:

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




# K-Means clustering

In [43]:

```
from sklearn.cluster import KMeans
```

In [44]:

```
km=KMeans()
km
```

Out[44]:

```
KMeans()
```

In [45]:

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

Out[45]:

```
array([6, 6, 6, ..., 0, 0, 0])
```

In [46]:

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

Out[46]:

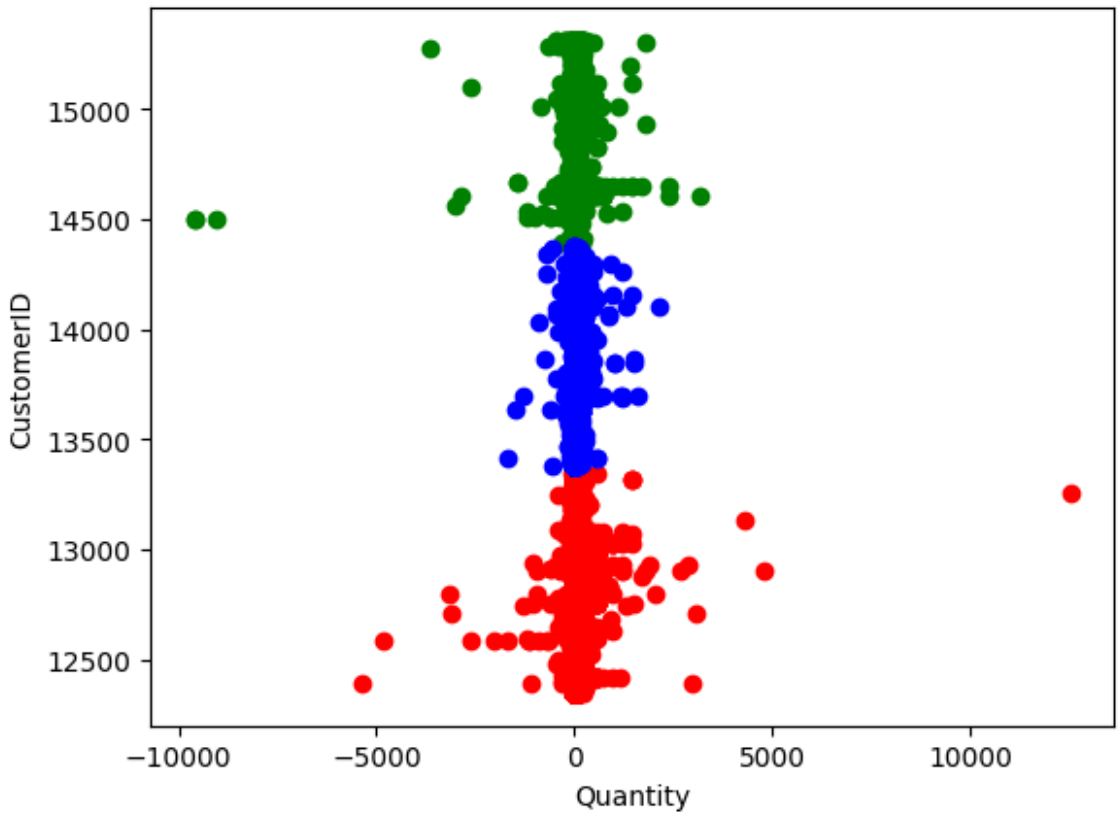
| | InvoiceNo | StockCode | Description | Quantity | InvoiceDate | UnitPrice | CustomerID | Country |
|---|---|---|---|---|---|---|---|---|
| **0** | 536365 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 01-12-2010 08:26 | 2.55 | 17850.0 | United Kingdom |
| **1** | 536365 | 71053 | WHITE METAL 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 [47]:

```
df1=df[df.Cluster==0]
df2=df[df.Cluster==2]
df3=df[df.Cluster==3]
plt.scatter(df1["Quantity"],df1["CustomerID"],color="red")
plt.scatter(df2["Quantity"],df2["CustomerID"],color="green")
plt.scatter(df3["Quantity"],df3["CustomerID"],color="blue")
plt.xlabel("Quantity")
plt.ylabel("CustomerID")
```

Out[47]:

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




In [48]:

```
from sklearn.preprocessing import MinMaxScaler
```

In [49]:

```
scaler=MinMaxScaler()
```

In [50]:

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

Out[50]:

| | 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 | 0.926443 | United Kingdom |
| **1** | 536365 | 71053 | WHITE METAL LANTERN | 6 | 01-12-2010 08:26 | 3.39 | 0.926443 | United Kingdom |
| **2** | 536365 | 84406B | CREAM CUPID HEARTS COAT HANGER | 8 | 01-12-2010 08:26 | 2.75 | 0.926443 | United Kingdom |
| **3** | 536365 | 84029G | KNITTED UNION FLAG HOT WATER BOTTLE | 6 | 01-12-2010 08:26 | 3.39 | 0.926443 | United Kingdom |
| **4** | 536365 | 84029E | RED WOOLLY HOTTIE WHITE HEART. | 6 | 01-12-2010 08:26 | 3.39 | 0.926443 | United Kingdom |

In [51]:

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

Out[51]:

| | 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 [52]:

```
km=KMeans()
```

In [53]:

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

Out[53]:

```
array([7, 7, 7, ..., 2, 2, 2])
```

In [54]:

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

Out[54]:

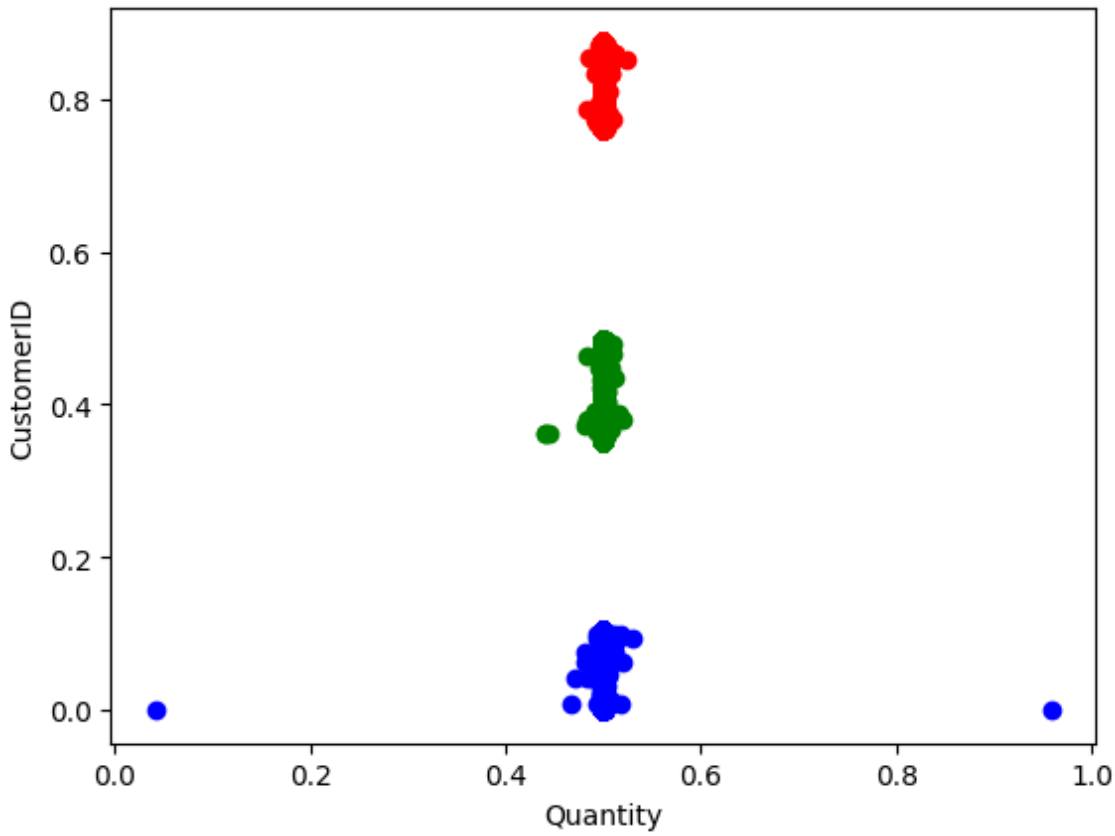
| | InvoiceNo | StockCode | Description | Quantity | InvoiceDate | UnitPrice | CustomerID | Country |
|---|---|---|---|---|---|---|---|---|
| **0** | 536365 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 0.500037 | 01-12-2010 08:26 | 2.55 | 0.926443 | United Kingdom |
| **1** | 536365 | 71053 | WHITE 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 [55]:

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

Out[55]:

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




In [56]:

```
km.cluster_centers_
```

Out[56]:

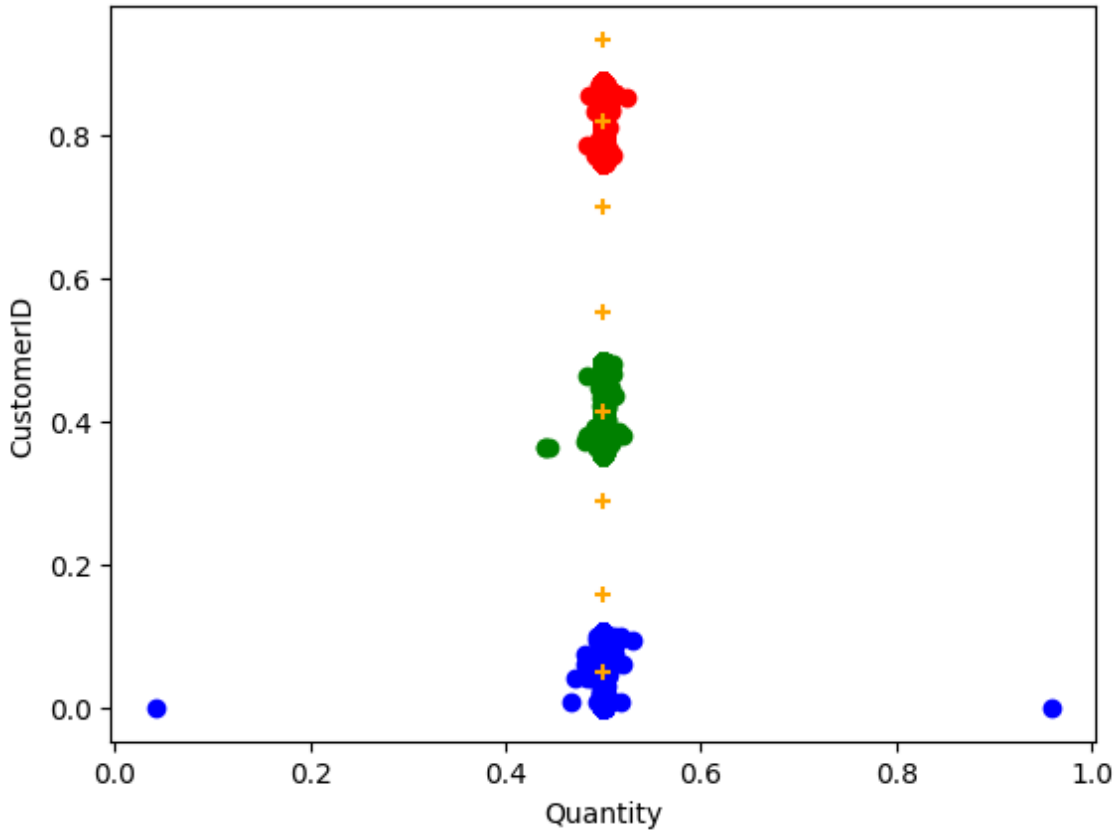
```
array([[0.5000603 , 0.81850179],
       [0.50005951, 0.41475394],
       [0.50006675, 0.05043685],
       [0.5000578 , 0.70054972],
       [0.50005702, 0.15869097],
       [0.5000539 , 0.55318473],
       [0.50006587, 0.28970201],
       [0.50005099, 0.93305234]])
```

In [57]:

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

Out[57]:

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




In [58]:

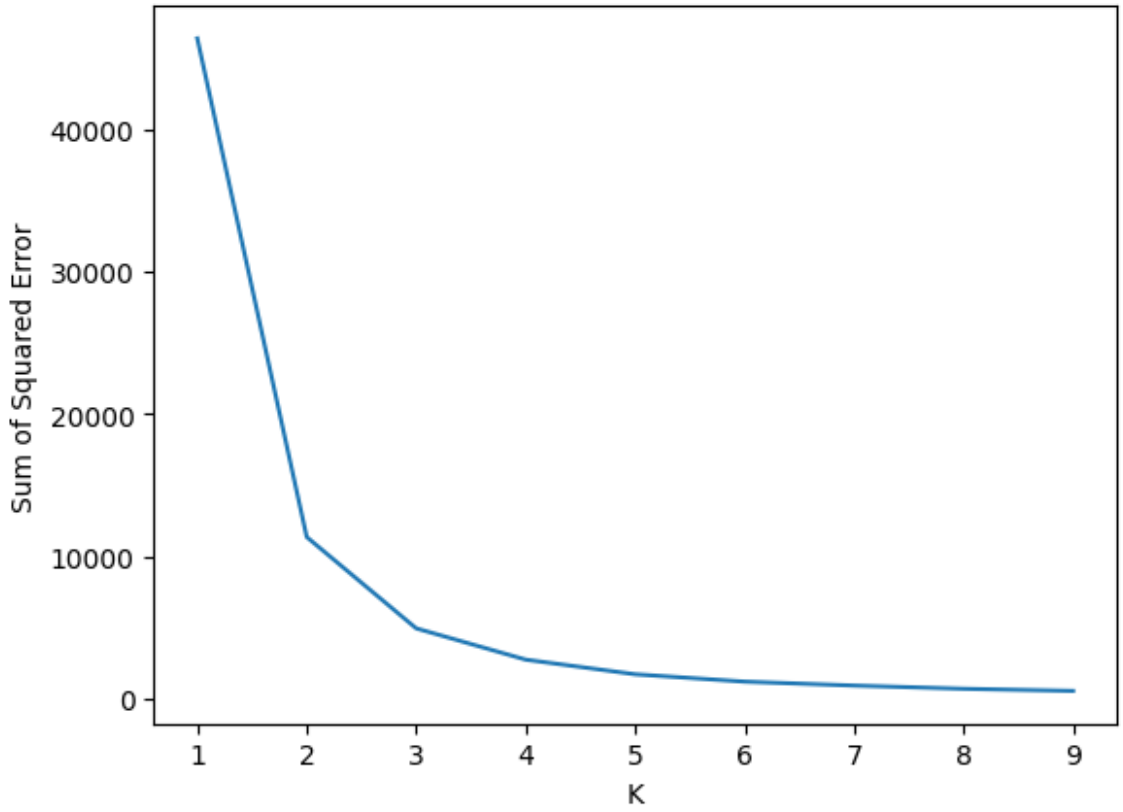
```
k_rng=range(1,10)
sse=[]
```

In [59]:

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

Out[59]:

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




# Conclusion:

By using kMeans Algorithm we are performing clustering on Quantity and Cus tomerID labels,
so we conclude that KMeans alogorithm is best fot this Dataset.