In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [3]:
```
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
```

In [4]:
```
df = pd.read_csv("sales_data_sample.csv", encoding="Latin-1")
```

In [7]:
```
df.head()
```

Out[7]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	OR
0	10107	30	95.70	2	2871.00	
1	10121	34	81.35	5	2765.90	
2	10134	41	94.74	2	3884.34	
3	10145	45	83.26	6	3746.70	
4	10159	49	100.00	14	5205.27	1(

5 rows × 25 columns

In [9]:
```
df.shape
```

Out[9]:
```
(2823, 25)
```

In [11]:
```
df.describe()
```

Out[11]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	S
count	2823.000000	2823.000000	2823.000000	2823.000000	2823.0
mean	10258.725115	35.092809	83.658544	6.466171	3553.8
std	92.085478	9.741443	20.174277	4.225841	1841.8
min	10100.000000	6.000000	26.880000	1.000000	482.1
25%	10180.000000	27.000000	68.860000	3.000000	2203.4
50%	10262.000000	35.000000	95.700000	6.000000	3184.8
75%	10333.500000	43.000000	100.000000	9.000000	4508.0
max	10425.000000	97.000000	100.000000	18.000000	14082.8

In [13]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2823 entries, 0 to 2822
Data columns (total 25 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   ORDERNUMBER       2823 non-null   int64  
 1   QUANTITYORDERED   2823 non-null   int64  
 2   PRICEEACH         2823 non-null   float64
 3   ORDERLINENUMBER   2823 non-null   int64  
 4   SALES             2823 non-null   float64
 5   ORDERDATE         2823 non-null   object 
 6   STATUS            2823 non-null   object 
 7   QTR_ID            2823 non-null   int64  
 8   MONTH_ID          2823 non-null   int64  
 9   YEAR_ID           2823 non-null   int64  
 10  PRODUCTLINE       2823 non-null   object 
 11  MSRP              2823 non-null   int64  
 12  PRODUCTCODE       2823 non-null   object 
 13  CUSTOMERNAME      2823 non-null   object 
 14  PHONE             2823 non-null   object 
 15  ADDRESSLINE1      2823 non-null   object 
 16  ADDRESSLINE2      302 non-null    object 
 17  CITY              2823 non-null   object 
 18  STATE             1337 non-null   object 
 19  POSTALCODE        2747 non-null   object 
 20  COUNTRY           2823 non-null   object 
 21  TERRITORY         1749 non-null   object 
 22  CONTACTLASTNAME   2823 non-null   object 
 23  CONTACTFIRSTNAME  2823 non-null   object 
 24  DEALSIZE          2823 non-null   object 
dtypes: float64(2), int64(7), object(16)
memory usage: 551.5+ KB
```

In [15]:

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

```
Out[15]: ORDERNUMBER            0
         QUANTITYORDERED        0
         PRICEEACH              0
         ORDERLINENUMBER        0
         SALES                  0
         ORDERDATE              0
         STATUS                 0
         QTR_ID                 0
         MONTH_ID               0
         YEAR_ID                0
         PRODUCTLINE            0
         MSRP                   0
         PRODUCTCODE            0
         CUSTOMERNAME           0
         PHONE                  0
         ADDRESSLINE1           0
         ADDRESSLINE2        2521
         CITY                   0
         STATE               1486
         POSTALCODE            76
         COUNTRY                0
         TERRITORY           1074
         CONTACTLASTNAME        0
         CONTACTFIRSTNAME       0
         DEALSIZE               0
         dtype: int64
```

```
In [17]: df.dtypes
```

```
Out[17]: ORDERNUMBER           int64
         QUANTITYORDERED       int64
         PRICEEACH           float64
         ORDERLINENUMBER       int64
         SALES               float64
         ORDERDATE            object
         STATUS               object
         QTR_ID                int64
         MONTH_ID              int64
         YEAR_ID               int64
         PRODUCTLINE          object
         MSRP                  int64
         PRODUCTCODE          object
         CUSTOMERNAME         object
         PHONE                object
         ADDRESSLINE1         object
         ADDRESSLINE2         object
         CITY                 object
         STATE                object
         POSTALCODE           object
         COUNTRY              object
         TERRITORY            object
         CONTACTLASTNAME      object
         CONTACTFIRSTNAME     object
         DEALSIZE             object
         dtype: object
```

```
In [19]: df_drop = ['ADDRESSLINE1','ADDRESSLINE2','STATUS','POSTALCODE','CITY','PHONE','C
```

```
In [21]: df = df.drop(df_drop,axis=1)
```

In [23]:
```
df.isnull().sum()
```

Out[23]:
```
ORDERNUMBER        0
QUANTITYORDERED    0
PRICEEACH          0
ORDERLINENUMBER    0
SALES              0
ORDERDATE          0
QTR_ID             0
MONTH_ID           0
YEAR_ID            0
PRODUCTLINE        0
MSRP               0
PRODUCTCODE        0
COUNTRY            0
DEALSIZE           0
dtype: int64
```

In [25]:
```
df.dtypes
```

Out[25]:
```
ORDERNUMBER          int64
QUANTITYORDERED      int64
PRICEEACH          float64
ORDERLINENUMBER      int64
SALES              float64
ORDERDATE           object
QTR_ID               int64
MONTH_ID             int64
YEAR_ID              int64
PRODUCTLINE         object
MSRP                 int64
PRODUCTCODE         object
COUNTRY             object
DEALSIZE            object
dtype: object
```

In [27]:
```
df['COUNTRY'].unique()
```

Out[27]:
```
array(['USA', 'France', 'Norway', 'Australia', 'Finland', 'Austria', 'UK',
       'Spain', 'Sweden', 'Singapore', 'Canada', 'Japan', 'Italy',
       'Denmark', 'Belgium', 'Philippines', 'Germany', 'Switzerland',
       'Ireland'], dtype=object)
```

In [29]:
```
df['PRODUCTLINE'].unique()
```

Out[29]:
```
array(['Motorcycles', 'Classic Cars', 'Trucks and Buses', 'Vintage Cars',
       'Planes', 'Ships', 'Trains'], dtype=object)
```

In [31]:
```
df['DEALSIZE'].unique()
```

Out[31]:
```
array(['Small', 'Medium', 'Large'], dtype=object)
```

In [33]:
```
productline = pd.get_dummies(df['PRODUCTLINE'])
Dealsize = pd.get_dummies(df['DEALSIZE'])
```

In [35]:
```
df = pd.concat([df, productline,Dealsize],axis=1)
```

In [37]:
```
df_drop = ['COUNTRY', 'PRODUCTLINE', 'DEALSIZE']
```

```
df = df.drop(df_drop, axis =1 )
```

In [39]:

```
df['PRODUCTCODE'] = pd.Categorical(df[ 'PRODUCTCODE']).codes
```

In [41]:

```
df.drop('ORDERDATE', axis = 1, inplace=True)
```

In [43]:

```
df.dtypes
```

Out[43]:

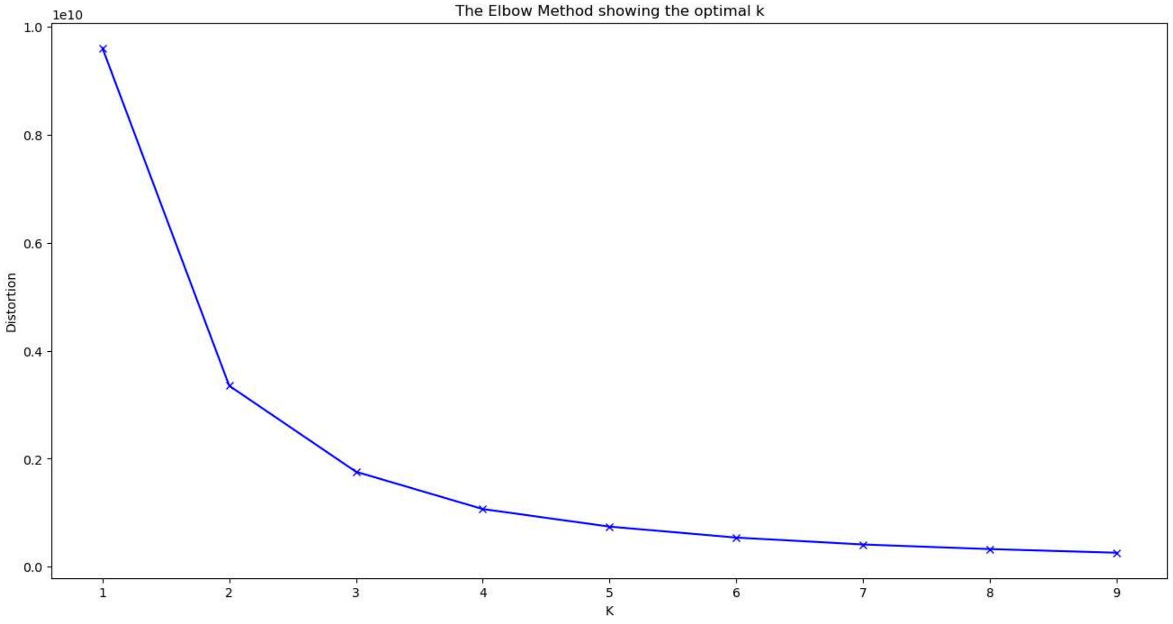
```
ORDERNUMBER            int64
QUANTITYORDERED        int64
PRICEEACH            float64
ORDERLINENUMBER        int64
SALES                float64
QTR_ID                 int64
MONTH_ID               int64
YEAR_ID                int64
MSRP                   int64
PRODUCTCODE             int8
Classic Cars            bool
Motorcycles             bool
Planes                  bool
Ships                   bool
Trains                  bool
Trucks and Buses        bool
Vintage Cars            bool
Large                   bool
Medium                  bool
Small                   bool
dtype: object
```

In [45]:

```
distortions = []
K = range(1,10)
for k in K:
    kmeanModel = KMeans(n_clusters=k)
    kmeanModel.fit(df)
    distortions.append(kmeanModel. inertia_)
```

In [47]:

```
plt.figure(figsize=(16,8))
plt.plot(K, distortions, 'bx-')
plt.xlabel('K')
plt.ylabel('Distortion')
plt.title('The Elbow Method showing the optimal k')
plt.show()
```



In [49]:
```
x_train = df.values
```

In [51]:
```
x_train.shape
```

Out[51]:
```
(2823, 20)
```

In [53]:
```
model = KMeans (n_clusters=3, random_state=2)
model =  model.fit(x_train)
predictions = model.predict(x_train)
```

In [55]:
```
unique, counts = np.unique(predictions, return_counts=True)
```

In [57]:
```
counts = counts.reshape(1,3)
```

In [59]:
```
counts_df = pd.DataFrame(counts, columns=['Cluster', 'Cluster2', 'Cluster3'])
```

In [61]:
```
counts_df.head()
```

Out[61]:

	Cluster	Cluster2	Cluster3
0	1344	398	1081

In [65]:
```
pca = PCA(n_components=2)
reduced_X = pd.DataFrame(pca.fit_transform(x_train), columns=['PCA1', 'PEA2'])
reduced_X.head()
```

Out[65]:

	PCA1	PEA2
0	-682.790370	-151.271539
1	-787.939342	-136.994834
2	330.482091	-125.876905
3	192.812426	-114.565402
4	1651.330150	-103.067424