In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
#Importing the required libraries.
```

In [2]:
```
from sklearn.cluster import KMeans, k_means
from sklearn.decomposition import PCA
```

In [6]:
```
df = pd.read_csv("sales_data_sample.csv", encoding="latin")
```

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 [8]:
```
df.shape
```

Out[8]:
```
(2823, 25)
```

In [9]:
```
df.describe()
```

Out[9]:

	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 [10]:
```
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 [11]:
```
df.isnull().sum()
```

Out[11]:
```
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 [12]:
```
df.dtypes
```

Out[12]:
```
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 [13]:
```
df_drop  = ['ADDRESSLINE1', 'ADDRESSLINE2', 'STATUS','POSTALCODE', 'CITY', 'TERR
df = df.drop(df_drop, axis=1)
```

In [14]:
```
df.isnull().sum()
```

Out[14]:
```
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 [15]:
```
df.dtypes
```

Out[15]:
```
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 [17]:
```
df['COUNTRY'].unique()
```

Out[17]:
```
array(['USA', 'France', 'Norway', 'Australia', 'Finland', 'Austria', 'UK',
       'Spain', 'Sweden', 'Singapore', 'Canada', 'Japan', 'Italy',
       'Denmark', 'Belgium', 'Philippines', 'Germany', 'Switzerland',
       'Ireland'], dtype=object)
```

In [18]:
```
df['PRODUCTLINE'].unique()
```

Out[18]:
```
array(['Motorcycles', 'Classic Cars', 'Trucks and Buses', 'Vintage Cars',
       'Planes', 'Ships', 'Trains'], dtype=object)
```

In [19]:
```
df['DEALSIZE'].unique()
```

Out[19]:
```
array(['Small', 'Medium', 'Large'], dtype=object)
```

In [20]:
```
productline = pd.get_dummies(df['PRODUCTLINE'])
Dealsize = pd.get_dummies(df['DEALSIZE'])
```

In [21]:
```
df = pd.concat([df,productline,Dealsize], axis = 1)
```

In [22]:
```
df_drop  = ['COUNTRY','PRODUCTLINE','DEALSIZE']
df = df.drop(df_drop, axis=1)
```

In [23]:
```
df['PRODUCTCODE'] = pd.Categorical(df['PRODUCTCODE']).codes
```

In [24]:
```
df.drop('ORDERDATE', axis=1, inplace=True)
```

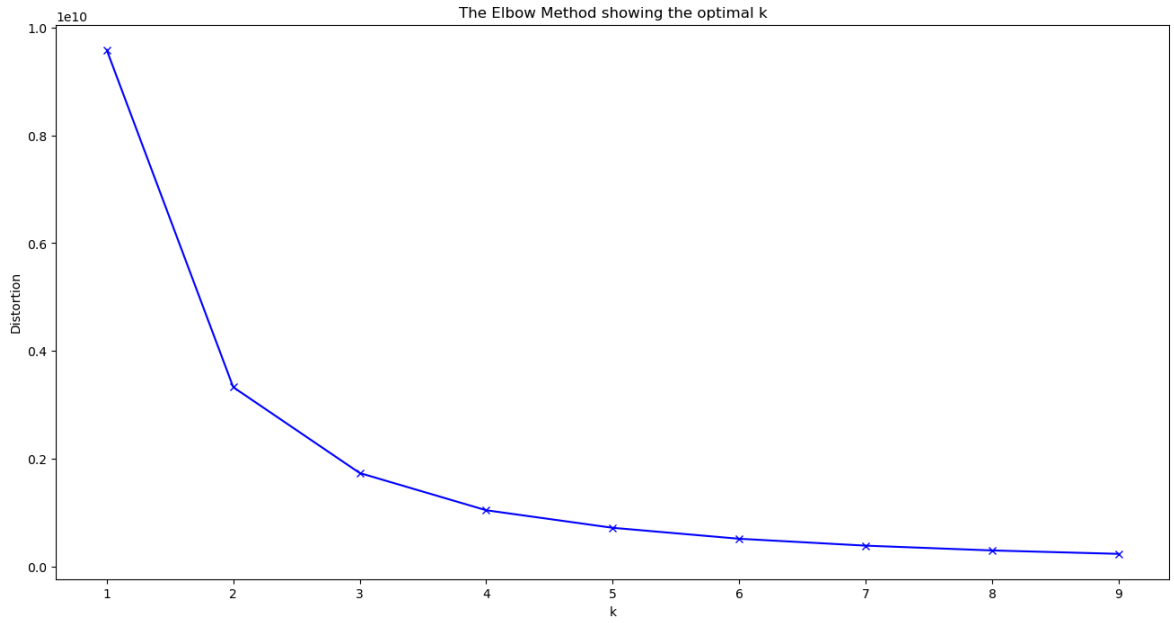
In [25]:
```
df.dtypes
```

Out[25]:
```
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
```

Plotting the Elbow Plot to determine the number of clusters.

In [26]:
```
distortions = []
K = range(1,10)
for k in K:
    kmeanModel = KMeans(n_clusters=k)
    kmeanModel.fit(df)
    distortions.append(kmeanModel.inertia_)
```

In [27]:
```
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 [30]:
```
X_train = df.values
```

In [31]:
```
X_train.shape
```

Out[31]: (2823, 19)

In [32]:
```
model = KMeans(n_clusters=3,random_state=2)
model = model.fit(X_train)
predictions = model.predict(X_train)
```

In [33]:
```
unique,counts = np.unique(predictions,return_counts=True)
```

In [34]:
```
counts = counts.reshape(1,3)
```

In [35]:
```
counts_df = pd.DataFrame(counts,columns=['Cluster1','Cluster2','Cluster3'])
```

In [36]:
```
counts_df.head()
```

Out[36]:

	Cluster1	Cluster2	Cluster3
0	1344	398	1081

Visualization

In [37]:
```
pca = PCA(n_components=2)
```

In [38]:
```
reduced_X = pd.DataFrame(pca.fit_transform(X_train),columns=['PCA1','PCA2'])
```

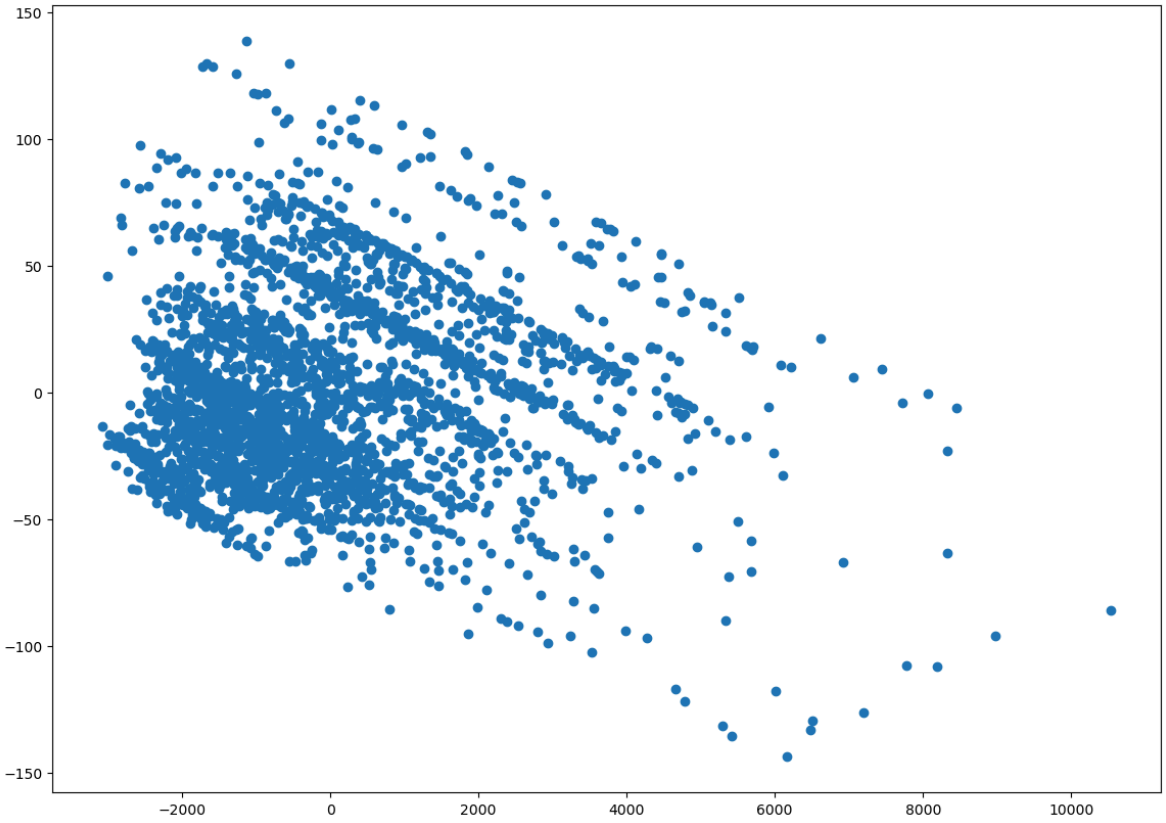
In [39]:
```
reduced_X.head()
```

Out[39]:

	PCA1	PCA2
0	-682.488323	42.819535
1	-787.665502	41.694991
2	330.732170	26.481208
3	193.040232	26.285766
4	1651.532874	6.891196

In [40]:
```
#Plotting the normal Scatter Plot
plt.figure(figsize=(14,10))
plt.scatter(reduced_X['PCA1'],reduced_X['PCA2'])
```

Out[40]: <matplotlib.collections.PathCollection at 0x2256e9c5510>



In [41]:
```
model.cluster_centers_
```

Out[41]:
```
array([[3.07723214e+01, 6.97585491e+01, 6.65178571e+00, 2.10716933e+03,
        2.71354167e+00, 7.08184524e+00, 2.00381696e+03, 7.81674107e+01,
        6.25811012e+01, 2.62648810e-01, 1.21279762e-01, 1.28720238e-01,
        1.01190476e-01, 3.79464286e-02, 9.30059524e-02, 2.55208333e-01,
        2.08166817e-17, 4.61309524e-02, 9.53869048e-01],
       [4.44623116e+01, 9.98998241e+01, 5.77135678e+00, 7.00029073e+03,
        2.70100503e+00, 7.03015075e+00, 2.00387688e+03, 1.44356784e+02,
        3.23869347e+01, 5.35175879e-01, 1.03015075e-01, 7.03517588e-02,
        2.01005025e-02, 1.25628141e-02, 1.28140704e-01, 1.30653266e-01,
        3.94472362e-01, 6.05527638e-01, 0.00000000e+00],
       [3.70148011e+01, 9.49606383e+01, 6.49121184e+00, 4.08369802e+03,
        2.72895467e+00, 7.12858464e+00, 2.00379001e+03, 1.12681776e+02,
        5.06965772e+01, 3.70952821e-01, 1.17483811e-01, 9.71322849e-02,
        8.32562442e-02, 1.94264570e-02, 1.15633673e-01, 1.96114709e-01,
        2.08166817e-17, 1.00000000e+00, 1.66533454e-16]])
```

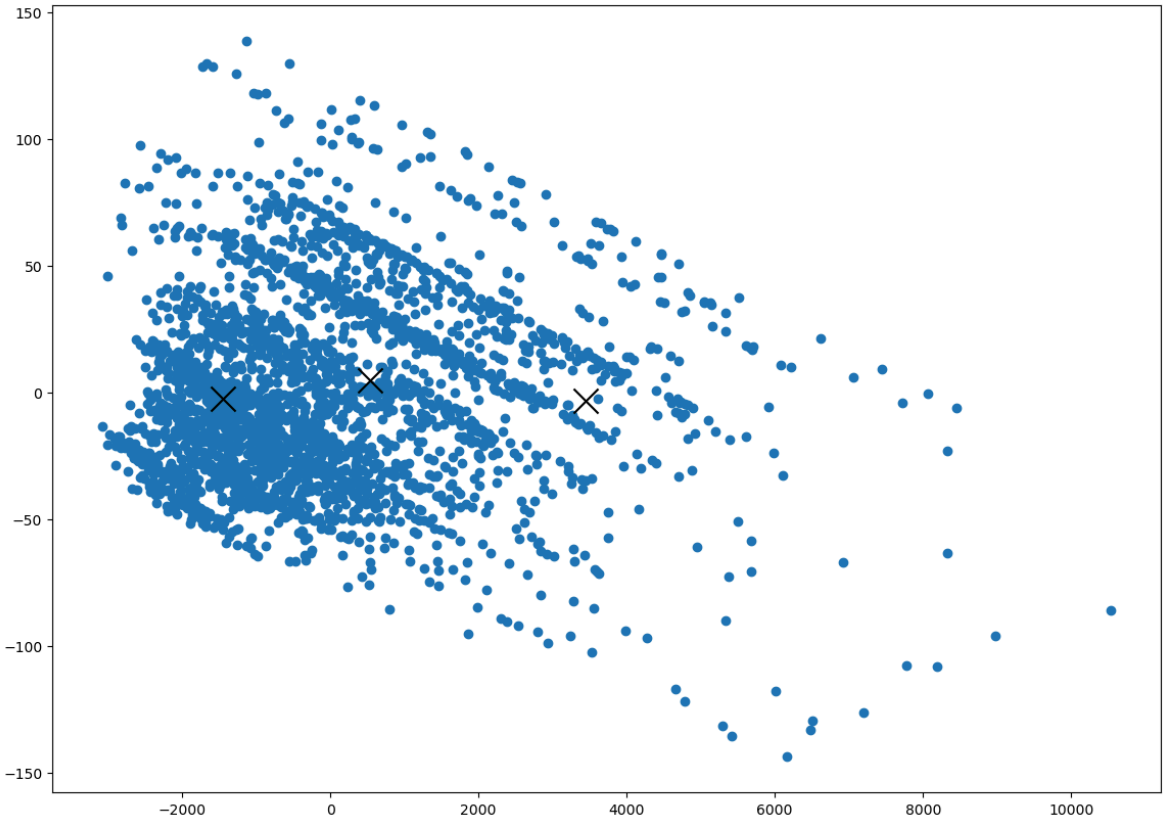
In [42]:
```
reduced_centers = pca.transform(model.cluster_centers_)
```

In [43]:
```
reduced_centers
```

Out[43]:
```
array([[-1.44698921e+03, -2.68456273e+00],
       [ 3.44678179e+03, -3.38057613e+00],
       [ 5.30004017e+02,  4.58235116e+00]])
```

In [44]:
```
plt.figure(figsize=(14,10))
plt.scatter(reduced_X['PCA1'],reduced_X['PCA2'])
plt.scatter(reduced_centers[:,0],reduced_centers[:,1],color='black',marker='x',s
```

Out[44]:
```
<matplotlib.collections.PathCollection at 0x2256f379510>
```



In [45]:
```
reduced_X['Clusters'] = predictions
```

In [46]:
```
reduced_X.head()
```

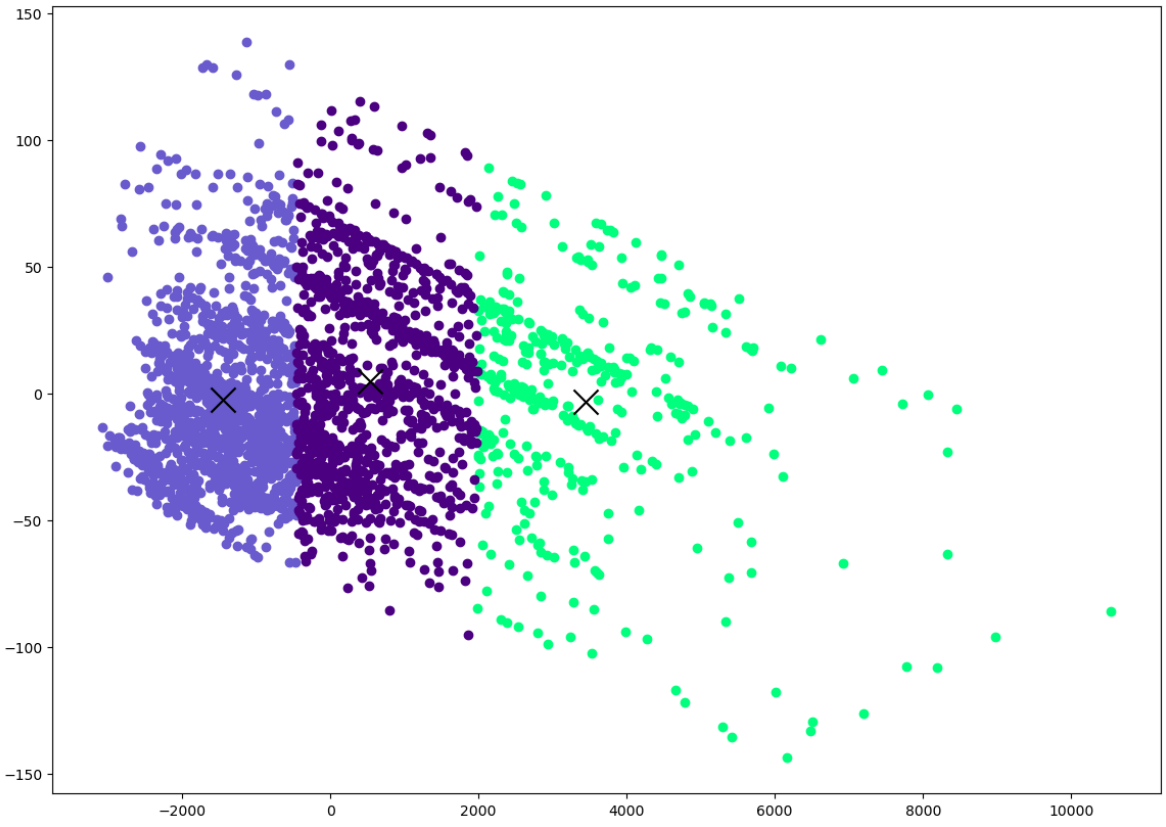
Out[46]:

	PCA1	PCA2	Clusters
0	-682.488323	42.819535	0
1	-787.665502	41.694991	0
2	330.732170	26.481208	2
3	193.040232	26.285766	2
4	1651.532874	6.891196	2

In [47]:
```
#Plotting the clusters
plt.figure(figsize=(14,10))
#                    taking the cluster number and first column           takin
plt.scatter(reduced_X[reduced_X['Clusters'] == 0].loc[:,'PCA1'],reduced_X[reduce
plt.scatter(reduced_X[reduced_X['Clusters'] == 1].loc[:,'PCA1'],reduced_X[reduce
plt.scatter(reduced_X[reduced_X['Clusters'] == 2].loc[:,'PCA1'],reduced_X[reduce


plt.scatter(reduced_centers[:,0],reduced_centers[:,1],color='black',marker='x',s
```

Out[47]: <matplotlib.collections.PathCollection at 0x2256f40a790>



In [ ]: