Aim:

Implement K-Means clustering/hierarchical clustering on sales_data_sample.csv dataset. Determine the number of clusters using the Elbow method.

Importing Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans,k_means
```

In [2]:
```
df = pd.read_csv("sales_data_sample.csv",encoding='latin-1')
```

In [3]:
```
df
```

Out[3]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORI
0	10107	30	95.70	2	2871.00	
1	10121	34	81.35	5	2765.90	5/7/
2	10134	41	94.74	2	3884.34	7/1/
3	10145	45	83.26	6	3746.70	
4	10159	49	100.00	14	5205.27	1
...	...	...	...	...	...	
2818	10350	20	100.00	15	2244.40	

In [4]:
```
df.head()
```

Out[4]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDA
0	10107	30	95.70	2	2871.00	2/24/20 0:
1	10121	34	81.35	5	2765.90	5/7/2003 0:
2	10134	41	94.74	2	3884.34	7/1/2003 0:
3	10145	45	83.26	6	3746.70	8/25/20 0:
4	10159	49	100.00	14	5205.27	10/10/20 0:

5 rows × 25 columns

In [5]:
```
df.describe()
```

Out[5]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES
count	2823.000000	2823.000000	2823.000000	2823.000000	2823.000000
mean	10258.725115	35.092809	83.658544	6.466171	3553.889072
std	92.085478	9.741443	20.174277	4.225841	1841.865106
min	10100.000000	6.000000	26.880000	1.000000	482.130000
25%	10180.000000	27.000000	68.860000	3.000000	2203.430000
50%	10262.000000	35.000000	95.700000	6.000000	3184.800000
75%	10333.500000	43.000000	100.000000	9.000000	4508.000000
max	10425.000000	97.000000	100.000000	18.000000	14082.800000

In [6]:
```
df.tail()
```

Out[6]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDER
2818	10350	20	100.00	15	2244.40	12/
2819	10373	29	100.00	1	3978.51	1/3
2820	10386	43	100.00	4	5417.57	3/1/200
2821	10397	34	62.24	1	2116.16	3/2
2822	10414	47	65.52	9	3079.44	5/6/200

5 rows × 25 columns

In [7]:
```
df.columns
```

Out[7]:
```
Index(['ORDERNUMBER', 'QUANTITYORDERED', 'PRICEEACH', 'ORDERLINENUMBER',
       'SALES', 'ORDERDATE', 'STATUS', 'QTR_ID', 'MONTH_ID', 'YEAR_ID',
       'PRODUCTLINE', 'MSRP', 'PRODUCTCODE', 'CUSTOMERNAME', 'PHONE',
       'ADDRESSLINE1', 'ADDRESSLINE2', 'CITY', 'STATE', 'POSTALCODE',
       'COUNTRY', 'TERRITORY', 'CONTACTLASTNAME', 'CONTACTFIRSTNAME',
       'DEALSIZE'],
      dtype='object')
```

In [8]:
```
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 [9]:
```
df.isnull().sum()
```

Out[9]:
```
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 [10]:
```
df.columns
```

Out[10]:
```
Index(['ORDERNUMBER', 'QUANTITYORDERED', 'PRICEEACH', 'ORDERLINENUMBER',
       'SALES', 'ORDERDATE', 'STATUS', 'QTR_ID', 'MONTH_ID', 'YEAR_ID',
       'PRODUCTLINE', 'MSRP', 'PRODUCTCODE', 'CUSTOMERNAME', 'PHONE',
       'ADDRESSLINE1', 'ADDRESSLINE2', 'CITY', 'STATE', 'POSTALCODE',
       'COUNTRY', 'TERRITORY', 'CONTACTLASTNAME', 'CONTACTFIRSTNAME',
       'DEALSIZE'],
      dtype='object')
```

In [11]:
```
df_drop=['ADDRESSLINE1','ADDRESSLINE2','STATUS','POSTALCODE','CITY','TERRITORY
        'CONTACTLASTNAME','CONTACTFIRSTNAME','ORDERNUMBER','CUSTOMERNAME']
```

In [12]:
```
df=df.drop(df_drop,axis=1)
```

In [13]:
```
df['COUNTRY'].unique()
```

Out[13]:
```
array(['USA', 'France', 'Norway', 'Australia', 'Finland', 'Austria', 'UK',
       'Spain', 'Sweden', 'Singapore', 'Canada', 'Japan', 'Italy',
       'Denmark', 'Belgium', 'Philippines', 'Germany', 'Switzerland',
       'Ireland'], dtype=object)
```

In [14]:
```
df['PRODUCTLINE'].unique()
```

Out[14]:
```
array(['Motorcycles', 'Classic Cars', 'Trucks and Buses', 'Vintage Cars',
       'Planes', 'Ships', 'Trains'], dtype=object)
```

In [15]:
```
df['DEALSIZE'].unique()
```

Out[15]:
```
array(['Small', 'Medium', 'Large'], dtype=object)
```

In [16]:
```
productline=pd.get_dummies(df['PRODUCTLINE'])
dealsize=pd.get_dummies(df['DEALSIZE'])
```

In [17]:
```
df=pd.concat([df,productline,dealsize],axis=1)
```

In [18]:
```
df_drop=['COUNTRY','PRODUCTLINE','DEALSIZE']
df=df.drop(df_drop,axis=1)
```

In [19]:
```
df['PRODUCTCODE']=pd.Categorical(df['PRODUCTCODE']).codes
```

In [20]:
```
df.drop('ORDERDATE',axis=1,inplace=True)
```

In [21]:
```
df.dtypes
```

Out[21]:
```
QUANTITYORDERED       int64
PRICEEACH           float64
ORDERLINENUMBER       int64
SALES               float64
QTR_ID                int64
MONTH_ID              int64
YEAR_ID               int64
MSRP                  int64
PRODUCTCODE            int8
Classic Cars          uint8
Motorcycles           uint8
Planes                uint8
Ships                 uint8
Trains                uint8
Trucks and Buses      uint8
Vintage Cars          uint8
Large                 uint8
Medium                uint8
Small                 uint8
dtype: object
```

In [22]:
```
dst=[]
K=range(1,10)
for k in K:
    kmeanModel=KMeans(n_clusters=k)
    kmeanModel.fit(df)
    dst.append(kmeanModel.inertia_)
```

In [23]:
```
plt.figure(figsize=(16,8))
plt.plot(K,dst,'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('The Elbow Method showing the optimal K')
plt.show()
```