In [31]:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import scale
from numpy import random,float,array
import numpy as np
import seaborn as sns
```

executed in 27ms, finished 01:46:14 2021-11-20

In [4]:

```
crime=pd.read_csv("crime_data.csv")
```

executed in 22ms, finished 00:41:58 2021-11-20

In [5]:

```
crime.head()
```

executed in 59ms, finished 00:42:20 2021-11-20

Out[5]:

	Place	Murder	Assault	UrbanPop	Rape
0	Alabama	13.2	236	58	21.2
1	Alaska	10.0	263	48	44.5
2	Arizona	8.1	294	80	31.0
3	Arkansas	8.8	190	50	19.5
4	California	9.0	276	91	40.6

```
crime.shape
```

In [37]:

```
crime.shape
```

executed in 56ms, finished 04:27:30 2021-11-20

Out[37]:

```
(50, 6)
```

In [6]:

```
def norm_func(i):
    x=(i-i.min())/(i.max()-i.min())
    return(x)
```

executed in 11ms, finished 00:45:12 2021-11-20

In [7]:

```
df_norm=norm_func(crime.iloc[:,1:])
df_norm.describe()
```

executed in 268ms, finished 00:46:37 2021-11-20

Out[7]:

	Murder	Assault	UrbanPop	Rape
count	50.000000	50.000000	50.000000	50.000000
mean	0.420964	0.430685	0.568475	0.360000
std	0.262380	0.285403	0.245335	0.242025
min	0.000000	0.000000	0.000000	0.000000
25%	0.197289	0.219178	0.381356	0.200904
50%	0.388554	0.390411	0.576271	0.330749
75%	0.629518	0.698630	0.775424	0.487726
max	1.000000	1.000000	1.000000	1.000000

In [8]:

```
from scipy.cluster.hierarchy import linkage

import scipy.cluster.hierarchy as sch
```

executed in 8ms, finished 00:48:52 2021-11-20

In [9]:

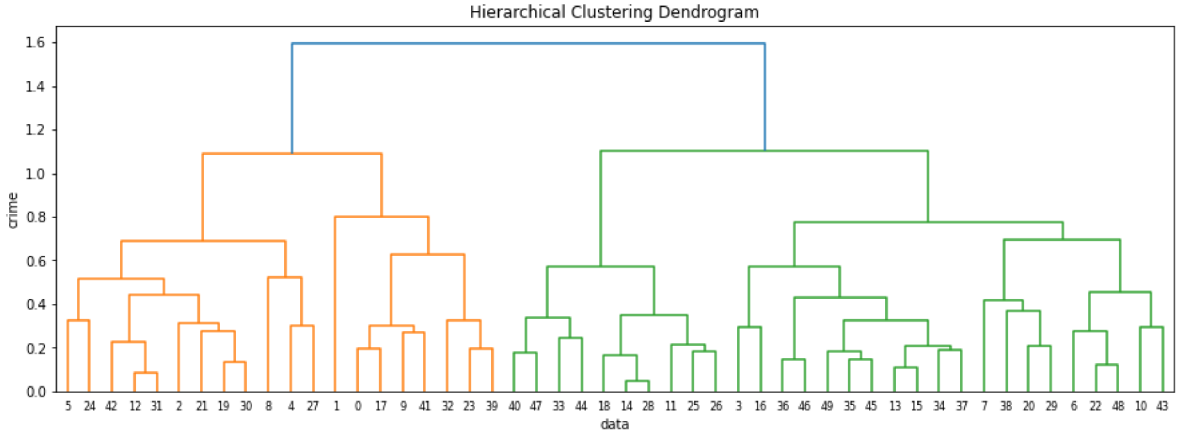
```
z=linkage(df_norm,method="complete",metric="euclidean")
```

executed in 19ms, finished 00:50:48 2021-11-20

In [10]:

```
plt.figure(figsize=(15,5))
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('data')
plt.ylabel('crime')
sch.dendrogram(z,leaf_rotation=0.,leaf_font_size=8.,)
plt.show()
```

executed in 8.80s, finished 00:55:05 2021-11-20



In [11]:

```
crime.corr()
```

executed in 124ms, finished 00:55:45 2021-11-20

Out[11]:

	Murder	Assault	UrbanPop	Rape
Murder	1.000000	0.801873	0.069573	0.563579
Assault	0.801873	1.000000	0.258872	0.665241
UrbanPop	0.069573	0.258872	1.000000	0.411341
Rape	0.563579	0.665241	0.411341	1.000000

In [35]:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
crime_subset = pd.DataFrame(scaler.fit_transform(crime.iloc[:,1:]))
crime_subset
```

executed in 105ms, finished 01:51:38 2021-11-20

Out[35]:

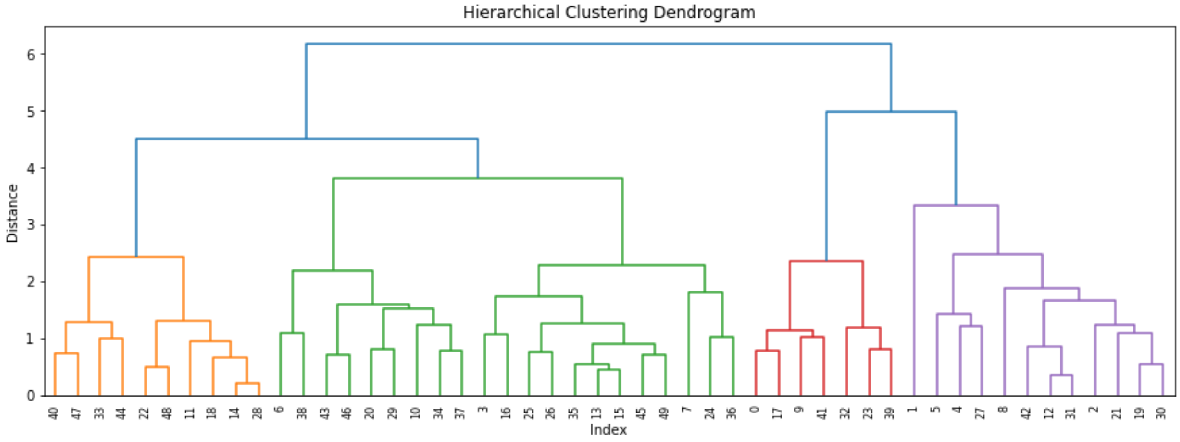
	0	1	2	3	4
0	1.255179	0.790787	-0.526195	-0.003451	1.651616
1	0.513019	1.118060	-1.224067	2.509424	-0.540795
2	0.072361	1.493817	1.009122	1.053466	-0.540795
3	0.234708	0.233212	-1.084492	-0.186794	-1.271598
4	0.281093	1.275635	1.776781	2.088814	-0.540795
5	0.025976	0.402909	0.869548	1.883901	-0.540795
6	-1.040880	-0.736484	0.799761	-1.092723	0.920813
7	-0.437875	0.815030	0.450825	-0.585834	-1.271598
8	1.765415	1.990786	1.009122	1.150530	-0.540795
9	2.229265	0.487757	-0.386621	0.492653	1.651616
10	-0.577030	-1.512241	1.218484	-0.111300	0.920813
11	-1.203228	-0.615272	-0.805344	-0.758392	0.190009
12	0.605789	0.948363	1.218484	0.298525	-0.540795
13	-0.136372	-0.700121	-0.037685	-0.025021	-1.271598
14	-1.295998	-1.391029	-0.595982	-1.071153	0.190009
15	-0.414682	-0.675878	0.032102	-0.348567	-1.271598
16	0.443441	-0.748605	-0.944918	-0.531910	-1.271598
17	1.765415	0.948363	0.032102	0.104398	1.651616
18	-1.319191	-1.063757	-1.014705	-1.448624	0.190009
19	0.814521	1.566544	0.101889	0.708350	-0.540795
20	-0.785763	-0.263757	1.358058	-0.531910	0.920813
21	1.000062	1.021090	0.590399	1.495646	-0.540795
22	-1.180036	-1.197090	0.032102	-0.682898	0.190009
23	1.927762	1.069575	-1.503215	-0.445631	1.651616
24	0.281093	0.087757	0.311251	0.751490	-1.271598
25	-0.414682	-0.748605	-0.875131	-0.521125	-1.271598
26	-0.808955	-0.833454	-0.247047	-0.510340	-1.271598
27	1.023254	0.984726	1.078909	2.671197	-0.540795
28	-1.319191	-1.378908	-0.665769	-1.265281	0.190009
29	-0.089987	-0.142545	1.637207	-0.262288	0.920813
30	0.837714	1.384726	0.311251	1.172100	-0.540795
31	0.768136	1.008969	1.427845	0.525008	-0.540795

	0	1	2	3	4
32	1.208794	2.015028	-1.433428	-0.553480	1.651616
33	-1.620693	-1.524362	-1.503215	-1.502548	0.190009
34	-0.113180	-0.615272	0.660186	0.018119	0.920813
35	-0.275527	-0.239515	0.171676	-0.132870	-1.271598
36	-0.669800	-0.142545	0.101889	0.870123	-1.271598
37	-0.345105	-0.784969	0.450825	-0.682898	0.920813
38	-1.017688	0.039273	1.497632	-1.394700	0.920813
39	1.533490	1.311999	-1.224067	0.136752	1.651616
40	-0.924918	-1.027393	-1.433428	-0.909380	0.190009
41	1.255179	0.208970	-0.456408	0.611287	1.651616
42	1.139217	0.366545	1.009122	0.460298	-0.540795
43	-1.064073	-0.615272	1.009122	0.179892	0.920813
44	-1.295998	-1.487999	-2.340661	-1.081938	0.190009
45	0.165131	-0.178909	-0.177259	-0.057376	-1.271598
46	-0.878533	-0.312242	0.520612	0.535792	0.920813
47	-0.484260	-1.087999	-1.852151	-1.286851	0.190009
48	-1.203228	-1.427393	0.032102	-1.125078	0.190009
49	-0.229142	-0.118303	-0.386621	-0.607404	-1.271598

In [36]:

```
from scipy.cluster.hierarchy import linkage
import scipy.cluster.hierarchy as sch # for creating dendrogram
#p = np.array(df_norm) # converting into numpy array format
z = linkage(crime_subset, method="complete",metric="euclidean")
plt.figure(figsize=(15, 5))
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Index')
plt.ylabel('Distance')
sch.dendrogram(
    z,
    #leaf_rotation=0.,  # rotates the x axis labels
    #leaf_font_size=8.,  # font size for the x axis labels
)
plt.show()
```

executed in 7.91s, finished 01:52:06 2021-11-20



In [12]:

```
k=list(range(2,5))
```

executed in 10ms, finished 00:56:32 2021-11-20

In [14]:

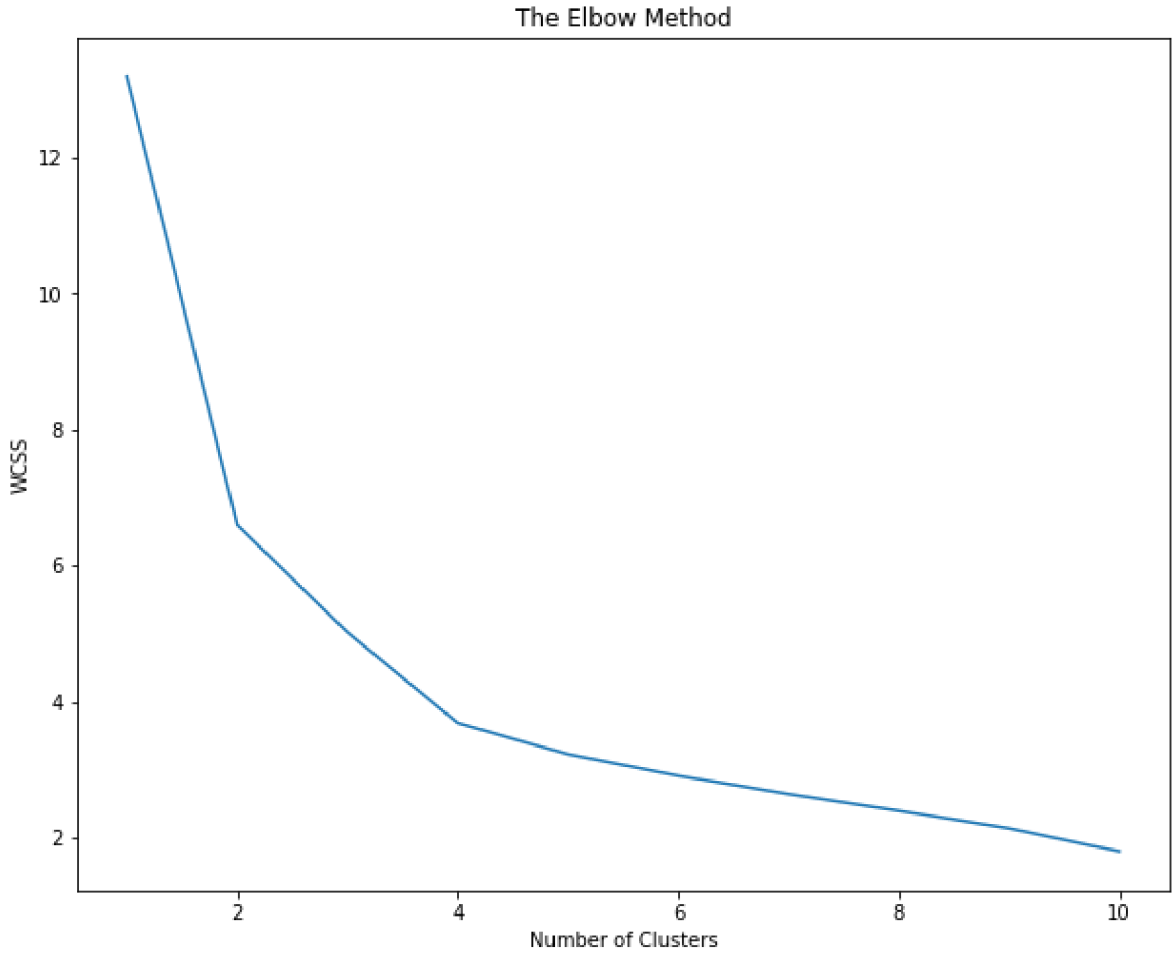
```
from sklearn.cluster import KMeans
from scipy.spatial.distance import cdist
import numpy as np
```

executed in 9ms, finished 00:58:45 2021-11-20

In [15]:

```
from sklearn.cluster import KMeans
fig = plt.figure(figsize=(10, 8))
WCSS = []
for i in range(1, 11):
    clf = KMeans(n_clusters=i)
    clf.fit(df_norm)
    WCSS.append(clf.inertia_) # inertia is another name for WCSS
plt.plot(range(1, 11), WCSS)
plt.title('The Elbow Method')
plt.ylabel('WCSS')
plt.xlabel('Number of Clusters')
plt.show()
```

executed in 4.75s, finished 01:01:12 2021-11-20



In [16]:

```
clf = KMeans(n_clusters=5)
y_kmeans = clf.fit_predict(df_norm)
```

executed in 347ms, finished 01:30:47 2021-11-20

In [17]:

```
y_kmeans
#clf.cluster_centers_
clf.labels_
```

executed in 24ms, finished 01:31:04 2021-11-20

Out[17]:

```
array([4, 1, 1, 0, 1, 1, 3, 0, 1, 4, 3, 2, 1, 0, 2, 0, 0, 4, 2, 1, 3, 1,
       2, 4, 0, 0, 0, 1, 2, 3, 1, 1, 4, 2, 3, 0, 0, 3, 3, 4, 2, 4, 1, 3,
       2, 0, 3, 2, 2, 0])
```

In [18]:

```
md=pd.Series(y_kmeans)  # converting numpy array into pandas series object
crime['clust']=md # creating a  new column and assigning it to new column
crime.describe()
```

executed in 322ms, finished 01:31:58 2021-11-20

Out[18]:

	Murder	Assault	UrbanPop	Rape	clust
count	50.00000	50.000000	50.000000	50.000000	50.000000
mean	7.78800	170.760000	65.540000	21.232000	1.740000
std	4.35551	83.337661	14.474763	9.366385	1.382249
min	0.80000	45.000000	32.000000	7.300000	0.000000
25%	4.07500	109.000000	54.500000	15.075000	1.000000
50%	7.25000	159.000000	66.000000	20.100000	2.000000
75%	11.25000	249.000000	77.750000	26.175000	3.000000
max	17.40000	337.000000	91.000000	46.000000	4.000000

In [19]:

```
crime.iloc[:,1:7].groupby(crime.clust).mean()
```

executed in 70ms, finished 01:32:36 2021-11-20

Out[19]:

clust	Murder	Assault	UrbanPop	Rape	clust
0	6.975000	148.416667	62.333333	19.775000	0
1	10.966667	264.000000	76.500000	33.608333	1
2	2.680000	70.100000	51.000000	10.910000	2
3	4.955556	125.444444	80.111111	17.788889	3
4	14.671429	251.285714	54.285714	21.685714	4

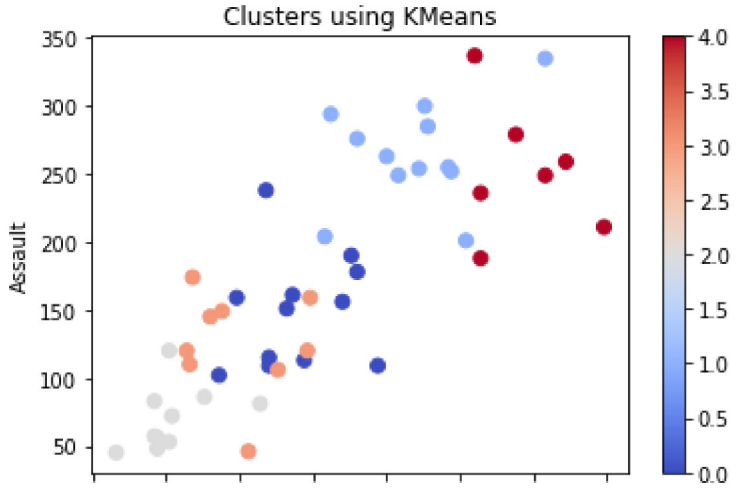
In [21]:

```
crime.plot(x="Murder",y ="Assault",c=clf.labels_,kind="scatter",s=50 ,cmap=plt.cm.coolwarm)
plt.title('Clusters using KMeans')
```

executed in 1.72s, finished 01:33:21 2021-11-20

Out[21]:

```
Text(0.5, 1.0, 'Clusters using KMeans')
```



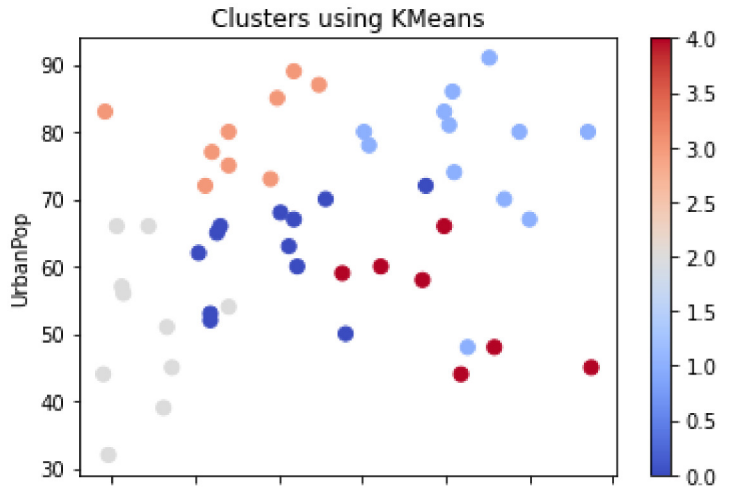
In [22]:

```
crime.plot(x ="Assault",y="UrbanPop",c=clf.labels_,kind="scatter",s=50 ,cmap=plt.cm.coolwar
plt.title('Clusters using KMeans')
```

executed in 1.40s, finished 01:35:03 2021-11-20

Out[22]:

```
Text(0.5, 1.0, 'Clusters using KMeans')
```



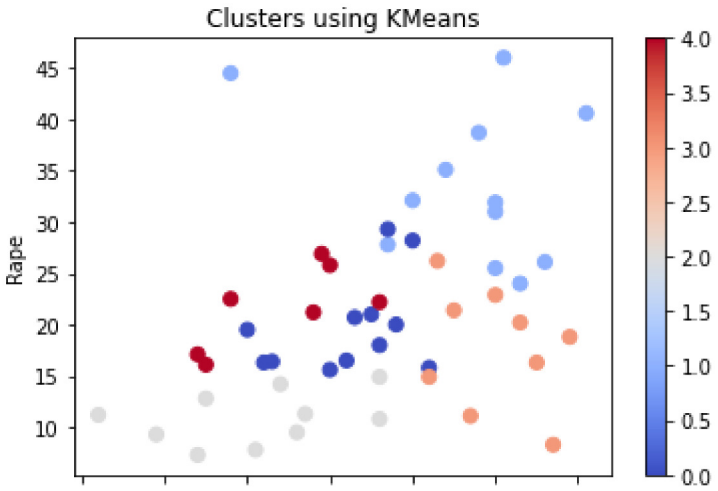
In [23]:

```
crime.plot(x ="UrbanPop",y="Rape",c=clf.labels_,kind="scatter",s=50 ,cmap=plt.cm.coolwarm)
plt.title('Clusters using KMeans')
```

executed in 1.64s, finished 01:36:44 2021-11-20

Out[23]:

```
Text(0.5, 1.0, 'Clusters using KMeans')
```



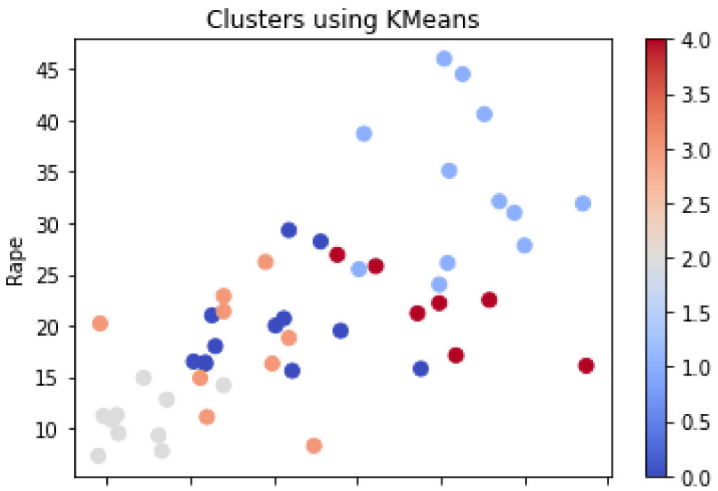
In [24]:

```
crime.plot(x ="Assault",y="Rape",c=clf.labels_,kind="scatter",s=50 ,cmap=plt.cm.coolwarm)
plt.title('Clusters using KMeans')
```

executed in 2.98s, finished 01:38:21 2021-11-20

Out[24]:

```
Text(0.5, 1.0, 'Clusters using KMeans')
```



In [25]:

```
clf.inertia_
```

executed in 16ms, finished 01:39:17 2021-11-20

Out[25]:

```
3.183157731676654
```

In [26]:

```
WCSS
```

executed in 15ms, finished 01:40:35 2021-11-20

Out[26]:

```
[13.184122550256445,
 6.596893867946199,
 5.0184999914891115,
 3.683456153585915,
 3.2232080552490108,
 2.909026352323765,
 2.6385756760229797,
 2.3935204406497688,
 2.1339551608073797,
 1.7961618985380747]
```