In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import normalize
```

In [2]:
```python
# Import Dataset
crime = pd.read_csv('crime_data.csv')
crime
```

Out[2]:

	Unnamed: 0	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
5	Colorado	7.9	204	78	38.7
6	Connecticut	3.3	110	77	11.1
7	Delaware	5.9	238	72	15.8
8	Florida	15.4	335	80	31.9
9	Georgia	17.4	211	60	25.8
10	Hawaii	5.3	46	83	20.2
11	Idaho	2.6	120	54	14.2
12	Illinois	10.4	249	83	24.0
13	Indiana	7.2	113	65	21.0
14	Iowa	2.2	56	57	11.3
15	Kansas	6.0	115	66	18.0
16	Kentucky	9.7	109	52	16.3
17	Louisiana	15.4	249	66	22.2
18	Maine	2.1	83	51	7.8
19	Maryland	11.3	300	67	27.8
20	Massachusetts	4.4	149	85	16.3
21	Michigan	12.1	255	74	35.1
22	Minnesota	2.7	72	66	14.9
23	Mississippi	16.1	259	44	17.1
24	Missouri	9.0	178	70	28.2
25	Montana	6.0	109	53	16.4
26	Nebraska	4.3	102	62	16.5
27	Nevada	12.2	252	81	46.0
28	New Hampshire	2.1	57	56	9.5
29	New Jersey	7.4	159	89	18.8
30	New Mexico	11.4	285	70	32.1
31	New York	11.1	254	86	26.1
32	North Carolina	13.0	337	45	16.1

	Unnamed: 0	Murder	Assault	UrbanPop	Rape
33	North Dakota	0.8	45	44	7.3
34	Ohio	7.3	120	75	21.4
35	Oklahoma	6.6	151	68	20.0
36	Oregon	4.9	159	67	29.3
37	Pennsylvania	6.3	106	72	14.9
38	Rhode Island	3.4	174	87	8.3
39	South Carolina	14.4	279	48	22.5
40	South Dakota	3.8	86	45	12.8
41	Tennessee	13.2	188	59	26.9
42	Texas	12.7	201	80	25.5
43	Utah	3.2	120	80	22.9
44	Vermont	2.2	48	32	11.2
45	Virginia	8.5	156	63	20.7
46	Washington	4.0	145	73	26.2
47	West Virginia	5.7	81	39	9.3
48	Wisconsin	2.6	53	66	10.8
49	Wyoming	6.8	161	60	15.6

In [3]:
```
crime2 = crime.drop(['Unnamed: 0'],axis=1)
crime2
```

Out[3]:

	Murder	Assault	UrbanPop	Rape
0	13.2	236	58	21.2
1	10.0	263	48	44.5
2	8.1	294	80	31.0
3	8.8	190	50	19.5
4	9.0	276	91	40.6
5	7.9	204	78	38.7
6	3.3	110	77	11.1
7	5.9	238	72	15.8
8	15.4	335	80	31.9
9	17.4	211	60	25.8
10	5.3	46	83	20.2
11	2.6	120	54	14.2
12	10.4	249	83	24.0
13	7.2	113	65	21.0
14	2.2	56	57	11.3
15	6.0	115	66	18.0
16	9.7	109	52	16.3
17	15.4	249	66	22.2
18	2.1	83	51	7.8
19	11.3	300	67	27.8
20	4.4	149	85	16.3
21	12.1	255	74	35.1
22	2.7	72	66	14.9
23	16.1	259	44	17.1
24	9.0	178	70	28.2
25	6.0	109	53	16.4
26	4.3	102	62	16.5
27	12.2	252	81	46.0
28	2.1	57	56	9.5
29	7.4	159	89	18.8
30	11.4	285	70	32.1
31	11.1	254	86	26.1
32	13.0	337	45	16.1
33	0.8	45	44	7.3

	Murder	Assault	UrbanPop	Rape
34	7.3	120	75	21.4
35	6.6	151	68	20.0
36	4.9	159	67	29.3
37	6.3	106	72	14.9
38	3.4	174	87	8.3
39	14.4	279	48	22.5
40	3.8	86	45	12.8
41	13.2	188	59	26.9
42	12.7	201	80	25.5
43	3.2	120	80	22.9
44	2.2	48	32	11.2
45	8.5	156	63	20.7
46	4.0	145	73	26.2
47	5.7	81	39	9.3
48	2.6	53	66	10.8
49	6.8	161	60	15.6

In [4]:
```
crime2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Murder    50 non-null     float64
 1   Assault   50 non-null     int64  
 2   UrbanPop  50 non-null     int64  
 3   Rape      50 non-null     float64
dtypes: float64(2), int64(2)
memory usage: 1.7 KB
```

In [6]:
```
# Normalize heterogenous numerical data
crime2_norm = pd.DataFrame(normalize(crime2),columns=crime2.columns)
crime2_norm
```

Out[6]:

	Murder	Assault	UrbanPop	Rape
0	0.054031	0.966016	0.237411	0.086778
1	0.036872	0.969739	0.176987	0.164081
2	0.026439	0.959624	0.261122	0.101185
3	0.044528	0.961392	0.252998	0.098669
4	0.030657	0.940134	0.309972	0.138295
5	0.035594	0.919142	0.351437	0.174367
6	0.024486	0.816202	0.571341	0.082362
7	0.023674	0.954965	0.288897	0.063397
8	0.044478	0.967547	0.231056	0.092134
9	0.078534	0.952332	0.270805	0.116446
10	0.054546	0.473419	0.854213	0.207893
11	0.019640	0.906483	0.407917	0.107267
12	0.039428	0.944007	0.314669	0.090989
13	0.054447	0.854521	0.491539	0.158805
14	0.027251	0.693660	0.706047	0.139971
15	0.044795	0.858568	0.492743	0.134385
16	0.079346	0.891624	0.425362	0.133335
17	0.059457	0.961347	0.254815	0.085710
18	0.021483	0.849097	0.521734	0.079795
19	0.036587	0.971339	0.216932	0.090011
20	0.025527	0.864425	0.493128	0.094565
21	0.045132	0.951126	0.276013	0.130920
22	0.027317	0.728452	0.667747	0.150749
23	0.061041	0.981958	0.166819	0.064832
24	0.046500	0.919676	0.361670	0.145701
25	0.048998	0.890131	0.432816	0.133928
26	0.035662	0.845935	0.514196	0.136842
27	0.045363	0.937005	0.301180	0.171041
28	0.026088	0.708107	0.695684	0.118018
29	0.040364	0.867284	0.485461	0.102547
30	0.038586	0.964660	0.236934	0.108651
31	0.041163	0.941927	0.318920	0.096789
32	0.038166	0.989371	0.132112	0.047267

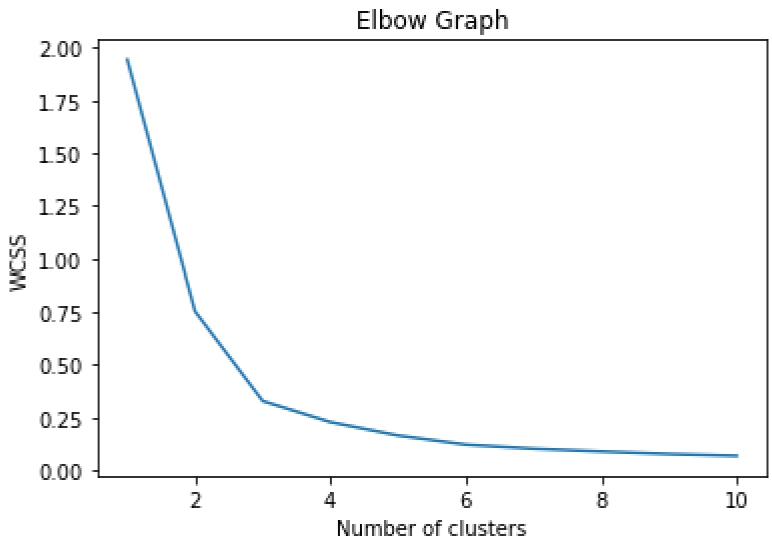
	Murder	Assault	UrbanPop	Rape
33	0.012626	0.710188	0.694406	0.115208
34	0.050940	0.837376	0.523360	0.149332
35	0.039535	0.904523	0.407335	0.119804
36	0.027987	0.908164	0.382685	0.167353
37	0.048778	0.820702	0.557458	0.115363
38	0.017459	0.893478	0.446739	0.042620
39	0.050641	0.981163	0.168802	0.079126
40	0.038785	0.877767	0.459297	0.130644
41	0.066230	0.943274	0.296027	0.134968
42	0.058203	0.921161	0.366631	0.116864
43	0.021908	0.821558	0.547706	0.156781
44	0.037410	0.816227	0.544152	0.190453
45	0.050082	0.919147	0.371194	0.121964
46	0.024318	0.881521	0.443800	0.159282
47	0.062942	0.894442	0.430657	0.102695
48	0.030455	0.620812	0.773086	0.126505
49	0.039384	0.932482	0.347509	0.090352

In [7]:
```
# Use Elbow Graph to find optimum number of  clusters (K value) from K values rar
# The K-means algorithm aims to choose centroids that minimise the inertia, or wi
# random state can be anything from 0 to 42, but the same number to be used every
```

In [19]:
```
# within-cluster sum-of-squares criterion
wcss=[]
for i in range (1,11):
    kmeans=KMeans(n_clusters=i,random_state=2)
    kmeans.fit(crime2_norm)
    wcss.append(kmeans.inertia_)
```

```
C:\Users\Lenovo\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:881: Use
rWarning: KMeans is known to have a memory leak on Windows with MKL, when there
are less chunks than available threads. You can avoid it by setting the environ
ment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

In [9]:
```python
# Plot K values range vs WCSS to get Elbow graph for choosing K (no. of clusters)
plt.plot(range(1,11),wcss)
plt.title('Elbow Graph')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



In [10]:
```python
# selecting 4 clusters from above scree plot
model=KMeans(n_clusters=4)
model.fit(crime2_norm)
model.labels_
```

Out[10]:
```
array([1, 1, 1, 1, 1, 3, 0, 1, 1, 1, 2, 3, 1, 0, 2, 0, 3, 1, 0, 1, 0, 1,
       2, 1, 3, 3, 0, 1, 2, 0, 1, 1, 1, 2, 0, 3, 3, 0, 3, 1, 3, 1, 3, 0,
       0, 3, 3, 3, 2, 3])
```

In [12]:
```
x = pd.Series(model.labels_)
crime['Clust']= x
crime
```

Out[12]:

	Unnamed: 0	Murder	Assault	UrbanPop	Rape	Clust
0	Alabama	13.2	236	58	21.2	1
1	Alaska	10.0	263	48	44.5	1
2	Arizona	8.1	294	80	31.0	1
3	Arkansas	8.8	190	50	19.5	1
4	California	9.0	276	91	40.6	1
5	Colorado	7.9	204	78	38.7	3
6	Connecticut	3.3	110	77	11.1	0
7	Delaware	5.9	238	72	15.8	1
8	Florida	15.4	335	80	31.9	1
9	Georgia	17.4	211	60	25.8	1
10	Hawaii	5.3	46	83	20.2	2
11	Idaho	2.6	120	54	14.2	3
12	Illinois	10.4	249	83	24.0	1
13	Indiana	7.2	113	65	21.0	0
14	Iowa	2.2	56	57	11.3	2
15	Kansas	6.0	115	66	18.0	0
16	Kentucky	9.7	109	52	16.3	3
17	Louisiana	15.4	249	66	22.2	1
18	Maine	2.1	83	51	7.8	0
19	Maryland	11.3	300	67	27.8	1
20	Massachusetts	4.4	149	85	16.3	0
21	Michigan	12.1	255	74	35.1	1
22	Minnesota	2.7	72	66	14.9	2
23	Mississippi	16.1	259	44	17.1	1
24	Missouri	9.0	178	70	28.2	3
25	Montana	6.0	109	53	16.4	3
26	Nebraska	4.3	102	62	16.5	0
27	Nevada	12.2	252	81	46.0	1
28	New Hampshire	2.1	57	56	9.5	2
29	New Jersey	7.4	159	89	18.8	0
30	New Mexico	11.4	285	70	32.1	1
31	New York	11.1	254	86	26.1	1
32	North Carolina	13.0	337	45	16.1	1

	Unnamed: 0	Murder	Assault	UrbanPop	Rape	Clust
33	North Dakota	0.8	45	44	7.3	2
34	Ohio	7.3	120	75	21.4	0
35	Oklahoma	6.6	151	68	20.0	3
36	Oregon	4.9	159	67	29.3	3
37	Pennsylvania	6.3	106	72	14.9	0
38	Rhode Island	3.4	174	87	8.3	3
39	South Carolina	14.4	279	48	22.5	1
40	South Dakota	3.8	86	45	12.8	3
41	Tennessee	13.2	188	59	26.9	1
42	Texas	12.7	201	80	25.5	3
43	Utah	3.2	120	80	22.9	0
44	Vermont	2.2	48	32	11.2	0
45	Virginia	8.5	156	63	20.7	3
46	Washington	4.0	145	73	26.2	3
47	West Virginia	5.7	81	39	9.3	3
48	Wisconsin	2.6	53	66	10.8	2
49	Wyoming	6.8	161	60	15.6	3

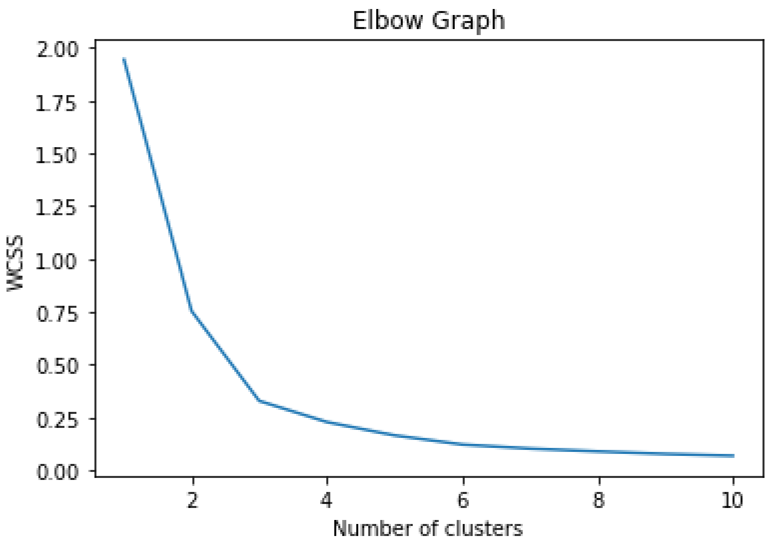
In [14]:

```
crime.iloc[:,1:5].groupby(crime.Clust).mean()
```

Out[14]:

Clust	Murder	Assault	UrbanPop	Rape
0	4.881818	111.363636	68.545455	16.354545
1	12.021053	260.526316	66.421053	27.694737
2	2.616667	54.833333	62.000000	12.333333
3	6.542857	145.285714	63.500000	20.107143

In [15]:
```
# Plot K values range vs WCSS to get Elbow graph for choosing K (no. of clusters)
plt.plot(range(1,11),wcss)
plt.title('Elbow Graph')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



In [ ]: