In [17]:
```python
import scipy.cluster.hierarchy as sch
from sklearn.cluster import AgglomerativeClustering
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sn
import sklearn.cluster as cluster
from sklearn.cluster import KMeans
from scipy.spatial.distance import cdist
from sklearn.cluster import DBSCAN
from sklearn.preprocessing import StandardScaler
import seaborn as sns
from sklearn import metrics
```

In [18]:
```python
df= pd.read_csv('/Users/SAURABH/Saurabh patil/DATA SCIENCE/Clustering/crime_data.csv')
```

In [19]:
```python
df.head()
```

Out[19]:

	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

In [20]:
```python
df.columns
```

Out[20]:
```
Index(['Unnamed: 0', 'Murder', 'Assault', 'UrbanPop', 'Rape'], dtype='object')
```

In [21]:
```python
df1= df.rename({'Unnamed: 0':'City'},axis=1)
```

In [22]:
```python
df1.columns
```

Out[22]:
```
Index(['City', 'Murder', 'Assault', 'UrbanPop', 'Rape'], dtype='object')
```

In [23]:
```python
df2 = df1.iloc[:,1:]
```

In [24]:
```python
df2.head()
```

Out[24]:

	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

In [25]:
```python
df2.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: 1.6 KB
```

In [26]:
```python
# Normalization function
def norm_func(i):
    x = (i-i.min())/(i.max()-i.min())
    return (x)
```

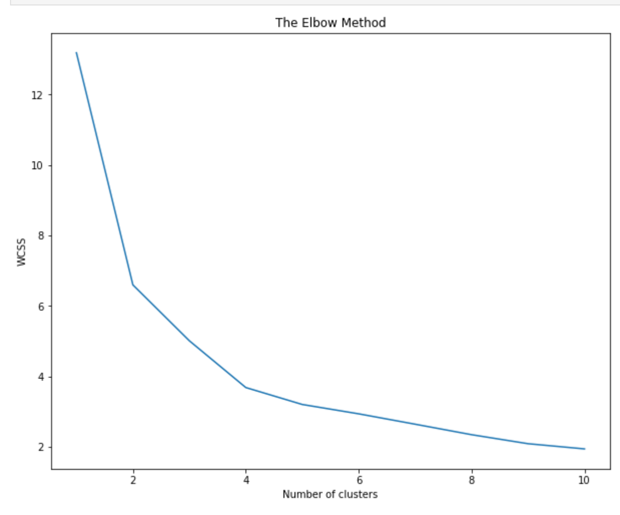
In [27]:
```python
# Normalized data frame (considering the numerical part of data)
X = norm_func(df2.iloc[:,:])
```

K-Means Clustering

Checking via Elbow method for optimum number of clusters

In [28]:
```python
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
%matplotlib inline

plt.figure(figsize=(10, 8))
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_) #criterion based on which K-means clustering works
plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



Confirming the same with Silhouette score

In [29]:
```python
for i in range(3,13):
    labels=cluster.KMeans(n_clusters=i,init="k-means++",random_state=200).fit(X).labels_
    print ("Silhouette score for k(clusters) = "+str(i)+" is "
           +str(metrics.silhouette_score(X,labels,metric="euclidean",sample_size=1000,random_state=200)))
```

```
Silhouette score for k(clusters) = 3 is 0.32393600472652184
Silhouette score for k(clusters) = 4 is 0.3408903560074363
Silhouette score for k(clusters) = 5 is 0.31047828422452595
Silhouette score for k(clusters) = 6 is 0.2707459119571354
Silhouette score for k(clusters) = 7 is 0.2702203242750498
Silhouette score for k(clusters) = 8 is 0.2771980367413108
Silhouette score for k(clusters) = 9 is 0.2927119887866063
Silhouette score for k(clusters) = 10 is 0.23905346059641 03
Silhouette score for k(clusters) = 11 is 0.24600415117232102
Silhouette score for k(clusters) = 12 is 0.24420922668233289
```

Hence, we can conclude that optimum no of clusters = 4

In [30]:
```python
model=KMeans(n_clusters=4)
model.fit(X)
model.labels_
```

Out[30]:
```
array([2, 1, 1, 2, 1, 1, 0, 0, 1, 2, 0, 3, 1, 0, 3, 0, 3, 2, 3, 1, 0, 1,
       3, 2, 0, 3, 3, 1, 3, 0, 1, 1, 2, 3, 0, 0, 0, 0, 0, 2, 3, 2, 1, 0,
       3, 0, 0, 3, 3, 0])
```

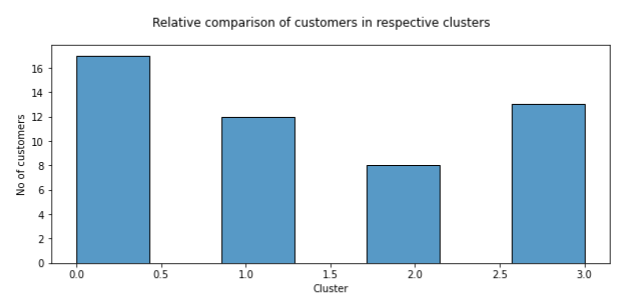
In [31]:
```python
km = pd.Series(model.labels_)
df['kclust']= km
df.iloc[:,1:5].groupby(df.kclust).mean()
```

Out[31]:

	Murder	Assault	UrbanPop	Rape
kclust				
0	5.852941	141.176471	73.647059	19.335294
1	10.966667	264.000000	76.500000	33.608333
2	13.937500	243.625000	53.750000	21.412500
3	3.600000	78.538462	52.076923	12.176923

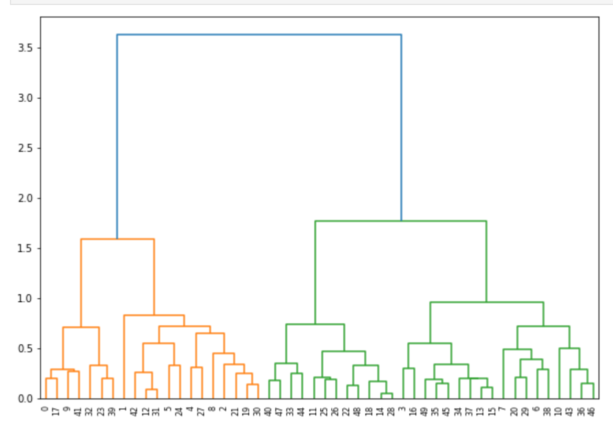
In [32]:
```python
plt.figure(figsize=(10,4))
sns.histplot (x='kclust', data=df)
plt.xlabel('Cluster')
plt.ylabel('No of customers')
plt.suptitle('Relative comparison of customers in respective clusters')
```

Out[32]:
```
Text(0.5, 0.98, 'Relative comparison of customers in respective clusters')
```



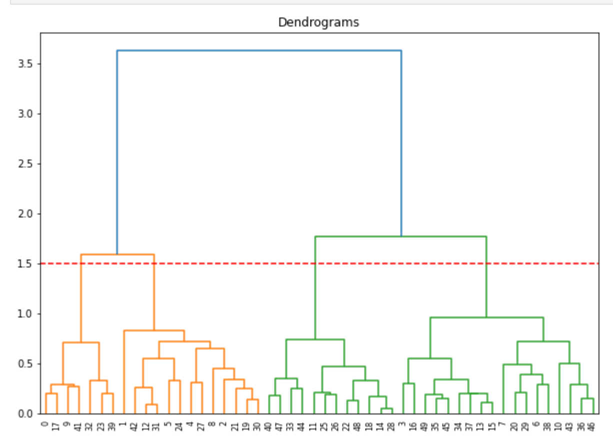
Hierarchical Clustering

In [33]:
```python
# create dendrogram
plt.figure(figsize=(10, 7))
dendrogram = sch.dendrogram(sch.linkage(X, method='ward'))
```



The x-axis contains the samples and y-axis represents the distance between these samples. The vertical line with maximum distance is the blue line. If we decide a threshold of 1.5 and cut the dendrogram:

In [34]:
```python
plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
dend = sch.dendrogram(sch.linkage(X, method='ward'))
plt.axhline(y=1.5, color='r', linestyle='--')
plt.show()
```



In [35]:
```python
# create clusters
hc = AgglomerativeClustering(n_clusters=4, affinity = 'euclidean', linkage = 'single')
```

In [36]:
```python
y_hc = hc.fit_predict(X)
Clusters=pd.DataFrame(y_hc,columns=['Clusters'])
```

In [37]:
```python
Clusters.value_counts()
```

Out[37]:
```
Clusters
0           47
3            1
2            1
1            1
dtype: int64
```

In [38]:
```python
Clusters
```

Out[38]:

	Clusters
0	0
1	3
2	0
3	0
4	0
5	0
6	0
7	1
8	2
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0
29	0
30	0
31	0
32	0
33	0
34	0
35	0
36	0
37	0
38	0
39	0
40	0
41	0
42	0
43	0
44	0
45	0
46	0
47	0
48	0
49	0

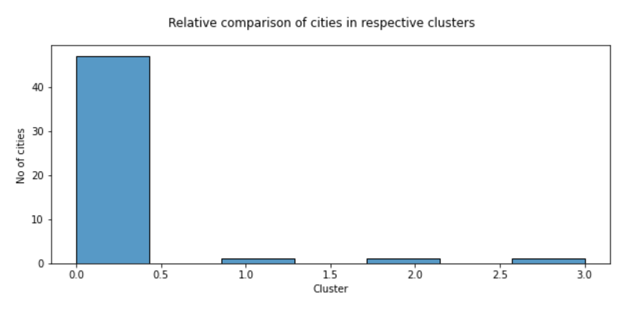
In [39]:
```python
df['hc_clust']= Clusters
df.iloc[:,1:7].groupby(df.hc_clust).mean()
```

Out[39]:

	Murder	Assault	UrbanPop	Rape	kclust	hc_clust
hc_clust						
0	7.619149	163.87234	65.468085	20.625532	1.382979	0
1	5.900000	238.00000	72.000000	15.800000	0.000000	1
2	15.400000	335.00000	80.000000	31.900000	1.000000	2
3	10.000000	263.00000	48.000000	44.500000	1.000000	3

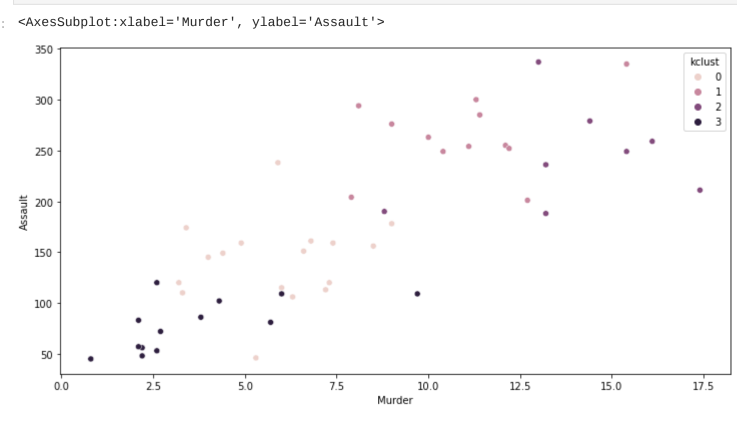
In [40]:
```python
plt.figure(figsize=(10,4))
sns.histplot (x='hc_clust', data=df)
plt.xlabel('Cluster')
plt.ylabel('No of cities')
plt.suptitle('Relative comparison of cities in respective clusters')
```

Out[40]:
```
Text(0.5, 0.98, 'Relative comparison of cities in respective clusters')
```



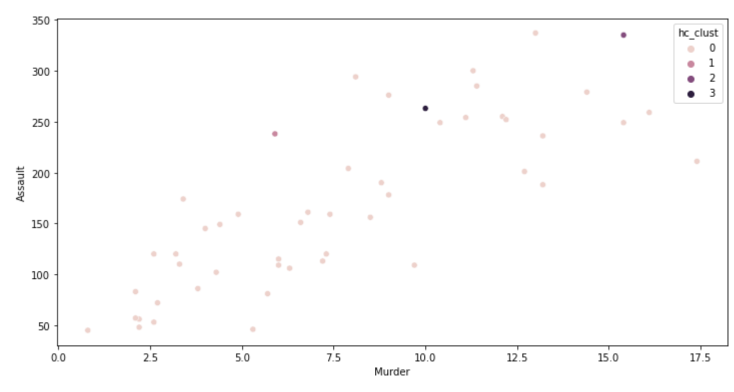
In [41]:
```python
plt.figure(figsize=(12,6))
sns.scatterplot(x=df['Murder'], y =df['Assault'], hue=df['kclust'])
```

Out[41]:
```
<AxesSubplot:xlabel='Murder', ylabel='Assault'>
```



In [42]:
```python
plt.figure(figsize=(12,6))
sns.scatterplot(x=df['Murder'], y =df['Assault'], hue=df['hc_clust'])
```

Out[42]:
```
<AxesSubplot:xlabel='Murder', ylabel='Assault'>
```



In [ ]: