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

import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
data = pd.read_csv('CC_GENERAL.csv')
data
```

Out[2]:

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	ONEOFF_
0	C10001	40.900749	0.818182	95.40	0.00	95.40	0.000000	0.166667	
1	C10002	3202.467416	0.909091	0.00	0.00	0.00	6442.945483	0.000000	
2	C10003	2495.148862	1.000000	773.17	773.17	0.00	0.000000	1.000000	
3	C10004	1666.670542	0.636364	1499.00	1499.00	0.00	205.788017	0.083333	
4	C10005	817.714335	1.000000	16.00	16.00	0.00	0.000000	0.083333	
...	...	...	...	...	...	...	...	...	
8945	C19186	28.493517	1.000000	291.12	0.00	291.12	0.000000	1.000000	
8946	C19187	19.183215	1.000000	300.00	0.00	300.00	0.000000	1.000000	
8947	C19188	23.398673	0.833333	144.40	0.00	144.40	0.000000	0.833333	
8948	C19189	13.457564	0.833333	0.00	0.00	0.00	36.558778	0.000000	
8949	C19190	372.708075	0.666667	1093.25	1093.25	0.00	127.040008	0.666667	

8950 rows × 18 columns

In [3]:
```
data = data.fillna(data.median())
data = data.drop('CUST_ID', axis=1)
data
```

Out[3]:

	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	ONEOFF_PURCHASE
0	40.900749	0.818182	95.40	0.00	95.40	0.000000	0.166667	
1	3202.467416	0.909091	0.00	0.00	0.00	6442.945483	0.000000	
2	2495.148862	1.000000	773.17	773.17	0.00	0.000000	1.000000	
3	1666.670542	0.636364	1499.00	1499.00	0.00	205.788017	0.083333	
4	817.714335	1.000000	16.00	16.00	0.00	0.000000	0.083333	
...	...	...	...	...	...	...	...	
8945	28.493517	1.000000	291.12	0.00	291.12	0.000000	1.000000	
8946	19.183215	1.000000	300.00	0.00	300.00	0.000000	1.000000	
8947	23.398673	0.833333	144.40	0.00	144.40	0.000000	0.833333	
8948	13.457564	0.833333	0.00	0.00	0.00	36.558778	0.000000	
8949	372.708075	0.666667	1093.25	1093.25	0.00	127.040008	0.666667	

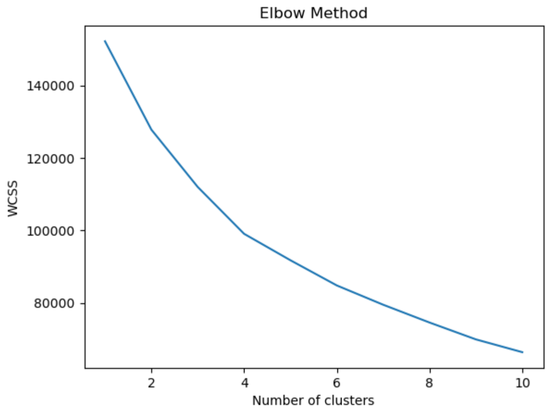
8950 rows × 17 columns

In [4]:
```
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data)
scaled_data
```

Out[4]:
```
array([[-0.73198937, -0.24943448, -0.42489974, ..., -0.3024    ,
        -0.52555097,  0.36067954],
       [ 0.78696085,  0.13432467, -0.46955188, ...,  0.09749953,
         0.2342269 ,  0.36067954],
       [ 0.44713513,  0.51808382, -0.10766823, ..., -0.0932934 ,
        -0.52555097,  0.36067954],
       ...,
       [-0.7403981 , -0.18547673, -0.40196519, ..., -0.32687479,
         0.32919999, -4.12276757],
       [-0.74517423, -0.18547673, -0.46955188, ..., -0.33830497,
         0.32919999, -4.12276757],
       [-0.57257511, -0.88903307,  0.04214581, ..., -0.3243581 ,
        -0.52555097, -4.12276757]])
```

In [5]:
```
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=0)
    kmeans.fit(scaled_data)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



In [6]:
```
kmeans = KMeans(n_clusters=4, init='k-means++', max_iter=300, n_init=10, random_state=0)
clusters = kmeans.fit_predict(scaled_data)
data['Cluster'] = clusters
print(data.groupby('Cluster').mean())
```

```
              BALANCE  BALANCE_FREQUENCY    PURCHASES  ONEOFF_PURCHASES  \
Cluster
0          894.768927           0.934715  1236.263333        593.995933
1         4602.449658           0.968389   501.862982        320.188797
2         3551.153761           0.986879  7681.620098       5095.878826
3         1012.745945           0.789977   270.213240        210.016021

         INSTALLMENTS_PURCHASES  CASH_ADVANCE  PURCHASES_FREQUENCY  \
Cluster
0                    642.541696    209.816318             0.885255
1                    181.759123   4521.509581             0.287832
2                   2587.208264    653.638891             0.946418
3                     60.464108    597.051145             0.170249

         ONEOFF_PURCHASES_FREQUENCY  PURCHASES_INSTALLMENTS_FREQUENCY  \
Cluster
0                          0.297109                          0.711930
1                          0.138911                          0.185671
2                          0.739031                          0.788060
3                          0.086322                          0.080642

         CASH_ADVANCE_FREQUENCY  CASH_ADVANCE_TRX  PURCHASES_TRX  \
Cluster
0                      0.042487          0.789067      22.092692
1                      0.484792         14.294904       7.665831
2                      0.071290          2.085575      89.359413
3                      0.114901          2.125943       2.907240

         CREDIT_LIMIT     PAYMENTS  MINIMUM_PAYMENTS  PRC_FULL_PAYMENT  \
Cluster
0         4214.013741  1331.325429        639.740452          0.269313
1         7546.160857  3484.054216       2003.674460          0.034888
2         9696.943765  7288.739497       1972.767288          0.286707
3         3278.193397   975.085151        553.868540          0.077983

            TENURE
Cluster
0        11.594474
1        11.386800
2        11.951100
3        11.446707
```

In [ ]:
```
```