In [31]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import codecs
from sklearn.cluster import KMeans
import warnings

warnings.filterwarnings('ignore')
```

In [32]:
```
with codecs.open('/home/aiml/Documents/sales_data_sample.csv', 'r', e
    df = pd.read_csv(f)
df.head()
```

Out[32]:

	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDEF
0	10107	30	95.70	2	2871.00	2/2
1	10121	34	81.35	5	2765.90	5/7/200
2	10134	41	94.74	2	3884.34	7/1/200
3	10145	45	83.26	6	3746.70	8/2
4	10159	49	100.00	14	5205.27	10/1

5 rows × 25 columns

In [33]:
```
df.dtypes
```

Out[33]:
```
ORDERNUMBER           int64
QUANTITYORDERED       int64
PRICEEACH           float64
ORDERLINENUMBER       int64
SALES               float64
ORDERDATE            object
STATUS               object
QTR_ID                int64
MONTH_ID              int64
YEAR_ID               int64
PRODUCTLINE          object
MSRP                  int64
PRODUCTCODE          object
CUSTOMERNAME         object
PHONE                object
ADDRESSLINE1         object
ADDRESSLINE2         object
CITY                 object
STATE                object
POSTALCODE           object
COUNTRY              object
TERRITORY            object
CONTACTLASTNAME      object
CONTACTFIRSTNAME     object
DEALSIZE             object
dtype: object
```

In [34]:
```
df.isna().sum()
```

Out[34]:
```
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 [35]:
```
df.shape
```

Out[35]:
```
(2823, 25)
```

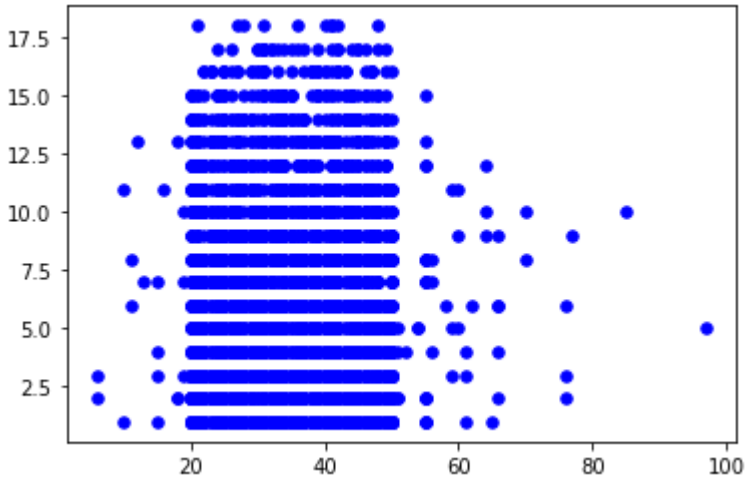
In [36]:
```
X = df[['QUANTITYORDERED', 'ORDERLINENUMBER']]
df = df.dropna()
X
```

Out[36]:

	QUANTITYORDERED	ORDERLINENUMBER
0	30	2
1	34	5
2	41	2
3	45	6
4	49	14
...	...	...
2818	20	15
2819	29	1
2820	43	4
2821	34	1
2822	47	9

2823 rows × 2 columns

In [37]:
```
p="QUANTITYORDERED"
q="ORDERLINENUMBER"
plt.scatter(X[p], X[q], s = 30, c = 'b')
plt.show()
```

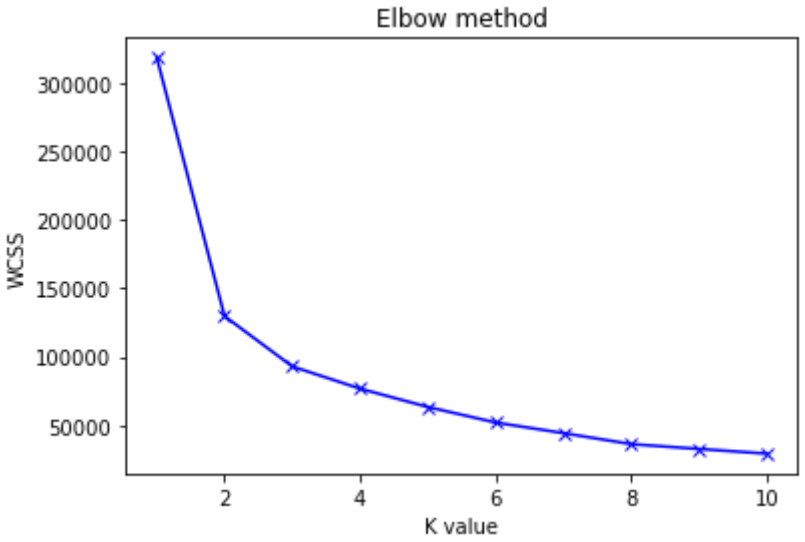


In [38]:
```
wcss = []   #within cluster sum of square

for i in range(1,11):
    kmeans = KMeans(n_clusters=i, init="k-means++", random_state=42)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)

ks = [1,2,3,4,5,6,7,8,9,10]
plt.plot(ks, wcss, 'bx-')
plt.title("Elbow method")
plt.xlabel("K value")
plt.ylabel("WCSS")
```

Out[38]:
```
Text(0, 0.5, 'WCSS')
```



In [39]:
```
from kneed import KneeLocator

k=KneeLocator(ks,wcss,curve="convex",direction="decreasing")
optimal_k=k.elbow
print(f"The optimal number of clusters ={optimal_k}")
```

```
The optimal number of clusters =3
```

In [40]:
```
Kmean = KMeans(n_clusters=optimal_k, init="k-means++", random_state=4
Kmean.fit(X)
```

Out[40]:
```
KMeans(n_clusters=3, random_state=42)
```

In [41]:
```
y_kmeans = Kmean.predict(X)
```

In [42]:
```
#plotting the results:
plt.scatter(X[p], X[q], c=y_kmeans, s=50, cmap='viridis')

centers = Kmean.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5
```