In [1]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:
```python
df=pd.read_csv(r"C:\Users\aditr\Desktop\project report\1. customer segmentation\Mall_Customers.csv")
df.head()
```

Out[5]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

In [9]:
```python
df.shape
```

Out[9]: (200, 5)

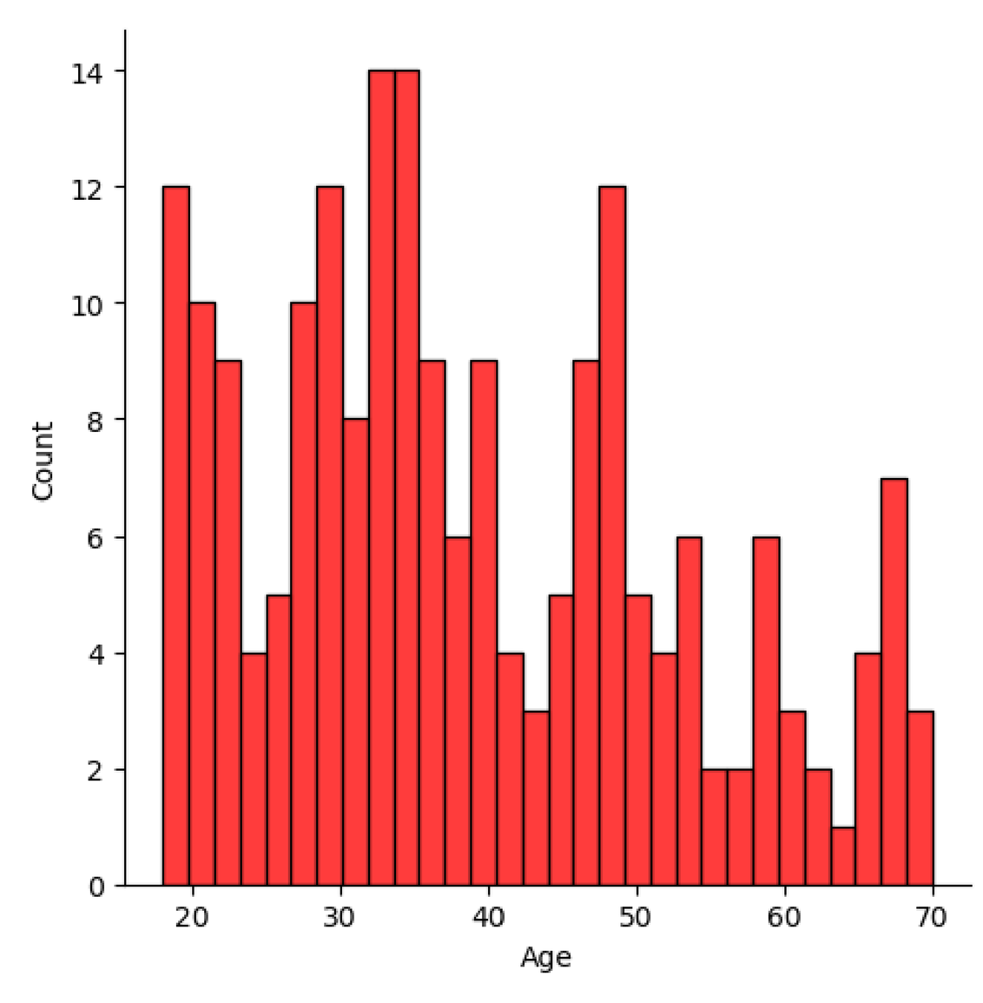
In [11]:
```python
df.describe()
```

Out[11]:

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000
25%	50.750000	28.750000	41.500000	34.750000
50%	100.500000	36.000000	61.500000	50.000000
75%	150.250000	49.000000	78.000000	73.000000
max	200.000000	70.000000	137.000000	99.000000

In [17]:
```python
sns.displot(df['Age'], kde = False, color ='red', bins = 30)
```

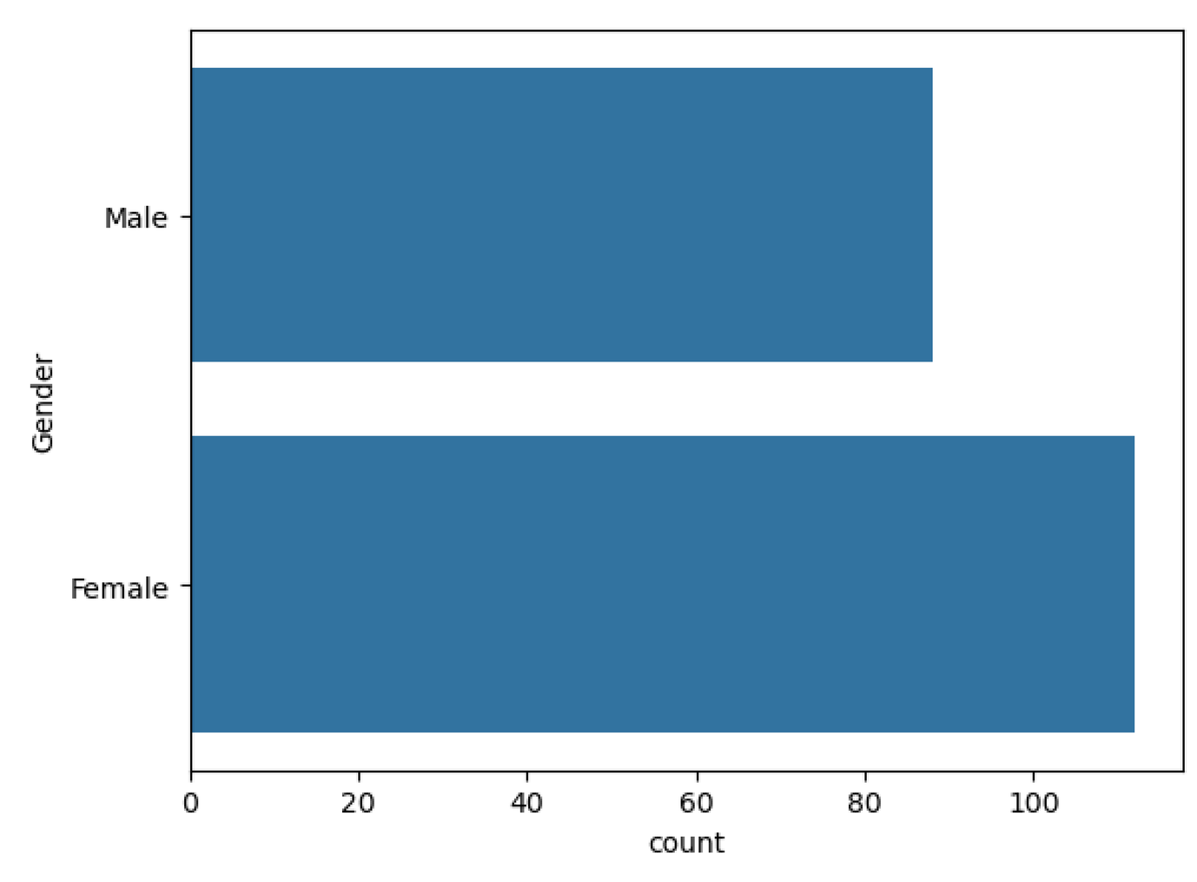
Out[17]: <seaborn.axisgrid.FacetGrid at 0x2a536648b90>



THE AVERAGE CUSTOMER AROUND THE MALL ARE OF GENERALLY AGE 31-34

In [22]:
```python
sns.countplot(y ='Gender', data = df)
```

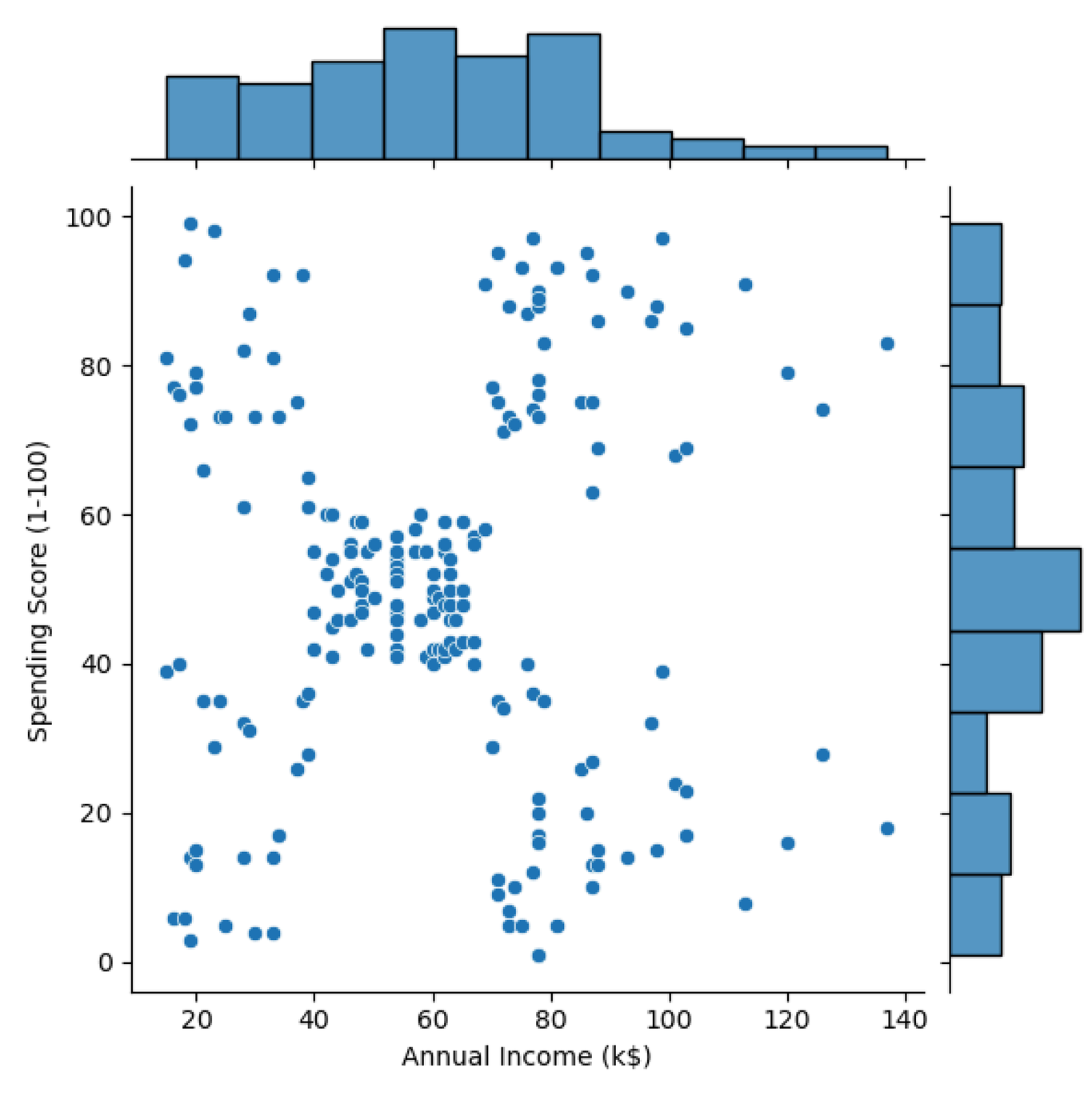
Out[22]: <Axes: xlabel='count', ylabel='Gender'>



THIS MALL HA GENERALLY MORE FEMALE CUSTOMER THEN MALE

In [31]:
```python
sns.jointplot(x ='Annual Income (k$)', y ='Spending Score (1-100)', data = df)
```

Out[31]: <seaborn.axisgrid.JointGrid at 0x2a53828d310>



Question : Why choosing K-Clustering over Hierarchical Clustering and Density based Clustering ?

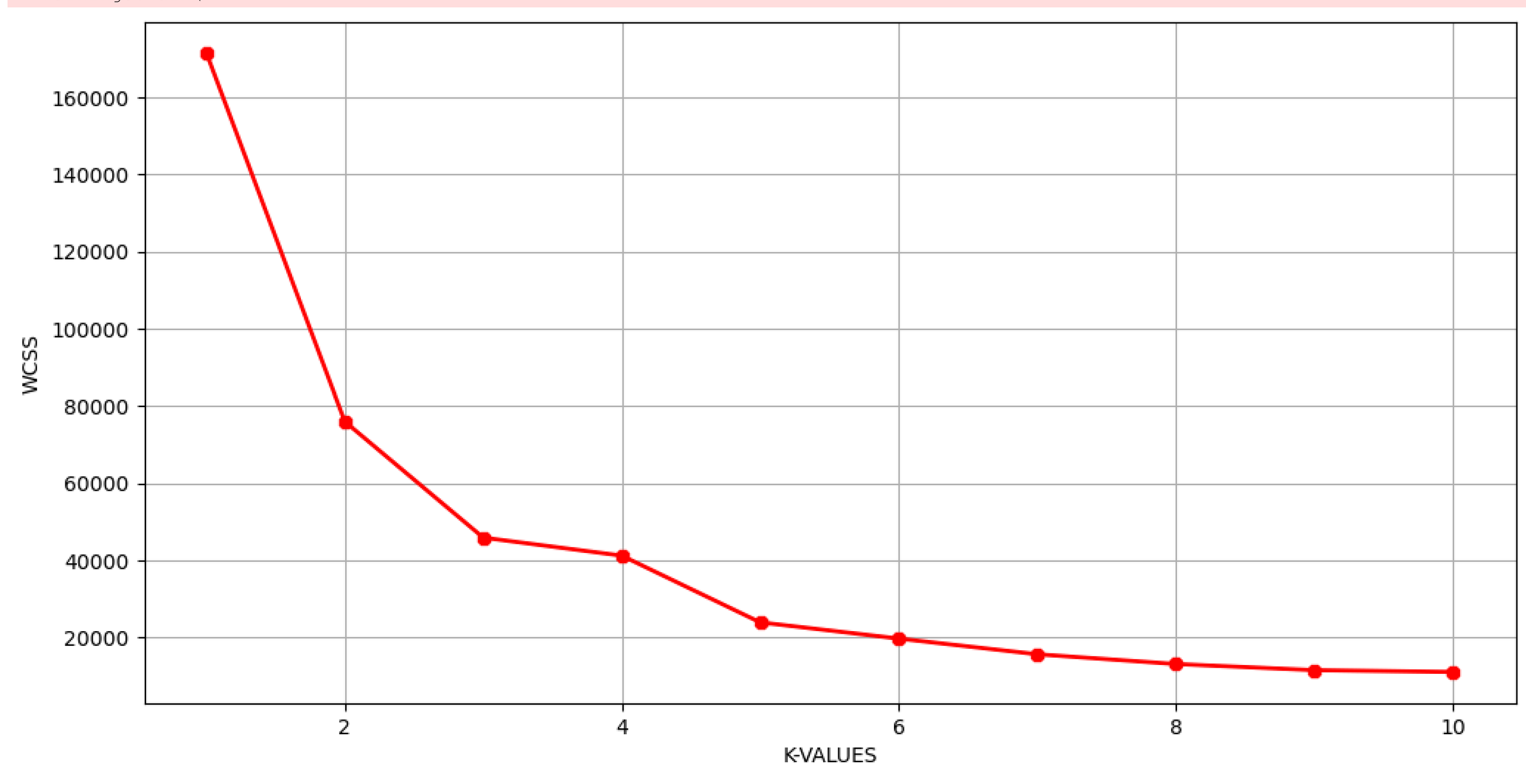
Answer : On these points notes: K-Means is relatively efficient and simpler to implement & visualise and has only one key parameter it depends upon that is the number of clusters(k). K-Means can easily work with large datasets and scales efficiently. It is commonly used for datasets with thousands or even millions of row and column leading to different outcomes.

In [44]:
```python
from sklearn.datasets import make_blobs
```

In [54]:
```python
import os
os.environ["OMP_NUM_THREADS"] = "1"
```

In [68]:
```python
k1=df.loc[:,["Age","Spending Score (1-100)"]].values
from sklearn.cluster import KMeans
wcss = []
for k in range(1,11):
    kmeans = KMeans(n_clusters= k, init="k-means++")
    kmeans.fit(k1)
    wcss.append(kmeans.inertia_)
plt.figure(figsize=(12,6))
plt.plot(range(1,11),wcss, linewidth=2 ,color="red",marker = "8")
plt.xlabel('K-VALUES')
plt.ylabel('WCSS')
plt.grid()
plt.show()
```

```
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
```



assuming cluster = 5, as after which the data seems to go constant

In [73]:
```python
kmeans =KMeans(n_clusters=4)
label = kmeans.fit_predict(k1)
print(label)
```

```
[1 2 3 2 1 2 3 2 3 2 3 2 3 2 3 2 1 1 3 2 1 2 3 2 3 2 3 1 3 2 3 2 3 2 3 2 3
 2 3 2 0 2 0 1 3 1 0 1 1 1 0 1 1 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0 1 1 0 0 0 0
 0 1 0 1 1 0 0 1 0 0 1 0 0 1 1 0 0 1 0 1 1 1 0 1 0 1 1 0 0 1 0 1 0 0 0 0 0
 1 1 1 1 0 0 0 0 1 1 1 2 1 2 0 2 3 2 3 2 1 2 3 2 3 2 3 2 3 2 3 2 1 2 3 2 0 2
 3 2 3 2 3 2 3 2 3 2 3 2 0 2 3 2 3 2 3 2 3 1 3 2 3 2 3 2 3 2 3 2 3 2 3 2 1
 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2]
```

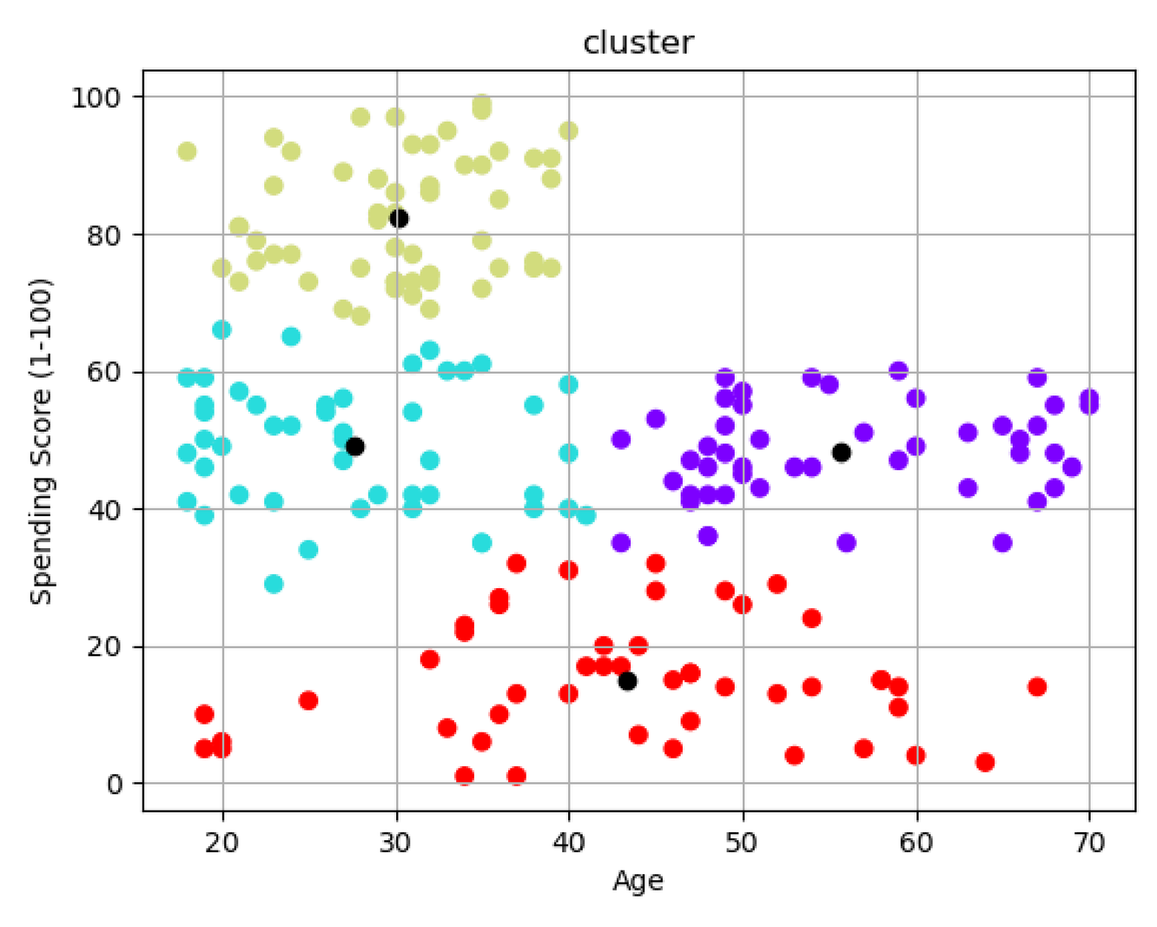
```
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

centroids

In [82]:
```python
print(kmeans.cluster_centers_)
```

```
[[55.70833333 48.22916667]
 [27.61702128 49.14893617]
 [30.1754386  82.35087719]
 [43.29166667 15.02083333]]
```

In [88]:
```python
plt.scatter(k1[:,0],k1[:,1], c=kmeans.labels_,cmap='rainbow')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],color='black')
plt.title('cluster')
plt.xlabel('Age')
plt.ylabel('Spending Score (1-100)')
plt.grid()
plt.show()
```

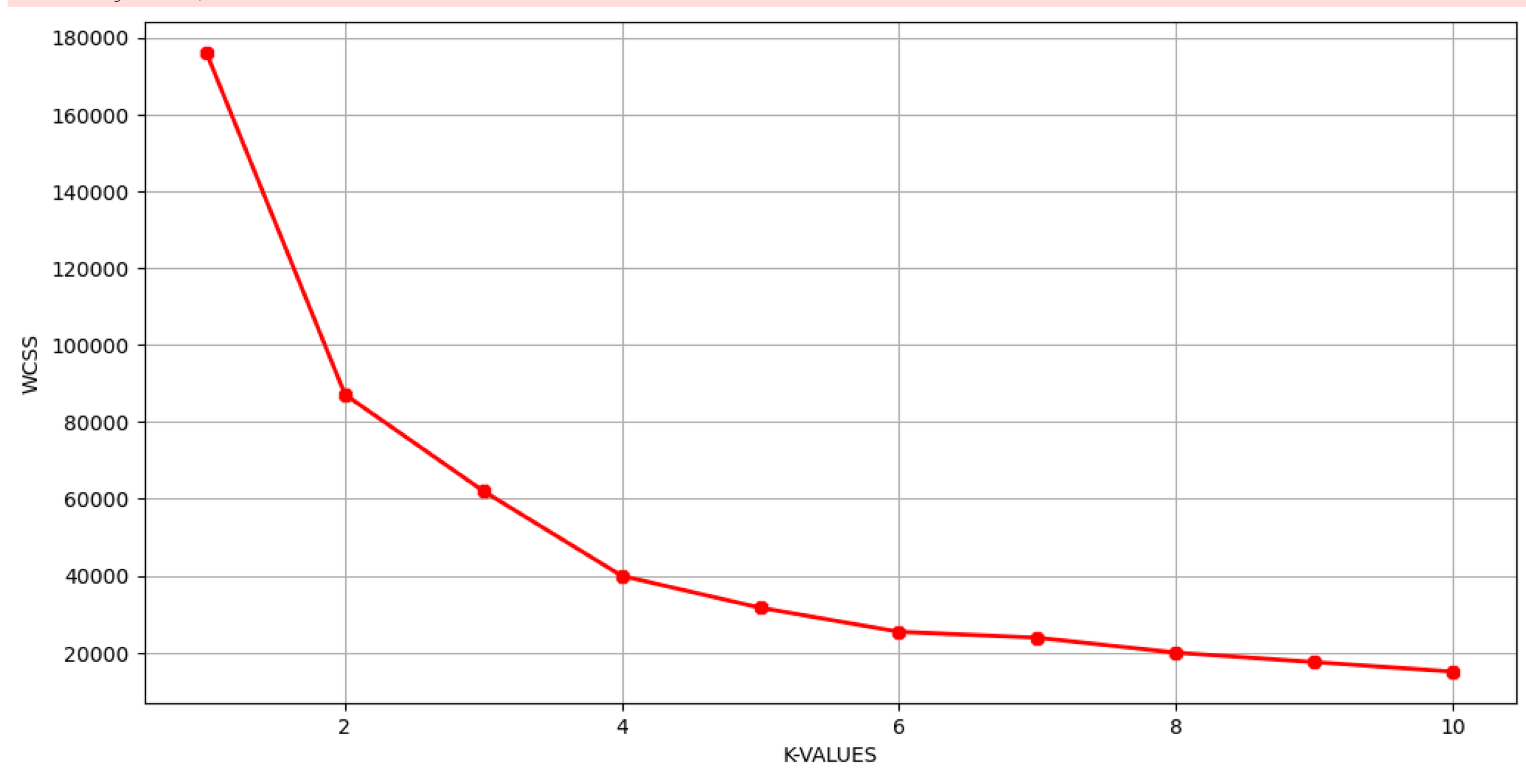


black dots shows the centroids

SIMILARLY THE OTHER CLUSTERS

In [92]:
```python
k2=df.loc[:,["Age","Annual Income (k$)"]].values
from sklearn.cluster import KMeans
wcss = []
for k in range(1,11):
    kmeans = KMeans(n_clusters= k, init="k-means++")
    kmeans.fit(k2)
    wcss.append(kmeans.inertia_)
plt.figure(figsize=(12,6))
plt.plot(range(1,11),wcss, linewidth=2 ,color="red",marker = "8")
plt.xlabel('K-VALUES')
plt.ylabel('WCSS')
plt.grid()
plt.show()
```

```
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

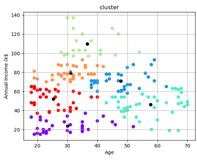


In [94]:
```python
kmeans =KMeans(n_clusters=6)
label = kmeans.fit_predict(k2)
print(label)
```

```
[0 0 0 0 0 0 0 0 2 0 2 0 2 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 2 0 2 0 2 0 0
 0 0 5 2 5 2 5 2 5 2 5 5 5 2 5 5 2 2 2 2 2 5 2 2 5 2 2 2 5 2 2 5 5 2 2 2 2
 2 5 2 5 5 2 2 5 2 2 5 2 2 5 5 1 2 5 1 1 5 5 1 5 1 5 5 1 2 5 1 5 2 1 2 2 2
 5 1 5 5 5 1 1 1 1 4 1 1 1 4 4 1 1 1 1 1 1 4 4 4 4 1 4 4 4 1 4 4 4 4 4 1 4
 4 4 1 4 1 4 1 4 4 4 4 4 1 4 4 4 1 4 4 4 4 4 4 4 4 4 1 4 1 4 1 4 3 3 3 3 3
 3 3 3 3 3 3 3 3 3 3 3 3 3 3]
```

```
C:\Users\aditr\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446: UserWarning: 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 e
nvironment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

In [96]:
```python
plt.scatter(k2[:,0],k2[:,1], c=kmeans.labels_,cmap='rainbow')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],color='black')
plt.title('cluster')
plt.xlabel('Age')
plt.ylabel('Annual Income (k$')
plt.grid()
plt.show()
```


cluster
Annual Income (k$
Age
140
120
100
80
60
40
20
20
30
40
50
60
70

In [ ]: