In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df = pd.read_csv("Mall_Customers.csv")
```

In [3]:

```
df.head()
```

Out[3]:

	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 [4]:

```
df.shape
```

Out[4]:

```
(200, 5)
```

In [5]:

```
df.describe()
```

Out[5]:

	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 [6]:

```
df.dtypes
```

Out[6]:

```
CustomerID                 int64
Gender                    object
Age                        int64
Annual Income (k$)         int64
Spending Score (1-100)     int64
dtype: object
```

In [7]:

```
df.isnull().sum()
```

Out[7]:

```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

In [8]:

```
df.drop(["CustomerID"],axis=1,inplace =True)
```

In [9]:

```
df.head()
```

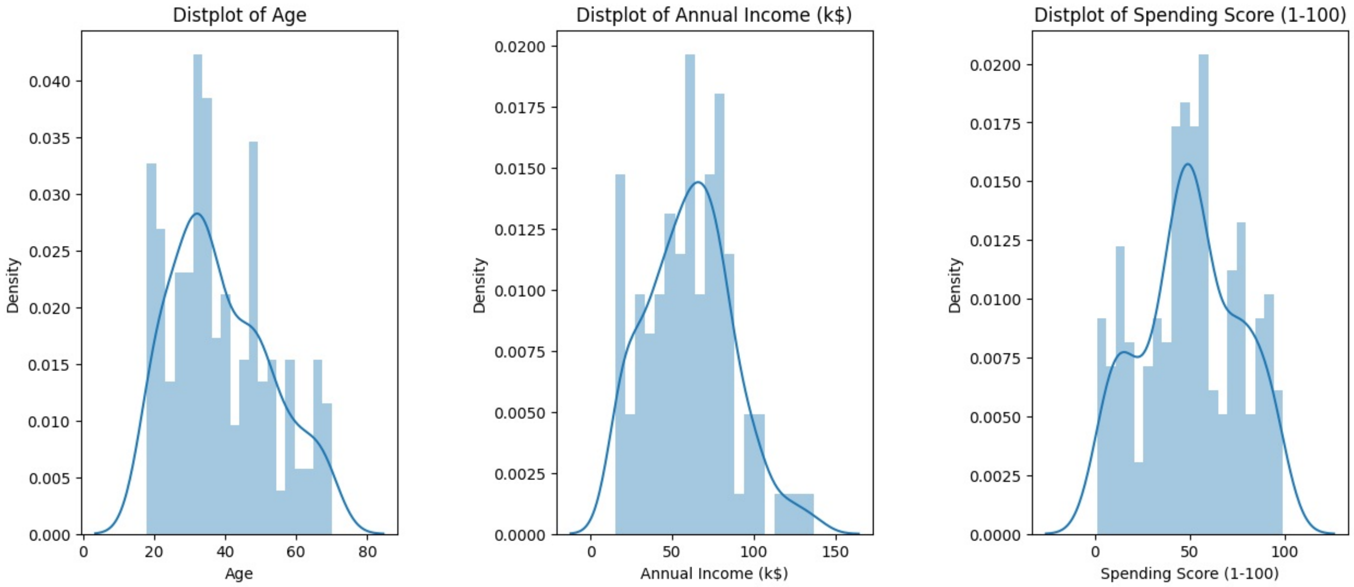
Out[9]:

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

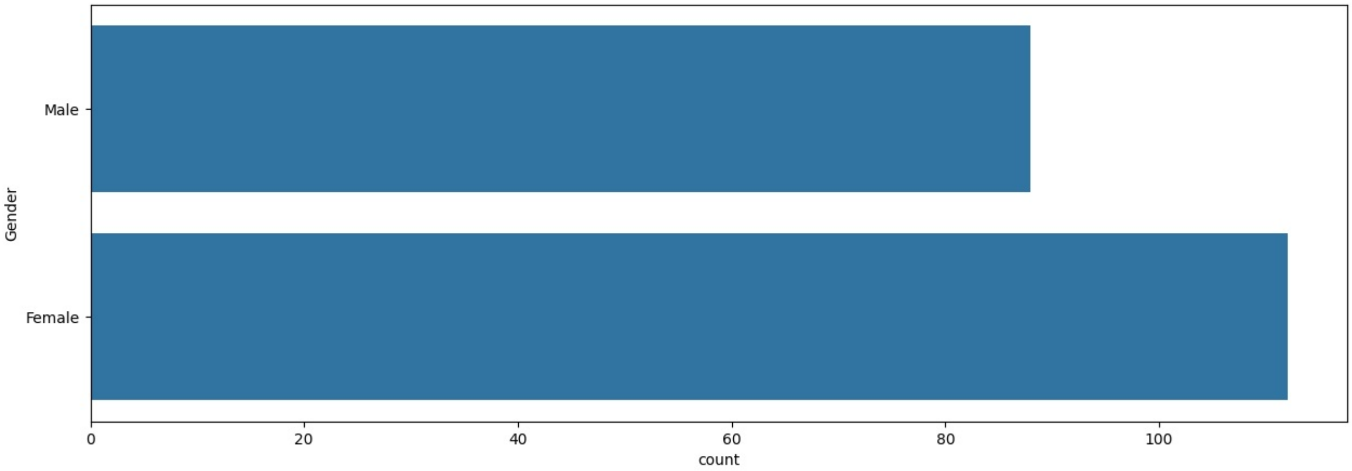
In [10]:

```
plt.figure(1, figsize=(15,6))
n=0

for x in ['Age', 'Annual Income (k$)', 'Spending Score (1-100)']:
    n += 1
    plt.subplot(1, 3, n)

    plt.subplots_adjust(hspace=0.5, wspace = 0.5)

    sns.distplot(df[x], bins = 20)

    plt.title('Distplot of {}'.format(x))

plt.show()
```

```
C:\Users\srinu\AppData\Local\Temp\ipykernel_6904\1035435271.py:11: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df[x], bins = 20)
C:\Users\srinu\AppData\Local\Temp\ipykernel_6904\1035435271.py:11: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df[x], bins = 20)
C:\Users\srinu\AppData\Local\Temp\ipykernel_6904\1035435271.py:11: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df[x], bins = 20)
```

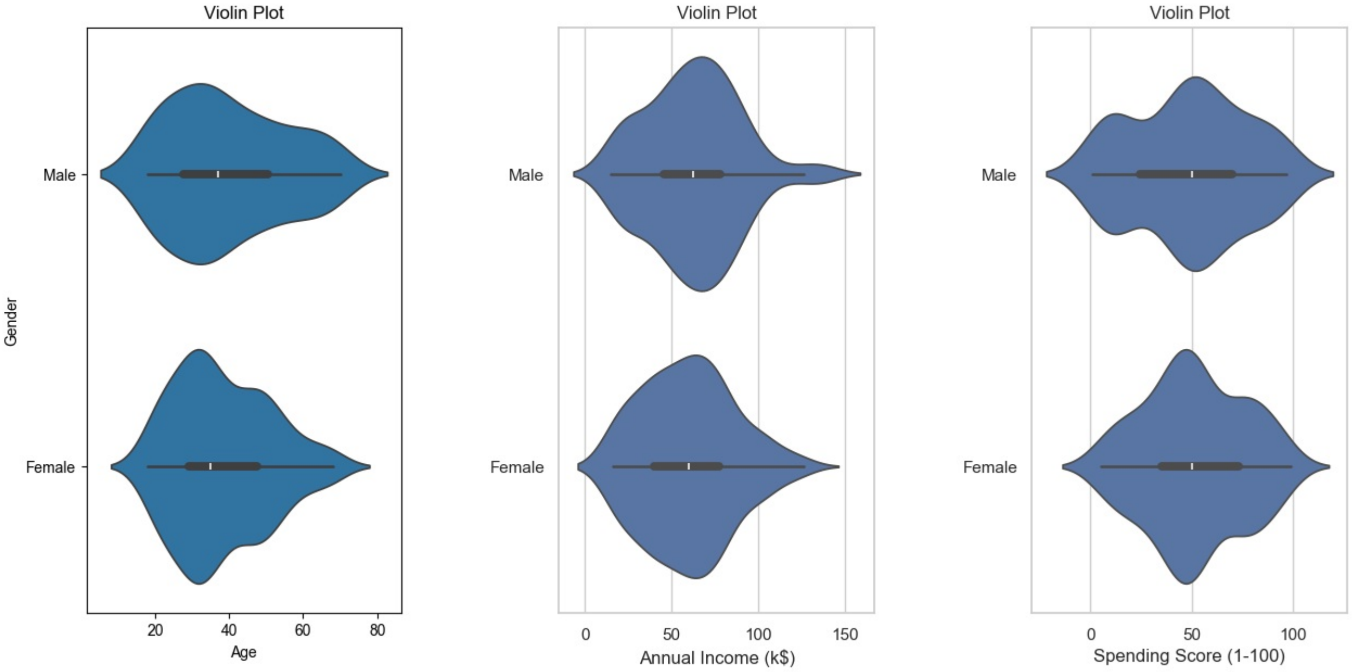


In [11]:
```
plt.figure(figsize=(15,5))
sns.countplot(y = 'Gender', data = df)
plt.show()
```



In [12]:

```
plt.figure(1,figsize=(15,7))
n=0

for cols in ['Age', 'Annual Income (k$)', 'Spending Score (1-100)']:

    n+=1

    plt.subplot(1, 3, n)

    sns.set(style="whitegrid")

    plt.subplots_adjust(hspace=0.5, wspace = 0.5)

    sns.violinplot(x = cols, y = 'Gender', data = df)

    plt.ylabel('Gender' if n == 1 else '')

    plt.title('Violin Plot')

plt.show()
```

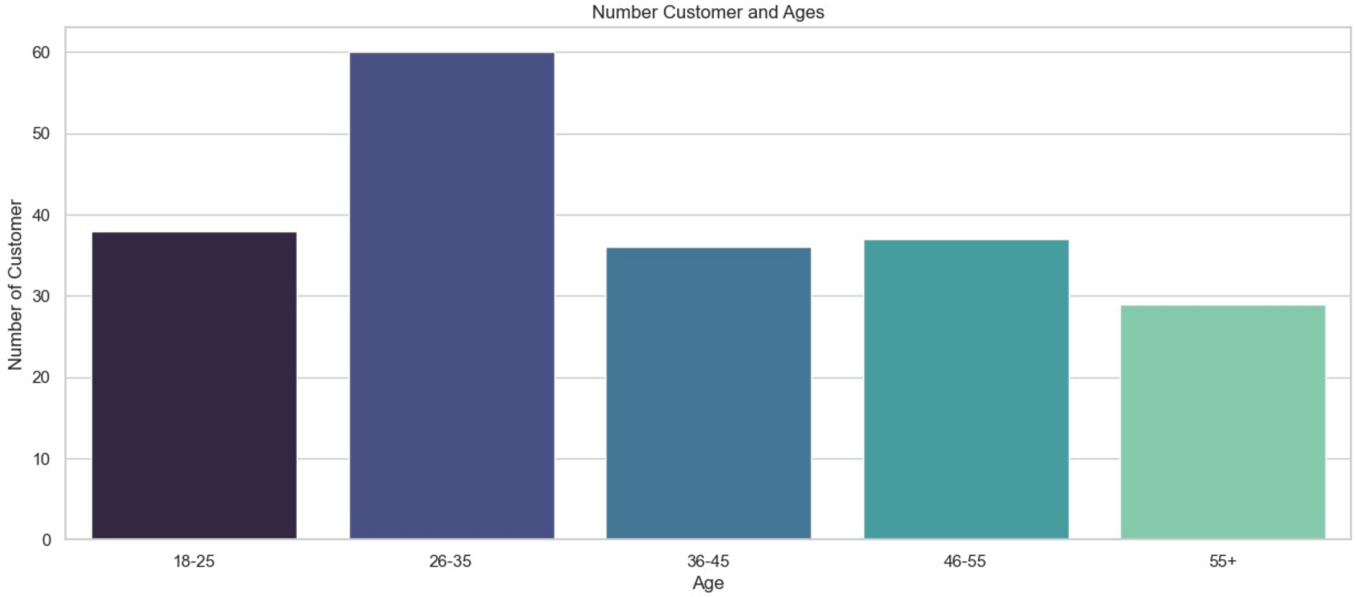


In [13]:

```
age_18_25 = df.Age[(df.Age >= 18 ) & (df.Age <= 25 )]

age_26_35 = df.Age[(df.Age >= 26 ) & (df.Age <= 35 )]

age_36_45 = df.Age[(df.Age >= 36 ) & (df.Age <= 45 )]

age_46_55 = df. Age[(df.Age >= 46 ) & (df.Age <= 55 )]

age_55above = df.Age [df.Age >= 56 ]

agex = ["18-25", "26-35", "36-45","46-55","55+"]

agey = [len(age_18_25.values), len(age_26_35.values), len(age_36_45.values), len(age_46_55.values), len (age_55a

plt.figure(figsize=(15,6))
```

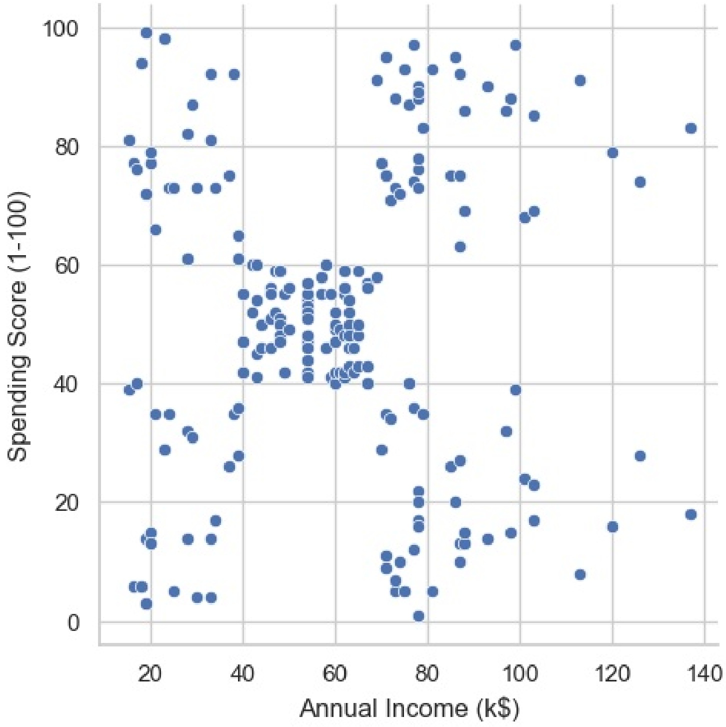
```
sns.barplot(x=agex, y=agey, palette="mako")

plt.title("Number Customer and Ages")

plt.xlabel("Age")

plt.ylabel("Number of Customer")

plt.show()
```

```
C:\Users\srinu\AppData\Local\Temp\ipykernel_6904\323071086.py:17: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=agex, y=agey, palette="mako")
```



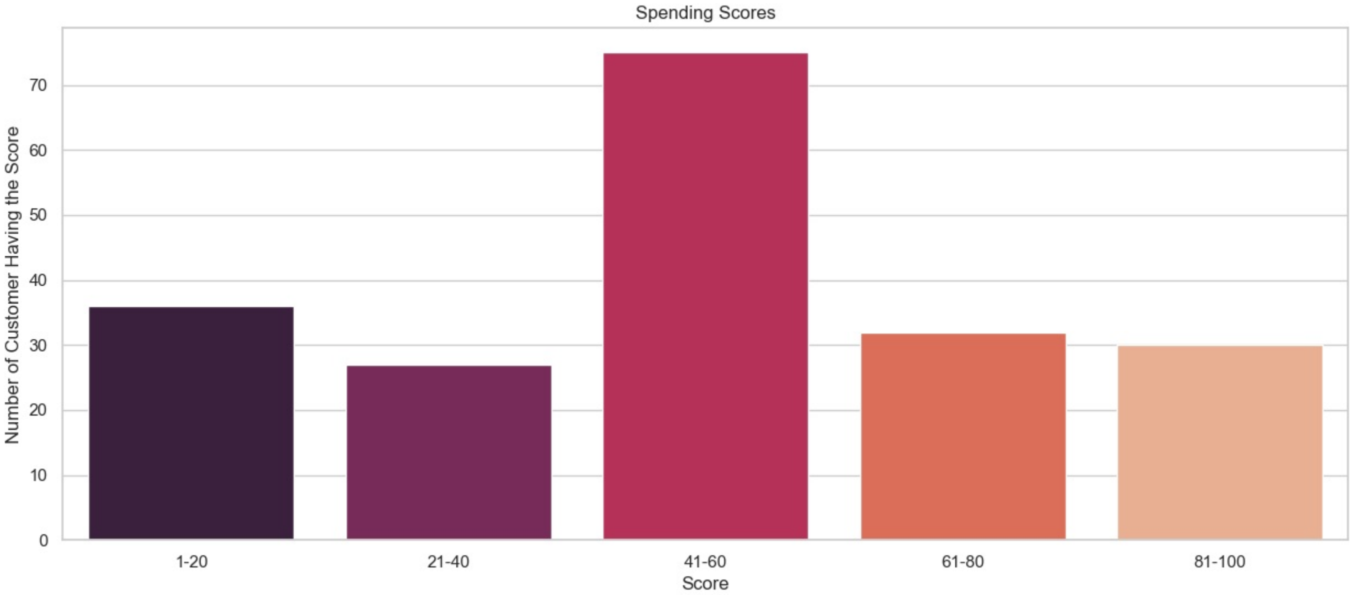
In [14]:
```
sns.relplot(x="Annual Income (k$)" , y = "Spending Score (1-100)" , data= df)
```

Out[14]:
```
<seaborn.axisgrid.FacetGrid at 0x14ef0610830>
```



In [15]:
```
ss_1_20 = df["Spending Score (1-100)"] [(df["Spending Score (1-100)"] >= 1) & (df["Spending Score (1-100)"] <= 2
ss_21_40 = df["Spending Score (1-100)"] [(df["Spending Score (1-100)"] >= 21) & (df["Spending Score (1-100)"] <=
ss_41_60 = df["Spending Score (1-100)"] [(df ["Spending Score (1-100)"] >= 41) & (df["Spending Score (1-100)"] <
ss_61_80 = df["Spending Score (1-100)"] [(df["Spending Score (1-100)"] >= 61) & (df["Spending Score (1-100)"] <=
ss_81_100 = df["Spending Score (1-100)"] [(df["Spending Score (1-100)"] >= 81) & (df["Spending Score (1-100)"] <
ssx = ["1-20", "21-40", "41-60", "61-80", "81-100"]

ssy = [len(ss_1_20.values), len(ss_21_40.values), len(ss_41_60.values), len (ss_61_80.values), len (ss_81_100.va

plt.figure(figsize=(15,6))

sns.barplot(x=ssx, y=ssy, palette="rocket")

plt.title("Spending Scores")

plt.xlabel("Score")

plt.ylabel("Number of Customer Having the Score")

plt.show()
```

```
C:\Users\srinu\AppData\Local\Temp\ipykernel_6904\2011223906.py:17: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=ssx, y=ssy, palette="rocket")
```



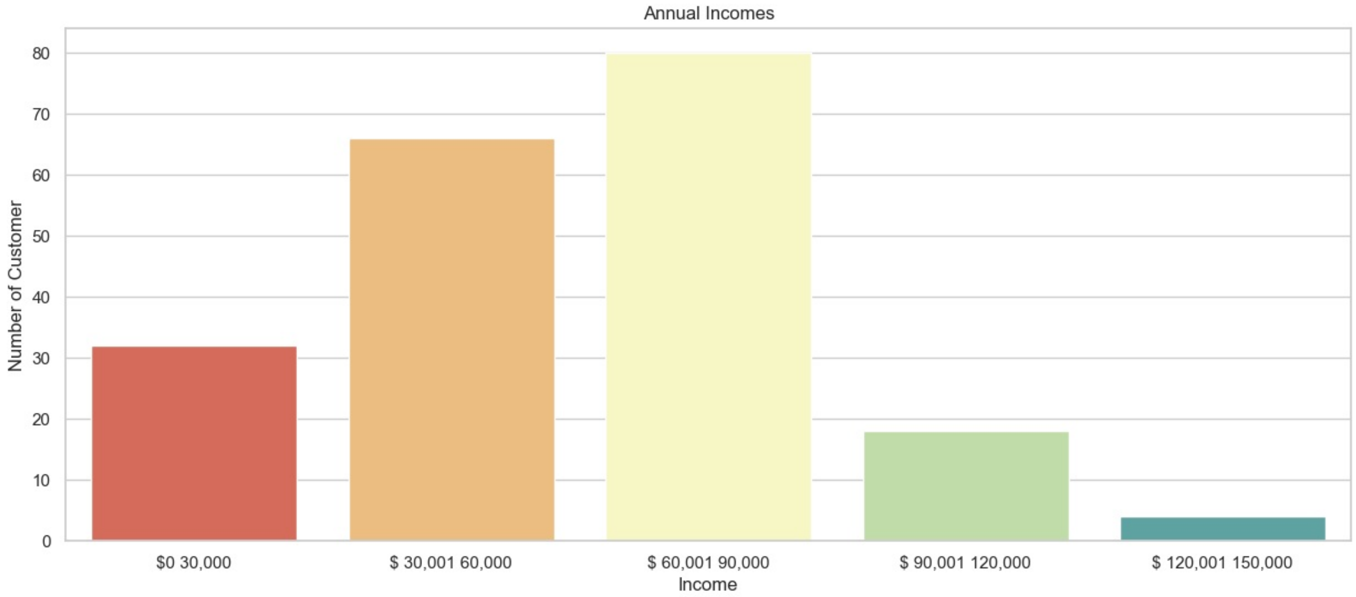
In [16]:

```
ai10_30  = df["Annual Income (k$)"] [(df ["Annual Income (k$)"] >= 0) & (df ["Annual Income (k$)"] <= 30)]

ai31_60 =df["Annual Income (k$)"] [(df["Annual Income (k$)"] >= 31) & (df ["Annual Income (k$)"] <= 60)]

ai61_90  = df["Annual Income (k$)"] [(df["Annual Income (k$)"] >= 61) & (df["Annual Income (k$)"] <= 90)]

ai91_120 =df["Annual Income (k$)"] [(df["Annual Income (k$)"] >= 91) & (df ["Annual Income (k$)"] <= 120)]

ai121_150 =df["Annual Income (k$)"] [(df ["Annual Income (k$)"] >= 121) & (df ["Annual Income (k$)"] <= 150)]

aix = ["$0 30,000", "$ 30,001 60,000", "$ 60,001 90,000", "$ 90,001 120,000", "$ 120,001 150,000"]

aiy = [len(ai10_30.values), len(ai31_60.values), len (ai61_90.values), len(ai91_120.values), len(ai121_150.valu

plt.figure(figsize=(15,6))

sns.barplot(x=aix, y=aiy, palette="Spectral")

plt.title("Annual Incomes")

plt.xlabel("Income")

plt.ylabel("Number of Customer")

plt.show
```

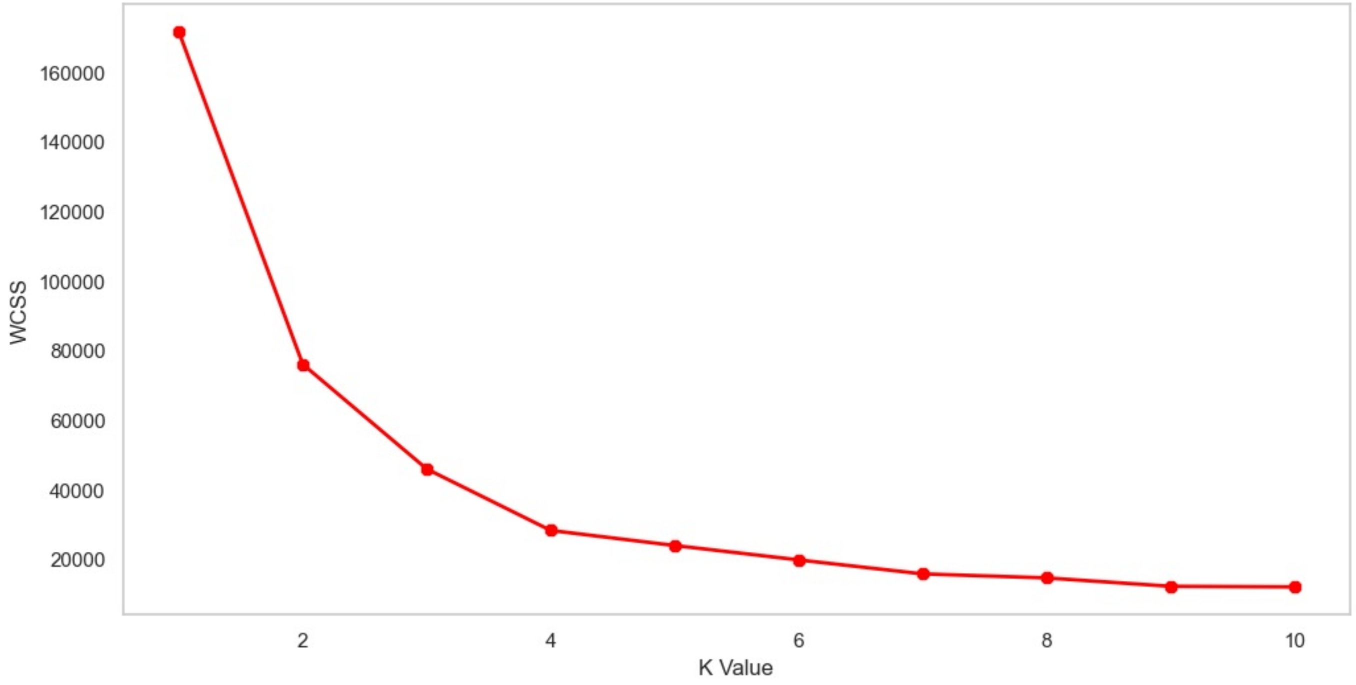
```
C:\Users\srinu\AppData\Local\Temp\ipykernel_6904\4007312437.py:17: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=aix, y=aiy, palette="Spectral")
```

Out[16]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [17]:
```
X1=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(X1)

    wcss.append(kmeans.inertia_)

plt.figure(figsize=(12,6))


plt.grid()
plt.plot(range(1,11), wcss, linewidth=2, color="red", marker ="8")
plt.xlabel("K Value")
plt.ylabel("WCSS")

plt.show()
```



In [18]:
```
kmeans = KMeans(n_clusters = 4)
label = kmeans.fit_predict(X1)
print(label)
```

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

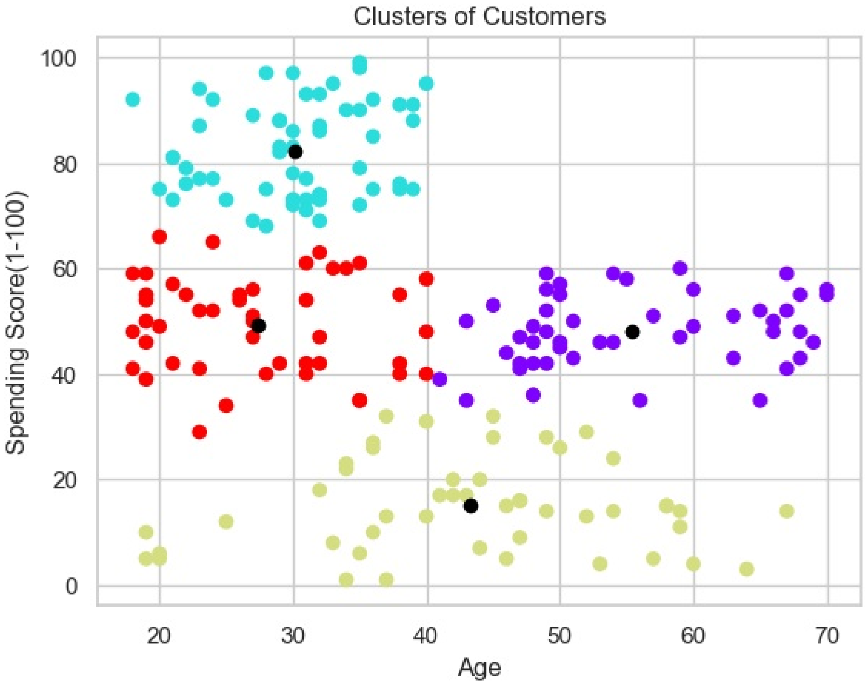
In [19]:

```python
print(kmeans.cluster_centers_)
```

```
[[55.40816327 48.04081633]
 [30.1754386  82.35087719]
 [43.29166667 15.02083333]
 [27.32608696 49.36956522]]
```

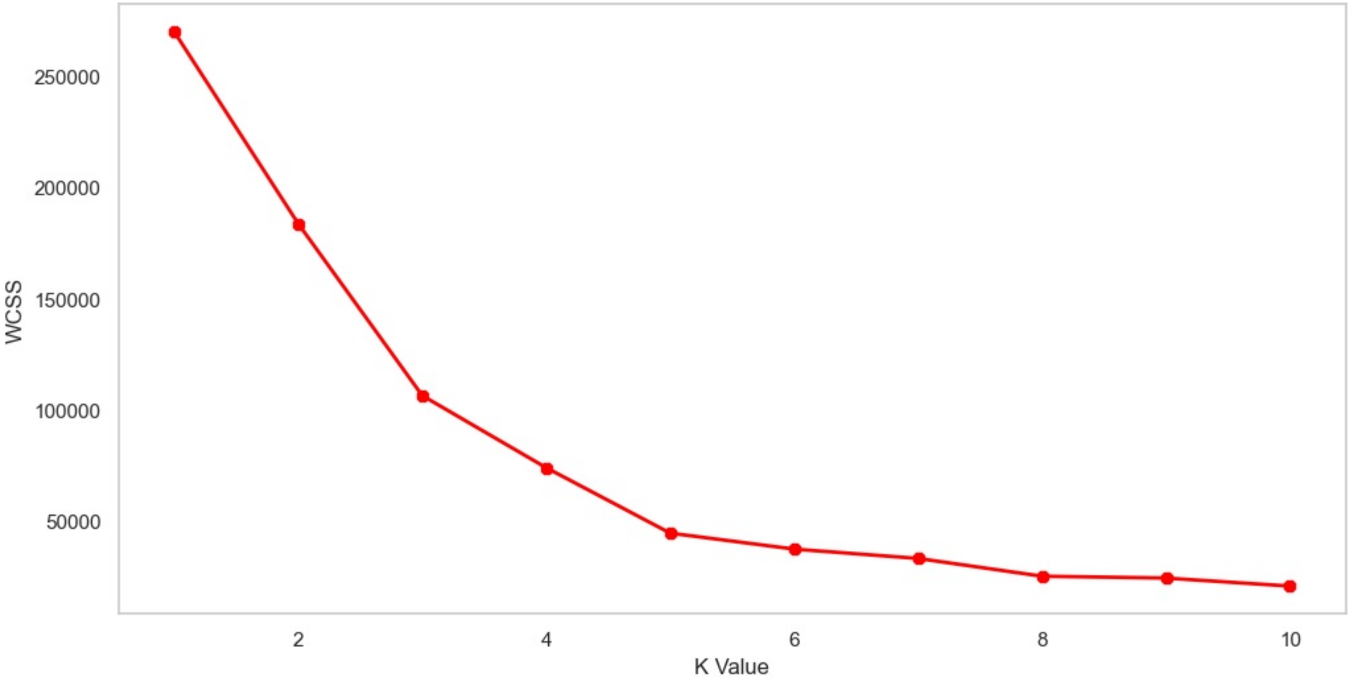
In [20]:

```python
plt.scatter (X1[:,0], X1[:,1], c=kmeans.labels_, cmap='rainbow')

plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1], color='black')

plt.title('Clusters of Customers')

plt.xlabel('Age')

plt.ylabel('Spending Score(1-100)')

plt.show()
```



In [21]:

```python
X2=df.loc[:, ["Annual Income (k$)", "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(X2)

    wcss.append(kmeans.inertia_)

plt.figure(figsize=(12,6))


plt.grid()
plt.plot(range(1,11), wcss, linewidth=2, color="red", marker ="8")
plt.xlabel("K Value")
plt.ylabel("WCSS")

plt.show()
```



In [22]:

```
kmeans = KMeans(n_clusters = 5)
label = kmeans.fit_predict(X2)
print(label)
```

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

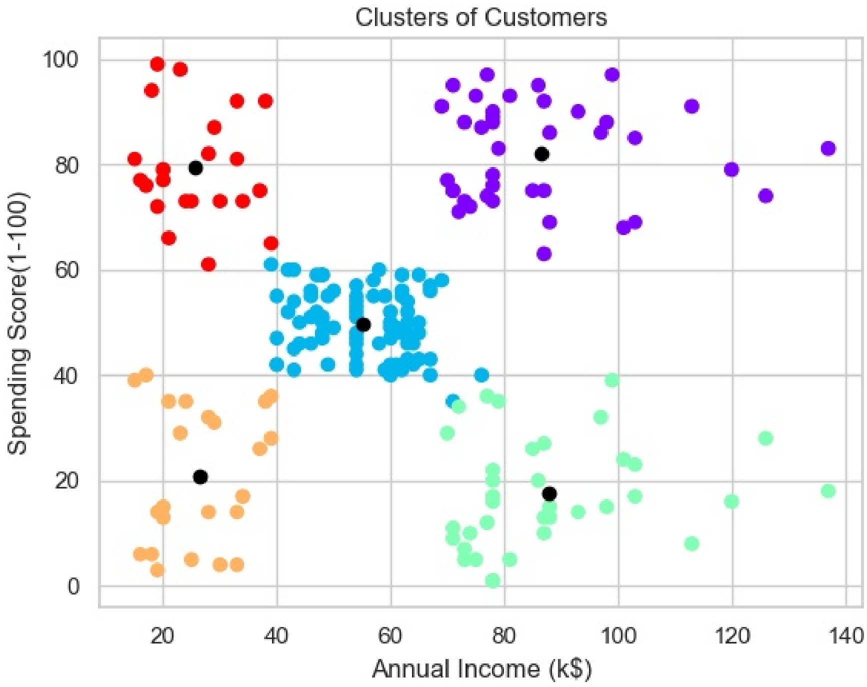
In [23]:

```
print(kmeans.cluster_centers_)
```

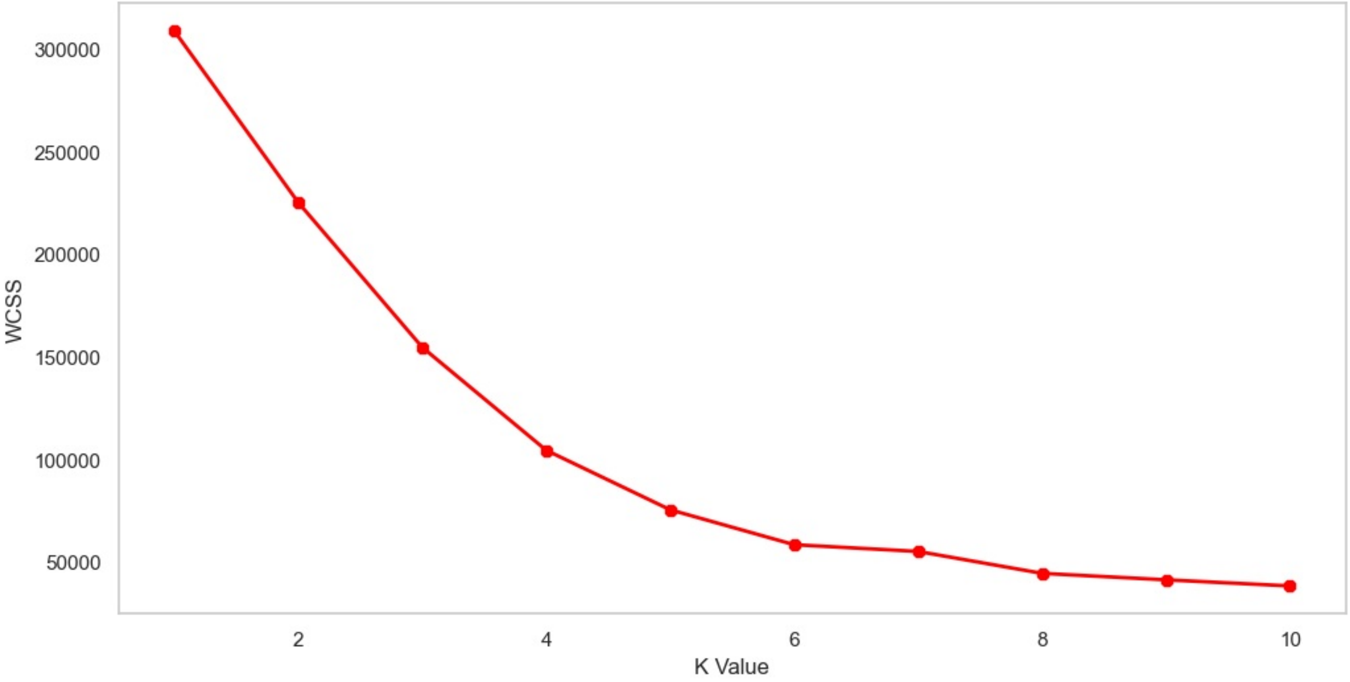
```
[[86.53846154 82.12820513]
 [55.0875     49.7125    ]
 [87.75       17.58333333]
 [26.30434783 20.91304348]
 [25.72727273 79.36363636]]
```

In [24]:

```
plt.scatter (X2[:,0], X1[:,1], c=kmeans.labels_, cmap='rainbow')

plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1], color = 'black')

plt.title('Clusters of Customers')

plt.xlabel('Annual Income (k$)')

plt.ylabel('Spending Score(1-100)')

plt.show()
```



```
In [25]: X3=df.iloc[:,1:]

         wcss = []

         for k in range(1,11):
             kmeans = KMeans (n_clusters=k, init="k-means++")

             kmeans.fit(X3)

             wcss.append(kmeans.inertia_)

         plt.figure(figsize=(12,6))

         plt.grid()

         plt.plot(range (1,11), wcss, linewidth=2, color="red", marker ="8")

         plt.xlabel("K Value")

         plt.ylabel("WCSS")

         plt.show()
```



```
In [26]: kmeans = KMeans(n_clusters = 5)
         label = kmeans.fit_predict(X3)
         print(label)
```

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

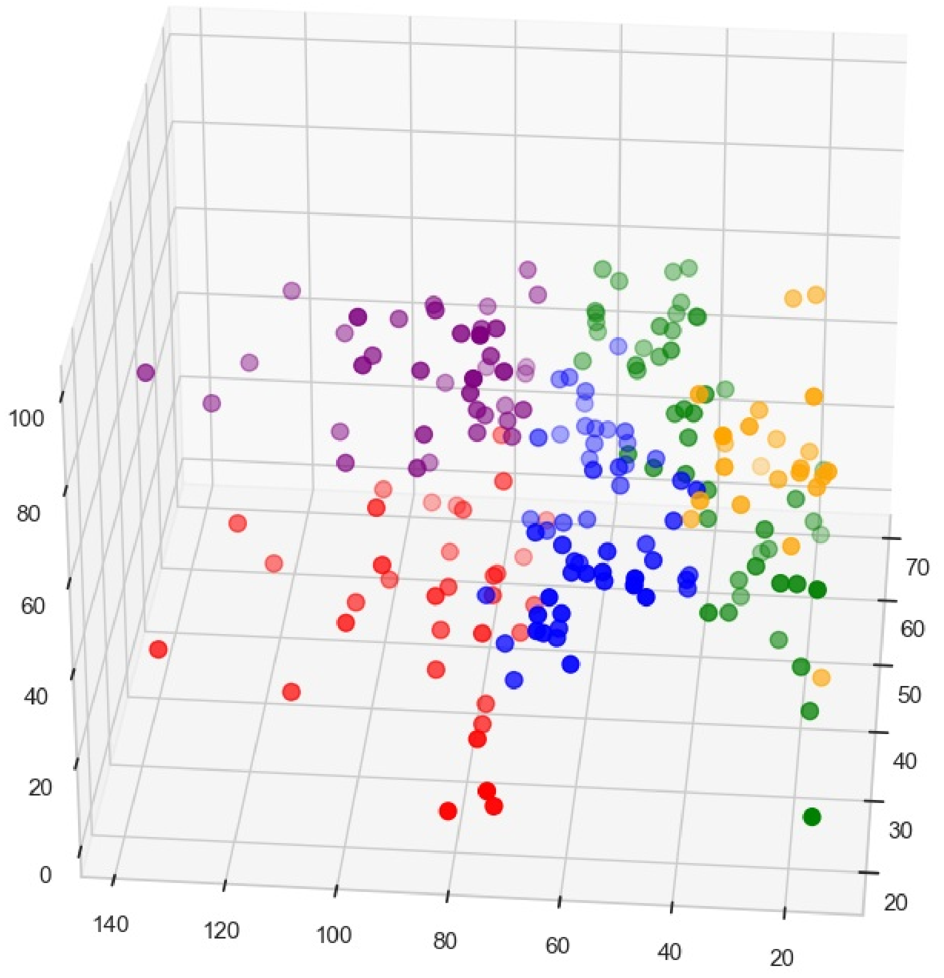
In [27]:

```
print(kmeans.cluster_centers_)
```

```
[[45.2173913  26.30434783 20.91304348]
 [32.69230769 86.53846154 82.12820513]
 [40.66666667 87.75       17.58333333]
 [24.96       28.04       77.        ]
 [43.72727273 55.48051948 49.32467532]]
```

In [28]:

```
clusters = kmeans.fit_predict(X3)
df["label"] = clusters

from mpl_toolkits.mplot3d import Axes3D

fig = plt.figure(figsize=(20,10))

ax = fig.add_subplot(111, projection='3d')

ax.scatter(df.Age[df.label==0], df["Annual Income (k$)"][df.label==0], df["Spending Score (1-100)"] [df.label==

ax.scatter(df.Age[df.label== 1], df["Annual Income (k$)"] [df.label ==1], df["Spending Score (1-100)"] [df.labe

ax.scatter(df.Age[df.label== 2], df["Annual Income (k$)"] [df.label== 2], df["Spending Score (1-100)"] [df.labe

ax.scatter(df.Age [df.label ==3], df["Annual Income (k$)"] [df.label== 3], df["Spending Score (1-100)"] [df.lab

ax.scatter(df.Age[df.label== 4], df["Annual Income (k$)"] [df.label== 4], df["Spending Score (1-100)"] [df.labe

ax.view_init(30, 185)
```



In [ ]:

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js