In [1]:

```python
# Import useful libraries (additional Python functionality):
import pandas as pd       # For data handling
import numpy as np        # For some fundamental calculations
import seaborn as sns     # For visualization
from matplotlib import pyplot as plt     # For scatterplots
from sklearn import cluster     # For machine learning (cluster analysis)
pd.reset_option('all')     # Ensure all settings are at their defaults
# Get data from Class Survey and assign to a dataframe:
surveyData = pd.read_csv('classSurvey2024a.csv')
```

```
C:\Users\jseyd\AppData\Local\Temp\ipykernel_45268\3152973706.py:7: FutureWarning: column_space is depre
cated and will be removed in a future version. Use df.to_string(col_space=...) instead.
  pd.reset_option('all')     # Ensure all settings are at their defaults
C:\Users\jseyd\AppData\Local\Temp\ipykernel_45268\3152973706.py:7: FutureWarning: As the xlwt package i
s no longer maintained, the xlwt engine will be removed in a future version of pandas. This is the only
engine in pandas that supports writing in the xls format. Install openpyxl and write to an xlsx file in
stead.
  pd.reset_option('all')     # Ensure all settings are at their defaults
C:\Users\jseyd\AppData\Local\Temp\ipykernel_45268\3152973706.py:7: FutureWarning:
: boolean
    use_inf_as_null had been deprecated and will be removed in a future
    version. Use `use_inf_as_na` instead.

  pd.reset_option('all')     # Ensure all settings are at their defaults
```

In [2]:

```python
# Explore the surveyData dataframe:
print('Here is the basic information: \n', surveyData.shape)
surveyData.info()
surveyData.describe()     # Note "object" means non-numeric (string) data
```

```
Here is the basic information:
 (65, 6)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65 entries, 0 to 64
Data columns (total 6 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   Name     65 non-null     object
 1   Major    65 non-null     object
 2   Excel    65 non-null     int64 
 3   Oral     65 non-null     int64 
 4   Written  65 non-null     int64 
 5   Stats    65 non-null     int64 
dtypes: int64(4), object(2)
memory usage: 3.2+ KB
```

Out[2]:

	Excel	Oral	Written	Stats
count	65.000000	65.000000	65.000000	65.000000
mean	3.000000	3.892308	3.723077	2.861538
std	0.883883	0.812463	0.875137	0.881705
min	1.000000	2.000000	2.000000	1.000000
25%	3.000000	3.000000	3.000000	2.000000
50%	3.000000	4.000000	4.000000	3.000000
75%	3.000000	4.000000	4.000000	3.000000
max	5.000000	5.000000	5.000000	5.000000

In [3]:

```
surveyData.head()     # First five observations
```

Out[3]:

	Name	Major	Excel	Oral	Written	Stats
0	Debra Adams	Mgt	1	3	3	1
1	Amy Allen	MBA	3	3	3	3
2	Nancy Anderson	MBA	3	5	5	3
3	Martha Baker	Fin	4	4	5	3
4	Michael Brown	Mgt	3	4	4	2

In [4]:

```
surveyData.tail()     # Last five observations
```

Out[4]:

	Name	Major	Excel	Oral	Written	Stats
60	Brian White	Econ	2	4	3	2
61	Robert Williams	nBus	1	3	2	2
62	Dorothy Wilson	Econ	2	4	3	1
63	Rebecca Wright	Econ	2	4	3	2
64	Brenda Young	Mktg	3	2	4	2

In [5]:

```
# Some more preliminary analysis:
surveyData['Excel'].value_counts()     # Frequencies
```

Out[5]:

```
3    36
4    14
2     8
1     5
5     2
Name: Excel, dtype: int64
```

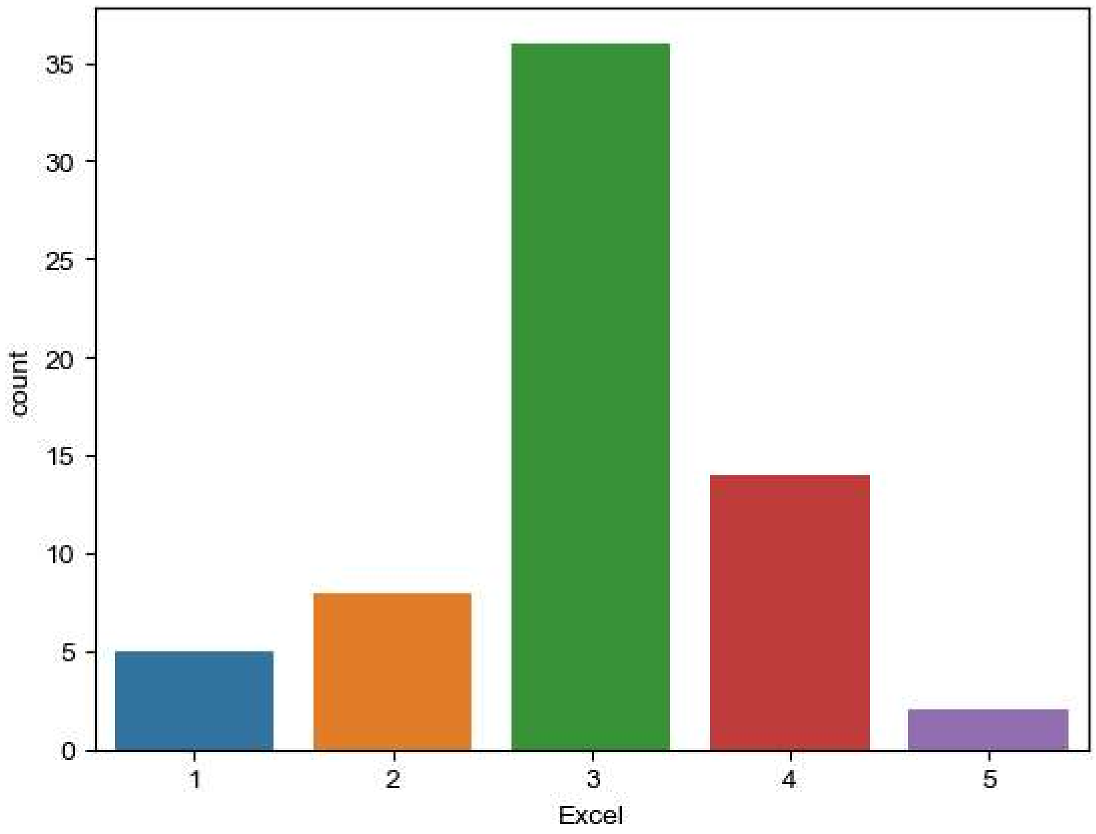
In [6]:

```python
# Convert some stats to string and concatenate, then display:
print('Average for Excel skills is '+str(surveyData['Excel'].mean()))
print('Median for Excel skills is '+str(surveyData['Excel'].median()))
```

```
Average for Excel skills is 3.0
Median for Excel skills is 3.0
```

In [7]:

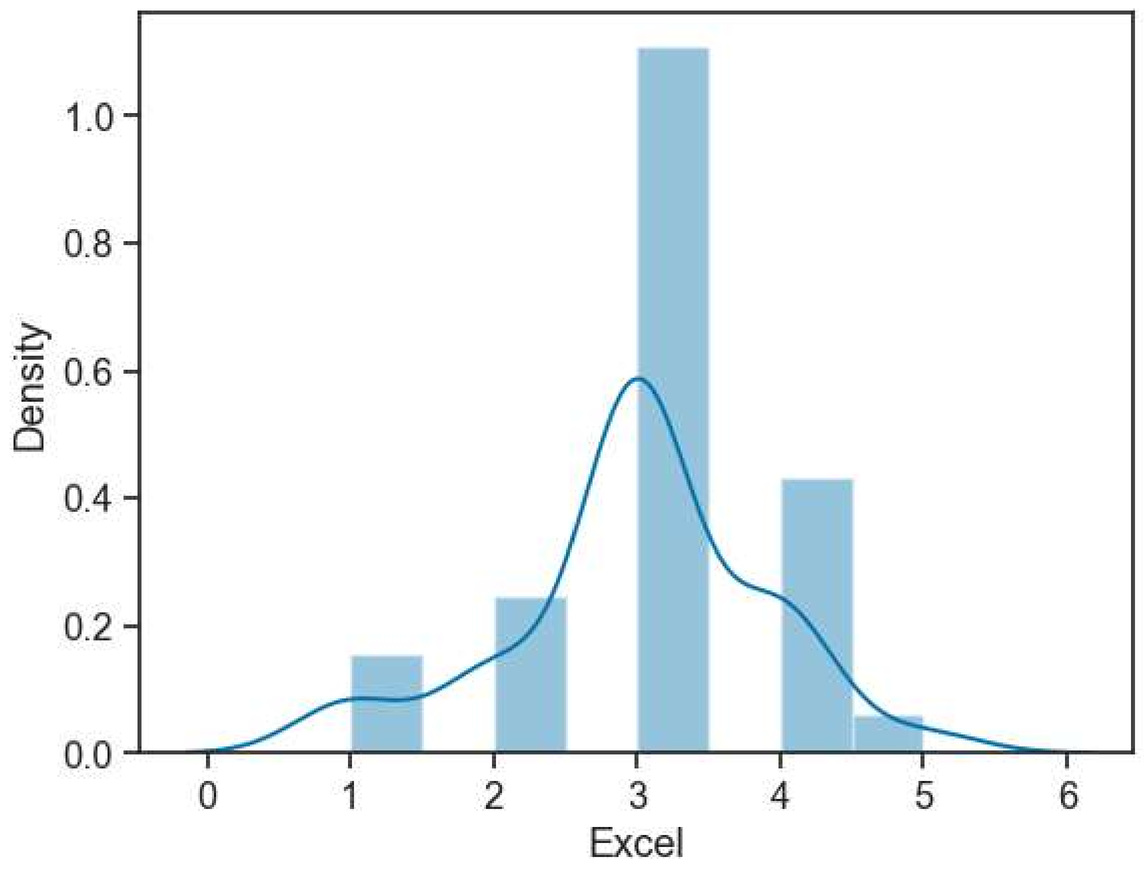
```python
# Now, some more interesting analytics:
# First, a bar chart:
ax = sns.countplot(x='Excel', data=surveyData)
sns.set(palette='colorblind', style='ticks', font_scale=1.2)
```



In [8]:

```python
# Now a histogram:
ax = sns.distplot(surveyData['Excel'])
```

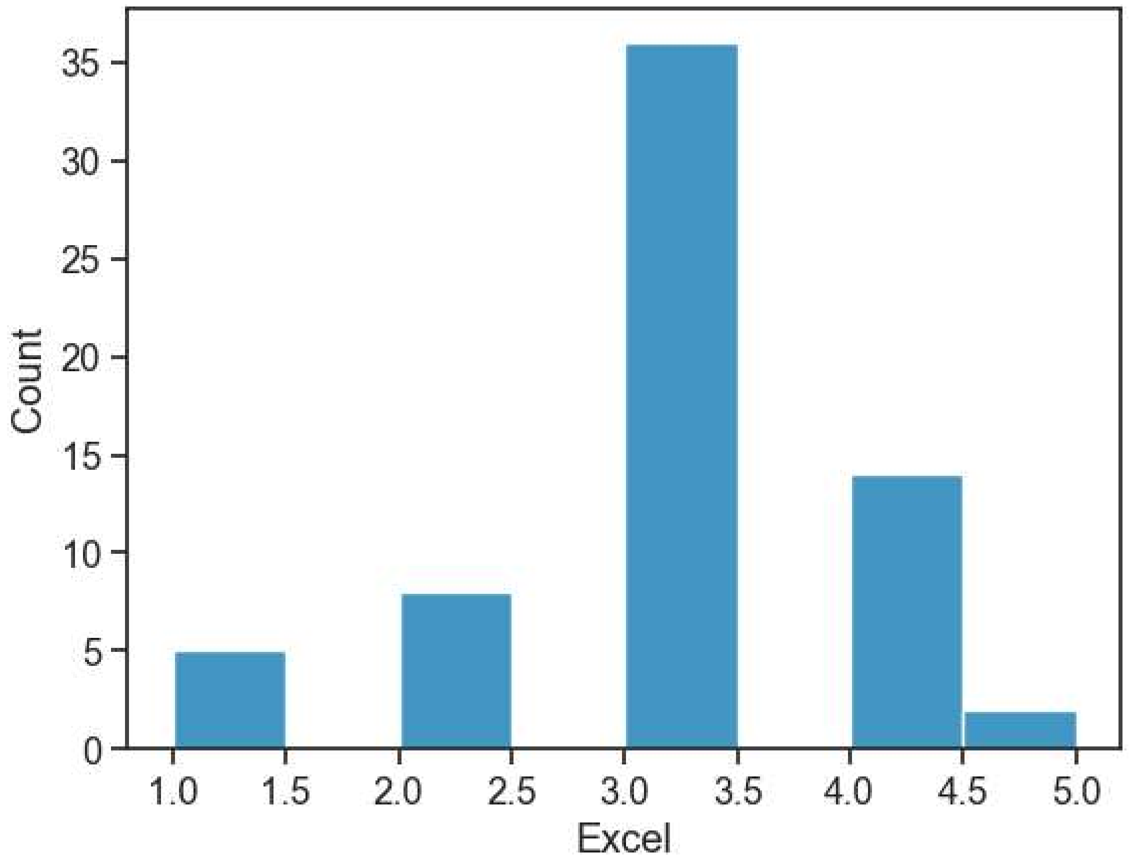
```
C:\Users\jseyd\AppData\Local\Temp\ipykernel_45268\163849847.py:2: UserWarning:

`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 (https://gist.github.com/mwaskom/de441
47ed2974457ad6372750bbe5751)

  ax = sns.distplot(surveyData['Excel'])
```



Density
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6
Excel

In [9]:

```python
# Without the approximated continuous distribution overlaid
ax = sns.histplot(surveyData['Excel'])
```



In [10]:

```
# Create new variables:  verbalSkills and quantSkills
surveyData['verbalSkills'] = (surveyData['Oral'] + surveyData['Written']) * 10
surveyData['quantSkills'] = (surveyData['Excel'] + surveyData['Stats']) * 10
surveyData.head()
```

Out[10]:

	Name	Major	Excel	Oral	Written	Stats	verbalSkills	quantSkills
0	Debra Adams	Mgt	1	3	3	1	60	20
1	Amy Allen	MBA	3	3	3	3	60	60
2	Nancy Anderson	MBA	3	5	5	3	100	60
3	Martha Baker	Fin	4	4	5	3	90	70
4	Michael Brown	Mgt	3	4	4	2	80	50

In [11]:

```
# Create a revised dataframe; first create arrays from the new columns:
lstData0 = surveyData['quantSkills']     # An array of quant skills
lstData1 = surveyData['verbalSkills']     # An array of verbal skills
# Create the new array, one column at a time
newData = pd.DataFrame(lstData0)
newData = newData.join(lstData1)
newData.info()
# Now, we have a dataframe containing only the variables for clustering
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65 entries, 0 to 64
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   quantSkills   65 non-null     int64
 1   verbalSkills  65 non-null     int64
dtypes: int64(2)
memory usage: 1.1 KB
```

In [12]:

```python
print(newData)
```

```
    quantSkills  verbalSkills
0            20            60
1            60            60
2            60           100
3            70            90
4            50            80
..          ...           ...
60           40            70
61           30            50
62           30            70
63           40            70
64           50            60

[65 rows x 2 columns]
```

In [13]:

```python
# Descriptive stats for the new dataframe:
newData.describe()
```

Out[13]:

	quantSkills	verbalSkills
count	65.000000	65.000000
mean	58.615385	76.153846
std	14.564182	14.327891
min	20.000000	40.000000
25%	50.000000	60.000000
50%	60.000000	80.000000
75%	70.000000	90.000000
max	100.000000	100.000000

In [14]:
```
# Take a look at the revised dataframe:
newData.head()
```

Out[14]:

	quantSkills	verbalSkills
0	20	60
1	60	60
2	60	100
3	70	90
4	50	80

In [15]:
```
newData.tail()
```

Out[15]:

	quantSkills	verbalSkills
60	40	70
61	30	50
62	30	70
63	40	70
64	50	60

K-means

The k-means algorithm applies the idea that an item should be closer to the center of its cluster than to that of any other cluster. It does so by selecting *k* points to be cluster centers, then it assigns the other points to the cluster with the closest center. It then calculates the new center of the cluster and assigns all the points to clusters based upon the new centers. This process repeats until there is a stable result.

Here is one general explanation of cluster analysis, along with two interesting visualizations that show how k-means works:

- https://www.datacamp.com/tutorial/k-means-clustering-python (https://www.datacamp.com/tutorial/k-means-clustering-python)
- http://shabal.in/visuals.html (http://shabal.in/visuals.html) click "series of 5 gif animations" link

- https://www.naftaliharris.com/blog/visualizing-k-means-clustering/ (https://www.naftaliharris.com/blog/visualizing-k-means-clustering/)

We use the `KMeans` function from the sklearn library to create a k-means model.

In [16]:

```
# Now, the cluster analysis:
numClusters = 4
# Calculate the centroid for each of the numClusters clusters:
kMeans = cluster.KMeans(n_clusters=numClusters, random_state=1000).fit(newData)
# Create a label for each of the numClusters clusters:
labels = cluster.KMeans(n_clusters = numClusters).fit(newData).labels_
centroids = kMeans.cluster_centers_
print('Centroids:')
print(centroids)
```

```
C:\Users\jseyd\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default v
alue of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress
the warning
  warnings.warn(
C:\Users\jseyd\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is know
n to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can
avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\jseyd\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default v
alue of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress
the warning
  warnings.warn(
C:\Users\jseyd\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is know
n to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can
avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(

Centroids:
[[57.72727273 87.27272727]
 [62.22222222 61.66666667]
 [35.83333333 66.66666667]
 [76.15384615 86.15384615]]
```

In [17]:
```
# Display the cluster labels, then the number of observations each
print(np.unique(kMeans.labels_, return_counts=True))
```

```
(array([0, 1, 2, 3]), array([22, 18, 12, 13], dtype=int64))
```

In [18]:
```
# Create a cluster data frame for later use
kMeansResults = pd.DataFrame(kMeans.labels_, columns=['Cluster'])
kMeansResults.head()     # Display each observation index and its assigned clusters
```

Out[18]:

	Cluster
0	2
1	1
2	0
3	3
4	0

In [19]:

```
# Get names from original dataframe and assign to the results dataframe:
lstDataNames = surveyData['Name']      # Create an array (series) of names
lstDataNames = lstDataNames.tolist()     # Convert that array to a list
kMeansResults['Names'] = lstDataNames     # Append a column of names
kMeansResults = kMeansResults[['Names','Cluster']]   # Rearrange columns
pd.set_option('display.max_rows', None)     # Specify printing all rows
sortedResults = kMeansResults.sort_values(by=['Cluster', 'Names'])  # Sort by cluster
print(kMeansResults)
```

```
                 Names  Cluster
0          Debra Adams        2
1            Amy Allen        1
2       Nancy Anderson        0
3         Martha Baker        3
4        Michael Brown        0
5   Christine Campbell        3
6     Catherine Carter        1
7        Deborah Clark        1
8        Julie Collins        0
9         Mildred Cook        3
10       Judith Cooper        0
11       Richard Davis        3
12        Jack Edwards        1
13           Ann Evans        1
14     Jonathan Flores        3
15      Charles Garcia        3
16      Sarah Gonzalez        0
17        Pamela Green        2
18        Melissa Hall        1
19        Jason Harris        0
20     Helen Hernandez        0
21      Stephanie Hill        1
22       Carol Jackson        3
23    Patricia Johnson        1
24       William Jones        2
25       Virginia King        0
26           Laura Lee        2
27       Jessica Lewis        2
28      Michelle Lopez        0
29        Donna Martin        0
30       Lisa Martinez        3
31        David Miller        0
32      Marie Mitchell        0
33        George Moore        1
34        Ralph Morgan        0
35       Justin Morris        1
36       Gerald Murphy        1
37       Amanda Nelson        1
38        Terry Nguyen        3
39        Alice Parker        0
40         Angela Perez        0
41   Lawrence Peterson        0
```

```
42       Carl Phillips        3
43     Carolyn Ramirez        0
44            Roy Reed        3
45        Keith Rivera        0
46     Douglas Roberts        1
47    Timothy Robinson        3
48  Margaret Rodriguez        0
49       Willie Rogers        1
50        Eric Sanchez        1
51      Kathleen Scott        0
52          Mary Smith        0
53      Albert Stewart        0
54        Karen Taylor        2
55         Mark Thomas        3
56       Ruth Thompson        2
57        Roger Torres        1
58         Ryan Turner        1
59      Cynthia Walker        2
60         Brian White        2
61     Robert Williams        2
62      Dorothy Wilson        2
63      Rebecca Wright        2
64        Brenda Young        1
```

In [20]:

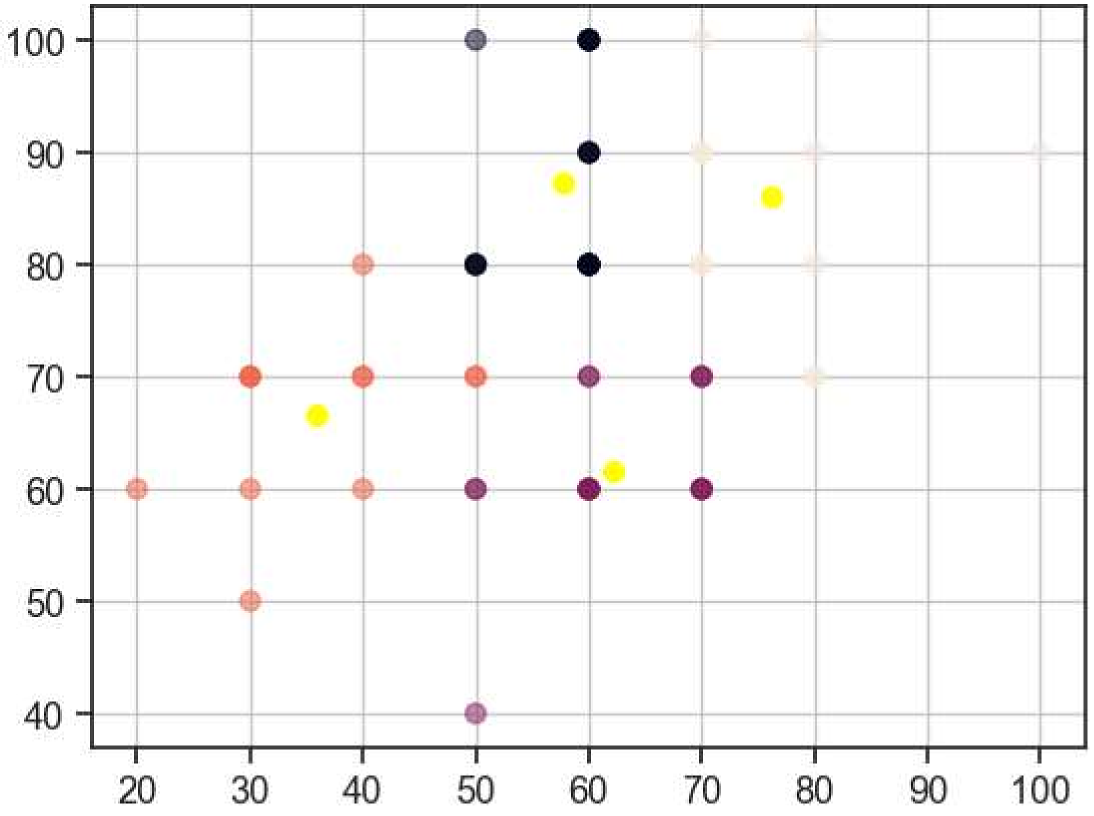
```
print(sortedResults)
```

```
                 Names  Cluster
53      Albert Stewart        0
39        Alice Parker        0
40        Angela Perez        0
43     Carolyn Ramirez        0
31        David Miller        0
29        Donna Martin        0
20     Helen Hernandez        0
19        Jason Harris        0
10       Judith Cooper        0
8       Julie Collins        0
51      Kathleen Scott        0
45        Keith Rivera        0
41   Lawrence Peterson        0
48  Margaret Rodriguez        0
32      Marie Mitchell        0
52          Mary Smith        0
4       Michael Brown        0
28      Michelle Lopez        0
2      Nancy Anderson        0
34        Ralph Morgan        0
16      Sarah Gonzalez        0
25       Virginia King        0
37       Amanda Nelson        1
1           Amy Allen        1
13           Ann Evans        1
64        Brenda Young        1
6     Catherine Carter        1
7       Deborah Clark        1
46     Douglas Roberts        1
50        Eric Sanchez        1
33        George Moore        1
36       Gerald Murphy        1
12        Jack Edwards        1
35       Justin Morris        1
18        Melissa Hall        1
23    Patricia Johnson        1
57        Roger Torres        1
58         Ryan Turner        1
21      Stephanie Hill        1
49       Willie Rogers        1
60         Brian White        2
59      Cynthia Walker        2
```

```
0          Debra Adams        2
62      Dorothy Wilson        2
27       Jessica Lewis        2
54        Karen Taylor        2
26           Laura Lee        2
17        Pamela Green        2
63      Rebecca Wright        2
61     Robert Williams        2
56       Ruth Thompson        2
24       William Jones        2
42       Carl Phillips        3
22       Carol Jackson        3
15      Charles Garcia        3
5   Christine Campbell        3
14     Jonathan Flores        3
30       Lisa Martinez        3
55         Mark Thomas        3
3         Martha Baker        3
9          Mildred Cook        3
11       Richard Davis        3
44            Roy Reed        3
38        Terry Nguyen        3
47     Timothy Robinson       3
```

In [21]:
```
# Create a scatterplot of the clusters, quantSkills on horizontal axis:
# Generate the grid:
plt.grid(visible=None, which='major', axis='both')
# Plot the data points (each cluster has a different color):
plt.scatter(newData['quantSkills'], newData['verbalSkills'], c= kMeans.labels_.astype(float), s=50,
# Plot the cluster centroids (in yellow):
plt.scatter(centroids[:, 0], centroids[:, 1], c="yellow", s=50)
```

Out[21]:
```
<matplotlib.collections.PathCollection at 0x243faf564d0>
```


100
90
80
70
60
50
40
20
30
40
50
60
70
80
90
100