Introduction to Numpy, Pandas and Matplotlib

In [4]:

```
#question1 solution-
import numpy as np
import pandas as pd

df = pd.read_csv('SalaryGender.csv')
df
```

Out[4]:

	Salary	Gender	Age	PhD
0	140.0	1	47	1
1	30.0	0	65	1
2	35.1	0	56	0
3	30.0	1	23	0
4	80.0	0	53	1
...	...	...	...	...
95	18.6	1	26	0
96	152.0	1	56	1
97	1.8	1	28	0
98	35.0	0	44	0
99	4.0	0	24	0

100 rows × 4 columns

In [8]:

```
#col0 = arr[:, 0]
#col1 = arr[:, 1]
#col2 = arr[:, 2]
#col3 = arr[:, 3]

salary = np.array(df["Salary"])
gender = np.array(df["Gender"])
age = np.array(df["Age"])
phd = np.array(df["PhD"])
```

In [11]:

```
#question2 solution
mens_phd = df.query("Gender == 1 and PhD == 1")
womens_phd = df.query("Gender == 0 and PhD == 0")
print("Mens Phd", mens_phd)
print("Womens Phd", womens_phd)
```

```
Mens Phd     Salary  Gender  Age  PhD
0    140.0       1   47    1
8    125.0       1   44    1
9     51.0       1   63    1
12   150.0       1   60    1
18   190.0       1   66    1
19    74.0       1   45    1
25    15.2       1   66    1
26    28.6       1   36    1
29    81.0       1   65    1
38    63.0       1   34    1
42   106.0       1   77    1
47    55.0       1   49    1
56   160.0       1   61    1
60   102.0       1   66    1
63    55.0       1   56    1
73   152.0       1   71    1
76    30.0       1   69    1
77   120.0       1   58    1
79    36.0       1   32    1
80    96.0       1   33    1
87    72.0       1   42    1
90    89.0       1   71    1
92    52.0       1   55    1
96   152.0       1   56    1
Womens Phd     Salary  Gender  Age  PhD
2    35.10       0   56    0
5    30.00       0   27    0
7    31.10       0   30    0
15   15.00       0   25    0
20   73.00       0   46    0
21   10.00       0   24    0
22   50.00       0   60    0
23    7.00       0   63    0
24    9.50       0   27    0
27   20.00       0   30    0
34   30.00       0   52    0
36   52.00       0   49    0
37    9.00       0   22    0
44    9.00       0   27    0
46   32.00       0   45    0
50   20.00       0   32    0
51   14.70       0   49    0
53   34.80       0   22    0
58   55.00       0   52    0
62   62.00       0   62    0
65   40.00       0   56    0
66   24.00       0   41    0
68   48.00       0   60    0
69   20.00       0   43    0
70   40.70       0   57    0
72    0.25       0   53    0
74   39.80       0   20    0
75   12.00       0   27    0
84   25.80       0   30    0
85   22.00       0   62    0
86   38.80       0   54    0
```

```
91   25.00       0   29    0
93  115.00       0   54    0
98   35.00       0   44    0
99    4.00       0   24    0
```

In [9]:

```
#solution 3
#filter columns
age_phd = df.filter(["Age", "PhD"])

#filter rows
new_df = age_phd.drop(age_phd[age_phd["PhD"] == 0].index)
new_df
```

Out[9]:

	Age	PhD
0	47	1
1	65	1
4	53	1
8	44	1
9	63	1
12	60	1
17	47	1
18	66	1
19	45	1
25	66	1
26	36	1
28	51	1
29	65	1
30	45	1
31	52	1
32	54	1
38	34	1
41	58	1
42	77	1
45	48	1
47	49	1
49	65	1
54	49	1
56	61	1
57	43	1
60	66	1
63	56	1
73	71	1
76	69	1
77	58	1
79	32	1
80	33	1
81	32	1
87	42	1
89	51	1

	Age	PhD
90	71	1
92	55	1
94	55	1
96	56	1

In [12]:

```
#question 4
phd_people = df.query("PhD==1")

print("Total number of people with phd degree >>" +str(len(phd_people)))
```

```
Total number of people with phd degree >>39
```

In [14]:

```
#question 5
array = [0, 5, 4, 0, 4, 4, 3, 0, 0, 5, 2, 1, 1, 9]

count_elements = [array.count(a) for a in set(array)]
count_elements
```

Out[14]:

```
[4, 2, 1, 1, 3, 2, 1]
```

In [9]:

```
#question 6
arr = np.array([[0, 1, 2],[ 3, 4, 5],[ 6, 7, 8],[ 9,10, 11]])

print(arr[arr > 5])
```

```
[ 6  7  8  9 10 11]
```

In [11]:

```
#question 7
np_array = np.array([np.NaN, 1, 2., np.NaN, 3., 4., 5.])

array_without_nan = np_array[~np.isnan(np_array)]
print(array_without_nan)
```

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

In [15]:

```
#question 8
np_array = np.random.random((10,10))

print("Max Value"+ str(np_array.max()))
print("Min Value"+ str(np_array.min()))
```

```
Max Value0.9843668485714524
Min Value0.00234364351886418
```

In [5]:

```
#question 9
import random
np_array = np.random.random(30)

x,y = 0,0
while x*y != 30:
    x = random.randint(1,31)
    y = random.randint(1,31)

np_array = np_array.reshape(x,y)
print(np_array)

print("\nMean of the above array is > " + str(np_array.mean()))
```

```
[[0.8636327  0.9713802  0.06669515 0.39683073 0.76173883]
 [0.12470795 0.25282334 0.02892806 0.31297895 0.90738924]
 [0.46671963 0.71890309 0.03705779 0.4957565  0.36106357]
 [0.05470808 0.6328011  0.71438694 0.88266042 0.78363201]
 [0.4119358  0.1757978  0.27318856 0.00532378 0.39106528]
 [0.03127197 0.03376745 0.75149817 0.10768493 0.34779459]]

Mean of the above array is > 0.4121374199494453
```

In [11]:

```
#question 10
array = np.arange(0,10)
array

new_array = [(-i if (i>3 and i<9) else i) for i in array]
new_array
```

Out[11]:

```
[0, 1, 2, 3, -4, -5, -6, -7, -8, 9]
```

In [23]:

```
#question 11
np_array = np.random.random(9).reshape(3,3)
print("np_array>>>\n", np_array)
array_new = np.sort(np_array, axis=0)
print("array_new >>>\n", array_new)
```

```
np_array>>>
 [[0.15314058 0.01473654 0.2283223 ]
 [0.65811904 0.59516974 0.11516681]
 [0.30160516 0.85318228 0.03881251]]
array_new >>>
 [[0.15314058 0.01473654 0.03881251]
 [0.30160516 0.59516974 0.11516681]
 [0.65811904 0.85318228 0.2283223 ]]
```

In [24]:

```
#question 12
array = np.random.random(16).reshape(2,2,2,2)

summed_array = np.sum(array, axis = 0)
print(summed_array)
```

```
[[[0.92352236 0.30135945]
  [0.68434259 0.80925283]]

 [[1.54890221 1.12037347]
  [0.88037448 1.04160125]]]
```

In [30]:

```
#question 13
np_array = np.random.random((3, 3))
print(np_array)

np_array[[0,1]] = np_array[[1,0]]
print(np_array)
```

```
[[0.21430576 0.18751135 0.22698247]
 [0.8886796  0.53322405 0.34526503]
 [0.81378057 0.26710841 0.28495848]]
[[0.8886796  0.53322405 0.34526503]
 [0.21430576 0.18751135 0.22698247]
 [0.81378057 0.26710841 0.28495848]]
```

In [31]:

```
#question 14
np_array = np.random.random(16).reshape(2, 2, 2, 2)
print(np_array)

array = np.array([4, 2, 7, 1])
temp = array.argsort()
ranks = np.empty_like(temp)
ranks[temp] = np.arange(len(array))

print(ranks)
```

```
[[[[0.17053989 0.35712421]
   [0.37350718 0.56990101]]

  [[0.20320805 0.36286588]
   [0.4971678  0.66129578]]]


 [[[0.20237198 0.97550742]
   [0.86548306 0.43985319]]

  [[0.63728093 0.86090059]
   [0.6033658  0.13855684]]]]
[2 1 3 0]
```

In [40]:

```
#question 15
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import norm
import seaborn as sns

#phase 1
df_school_data = pd.read_csv("middle_tn_schools.csv")
print(df_school_data.head())
df_school_data.describe()
```

```
                          name  school_rating   size  reduced_lunch  \
0  Allendale Elementary School            5.0  851.0           10.0   
1          Anderson Elementary            2.0  412.0           71.0   
2             Avoca Elementary            4.0  482.0           43.0   
3                Bailey Middle            0.0  394.0           91.0   
4          Barfield Elementary            4.0  948.0           26.0   

   state_percentile_16  state_percentile_15  stu_teach_ratio    school_type  
\
0                 90.2                 95.8             15.7         Public   
1                 32.8                 37.3             12.8         Public   
2                 78.4                 83.6             16.6         Public   
3                  1.6                  1.0             13.1  Public Magnet   
4                 85.3                 89.2             14.8         Public   

   avg_score_15  avg_score_16  full_time_teachers  percent_black  \
0          89.4          85.2                54.0            2.9   
1          43.0          38.3                32.0            3.9   
2          75.7          73.0                29.0            1.0   
3           2.1           4.4                30.0           80.7   
4          81.3          79.6                64.0           11.8   

   percent_white  percent_asian  percent_hispanic  
0           85.5            1.6               5.6  
1           86.7            1.0               4.9  
2           91.5            1.2               4.4  
3           11.7            2.3               4.3  
4           71.2            7.1               6.0  
```

Out[40]:

	school_rating	size	reduced_lunch	state_percentile_16	state_percentile_15	stu_te
count	347.000000	347.000000	347.000000	347.000000	341.000000	3
mean	2.968300	699.472622	50.279539	58.801729	58.249267	
std	1.690377	400.598636	25.480236	32.540747	32.702630	
min	0.000000	53.000000	2.000000	0.200000	0.600000	
25%	2.000000	420.500000	30.000000	30.950000	27.100000	
50%	3.000000	595.000000	51.000000	66.400000	65.800000	
75%	4.000000	851.000000	71.500000	88.000000	88.600000	
max	5.000000	2314.000000	98.000000	99.800000	99.800000	1

In [42]:

```
#phase 2
df_grouped_data = df_school_data.groupby("school_rating").describe()
df_grouped_data.head()
```

Out[42]:

	size								reduced_lunc	
	count	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**	**count**	**mean**
school_rating										
0.0	43.0	501.325581	217.273880	71.0	367.00	426.0	563.00	1002.0	43.0	83.58
1.0	40.0	691.250000	476.695395	118.0	409.50	507.5	759.75	2314.0	40.0	74.95
2.0	44.0	628.500000	349.591755	53.0	368.25	558.0	752.75	1771.0	44.0	64.27
3.0	56.0	762.482143	399.760564	249.0	491.00	652.5	880.50	1983.0	56.0	50.28
4.0	86.0	742.732558	403.389242	141.0	452.50	641.5	934.75	2025.0	86.0	41.00

5 rows × 96 columns

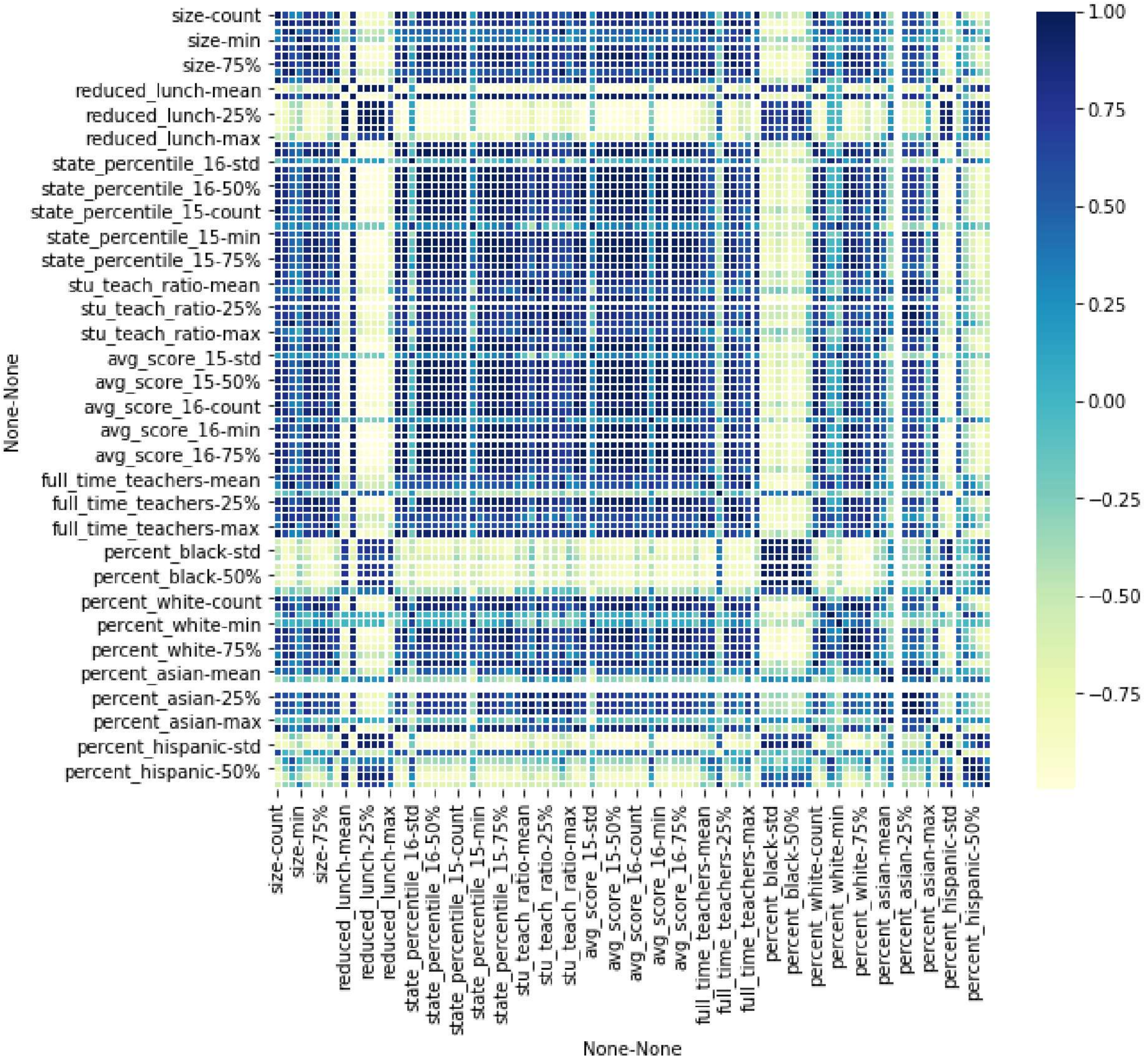
In [47]:

```
#phase 3- correlation analysis
corr_data = df_grouped_data.corr()
print(corr_data)

f,ax = plt.subplots(figsize=(9,8))
sns.heatmap(corr_data, ax=ax, cmap='YlGnBu', linewidths = 0.1)
```

```
                            size                                          \
                           count      mean       std       min       25%  
size             count  1.000000  0.638544  0.310568  0.307444  0.714111
                 mean   0.638544  1.000000  0.850572  0.690995  0.860659
                 std    0.310568  0.850572  1.000000  0.419827  0.574870
                 min    0.307444  0.690995  0.419827  1.000000  0.723631
                 25%    0.714111  0.860659  0.574870  0.723631  1.000000
...                          ...       ...       ...       ...       ...
percent_hispanic min    0.631280 -0.009199 -0.195401 -0.479453  0.117490
                 25%   -0.444132  0.192275  0.628663  0.084906 -0.171422
                 50%   -0.734607 -0.126214  0.337831 -0.059610 -0.418815
                 75%   -0.873944 -0.544436 -0.080780 -0.348794 -0.711592
                 max   -0.588382 -0.333052 -0.251458  0.357119 -0.241869

                                                     reduced_lunch
\
                             50%       75%       max         count      mean
size             count  0.774017  0.762666  0.387119      1.000000 -0.872671
                 mean   0.912088  0.958276  0.882831      0.638544 -0.797098
                 std    0.602167  0.741112  0.991256      0.310568 -0.494689
                 min    0.593791  0.612925  0.401803      0.307444 -0.387303
                 25%    0.872567  0.790153  0.629227      0.714111 -0.871423
...                          ...       ...       ...           ...       ...
percent_hispanic min    0.289872  0.164128 -0.086918      0.631280 -0.539579
                 25%   -0.197570  0.046811  0.532618     -0.444132  0.364578
                 50%   -0.481354 -0.287835  0.231593     -0.734607  0.639141
                 75%   -0.819486 -0.667890 -0.187965     -0.873944  0.885189
                 max   -0.538577 -0.458849 -0.359289     -0.588382  0.674689

                        ... percent_asian           percent_hispanic  \
                        ...           75%       max            count   
size             count  ...      0.539632 -0.235073         1.000000
                 mean   ...      0.556293 -0.116374         0.638544
                 std    ...      0.489890  0.132930         0.310568
                 min    ...      0.138746 -0.209871         0.307444
                 25%    ...      0.752009  0.126393         0.714111
...                     ...           ...       ...              ...
percent_hispanic min    ...      0.410432 -0.059789         0.631280
                 25%    ...     -0.162483  0.176489        -0.444132
                 50%    ...     -0.340187  0.245122        -0.734607
                 75%    ...     -0.503310  0.299353        -0.873944
                 max    ...     -0.508199  0.097022        -0.588382

                                                                          \
                            mean       std       min       25%       50%  
size             count -0.856016 -0.766610  0.631280 -0.444132 -0.734607
                 mean  -0.617282 -0.839807 -0.009199  0.192275 -0.126214
                 std   -0.184982 -0.577295 -0.195401  0.628663  0.337831
                 min   -0.344294 -0.376795 -0.479453  0.084906 -0.059610
                 25%   -0.714818 -0.726424  0.117490 -0.171422 -0.418815
...                          ...       ...       ...       ...       ...
percent_hispanic min   -0.626755 -0.444484  1.000000 -0.677589 -0.770171
                 25%    0.622110  0.202805 -0.677589  1.000000  0.932617
                 50%    0.831255  0.488467 -0.770171  0.932617  1.000000
                 75%    0.986343  0.803746 -0.642217  0.707622  0.896360
                 max    0.663737  0.725483 -0.843420  0.321273  0.503302
```

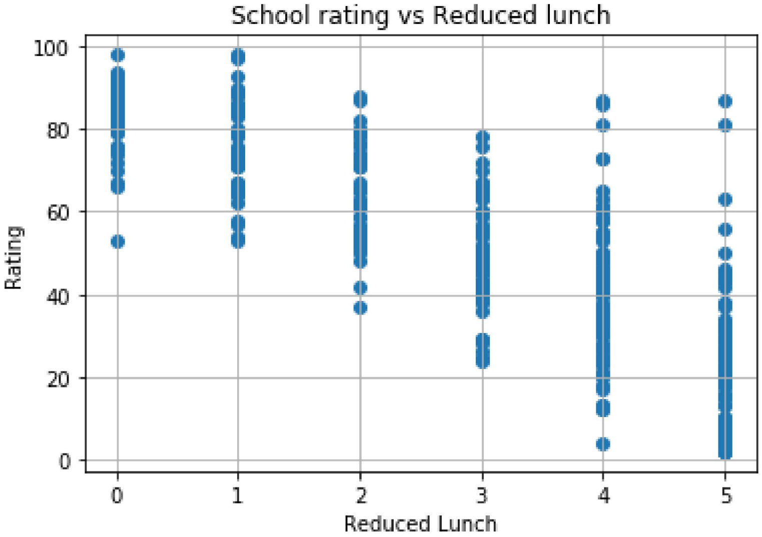
```
                                   75%       max
size              count -0.873944 -0.588382
                  mean  -0.544436 -0.333052
                  std   -0.080780 -0.251458
                  min   -0.348794  0.357119
                  25%   -0.711592 -0.241869
...                          ...       ...
percent_hispanic min   -0.642217 -0.843420
                  25%    0.707622  0.321273
                  50%    0.896360  0.503302
                  75%    1.000000  0.591321
                  max    0.591321  1.000000

[96 rows x 96 columns]
```

Out[47]: `<matplotlib.axes._subplots.AxesSubplot at 0x1ed56032788>`



In [50]:

```
#phase 4
plt.scatter(df_school_data["school_rating"], df_school_data["reduced_lunch"])
plt.grid()
plt.xlabel("Reduced Lunch")
plt.ylabel("Rating")
plt.title("School rating vs Reduced lunch")
plt.show()
```



In [ ]: