```
In [1]: import pandas as pd
```

Suppose you are a public school administrator. Some schools in your state of Tennessee are performing below average academically. Your superintendent under pressure from frustrated parents and voters approached you with the task of understanding why these schools are under-performing.

To improve school performance, you need to learn more about these schools and their students, just as a business needs to understand its own strengths and weaknesses and its customers. The data includes various demographic, school faculty, and income variables.

read CSV data file

```
In [2]: data = pd.read_csv('~/MyWorkPython/data/middle_tn_schools.csv')
```

check the data

```
In [33]: #number of rows and columns
         print(data.shape)

(347, 15)
```

```
In [4]: data.describe()
```

Out[4]:

	school_rating	size	reduced_lunch	state_percentile_16	state_percentile_15	stu_te
count	347.000000	347.000000	347.000000	347.000000	341.000000	34
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 [5]:
```
data.head()
```

Out[5]:

	name	school_rating	size	reduced_lunch	state_percentile_16	state_percentile_15	stu_t
0	Allendale Elementary School	5.0	851.0	10.0	90.2	95.8	
1	Anderson Elementary	2.0	412.0	71.0	32.8	37.3	
2	Avoca Elementary	4.0	482.0	43.0	78.4	83.6	
3	Bailey Middle	0.0	394.0	91.0	1.6	1.0	
4	Barfield Elementary	4.0	948.0	26.0	85.3	89.2	

In [6]:
```
# check the null value in the columns
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 347 entries, 0 to 346
Data columns (total 15 columns):
name                   347 non-null object
school_rating          347 non-null float64
size                   347 non-null float64
reduced_lunch          347 non-null float64
state_percentile_16    347 non-null float64
state_percentile_15    341 non-null float64
stu_teach_ratio        347 non-null float64
school_type            347 non-null object
avg_score_15           341 non-null float64
avg_score_16           347 non-null float64
full_time_teachers     347 non-null float64
percent_black          347 non-null float64
percent_white          347 non-null float64
percent_asian          347 non-null float64
percent_hispanic       347 non-null float64
dtypes: float64(13), object(2)
memory usage: 40.8+ KB
```

In [7]:
```
#check wheather there are any missing values or null
data.isnull().sum()
```

Out[7]:
```
name                   0
school_rating          0
size                   0
reduced_lunch          0
state_percentile_16    0
state_percentile_15    6
stu_teach_ratio        0
school_type            0
avg_score_15           6
avg_score_16           0
full_time_teachers     0
percent_black          0
percent_white          0
percent_asian          0
percent_hispanic       0
dtype: int64
```

replace the null value with the mean or 0

In [8]:
```
print ("state_percentile_15 column MEAN is  "+ str(data["state_percentile_15"]
.mean())+"\n")
print ("avg_score_15 column MEAN is  "+ str(data["avg_score_15"].mean())+"\n")
```

```
state_percentile_15 column MEAN is  58.24926686217009

avg_score_15 column MEAN is  57.00469208211145
```

In [9]:
```
# get the state_percentile_15 column
print(data.iloc[:,5:6].head())
#represent the singler dimentional array
print(data['state_percentile_15'])
```

```
   state_percentile_15
0                 95.8
1                 37.3
2                 83.6
3                  1.0
4                 89.2
0      95.8
1      37.3
2      83.6
3       1.0
4      89.2
       ...
342    65.2
343    97.0
344    76.7
345    97.1
346     1.2
Name: state_percentile_15, Length: 347, dtype: float64
```

In [10]:
```
from sklearn.impute import SimpleImputer
import numpy as np
```

In [11]:
```
# updated the state_percentile_15 missing values
# we added the strategy = mean

imputer = SimpleImputer(np.nan,strategy ="mean")

imputer.fit(data.iloc[:,5:6])

data.iloc[:,5:6] = imputer.transform(data.iloc[:,5:6])

data.isnull().sum()
```

Out[11]:
```
name                   0
school_rating          0
size                   0
reduced_lunch          0
state_percentile_16    0
state_percentile_15    0
stu_teach_ratio        0
school_type            0
avg_score_15           6
avg_score_16           0
full_time_teachers     0
percent_black          0
percent_white          0
percent_asian          0
percent_hispanic       0
dtype: int64
```

In [12]:
```python
# get the avg_score_15
print(data.iloc[:,8:9].head())
#represent the singler dimentional array
print(data['avg_score_15'])
```

```
   avg_score_15
0          89.4
1          43.0
2          75.7
3           2.1
4          81.3
0      89.4
1      43.0
2      75.7
3       2.1
4      81.3
       ...
342    61.4
343    92.0
344    69.4
345    89.8
346     2.7
Name: avg_score_15, Length: 347, dtype: float64
```

In [13]:
```python
# updated the avg_score_15 missing values
# we added the strategy = mean

imputer = SimpleImputer(np.nan,strategy ="mean")

imputer.fit(data.iloc[:,8:9])

data.iloc[:,8:9] = imputer.transform(data.iloc[:,8:9])

data.isnull().sum()
```

Out[13]:
```
name                   0
school_rating          0
size                   0
reduced_lunch          0
state_percentile_16    0
state_percentile_15    0
stu_teach_ratio        0
school_type            0
avg_score_15           0
avg_score_16           0
full_time_teachers     0
percent_black          0
percent_white          0
percent_asian          0
percent_hispanic       0
dtype: int64
```

In [14]:
```
data["school_type"].value_counts()
```

Out[14]:
```
Public            292
Public Magnet      46
Public Charter      8
Public Virtual      1
Name: school_type, dtype: int64
```

In [22]:
```
data.head()
```

Out[22]:

	name	school_rating	size	reduced_lunch	state_percentile_16	state_percentile_15	stu_t
0	Allendale Elementary School	5.0	851.0	10.0	90.2	95.8	
1	Anderson Elementary	2.0	412.0	71.0	32.8	37.3	
2	Avoca Elementary	4.0	482.0	43.0	78.4	83.6	
3	Bailey Middle	0.0	394.0	91.0	1.6	1.0	
4	Barfield Elementary	4.0	948.0	26.0	85.3	89.2	

Group data by school ratings, Chooses indicators that describe the student body (for example,reduced_lunch) or school administration (stu_teach_ratio) hoping they will explain school_rating. reduced_lunch is a variable measuring the average percentage of students per school enrolled in a federal program that provides lunches for students from lower-income households.

In short,reduced_lunchis a good proxy for household income. Isolates‘reduced_lunch’and groups the data by‘school_rating’using pandas groupby method and then uses describeon the re-shaped data

In [24]:
```
df_grouped_data = data.groupby("school_rating").describe()
df_grouped_data.head()
```

Out[24]:

	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 [16]:
```
# lets find out the school rating based on the school type

School_Rating_Type = data[['name','school_rating','school_type','reduced_lunc
h']]
School_Rating_Type
```

Out[16]:

	name	school_rating	school_type	reduced_lunch
0	Allendale Elementary School	5.0	Public	10.0
1	Anderson Elementary	2.0	Public	71.0
2	Avoca Elementary	4.0	Public	43.0
3	Bailey Middle	0.0	Public Magnet	91.0
4	Barfield Elementary	4.0	Public	26.0
...	...	...	...	...
342	Winfree Bryant Middle School	3.0	Public	57.0
343	Winstead Elementary School	5.0	Public	8.0
344	Woodland Elementary	4.0	Public	55.0
345	Woodland Middle School	5.0	Public	2.0
346	Wright Middle	0.0	Public	89.0

347 rows × 4 columns

lets findout the correlation between School Rating and Reduced Lunch

Correlation is a bi-variate analysis that measures the strength of association between two variables and the direction of the relationship.

In terms of the strength of relationship, the value of the correlation coefficient varies between +1 and -1. A value of ± 1 indicates a perfect degree of association between the two variables.

As the correlation coefficient value goes towards 0, the relationship between the two variables will be weaker. The direction of the relationship is indicated by the sign of the coefficient; a + sign indicates a positive relationship and a - sign indicates a negative relationship.

In [17]:
```
# Now use corr() function to find the correlation among the columns. We are only having two numeric columns in the dataframe.
# method :
# pearson : standard correlation coefficient
# kendall : Kendall Tau correlation coefficient
# spearman : Spearman rank correlation

# Pearson correlation coefficient is a measure of the strength of a linear association between two variables — denoted by r. You’ll come across Pearson r correlation

School_Rating_Type[['school_rating','reduced_lunch']].corr(method ='pearson')

# For the Pearson r correlation, both variables should be normally distributed. i.e the normal distribution describes how the values of a variable are distributed.
# This is sometimes called the ‘Bell Curve’ or the ‘Gaussian Curve’. A simple way to do this is to determine the normality of each variable separately using the Shapiro-Wilk Test.
```

Out[17]:

	school_rating	reduced_lunch
school_rating	1.000000	-0.815757
reduced_lunch	-0.815757	1.000000

In [18]:
```
School_Rating_Type[['school_rating','reduced_lunch']].corr(method ='kendall')
```

Out[18]:

	school_rating	reduced_lunch
school_rating	1.000000	-0.681755
reduced_lunch	-0.681755	1.000000

In [21]:
```
School_Rating_Type[['school_rating','reduced_lunch']].corr(method ='spearman')
```

Out[21]:

	school_rating	reduced_lunch
school_rating	1.000000	-0.821627
reduced_lunch	-0.821627	1.000000

In above example the correlation between two variable is same

In [27]:
```
df_grouped_data.head()
```

Out[27]:

	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

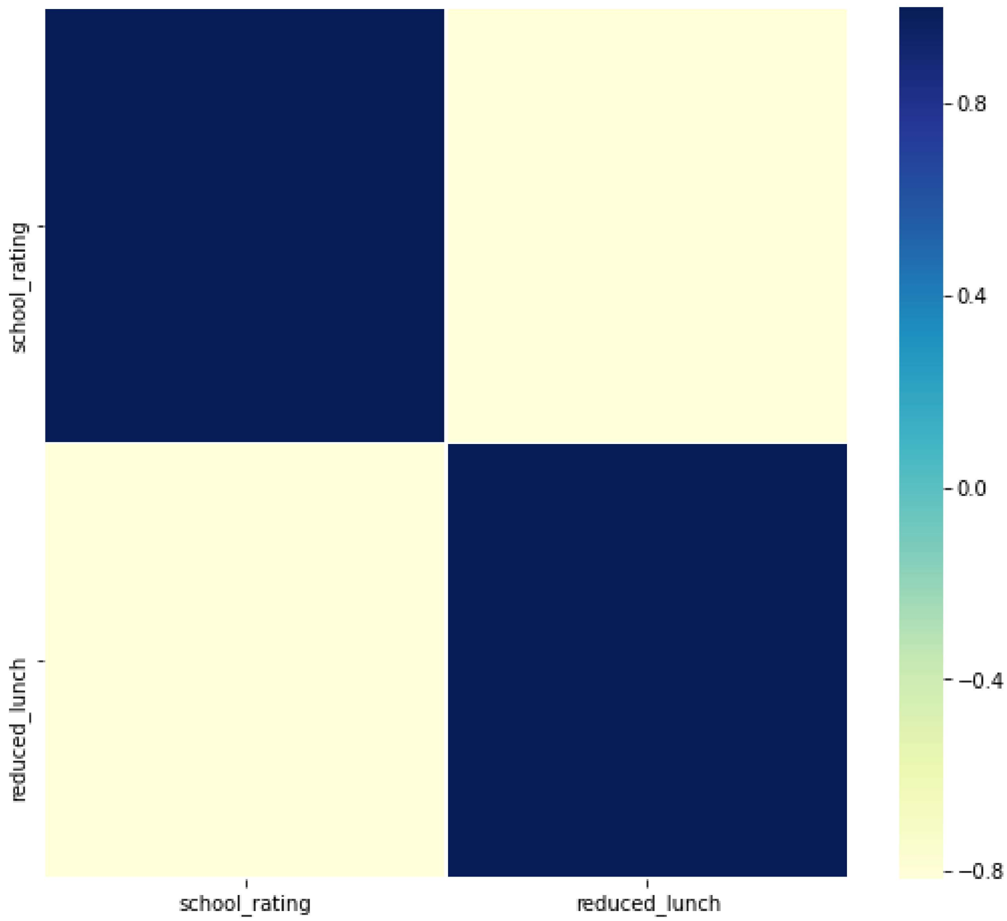
5 rows × 96 columns

In [28]:
```
import matplotlib.pyplot as plt
from scipy.stats import norm
import seaborn as sns
```

In [31]:
```
#heatmap

f, ax = plt.subplots(figsize=(9, 8))
sns.heatmap(School_Rating_Type[['school_rating','reduced_lunch']].corr(method
='pearson'), ax=ax, cmap="YlGnBu", linewidths=0.1)
```

Out[31]: `<matplotlib.axes._subplots.AxesSubplot at 0x2cd4e1b6748>`



Scatter PlotFind the relationship between school_rating and reduced_lunch, Plot a graph with the two variables on a scatter plot. Each dot represents a school. The placement of the dot represents that school's rating (Y-axis) and the percentageof its students on reduced lunch (x-axis). The downward trend line shows the negative correlation between school_rating and reduced_lunch(as one increases, the other decreases).

The slope of the trend line indicates how muchschool_ratingdecreases asreduced_lunchincreases. A steeper slope would indicate that a small change inreduced_lunchhas a big impact onschool_ratingwhile a more horizontal slope would indicate that the same small change inreduced_lunchhas a smaller impact onschool_rating

In [32]:
```
plt.scatter(data["school_rating"], data["reduced_lunch"])
plt.grid()
plt.xlabel("Rating")
plt.ylabel("Reduced lunch")
plt.title("School rating vs Reduced lunch")
plt.show()
```