Although the amount of data may appear large size, Python Pandas will help to successfully parse through it.

Required libraries are Imported:

```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
```

Data is imported using the following:

```
from google.colab import files
import io
data = files.upload()
```

```
school_df=pd.read_csv(io.StringIO(data['schools_complete.csv'].decode('utf-8')))
```

```
school_df.info
```

```
<bound method DataFrame.info of     School ID                   name      type  size
0           0      Huang High School  District  2917  1910635
1           1   Figueroa High School  District  2949  1884411
2           2    Shelton High School   Charter  1761  1056600
3           3  Hernandez High School  District  4635  3022020
4           4    Griffin High School   Charter  1468   917500
5           5     Wilson High School   Charter  2283  1319574
6           6    Cabrera High School   Charter  1858  1081356
7           7     Bailey High School  District  4976  3124928
8           8     Holden High School   Charter   427   248087
9           9       Pena High School   Charter   962   585858
10         10     Wright High School   Charter  1800  1049400
11         11  Rodriguez High School  District  3999  2547363
12         12    Johnson High School  District  4761  3094650
13         13       Ford High School  District  2739  1763916
14         14     Thomas High School   Charter  1635  1043130>
```

```
school_df.head()
```

	School ID	school name	type	size	budget
0	0	Huang High School	District	2917	1910635
1	1	Figueroa High School	District	2949	1884411
2	2	Shelton High School	Charter	1761	1056600
4	4	Griffin High School	Charter	1468	917500

```
student_df=pd.read_csv(io.StringIO(data['students_complete.csv'].decode('utf-8')))
```

```
student_df.info
```

```
<bound method DataFrame.info of        Student ID                name  ... reading_sco
0               0        Paul Bradley  ...            66          79
1               1        Victor Smith  ...            94          61
2               2     Kevin Rodriguez  ...            90          60
3               3  Dr. Richard Scott  ...            67          58
4               4          Bonnie Ray  ...            97          84
...           ...                 ...  ...           ...         ...
39165       39165        Donna Howard  ...            99          90
39166       39166           Dawn Bell  ...            95          70
39167       39167     Rebecca Tanner  ...            73          84
39168       39168        Desiree Kidd  ...            99          90
39169       39169     Carolyn Jackson  ...            95          75

[39170 rows x 7 columns]>
```

The name of the column in the data can be adjusted as needed:

```
# Rename col "name" in school_df to "school name"
school_df.rename({"name" : "school name"}, axis=1, inplace=True)
```

```
# Rename col "school" in student_df to "school name"
# Store renamed student_df as new df
renamed_student_df = student_df.rename(columns={"school" : "school name"})
```

A high-level snapshot (in table format) of the district's key measurements is generated below.

```
## Total Schools
total_schools = school_df["school name"].count()

## Total Students
total_students = renamed_student_df["school name"].count()

## Total Budget
total_budget = school_df["budget"].sum()
```

```
## Average Math Score
avg_math_score = renamed_student_df["math_score"].mean()

## Average Reading Score
avg_read_score = renamed_student_df["reading_score"].mean()


## % Passing Math based on 70
math_pass = renamed_student_df.loc[(student_df["math_score"] >= 70)]

count_pass_math = math_pass["math_score"].count()

per_math_pass = (count_pass_math/total_students)*100



## % Passing Reading based on 70
read_pass = renamed_student_df.loc[(student_df["reading_score"] >= 70)]

count_pass_read = read_pass["reading_score"].count()

per_read_pass = (count_pass_read/total_students)*100



## Overall Passing Rate (Average of the above two)
overall_pass = (per_math_pass + per_read_pass)/2



district_summary = {"Total Schools" : total_schools,
                   "Total Students" : total_students,
                   "Total Budget" : total_budget,
                   "Average Math Score" : avg_math_score,
                   "Average Reading Score" : avg_read_score,
                    "% Passing Math" : per_math_pass,
                   "% Passing Reading" : per_read_pass,
                   "% Overall Passing" : overall_pass
                  }


district_summary_df = pd.DataFrame([district_summary])

district_summary_df = district_summary_df[["Total Schools",
                   "Total Students","Total Budget","Average Math Score", "Average Reading
                   "% Passing Math", "% Passing Reading", "% Overall Passing" ]]
```

District_Summary

```
district_summary_df
```

```
        Total     Total     Total    Average   Average          %          %          %
                                       Math    Reading     Passing    Passing    Overall
```

An overview table summarizing the key criteria for each school is developed as follows:

```
# First, we will delete the School ID col from copy_school_sum because we will not need it
del copy_school_sum['School ID']

# We can print copy_school_sum to verify the del
# copy_school_sum


## Calculate the Per Student Budget
## Then, we will create a new col named 'Per Student Budget'

copy_school_sum['Per Student Budget'] = copy_school_sum['budget']/copy_school_sum['size']




## Average Math Score & Average Reading Score
## We will use a groupby function to group on school name and display both reading and mat

avg_math_read_tbl = renamed_student_df.groupby(['school name'])['reading_score', 'math_sco

# avg_math_read_tbl
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: FutureWarning: Indexi
  """
```

```
## Now, we merge avg_math_read_tbl with copy_school_sum df. We want to merge on School Nam

copy_school_sum = copy_school_sum.merge(avg_math_read_tbl, on='school name', how="outer")

#copy_school_sum


## We will use conditionals to find the % Passing Math and % Passing Reading. Then, we wil

# % Passing Reading
summary_passing_read = renamed_student_df[renamed_student_df['reading_score']>=70]

#% Passing Math
summary_passing_math = renamed_student_df[renamed_student_df['math_score']>=70]

#summary_passing_read
#summary_passing_math


## Count the number of students passing in reading
```

```
pass_read_count_sum = summary_passing_read.groupby(["school name"])['reading_score'].count

## Then, rename the column the 'reading_score' to 'Reading Count'
pass_read_count_sum.rename({'reading_score' : 'Reading Count'}, axis=1, inplace=True)


## Count the number of students passing in math
pass_math_count_sum = summary_passing_math.groupby(["school name"])['math_score'].count().

## Then, rename the column the 'math_score' to 'Math Count'
pass_math_count_sum.rename({'math_score' : 'Math Count'}, axis=1, inplace=True)


## Merge pass_math_count_sum with pass_read_count_sum
## We want to merge on School Name and by 'inner' to include only the contents found in bo

pass_count = pass_math_count_sum.merge(pass_read_count_sum, on="school name", how='inner')

#pass_count


## Merge copy_school_sum with pass_count we just created
## We want to merge on School Name and by 'outer' to include everything

copy_school_sum = copy_school_sum.merge(pass_count, on="school name", how='outer')

#copy_school_sum


## Calc % passing math and reading
## Take the subject count and divide by the school size, then pmultiply by 100 to get perc

# % Passing Math
copy_school_sum['% Passing Math'] = (copy_school_sum['Math Count']/copy_school_sum['size']

# % Passing Reading
copy_school_sum['% Passing Reading'] = (copy_school_sum['Reading Count']/copy_school_sum['

#copy_school_sum


# Now, we will delete the Math Count and Reading Count cols from copy_school_sum. Because

del copy_school_sum['Math Count']
del copy_school_sum['Reading Count']


## Calc % overall passing
## Overall Passing Rate (Average of the above two)

copy_school_sum['% Overall Passing'] = (copy_school_sum['% Passing Math'] + copy_school_su

#copy_school_sum



# now, rename axis for reading and math scores to Avg. Reading Score and Avg. Math Score i
```

```
copy_school_sum.rename({'reading_score':'Avg. Reading Score',
                             'math_score':'Avg. Math Score'}, axis=1, inplace=True)
```

```
copy_school_sum
```

	school name	type	size	budget	Per Student Budget	Avg. Reading Score	Avg. Math Score	% Passing Math	Pa Re
0	Huang High School	District	2917	1910635	655.0	81.182722	76.629414	65.683922	81.3
1	Figueroa High School	District	2949	1884411	639.0	81.158020	76.711767	65.988471	80.7
2	Shelton High School	Charter	1761	1056600	600.0	83.725724	83.359455	93.867121	95.8
3	Hernandez High School	District	4635	3022020	652.0	80.934412	77.289752	66.752967	80.8
4	Griffin High School	Charter	1468	917500	625.0	83.816757	83.351499	93.392371	97.1
5	Wilson High School	Charter	2283	1319574	578.0	83.989488	83.274201	93.867718	96.5
6	Cabrera High School	Charter	1858	1081356	582.0	83.975780	83.061895	94.133477	97.0

The following is a list of the 5 best performing schools in terms of overall pass rate:

```
## Create a table that highlights the top 5 performing schools based on Overall Passing Ra
## Found the top 5 performing by sorting copy_school_sum on the '% Overall Passing' col in
## By sorting, we can find the five top performing

top_performing_by_pr_df = copy_school_sum.sort_values(by=['% Overall Passing'], ascending=
```

```
top_performing_by_pr_df.head(5)
```

	school name	type	size	budget	Per Student Budget	Avg. Reading Score	Avg. Math Score	% Passing Math	Pass Reac
6	Cabrera High School	Charter	1858	1081356	582.0	83.975780	83.061895	94.133477	97.039

Bottom 5 performing schools

```
## Create a table that highlights the bottom 5 performing schools based on Overall Passing
## Using the copy_school_sum df found the bottom 5 performing schools based on Overall Pas

bottom_five_by_pr_df = copy_school_sum.sort_values(by=['% Overall Passing']).head(5)
```

```
bottom_five_by_pr_df
```

	school name	type	size	budget	Per Student Budget	Avg. Reading Score	Avg. Math Score	% Passing Math	Pas Rea
11	Rodriguez High School	District	3999	2547363	637.0	80.744686	76.842711	66.366592	80.22
1	Figueroa High School	District	2949	1884411	639.0	81.158020	76.711767	65.988471	80.73
0	Huang High School	District	2917	1910635	655.0	81.182722	76.629414	65.683922	81.31
	Johnson								

A table lists the average math score for each grade level (9th, 10th, 11th, 12th) students in each school:

```
# We will use the pivot table group to display the requested information
math_scores_by_grade_df = pd.pivot_table(student_df, values=['math_score'], index=['school
                                         columns=['grade'])
math_scores_by_grade_df = math_scores_by_grade_df.reindex(labels=['9th',
                                                                   '10th',
                                                                   '11th',
                                                                   '12th'], axis=1, level
```

```
math_scores_by_grade_df
```

	math_score			
grade	9th	10th	11th	12th
school				
Bailey High School	77.083676	76.996772	77.515588	76.492218
Cabrera High School	83.094697	83.154506	82.765560	83.277487
Figueroa High School	76.403037	76.539974	76.884344	77.151369
Ford High School	77.361345	77.672316	76.918058	76.179963
Griffin High School	82.044010	84.229064	83.842105	83.356164
Hernandez High School	77.438495	77.337408	77.136029	77.186567
Holden High School	83.787402	83.429825	85.000000	82.855422
Huang High School	77.027251	75.908735	76.446602	77.225641
Johnson High School	77.187857	76.691117	77.491653	76.863248
Pena High School	83.625455	83.372000	84.328125	84.121547
Rodriguez High School	76.859966	76.612500	76.395626	77.690748
Shelton High School	83.420755	82.917411	83.383495	83.778976
Thomas High School	83.590022	83.087886	83.498795	83.497041
Wilson High School	83.085578	83.724422	83.195326	83.035794
Wright High School	83.264706	84.010288	83.836782	83.644986

A table listing the average reading marks for each grade level (9th, 10th, 11th, 12th) students in each school is created as follows.

```
# will use the pivot table group to display
read_scores_by_grade_df = pd.pivot_table(student_df, values=['reading_score'], index=['sch
                                         columns=['grade'])
read_scores_by_grade_df = read_scores_by_grade_df.reindex(labels=['9th',
                                                                    '10th',
                                                                    '11th',
                                                                    '12th'], axis=1, level


read_scores_by_grade_df
```

	reading_score			
grade	9th	10th	11th	12th
school				
Bailey High School	81.303155	80.907183	80.945643	80.912451
Cabrera High School	83.676136	84.253219	83.788382	84.287958
Figueroa High School	81.198598	81.408912	80.640339	81.384863
Ford High School	80.632653	81.262712	80.403642	80.662338
Griffin High School	83.369193	83.706897	84.288089	84.013699
Hernandez High School	80.866860	80.660147	81.396140	80.857143
Holden High School	83.677165	83.324561	83.815534	84.698795
Huang High School	81.290284	81.512386	81.417476	80.305983
Johnson High School	81.260714	80.773431	80.616027	81.227564
Pena High School	83.807273	83.612000	84.335938	84.591160
Rodriguez High School	80.993127	80.629808	80.864811	80.376426
Shelton High School	84.122642	83.441964	84.373786	82.781671

A schedule of breaking down school programs (for each student) is created based on average spending limits. 4 reasonable bins are used to spend on the school board. The table is being created.

```
# Copy the copy_school_sum and save as scores_by_school_spending
scores_by_school_spending = copy_school_sum.copy()


# Create bins - we will need labels and bins
bins = [0, 585, 615, 645, 675]
spending_labels = ['$0-585', '$586-615', '$616-645', '$646-675']


# Use bins and labels to sort through data and divide it up appropriately
# save bined data as bins_school_spending variable
bins_school_spending = pd.cut(scores_by_school_spending['Per Student Budget'], bins, label

# Convert bins_school_spending to df
bins_school_spending = pd.DataFrame(bins_school_spending)

# add Spending Level col
copy_school_sum['Spending Level'] = bins_school_spending
```

```
# Show cols for bins_school_spending to verify
#bins_school_spending.columns
```

```
# Do a groupby on Spending Level and school name
scores_by_school_spending = copy_school_sum.groupby(['Spending Level'])['Avg. Reading Scor
                                                          'Avg. Math Score',
                                                          '% Passing Reading',
                                                          '% Passing Math',
                                                          '% Overall Passing'
                                                          ].mean()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: FutureWarning: Indexi
```

```
scores_by_school_spending
```

	Avg. Reading Score	Avg. Math Score	% Passing Reading	% Passing Math	% Overall Passing
Spending Level					
$0-585	83.933814	83.455399	96.610877	93.460096	95.035486
$586-615	83.885211	83.599686	95.900287	94.230858	95.065572
$616-645	81.891436	79.079225	86.106569	75.668212	80.887391

Group schools based on a reasonable approximation of school size (Small, Medium, Large).

```
# Create a copy of copy_school_sum and save as scores_by_school_size
scores_by_school_size = copy_school_sum.copy()

# Print scores_by_school_size to verify
# scores_by_school_size
```

```
# Create bins - we will need labels and bins
bins = [0, 1000, 2000, 5000]
size_labels = ['Small', 'Medium', 'Large']
```

```
# Use bins and labels to sort through data and divide it up appropriately
# save bin data as bins_school_size variable
bins_school_size = pd.cut(scores_by_school_size['size'], bins, labels = size_labels)

# Convert bins_school_spending to df
bins_school_size = pd.DataFrame(bins_school_size)
```

```
# add 'School Population' col
copy_school_sum['School Population'] = bins_school_size
```

```
# Show cols for bins_school_size to verify
# bins_school_size.columns
# Do a groupby on School Population and school name
scores_by_school_size = copy_school_sum.groupby(['School Population','school name'])['Avg.
                                                         'Avg. Math Score',
                                                         '% Passing Reading',
                                                         '% Passing Math',
                                                         '% Overall Passing'
                                                         ].mean()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: FutureWarning: Indexi
  after removing the cwd from sys.path.
```

```
scores_by_school_size
```

		Avg. Reading Score	Avg. Math Score	% Passing Reading	% Passing Math	% Overall Passing
School Population	school name					
Large	Bailey High School	81.033963	77.048432	81.933280	66.680064	74.306672
	Figueroa High School	81.158020	76.711767	80.739234	65.988471	73.363852
	Ford High School	80.746258	77.102592	79.299014	68.309602	73.804308
	Hernandez High School	80.934412	77.289752	80.862999	66.752967	73.807983
	Huang High School	81.182722	76.629414	81.316421	65.683922	73.500171
	Johnson High School	80.966394	77.072464	81.222432	66.057551	73.639992
	Rodriguez High School	80.744686	76.842711	80.220055	66.366592	73.293323
	Wilson High School	83.989488	83.274201	96.539641	93.867718	95.203679
Medium	Cabrera High School	83.975780	83.061895	97.039828	94.133477	95.586652

Group schools based on school type (Charter vs. District).

```
# Create a copy of copy_school_sum and save as scores_by_school_type
scores_by_school_type = copy_school_sum.copy()

# Convert scores_by_school_type to df
scores_by_school_type = pd.DataFrame(scores_by_school_type)

# Print scores_by_school_type cols to verify
# scores_by_school_type.columns
```

```
# Do a groupby on School Type
scores_by_school_type = copy_school_sum.groupby(['type'])['Avg. Reading Score',
                                                          'Avg. Math Score',
                                                          '% Passing Reading',
                                                          '% Passing Math',
                                                          '% Overall Passing'
                                                          ].mean()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: FutureWarning: Indexi
```

```
scores_by_school_type.head()
```

	Avg. Reading Score	Avg. Math Score	% Passing Reading	% Passing Math	% Overall Passing
type					
Charter	83.896421	83.473852	96.586489	93.620830	95.103660
District	80.966636	76.956733	80.799062	66.548453	73.673757

Observations:

- **We can observe that charter schools have greater pass rates in reading and math than district schools when we look at Scores by School Type.**
- **The fact that the top five performing schools had all been charter schools adds to this.**
- **When comparing Math and Reading Scores by Grade, we can see that pupils' math scores are generally lower than their reading scores.**
- **We can see from the Results by School Spending that there isn't always a link between school expenditure and overall math or reading scores.**