Note

- Instructions have been included for each segment. You do not have to follow them exactly, but they are included to help you think through the steps.

In [668]:
```
# Dependencies and Setup
import pandas as pd

# File to Load (Remember to Change These)
school_data_to_load = "Resources/schools_complete.csv"
student_data_to_load = "Resources/students_complete.csv"

# Read School and Student Data File and store into Pandas DataFrames
school_data = pd.read_csv(school_data_to_load)
student_data = pd.read_csv(student_data_to_load)

# Combine the data into a single dataset.
school_data_complete = pd.merge(student_data, school_data, how="left", o
n=["school_name", "school_name"])
school_data_complete.head()
```

Out[668]:

	Student ID	student_name	gender	grade	school_name	reading_score	math_score	School ID	t
0	0	Paul Bradley	M	9th	Huang High School	66	79	0	Dis
1	1	Victor Smith	M	12th	Huang High School	94	61	0	Dis
2	2	Kevin Rodriguez	M	12th	Huang High School	90	60	0	Dis
3	3	Dr. Richard Scott	M	12th	Huang High School	67	58	0	Dis
4	4	Bonnie Ray	F	9th	Huang High School	97	84	0	Dis

District Summary

- Calculate the total number of schools
- Calculate the total number of students
- Calculate the total budget
- Calculate the average math score
- Calculate the average reading score
- Calculate the percentage of students with a passing math score (70 or greater)
- Calculate the percentage of students with a passing reading score (70 or greater)
- Calculate the percentage of students who passed math **and** reading (% Overall Passing)
- Create a dataframe to hold the above results
- Optional: give the displayed data cleaner formatting

In [684]:
```
district_totals = school_data_complete.nunique()
district_totals.head(11)
```

Out[684]:
```
Student ID       39170
student_name     32715
gender               2
grade                4
school_name         15
reading_score       37
math_score          45
School ID           15
type                 2
size                15
budget              15
dtype: int64
```

In [685]:
```
# Check for any null values
# school_data_complete.isnull().sum().sum()
```

In [686]:

```
# Calculate the total number of schools
total_schools = district_totals["School ID"]
#total_schools

# Calculate the total number of students
total_students = district_totals["Student ID"]
#total_students

# Calculate the total budget
total_budget = school_data["budget"].sum()
#total_budget

# Calculate the average math score
avg_math = student_data["math_score"].mean()
#avg_math

# Calculate the average reading score
avg_read = student_data["reading_score"].mean()
#avg_read


# Calculate the percentage of students with a passing math score (70 or
 greater)
math_pass = len(school_data_complete.loc[school_data_complete["math_scor
e"]>=70])
math_pass_percent = (math_pass/total_students) *100
#math_pass_percent

# Calculate the percentage of students with a passing reading score (70
 or greater)
read_pass = len(school_data_complete.loc[school_data_complete["reading_s
core"]>=70])
read_pass_percent = (read_pass/total_students) *100
#read_pass_percent

# Calculate the percentage of students who passed math and reading (% Ov
erall Passing)
both_pass = len(school_data_complete.loc[(school_data_complete["math_sco
re"]>=70) & (school_data_complete["reading_score"]>=70)])
both_percent = (both_pass/total_students) *100
#both_percent
```

In [687]:
```
#Create a dataframe to hold the above results
district_summary =pd.DataFrame({"Total Schools": [total_schools],
                                "Total Student": [total_students],
                                "Total Budget": [total_budget],
                                "Average Math Score": [avg_math],
                                "Average Reading Score": [avg_read],
                                "% Passing Math": [math_pass_percent],
                                "% Passing Reading": [read_pass_percent
],
                                "% Overall Passing": [both_percent]})

# Optional: give the displayed data cleaner formatting
district_summary["Total Student"] = district_summary["Total Student"].ma
p("{:,}".format)
district_summary["Total Budget"] = district_summary["Total Budget"].map(
"${:,.2f}".format)
district_summary
```

Out[687]:

	Total Schools	Total Student	Total Budget	Average Math Score	Average Reading Score	% Passing Math	% Passing Reading	% Overall Passing
0	15	39,170	$24,649,428.00	78.985371	81.87784	74.980853	85.805463	65.172326

School Summary

- Create an overview table that summarizes key metrics about each school, including:
  - School Name
  - School Type
  - Total Students
  - Total School Budget
  - Per Student Budget
  - Average Math Score
  - Average Reading Score
  - % Passing Math
  - % Passing Reading
  - % Overall Passing (The percentage of students that passed math **and** reading.)
- Create a dataframe to hold the above results

In [688]:

```
# School Name and Type
school_type = school_data.set_index(["school_name"])["type"]

# Group by and merge data
school_total = school_data_complete.groupby(["school_name"]).mean()
school_total = pd.merge(school_type, school_total, on="school_name")


# Total Students per school
total_stu_per_school = school_total["size"]


# Total School Budget per school
school_budget = school_total["budget"]


# Per Student Budget per school
student_budget = school_total["budget"]/school_total["size"]


# Average Math Score per school
math_school_avg = school_total["math_score"]


# Average Reading Score per school
read_school_avg = school_total["reading_score"]


# % Passing Math
math_schools = school_data_complete.loc[school_data_complete["math_scor
e"]>=70, ["school_name","math_score"]]
math_schools = math_schools.groupby(["school_name"]).count()


# % Passing Reading
read_schools = school_data_complete.loc[school_data_complete["reading_sc
ore"]>=70, ["school_name","reading_score"]]
read_schools = read_schools.groupby(["school_name"]).count()


# % Overall Passing (The percentage of students that passed math and rea
ding.)
overall_schools = school_data_complete.loc[(school_data_complete["math_s
core"]>=70) & (school_data_complete["reading_score"]>=70), ["school_nam
e", "reading_score"]]
overall_schools = overall_schools.groupby(["school_name"]).count()


# Add new columns to DataFrame
school_total["Per Student Budget"] = school_total["budget"]/school_total
["size"]
school_total["% Passing Math"] = (math_schools["math_score"]/total_stu_p
er_school)*100
school_total["% Passing Reading"] = (read_schools["reading_score"]/total
_stu_per_school)*100
school_total["% Overall Passing"] = (overall_schools["reading_score"]/to
```

```
tal_stu_per_school)*100

# Remove unwanted columns
school_total.pop("Student ID")
school_total.pop("School ID")

# Sort school names alphabetically
school_results = school_total.sort_values(["school_name"], ascending=Tru
e)
```

In [689]:

```
# Create a dataframe to hold the above results

# Reorganize the columns
school_summary = school_results[["type",
                                 "size",
                                 "budget",
                                 "Per Student Budget",
                                 "math_score",
                                 "reading_score",
                                 "% Passing Math",
                                 "% Passing Reading",
                                 "% Overall Passing"]]



# Rename the columns
school_summary = school_summary.rename(columns={"type":"School Type",
                                                "size":"Total Students",
                                                "budget":"Total School B
udget",
                                                "math_score":"Average Ma
th Score",
                                                "reading_score":"Average
Reading Score"})



# Remove the index header
school_summary.index.name = None


# Give the displayed data cleaner formatting
school_summary["Total Students"] = school_summary["Total Students"].map(
"{:.0f}".format)
school_summary["Total School Budget"] = school_summary["Total School Bud
get"].map("${:,.2f}".format)
school_summary["Per Student Budget"] = school_summary["Per Student Budge
t"].map("${:.2f}".format)

# Display DataFrame
school_summary
```

Out[689]:

	School Type	Total Students	Total School Budget	Per Student Budget	Average Math Score	Average Reading Score	% Passing Math	% Passing Reading
Bailey High School	District	4976	$3,124,928.00	$628.00	77.048432	81.033963	66.680064	81.933280
Cabrera High School	Charter	1858	$1,081,356.00	$582.00	83.061895	83.975780	94.133477	97.039828
Figueroa High School	District	2949	$1,884,411.00	$639.00	76.711767	81.158020	65.988471	80.739234
Ford High School	District	2739	$1,763,916.00	$644.00	77.102592	80.746258	68.309602	79.299014
Griffin High School	Charter	1468	$917,500.00	$625.00	83.351499	83.816757	93.392371	97.138965
Hernandez High School	District	4635	$3,022,020.00	$652.00	77.289752	80.934412	66.752967	80.862999
Holden High School	Charter	427	$248,087.00	$581.00	83.803279	83.814988	92.505855	96.252927
Huang High School	District	2917	$1,910,635.00	$655.00	76.629414	81.182722	65.683922	81.316421
Johnson High School	District	4761	$3,094,650.00	$650.00	77.072464	80.966394	66.057551	81.222432
Pena High School	Charter	962	$585,858.00	$609.00	83.839917	84.044699	94.594595	95.945946
Rodriguez High School	District	3999	$2,547,363.00	$637.00	76.842711	80.744686	66.366592	80.220055
Shelton High School	Charter	1761	$1,056,600.00	$600.00	83.359455	83.725724	93.867121	95.854628
Thomas High School	Charter	1635	$1,043,130.00	$638.00	83.418349	83.848930	93.272171	97.308869
Wilson High School	Charter	2283	$1,319,574.00	$578.00	83.274201	83.989488	93.867718	96.539641
Wright High School	Charter	1800	$1,049,400.00	$583.00	83.682222	83.955000	93.333333	96.611111

In [ ]:

Top Performing Schools (By % Overall Passing)

- Sort and display the top five performing schools by % overall passing.

In [676]:
```
# Sort and display top five performing schools by % overall passing

top_five = school_summary.sort_values(["% Overall Passing"], ascending=False)
top_five.head(5)
```

Out[676]:

	School Type	Total Students	Total School Budget	Per Student Budget	Average Math Score	Average Reading Score	% Passing Math	% Passing Reading	
Cabrera High School	Charter	1858	$1,081,356.00	$582.00	83.061895	83.975780	94.133477	97.039828	9
Thomas High School	Charter	1635	$1,043,130.00	$638.00	83.418349	83.848930	93.272171	97.308869	9
Griffin High School	Charter	1468	$917,500.00	$625.00	83.351499	83.816757	93.392371	97.138965	9
Wilson High School	Charter	2283	$1,319,574.00	$578.00	83.274201	83.989488	93.867718	96.539641	9
Pena High School	Charter	962	$585,858.00	$609.00	83.839917	84.044699	94.594595	95.945946	9

In [ ]:
```

```

Bottom Performing Schools (By % Overall Passing)

- Sort and display the five worst-performing schools by % overall passing.

In [677]:
```
# Sort and display the five worst-performing schools by % overall passin
g

bottom_five = school_summary.sort_values(["% Overall Passing"], ascendin
g=True)
bottom_five.head(5)
```

Out[677]:

	School Type	Total Students	Total School Budget	Per Student Budget	Average Math Score	Average Reading Score	% Passing Math	% Passing Reading
Rodriguez High School	District	3999	$2,547,363.00	$637.00	76.842711	80.744686	66.366592	80.220055
Figueroa High School	District	2949	$1,884,411.00	$639.00	76.711767	81.158020	65.988471	80.739234
Huang High School	District	2917	$1,910,635.00	$655.00	76.629414	81.182722	65.683922	81.316421
Hernandez High School	District	4635	$3,022,020.00	$652.00	77.289752	80.934412	66.752967	80.862999
Johnson High School	District	4761	$3,094,650.00	$650.00	77.072464	80.966394	66.057551	81.222432

Math Scores by Grade

- Create a table that lists the average Math Score for students of each grade level (9th, 10th, 11th, 12th) at each school.
  - Create a pandas series for each grade. Hint: use a conditional statement.
  - Group each series by school
  - Combine the series into a dataframe
  - Optional: give the displayed data cleaner formatting

In [690]:

```
# Create a series for each grade
grades = school_data_complete.set_index(["school_name"])["grade"]
grades

math_ninth = school_data_complete.loc[school_data_complete["grade"]=="9th", ["school_name","math_score"]]
math_ninth_avg = math_ninth.groupby(["school_name"]).mean()
math_ninth_avg

math_tenth = school_data_complete.loc[school_data_complete["grade"]=="10th", ["school_name","math_score"]]
math_tenth_avg = math_tenth.groupby(["school_name"]).mean()
math_tenth_avg

math_eleventh = school_data_complete.loc[school_data_complete["grade"]=="11th", ["school_name","math_score"]]
math_eleventh_avg = math_eleventh.groupby(["school_name"]).mean()
math_eleventh_avg

math_twelfth = school_data_complete.loc[school_data_complete["grade"]=="12th", ["school_name","math_score"]]
math_twelfth_avg = math_twelfth.groupby(["school_name"]).mean()
math_twelfth_avg


# Add new columns to DataFrame
math_ninth_avg["10th"] = math_tenth_avg
math_ninth_avg["11th"] = math_eleventh_avg
math_ninth_avg["12th"] = math_twelfth_avg
math_ninth_avg

# Rename the DataFrame and columns
math_by_grade = math_ninth_avg.rename(columns={"math_score":"9th"})
math_by_grade

# Remove the index header
math_by_grade.index.name = None
math_by_grade
```

Out[690]:

	9th	10th	11th	12th
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

In [ ]:

Reading Score by Grade

- Perform the same operations as above for reading scores

In [691]:

```
# Create a series for each grade
grades = school_data_complete.set_index(["school_name"])["grade"]
grades

read_ninth = school_data_complete.loc[school_data_complete["grade"]=="9th", ["school_name","reading_score"]]
read_ninth_avg = read_ninth.groupby(["school_name"]).mean()
read_ninth_avg

read_tenth = school_data_complete.loc[school_data_complete["grade"]=="10th", ["school_name","reading_score"]]
read_tenth_avg = read_tenth.groupby(["school_name"]).mean()
read_tenth_avg

read_eleventh = school_data_complete.loc[school_data_complete["grade"]=="11th", ["school_name","reading_score"]]
read_eleventh_avg = read_eleventh.groupby(["school_name"]).mean()
read_eleventh_avg

read_twelfth = school_data_complete.loc[school_data_complete["grade"]=="12th", ["school_name","reading_score"]]
read_twelfth_avg = read_twelfth.groupby(["school_name"]).mean()
read_twelfth_avg


# Add new columns to DataFrame
read_ninth_avg["10th"] = read_tenth_avg
read_ninth_avg["11th"] = read_eleventh_avg
read_ninth_avg["12th"] = read_twelfth_avg
read_ninth_avg

# Rename the DataFrame and columns
read_by_grade = read_ninth_avg.rename(columns={"reading_score":"9th"})
read_by_grade

# Remove the index header
read_by_grade.index.name = None
read_by_grade
```

Out[691]:

	9th	10th	11th	12th
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
Thomas High School	83.728850	84.254157	83.585542	83.831361
Wilson High School	83.939778	84.021452	83.764608	84.317673
Wright High School	83.833333	83.812757	84.156322	84.073171

In [ ]:

Scores by School Spending

- Create a table that breaks down school performances based on average Spending Ranges (Per Student). Use 4 reasonable bins to group school spending. Include in the table each of the following:
  - Average Math Score
  - Average Reading Score
  - % Passing Math
  - % Passing Reading
  - Overall Passing Rate (Average of the above two)

In [692]:

```
# Create the bins in which data will be held
bins = [0, 584.9, 629.9, 644.9, 675]


# Create the names for the four bins
group_names = ["<$584", "$585-629", "$630-644", "$645-675"]


# Cut the bins
school_total["Spending Ranges (Per Student)"] = pd.cut(school_total["Per
Student Budget"], bins, labels=group_names, include_lowest=True)


# Groupby spending ranges
school_spending_group = school_total.groupby(["Spending Ranges (Per Stud
ent)"]).mean()
school_spending_group


# Reorganize the columns
school_spending = school_spending_group[["math_score",
                                         "reading_score",
                                         "% Passing Math",
                                         "% Passing Reading",
                                         "% Overall Passing"]]


# Rename the columns
school_spending = school_spending.rename(columns={"math_score":"Average
 Math Score",
                                                   "reading_score":"Average
Reading Score"})

# Give the displayed data cleaner formatting
school_spending["Average Math Score"] = school_spending["Average Math Sc
ore"].map("{:.2f}".format)
school_spending["Average Reading Score"] = school_spending["Average Read
ing Score"].map("{:.2f}".format)
school_spending["% Passing Math"] = school_spending["% Passing Math"].ma
p("{:.2f}".format)
school_spending["% Passing Reading"] = school_spending["% Passing Readin
g"].map("{:.2f}".format)
school_spending["% Overall Passing"] = school_spending["% Overall Passin
g"].map("{:.2f}".format)

school_spending
```

Out[692]:

Spending Ranges (Per Student)	Average Math Score	Average Reading Score	% Passing Math	% Passing Reading	% Overall Passing
<$584	83.46	83.93	93.46	96.61	90.37
$585-629	81.90	83.16	87.13	92.72	81.42
$630-644	78.52	81.62	73.48	84.39	62.86
$645-675	77.00	81.03	66.16	81.13	53.53

In [ ]:

Scores by School Size

- Perform the same operations as above, based on school size.

In [693]:

```
# Create the bins in which data will be held
bins = [0, 999.9, 1999.9, 4999.9]


# Create the names for the three bins
group_names = ["Small(<1000)", "Medium (1000-2000)", "Large(2000-5000)"]


# Cut the bins
school_total["School Size"] = pd.cut(school_total["size"], bins, labels=
group_names, include_lowest=True)


# Groupby size ranges
school_size_group = school_total.groupby(["School Size"]).mean()
school_size_group


# Reorganize the columns
school_size = school_size_group[["math_score",
                                 "reading_score",
                                 "% Passing Math",
                                 "% Passing Reading",
                                 "% Overall Passing"]]



# Rename the columns
school_size = school_size.rename(columns={"math_score":"Average Math Sco
re",
                                          "reading_score":"Average Readi
ng Score"})

# Give the displayed data cleaner formatting
school_size["Average Math Score"] = school_size["Average Math Score"].ma
p("{:.2f}".format)
school_size["Average Reading Score"] = school_size["Average Reading Scor
e"].map("{:.2f}".format)
school_size["% Passing Math"] = school_size["% Passing Math"].map("{:.2
f}".format)
school_size["% Passing Reading"] = school_size["% Passing Reading"].map(
"{:.2f}".format)
school_size["% Overall Passing"] = school_size["% Overall Passing"].map(
"{:.2f}".format)

school_size
```

Out[693]:

School Size	Average Math Score	Average Reading Score	% Passing Math	% Passing Reading	% Overall Passing
Small(<1000)	83.82	83.93	93.55	96.10	89.88
Medium (1000-2000)	83.37	83.86	93.60	96.79	90.62
Large(2000-5000)	77.75	81.34	69.96	82.77	58.29

Scores by School Type

- Perform the same operations as above, based on school type

In [694]:
```
school_total = school_total.rename(columns={"type": "School Type"})

# Groupby school type
school_type_group = school_total.groupby(["School Type"]).mean()
school_type_group

# Reorganize the columns
school_type = school_type_group[["math_score",
                                 "reading_score",
                                 "% Passing Math",
                                 "% Passing Reading",
                                 "% Overall Passing"]]

# Rename the columns
school_type = school_type.rename(columns={"math_score":"Average Math Sco
re",
                                          "reading_score":"Average Readi
ng Score"})


school_type
```

Out[694]:

School Type	Average Math Score	Average Reading Score	% Passing Math	% Passing Reading	% Overall Passing
Charter	83.473852	83.896421	93.620830	96.586489	90.432244
District	76.956733	80.966636	66.548453	80.799062	53.672208

In [ ]: