PyCity Schools Analysis

- As a whole, schools with higher budgets, did not yield better test results. By contrast, schools with higher spending per student actually (\$645 - 675) underperformed compared to schools with smaller budgets (\$585 per student).
- As a whole, smaller and medium sized schools dramatically out-performed large sized schools on passing math performances (89-91% passing vs 67%).
- As a whole, charter schools out-performed the public district schools across all metrics. However, more analysis will be required to glean if the effect is due to school practices or the fact that charter schools tend to serve smaller student populations per school.

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 [35]:
```
# Dependencies and Setup
import pandas as pd
import numpy as np
import sqlite3
import os
```

In [36]:
```
current_directory = os.getcwd()
print(current_directory)
```

```
/Users/jayhawkj/Desktop/Git_Repo/DS311-Technologies-in-Data-Analytic-SP
25/Week_04_Pandas_and_Matplotlib/Lab_Assignment
```

In [37]:
```
con = sqlite3.connect(r'./data/python.db')

school_data = pd.read_sql_query("SELECT * FROM school", con)
school_data
#student_data = pd.read_sql_query("SELECT * FROM student_data", con)
```

Out[37]:

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

In [38]:
```
# join the two tables into a single dataframe
complete_df = pd.merge(student_data, school_data, how="left", on="scho
complete_df.head()
```

Out[38]:

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

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 overall passing rate (overall average score), i.e. (avg. math score + avg. reading score)/2
- 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)
- Create a dataframe to hold the above results
- Optional: give the displayed data cleaner formatting

In [39]:
```
# Create a District Summary

# 1. Calculate the total number of unique schools
total_schools = len(complete_df["school_name"].unique())

# 2. Calculate the total number of students
total_students = len(complete_df["student_name"])

# 3. Calculate the total budget
#    We drop duplicates on "school_name" to avoid summing each school'
total_budget = complete_df.drop_duplicates(subset="school_name")["budg

# 4. Calculate the average math score
avg_math_score = complete_df["math_score"].mean()

# 5. Calculate the average reading score
avg_reading_score = complete_df["reading_score"].mean()

# 6. Calculate the percentage of students passing math (score >= 70)
passing_math = complete_df[complete_df["math_score"] >= 70]
passing_math_percent = (len(passing_math) / total_students) * 100

# 7. Calculate the percentage of students passing reading (score >= 70
passing_reading = complete_df[complete_df["reading_score"] >= 70]
```

```
passing_reading_percent = (len(passing_reading) / total_students) * 10

# 8. Calculate the overall passing percentage (students passing both m
passing_both = complete_df[
    (complete_df["math_score"] >= 70) & (complete_df["reading_score"]
]
passing_both_percent = (len(passing_both) / total_students) * 100

# 9. Create a summary DataFrame
district_summary = pd.DataFrame({
    "Total Schools": [total_schools],
    "Total Students": [total_students],
    "Total Budget": [total_budget],
    "Average Math Score": [avg_math_score],
    "Average Reading Score": [avg_reading_score],
    "% Passing Math": [passing_math_percent],
    "% Passing Reading": [passing_reading_percent],
    "% Overall Passing": [passing_both_percent]
})

# 10. (Optional) Format the displayed data for readability
district_summary["Total Students"] = district_summary["Total Students"
district_summary["Total Budget"] = district_summary["Total Budget"].ma

# Display the District Summary
district_summary
```

Out[39]:

	Total Schools	Total Students	Total Budget	Average Math Score	Average Reading Score	% Passing Math	% Passi Readi
0	15	39,170	$24,649,428.00	78.985371	81.87784	74.980853	85.8054

In [40]:

```
# Total number of schools

# Calculate the total number of unique schools
total_schools = len(complete_df["school_name"].unique())

# Print the result
print("Total number of schools:", total_schools)
```

```
Total number of schools: 15
```

In [41]:

```
# Total number of students

total_students = complete_df["student_name"].count()

# Print the total number of students with commas as thousand separator
print(f"Total number of students: {total_students:,}")
```

```
Total number of students: 39,170
```

In [42]:
```
# Total budget

# Drop duplicate rows based on 'school_name' to get each school's budg
unique_schools = complete_df.drop_duplicates(subset="school_name")

# Sum the 'budget' column from the unique schools DataFrame
total_budget = unique_schools["budget"].sum()

# Print the total budget formatted as currency
print(f"Total budget: ${total_budget:,.2f}")
```

```
Total budget: $24,649,428.00
```

In [43]:
```
# Average math score

# Calculate the average math score
avg_math_score = complete_df["math_score"].mean()

# Print the result, formatted as a percentage with two decimal places
print(f"Average math score: {avg_math_score:.2f}%")
```

```
Average math score: 78.99%
```

In [44]:
```
# Average reading score
# Calculate the average reading score
avg_reading_score = complete_df["reading_score"].mean()

# Print the result formatted as a percentage with two decimal places
print(f"Average reading score: {avg_reading_score:.2f}%")
```

```
Average reading score: 81.88%
```

In [45]:
```
# Overall average score

# Calculate the overall average score for each student
overall_student_avg = (complete_df["math_score"] + complete_df["readin

# Calculate the overall average across all students
overall_avg_score = overall_student_avg.mean()

# Print the overall average score formatted as a percentage with two d
print(f"Overall average score: {overall_avg_score:.2f}%")
```

```
Overall average score: 80.43%
```

In [46]:
```
# Percentage of passing math (70 or greater)

# Calculate the total number of students
total_students = complete_df.shape[0]

# Calculate the number of students with a math score of 70 or greater
passing_math_count = complete_df[complete_df["math_score"] >= 70].shap

# Calculate the percentage of students passing math
```

```
passing_math_percentage = (passing_math_count / total_students) * 100

# Print the percentage formatted to two decimal places with a "%" sign
print(f"Percentage of passing math: {passing_math_percentage:.2f}%")
```

```
Percentage of passing math: 74.98%
```

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 Rate (Average of the above two)
- Create a dataframe to hold the above results

Top Performing Schools (By Passing Rate)

- Sort and display the top five schools in overall passing rate

In [48]:
```
import pandas as pd

# Group the merged DataFrame by school_name
school_groups = complete_df.groupby("school_name")

# 1) School Type (Charter or District)
school_type = school_groups["type"].first()

# 2) Total Students (count of student rows per school)
total_students = school_groups["Student ID"].count()

# 3) Total School Budget (each school has one budget value, so we can
total_school_budget = school_groups["budget"].first()

# 4) Per Student Budget
per_student_budget = total_school_budget / total_students
```

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

# 6) Average Reading Score
avg_reading_score = school_groups["reading_score"].mean()
```

In [49]:
```
# Filter the data for passing math and passing reading
passing_math_df = complete_df[complete_df["math_score"] >= 70]
passing_reading_df = complete_df[complete_df["reading_score"] >= 70]

# Count how many passed math/reading in each school
passing_math_by_school = passing_math_df.groupby("school_name")["Stude
passing_reading_by_school = passing_reading_df.groupby("school_name")[

# Percentage passing math & reading
percent_passing_math = (passing_math_by_school / total_students) * 100
percent_passing_reading = (passing_reading_by_school / total_students)

# Overall Passing Rate (Option A: average of the two percentages)
overall_passing_rate = (percent_passing_math + percent_passing_reading

# If your assignment wants "overall passing" to mean "passing both mat
# passing_both_df = complete_df[
#     (complete_df["math_score"] >= 70) & (complete_df["reading_score"
# ]
# passing_both_by_school = passing_both_df.groupby("school_name")["Stu
# overall_passing_rate = (passing_both_by_school / total_students) * 1
```

In [51]:
```
school_summary = pd.DataFrame({
    "School Type": school_type,
    "Total Students": total_students,
    "Total School Budget": total_school_budget,
    "Per Student Budget": per_student_budget,
    "Average Math Score": avg_math_score,
    "Average Reading Score": avg_reading_score,
    "% Passing Math": percent_passing_math,
    "% Passing Reading": percent_passing_reading,
    "% Overall Passing": overall_passing_rate
})

school_summary.head()
```

Out[51]:

school_name	School Type	Total Students	Total School Budget	Per Student Budget	Average Math Score	Average Reading Score	% P
Bailey High School	District	4976	3124928	628.0	77.048432	81.033963	66.6
Cabrera High School	Charter	1858	1081356	582.0	83.061895	83.975780	94.1
Figueroa High School	District	2949	1884411	639.0	76.711767	81.158020	65.9
Ford High School	District	2739	1763916	644.0	77.102592	80.746258	68.3
Griffin High School	Charter	1468	917500	625.0	83.351499	83.816757	93.3

In [52]:
```
#  Sort and display the top five schools in overall passing rate

# Sort by % Overall Passing in descending order, then take the top 5
top_five_schools = school_summary.sort_values(
    by="% Overall Passing", ascending=False
).head(5)

top_five_schools
```

Out[52]:

school_name	School Type	Total Students	Total School Budget	Per Student Budget	Average Math Score	Average Reading Score	% P
Cabrera High School	Charter	1858	1081356	582.0	83.061895	83.975780	94.1
Thomas High School	Charter	1635	1043130	638.0	83.418349	83.848930	93.
Pena High School	Charter	962	585858	609.0	83.839917	84.044699	94.5
Griffin High School	Charter	1468	917500	625.0	83.351499	83.816757	93.3
Wilson High School	Charter	2283	1319574	578.0	83.274201	83.989488	93.8

In [55]:
```
# Group the merged data by school_name and extract the first budget va
```

```
total_school_budget = complete_df.groupby("school_name")["budget"].fir

# Format the budget values with a dollar sign, commas, and two decimal
formatted_budget = total_school_budget.map("${:,.2f}".format)

# Display the formatted budget for each school
print(formatted_budget)
```

```
school_name
Bailey High School       $3,124,928.00
Cabrera High School      $1,081,356.00
Figueroa High School     $1,884,411.00
Ford High School         $1,763,916.00
Griffin High School        $917,500.00
Hernandez High School    $3,022,020.00
Holden High School         $248,087.00
Huang High School        $1,910,635.00
Johnson High School      $3,094,650.00
Pena High School           $585,858.00
Rodriguez High School    $2,547,363.00
Shelton High School      $1,056,600.00
Thomas High School       $1,043,130.00
Wilson High School       $1,319,574.00
Wright High School       $1,049,400.00
Name: budget, dtype: object
```

In [56]:
```
# Calculate per student budget

# Group the merged DataFrame by school_name
school_groups = complete_df.groupby("school_name")

# Calculate the total number of students per school
total_students_per_school = school_groups["student_name"].count()

# Extract the total school budget for each school (each school has one
total_school_budget = school_groups["budget"].first()

# Calculate the per student budget by dividing the school's budget by
per_student_budget = total_school_budget / total_students_per_school

# Optional: Format the per student budget as currency (with a $ sign,
formatted_per_student_budget = per_student_budget.map("${:,.2f}".forma

# Display the result
formatted_per_student_budget
```

Out[56]:
```
school_name
Bailey High School       $628.00
Cabrera High School      $582.00
Figueroa High School     $639.00
Ford High School         $644.00
Griffin High School      $625.00
Hernandez High School    $652.00
Holden High School       $581.00
Huang High School        $655.00
Johnson High School      $650.00
Pena High School         $609.00
Rodriguez High School    $637.00
Shelton High School      $600.00
Thomas High School       $638.00
Wilson High School       $578.00
Wright High School       $583.00
dtype: object
```

In [57]:
```
# Cacluate the avg math and reading score

# Calculate the average math score
avg_math_score = complete_df["math_score"].mean()

# Calculate the average reading score
avg_reading_score = complete_df["reading_score"].mean()

# Print the average scores formatted as percentages with two decimal p
print(f"Average Math Score: {avg_math_score:.2f}%")
print(f"Average Reading Score: {avg_reading_score:.2f}%")
```

```
Average Math Score: 78.99%
Average Reading Score: 81.88%
```

Find the passing rate for math and reading (above 70 points)

In [60]:
```
# Find the total counts of math result
math_result_counts = complete_df["math_score"].value_counts().sort_ind

# Display the counts
print("Total counts of math scores:")
print(math_result_counts)

# Find the counts for math result in each school that pass 70 or highe
passing_math_df = complete_df[complete_df["math_score"] >= 70]
passing_math_counts = passing_math_df.groupby("school_name")["math_sco
print("\nCounts of math scores (>=70) by school:")
print(passing_math_counts)

# Calculate the math passing rate
# First, get the total number of students per school
total_students_per_school = complete_df.groupby("school_name")["studen
```

```
# Calculate the passing rate as: (passing count / total students) * 10
# Note: Use .fillna(0) to handle any schools that might have zero pass
# Round the result to 2 decimal points
math_passing_rate = (passing_math_counts / total_students_per_school *

print("\nMath passing rate by school (%):")
print(math_passing_rate)
```

```
Total counts of math scores:
math_score
55     574
56     584
57     617
58     601
59     583
60     603
61     633
62     610
63     619
64     558
65     633
66     613
67     575
68    1005
69     992
70    1014
71     971
72    1003
73     979
74     983
75     964
76     956
77     980
78    1019
79     979
80     917
81    1017
82     950
83    1017
84    1028
85     989
86     925
87     964
88     926
89     998
90     957
91    1016
92     970
93     980
94    1004
95     999
96     956
97     949
```

```
98     973
99     987
Name: count, dtype: int64

Counts of math scores (>=70) by school:
school_name
Bailey High School       3318
Cabrera High School      1749
Figueroa High School     1946
Ford High School         1871
Griffin High School      1371
Hernandez High School    3094
Holden High School        395
Huang High School        1916
Johnson High School      3145
Pena High School          910
Rodriguez High School    2654
Shelton High School      1653
Thomas High School       1525
Wilson High School       2143
Wright High School       1680
Name: math_score, dtype: int64

Math passing rate by school (%):
school_name
Bailey High School       66.68
Cabrera High School      94.13
Figueroa High School     65.99
Ford High School         68.31
Griffin High School      93.39
Hernandez High School    66.75
Holden High School       92.51
Huang High School        65.68
Johnson High School      66.06
Pena High School         94.59
Rodriguez High School    66.37
Shelton High School      93.87
Thomas High School       93.27
Wilson High School       93.87
Wright High School       93.33
dtype: float64
```

In [61]:
```
# Find the total counts of read result
read_result_counts = complete_df["reading_score"].value_counts().sort_
print("Total counts of reading scores:")
print(read_result_counts)

# Find the counts for read result in each school that pass 70 or highe
passing_reading_df = complete_df[complete_df["reading_score"] >= 70]
passing_reading_counts = passing_reading_df.groupby("school_name")["re
print("\nCounts of reading scores (>=70) by school:")
print(passing_reading_counts)
```

```
# Calculate the read passing rate
# First, get the total number of students per school
total_students_per_school = complete_df.groupby("school_name")["studen

# Calculate the passing rate as: (passing count / total students) * 10
# then round to two decimal points and fill any missing values with 0.
read_passing_rate = (passing_reading_counts / total_students_per_schoo

print("\nReading passing rate by school (%):")
print(read_passing_rate)
```

```
Total counts of reading scores:
reading_score
63     751
64     738
65     758
66     743
67     714
68     746
69    1110
70    1110
71    1073
72    1120
73    1112
74    1106
75    1149
76    1087
77    1178
78    1078
79    1116
80    1150
81    1121
82    1162
83    1176
84    1166
85    1128
86    1113
87    1161
88    1125
89    1095
90    1105
91    1076
92    1197
93    1110
94    1114
95    1126
96    1089
97    1026
98    1166
99    1075
Name: count, dtype: int64

Counts of reading scores (>=70) by school:
```

```
school_name
Bailey High School       4077
Cabrera High School      1803
Figueroa High School     2381
Ford High School         2172
Griffin High School      1426
Hernandez High School    3748
Holden High School        411
Huang High School        2372
Johnson High School      3867
Pena High School          923
Rodriguez High School    3208
Shelton High School      1688
Thomas High School       1591
Wilson High School       2204
Wright High School       1739
Name: reading_score, dtype: int64

Reading passing rate by school (%):
school_name
Bailey High School       81.93
Cabrera High School      97.04
Figueroa High School     80.74
Ford High School         79.30
Griffin High School      97.14
Hernandez High School    80.86
Holden High School       96.25
Huang High School        81.32
Johnson High School      81.22
Pena High School         95.95
Rodriguez High School    80.22
Shelton High School      95.85
Thomas High School       97.31
Wilson High School       96.54
Wright High School       96.61
dtype: float64
```

In [62]:
```
# Calculate the overall passing rate (average of the math and reading
overall_passing_rate = ((math_passing_rate + read_passing_rate) / 2).r

# Display the overall passing rate by school
print("Overall passing rate by school (%):")
print(overall_passing_rate)
```

```
Overall passing rate by school (%):
school_name
Bailey High School       74.31
Cabrera High School      95.58
Figueroa High School     73.36
Ford High School         73.81
Griffin High School      95.26
Hernandez High School    73.81
Holden High School       94.38
Huang High School        73.50
Johnson High School      73.64
Pena High School         95.27
Rodriguez High School    73.30
Shelton High School      94.86
Thomas High School       95.29
Wilson High School       95.21
Wright High School       94.97
dtype: float64
```

Bottom Performing Schools (By Passing Rate)

- Sort and display the five worst-performing schools

In [63]:
```
#  Sort and display the worst five schools in overall passing rate
# Sort the overall passing rate in ascending order and get the first f
worst_five_schools = overall_passing_rate.sort_values(ascending=True).

# Display the worst five schools in overall passing rate
print("Worst five schools in overall passing rate (%):")
print(worst_five_schools)
```

```
Worst five schools in overall passing rate (%):
school_name
Rodriguez High School    73.30
Figueroa High School     73.36
Huang High School        73.50
Johnson High School      73.64
Ford High School         73.81
dtype: float64
```

Math Scores by Grade

- Create a table that lists the average Reading 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 [69]:
```
# Create table that lists the average math score for each school of ea

# Separate the DataFrame by grade
ninth_graders = complete_df[complete_df["grade"] == "9th"]
tenth_graders = complete_df[complete_df["grade"] == "10th"]
eleventh_graders = complete_df[complete_df["grade"] == "11th"]
twelfth_graders = complete_df[complete_df["grade"] == "12th"]

# Group each grade-level subset by school_name and calculate the avera
avg_math_9th = ninth_graders.groupby("school_name")["math_score"].mean
avg_math_10th = tenth_graders.groupby("school_name")["math_score"].mea
avg_math_11th = eleventh_graders.groupby("school_name")["math_score"].
avg_math_12th = twelfth_graders.groupby("school_name")["math_score"].m

# Combine these Series into one DataFrame
math_scores_by_grade = pd.DataFrame({
    "9th": avg_math_9th,
    "10th": avg_math_10th,
    "11th": avg_math_11th,
    "12th": avg_math_12th
})

# Format the table by rounding scores to two decimal places
math_scores_by_grade = math_scores_by_grade.round(2)

# Display the resulting DataFrame
math_scores_by_grade
```

Out[69]:

school_name	9th	10th	11th	12th
Bailey High School	77.08	77.00	77.52	76.49
Cabrera High School	83.09	83.15	82.77	83.28
Figueroa High School	76.40	76.54	76.88	77.15
Ford High School	77.36	77.67	76.92	76.18
Griffin High School	82.04	84.23	83.84	83.36
Hernandez High School	77.44	77.34	77.14	77.19
Holden High School	83.79	83.43	85.00	82.86
Huang High School	77.03	75.91	76.45	77.23
Johnson High School	77.19	76.69	77.49	76.86
Pena High School	83.63	83.37	84.33	84.12
Rodriguez High School	76.86	76.61	76.40	77.69
Shelton High School	83.42	82.92	83.38	83.78
Thomas High School	83.59	83.09	83.50	83.50
Wilson High School	83.09	83.72	83.20	83.04
Wright High School	83.26	84.01	83.84	83.64

In [70]:
```
# Calculate the average math score for 9th grade in each school

# Filter the DataFrame for 9th-grade students
ninth_graders = complete_df[complete_df["grade"] == "9th"]

# Group by school_name and calculate the average math score
avg_math_9th = ninth_graders.groupby("school_name")["math_score"].mean

# Round to two decimal places
avg_math_9th = avg_math_9th.round(2)

# Convert to a percentage-style string if your scores are out of 100
avg_math_9th_formatted = avg_math_9th.map(lambda x: f"{x:.2f}%")

# Display the results
print("Average Math Score for 9th Grade by School (as a percentage):")
print(avg_math_9th_formatted)
```

```
Average Math Score for 9th Grade by School (as a percentage):
school_name
Bailey High School       77.08%
Cabrera High School      83.09%
Figueroa High School     76.40%
Ford High School         77.36%
Griffin High School      82.04%
Hernandez High School    77.44%
Holden High School       83.79%
Huang High School        77.03%
Johnson High School      77.19%
Pena High School         83.63%
Rodriguez High School    76.86%
Shelton High School      83.42%
Thomas High School       83.59%
Wilson High School       83.09%
Wright High School       83.26%
Name: math_score, dtype: object
```

In [71]:
```python
# Calculate the average math score for 10th grade in each school

# Filter the DataFrame for 10th-grade students
tenth_graders = complete_df[complete_df["grade"] == "10th"]

# Group by school_name and calculate the average math score for 10th g
avg_math_10th = tenth_graders.groupby("school_name")["math_score"].mea

# Round the average scores to two decimal places
avg_math_10th = avg_math_10th.round(2)

# Format the scores as percentages
avg_math_10th_formatted = avg_math_10th.map(lambda x: f"{x:.2f}%")

# Display the results
print("Average Math Score for 10th Grade by School (as a percentage):"
print(avg_math_10th_formatted)
```

```
Average Math Score for 10th Grade by School (as a percentage):
school_name
Bailey High School       77.00%
Cabrera High School      83.15%
Figueroa High School     76.54%
Ford High School         77.67%
Griffin High School      84.23%
Hernandez High School    77.34%
Holden High School       83.43%
Huang High School        75.91%
Johnson High School      76.69%
Pena High School         83.37%
Rodriguez High School    76.61%
Shelton High School      82.92%
Thomas High School       83.09%
Wilson High School       83.72%
Wright High School       84.01%
Name: math_score, dtype: object
```

In [72]:
```python
# Calculate the average math score for 11th grade in each school

# Filter the DataFrame for 11th-grade students
eleventh_graders = complete_df[complete_df["grade"] == "11th"]

# Group by school_name and calculate the average math score for 11th g
avg_math_11th = eleventh_graders.groupby("school_name")["math_score"].

# Round the average scores to two decimal places
avg_math_11th = avg_math_11th.round(2)

# Format the scores as percentages
avg_math_11th_formatted = avg_math_11th.map(lambda x: f"{x:.2f}%")

# Display the results
print("Average Math Score for 11th Grade by School (as a percentage):"
print(avg_math_11th_formatted)
```

```
Average Math Score for 11th Grade by School (as a percentage):
school_name
Bailey High School       77.52%
Cabrera High School      82.77%
Figueroa High School     76.88%
Ford High School         76.92%
Griffin High School      83.84%
Hernandez High School    77.14%
Holden High School       85.00%
Huang High School        76.45%
Johnson High School      77.49%
Pena High School         84.33%
Rodriguez High School    76.40%
Shelton High School      83.38%
Thomas High School       83.50%
Wilson High School       83.20%
Wright High School       83.84%
Name: math_score, dtype: object
```

In [73]:
```python
# Calculate the average math score for 12th grade in each school

# Filter the DataFrame for 12th-grade students
twelfth_graders = complete_df[complete_df["grade"] == "12th"]

# Group by school_name and calculate the average math score for 12th g
avg_math_12th = twelfth_graders.groupby("school_name")["math_score"].m

# Round the average scores to two decimal places
avg_math_12th = avg_math_12th.round(2)

# Format the scores as percentages
avg_math_12th_formatted = avg_math_12th.map(lambda x: f"{x:.2f}%")

# Display the results
print("Average Math Score for 12th Grade by School (as a percentage):"
print(avg_math_12th_formatted)
```

```
Average Math Score for 12th Grade by School (as a percentage):
school_name
Bailey High School       76.49%
Cabrera High School      83.28%
Figueroa High School     77.15%
Ford High School         76.18%
Griffin High School      83.36%
Hernandez High School    77.19%
Holden High School       82.86%
Huang High School        77.23%
Johnson High School      76.86%
Pena High School         84.12%
Rodriguez High School    77.69%
Shelton High School      83.78%
Thomas High School       83.50%
Wilson High School       83.04%
Wright High School       83.64%
Name: math_score, dtype: object
```

Reading Score by Grade

- Perform the same operations as above for reading scores

In [76]:
```python
# Create table that lists the average reading score for each school of

import pandas as pd

# Calculate the average reading score for 9th grade per school
avg_reading_9th = complete_df[complete_df["grade"] == "9th"].groupby("

# Calculate the average reading score for 10th grade per school
avg_reading_10th = complete_df[complete_df["grade"] == "10th"].groupby

# Calculate the average reading score for 11th grade per school
avg_reading_11th = complete_df[complete_df["grade"] == "11th"].groupby

# Calculate the average reading score for 12th grade per school
avg_reading_12th = complete_df[complete_df["grade"] == "12th"].groupby

# Combine these results into a single DataFrame
reading_scores_by_grade = pd.DataFrame({
    "9th": avg_reading_9th,
    "10th": avg_reading_10th,
    "11th": avg_reading_11th,
    "12th": avg_reading_12th
})

# Format each score as a percentage string with two decimal places.
reading_scores_by_grade = reading_scores_by_grade.apply(
    lambda col: col.map(lambda x: f"{x:.2f}%" if pd.notnull(x) else x)
```

```
)

# Display the resulting table
reading_scores_by_grade
```

Out[76]:

school_name	9th	10th	11th	12th
Bailey High School	81.30%	80.91%	80.95%	80.91%
Cabrera High School	83.68%	84.25%	83.79%	84.29%
Figueroa High School	81.20%	81.41%	80.64%	81.38%
Ford High School	80.63%	81.26%	80.40%	80.66%
Griffin High School	83.37%	83.71%	84.29%	84.01%
Hernandez High School	80.87%	80.66%	81.40%	80.86%
Holden High School	83.68%	83.32%	83.82%	84.70%
Huang High School	81.29%	81.51%	81.42%	80.31%
Johnson High School	81.26%	80.77%	80.62%	81.23%
Pena High School	83.81%	83.61%	84.34%	84.59%
Rodriguez High School	80.99%	80.63%	80.86%	80.38%
Shelton High School	84.12%	83.44%	84.37%	82.78%
Thomas High School	83.73%	84.25%	83.59%	83.83%
Wilson High School	83.94%	84.02%	83.76%	84.32%
Wright High School	83.83%	83.81%	84.16%	84.07%

In [77]:

```
# Calculate the average reading score for 9th grade in each school

# Filter the DataFrame for 9th-grade students
ninth_graders = complete_df[complete_df["grade"] == "9th"]

# Group by school_name and calculate the average reading score for 9th
avg_reading_9th = ninth_graders.groupby("school_name")["reading_score"

# Round the average scores to two decimal places
avg_reading_9th = avg_reading_9th.round(2)

# Format the scores as percentages
avg_reading_9th_formatted = avg_reading_9th.map(lambda x: f"{x:.2f}%")

# Display the results
print("Average Reading Score for 9th Grade by School (as a percentage)
print(avg_reading_9th_formatted)
```

```
Average Reading Score for 9th Grade by School (as a percentage):
school_name
Bailey High School       81.30%
Cabrera High School      83.68%
Figueroa High School     81.20%
Ford High School         80.63%
Griffin High School      83.37%
Hernandez High School    80.87%
Holden High School       83.68%
Huang High School        81.29%
Johnson High School      81.26%
Pena High School         83.81%
Rodriguez High School    80.99%
Shelton High School      84.12%
Thomas High School       83.73%
Wilson High School       83.94%
Wright High School       83.83%
Name: reading_score, dtype: object
```

In [78]:
```python
# Calculate the average reading score for 10th grade in each school

# Filter the DataFrame for 10th-grade students
tenth_graders = complete_df[complete_df["grade"] == "10th"]

# Group by school_name and calculate the average reading score for 10t
avg_reading_10th = tenth_graders.groupby("school_name")["reading_score

# Round the average scores to two decimal places
avg_reading_10th = avg_reading_10th.round(2)

# Format the scores as percentages
avg_reading_10th_formatted = avg_reading_10th.map(lambda x: f"{x:.2f}%

# Display the results
print("Average Reading Score for 10th Grade by School (as a percentage
print(avg_reading_10th_formatted)
```

```
Average Reading Score for 10th Grade by School (as a percentage):
school_name
Bailey High School       80.91%
Cabrera High School      84.25%
Figueroa High School     81.41%
Ford High School         81.26%
Griffin High School      83.71%
Hernandez High School    80.66%
Holden High School       83.32%
Huang High School        81.51%
Johnson High School      80.77%
Pena High School         83.61%
Rodriguez High School    80.63%
Shelton High School      83.44%
Thomas High School       84.25%
Wilson High School       84.02%
Wright High School       83.81%
Name: reading_score, dtype: object
```

In [79]:
```python
# Calculate the average reading score for 11th grade in each school

# Filter the DataFrame for 11th-grade students
eleventh_graders = complete_df[complete_df["grade"] == "11th"]

# Group by school_name and calculate the average reading score for 11t
avg_reading_11th = eleventh_graders.groupby("school_name")["reading_sc

# Round the average scores to two decimal places
avg_reading_11th = avg_reading_11th.round(2)

# Format the scores as percentages
avg_reading_11th_formatted = avg_reading_11th.map(lambda x: f"{x:.2f}%

# Display the results
print("Average Reading Score for 11th Grade by School (as a percentage
print(avg_reading_11th_formatted)
```

```
Average Reading Score for 11th Grade by School (as a percentage):
school_name
Bailey High School       80.95%
Cabrera High School      83.79%
Figueroa High School     80.64%
Ford High School         80.40%
Griffin High School      84.29%
Hernandez High School    81.40%
Holden High School       83.82%
Huang High School        81.42%
Johnson High School      80.62%
Pena High School         84.34%
Rodriguez High School    80.86%
Shelton High School      84.37%
Thomas High School       83.59%
Wilson High School       83.76%
Wright High School       84.16%
Name: reading_score, dtype: object
```

In [80]:
```python
# Calculate the average reading score for 12th grade in each school

# Filter the DataFrame for 12th-grade students
twelfth_graders = complete_df[complete_df["grade"] == "12th"]

# Group by school_name and calculate the average reading score for 12t
avg_reading_12th = twelfth_graders.groupby("school_name")["reading_sco

# Round the average scores to two decimal places
avg_reading_12th = avg_reading_12th.round(2)

# Format the scores as percentages
avg_reading_12th_formatted = avg_reading_12th.map(lambda x: f"{x:.2f}%

# Display the results
print("Average Reading Score for 12th Grade by School (as a percentage
print(avg_reading_12th_formatted)
```

```
Average Reading Score for 12th Grade by School (as a percentage):
school_name
Bailey High School       80.91%
Cabrera High School      84.29%
Figueroa High School     81.38%
Ford High School         80.66%
Griffin High School      84.01%
Hernandez High School    80.86%
Holden High School       84.70%
Huang High School        80.31%
Johnson High School      81.23%
Pena High School         84.59%
Rodriguez High School    80.38%
Shelton High School      82.78%
Thomas High School       83.83%
Wilson High School       84.32%
Wright High School       84.07%
Name: reading_score, dtype: object
```

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 [83]:
```
# Sample bins. Feel free to create your own bins.
spending_bins = [0, 585, 630, 645, 675]
spending_labels = ["< $585", "$585-630", "$630-645", "$645-675"]
```

In [84]:
```
# Categorize Schools into Spending Bins

school_summary["Spending Ranges (Per Student)"] = pd.cut(
    school_summary["Per Student Budget"],
    spending_bins,
    labels=spending_labels
)
```

In [89]:
```
#Group by Spending Ranges and Calculate Averages

# Group by the new bin column
spending_groups = school_summary.groupby("Spending Ranges (Per Student
```

```
# Calculate the average of each metric within each spending bin
spending_summary = spending_groups[[
    "Average Math Score",
    "Average Reading Score",
    "% Passing Math",
    "% Passing Reading",
    "% Overall Passing"
]].mean()

# Format the Results
spending_summary["Average Math Score"] = spending_summary["Average Mat
spending_summary["Average Reading Score"] = spending_summary["Average
spending_summary["% Passing Math"] = spending_summary["% Passing Math"
spending_summary["% Passing Reading"] = spending_summary["% Passing Re
spending_summary["% Overall Passing"] = spending_summary["% Overall Pa

spending_summary
```

Out[89]:

Spending Ranges (Per Student)	Average Math Score	Average Reading Score	% Passing Math	% Passing Reading	% Overall Passing
< $585	83.46	83.93	93.46%	96.61%	95.04%
$585-630	81.90	83.16	87.13%	92.72%	89.93%
$630-645	78.52	81.62	73.48%	84.39%	78.94%
$645-675	77.00	81.03	66.16%	81.13%	73.65%

In [90]:

```
# Create a new column to show budget per student in each row

# Calculate total students per school
#    transform("count") returns a Series the same length as complete_d
#    where each row has the count of students in that row's school.
total_students_per_school = complete_df.groupby("school_name")["studen

# Create the new column: budget per student
complete_df["budget_per_student"] = complete_df["budget"] / total_stud

# round to two decimals or format
complete_df["budget_per_student"] = complete_df["budget_per_student"].

# Preview the updated DataFrame
complete_df.head()
```

Out[90]:

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

In [94]:

```
# Create a new column to define the spending ranges per student

import pandas as pd

# Define the spending bins and labels
spending_bins = [0, 585, 630, 645, 675]
spending_labels = ["< $585", "$585–630", "$630–645", "$645–675"]

# Use pd.cut to categorize each row's budget_per_student
complete_df["Spending Ranges (Per Student)"] = pd.cut(
    complete_df["budget_per_student"],
    bins=spending_bins,
    labels=spending_labels
)


# Preview the updated DataFrame
complete_df.head()
```

Out[94]:

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

In [96]:
```
# Calculate the average math score within each spending range

# Group by the spending ranges and calculate the average math score fo
avg_math_by_spending = complete_df.groupby(
    "Spending Ranges (Per Student)",
    observed=False
)["math_score"].mean().round(2)

# (Optional) Format the average math scores as percentages
avg_math_by_spending_formatted = avg_math_by_spending.map(lambda x: f"

# Display the results
print("Average Math Score by Spending Range:")
print(avg_math_by_spending_formatted)
```

```
Average Math Score by Spending Range:
Spending Ranges (Per Student)
< $585      83.36%
$585-630    79.98%
$630-645    77.82%
$645-675    77.05%
Name: math_score, dtype: object
```

In [ ]:
```
# Calculate the percentage passing rate for math in each spending rang
```

In [ ]:
```
# Calculate the percentage passing rate for reading in each spending r
```

In [ ]:
```
# Calculate the percentage overall passing rate in each spending range
```

Scores by School Size

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

In [ ]:
```
# Sample bins. Feel free to create your own bins.
size_bins = [0, 1000, 2000, 5000]
group_names = ["Small (<1000)", "Medium (1000-2000)", "Large (2000-500
```

In [98]:
```
# Create a new column for the bin groups

# Filter rows where math_score >= 70
passing_math_df = complete_df[complete_df["math_score"] >= 70]

# Count how many students passed math in each spending range
passing_math_by_spending = passing_math_df.groupby(
    "Spending Ranges (Per Student)",
    observed=True
)["math_score"].count()


# Count the total number of students in each spending range
total_students_by_spending = complete_df.groupby(
    "Spending Ranges (Per Student)",
    observed=True
)["math_score"].count()

# Calculate the passing rate as (passing / total) * 100
math_passing_rate_by_spending = (passing_math_by_spending / total_stud

# Round to two decimal places (optional)
math_passing_rate_by_spending = math_passing_rate_by_spending.round(2)

# Format as percentages
math_passing_rate_by_spending_formatted = math_passing_rate_by_spendin

# Display the results
print("Math Passing Rate by Spending Range:")
print(math_passing_rate_by_spending_formatted)
```

```
Math Passing Rate by Spending Range:
Spending Ranges (Per Student)
< $585       93.70%
$585-630     79.11%
$630-645     70.62%
$645-675     66.23%
Name: math_score, dtype: object
```

Look for the total count of test scores that pass 70% or higher

In [100…
```
# math_pass_size

#Define School Size Bins and Labels
size_bins = [0, 1000, 2000, 5000]
size_labels = ["Small (<1000)", "Medium (1000-2000)", "Large (2000-500
```

```
#Create a "School Size" Column
complete_df["School Size"] = pd.cut(
    complete_df["size"],
    bins=size_bins,
    labels=size_labels
)

# Calculate the Passing Math Rate By School Size

# Filter rows where math_score >= 70
passing_math_df = complete_df[complete_df["math_score"] >= 70]

# Count passing students by school size
passing_math_size_counts = passing_math_df.groupby(
    "School Size",
    observed=False
)["math_score"].count()

# Count total students by school size
total_students_size_counts = complete_df.groupby(
    "School Size",
    observed=False
)["math_score"].count()

# Calculate passing rate
math_pass_size = (passing_math_size_counts / total_students_size_count

# Round to two decimals (optional)
math_pass_size = math_pass_size.round(2)

# Format as a percentage string
math_pass_size_formatted = math_pass_size.map(lambda x: f"{x:.2f}%")

print("Math Passing Rate by School Size:")
print(math_pass_size_formatted)
```

```
Math Passing Rate by School Size:
School Size
Small (<1000)         93.95%
Medium (1000-2000)    93.62%
Large (2000-5000)     68.65%
Name: math_score, dtype: object
```

In [102…

```
# read_pass_size

# Filter rows where reading_score >= 70
passing_reading_df = complete_df[complete_df["reading_score"] >= 70]

# Count how many students passed reading in each size bin
passing_reading_size_counts = passing_reading_df.groupby(
    "School Size",
    observed=False
```

```
)["reading_score"].count()

# Count the total number of students in each size bin
total_students_size_counts = complete_df.groupby(
    "School Size",
    observed=False
)["reading_score"].count()

# Calculate the passing rate as (passing / total) * 100
read_pass_size = (passing_reading_size_counts / total_students_size_co

# Round to two decimals (optional)
read_pass_size = read_pass_size.round(2)

# Format as a percentage string
read_pass_size_formatted = read_pass_size.map(lambda x: f"{x:.2f}%")

print("Reading Passing Rate by School Size:")
print(read_pass_size_formatted)
```

```
Reading Passing Rate by School Size:
School Size
Small (<1000)          96.04%
Medium (1000-2000)     96.77%
Large (2000-5000)      82.13%
Name: reading_score, dtype: object
```

In [104…

```
# Calculate the overall passing rate for different school size

# Filter rows where both math_score >= 70 and reading_score >= 70
passing_both_df = complete_df[
    (complete_df["math_score"] >= 70) & (complete_df["reading_score"]
]

# Count how many students passed both subjects in each school-size bin
passing_both_size_counts = passing_both_df.groupby(
    "School Size",
    observed=True
)["student_name"].count()

# Count the total number of students in each school-size bin
total_students_size_counts = complete_df.groupby(
    "School Size",
    observed=True
)["student_name"].count()

# Calculate the overall passing rate as (passing both / total) * 100
overall_passing_size = (passing_both_size_counts / total_students_size

# Round to two decimals
overall_passing_size = overall_passing_size.round(2)

# Format as a percentage string
```

```
overall_passing_size_formatted = overall_passing_size.map(lambda x: f"

print("Overall Passing Rate by School Size:")
print(overall_passing_size_formatted)
```

```
Overall Passing Rate by School Size:
School Size
Small (<1000)          90.14%
Medium (1000-2000)     90.62%
Large (2000-5000)      56.57%
Name: student_name, dtype: object
```

Scores by School Type

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

In [110…

```
# Create bins and groups, school type {'Charter', 'District'}

import pandas as pd

# Define the possible categories for school type
type_categories = ["Charter", "District"]

# Convert the 'type' column into a categorical with these categories
complete_df["School Type"] = pd.Categorical(complete_df["type"], categ

# Group by "School Type"
type_groups = complete_df.groupby("School Type", observed=True)

# Example: Calculate average math score by school type
avg_math_by_type = type_groups["math_score"].mean()

print(avg_math_by_type)
```

```
School Type
Charter     83.406183
District    76.987026
Name: math_score, dtype: float64
```

Find counts of the passing 70 or higher score for the both test

In [112…

```
# math pass size

# Create the "School Size" bins/labels if you haven't already
size_bins = [0, 1000, 2000, 5000]
size_labels = ["Small (<1000)", "Medium (1000-2000)", "Large (2000-500

# Categorize each row's school size
complete_df["School Size"] = pd.cut(
    complete_df["size"],
    bins=size_bins,
```

```
    labels=size_labels
)

# Filter for students passing math
passing_math_df = complete_df[complete_df["math_score"] >= 70]

# Count passing students vs. total students per size category
passing_math_size_counts = passing_math_df.groupby(
    "School Size",
    observed=True
)["student_name"].count()

total_students_size_counts = complete_df.groupby(
    "School Size",
    observed=True
)["student_name"].count()

# Calculate the passing rate and round to two decimals
math_pass_size = (passing_math_size_counts / total_students_size_count
math_pass_size = math_pass_size.round(2)

# Format as percentage strings
math_pass_size_formatted = math_pass_size.map(lambda x: f"{x:.2f}%")

print("Math Passing Rate by School Size:")
print(math_pass_size_formatted)
```

```
Math Passing Rate by School Size:
School Size
Small (<1000)          93.95%
Medium (1000-2000)     93.62%
Large (2000-5000)      68.65%
Name: student_name, dtype: object
```

In [114…

```
# reading pass size

import pandas as pd

# Ensure your DataFrame has a "School Size" column (bins/labels)
#    For example:
# size_bins = [0, 1000, 2000, 5000]
# size_labels = ["Small (<1000)", "Medium (1000-2000)", "Large (2000-5
# complete_df["School Size"] = pd.cut(complete_df["size"], bins=size_b

# Filter for students passing reading (score >= 70)
passing_reading_df = complete_df[complete_df["reading_score"] >= 70]

# Count passing students vs. total students per size category
passing_reading_size_counts = passing_reading_df.groupby(
    "School Size",
    observed=True
)["student_name"].count()
```

```
total_students_size_counts = complete_df.groupby(
    "School Size",
    observed=True
)["student_name"].count()

# Calculate the passing rate and round to two decimals
read_pass_size = (passing_reading_size_counts / total_students_size_co
read_pass_size = read_pass_size.round(2)

# Format as percentage strings
read_pass_size_formatted = read_pass_size.map(lambda x: f"{x:.2f}%")

print("Reading Passing Rate by School Size:")
print(read_pass_size_formatted)
```

```
Reading Passing Rate by School Size:
School Size
Small (<1000)          96.04%
Medium (1000-2000)     96.77%
Large (2000-5000)      82.13%
Name: student_name, dtype: object
```

In [115…

```
# Calculate the overall passing rate

# Filter rows where both math_score >= 70 and reading_score >= 70
passing_both_df = complete_df[
    (complete_df["math_score"] >= 70) & (complete_df["reading_score"]
]

# Count how many students passed both
passing_both_count = passing_both_df.shape[0]

# Count the total number of students
total_students = complete_df.shape[0]

# Calculate the overall passing rate
overall_passing_rate = (passing_both_count / total_students) * 100

# Round to two decimals (optional)
overall_passing_rate = round(overall_passing_rate, 2)

# Format as a percentage string
overall_passing_rate_str = f"{overall_passing_rate:.2f}%"

print("Overall Passing Rate:", overall_passing_rate_str)
```

```
Overall Passing Rate: 65.17%
```