# Dependencies and Setup

import pandas as pd

from pathlib import Path

import numpy as np

import os

# File to Load (Remember to Change These)

school\_data\_to\_load = os.path.join("../PyCitySchools\Resources\schools\_completePKA.csv")

student\_data\_to\_load = os.path.join("../PyCitySchools\Resources\students\_completePKA.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", on=["school\_name", "school\_name"])

school\_data\_complete.head()

**Student ID student\_name gender grade school\_name reading\_score math\_score School ID type size budget**

**0 0 Paul Bradley M 9th Huang High School 66 79 0 District 2917 1910635**

**1 1 Victor Smith M 12th Huang High School 94 61 0 District 2917 1910635**

**2 2 Kevin Rodriguez M 12th Huang High School 90 60 0 District 2917 1910635**

**3 3 Dr. Richard Scott M 12th Huang High School 67 58 0 District 2917 1910635**

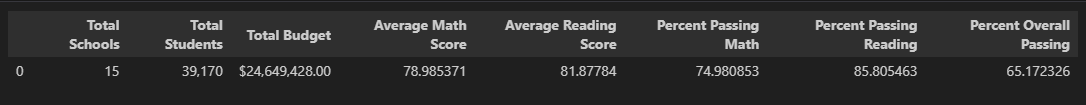
**4 4 Bonnie Ray F 9th Huang High School 97 84 0 District 2917 1910635**

**District Summary**

Perform the necessary calculations and then create a high-level snapshot of the district's key metrics in a DataFrame.

Include the following:

* Total number of unique schools
* Total students
* Total budget
* Average math score
* Average reading score
* % passing math (the percentage of students who passed math)
* % passing reading (the percentage of students who passed reading)
* % overall passing (the percentage of students who passed math AND reading)
* # Total number of unique schools
* school\_count = school\_data\_complete["school\_name"].count()
* school\_count = school\_data\_complete["school\_name"].nunique()
* school\_count
* # Total students
* student\_count = school\_data\_complete["student\_name"].count()
* student\_count = school\_data\_complete["Student ID"].nunique()
* student\_count
* # Total budget
* total\_budget = school\_data["budget"].sum()
* total\_budget
* # Average Math Score
* average\_math\_score = school\_data\_complete["math\_score"].mean()
* average\_math\_score
* # Average reading Score
* average\_reading\_score = school\_data\_complete["reading\_score"].mean()
* average\_reading\_score
* # % Passing Math (the percentage of students who passed math)
* passing\_math\_count = school\_data\_complete[(school\_data\_complete["math\_score"] >= 70)].count()["student\_name"]
* passing\_math\_percentage = passing\_math\_count / float(student\_count) \* 100
* passing\_math\_percentage
* # % Passing Reading (the percentage of students who passed reading)
* passing\_reading\_count = school\_data\_complete[(school\_data\_complete["reading\_score"] >= 70)].count()["student\_name"]
* passing\_reading\_percentage = passing\_reading\_count / float(student\_count) \* 100
* passing\_reading\_percentage
* # % Overall Passing (the percentage of students who passed math AND reading)
* passing\_math\_reading\_count = school\_data\_complete[
* (school\_data\_complete["math\_score"] >= 70) & (school\_data\_complete["reading\_score"] >= 70)
* ].count()["student\_name"]
* overall\_passing\_rate = passing\_math\_reading\_count /  float(student\_count) \* 100
* overall\_passing\_rate
* # Create a high-level snapshot of the district's key metrics in a DataFrame
* district\_summary = pd.DataFrame({"Total Schools" : [school\_count],
* "Total Students" : [student\_count],
* "Total Budget" : [total\_budget],
* "Average Math Score" : [average\_math\_score],
* "Average Reading Score" : [average\_reading\_score],
* "Percent Passing Math" : [passing\_math\_percentage],
* "Percent Passing Reading" : [passing\_reading\_percentage],
* "Percent Overall Passing" : [overall\_passing\_rate]
* })
* # Formatting
* district\_summary["Total Students"] = district\_summary["Total Students"].map("{:,}".format)
* district\_summary["Total Budget"] = district\_summary["Total Budget"].map("${:,.2f}".format)
* # Display the DataFrame
* district\_summary



**School Summary**

Perform the necessary calculations and then create a DataFrame that summarizes key metrics about each school.

Include the following:

* School name
* School type
* Total students
* Total school budget
* Per student budget
* Average math score
* Average reading score
* % passing math (the percentage of students who passed math)
* % passing reading (the percentage of students who passed reading)
* % overall passing (the percentage of students who passed math AND reading)
* # Perform the necessary calculations and then create a DataFrame that summarizes key metrics about each school
* school\_types = school\_data.set\_index(["school\_name"])["type"]
* school\_types

school\_name

Huang High School District

Figueroa High School District

Shelton High School Charter

Hernandez High School District

Griffin High School Charter

Wilson High School Charter

Cabrera High School Charter

Bailey High School District

Holden High School Charter

Pena High School Charter

Wright High School Charter

Rodriguez High School District

Johnson High School District

Ford High School District

Thomas High School Charter

Name: type, dtype: object

# total student count per school from school\_data

per\_school\_counts = school\_data\_complete["school\_name"].value\_counts()

per\_school\_counts = per\_school\_counts.reset\_index()

per\_school\_counts.columns = ["school\_name", "count"]

per\_school\_counts

per\_school\_counts = per\_school\_counts.set\_index(["school\_name"])

per\_school\_counts

count

school\_name

Bailey High School 4976

Johnson High School 4761

Hernandez High School 4635

Rodriguez High School 3999

Figueroa High School 2949

Huang High School 2917

Ford High School 2739

Wilson High School 2283

Cabrera High School 1858

Wright High School 1800

Shelton High School 1761

Thomas High School 1635

Griffin High School 1468

Pena High School 962

Holden High School 427

# Calculate the total school budget per school from school\_data

per\_school\_budget = school\_data\_complete[["school\_name", "budget"]].groupby(["school\_name"]).mean()

per\_school\_budget

budget

school\_name

Bailey High School 3124928.0

Cabrera High School 1081356.0

Figueroa High School 1884411.0

Ford High School 1763916.0

Griffin High School 917500.0

Hernandez High School 3022020.0

Holden High School 248087.0

Huang High School 1910635.0

Johnson High School 3094650.0

Pena High School 585858.0

Rodriguez High School 2547363.0

Shelton High School 1056600.0

Thomas High School 1043130.0

Wilson High School 1319574.0

Wright High School 1049400.0

# Calculate per capita spending per school from school\_data

per\_school\_capita = per\_school\_budget["budget"]  / per\_school\_counts["count"]

per\_school\_capita

school\_name

Bailey High School 628.0

Cabrera High School 582.0

Figueroa High School 639.0

Ford High School 644.0

Griffin High School 625.0

Hernandez High School 652.0

Holden High School 581.0

Huang High School 655.0

Johnson High School 650.0

Pena High School 609.0

Rodriguez High School 637.0

Shelton High School 600.0

Thomas High School 638.0

Wilson High School 578.0

Wright High School 583.0

dtype: float64

# Calculate the average test scores per school from school\_data\_complete

per\_school\_math = school\_data\_complete.set\_index(["school\_name"])["math\_score"].mean()

per\_school\_math

# Reading Score

per\_school\_reading = school\_data\_complete.set\_index(["school\_name"])["reading\_score"].mean()

per\_school\_reading

# Calculate the average test scores per school from school\_data\_complete

per\_school\_math\_a = school\_data\_complete.groupby(["school\_name"])["math\_score"].mean()

per\_school\_math\_a

school\_name

Bailey High School 77.048432

Cabrera High School 83.061895

Figueroa High School 76.711767

Ford High School 77.102592

Griffin High School 83.351499

Hernandez High School 77.289752

Holden High School 83.803279

Huang High School 76.629414

Johnson High School 77.072464

Pena High School 83.839917

Rodriguez High School 76.842711

Shelton High School 83.359455

Thomas High School 83.418349

Wilson High School 83.274201

Wright High School 83.682222

 Average reading score

per\_school\_reading\_a = school\_data\_complete.groupby(["school\_name"])['reading\_score'].mean()

per\_school\_reading\_a

# Calculate the number of students per school with math scores of 70 or higher from school\_data\_complete

students\_passing\_math = school\_data\_complete[(school\_data\_complete["math\_score"] >= 70)]

students\_passing\_math

**A screen shot of a black screen

Description automatically generated**

school\_students\_passing\_math = school\_data\_complete[(school\_data\_complete["math\_score"] >= 70)]

per\_school\_students\_passing\_math = school\_students\_passing\_math.groupby(["school\_name"]).count()["student\_name"]

per\_school\_students\_passing\_math

# Calculate the number of students per school with reading scores of 70 or higher from school\_data\_complete

students\_passing\_reading = school\_data\_complete[(school\_data\_complete["reading\_score"] >= 70)]

students\_passing\_reading

school\_students\_passing\_reading = school\_data\_complete[(school\_data\_complete["reading\_score"] >= 70)]

per\_school\_students\_passing\_reading = school\_students\_passing\_reading.groupby(["school\_name"]).count()["student\_name"]

per\_school\_students\_passing\_reading

# Use the provided code to calculate the number of students per school that passed both math and reading with scores of 70 or higher

students\_passing\_math\_and\_reading = school\_data\_complete[(school\_data\_complete["reading\_score"] >= 70) & (school\_data\_complete["math\_score"] >= 70)]

students\_passing\_math\_and\_reading

# by school passing math and reading

school\_students\_passing\_math\_and\_reading = school\_data\_complete[(school\_data\_complete["reading\_score"] >= 70) & (school\_data\_complete["math\_score"] >= 70)]

per\_school\_students\_passing\_math\_and\_reading = school\_students\_passing\_math\_and\_reading.groupby(["school\_name"]).count()["student\_name"]

per\_school\_students\_passing\_math\_and\_reading

# Use the provided code to calculate the passing rates

school\_students\_passing\_math\_n = per\_school\_students\_passing\_math / per\_school\_counts["count"] \* 100

school\_students\_passing\_reading\_n = per\_school\_students\_passing\_reading / per\_school\_counts["count"] \* 100

overall\_passing\_rate = per\_school\_students\_passing\_math\_and\_reading  / per\_school\_counts["count"] \* 100

# school\_students\_passing\_math\_n

# school\_students\_passing\_reading\_n

overall\_passing\_rate

School summary

**A screenshot of a graph

Description automatically generated**

**Highest-Performing Schools (by % Overall Passing)**

Sort the schools by % Overall Passing in descending order and display the top 5 rows.

Save the results in a DataFrame called "top\_schools".

A screenshot of a computer

Description automatically generated

**Lowest-Performing Schools (by % Overall Passing)**

Sort the schools by % Overall Passing in ascending order and display the top 5 rows.

Save the results in a DataFrame called "bottom\_schools".

A screenshot of a computer

Description automatically generated

**Math Scores by Grade**

Perform the necessary calculations to create a DataFrame that lists the average math score for students of each grade level (9th, 10th, 11th, 12th) at each school.

A screenshot of a graph

Description automatically generated

**Reading Scores by Grade**

Create a DataFrame that lists the average reading score for students of each grade level (9th, 10th, 11th, 12th) at each school.

A screenshot of a computer

Description automatically generated

**Scores by School Spending**

Create a table that breaks down school performance based on average spending ranges (per student).

Use the code provided below to create four bins with reasonable cutoff values to group school spending.

spending\_bins = [0, 585, 630, 645, 680]

labels = ["<$585", "$585-630", "$630-645", "$645-680"]

Use pd.cut to categorize spending based on the bins.

Use the following code to then calculate mean scores per spending range.

spending\_math\_scores = school\_spending\_df.groupby(["Spending Ranges (Per Student)"])["Average Math Score"].mean()

spending\_reading\_scores = school\_spending\_df.groupby(["Spending Ranges (Per Student)"])["Average Reading Score"].mean()

spending\_passing\_math = school\_spending\_df.groupby(["Spending Ranges (Per Student)"])["% Passing Math"].mean()

spending\_passing\_reading = school\_spending\_df.groupby(["Spending Ranges (Per Student)"])["% Passing Reading"].mean()

overall\_passing\_spending = school\_spending\_df.groupby(["Spending Ranges (Per Student)"])["% Overall Passing"].mean()

Use the scores above to create a DataFrame called spending\_summary.

Include the following metrics in the table:

* Average math score
* Average reading score
* % passing math (the percentage of students who passed math)
* % passing reading (the percentage of students who passed reading)
* % overall passing (the percentage of students who passed math AND reading)

A screenshot of a computer

Description automatically generated

**Scores by School Size**

Use the following code to bin the per\_school\_summary.

size\_bins = [0, 1000, 2000, 5000]

labels = ["Small (<1000)", "Medium (1000-2000)", "Large (2000-5000)"]

Use pd.cut on the "Total Students" column of the per\_school\_summary DataFrame.

Create a DataFrame called size\_summary that breaks down school performance based on school size (small, medium, or large).

A screenshot of a computer

Description automatically generated

**Scores by School Type**

Use the per\_school\_summary DataFrame from the previous step to create a new DataFrame called type\_summary.

This new DataFrame should show school performance based on the "School Type".

A screenshot of a computer

Description automatically generated