**Module 4 Challenge – Notes/Brain Dump**

**syntax debug per\_school\_counts = school\_data\_complete["Student ID","school\_name"].nunique**

It looks like there is a syntax error in the code you provided. The correct syntax to count unique values in two columns in a Pandas DataFrame is as follows:

per\_school\_counts = school\_data\_complete.groupby("school\_name")["Student ID"].nunique()

This code will count the unique number of students in each school in the DataFrame school\_data\_complete.

**Did below and second guessed myself so checked to see if reading how I intended.**  
**read per\_school\_budget = school\_data.groupby("school\_name")["budget"].sum() for school\_name in school\_data: per\_school\_capita = per\_school\_budget/per\_school\_counts per\_school\_capita**

It seems like you are trying to calculate the per capita budget for each school. Here is the correct way to calculate it:

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

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

per\_school\_capita = per\_school\_budget / per\_school\_counts

per\_school\_capita

This code will calculate the per capita budget for each school by dividing the total budget of each school by the number of students in that school.

**help # Calculate the number of students per school with reading scores of 70 or higher from school\_data\_complete students\_passing\_reading = school\_students\_passing\_reading =**

To calculate the number of students per school with reading scores of 70 or higher from the school\_data\_complete dataset, you can use the following code snippet in Python using pandas:

# Assuming 'school\_data\_complete' is the DataFrame containing the data

students\_passing\_reading = school\_data\_complete[school\_data\_complete['reading\_score'] >= 70]

school\_students\_passing\_reading = students\_passing\_reading.groupby('school\_name')['Student ID'].count()

In this code:

* students\_passing\_reading filters the DataFrame to include only students with reading scores of 70 or higher.
* school\_students\_passing\_reading then groups this filtered data by school name and counts the number of students passing reading for each school.

This will give you the number of students per school with reading scores of 70 or higher.

**explain the above logic or template**

The code snippet provided is written in Python using the pandas library for data manipulation. Let's break down the logic step by step:

1. **Filtering Students Passing Reading**:
   * students\_passing\_reading = school\_data\_complete[school\_data\_complete['reading\_score'] >= 70]
     + This line filters the school\_data\_complete DataFrame to include only those students who have a reading score of 70 or higher.
     + The condition school\_data\_complete['reading\_score'] >= 70 checks if the reading score of each student is 70 or greater.
     + The resulting DataFrame students\_passing\_reading will contain only the rows where students have passed reading with a score of 70 or higher.
2. **Grouping by School and Counting Students**:
   * school\_students\_passing\_reading = students\_passing\_reading.groupby('school\_name')['Student ID'].count()
     + This line groups the filtered DataFrame students\_passing\_reading by the 'school\_name' column.
     + It then counts the number of students (using the 'Student ID' column) in each group, which represents the number of students per school who have passed reading with a score of 70 or higher.
     + The resulting school\_students\_passing\_reading Series will have the school names as the index and the count of students passing reading for each school as the values.

By following this logic, you can calculate the number of students per school with reading scores of 70 or higher from the school\_data\_complete dataset.

**pandas datatframe .size explain**

In Pandas, the .size attribute is used to return the number of elements in a DataFrame. It returns a single integer value representing the total number of elements in the DataFrame, which is equivalent to the number of rows multiplied by the number of columns.

Here's a simple example to illustrate how .size works:

import pandas as pd

# Create a sample DataFrame

data = {'A': [1, 2, 3], 'B': [4, 5, 6]}

df = pd.DataFrame(data)

# Get the size of the DataFrame

df\_size = df.size

print(df)

print("Size of the DataFrame:", df\_size)

Output:

A B

0 1 4

1 2 5

2 3 6

Size of the DataFrame: 6

In this example, the DataFrame df has 3 rows and 2 columns, so the total number of elements in the DataFrame is 3 \* 2 = 6. Therefore, df.size returns 6 as the size of the DataFrame.