4.8.4 Get the Score Averages Per School

You're almost done with getting all the data for the school summary! Now you need to calculate the average math score and the average reading score for each school.

Next, we need to perform a few more calculations for the final data to be added to the school summary DataFrame.

First, let's get the average reading and math scores for each school.

↻ Retake

REWIND

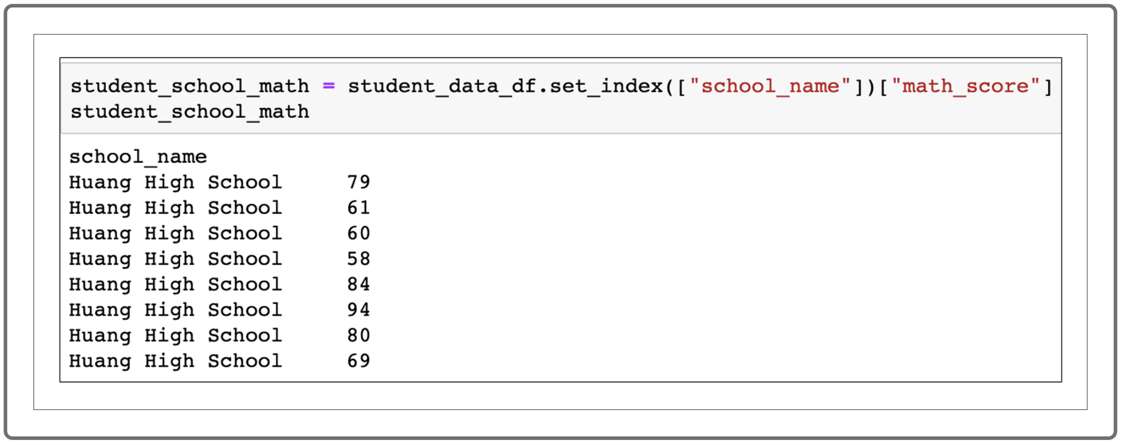
Make sure that the averages have an index of `school_name` so the data can be added to the DataFrame.

We've used the `set_index()` method on the school_name column in `student_data_df` to get data from another column, just like how we retrieved the school budget using `school_data_df.set_index(["school_name"])["budget"]`.

Let's use this procedure to replace `budget` with `math_score`. Add the following code to a new cell and run the cell.

```
# Calculate the math scores.
student_school_math = student_data_df.set_index(["school_name"])["math_score
```

The output from the code will look like the following, where we get every occurrence of the high school as the index, and the math grade from each student in that school.



```
student_school_math = student_data_df.set_index(["school_name"])["math_score"]
student_school_math
```

```
school_name
Huang High School     79
Huang High School     61
Huang High School     60
Huang High School     58
Huang High School     84
Huang High School     94
Huang High School     80
Huang High School     69
```

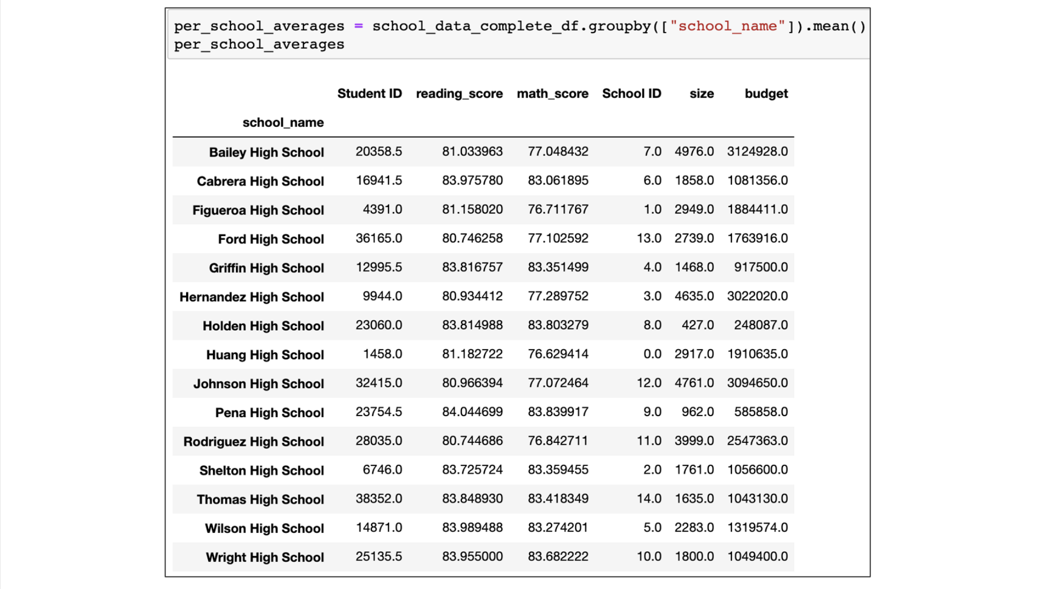
Unfortunately, we can't use the `school_data_df` DataFrame, as there aren't any columns containing grades. We also can't use the `set_index()` method on the school_name column in `student_data_df` because there are too many occurrences of the school_name column.

Instead, we need to use the Pandas `groupby()` function. The `groupby()` function will split an object (like a DataFrame), apply a mathematical operation, and combine the results. This can be used to group large amounts of data when we want to compute mathematical operations on these groups.

The mathematical operation we will apply to the `groupby()` function is the `mean()` method. Let's see how this will look when we apply it to `school_data_complete_df` to get the grade averages for each column. Add the following code to a new cell and run the cell.

```
# Calculate the average math scores.
per_school_averages = school_data_complete_df.groupby(["school_name"]).mean(
per_school_averages
```

The output will be the average of each column in the `school_data_complete_df` DataFrame:



```
per_school_averages = school_data_complete_df.groupby(["school_name"]).mean()
per_school_averages
```

	Student ID	reading_score	math_score	School ID	size	budget
school_name						
Bailey High School	20358.5	81.033963	77.048432	7.0	4976.0	3124928.0
Cabrera High School	16941.5	83.975780	83.061895	6.0	1858.0	1081356.0
Figueroa High School	4391.0	81.158020	76.711767	1.0	2949.0	1884411.0
Ford High School	36165.0	80.746258	77.102592	13.0	2739.0	1763916.0
Griffin High School	12995.5	83.816757	83.351499	4.0	1468.0	917500.0
Hernandez High School	9944.0	80.934412	77.289752	3.0	4635.0	3022020.0
Holden High School	23060.0	83.814988	83.803279	8.0	427.0	248087.0
Huang High School	1458.0	81.182722	76.629414	0.0	2917.0	1910635.0
Johnson High School	32415.0	80.966394	77.072464	12.0	4761.0	3094650.0
Pena High School	23754.5	84.044699	83.839917	9.0	962.0	585858.0
Rodriguez High School	28035.0	80.744686	76.842711	11.0	3999.0	2547363.0
Shelton High School	6746.0	83.725724	83.359455	2.0	1761.0	1056600.0
Thomas High School	38352.0	83.848930	83.418349	14.0	1635.0	1043130.0
Wilson High School	14871.0	83.989488	83.274201	5.0	2283.0	1319574.0
Wright High School	25135.5	83.955000	83.682222	10.0	1800.0	1049400.0

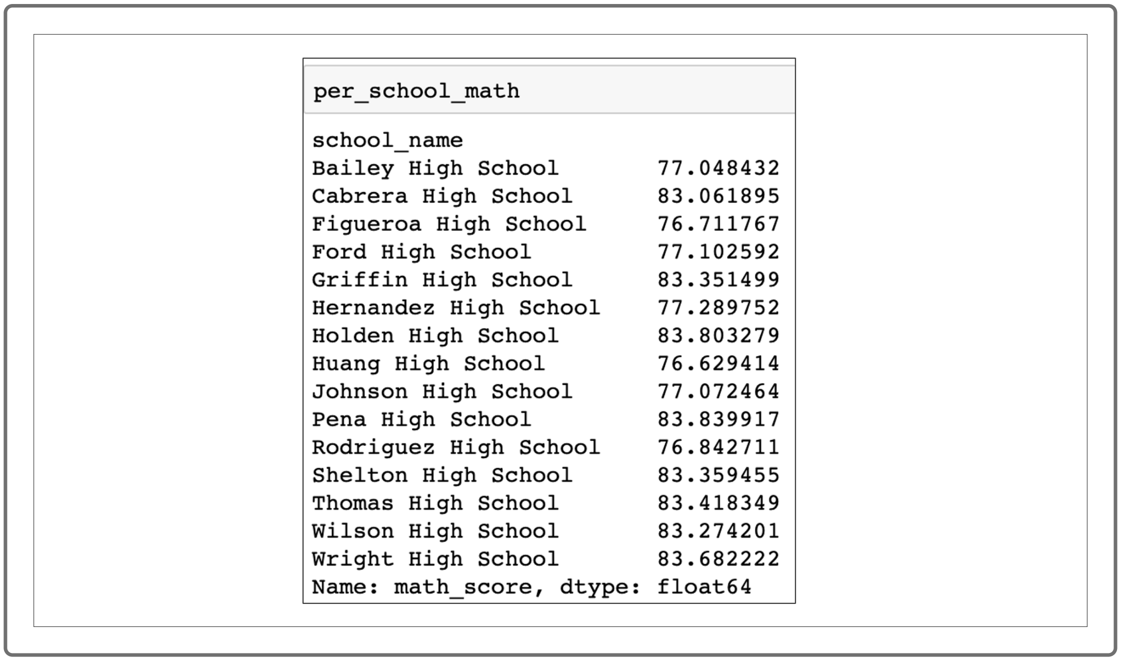
But we don't want all of this data in the school summary DataFrame, just the reading and math scores. To get the average math score and reading score for each school, we can add the `math_score` and `reading_score` columns at the end. Add the following code to a new cell and run the cell.

```
# Calculate the average test scores.
per_school_math = school_data_complete_df.groupby(["school_name"]).mean()["m

per_school_reading = school_data_complete_df.groupby(["school_name"]).mean()
```

When we run this cell and reference each Series, we get a Series like the other Series we have created, where the index is on the `school_name`, and the column is the average `math_score` or average `reading_score`.

The Series with the average math scores for each school will look like this:



```
per_school_math

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
Name: math_score, dtype: float64
```

The `per_school_reading` results will have the same format, with the column being the average `reading_score`.

NOTE

> For more information, read the **Pandas documentation on the groupby() function (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.groupby.html)** .