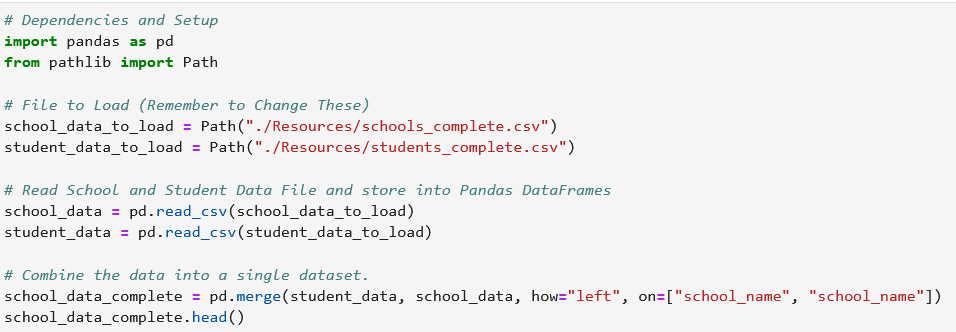
**PyCity Schools Analysis**

There are two documents with extension .csv called “schools\_complete.csv” and “students\_complete.csv”, where you have to analyze information of 15 Schools.

You will need showing information about student's math and reading scores, as well as various information on the schools they attend. Your responsibility is to aggregate the data to and showcase obvious trends in school performance.



Your final report should include each of the following:

**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

school\_count = len(school\_data\_complete.school\_name.unique())

* Total students

student\_count = school\_data\_complete.student\_name.count()

* Total budget

total\_budget = sum(school\_data\_complete.budget.unique())

* Average math score

average\_math\_score = school\_data\_complete.math\_score.mean()

* Average reading score

average\_reading\_score = school\_data\_complete.reading\_score.mean()

* % 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 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

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

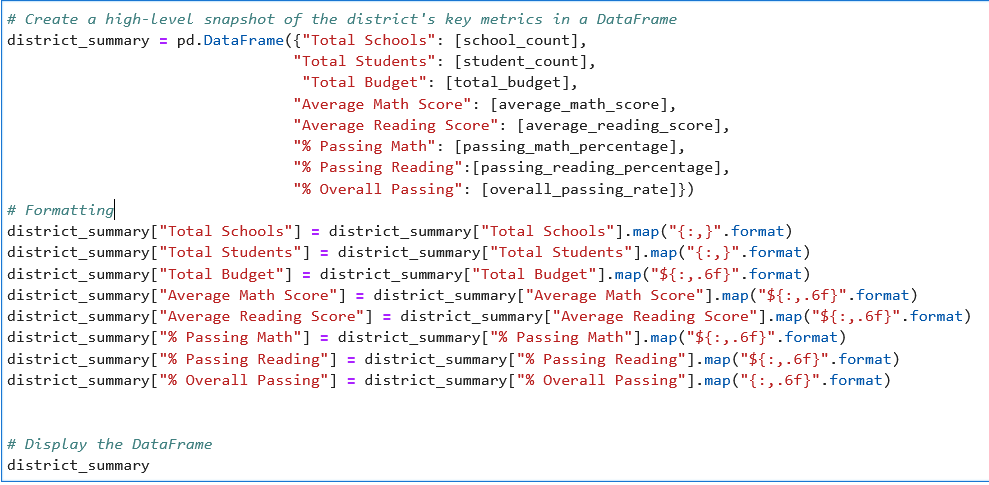
passing\_math\_reading\_count = school\_data\_complete[

(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

Creating DataFrame with information previously got.





**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

school\_names =school\_data\_complete.groupby(["school\_name"])

school\_types = school\_names["type"].first()

* Total students

per\_school\_counts = school\_names["Student ID"].count()

* Total school budget

per\_school\_budget = school\_names["budget"].mean()

* Per student budget

per\_school\_capita = per\_school\_budget / totalStudents

* Average math score

per\_school\_math = school\_names.math\_score.mean()

* Average reading score

per\_school\_reading = grouped\_school.reading\_score.mean()

* % passing math (the percentage of students who passed math)

per\_school\_passing\_math = school\_students\_passing\_math / per\_school\_counts \* 100

* % passing reading (the percentage of students who passed reading)

per\_school\_passing\_reading = school\_students\_passing\_reading / per\_school\_counts \* 100

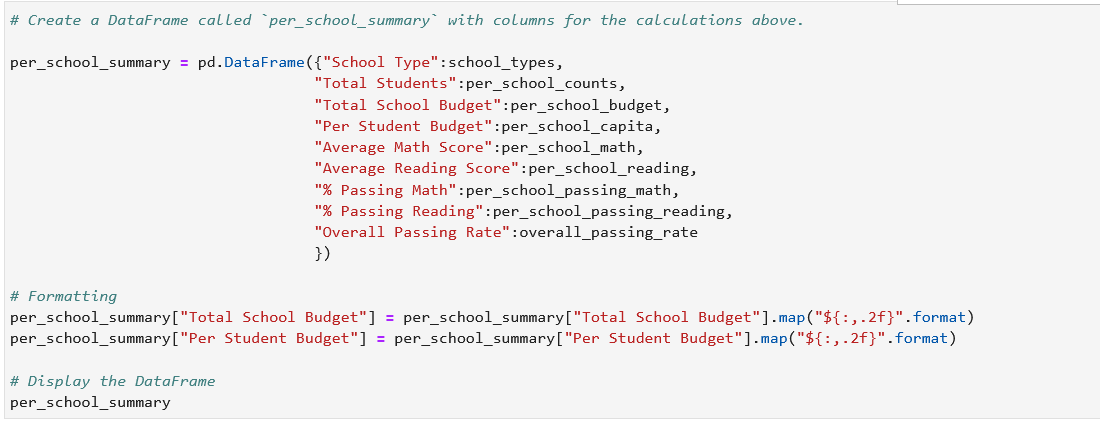
* % overall passing (the percentage of students who passed math AND reading)

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

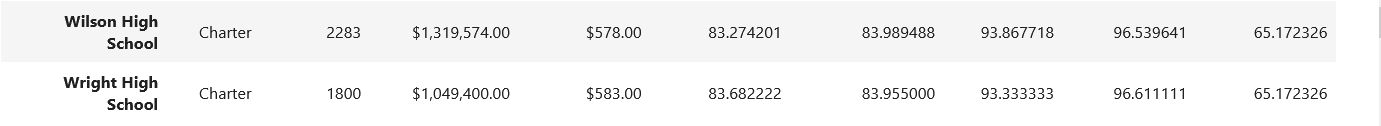
school\_students\_passing\_math\_and\_reading =

students\_passing\_math\_and\_reading.groupby(["school\_name"]).size()

overall\_passing\_rate = school\_students\_passing\_math\_and\_reading / per\_school\_counts \* 100







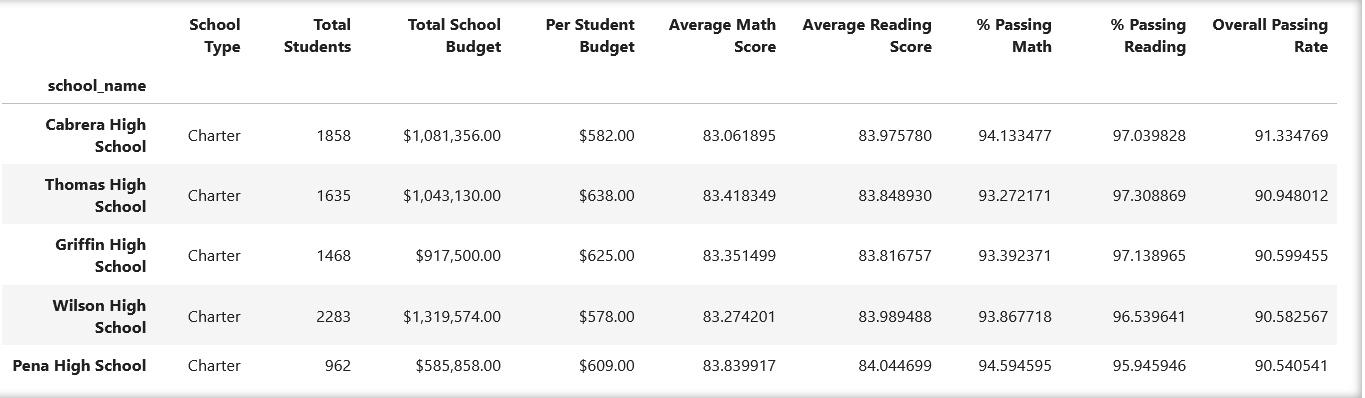
#### Highest-Performing Schools (by % Overall Passing)

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

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

top\_schools = per\_school\_summary.sort\_values(by=['Overall Passing Rate'], ascending=False)

top\_schools.head(5)



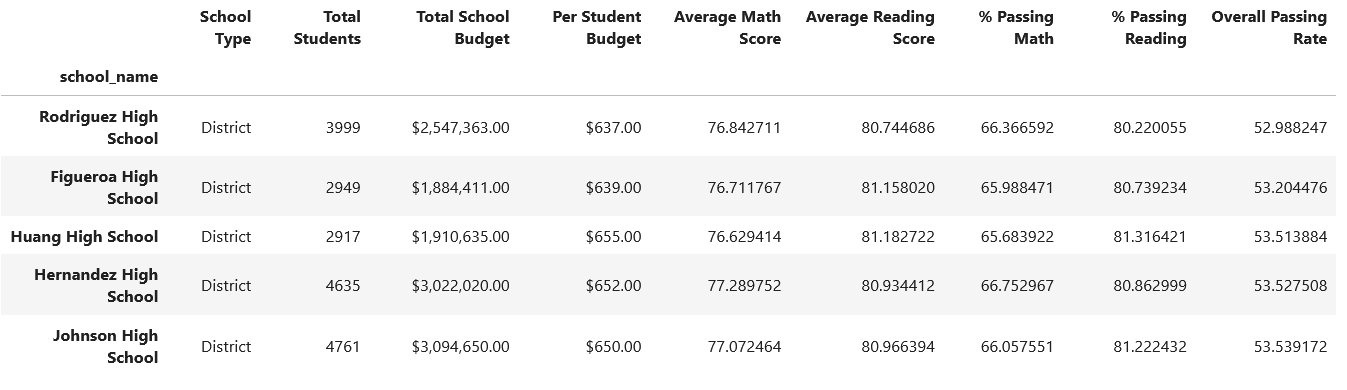
#### Lowest-Performing Schools (by % Overall Passing)

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

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

bottom\_schools = per\_school\_summary.sort\_values(by=['Overall Passing Rate'], ascending=True)

bottom\_schools.head(5)



#### 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.

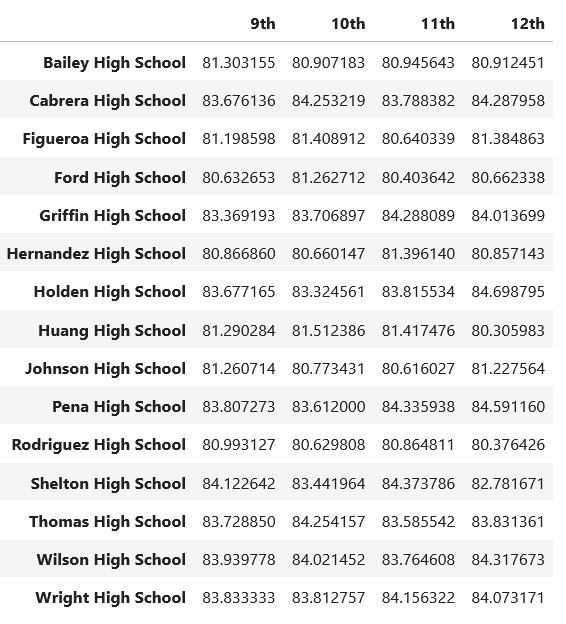
#### 

#### 

#### 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.





#### Scores by School Spending

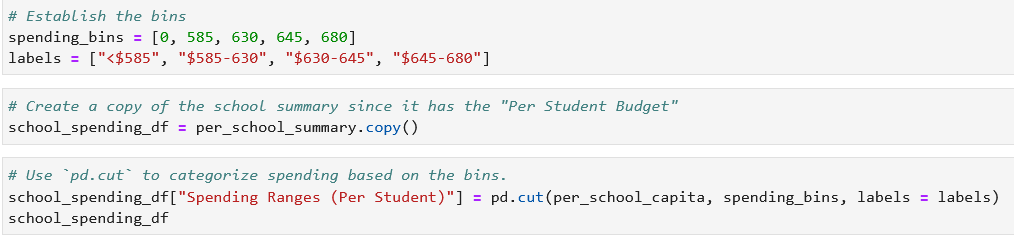
Create a table that breaks down school performance based on average spending ranges (per student).

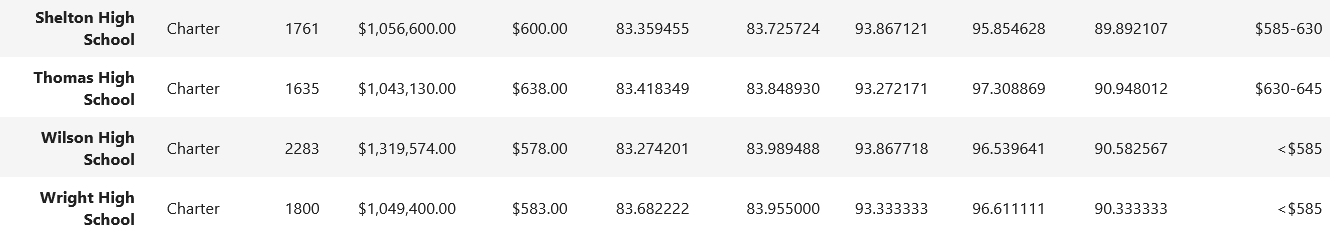
Use the code provided below to create four bins with reasonable cutoff values to group school spending.

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()

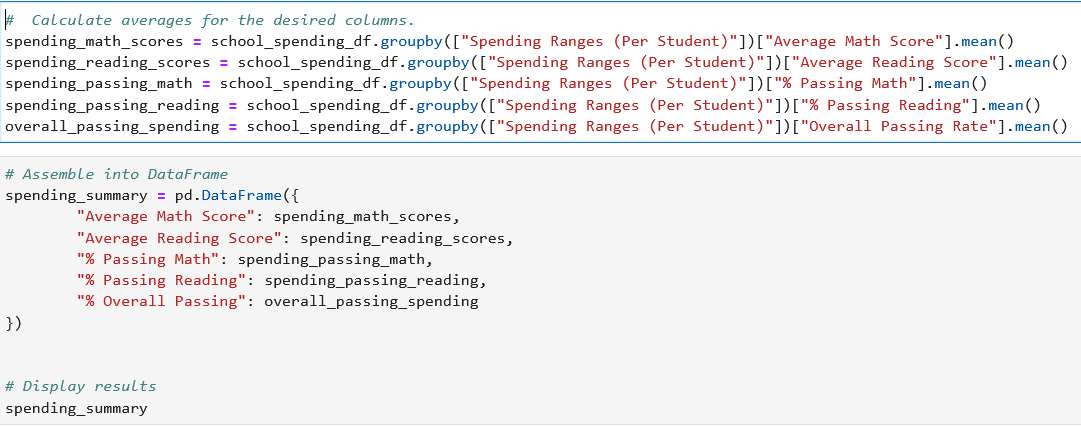
spending\_passing\_reading = school\_spending\_df.groupby(["Spending Ranges (Per Student)"])["% Passing Reading"].mean()

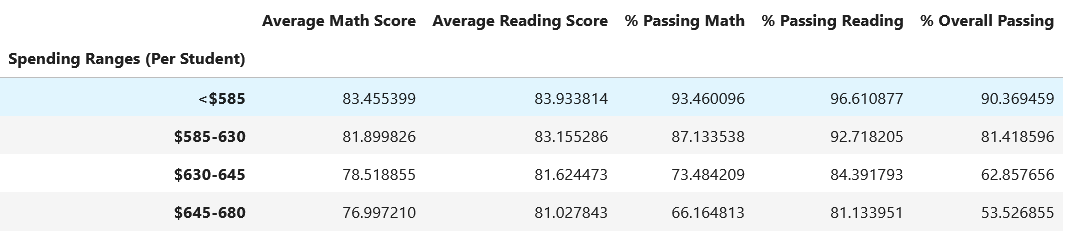
overall\_passing\_spending = school\_spending\_df.groupby(["Spending Ranges (Per Student)"])["% Overall Passing"].mean()

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)





#### Scores by School Size

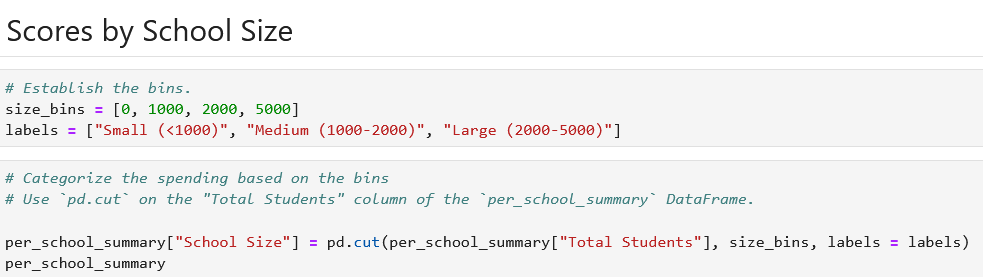
Use the following code to bin the per\_school\_summary.

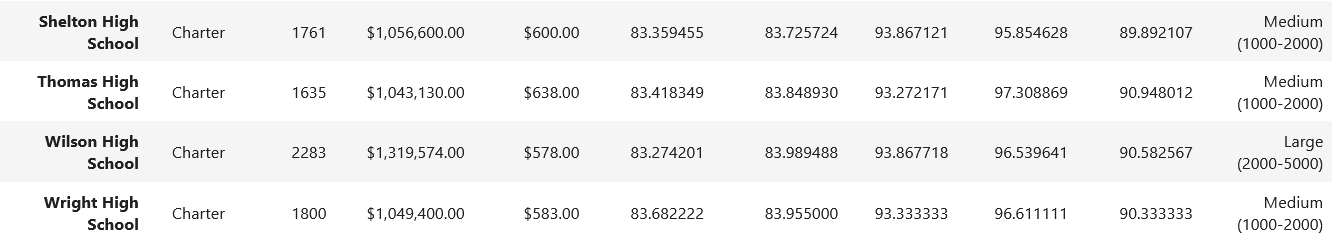
size\_bins = [0, 1000, 2000, 5000]

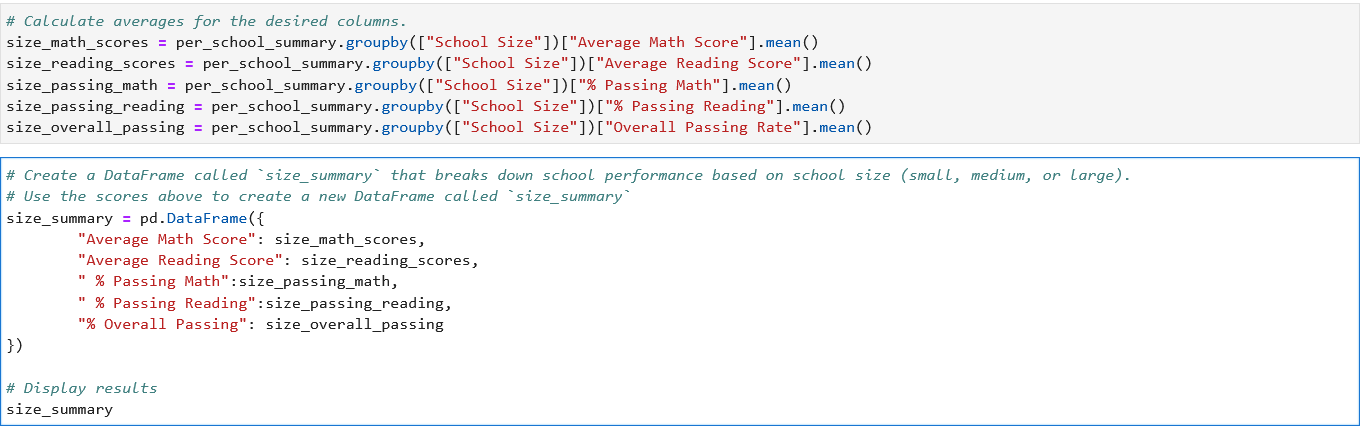
labels = ["Small (<1000)", "Medium (1000-2000)", "Large (2000-5000)"]

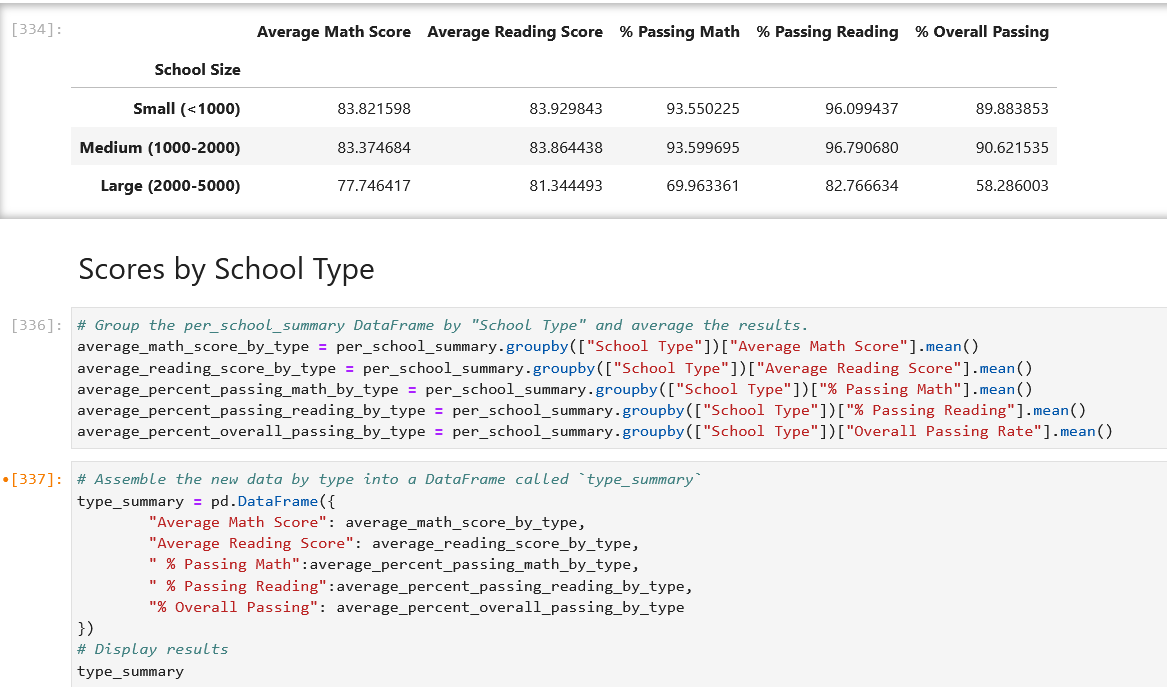
Use pd.cut on the "Total Students" column of the per\_school\_summary DataFrame.

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



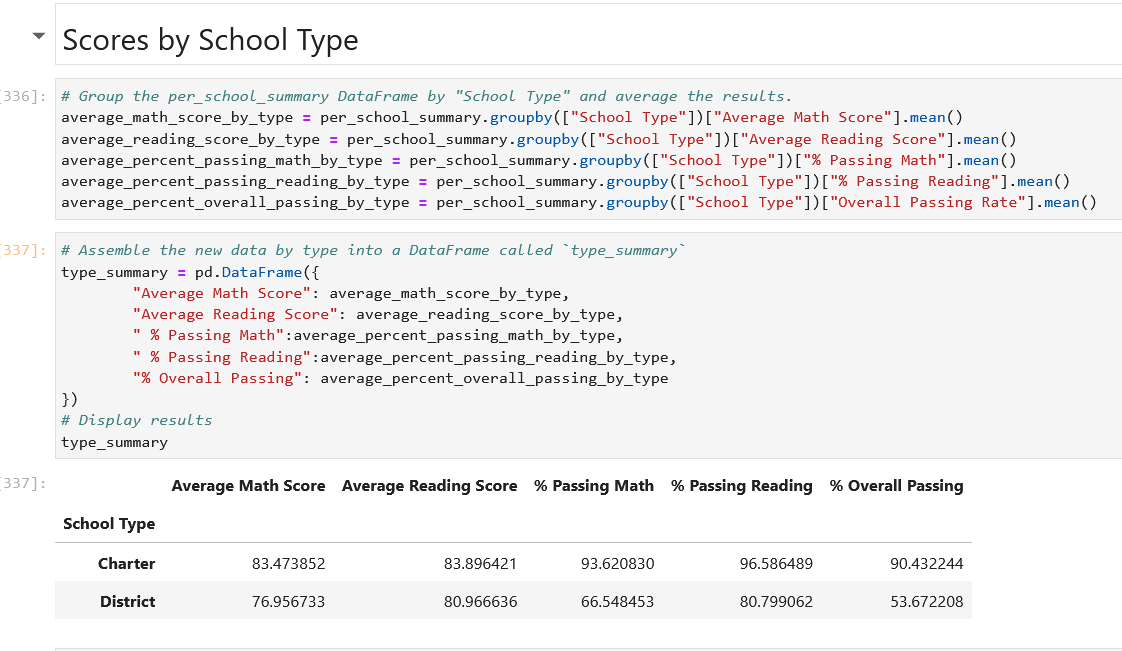




#### Scores by School Type

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".



References

Data generated by [Mockaroo, LLCLinks to an external site.](https://mockaroo.com/" \t "_blank), (2022). Realistic Data Generator. Data for this dataset was generated by edX Boot Camps LLC, and is intended for educational purposes only.