1. Importing and creating the data frames
   1. In this section we are importing both .csv files and merging them into a single dataframe then displaying the first 5 rows
2. Finding the total number of unique schools
   1. This section found and printed the number of unique schools by using the complete dataframe and the .nunique function for number of unique items in the school\_name column
3. Finding the total number of students
   1. In this section we used the .count function on the student\_name column for the complete dataframe then printed results
4. Finding the total budget
   1. For this we used only the school dataframe and the .sum function for the budget column then printed the results
5. Finding the average math score
   1. In this section we used the complete dataframe and the .mean function on the math\_score column then printed the results
6. Finding the average reading score
   1. In this section we used the complete dataframe and the .mean function on the reading\_score column then printed the results
7. Finding the # of students who pass math
   1. This section we use the complete dataframe and the math\_score column with the extra step of adding a condition of >=70 then the .count function with the student\_name column to find the total number of students with a 70 or greater score in math.
8. Finding the # of students who pass reading
   1. This section we use the complete dataframe and the reading\_score column with the extra step of adding a condition of >=70 then the .count function with the student\_name column to find the total number of students with a 70 or greater score in reading.
9. Finding the # of students who pass math & reading
   1. This section we create a new variable and set it equal to the complete dataframe and the math\_score column with the extra step of adding a condition of >=70, then the same for the reading score, then .count function with the student\_name column to find the total number of students with a 70 or greater score in math and reading.
10. District Summary
    1. Here we create a new dataframe with all the shown headers in “” the set them to all the previous variables we established in the previous code steps.
    2. From here we use formatting to format “Total Students” to include commas and the “Total Budget” Column to be dollar formatted with 2 decimal places
11. Finding the total number of school types
    1. This section found and printed the number of unique school types by using the complete dataframe and the .unique function for number of unique items in the school\_types column
12. Finding the students per school
    1. Using the complete dataframe we used the .value\_counts function in the school\_name column to count the number of times the school name appears since each line is a different student.
13. Finding budget per school and per student
    1. For the budget per school we use the school dataframe and the group function to group by school\_name column the .sum for the budget column of the groups
    2. For per capita/per student we take the new variable of budget per school we just found divided by the previous variable of students per school
14. Finding the average test scores per school
    1. Similar to the previous section we use the group function on the school\_name column but for the complete dataframe. Then the .mean function for the groups scores columns
15. Finding students per school passing math
    1. In this section we want to use the complete dataframe and setting the math\_scores column to the >=70 condition. From here we group by the school\_name then count function for students\_name for students that met the condition per school
16. Finding students per school passing reading
    1. In this section we want to use the complete dataframe and setting the reading\_scores column to the >=70 condition. From here we group by the school\_name then count function for students\_name for students that met the condition per school
17. Combined math & reading passes per school
    1. This provided code took the previous two variables and combined them into a new variable then used the .size function to calculate results
18. Passing Rates
    1. This section takes the 3 prior variables and divides them by the students per school variable. Then this is multiplied by 100 to get percents
19. School Summary
    1. Here we create a new dataframe with all the shown headers in “” the set them to the previous variables we established in the previous code steps 11 through 18.
    2. From here we use formatting to format “Total School Budget” Column and “Per Student Budget” to be dollar formatted with 2 decimal places
    3. “Per Student Budget” was commented out to fix future code
20. Highest Performing Schools
    1. Using the new school summary dataframe we can use the function .sort\_values set on the % Overall Passing column to sort the scores in a descending order
21. Bottom Performing schools
    1. Using the new school summary dataframe we can use the function .sort\_values set on the % Overall Passing column to sort the scores in a ascending order
22. Math scores by Grade
    1. Here we create a new variable per grade by setting the grade column equal to the grade in the data. From here we can use that variable and group functions to group the grade with the school and their scores from the math\_score column then the .mean function to find the average math score per grade per school
23. Reading scores per grade
    1. Here we create a new variable per grade by setting the grade column equal to the grade in the data. From here we can use that variable and group functions to group the grade with the school and their scores from the reading\_score column then the .mean function to find the average reading score per grade per school
24. Scores by School Spending
    1. Here we use bins to group the rows by the budget per student. We establish this by making a copy of our school summary dataframe and pd.cut to cut out the “Per Student Budget” column into our bins with our set labels
25. Averages for spending bins
    1. We now use these bins to create new variables for each of the desired scores
    2. Using the group function we are able to group spending ranges per student with the column corresponding to our new variable and use the .mean function to get the requested average scores for each requested score.
    3. We then take these new variables and create a new dataframe with them
26. Schools by Size
    1. In this step we again use bins to group the schools based off the total number of students. We establish this by using our school summary dataframe and pd.cut to cut out the “Total Students” column into our bins with our set labels
27. Averages for size bins
    1. We now use these bins to create new variables for each of the desired scores
    2. Using the group function we are able to group schools by total student counts with the column corresponding to our new variable and use the .mean function to get the requested average scores for each requested score.
    3. We then take these new variables and create a new dataframe with them
28. Scores by school type
    1. In this section we use the group function and are able to group schools by total school type with the column corresponding to our new variable and use the .mean function to get the requested average scores for each requested score.
    2. We then take these new variables and create a new dataframe with them
29. Adding in Formatting
    1. Lastly we add the dollar formatting back in to “Per Student Budget” to our per school summary now that the code has ran successfully

In this full dataset we are able to completely break down our csv data and draw conclusions from the new analysis.

1. Charter Schools greatly outperform in all presented testing and passing data provided
2. Schools that have the lowest budget per student actually outperform all the higher budgets per student in all provided test scoring and passing data.