How to Easily Calculate Group Sums in Pandas

Authored by

stats writer

December 3, 2025

The Pandas GroupBy Sum operation is a powerful way to quickly summarize data and aggregate results from a dataframe. It allows you to split a dataframe into groups based on certain criteria and then apply a function such as sum, mean, or count to each group. This can be done in one line of code using the groupby() and sum() functions. Examples are provided to demonstrate the syntax of this operation and how it can be used to answer different types of questions.

You can use the following basic syntax to find the sum of values by group in pandas:

df.groupby().sum().reset_index()

The following examples show how to use this syntax in practice with the following pandas DataFrame:

import pandas as pd

#create DataFrame
df = pd.DataFrame({'team': ,
'position': ,
'points': ,
'rebounds': })

#view DataFrame
df

team position points rebounds
0 A G 25 11
1 A G 17 8
2 A F 14 10
3 A C 9 6
4 B G 12 6
5 B F 9 5
6 B F 6 9
7 B C 4 12

Example 1: Group by One Column, Sum One Column

The following code shows how to group by one column and sum the values in one column:

#group by team and sum the points
df.groupby().sum().reset_index()

team points
0 A 65
1 B 31

From the output we can see that:

The players on team A scored a sum of **65** points.
The players on team B scored a sum of **31** points.

Example 2: Group by Multiple Columns, Sum Multiple Columns

The following code shows how to group by multiple columns and sum multiple columns:

#group by team and position, sum points and rebounds
df.groupby().sum().reset_index()

team position points rebounds
0 A C 9 6
1 A F 14 10
2 A G 42 19
3 B C 4 12
4 B F 15 14
5 B G 12 6

From the output we can see that:

The players on team A in the 'C' position scored a sum of **9** points and **6** rebounds.
The players on team A in the 'F' position scored a sum of **14** points and **10** rebounds.
The players on team A in the 'G' position scored a sum of **42** points and **19** rebounds.

And so on.

Note that the **reset_index()** function prevents the grouping columns from becoming part of the index.

For example, here's what the output looks like if we don't use it:

#group by team and position, sum points and rebounds
df.groupby().sum()

points rebounds
team position

A C 9 6
F 14 10
G 42 19
B C 4 12
F 15 14
G 12 6

Depending on how you'd like the results to appear, you may or may not choose to use the **reset_index()** function.

The following tutorials explain how to perform other common grouping operations in pandas: