How can I append multiple Pandas DataFrames in Python?

Authored by

stats writer

June 28, 2024

Appending multiple Pandas DataFrames in Python can be achieved by using the `concat()` function from the Pandas library. This function takes in a list of DataFrames and combines them into a single DataFrame by appending the rows of each DataFrame one after another. The resulting DataFrame will have all the columns from the original DataFrames and the rows will be in the same order as the DataFrames in the list. For example, if we have two DataFrames with the same columns, like the ones shown in the example, we can use `concat()` to create a new DataFrame with all the rows from both DataFrames. In this case, the resulting DataFrame will have 6 rows and the columns will remain the same, as shown in the desired result.

Append Multiple Pandas DataFrames (With Example)

You can use the following basic syntax to append multiple pandas DataFrames at once:

import pandas as pd

#append multiple DataFrames
df_big = pd.concat(, ignore_index=True)

This particular syntax will append **df1**, **df2**, and **df3** into a single pandas DataFrame called **df_big**.

The following example shows how to use this syntax in practice.

Example 1: Append Multiple Pandas DataFrames at Once

The following code shows how to append multiple pandas DataFrames at once:

import pandas as pd

#create three DataFrames
df1 = pd.DataFrame({'player': ,
'points':})

df2 = pd.DataFrame({'player': ,
'points':})

df3 = pd.DataFrame({'player': ,
'points':})

#append all DataFrames into one DataFrame
df_big = pd.concat(, ignore_index=True)

#view resulting DataFrameprint(df_big)

player points
0 A 12
1 B 5
2 C 13
3 D 17
4 E 27
5 F 24
6 G 26
7 H 27
8 I 27
9 J 12
10 K 9
11 L 5
12 M 5
13 N 13
14 O 17

The result is one big DataFrame that contains all of the rows from each of the three individual DataFrames.

The argument **ignore_index=True** tells pandas to ignore the original index numbers in each DataFrame and to create a new index that starts at 0 for the new DataFrame.

For example, consider what happens when we don't use **ignore_index=True** when stacking the following two DataFrames:

import pandas as pd

#create two DataFrames with indices
df1 = pd.DataFrame({'player': ,
'points':},
index=)

df2 = pd.DataFrame({'player': ,
'points':},
index=)

#stack the two DataFrames together
df_big = pd.concat()

#view resulting DataFrameprint(df_big)

player points
0 A 12
1 B 5
2 C 13
3 D 17
4 E 27
2 F 24
4 G 26
5 H 27
6 I 27
9 J 12

The resulting DataFrame kept its original index values from the two DataFrames.

In general, you should use **ignore_index=True** when appending multiple DataFrames unless you have a specific reason for keeping the original index values.

Additional Resources

How to Add an Empty Column to a Pandas DataFrame
How to Insert a Column Into a Pandas DataFrame
How to Export a Pandas DataFrame to Excel