How can I create a nested DataFrame in Pandas?

Authored by

stats writer

June 25, 2024

To create a nested DataFrame in Pandas, one can use the "from_dict" function and pass a nested dictionary as the data parameter. The outer keys of the dictionary will become the column labels, while the inner keys will become the row index labels. This allows for the creation of a multi-level DataFrame with hierarchical columns and rows. Additionally, the "concat" function can also be used to combine multiple DataFrames into a single nested DataFrame. By specifying the "keys" parameter with a list of labels, each DataFrame can be identified and accessed as a separate level within the nested DataFrame.

Create a Nested DataFrame in Pandas (With Example)

You can use the following syntax to nest multiple pandas DataFrames within another DataFrame:

df_all = pd.DataFrame({'idx':, 'dfs':})

This particular example nests three DataFrames (df1, df2, df3) inside a larger DataFrame called df_all.

You can then use the following syntax to access one of the specific nested DataFrames:

#display first nested DataFrame
print(df_all.iloc)

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

Example: Create Nested DataFrame in Pandas

Suppose we have three pandas DataFrames:

import pandas as pd

#create first DataFrame
df1 = pd.DataFrame({'item': ,
'sales': })

print(df1)

item sales
0 A 18
1 B 22
2 C 19
3 D 14
4 E 30

#create second DataFrame
df2 = pd.DataFrame({'item': ,
'sales': })

print(df2)

item sales

0 F 10

1 G 12

2 H 13

3 I 13

4 J 19

#create third DataFrame

df3 = pd.DataFrame({'item': ,

'sales': })

print(df3)

item sales

0 K 41

1 L 22

2 M 28

3 N 25

4 O 18

Now suppose that we would like to create one big DataFrame to hold all three of these DataFrames.

We can use the following syntax to do so:

df_all = pd.DataFrame({'idx':, 'dfs':})

We can then use the pandas iloc function to access specific nested DataFrames.

For example, we can use the following syntax to access the first nested DataFrame:

```
#display first nested DataFrame
print(df_all.iloc)
```

```
item sales
0 A 18
1 B 22
2 C 19
3 D 14
4 E 30
```

Or we could use the following syntax to access the second nested DataFrame:

```
#display second nested DataFrame
print(df_all.iloc)
```

```
item sales
0 F 10
1 G 12
```

2 H 13

3 I 13

4 J 19

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