Fictitious Names

Introduction:

This time you will create a data again

Special thanks to <u>Chris Albon</u> for sharing the dataset and materials. All the credits to this exercise belongs to him.

In order to understand about it go here.

Step 1. Import the necessary libraries

```
import pandas as pd
```

Step 2. Create the 3 DataFrames based on the following raw data

Step 3. Assign each to a variable called data1, data2, data3

```
data1 = raw_data_1
data2 = raw_data_2
data3 = raw_data_3
```

Step 4. Join the two dataframes along rows and assign all_data

```
all_data = pd.concat([pd.DataFrame(data1), pd.DataFrame(data2)])
```

Step 5. Join the two dataframes along columns and assing to all_data_col

```
all_data_col = pd.concat([pd.DataFrame(data1), pd.DataFrame(data2)], axis=1)
```

Step 6. Print data3

```
print(data3)

→ {'subject_id': ['1', '2', '3', '4', '5', '7', '8', '9', '10', '11'], 'te
```

Vui lòng giải thích lỗi này:

TypeError: Can only merge Series or Data

Hypothesis

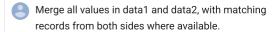
The error message TypeError: Can only merge Series or DataFrame objects, a <class 'dict'> was passed indicates that the pd.merge function was called with arguments data1 and data2, which are of type dict. However, pd.merge expects either pandas.Series or pandas.DataFrame objects as input. The user likely forgot to convert these dictionaries into DataFrames before attempting the merge.

Suggested Changes

merged_data_inner = pd.merge(pd.DataFrame print(merged_data_inner)

Hãy thận trọng khi sử dụng mã

Xếp hạng câu trả lời này 🖒 Ӆ



◆ Okay, to merge all values in data1 and data2, with matching records from both sides where available, you should use the outer join. This will include all records from both DataFrames, regardless of whether there is a match in the subject_id column. Non-matching values will be filled with NaN.

merged_data_outer = pd.merge(pd.DataFrame print(merged_data_outer)

Hãy thận trọng khi sử dụng mã

⊕ □

Explanation:

- pd.DataFrame(data1) and
 pd.DataFrame(data2): Convert data1
 and data2 dictionaries into DataFrames.
- on='subject_id': Specifies that the merge should be performed based on the 'subject_id' column.
- how='outer': Performs a full outer join, including all records from both DataFrames.

This code will achieve the desired outcome of merging all values in data1 and data2 while preserving records from both sides, even if they don't have matching 'subject_id' values.

Xếp hạng câu trả lời này 🖒 🗇

Step 7. Merge all_data and data3 along the subject_id value

merged_data = pd.merge(all_data, data3, on='subject_id')
print(merged_data)

				_	
		subject_id	first_name	last_name	test_id
	0	1	Alex	Anderson	51
	1	2	Amy	Ackerman	15
	2	3	Allen	Ali	15
	3	4	Alice	Aoni	61
	4	5	Ayoung	Atiches	16
	5	4	Billy	Bonder	61
	6	5	Brian	Black	16
	7	7	Bryce	Brice	14
	8	8	Betty	Btisan	15

Step 8. Merge only the data that has the same 'subject_id' on both data1 and data2

merged_data_inner = pd.merge(pd.DataFrame(data1), pd.DataFrame(data2), on='su
print(merged_data_inner)

Step 9. Merge all values in data1 and data2, with matching records from both sides where available.

merged_data_outer = pd.merge(pd.DataFrame(data1), pd.DataFrame(data2), on='su
print(merged_data_outer)

_		subject_id	first_name_x	last_name_x	first_name_y	last_name_y
	0	1	Alex	Anderson	NaN	NaN
	1	2	Amy	Ackerman	NaN	NaN
	2	3	Allen	Ali	NaN	NaN
	3	4	Alice	Aoni	Billy	Bonder
	4	5	Ayoung	Atiches	Brian	Black
	5	6	NaN	NaN	Bran	Balwner
	6	7	NaN	NaN	Bryce	Brice
	7	8	NaN	NaN	Betty	Btisan

Nhập câu lệnh tại đây

+