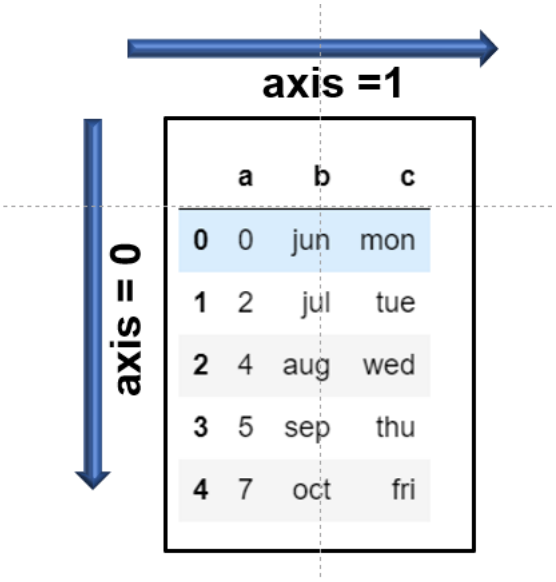
PANDAS : DataFrames Concat() VS Append()

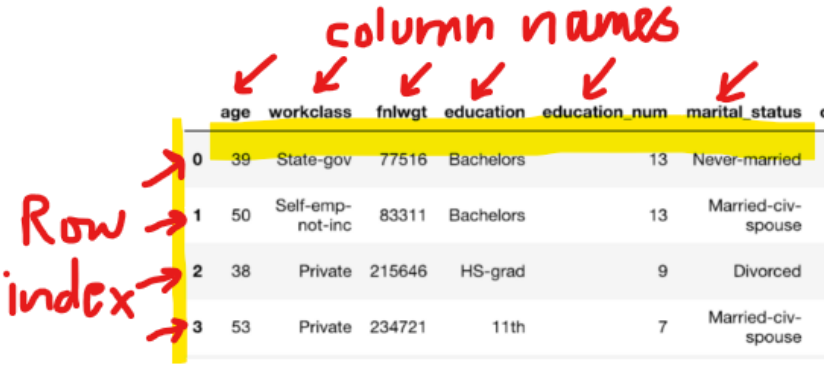
DataFrame axis :



What is a DataFrame Index ?

A DataFrame is **a Python object that stores data in a row-and-column format**

Data Frame Index are the Column Labels and Row Indices. They are nothing but the addresses through which a particular column or a Row is identified.



So, when we say ignore_index = True , all the labels are reset to Pandas default Index labels.

(And then later, we will have to add the column names again)

Importing necessary libraries:

```
In [44]:
1 import pandas as pd
2 import numpy as np
```

Creating DataFrames from Dictionaries:

```
In [47]:
1 import pandas as pd
2 df1 = pd.DataFrame({'a': [1,3,6,8,9], 'b': ['red', 'green', 'blue', 'white', 'black']})
3 df2 = pd.DataFrame({'a': [0,2,4,5,7], 'b': ['jun', 'jul', 'aug', 'sep', 'oct']})
4 df3 = pd.DataFrame({'a': [0,2,4,5,7], 'b': ['jun', 'jul', 'aug', 'sep', 'oct'] , 'c': ['mon', 'tue', 'wed', 'thu', 'fri']})
```

```
In [46]:
1 df4 = pd.DataFrame({'a': [1,3,6,8,9], 'b': ['red', 'green', 'blue', 'white', 'black']})
2 df5 = pd.DataFrame({'w': [0,2,4,5,7,10,11], 'z': ['jun', 'jul', 'aug', 'sep', 'oct','nov' ,'dec']})
3 df6 = pd.DataFrame({'r': [0,2,4,5,7], 's': ['jun', 'jul', 'aug', 'sep', 'oct'] , 'c': ['mon', 'tue', 'wed', 'thu', 'fri']})
```

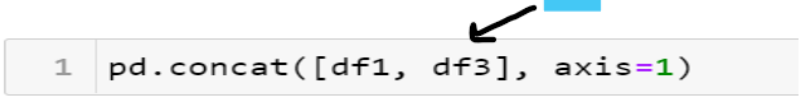
concat()

Use pandas.concat() to concatenate/merge two or multiple pandas DataFrames across rows or columns.

When you concat() two pandas DataFrames on rows, it creates a new Dataframe containing all rows of two DataFrames . Under the hood it is appending one DataFrame with another.

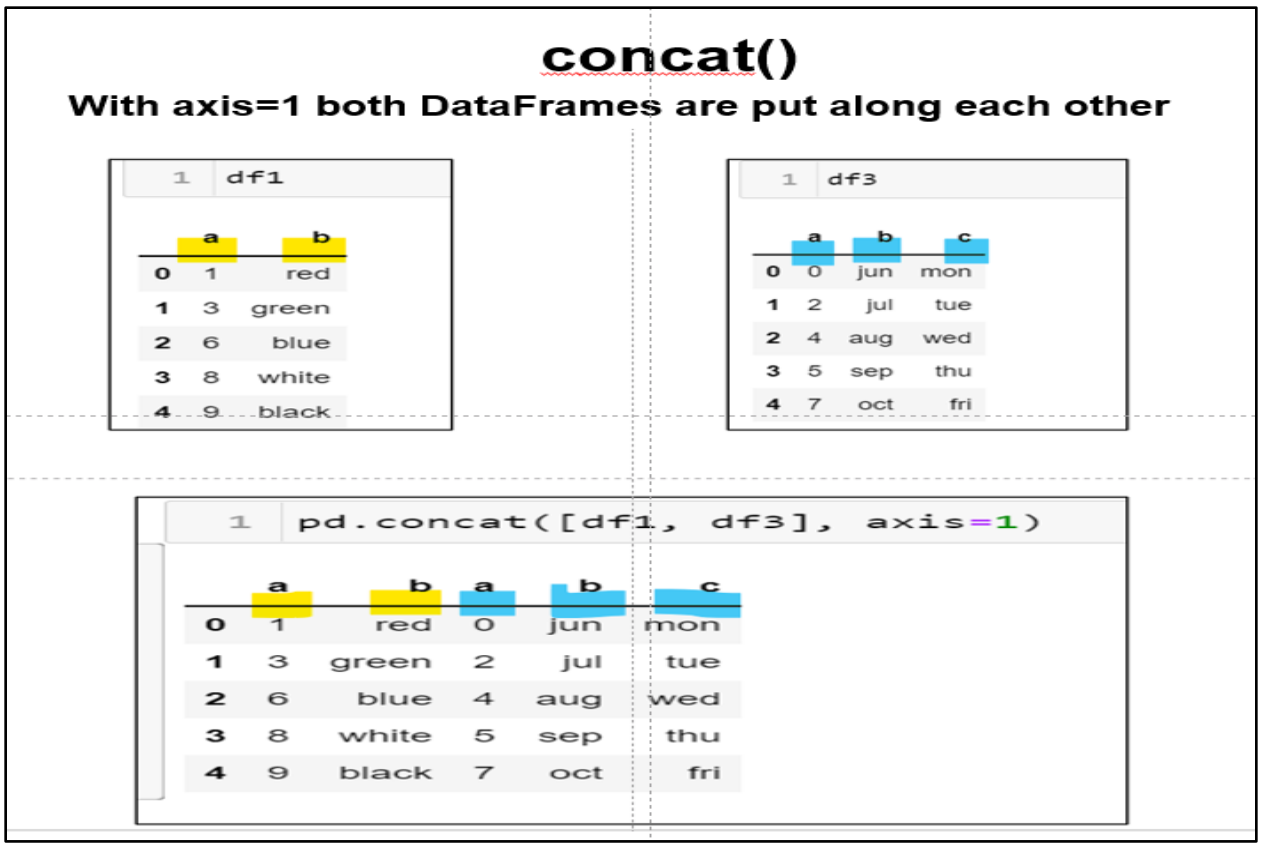
When you use concat() on columns it performs the join operation.

Pandas.concat is a function that takes a list of objects as input.



```
1 pd.concat([df1, df3], axis=1)
```

In the below example since we did not set “ignore_index = True” , the Column labels are repeated.



Below, **ignore_index =True** has reset the column Names

```
In [45]:  pd.concat([df1, df3], axis=1 ,ignore_index = True)
```

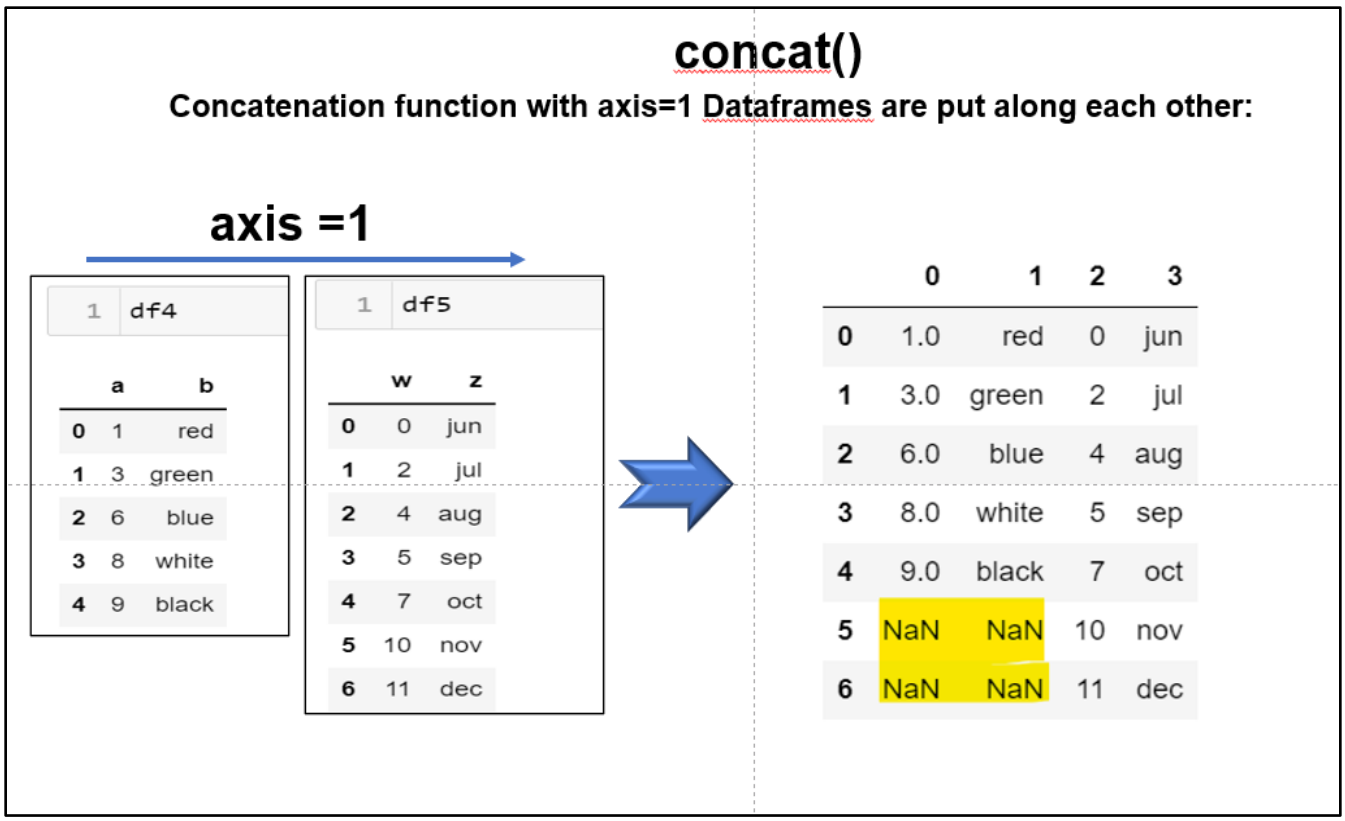
Out[45]:

	0	1	2	3	4
0	1	red	0	jun	mon
1	3	green	2	jul	tue
2	6	blue	4	aug	wed
3	8	white	5	sep	thu
4	9	black	7	oct	fri

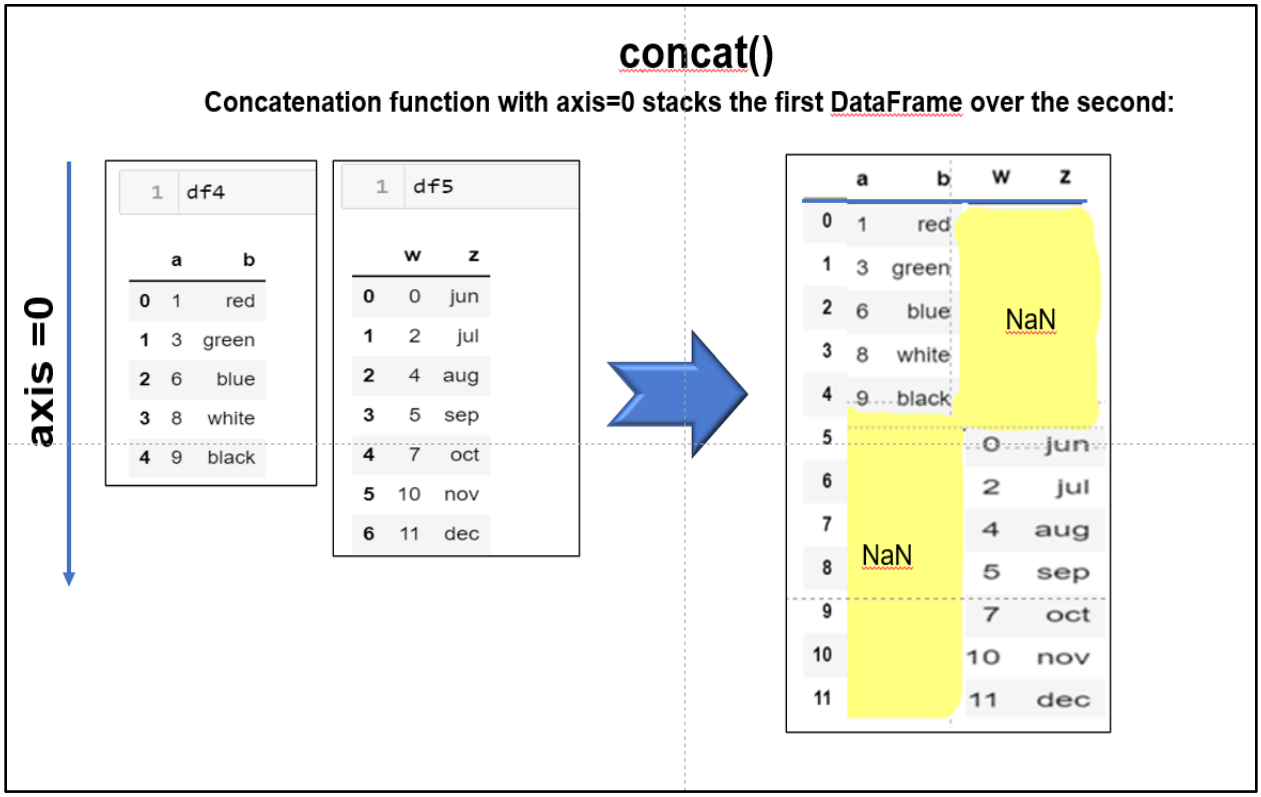
For unequal no. of columns in the data frame, a non-existent value in one of the dataframe will be filled with NaN values.

```
In [40]:  pd.concat([df4, df5], axis=1 , ignore_index = True)
```

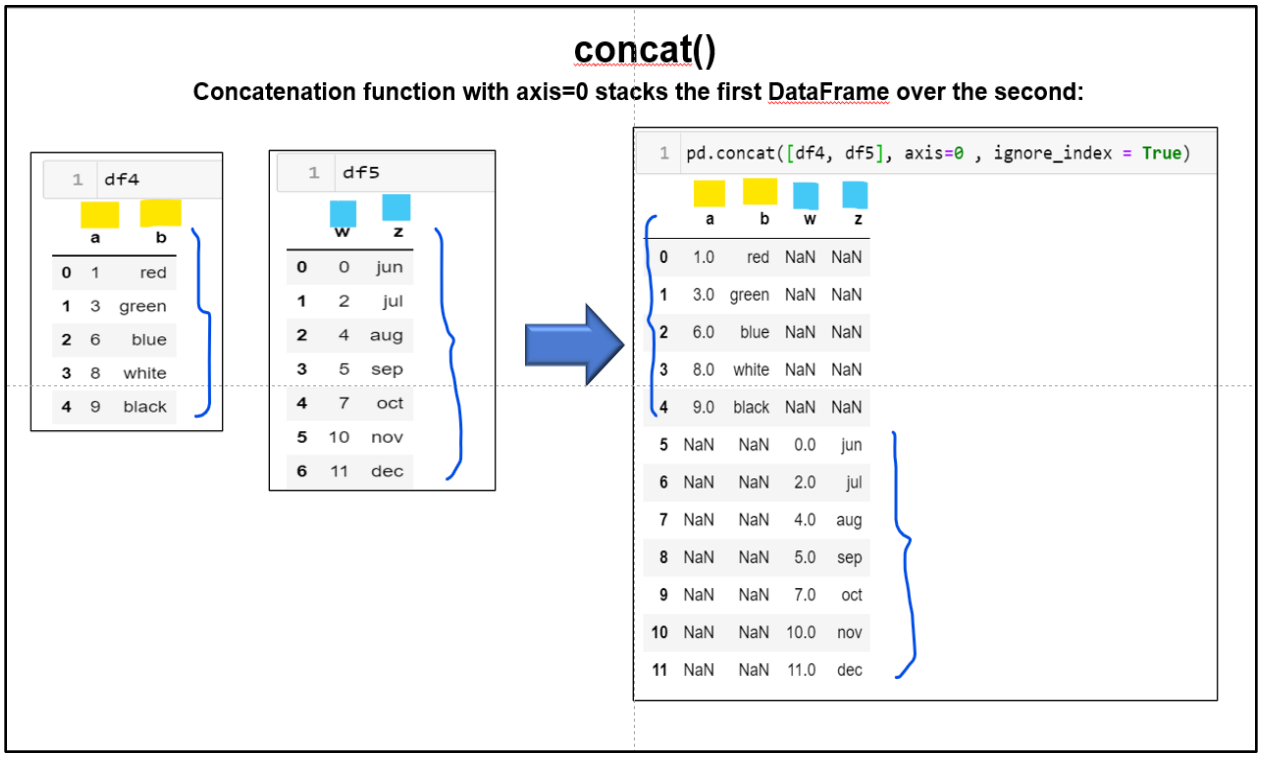
axis 1 means , perform concat along the columns



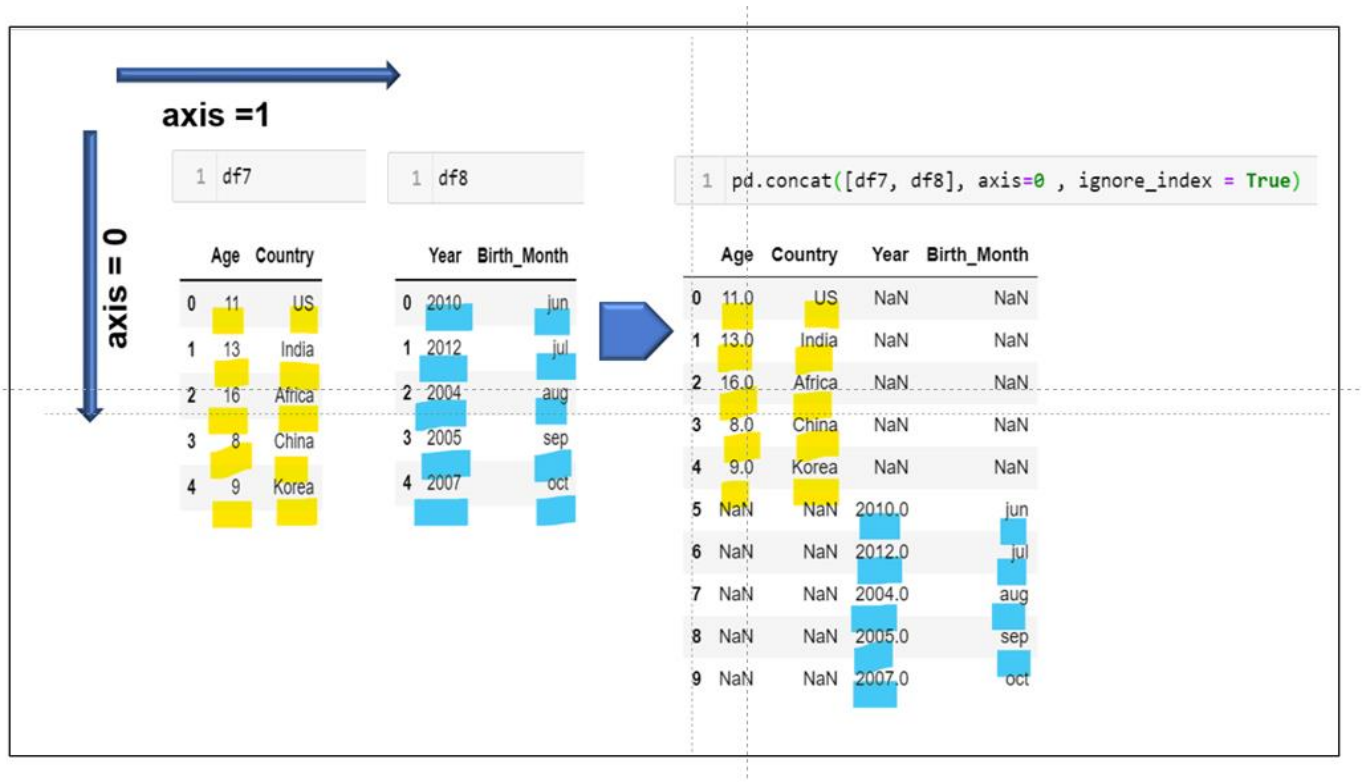
axis 0 means →perform concat across rows or vertically



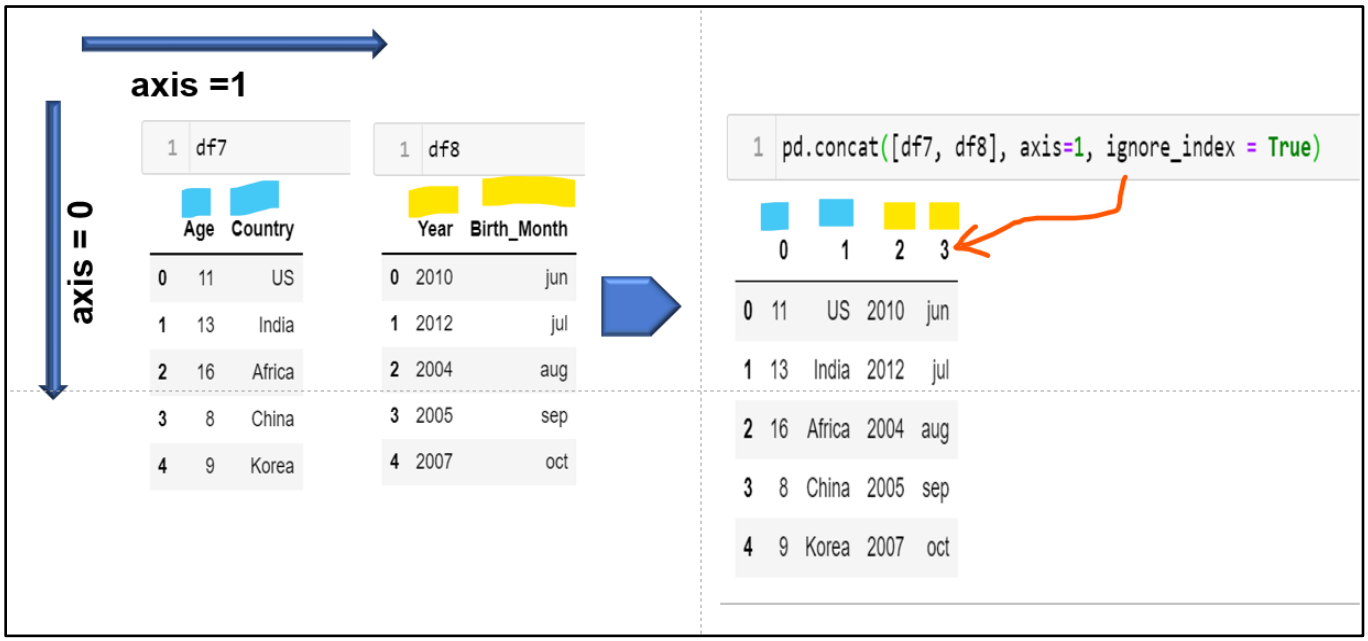
Below : since we are Concatenating along axis = 0 , that is , vertically along the Rows , the Row indices have been Reset , when we set “ignore_index = True”



Another example : Concat Vertically on axis 0 , ignore Index



Alternatively if we concat horizontally and say ignore_index = True , then the Column names will be lost:



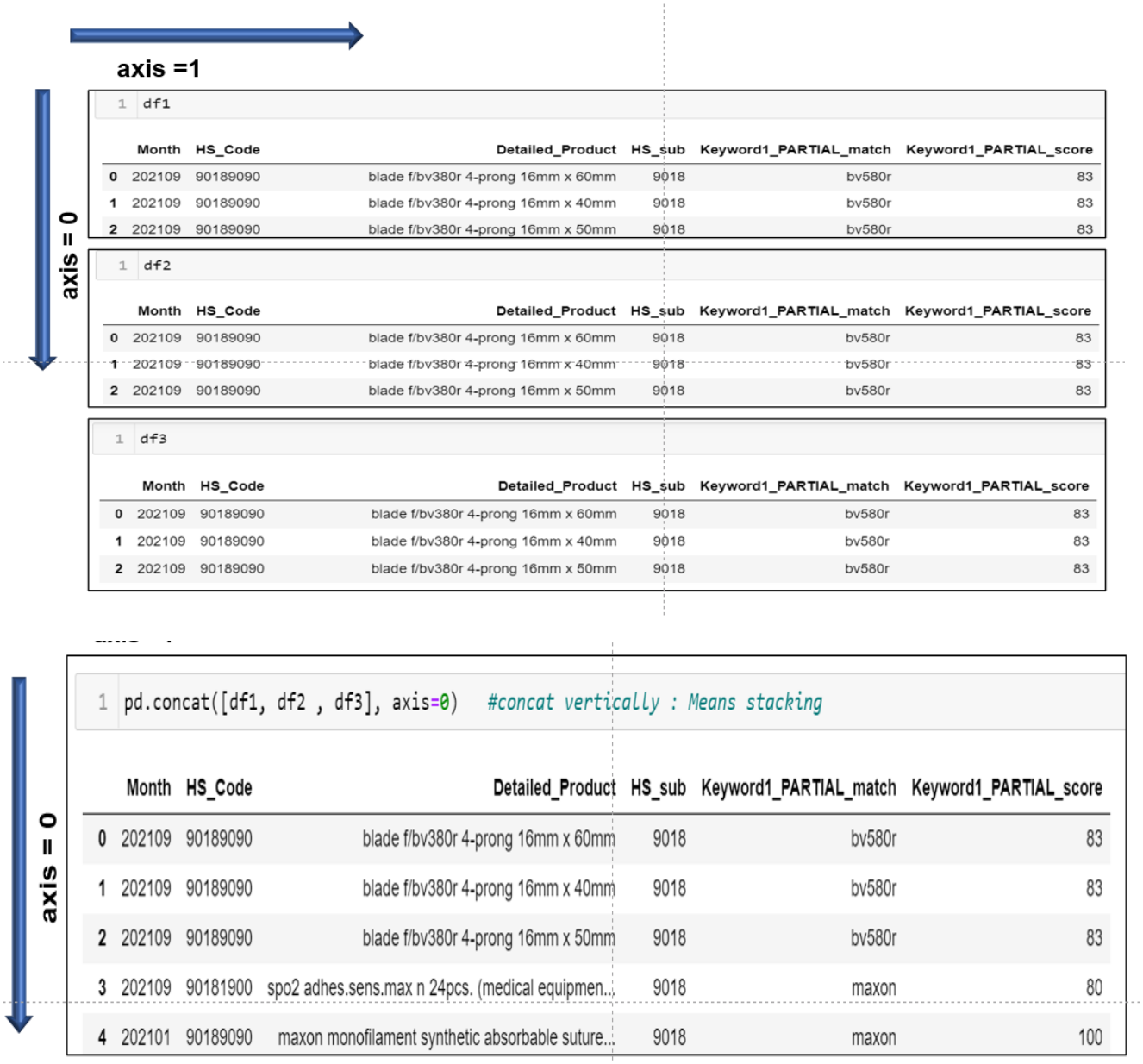
Another example :

Read the dfs :

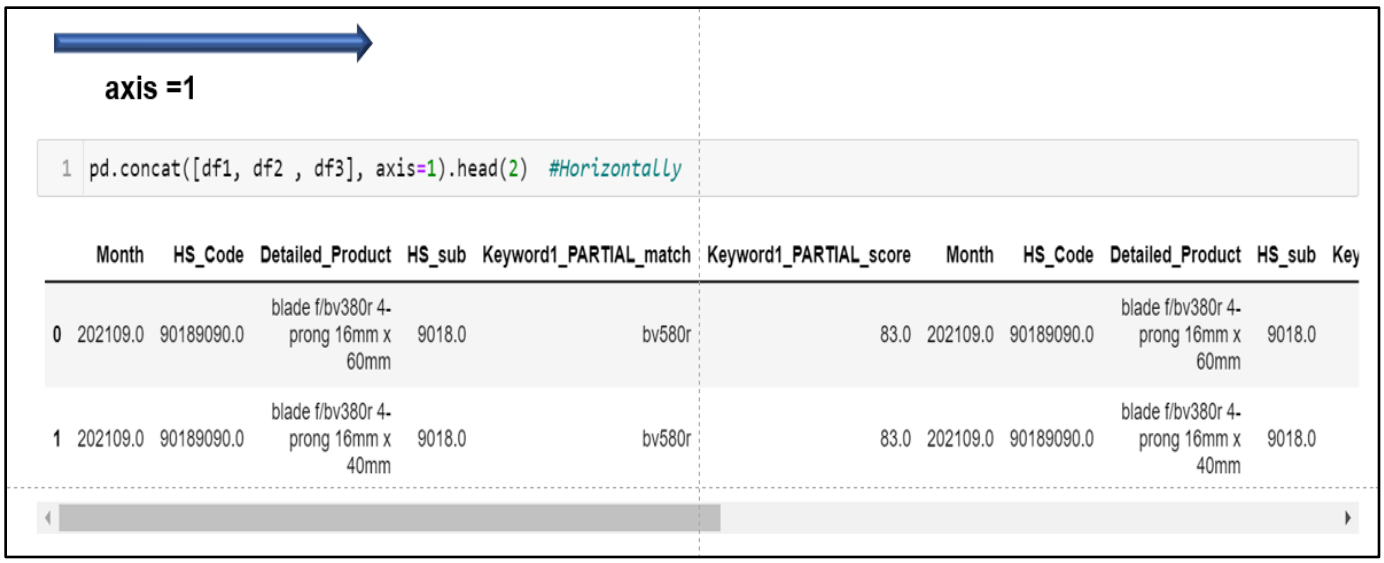
```
df1 = pd.read_excel("df1.xlsx")
```

```
df2 = pd.read_excel("df2.xlsx")
```

```
df3 = pd.read_excel("df3.xlsx")
```



When we concat on axis 1 , that is along the columns, observe the repeated column names

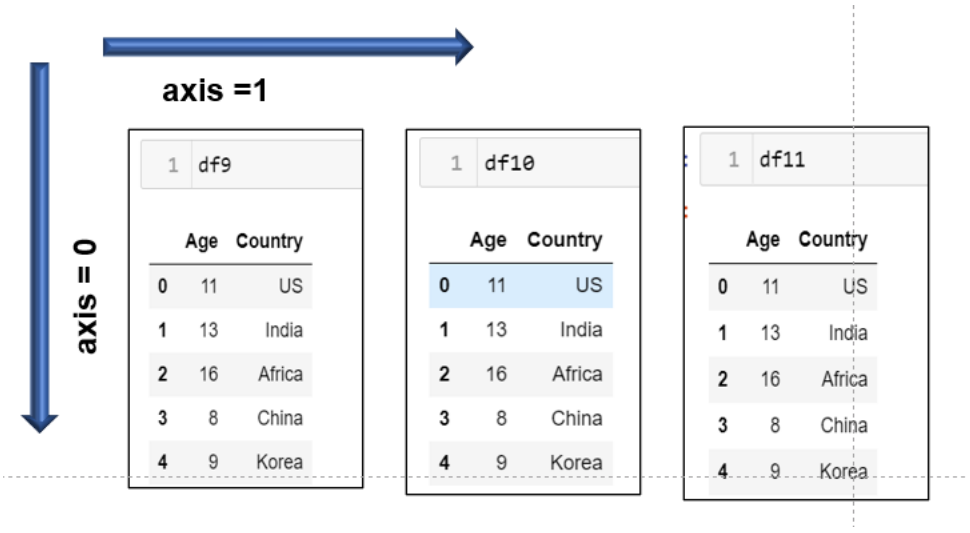


	Month	HS_Code	Detailed_Product	HS_sub	Keyword1_PARTIAL_match	Keyword1_PARTIAL_score	Month	HS_Code	Detailed_Product	HS_sub	Key
0	202109.0	90189090.0	blade f/bv380r 4-prong 16mm x 60mm	9018.0	bv580r	83.0	202109.0	90189090.0	blade f/bv380r 4-prong 16mm x 60mm	9018.0	
1	202109.0	90189090.0	blade f/bv380r 4-prong 16mm x 40mm	9018.0	bv580r	83.0	202109.0	90189090.0	blade f/bv380r 4-prong 16mm x 40mm	9018.0	

Concatenating DataFrames from List of dfs

Concat along axis 0 (that is vertically)

```
1 dflist = [df9 , df10 , df11]
```



	Age	Country
0	11	US
1	13	India
2	16	Africa
3	8	China
4	9	Korea

	Age	Country
0	11	US
1	13	India
2	16	Africa
3	8	China
4	9	Korea

	Age	Country
0	11	US
1	13	India
2	16	Africa
3	8	China
4	9	Korea

```
In [116]:  1  pd.concat([dfs for dfs in dflist], axis= 0 , ignore_index=True)
```

Out[116]:

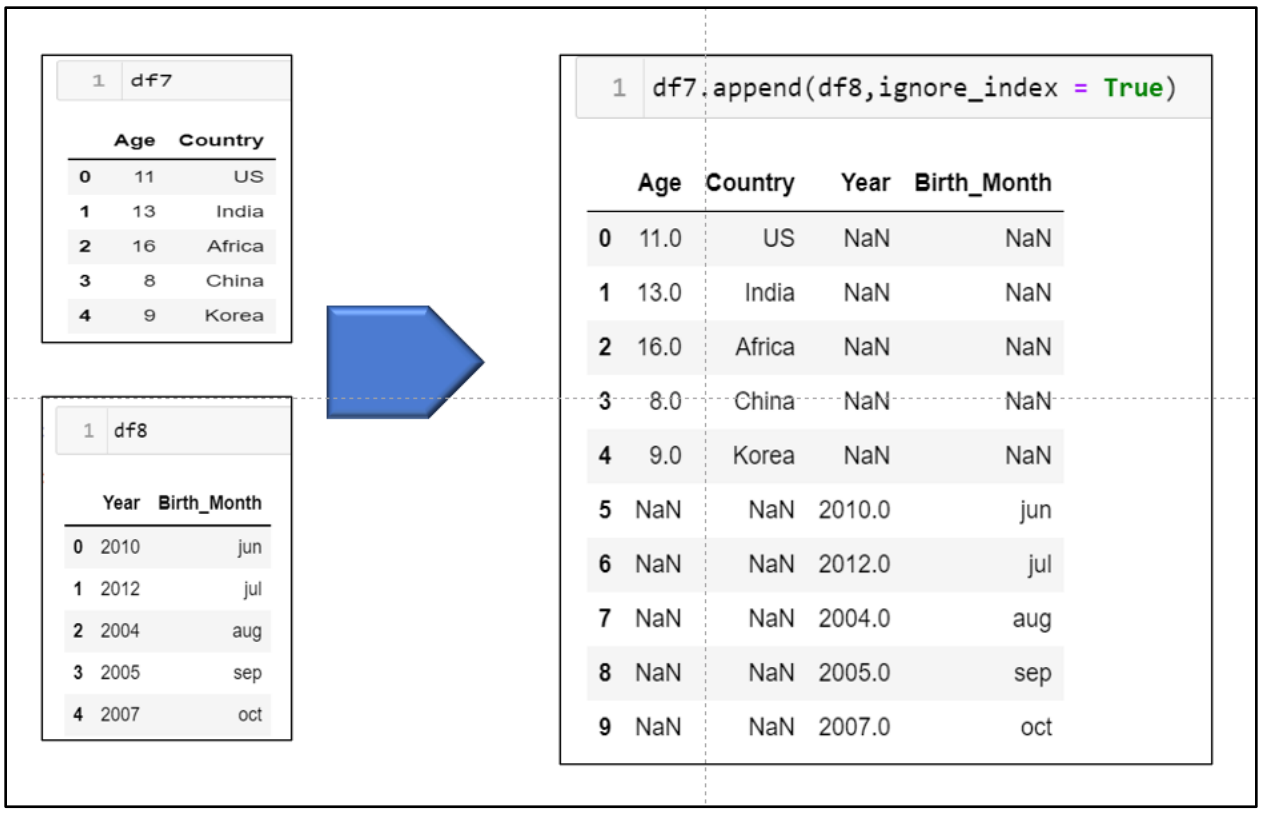
	Age	Country
0	11	US
1	13	India
2	16	Africa
3	8	China

append()

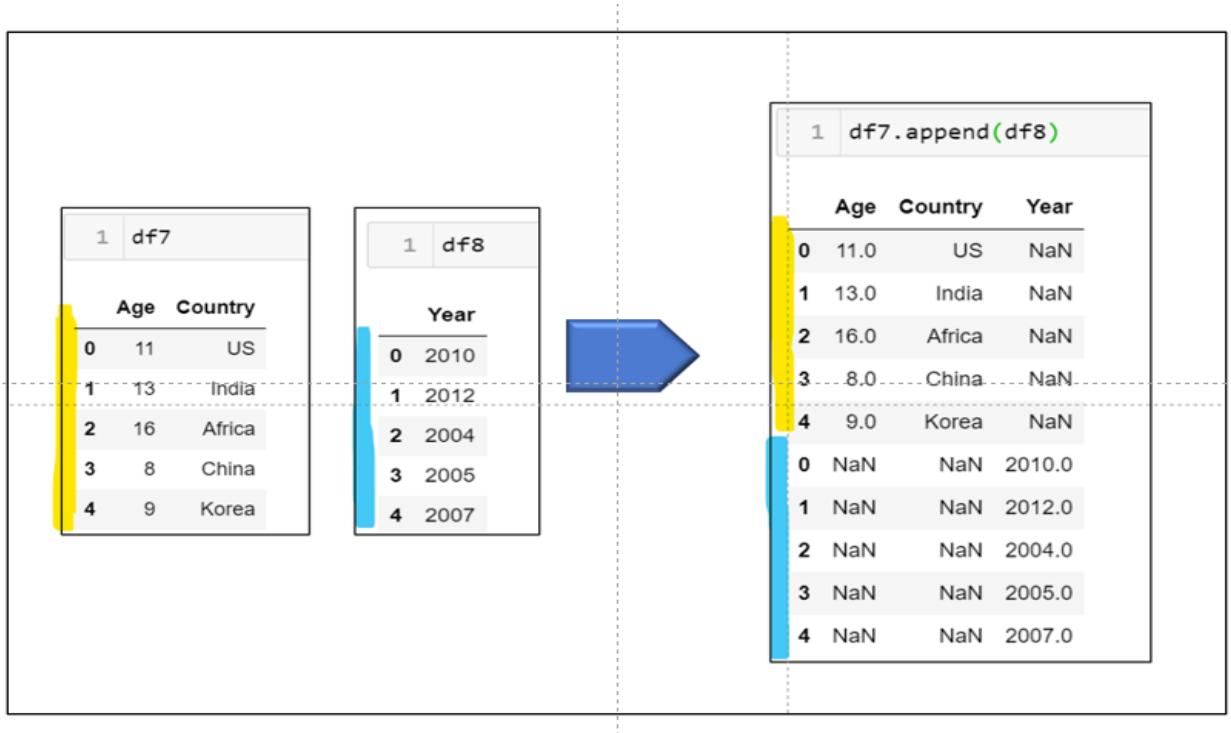
dataframe.append() function is used to append rows of other dataframe to the end of the given dataframe, returning a new dataframe object. Columns not in the original dataframes are added as new columns and the new cells are populated with NaN value.

ignore_index : If True, do not use the index labels. (labels → Column Names or Row Labels)

Alternatively : Append is the specific case of concat, which concats the second dataframe's records at the end of first dataframe.



If we do not set ignore_index = True :



Append( ) method does not change the original DataFrame

The difference here is Append takes one dataframe at a time and appends it to the first dataframe vertically whereas concat can take a whole list of data frames and append either horizontally or vertically depending on the axis specified.

……

Thank you

Aisha Khalid