

Module 02

# Merging, Joining, and Concatenating

Data Science Developer

# Create DataFrames

```
In [1]: import pandas as pd
```

```
In [2]: df1 = pd.DataFrame({'A': ['A0', 'A1', 'A2', 'A3'],  
                             'B': ['B0', 'B1', 'B2', 'B3'],  
                             'C': ['C0', 'C1', 'C2', 'C3'],  
                             'D': ['D0', 'D1', 'D2', 'D3']},  
                             index=[0, 1, 2, 3])
```

```
In [3]: df2 = pd.DataFrame({'A': ['A4', 'A5', 'A6', 'A7'],  
                             'B': ['B4', 'B5', 'B6', 'B7'],  
                             'C': ['C4', 'C5', 'C6', 'C7'],  
                             'D': ['D4', 'D5', 'D6', 'D7']},  
                             index=[4, 5, 6, 7])
```

```
In [4]: df3 = pd.DataFrame({'A': ['A8', 'A9', 'A10', 'A11'],  
                             'B': ['B8', 'B9', 'B10', 'B11'],  
                             'C': ['C8', 'C9', 'C10', 'C11'],  
                             'D': ['D8', 'D9', 'D10', 'D11']},  
                             index=[8, 9, 10, 11])
```

# The DataFrames

In [5]:

df1

Out[5]:

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3

In [6]:

df2

Out[6]:

	A	B	C	D
4	A4	B4	C4	D4
5	A5	B5	C5	D5
6	A6	B6	C6	D6
7	A7	B7	C7	D7

In [7]:

df3

Out[7]:

	A	B	C	D
8	A8	B8	C8	D8
9	A9	B9	C9	D9
10	A10	B10	C10	D10
11	A11	B11	C11	D11

# Concatenation

Concatenation basically glues together DataFrames. Keep in mind that dimensions should match along the axis you are concatenating on. You can use **pd.concat** and pass in a list of DataFrames to concatenate together:

```
In [8]: pd.concat([df1,df2,df3])
```

Out[8]:

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3
4	A4	B4	C4	D4
5	A5	B5	C5	D5
6	A6	B6	C6	D6
7	A7	B7	C7	D7
8	A8	B8	C8	D8
9	A9	B9	C9	D9
10	A10	B10	C10	D10
11	A11	B11	C11	D11

# Concatenation

```
In [9]: pd.concat([df1,df2,df3],axis=1)
```

```
Out[9]:
```

	A	B	C	D	A	B	C	D	A	B	C	D
0	A0	B0	C0	D0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	A1	B1	C1	D1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	A2	B2	C2	D2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	A3	B3	C3	D3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	A4	B4	C4	D4	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	A5	B5	C5	D5	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	A6	B6	C6	D6	NaN	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN	A7	B7	C7	D7	NaN	NaN	NaN	NaN
8	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A8	B8	C8	D8
9	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A9	B9	C9	D9
10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A10	B10	C10	D10
11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A11	B11	C11	D11

# Create DataFrames

```
In [10]: left = pd.DataFrame({'key': ['K0', 'K1', 'K2', 'K3'],  
                             'A': ['A0', 'A1', 'A2', 'A3'],  
                             'B': ['B0', 'B1', 'B2', 'B3']})  
  
right = pd.DataFrame({'key': ['K0', 'K1', 'K2', 'K3'],  
                     'C': ['C0', 'C1', 'C2', 'C3'],  
                     'D': ['D0', 'D1', 'D2', 'D3']})
```

```
In [11]: left
```

```
Out[11]:
```

	key	A	B
0	K0	A0	B0
1	K1	A1	B1
2	K2	A2	B2
3	K3	A3	B3

```
In [12]: right
```

```
Out[12]:
```

	key	C	D
0	K0	C0	D0
1	K1	C1	D1
2	K2	C2	D2
3	K3	C3	D3

# Merging

The **merge** function allows you to merge DataFrames together using a similar logic as merging SQL Tables together. For example:

```
In [13]: pd.merge(left, right, how='inner', on='key')
```

Out[13]:

	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K1	A1	B1	C1	D1
2	K2	A2	B2	C2	D2
3	K3	A3	B3	C3	D3

# Merging

```
In [14]: left = pd.DataFrame({'key1': ['K0', 'K0', 'K1', 'K2'],
                              'key2': ['K0', 'K1', 'K0', 'K1'],
                              'A': ['A0', 'A1', 'A2', 'A3'],
                              'B': ['B0', 'B1', 'B2', 'B3']})

right = pd.DataFrame({'key1': ['K0', 'K1', 'K1', 'K2'],
                      'key2': ['K0', 'K0', 'K0', 'K0'],
                      'C': ['C0', 'C1', 'C2', 'C3'],
                      'D': ['D0', 'D1', 'D2', 'D3']})
```

```
In [15]: pd.merge(left, right, on=['key1', 'key2'])
```

Out[15]:

	key1	key2	A	B	C	D
0	K0	K0	A0	B0	C0	D0
1	K1	K0	A2	B2	C1	D1
2	K1	K0	A2	B2	C2	D2



# Merging

```
In [16]: pd.merge(left, right, how='outer', on=['key1', 'key2'])
```

Out[16]:

	key1	key2	A	B	C	D
0	K0	K0	A0	B0	C0	D0
1	K0	K1	A1	B1	NaN	NaN
2	K1	K0	A2	B2	C1	D1
3	K1	K0	A2	B2	C2	D2
4	K2	K1	A3	B3	NaN	NaN
5	K2	K0	NaN	NaN	C3	D3

```
In [17]: pd.merge(left, right, how='right', on=['key1', 'key2'])
```

Out[17]:

	key1	key2	A	B	C	D
0	K0	K0	A0	B0	C0	D0
1	K1	K0	A2	B2	C1	D1
2	K1	K0	A2	B2	C2	D2
3	K2	K0	NaN	NaN	C3	D3

```
In [18]: pd.merge(left, right, how='left', on=['key1', 'key2'])
```

Out[18]:

	key1	key2	A	B	C	D
0	K0	K0	A0	B0	C0	D0
1	K0	K1	A1	B1	NaN	NaN
2	K1	K0	A2	B2	C1	D1
3	K1	K0	A2	B2	C2	D2
4	K2	K1	A3	B3	NaN	NaN

# Joining

Joining is a convenient method for combining the columns of two potentially differently-indexed DataFrames into a single result DataFrame.

```
In [19]: left = pd.DataFrame({'A': ['A0', 'A1', 'A2'],  
                             'B': ['B0', 'B1', 'B2']},  
                             index=['K0', 'K1', 'K2'])  
  
right = pd.DataFrame({'C': ['C0', 'C2', 'C3'],  
                     'D': ['D0', 'D2', 'D3']},  
                     index=['K0', 'K2', 'K3'])
```

```
In [20]: left.join(right)
```

Out[20]:

	A	B	C	D
K0	A0	B0	C0	D0
K1	A1	B1	NaN	NaN
K2	A2	B2	C2	D2

```
In [21]: left.join(right, how='outer')
```

Out[21]:

	A	B	C	D
K0	A0	B0	C0	D0
K1	A1	B1	NaN	NaN
K2	A2	B2	C2	D2
K3	NaN	NaN	C3	D3