20. Joining/merging

Pandas có đầy đủ tính năng, hiệu suất cao trong hoạt động in-memory join rất giống với cơ sở dữ liệu quan hệ như SQL. Các phương pháp này thực hiện tốt hơn đáng kể so với các mã nguồn mở khác (như merge.data.frame trong R). Lý do của việc này là thiết kế thuật toán cẩn thận và cách bố trí nội bộ của dữ liệu trong dataframe.

Pandas cung cấp một hàm duy nhất cho tất cả các kiểu joining/merging. Cú pháp như sau:

pd.merge(left, right, how='inner', on=None, left_on=None, right_on=None,

left_index=False, right_index=False, sort=True,

suffixes=('_x', '_y'), copy=True, indicator=False)

left	một đối tượng dataframe
right	đối tượng dataframe khác
on	tên các cột sẽ làm key để join với điều kiện các cột phải nằm trong cả hai DataFrame. Mặc định nó sẽ tự nhận các cột có cùng tên làm keys để jo
left_on	Dùng các cột từ left dataframe để làm key cho việc join. Có thể là tên cột hoặc mảng có chiều dài bằng độ dài của dataframe
right_on	Dùng các cột từ right dataframe để làm key cho việc join. Có thể là tên cột hoặc mảng có chiều dài bằng độ dài của dataframe.
left_index	Nếu là True, sử dụng index (row labels) left từ dataframe như là key để join. Trong trương hợp MultiIndex dataframe, key join xem xét cả level củ
right_index	Nếu là True, sử dụng index (row labels) right từ dataframe như là key để join. Trong trương hợp MultiIndex dataframe, key join xem xét cả level c
how	kiểu join 'left', 'right', 'outer', 'inner'. Mặc định là inner.
sort	dataframe trả về có được sort theo key hay không.
suffixes	Một tuple của các hậu tố của string sử dụng cho các cột trùng lặp. Mặc định là (_x, _y).
copy	Mặc định là True, dữ liệu được xử lý trên đối tượng dataframe mới hay không
indicator	Thêm cột vào dataframe đầu ra được gọi là _merge với thông tin về nguồn của mỗi hàng. _merge là phần phân loại và lấy giá trị left_only cho cá

Giải thích 'how: Nếu bạn đã quen với joining trong SQL thì bảng sau cho ta một so sánh giữa joining trong pandas và SQL. Mọi người có thể tham khảo thêm sự so sánh này tại trang web

"how"	Tương tự trong SQL	Giải Thích
left	LEFT OUTER JOIN	Sử dụng các keys từ bảng bên trái
right	RIGHT OUTER JOIN	Sử dụng các keys từ bảng bên phải
outer	FULL OUTER JOIN	Sử dụng các keys từ cả hai bảng
inner	INNER JOIN	Sử dụng các keys chung giữa hai bảng

Chúng ta cùng đi vào ví dụ cụ thể sẽ dễ hiểu hơn.

Ta có hai bảng dữ liệu bên trái, sau khi merge sẽ cho bảng cuối cùng bên phải.

```
>>> import pandas as pd
>>> import numpy as np
>>> 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']})
>>>
```

Cụ thể hình dạng left, right như sau và mặc định how="inner"

```
>>> left
   A   B key1 key2
```

```
0  A0  B0   K0   K0
1  A1  B1   K0   K1
2  A2  B2   K1   K0
3  A3  B3   K2   K1
>>> right
   C   D key1 key2
0  C0  D0   K0   K0
1  C1  D1   K1   K0
2  C2  D2   K1   K0
3  C3  D3   K2   K0
>>> pd.merge(left, right, on=['key1', 'key2'])
   A   B key1 key2   C   D
0  A0  B0   K0   K0  C0  D0
1  A2  B2   K1   K0  C1  D1
2  A2  B2   K1   K0  C2  D2
>>>
```

Kết quả của phép join với “how” = ‘left’.

```
>>> pd.merge(left, right, how='left', on=['key1', 'key2'])
   A   B key1 key2    C    D
0  A0  B0   K0   K0   C0   D0
1  A1  B1   K0   K1  NaN  NaN
2  A2  B2   K1   K0   C1   D1
3  A2  B2   K1   K0   C2   D2
4  A3  B3   K2   K1  NaN  NaN
>>>
```

Giải thích về suffixes: Ý nghĩa của suffixes được giải thích qua ví dụ sau: Trường hợp hai bảng có tên cột giống nhau khi joining (chú ý tên cột giống nhau, không phải tên key giống nhau). Từ khóa suffixes sẽ giúp phân biệt cột giống nhau đến từ dataframe nào bằng cách cho thêm hậu tố vào tên cột.

```
>>> pd.merge(left, right, on=[left.A,right.D], how='outer',suffixes=('_left','_right'))
   A   B key1_left key2_left   C   D key1_right key2_right
0  A0  B0       K0       K0  C0  D0        K0         K0
1  A1  B1       K0       K1  C1  D1        K1         K0
2  A2  B2       K1       K0  C2  D2        K1         K0
3  A3  B3       K2       K1  C3  D3        K2         K0
>>>
```

Giải thích về indicator: Ý nghĩa của indicator được giải thích qua ví dụ sau. Bạn có thể so sánh với indicator=False và True qua hai đoạn mã bên dưới. Như vậy indicator giúp chỉ rõ hàng đó đến từ dataframe nào.

```
>>> pd.merge(left, right, on=['key1','key2'], how='outer')
    A    B key1 key2    C    D
0   A0   B0   K0   K0   C0   D0
1   A1   B1   K0   K1  NaN  NaN
2   A2   B2   K1   K0   C1   D1
3   A2   B2   K1   K0   C2   D2
4   A3   B3   K2   K1  NaN  NaN
5  NaN  NaN   K2   K0   C3   D3
>>> pd.merge(left, right, on=['key1','key2'], how='outer',indicator =True)
    A    B key1 key2    C    D      _merge
0   A0   B0   K0   K0   C0   D0        both
1   A1   B1   K0   K1  NaN  NaN   left_only
2   A2   B2   K1   K0   C1   D1        both
3   A2   B2   K1   K0   C2   D2        both
4   A3   B3   K2   K1  NaN  NaN   left_only
5  NaN  NaN   K2   K0   C3   D3  right_only
>>>
```

Joining on index

.join() là một phương pháp thuận tiện để kết hợp các cột của hai dataframe được lập chỉ mục khác nhau có khả năng phân loại khác nhau vào một datafram

```
>>> 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'])
>>>
```

Hình dạng của left và right sẽ như sau:

```
>>> left
    A   B
K0  A0  B0
K1  A1  B1
K2  A2  B2
>>> right
    C   D
K0  C0  D0
K2  C2  D2
K3  C3  D3
>>> left.join(right)
    A   B    C    D
K0  A0  B0   C0   D0
K1  A1  B1  NaN  NaN
K2  A2  B2   C2   D2
>>>
```

Với ví dụ phía trên tương đương lệnh sau khi dùng merge:

```
>>> result = pd.merge(left, right, left_index=True, right_index=True, how='left')
```

Mọi người có thể thử với các loại join khác qua từ khóa ‘how’: how = ‘right’ , ‘outer’.

Joining key columns on an index

.join() cũng cung cấp cho ta một đối số tùy chọn là ‘on’ để ta truyền vào. Đối số ‘on’ có thể là tên cột hoặc nhiều tên cột, xác định rằng chỉ mục của right data

Ví dụ minh họa:

```
>>> left = pd.DataFrame({'A': ['A0', 'A1', 'A2', 'A3'],'B': ['B0', 'B1', 'B2', 'B3'],'key': ['K0', 'K1', 'K0', 'K1']})
>>> right = pd.DataFrame({'C': ['C0', 'C1'],'D': ['D0', 'D1']},index=['K0', 'K1'])
>>>
```

Hình dạng của left và right sẽ như sau:

```
>>> left
    A   B key
0  A0  B0  K0
1  A1  B1  K1
2  A2  B2  K0
3  A3  B3  K1
>>> right
    C   D
K0  C0  D0
K1  C1  D1
>>> left.join(right, on='key')
    A   B key   C   D
0  A0  B0  K0  C0  D0
1  A1  B1  K1  C1  D1
2  A2  B2  K0  C0  D0
3  A3  B3  K1  C1  D1
>>>
```

Ví dụ tiếp theo cho trường hợp multikey, được truyền đến dataframe có MultiIndex

```
>>> left = pd.DataFrame({'A': ['A0', 'A1', 'A2', 'A3'],'B': ['B0', 'B1', 'B2', 'B3'],'key1': ['K0', 'K0', 'K1', 'K2'],'key2': ['K0', 'K1', 'K0', 'K1']})
>>> index = pd.MultiIndex.from_tuples([('K0', 'K0'), ('K1', 'K0'),('K2', 'K0'), ('K2', 'K1')])
>>> right = pd.DataFrame({'C': ['C0', 'C1', 'C2', 'C3'],'D': ['D0', 'D1', 'D2', 'D3']},index=index)
>>> result = left.join(right, on=['key1', 'key2'])
>>>
```

Kết quả được trình bầy trong hình sau:

```
>>> left
    A   B key1 key2
0  A0  B0   K0   K0
1  A1  B1   K0   K1
2  A2  B2   K1   K0
3  A3  B3   K2   K1
>>> right
        C   D
K0 K0  C0  D0
K1 K0  C1  D1
K2 K0  C2  D2
   K1  C3  D3
>>> index
MultiIndex(levels=[['K0', 'K1', 'K2'], ['K0', 'K1']],
           labels=[[0, 1, 2, 2], [0, 0, 0, 1]])
>>> left.join(right, on=['key1', 'key2'])
    A   B key1 key2    C    D
0  A0  B0   K0   K0   C0   D0
1  A1  B1   K0   K1  NaN  NaN
2  A2  B2   K1   K0   C1   D1
3  A3  B3   K2   K1   C3   D3
>>>
```

Joining with two multi-indexes

.join() không được implemented với mục đích này, nhưng chúng ta có thể thực hiện với **pd.merge().** Cùng học hỏi qua ví dụ sau:

```
>>> index_right = pd.MultiIndex.from_tuples([('K0', 'Y0'), ('K1', 'Y1'),('K2', 'Y2'), ('K2', 'Y3')],names=['key', 'Y'])
>>> right = pd.DataFrame({'C': ['C0', 'C1', 'C2', 'C3'],'D': ['D0', 'D1', 'D2', 'D3']},index= index_right)
>>> index_left = pd.MultiIndex.from_tuples([('K0', 'X0'), ('K0', 'X1'),('K1', 'X2')],names=['key', 'X'])
>>> left = pd.DataFrame({'A': ['A0', 'A1', 'A2'],'B': ['B0', 'B1', 'B2']},index=index_left)
>>>
```

Kết quả:

```
>>> left
        A   B
key X
K0  X0  A0  B0
    X1  A1  B1
K1  X2  A2  B2
>>> right
        C   D
key Y
K0  Y0  C0  D0
K1  Y1  C1  D1
K2  Y2  C2  D2
    Y3  C3  D3
>>> pd.merge(left.reset_index(), right.reset_index(),on=['key'], how='inner').set_index(['key','X','Y'])
          A   B   C   D
key X  Y
K0  X0 Y0  A0  B0  C0  D0
    X1 Y0  A1  B1  C0  D0
K1  X2 Y1  A2  B2  C1  D1
>>>
```

Giải thích quá trình trên như sau: Thay vì truyền vào left, right ta truyền vào left reset_index() và right.reset_index() tương tứng.

```
>>> right
        C   D
key Y
K0  Y0  C0  D0
K1  Y1  C1  D1
K2  Y2  C2  D2
    Y3  C3  D3
>>> right.reset_index()
  key   Y   C   D
0  K0  Y0  C0  D0
1  K1  Y1  C1  D1
2  K2  Y2  C2  D2
3  K2  Y3  C3  D3
>>> left
        A   B
key X
K0  X0  A0  B0
    X1  A1  B1
K1  X2  A2  B2
>>> left.reset_index()
  key   X   A   B
0  K0  X0  A0  B0
1  K0  X1  A1  B1
2  K1  X2  A2  B2
>>>
```

Ta sẽ truyền về phéo join cơ bản trên key.

```
>>> result = pd.merge(left.reset_index(), right.reset_index(),on=['key'], how='inner')
>>> result
  key   X   A   B   Y   C   D
0  K0  X0  A0  B0  Y0  C0  D0
1  K0  X1  A1  B1  Y0  C0  D0
2  K1  X2  A2  B2  Y1  C1  D1
>>>
```

Chuyển ba cột “key”,’X’,’Y’ làm index qua phương thức đã học set_index().

```
>>> result = pd.merge(left.reset_index(), right.reset_index(),on=['key'], how='inner').set_index(["key",'X','Y'])
>>> result
          A   B   C   D
key X  Y
K0  X0 Y0  A0  B0  C0  D0
    X1 Y0  A1  B1  C0  D0
K1  X2 Y1  A2  B2  C1  D1
>>>
```

Kết Luận

Ở đây, bạn đã học tất cả các cách thức hợp nhất các cấu trúc dữ liệu trong pandas. Bạn đã khám phá ra các kỹ thuật khác nhau để hợp nhất và tìm hiểu về các liên kết như inner, outer, right, left, join, cũng như thời điểm để sử dụng qua hai phương thức chính là .join() và pd.merge(). Và bạn nên thực hành nhiều để có thể hiểu rõ hơn về các tham số như **on, left_on, right_on, how, suffixes**…