Giới thiệu về DataFrame

23-12-2020

1 Giới thiệu

pandas.DataFrame (gọi tắt là DataFrame) là một mảng hai chiều có gắn nhãn.
DataFrame có một số đặc điểm sau :

- DataFrame là một mảng hai chiều.
- DataFrame có thể xem như là nhiều Series có chung label (index) được ghép kế tiếp nhau.
- Dữ liệu trong một cột là đồng nhất.
- Tuy nhiên, hai cột khác nhau có thể có kiểu dữ liệu khác nhau.
- Có thể thay đổi kích thước DataFrame bằng cách thêm bớt dòng cột.

Ví dụ một DataFrame

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

        pd.DataFrame({
            'Product' : ['Apple', 'Banana', 'Cherry'],
            'Quantity' : [12, 34, 56],
            'Price' : [10, 5, 8]
        })
```

```
Out[1]:   Product  Quantity  Price
        0   Apple        12     10
        1  Banana        34      5
        2  Cherry        56      8
```

2 Cách khởi tạo DataFrame

Để khởi tạo một DataFrame, bạn có thể dùng cú pháp sau :

```
pandas.DataFrame(data, index, columns)
```

Trong đó:

- data: dữ liệu truyền vào, có thể bỏ trống.
- index: label của từng dòng, có thể bỏ trống.
- columns: label của từng cột, có thể bỏ trống.

Cùng xem một số ví dụ để hiểu rõ hơn về cách tạo DataFrame. DataFrame
Tạo DataFrame từ một list

```
In [2]: # list
        l = [0, 1, 2, 3, 5]
        # DataFrame từ list
        pd.DataFrame(data = l) # pd.DataFrame(l)

Out[2]:    0
        0  0
        1  1
        2  2
        3  3
        4  5
```

Tạo DataFrame từ list của list

```
In [3]: # list của list
        ll = [
          ['Apple', 100],
          ['Banana', 25],
          ['Cherry', 36]
        ]
        # tạo DataFrame
        pd.DataFrame(ll)

Out[3]:         0    1
        0   Apple  100
        1  Banana   25
        2  Cherry   36

In [4]: # tạo DataFrame, có chỉ ra index và columns
        pd.DataFrame(ll, index = ['a', 'b', 'c'], columns = ['Product', 'Quantity'])

Out[4]:   Product  Quantity
        a   Apple       100
        b  Banana        25
        c  Cherry        36
```

Câu hỏi : Nếu các list con trong ll có độ dài khác nhau thì có thể tạo được DataFrame từ ll hay không?

Tạo DataFrame từ dictionary

```
In [5]: # tạo dictionary
        d1 = {
            'col1' : ['Apple', 'Banana'],
            'col2' : [1, 2],
            'col3' : ['2019-10-02', '2019-11-01']
        }
        # tạo DataFrame
        pd.DataFrame(d1)

Out[5]:      col1  col2        col3
        0   Apple     1  2019-10-02
        1  Banana     2  2019-11-01
```

Câu hỏi: Nếu độ dài các `list` khác nhau thì có thể tạo được `DataFrame` từ `d1` hay không?
Tạo `DataFrame` từ `list` các `dictionary`

```
In [6]: # tạo list mà mỗi phần tử là một dictionary
        ld = [
              {'Date' : '2019-10-01', 'Ticker' : 'AAA', 'Close' : 100},
              {'Date' : '2019-10-01', 'Ticker' : 'BBB', 'Close' : 200}
        ]
        # tạo DataFrame
        pd.DataFrame(ld)
```

```
Out[6]:          Date Ticker  Close
        0  2019-10-01    AAA    100
        1  2019-10-01    BBB    200
```

Câu hỏi : Trong trường hợp `ld` ở trên, nếu `dictionary` thứ 2 là

```
{'Date' : '2019-10-01', 'Ticker' : 'BBB', 'Open': 205, 'Close' : 200}
```

thì có tạo được `DataFrame` từ `d2` hay không?

3 Các thao tác với `DataFrame`

Trong phần này, chúng ta sẽ sử dụng `DataFrame` mẫu sau

```
In [7]: import numpy as np

        d2 = {
            'col1' : pd.date_range(start = '2019-11-20', periods = 10, freq = 'D'),
            'col2' : np.random.choice(['Apple', 'Banana', 'Cherry'], size = 10),
            'col3' : np.random.randint(100, size = 10)
        }
        df = pd.DataFrame(d2, index = list('abcdefghij'))
        df
```

```
Out[7]:         col1    col2  col3
        a 2019-11-20   Apple    46
        b 2019-11-21   Apple    52
        c 2019-11-22  Cherry    92
        d 2019-11-23  Cherry    50
        e 2019-11-24  Cherry    82
        f 2019-11-25   Apple    97
        g 2019-11-26   Apple    57
        h 2019-11-27   Apple    32
        i 2019-11-28  Banana    27
        j 2019-11-29   Apple    22
```

3.1 Xem một số thông tin cơ bản của một `DataFrame` :

Bạn có thể xem một số thông tin cơ bản của `DataFrame` thông qua những thuộc tính sau:

- `.size`: trả về số lượng phần tử của `DataFrame`.
- `.shape`: trả về kích thước của `DataFrame` với định dạng (dòng, cột).
- `.empty`: trả về `True` nếu `DataFrame` là rỗng.
- `.dtypes`: trả về kiểu dữ liệu của từng cột.
- `.columns`: trả về danh sách các cột của `DataFrame`.
- `.index`: trả về label các dòng của `DataFrame`.

Câu hỏi : Giả sử một `DataFrame` được tạo như sau

```
df1 = pd.DataFrame(columns = ['x', 'y', 'z'])
```

Vậy, `df1` có rỗng hay không?

3.2 Đổi tên dòng và cột

Để đổi tên dòng và cột, ta có thể dùng phương thức `.rename()` như sau:

```
<tên_DataFrame>.rename(index = <tên_dòng_mới>, columns = <tên_cột_mới>)
```

Trong đó, `tên_dòng_mới` và `tên_cột_mới` là `dictionary` với cấu trúc

```
{<tên_cũ_1>: <tên_mới_1>, <tên_cũ_2>: <tên_mới_2>, ...}
```

```
In [8]: # đổi tên cột `col1` thành `Date`
        df.rename(columns = {'col1' : 'Date'})

Out[8]:         Date    col2  col3
        a 2019-11-20   Apple    46
        b 2019-11-21   Apple    52
        c 2019-11-22  Cherry    92
        d 2019-11-23  Cherry    50
        e 2019-11-24  Cherry    82
        f 2019-11-25   Apple    97
        g 2019-11-26   Apple    57
        h 2019-11-27   Apple    32
        i 2019-11-28  Banana    27
        j 2019-11-29   Apple    22
```

Lưu ý: có thể bỏ qua `index` nếu chỉ đổi tên cột và bỏ qua `columns` nếu chỉ đổi tên dòng. Ngoài ra, bạn có thể đổi tên *nhanh* toàn bộ cột theo cách sau :

```
<tên_DataFrame>.columns = <danh_sách_tên_cột_mới>
```

Cách này thay đổi trực tiếp tên cột trong `DataFrame`.
Tương tự, bạn có thể đổi tên *nhanh* toàn bộ dòng theo cách sau :

```
<tên_DataFrame>.index = <danh_sách_tên_dòng_mới>
```

```
In [9]: # tạo bản deep copy của df
        df2 = df.copy()
        # đổi tên toàn bộ cột của df2
        df2.columns = ['Date', 'Product', 'Quantity']
        df2

Out[9]:         Date Product  Quantity
        a 2019-11-20   Apple        46
        b 2019-11-21   Apple        52
        c 2019-11-22  Cherry        92
        d 2019-11-23  Cherry        50
        e 2019-11-24  Cherry        82
        f 2019-11-25   Apple        97
        g 2019-11-26   Apple        57
        h 2019-11-27   Apple        32
        i 2019-11-28  Banana        27
        j 2019-11-29   Apple        22
```

3.3 Trích xuất dữ liệu theo cột

Để lấy dữ liệu từ 1 cột, ta có thể thực hiện theo 2 cách:

1. `<tên_DataFrame>.<tên_cột>`.
2. `<tên_DataFrame>[<tên_cột>]`

Lưu ý: chỉ có thể dùng cách 1 khi mà tên cột **không** chứa khoảng trắng.
Ví dụ:

```
In [10]: # lấy dữ liệu từ cột có tên 'col1'
         df['col1'] # df.col1

Out[10]: a   2019-11-20
         b   2019-11-21
         c   2019-11-22
         d   2019-11-23
         e   2019-11-24
         f   2019-11-25
         g   2019-11-26
         h   2019-11-27
         i   2019-11-28
         j   2019-11-29
         Name: col1, dtype: datetime64[ns]
```

Câu hỏi : Kiểu dữ liệu của `df['col1']` trong ví dụ trên là gì?
Để lấy dữ liệu từ nhiều cột, thực hiện như sau:

```
<tên_DataFrame>[<danh_sách_tên_cột>]
```

Ví dụ:

```
In [11]: # lấy dữ liệu từ hai cột 'col1' và 'col2'
         df[['col1', 'col2']]
```

```
Out[11]:         col1    col2
         a 2019-11-20   Apple
         b 2019-11-21   Apple
         c 2019-11-22  Cherry
         d 2019-11-23  Cherry
         e 2019-11-24  Cherry
         f 2019-11-25   Apple
         g 2019-11-26   Apple
         h 2019-11-27   Apple
         i 2019-11-28  Banana
         j 2019-11-29   Apple
```

Câu hỏi: Phân biệt `df['col1']` và `df[['col1']]`.

3.4 Trích xuất dữ liệu theo dòng

Giống như `Series`, bạn có thể trích xuất dữ liệu từ một (hoặc nhiều) dòng thông qua label hoặc số thứ tự của dòng.

Tuy nhiên, bạn lại không thể dùng

```
<tên_DataFrame>[<label_dòng_cần lấy>]
```

vì đây là cách dùng để trích xuất cột.

```
In [ ]: # Thử gọi dòng có label là 'a', sẽ báo lỗi
        df['a']
```

Vậy phải làm sao để có thể lấy dữ liệu theo dòng?

Câu trả lời chính là dùng 2 đối tượng phái sinh từ `DataFrame` gốc ban đầu:

- `.loc` sẽ giúp bạn lấy dòng theo label.
- `.iloc` sẽ giúp bạn lấy dòng theo số thứ tự

```
In [12]: # Gọi dòng có label là 'a' bằng .loc
         df.loc['a']

Out[12]: col1    2019-11-20 00:00:00
         col2                  Apple
         col3                     46
         Name: a, dtype: object

In [13]: # Gọi dòng có số thứ tự 2 bằng .iloc
         df.iloc[2]

Out[13]: col1    2019-11-22 00:00:00
         col2                 Cherry
         col3                     92
         Name: c, dtype: object
```

Bài tập : Lấy hai dòng có label là `'a'` và `'c'` từ `df`.

3.5 Trích xuất theo cả dòng và cột

Ngoài việc giúp bạn có thể lấy dòng theo label, `.loc` còn thể giúp bạn lấy thêm những cột mong muốn (bằng label) theo cú pháp sau

```
<tên_DataFrame>.loc[<dòng_cần_lấy>, <cột_cần_lấy>]
```

Tương tự, `.iloc` sẽ giúp bạn lấy dòng và cột theo số thứ tự

```
In [14]: # Lấy dòng có label 'a' và 'c', cột có label là 'col2' và 'col3'
         df.loc[['a', 'c'], ['col2', 'col3']]

Out[14]:      col2  col3
         a   Apple    46
         c  Cherry    92
```

3.6 Trích xuất theo điều kiện (Lọc)

Ngoài ra, `DataFrame` còn cho phép bạn lấy dữ liệu theo một điều kiện nào đó theo cú pháp

```
<tên_DataFrame>[điều_kiện]
```

`điều_kiện` ở đây có thể là điều kiện đơn hoặc điều kiện ghép (tức là được ghép từ các điều kiện đơn)

Ví dụ lọc theo điều kiện đơn

```
In [15]: # Lọc từ df những dòng có cột 'col2' là 'Apple'
         df[df.col2 == 'Apple']

Out[15]:         col1   col2  col3
         a 2019-11-20  Apple    46
         b 2019-11-21  Apple    52
         f 2019-11-25  Apple    97
         g 2019-11-26  Apple    57
         h 2019-11-27  Apple    32
         j 2019-11-29  Apple    22
```

Để ghép nhiều điều kiện lại với nhau bạn có thể dùng toán tử `&` cho phép toán `and` và `|` cho phép toán hoặc.

Một điểm cần lưu ý khi ghép, các điều kiện đơn phải nằm trong dấu `()`. Ví dụ:

```
(điều_kiện_1) & (điều_kiện_2) | (điều_kiện_3)
```

Bài tập : Lấy những dòng có `'col2'` là `'Banana'` và `'col3'` > 50