6. Làm quen với dataframe qua một số thao tác trên hàng và cột

Để thuận tiện cho việc diễn giải các ví dụ, tôi tạo một dataframe như sau:

```
>>> import pandas as pd
>>> crimes_rates = {"year":[1960,1961,1962,1963,1964],"Population":[179323175,182992000,185771000,188483000,191141000],"Total":[3384200,3488000,3752200,4109500,4564600],"Viol
>>> crimes_dataframe = pd.DataFrame(crimes_rates)
>>> crimes_dataframe
     Population    Total        Violent  year
0   179323175  3384200   288460  1960
1   182992000  3488000   289390  1961
2   185771000  3752200   301510  1962
3   188483000  4109500   316970  1963
4   191141000  4564600   364220  1964
>>>
```

Thay đổi các chỉ mục hàng

Ta có thể thấy các chỉ mục cho hàng (0,1,2…) được gán tự động cho dataframe, và ta có thể thay đổi các chỉ mục đó theo cách riêng. Ví dụ:

```
>>> ordinals = ["first", "second", "third", "fourth","fifth",]
>>> crimes_dataframe = pd.DataFrame(crimes_rates,index=ordinals)
>>> crimes_dataframe
       Population    Total  Violent  year
first    179323175  3384200   288460  1960
second   182992000  3488000   289390  1961
third    185771000  3752200   301510  1962
fourth   188483000  4109500   316970  1963
fifth    191141000  4564600   364220  1964
>>>
```

Có thể sử dụng một cột làm chỉ mục. Ví dụ sau sử dụng cột ‘year’ làm chỉ mục cho crimes_dataframe.

```
>>> crimes_dataframe = pd.DataFrame(crimes_rates,columns=["Violent","Population","Total"],index=crimes_rates["year"])
>>> crimes_dataframe
     Violent  Population    Total
1960   288460   179323175  3384200
1961   289390   182992000  3488000
1962   301510   185771000  3752200
1963   316970   188483000  4109500
1964   364220   191141000  4564600
>>>
```

Ngoài ra, phương thức set_index dùng để biến một cột thành một chỉ mục. "set_index" không hoạt động tại chỗ, nó trả về một khung dữ liệu mới với cột được chọn là chỉ mục, ví dụ:

```
>>> crimes_dataframe = pd.DataFrame(crimes_rates,columns=["year","Violent","Population","Total"])
>>> crimes_dataframe
   year  Violent  Population    Total
0  1960   288460   179323175  3384200
1  1961   289390   182992000  3488000
2  1962   301510   185771000  3752200
3  1963   316970   188483000  4109500
4  1964   364220   191141000  4564600
>>> new_crimes_df = crimes_dataframe.set_index("year")
>>> new_crimes_df
     Violent  Population    Total
year
1960   288460   179323175  3384200
1961   289390   182992000  3488000
1962   301510   185771000  3752200
```

```
1963   316970   188483000  4109500
1964   364220   191141000  4564600
>>>
```

Ví dụ trên cho thấy set_index đã tạo ra một dataframe mới chứ không thay đổi trực tiếp dataframe hiện tại. Kiểm tra lại crimes_dataframe, ta thấy không bị thay đổi.

```
>>> crimes_dataframe
   year  Violent  Population    Total
0  1960   288460   179323175  3384200
1  1961   289390   182992000  3488000
2  1962   301510   185771000  3752200
3  1963   316970   188483000  4109500
4  1964   364220   191141000  4564600
>>>
```

Tuy nhiên set_index cũng cung cấp một tùy chọn cho phép nó thay đổi dữ liệu tại chỗ, không sinh ra thêm bất cứ đối tượng dataframe mới nào.

```
>>> crimes_dataframe.set_index("year",inplace=True)
>>> crimes_dataframe
      Violent  Population    Total
year
1960   288460   179323175  3384200
1961   289390   182992000  3488000
1962   301510   185771000  3752200
1963   316970   188483000  4109500
1964   364220   191141000  4564600
>>>
```

'year' là tên của index, có thể lấy giá trị này bằng cách gọi name_of_df.index.name (crimes_dataframe.index.name)

Chọn (Selecting), Gán (setting), Xóa(deleting)

Chúng ta có thể xử lý một dataframe theo ngữ nghĩa giống như xử lý với một dictionary của các đối tượng Series được lập chỉ mục. Các thao tác selecting, setting, deleting các cột với cú pháp giống như các thao tác tương tự với dictionary. Chúng ta sẽ đi lần lượt các ví dụ sau để thấy được sự tương đồng này.

(1) Chọn

Có hai cách để truy cập cột của dataframe. Kết quả là trong cả hai trường hợp đều là một Series:

Cách 1 df[Tên cột]

```
>>> print crimes_dataframe['Violent']
first     288460
second    289390
third     301510
fourth    316970
fifth     364220
Name: Violent, dtype: int64
>>> print type(crimes_dataframe['Violent'])
<class 'pandas.core.series.Series'>
>>>
```

Cách 2 df.<Tên cột>

```
>>> crimes_dataframe.Population
first     179323175
second    182992000
third     185771000
fourth    188483000
fifth     191141000
Name: Population, dtype: int64
>>> print type(crimes_dataframe.Population)
```

```
<class 'pandas.core.series.Series'>
>>>
```

(2) Gán

Gán một giá trị mới cho cột. Biến gán phải là một pandas Series. Ví dụ sau.

```
>>> vio = pd.Series([12,34,56,23,45],index=crimes_dataframe.index)
>>> vio
first     12
second    34
third     56
fourth    23
fifth     45
dtype: int64
>>> crimes_dataframe["Violent"] = vio
>>> crimes_dataframe
           year  Violent  Population    Total
first      1960     12   179323175  3384200
second  1961     34   182992000  3488000
third     1962     56   185771000  3752200
fourth   1963     23   188483000  4109500
fifth   1964       45   191141000  4564600
>>> import numpy as np
>>> crimes_dataframe['Murder'] = np.nan
>>> crimes_dataframe
           year  Violent  Population    Total        Murder
first      1960       12   179323175  3384200     NaN
second  1961       34   182992000  3488000     NaN
third     1962       56   185771000  3752200     NaN
fourth   1963       23   188483000  4109500     NaN
fifth       1964       45   191141000  4564600     NaN
>>>
```

Nếu ta tạo ra một cột mới mà không gán giá trị nó sẽ nhận giá trị mặc định NaN

```
>>> crimes_dataframe = pd.DataFrame(crimes_rates,columns=["year","Violent","Population","Total","NEW"])
>>> crimes_dataframe
    year  Violent     Population      Total  NEW
0  1960   288460   179323175  3384200  NaN
1  1961   289390   182992000  3488000  NaN
2  1962   301510   185771000  3752200  NaN
3  1963   316970   188483000  4109500  NaN
4  1964   364220   191141000  4564600  NaN
>>>
```

Ngay cả khi cột vẫn chưa được xác định. Chúng ta có thể thiết lập tất cả các phần tử của cột có cùng giá trị.

```
>>> crimes_dataframe['Murder'] = 9110
>>> crimes_dataframe
    year    Violent    Population    Total     NEW  Murder
0  1960   288460   179323175  3384200  NaN    9110
1  1961   289390   182992000  3488000  NaN    9110
2  1962   301510   185771000  3752200  NaN    9110
3  1963   316970   188483000  4109500  NaN    9110
4  1964   364220   191141000  4564600  NaN    9110
>>>
```

Trong trường hợp này, chắc chắn sẽ tốt hơn nếu bạn chỉ định chính xác số liệu ứng với chỉ mục. Danh sách với các giá trị “Murder” cần có chiều dài tương tự như số hàng trong dataframe của chúng tôi.

```
>>> murder = [9110,8740,8530,8640,9360]
>>> crimes_dataframe['Murder']=murder
>>> crimes_dataframe
   year   Violent   Population   Total     NEW  Murder
0  1960   288460   179323175  3384200  NaN    9110
1  1961   289390   182992000  3488000  NaN    8740
2  1962   301510   185771000  3752200  NaN    8530
3  1963   316970   188483000  4109500  NaN    8640
4  1964   364220   191141000  4564600  NaN    9360
>>>
```

(3) Xóa

Xóa cột “NEW” vừa thêm vào

```
>>> del crimes_dataframe['NEW']
>>> crimes_dataframe
   year   Violent   Population   Total     Murder
0  1960   288460   179323175  3384200   9110
1  1961   289390   182992000  3488000   8740
2  1962   301510   185771000  3752200   8530
3  1963   316970   188483000  4109500   8640
4  1964   364220   191141000  4564600   9360
>>>
```

Tính tổng và tổng tích lũy qua sum và cumsum(). Hàm cumsum() trả về kiểu Pandas Series.

```
>>> crimes_dataframe['Violent'].sum()
1560550
>>> crimes_dataframe['Violent'].cumsum()
0    288460
1    577850
2    879360
3   1196330
4   1560550
Name: Violent, dtype: int64
>>>
```

Thậm chí ta còn có thể chuyển vị (transpose) bảng.

```
>>> crimes_dataframe.T
                  0           1           2           3           4
year           1960        1961        1962        1963        1964
Violent      288460      289390      301510      316970      364220
Population  179323175  182992000  185771000  188483000  191141000
Total       3384200     3488000     3752200     4109500     4564600
Murder         9110        8740        8530        8640        9360
>>>
```

Kết Luận

Qua bài này các bạn đã biết thêm rất nhiều thao tác để tùy chỉnh hàng, cột trong Dataframe. Qua đó có thể thấy cấu trúc dữ liệu này linh hoạt như thế nào. Chúng ta có thể tùy chỉnh kiểu dữ liệu của index, sử dụng một cột làm index thông qua khai báo trực tiếp hoặc sử dụng phương thức set_index(). Set_index() có thể tạo ra một đối tượng dataframe mới hoặc thay đổi trên chính dataframe hiện tại dựa vào việc cài đặt tùy chọn inplace là true hoặc false. Ngoài ra các thao tác Chọn, Gán, Xóa các cột trên dataframe được thực hiện với cú pháp giống như các thao tác với dictionary.