Selecting, Filtering and Sorting Data

June 6, 2018

```
In [44]: import numpy as np
         import pandas as pd
         dataFrame = pd.read_csv('weather.csv').head()

In [45]: dataFrame
```

Out[45]:

	MONTH	DAY	TIME	TEMP	PRESSURE
0	1	1	1	6.8	10207
1	1	1	2	5.8	10214
2	1	1	3	5.7	10220
3	1	1	4	6.0	10225
4	1	1	5	4.5	10230

```
In [5]: #seleciona sempre as colunas
        dataFrame['TEMP']
```

```
Out[5]: 0    6.8
        1    5.8
        2    5.7
        3    6.0
        4    4.5
        Name: TEMP, dtype: float64

In [6]: dataFrameTranspose = dataFrame.T

In [7]: dataFrameTranspose
```

Out[7]:

	0	1	2	3	4
MONTH	1.0	1.0	1.0	1.0	1.0
DAY	1.0	1.0	1.0	1.0	1.0
TIME	1.0	2.0	3.0	4.0	5.0
TEMP	6.8	5.8	5.7	6.0	4.5
PRESSURE	10207.0	10214.0	10220.0	10225.0	10230.0

```
In [9]: dataFrameTranspose[2]
```

```
Out[9]: MONTH       1.0
        DAY         1.0
```

```
TIME            3.0
TEMP            5.7
PRESSURE    10220.0
Name: 2, dtype: float64
```

```
In [10]: dataFrameTranspose[2]['TIME']
```

```
Out[10]: 3.0
```

```
In [11]: dtNovo = pd.DataFrame([['Tadeu'],['Gabriel'],['Luis Fernando']], index=[4,3,4])
         dtNovo
```

```
Out[11]:                0
         4          Tadeu
         3        Gabriel
         4  Luis Fernando
```

```
In [12]: dtNovo[0][4]
```

```
Out[12]: 4            Tadeu
         4    Luis Fernando
         Name: 0, dtype: object
```

```
In [14]: dataFrame
```

```
Out[14]:    MONTH  DAY  TIME  TEMP  PRESSURE
         0      1    1     1   6.8     10207
         1      1    1     2   5.8     10214
         2      1    1     3   5.7     10220
         3      1    1     4   6.0     10225
         4      1    1     5   4.5     10230
```

```
In [15]: dataFrame[['TEMP', 'PRESSURE']]
```

```
Out[15]:    TEMP  PRESSURE
         0   6.8     10207
         1   5.8     10214
         2   5.7     10220
         3   6.0     10225
         4   4.5     10230
```

```
In [16]: dataFrame['TEMP'][[2,4]]
```

```
Out[16]: 2    5.7
         4    4.5
         Name: TEMP, dtype: float64
```

```
In [21]: #slicing
         dataFrame[1:4]
```

```
Out[21]:    MONTH  DAY  TIME  TEMP  PRESSURE
          1      1    1     2   5.8     10214
          2      1    1     3   5.7     10220
          3      1    1     4   6.0     10225

In [26]: #slicing
          dataFrame[2:4][['TEMP','PRESSURE']]

Out[26]:    TEMP  PRESSURE
          2   5.7     10220
          3   6.0     10225

In [25]: dataFrameTranspose

Out[25]:                 0        1        2        3        4
          MONTH         1.0      1.0      1.0      1.0      1.0
          DAY           1.0      1.0      1.0      1.0      1.0
          TIME          1.0      2.0      3.0      4.0      5.0
          TEMP          6.8      5.8      5.7      6.0      4.5
          PRESSURE  10207.0  10214.0  10220.0  10225.0  10230.0

In [28]: dataFrameTranspose[3:5]

Out[28]:                 0        1        2        3        4
          TEMP          6.8      5.8      5.7      6.0      4.5
          PRESSURE  10207.0  10214.0  10220.0  10225.0  10230.0

In [31]: #mesmo resultado que Out[26]
          dataFrameTranspose[3:5][[2,3]]

Out[31]:                 2        3
          TEMP          5.7      6.0
          PRESSURE  10220.0  10225.0

In [32]: dataFrame['TEMP']

Out[32]: 0    6.8
          1    5.8
          2    5.7
          3    6.0
          4    4.5
          Name: TEMP, dtype: float64

In [33]: #dos quatro primeiros
          dataFrame['TEMP'][:4]

Out[33]: 0    6.8
          1    5.8
          2    5.7
          3    6.0
          Name: TEMP, dtype: float64
```

```
In [37]: dataFrameTranspose['DAY':'TEMP']
```

Out[37]:

	0	1	2	3	4
DAY	1.0	1.0	1.0	1.0	1.0
TIME	1.0	2.0	3.0	4.0	5.0
TEMP	6.8	5.8	5.7	6.0	4.5

```
In [38]: #using ioc and iloc
```

```
In [52]: capitals = pd.DataFrame(
             [
                 ["Ngerulmud",391,1.87],
                 ["Vatican City",826,100],
                 ["Yaren",1100,10.91],
                 ["Funafuti",4492,45.48],
                 ["City of San Marino",4493]
             ],
             index=["Palau","Vatican City", "Nauru", "Tuvalu", "San Marino"],
             columns=['Capital','Population','Percentage']
         )
```

```
In [54]: capitals
```

Out[54]:

	Capital	Population	Percentage
Palau	Ngerulmud	391	1.87
Vatican City	Vatican City	826	100.00
Nauru	Yaren	1100	10.91
Tuvalu	Funafuti	4492	45.48
San Marino	City of San Marino	4493	NaN

```
In [64]: #uma única operação
         capitals.loc['Nauru','Population']
```

```
Out[64]: 1100
```

```
In [65]: #duas operações
         capitals['Population']['Nauru']
```

```
Out[65]: 1100
```

```
In [67]: #uma unica operação filtrando
         capitals.loc['Palau':'Nauru',['Population','Capital']]
```

Out[67]:

	Population	Capital
Palau	391	Ngerulmud
Vatican City	826	Vatican City
Nauru	1100	Yaren

```
In [71]: #Select the rows of San Marino and Vatican
         #loc funciona apenas como label e não como position (index)
         capitals.loc[['San Marino', 'Vatican City']]
```

Out[71]:

	Capital	Population	Percentage
San Marino	City of San Marino	4493	NaN
Vatican City	Vatican City	826	100.0

In [73]:
```
#seleciona o indice 1 e 4 do dataframe
capitals.iloc[[4,1]]
```

Out[73]:

	Capital	Population	Percentage
San Marino	City of San Marino	4493	NaN
Vatican City	Vatican City	826	100.0

In [74]:
```
#seleciona todas colunas exceto a primeiro de capitals
capitals.iloc[[4,1], 1:]
```

Out[74]:

	Population	Percentage
San Marino	4493	NaN
Vatican City	826	100.0

In [76]:
```
#retorno as duas primeiras linhas
capitals[[True, True, False, False, False]]
```

Out[76]:

	Capital	Population	Percentage
Palau	Ngerulmud	391	1.87
Vatican City	Vatican City	826	100.00

In [77]:
```
#retorna apenas as capitais com porcentagme superior a 25
capitals[capitals['Percentage'] > 25]
```

Out[77]:

	Capital	Population	Percentage
Vatican City	Vatican City	826	100.00
Tuvalu	Funafuti	4492	45.48

In [119]:
```
grades = pd.DataFrame(
    [
        [10,9],
        [7,8],
        [6,7],
        [6,5],
        [5,2]
    ],
    index = ['Tadeu','Jose Carlos','Maria','Amanda','Sertão'],
    columns=['test_1','test_2']
 )
```

In [81]:
```
grades
```

Out[81]:

	test_1	test_2
Tadeu	10	9
Jose Carlos	7	8
Maria	6	7
Amanda	6	5
Sertão	5	2

```
In [83]: #Os estudantes que tiveram uma nota na segunda prova menor ou igual a primeira
         grades[grades['test_2'] <= grades['test_1']]
```

Out[83]:

	test_1	test_2
Tadeu	10	9
Amanda	6	5
Sertão	5	2

```
In [120]: #adicionei 1 as notas da prova 2 do tadeu e da amanda
          grades.loc[['Amanda', 'Tadeu'], 'test_2'] += 1

In [111]: grades
```

Out[111]:

	test_1	test_2
Tadeu	10	9
Jose Carlos	7	8
Maria	6	7
Amanda	6	5
Sertão	5	2

```
In [112]: reprovado = grades < 6
          aprovado = grades >=6

In [117]: grades[reprovado] = "Fail"
          grades[aprovado]= "Passou"
          grades
```

Out[117]:

	test_1	test_2
Tadeu	Passou	Passou
Jose Carlos	Passou	Passou
Maria	Passou	Passou
Amanda	Passou	Fail
Sertão	Fail	Fail

```
In [121]: grades = pd.DataFrame(
              [
                  [10,9],
                  [7,8],
                  [6,7],
                  [6,5],
                  [5,2]
              ],
              index = ['Tadeu','Jose Carlos','Maria','Amanda','Sertão'],
              columns=['test_1','test_2']
          )

In [122]: grades
```

```
Out[122]:              test_1  test_2
          Tadeu            10       9
          Jose Carlos       7       8
          Maria             6       7
          Amanda            6       5
          Sertão            5       2

In [123]: grades.mean(axis=1)

Out[123]: Tadeu          9.5
          Jose Carlos    7.5
          Maria          6.5
          Amanda         5.5
          Sertão         3.5
          dtype: float64

In [124]: grades.mean(axis=1) > 6

Out[124]: Tadeu           True
          Jose Carlos     True
          Maria           True
          Amanda         False
          Sertão         False
          dtype: bool

In [125]: grades['passou']=grades.mean(axis=1) > 6

In [126]: grades

Out[126]:              test_1  test_2  passou
          Tadeu            10       9    True
          Jose Carlos       7       8    True
          Maria             6       7    True
          Amanda            6       5   False
          Sertão            5       2   False
```