Cheat Sheet **Pandas**

Pandas

A biblioteca **Pandas** foi construída sob o NumPy e fornece ferramentas de estruturas de dados e análise de dados fáceis de serem utilizadas com a linguagem de programação **Python**

Importe o Pandas com a seguinte convenção:

```
>>> import pandas as pd
```

Estruturas de Dados Pandas

// Series

Um array rotulado **unidimensional** capaz de guardar qualquer tipo de dado

Index >

a	3
b	-5
c	7
d	4

```
>>> s = pd.Series([3, -5, 7, 4], index=[ 'a', 'b', 'c', 'd'])
```

Uma estrutura de dados **bidimensional** rotulada com colunas de potenciais diferentes tipos

Colunas >

	País	Capital	População
0	Bélgica	Bruxelas	11190846
1	India	Nova Delhi	1303171035
2	Brasil	Brasília	207847528

Index >

```
>>> data = {'País': ['Bélgica', 'Índia', 'Brasil'],
            'Capital': ['Bruxelas', 'Nova Delhi', 'Brasília'],
            'População': [11190846,  1303171035, 207847528]}
>>> df = pd.DataFrame(data,
                      columns=['País', 'Capital', 'População'])
```

Criando uma coluna nova

```
>>> df['Continente'] = ['Europa','Ásia',None] #Atribuindo valores através de uma lista
>>> df['Maisde100M'] = df['População'] > 100000000 #Atribuindo valores através do cálculo de outra coluna
```

Deletando

```
>>> s.drop(['a', 'c']) #Para deletar valores das linhas (axis=0)
>>> df.drop('País', axis=1) #Para deletar os valores de colunas(axis=1)
```

Ordenando

```
>>> df.sort_index() #Para definir por rotulos junto do Axis
>>> df.sort_values(by='País') #Para definir pelos valores juntamente de um Axis
>>> df.rank() #Para atribuir ranks as entradas
```

I/O

// Ler e gravar em CSV

```
>>> pd.read_csv('file.csv', header=None, nrows=5)
>>> df.to_csv('myDataFrame.csv')
```

// Ler e gravar em Excel

```
>>> pd.read_excel('file.xlsx')
>>> df.to_excel('dir/myDataFrame.xlsx', sheet_name='Sheet1')
```

Para ler múltiplas planilhas do mesmo arquivo:

```
>>> xlsx = pd.ExcelFile('file.xlsx')
>>> df = pd.read_excel(xlsx, 'Sheet1')
```

// Ler e gravar em SQL ou tabelas de banco de dados

```
>>> from sqlalchemy import create_engine
>>> engine = create_engine('sqlite://:memory:')
>>> pd.read_sql("SELECT * FROM minha_tabela;", engine)
>>> pd.read_sql_table('minha_tabela', engine)
>>> pd.read_sql_query("SELECT * FROM minha_tabela;", engine)
```

read_sql() **é uma conveniência wrapper around** read_sql_table() **e** read_sql_query

```
>>> df.to_sql('myDf', engine)
```

Filtros

// Selecionando um valor ou subconjunto

```
>>> df['País'][0] #pegando um valor
>>> df[1:] #Pegando um subconjunto do DatraFrame através de índice
>>> df['Capital'][1:] #Pegando um subconjunto de determinada coluna
>>> df[['País','Capital']] #Selecionando apenas determinadas colunas do DataFrame
>>> df.head(2) #Primeiros 2 elementos
>>> df.tail(2) #Últimos 2 elementos
```

// Selecionando com operadores lógicos: loc e iloc

```
>>> df.iloc[0,0] #Usando índices
>>> df.iloc[0]['País'] #Usando índice e coluna
>>> df.loc[df['País'] == 'Brasil'] #Usando operadores lógicos com loc
>>> df.loc[df['População'] > 200000000]
>>> df.loc[~(df['População'] > 200000000)] #Negando um operador lógico (resultado inverso)
>>> df.loc[(df['País'] == 'Brasil') | (df['Capital'] == 'Nova Delhi')]
#Usando vários operadores com a cláusula OR
```

// Selecionando com operadores lógicos: loc e iloc

```
>>> df.loc[(df['País'] == 'Brasil') & (df['População'] > 1000000)] #Usando vários operadores com a cláusula AND
>>> df.loc[df['Continente'].isnull()] #Selecionando valores nulos
```

Consultando o Dataframe

```
df.shape #Retorna o número de colunas e linhas
df.info() #Retorna a estrutura do DataFrame
df.index #Retorna os valores do índice
df.columns #Retorna as colunas
df.describe() #Retorna dados estatísticos
df.count() #Conta o número de itens excluindo None ou Na
```

Transformando tipo de dado da coluna

```
>>> df['População'] = df['População'].astype(str) #transformando para str
>>> df['População'] = df['População'].astype(int) #transformando para inteiro
```

Funções de agregação

```
>>> df['População'].sum() #Somando valores da coluna
>>> df['População'].mean() #Média de valores da coluna
>>> df['População'].min() #Valores mínimos
>>> df['População'].max() #Valores máximos
```

Percorrendo um dataframe

```
#Usando for
for key,value in df.iterrows():
  if value['País'] == 'Brasil':
    print('Achamos o Brasil no índice '+str(key))

#Usando apply
def isBrasil(value):
  if value == 'Brasil':
    print('Achamos o Brasil')

df['País'].apply(lambda x: isBrasil(x))
```

Unindo dataframes

```
>>> pd.concat([df,df2]) #Unindo linhas (union do sql)
>>> pd.merge(df2,df3,how="inner",left_on='Capital',right_on='Cidade')
```

pandas

harve