

# Pandas: Indexação, Seleção e atribuição

Professor: Me. **Nator Junior Carvalho da Costa**

October 24, 2025

Abra o Google Colab

# Sumário

Pandas: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

- 1 Pandas;
- 2 Series;
- 3 DataFrame;
- 4 LOC, ILOC;



- Pandas é uma biblioteca Python para análise de dados.

Instalação da biblioteca via PyPI:

```
1 pip install pandas
```

# Iniciando com Pandas

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)



- Para utilizar a biblioteca basta realizar a importação

Instalação da biblioteca via PyPI:

```
1 # Forma tradicional de importar a biblioteca pandas
2 import pandas as pd
```

# Pandas Series

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

## Series

The diagram shows two Pandas Series side-by-side. The first series, titled 'NOME', has a red header and red data cells. The second series, titled 'IDADE', has a green header and green data cells. Both series share a common set of indices (0 to 4) shown in a grey column to the left. Brackets on the left group the indices under the label 'Índices', and brackets on the right group the values under the label 'Valores'.

	NOME	IDADE
0	John	22
1	Maria	27
2	Joao	25
3	Paulo	35
4	Ana	40

Figura 1: Classe Series do Pandas.

# Series

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

Construindo séries a partir de um dicionário

```
1 >>> d = {'a': 1, 'b': 2, 'c': 3}
2 >>> ser = pd.Series(data=d)
3 >>> ser
4 a    1
5 b    2
6 c    3
7 dtype: int64
```

# Operações com Series

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

Construindo séries a partir de uma lista.

```
1 >>> d = [4,6,3,7]
2 >>> ser = pd.Series(data=d)
```

Operações:

- Para calcular a média, **ser.mean()**;
- Para calcular a mediana, **ser.median()**;
- Para calcular a variância, **ser.var()**;
- Para calcular a desvio padrão, **ser.std()**.



# Atividade 01

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

Determine a média e variância das notas finais de 12 alunos de uma aula de Química. As notas são apresentadas na tabela abaixo e faça um programa que imprima o resultado das operações.

$$\begin{bmatrix} 8.0 & 9.1 & 7.5 & 7.4 & 7.6 & 7.8 & 8.0 & 9.3 & 7.0 & 7.0 \\ 7.2 & 7.5 & & & & & & & & \end{bmatrix} \quad (1)$$

## Construindo séries a partir de um dicionário

```
1 >>> d = [8.0, 9.1, 7.5, 7.4, 7.6, 7.8, 8.0, 9.3, 7.0, 7.0, 7.2, 7.5]
2 >>> ser = pd.Series(data=d)
3 >>> print("A média é:", ser.mean())
4 >>> print("A mediana é:", ser.median())
5 >>> print("A variância é:", ser.var())
6 >>> print("A desvio padrão é:", ser.std())
```

# Series: Mesmo tipo de dados

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

```
1 >>> import pandas as pd
2 >>> serie = pd.Series([42, 99, -1])
3 >>> serie
4 0      42
5 1      99
6 2      -1
7 dtype: int64
```

# Series: Tipos diferentes de dados

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

```
1 >>> serie2 = pd.Series(['hello', 2.3, True])
2 >>> serie2
3 0      hello
4 1      2.3
5 2      True
6 dtype: object
```

# Acessando elementos

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

```
1 >>> serie[0]
2 42
3 >>> serie[2]
4 -1
```

# Criando índices personalizados

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

```
1 >>> import pandas as pd
2 >>> serie = pd.Series(
3     [200, 350, 550],
4     index=['banana', 'prato feito', 'big mac']
5 )
6 >>> serie
7 banana                200
8 prato feito           350
9 big mac               550
10 dtype: int64
```

# Acessando elementos

Series: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

```
1 >>> serie['big mac']  
2 550
```

```
1 >>> import pandas as pd
2 >>> serie = pd.Series(
3     [200, 350, 550],
4     index=[(0, 0), (0, 1), (0, 2)]
5 )
6 >>> serie
7 (0, 0)    200
8 (0, 1)    350
9 (0, 2)    550
10 dtype: int64
11 >>> serie[(0,1)]
12 350
```



# DataFrame

DataFrame: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

## DataFrame

		Colunas	
		NOME	IDADE
Índices	0	John	22
	1	Maria	27
	2	Joao	25
	3	Paulo	35
	4	Ana	40
		Valores	

Figura 2: DataFrame do Pandas.

# DataFrame

DataFrame: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

## Criando um DataFrame

```
1 >>> import pandas as pd
2 >>> df = pd.DataFrame({
3     'calorias':[200, 350, 550],
4     'gordura (%)':[0, 6, 15]
5 })
6 >>> df
7      calorias  gordura (%)
8 0         200           0
9 1         350           6
10 2         550          15
```

# DataFrame

DataFrame: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

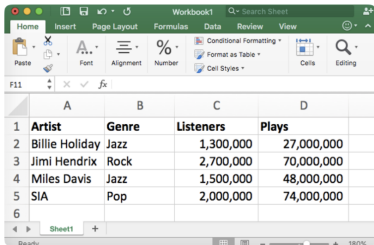
Criando um DataFrame e definindo um índice.

```
1 >>> import pandas as pd
2 >>> df = pd.DataFrame({
3     'calorias':[200, 350, 550],
4     'gordura (%)':[0, 6, 15]
5 },
6     index=['banana', 'prato feito', 'big mac']
7 )
8 >>> df
9
10      calorias  gordura (%)
11 banana      200          0
12 prato feito  350          6
13 big mac     550         15
```

# Lendo arquivos com pandas

DataFrame: [https://notepad.acilab.com.br/workshop\\_senac](https://notepad.acilab.com.br/workshop_senac)

music.csv



	Artist	Genre	Listeners	Plays
1	Billie Holiday	Jazz	1,300,000	27,000,000
2	Jimi Hendrix	Rock	2,700,000	70,000,000
3	Miles Davis	Jazz	1,500,000	48,000,000
4	SIA	Pop	2,000,000	74,000,000



```
pandas.read_csv('music.csv')
```

	Artist	Genre	Listeners	Plays
0	Billie Holiday	Jazz	1,300,000	27,000,000
1	Jimi Hendrix	Rock	2,700,000	70,000,000
2	Miles Davis	Jazz	1,500,000	48,000,000
3	SIA	Pop	2,000,000	74,000,000

(a) matriz

# Lendo arquivos com pandas

## Lendo arquivos com pandas

### CSV

```
1 import pandas as pd
2 data = pd.read_csv("arquivo.csv")
```

*Código 1: Lendo arquivos CSV*

# Lendo arquivos com pandas

## Lendo arquivos com pandas

### Excel

```
1 import pandas as pd
2 data = pd.read_excel("arquivo.xlsx")
```

*Código 2: Lendo arquivos Excel*

# Lendo arquivos com pandas

## Lendo arquivos com pandas

### JSON

```
1 import pandas as pd
2 data = pd.read_json("arquivo.json")
```

*Código 3: Lendo arquivos JSON*

# Filtrando um valor com pandas

## Filtrando valores com pandas

**Função** query no pandas.

```
1
2 # Usando a função query para selecionar linhas com idade >
3 resultado = df.query('Idade > 25')
4 print(resultado)
```

*Código 4: Função query*



# Filtrando um valor com pandas

## Filtrando valores com pandas

**Função** query no pandas.

```
1 resultado = df.query('Idade > 25 & Salário < 6000')
```

*Código 5: Filtro composto (> & <)*

```
1 resultado = df.query('Idade == 25')
```

*Código 6: Filtro por igualdade*

# Filtrando um valor com pandas

## Filtrando valores com pandas

Para esta atividade, utilize a base de dados:

[https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/ibge\\_populacao.csv](https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/ibge_populacao.csv)

- 1 Descobrir as 10 maiores cidades do Ceará no ano de 2019;
- 2 Descobrir as 10 maiores cidades do Piauí;

Para gerar as curvas de crescimento populacional, utilize a biblioteca **Plotly**.

# DataFrame

## Filtrando valores com pandas

Acessando os valores de uma coluna do DataFrame.

```
1 >>> df['gordura (%)']
2 banana          0
3 prato feito     6
4 big mac         15
5 Name: gordura (%), dtype: int64
6
```

# DataFrame

## Filtrando valores com pandas

A forma de acessar um dado de um DataFrame por meio de índices é a seguinte:

```
1 dataframe[<coluna>][<linha>]
```

Exemplo:

```
1 dados = [('Ana', 21), ('Bruno', 20), ('Carla', 22)]
2 df = pd.DataFrame(data = dados)
3 print(df)
4 #      0  1
5 #0   Ana  21
6 #1 Bruno  20
7 #2 Carla  22
8 print(df[0][0], df[0][1], df[0][2]) # Ana Bruno Carla
```

# DataFrame

## Filtrando valores com pandas

Acessando os valores de uma linha do DataFrame.

```
1 >>> df = pd.DataFrame({
2     'calorias':[200, 350, 550],
3     'gordura (%)':[0, 6, 15]
4 })
5 >>> df
6 banana          0
7 prato feito     6
8 big mac         15
9 Name: gordura (%), dtype: int64
10 >>> df[1:2]
11      calorias  gordura (%)
12 1         350          6
```

# LOC

## Filtrando valores com pandas

```
1 #podemos chamar uma linha pelo seu índice
2 df.loc[5]
3 #ou com um array de índices
4 df.loc[[0,1,2]]
5
6 # uma fatia, do quarto ao sétimo elemento
7 #(note que diferente do python puro, neste método
8 #a chave inicial e final estarão presente no resultado)
9 df.loc[4:8]
```

# LOC

## Filtrando valores com pandas

```
1 #busca simples
2 df.loc[df['calorias'] >= 200]
3 #busca composta
4 df.loc[(df['calorias'] >= 10)&(df['calorias'] <= 350)]
5
```

# LOC

## Filtrando valores com pandas

```
1 #Redefinindo valores.  
2 df.loc[(df['calorias']) >= 200, 'calorias'] = 210
```



```
1 # Linhas:  
2 df.iloc[0] # Selecionado a primeira linha do dataset  
3 df.iloc[-1] # Selecionando a última linha  
4
```

```
1 # Colunas:  
2 # Todos os dados da primeira coluna do dataset  
3 df.iloc[:,0]  
4 # Do primeiro ao quinto dado da última coluna  
5 df.iloc[0:5,-1]  
6
```

```
1 # Seleção de múltiplas linhas e colunas:  
2 # resgatando as primeiras três linhas do dataset  
3 df.iloc[0:3]  
4 # todos os dados da segunda e terceira coluna  
5 df.iloc[:, 1:3]  
6 # 1º, 3º e 5º elementos e 6ª a 8ª colunas  
7 df.iloc[[0,2,4], 5:8]
```

# Atividade LOC e ILOC

## Filtrando valores com pandas

- 1 Ler o **csv** disponibilizado no link abaixo e somar os elementos da terceira linha;

```
1 df = pd.read_csv(url, sep=',', index_col=0)
```

link: [https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/dados\\_atividade\\_01.csv](https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/dados_atividade_01.csv)

	0	1	2	3	4	5	6	7	8	9	10	11
0												
1												
2												
3												
4												
5												
6												
7												
8												
9												
10												
11												

(a) Linha da matriz

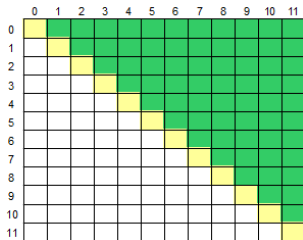
# Atividade LOC e ILOC

## Filtrando valores com pandas

- 1 Ler o **csv** disponibilizado no link abaixo e somar os elementos acima da diagonal principal.

```
1 df = pd.read_csv(url,sep=',',index_col=0)
```

link: [https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/dados\\_atividade\\_01.csv](https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/dados_atividade_01.csv)



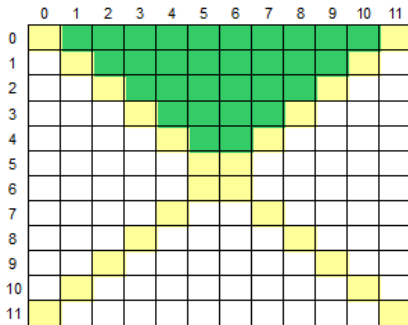
(a) Acima diagonal Principal

# Atividade LOC e ILOC

## Filtrando valores com pandas

- 1 Ler o **csv** disponibilizado no link abaixo e somar os elementos que estão em verde .

link: [https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/dados\\_atividade\\_01.csv](https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/dados_atividade_01.csv)



(a) matriz

# Atividade LOC e ILOC

## Filtrando valores com pandas

- 1 Crie uma Series com os nomes de países e suas respectivas populações. Em seguida, filtre a Series para exibir apenas os países com população superior a 100;
- 2 Crie uma Series com os nomes de diferentes cidades. Em seguida, ordene a Series em ordem alfabética;
- 3 Crie uma Series com as alturas de diferentes estudantes da sala. Em seguida, calcule a média de altura.

# Filtrando um valor com pandas

## Filtrando valores com pandas

Para esta atividade, utilize a base de dados:

[https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/ibge\\_populacao.csv](https://raw.githubusercontent.com/natorjunior/pandas/main/Aula-02/ibge_populacao.csv)

- 1 Descobrir as 10 maiores cidades do Ceará no ano de 2019;
- 2 Descobrir as 10 maiores cidades do Piauí;

Para gerar as curvas de crescimento populacional, utilize a biblioteca **Plotly**.