Trabalhando com Pandas

O que é Pandas?

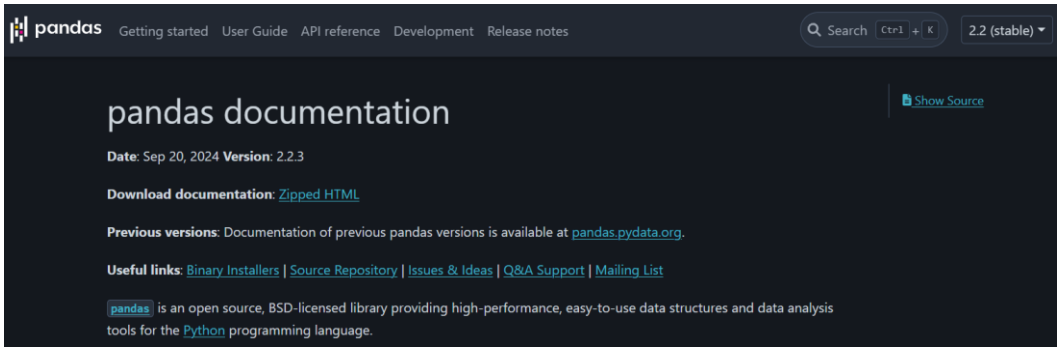
Pandas é uma biblioteca do **Python** usada para **importar**, **manipular** e **analisar dados** de forma simples e eficiente.

Ela permite carregar dados de diversas fontes — como arquivos **Excel**, **JSON**, **CSV**, **bancos de dados e APIs** — e oferece ferramentas poderosas para filtrar, transformar, limpar e organizar esses dados.

Além disso, o **Pandas** é muito útil na preparação de dados para visualizações **gráficas** ou como entrada para modelos de **Machine Learning (ML)**.

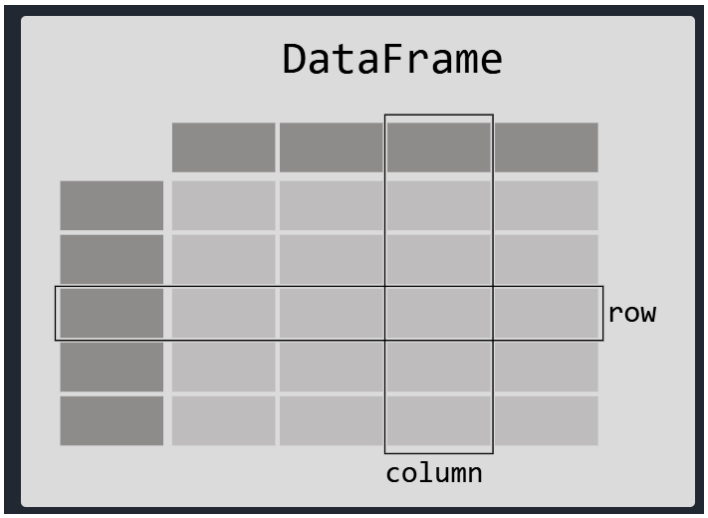
Documentação Pandas

E antes de iniciar com o Pandas na prática, segue a documentação dele, onde será muito importante termos como referência, e consultar sempre que necessário no nosso dia a dia.

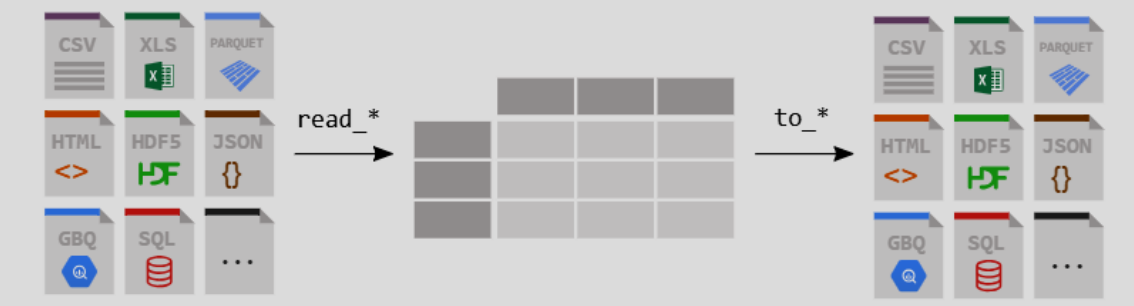


https://pandas.pydata.org/docs/

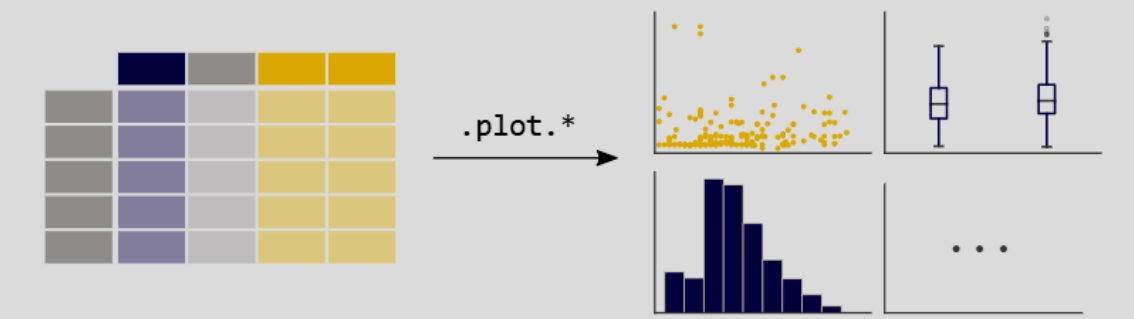
E veja que na documentação é mostrado todo o detalhe que temos que saber sobre o **Pandas**, inclusive que ele trabalha no formato de **DataFrame**, que basicamente é uma estrutura de tabelas, usando linhas e colunas.



E ainda na documentação veremos também o que já foi citado anteriormente, que o **Pandas** pode ler vários tipos de entrada de dados diferentes **(csv, json, etc)**, e que podemos gerar outros relatórios com esses mesmos vários tipos de dados diferentes também!

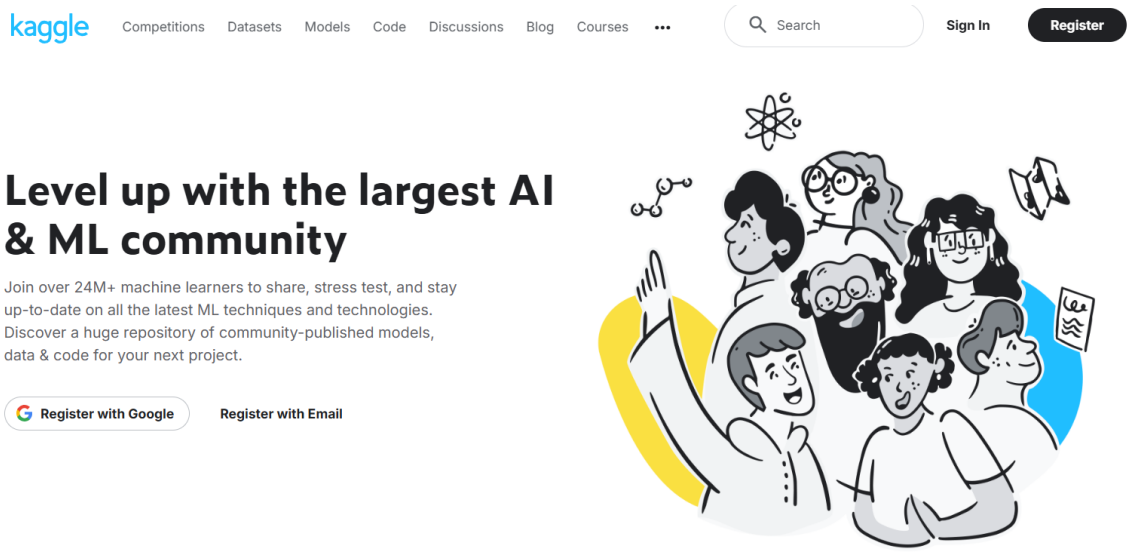


E claro, podemos também com a ajuda de uma outra biblioteca do **python**, o **matplotlib**, podemos gerar **gráficos** nessas saídas ao invés de apenas relatórios!



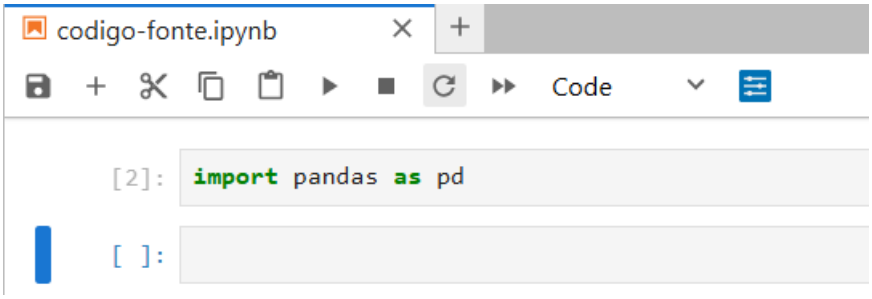
Importando arquivos no Pandas

Primeiramente precisamos de um **arquivo de dados** para que possamos importar esse arquivo utilizando o **Pandas**. Sabendo disso vamos utilizar o **kaggle**, ferramenta gratuita onde conseguimos baixar arquivos com dados de **n extensões** diferentes!!



https://www.kaggle.com/

Baixando um arquivo de dados, vamos **importar** enfim a **library** do **pandas** diretamente no nosso arquivo do **notebook** do **Python**!



Após importar a biblioteca **pandas**, chamaremos a função **read_csv(),** para podermos carregar os dados do nosso arquivo **csv**.

```
[4]: dados = pd.read_csv('fifa.csv')
```

Obs: note que como o nosso arquivo se trata de um csv usamos a função read_csv(), porém existe uma função para cada tipo de extensão diferente, como por exemplo a função read_json() para arquivos json.

Com os dados obtidos pela função **read_csv()**, vamos ilustrar o resultado através pelo **print**!!

```
[5]: print(dados)
```

```
       Unnamed: 0      ID                Name  Age  \
0               0  158023            L. Messi   31
1               1   20801   Cristiano Ronaldo   33
2               2  190871          Neymar Jr   26
3               3  193080             De Gea   27
4               4  192985       K. De Bruyne   27
...           ...     ...                ...  ...
18202       18202  238813        J. Lundstram   19
18203       18203  243165  N. Christoffersson   19
18204       18204  241638           B. Worman   16
18205       18205  246268      D. Walker-Rice   17
18206       18206  246269           G. Nugent   16

                                                Photo Nationality  \
0      https://cdn.sofifa.org/players/4/19/158023.png   Argentina
1       https://cdn.sofifa.org/players/4/19/20801.png    Portugal
2      https://cdn.sofifa.org/players/4/19/190871.png      Brazil
3      https://cdn.sofifa.org/players/4/19/193080.png       Spain
4      https://cdn.sofifa.org/players/4/19/192985.png     Belgium
...                                                ...         ...
18202  https://cdn.sofifa.org/players/4/19/238813.png     England
18203  https://cdn.sofifa.org/players/4/19/243165.png      Sweden
18204  https://cdn.sofifa.org/players/4/19/241638.png     England
```

Porém queremos mostrar os resultados de uma forma mais amigável, logo iremos listar o resultado da **tabela formatada** diretamente pelo **pandas**, sem a utilização do **print**!!

```
[6]: dados
```

[6]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	94	
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	94	
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	93	P
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	93	M
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	92	M
...	...	...	...	...	...	...	...	...	...	...

Mas por que podemos visualizar dessa forma e ainda assim sem utilizar o **print()**?

Isso acontece porque, ao trabalharmos com **Python** em arquivos do tipo **notebook**, a execução de blocos de código **(células)** utiliza um mecanismo chamado **rich display**. Esse recurso **formata** automaticamente a saída de certos objetos, e que nesse caso com o **DataFrames** do **pandas**, torna uma tabela mais bonita e interativa, através de uma funcionalidade do próprio **pandas**, que chama uma função por "debaixo dos panos" que lista os dados através de uma tabela em **html**, facilitando a análise dos dados.

Vale destacar também, que ao listar os dados do nosso arquivo pelo **pandas**, será listado de forma resumida, tanto em linhas e colunas, para que seja possível mostrar o máximo de dados em nossa tela, tanto que será listado de forma padrão as **5 primeiras linhas** e as **últimas 5 linhas**!

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	Clu
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	94	F Barce or
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	94	Juventu
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	93	Paris Sain Germai
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	93	Manchest Unite
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	92	Manchest Ci
...	...	...	...	...	...	...	...	...	...	
18202	18202	238813	J. Lundstram	19	https://cdn.sofifa.org/players/4/19/238813.png	England	https://cdn.sofifa.org/flags/14.png	47	65	Crew Alexand
18203	18203	243165	N. Christoffersson	19	https://cdn.sofifa.org/players/4/19/243165.png	Sweden	https://cdn.sofifa.org/flags/46.png	47	63	Trelleborg
18204	18204	241638	B. Worman	16	https://cdn.sofifa.org/players/4/19/241638.png	England	https://cdn.sofifa.org/flags/14.png	47	67	Cambridg Unite
18205	18205	246268	D. Walker-Rice	17	https://cdn.sofifa.org/players/4/19/246268.png	England	https://cdn.sofifa.org/flags/14.png	47	66	Tranmer Rove

Lendo alguns dados com Pandas

Já vimos anteriormente como listamos dados com o **Pandas**, e que essa listagem é feita de forma resumida com as 5 primeiras linhas e as 5 últimas linhas, porém podemos listar esses dados de forma separada!

Dito isso, veremos algumas **funções** e **funcionalidades** que podemos utilizar no nosso **DataFrame** obtido pelo **Pandas**!

➥ Listar os **primeiros registros** do nosso **DataFrame**, chamamos a função **head()**.

```
[10]: dados_head = dados.head()
      dados_head
```

[10]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	Club	...	Com
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	94	FC Barcelona	...	
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	94	Juventus	...	
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	93	Paris Saint-Germain	...	
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	93	Manchester United	...	
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	92	Manchester City	...	

5 rows × 89 columns

Podemos também definir a quantidade dos primeiros registros que queremos listar, e não só os 5 registros acima por se tratar do padrão! Para isso basta definir a quantidade no parâmetro da função **head([N])**.

```
[16]: dados_head = dados.head(2)
      dados_head
```

[16]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/

2 rows × 89 columns

➥ Listar os **últimos registros** do nosso **DataFrame,** chamamos a função **tail()**.

```
[11]: dados_head = dados.tail()
      dados_head
```

[11]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Potential	Club
18202	18202	238813	J. Lundstram	19	https://cdn.sofifa.org/players/4/19/238813.png	England	https://cdn.sofifa.org/flags/14.png	47	65	Crewe Alexandra
18203	18203	243165	N. Christoffersson	19	https://cdn.sofifa.org/players/4/19/243165.png	Sweden	https://cdn.sofifa.org/flags/46.png	47	63	Trelleborgs FF
18204	18204	241638	B. Worman	16	https://cdn.sofifa.org/players/4/19/241638.png	England	https://cdn.sofifa.org/flags/14.png	47	67	Cambridge United
18205	18205	246268	D. Walker-Rice	17	https://cdn.sofifa.org/players/4/19/246268.png	England	https://cdn.sofifa.org/flags/14.png	47	66	Tranmere Rovers
18206	18206	246269	G. Nugent	16	https://cdn.sofifa.org/players/4/19/246269.png	England	https://cdn.sofifa.org/flags/14.png	46	66	Tranmere Rovers

5 rows × 89 columns

Podemos também definir a quantidade dos primeiros registros que queremos listar, e não só os 5 registros acima por se tratar do padrão! Para isso basta definir a quantidade no parâmetro da função **head([N])**.

```
[15]: dados_head = dados.tail(2)
      dados_head
```

[15]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	
18205	18205	246268	D. Walker-Rice	17	https://cdn.sofifa.org/players/4/19/246268.png	England	https://cdn.sofifa.org/flags/
18206	18206	246269	G. Nugent	16	https://cdn.sofifa.org/players/4/19/246269.png	England	https://cdn.sofifa.org/flags/

2 rows × 89 columns

➥ Listar todas as **colunas** do nosso **DataFrame**

```
[19]: dados.columns
```

```
[19]: Index(['Unnamed: 0', 'ID', 'Name', 'Age', 'Photo', 'Nationality', 'Flag',
       'Overall', 'Potential', 'Club', 'Club Logo', 'Value', 'Wage', 'Special',
       'Preferred Foot', 'International Reputation', 'Weak Foot',
       'Skill Moves', 'Work Rate', 'Body Type', 'Real Face', 'Position',
       'Jersey Number', 'Joined', 'Loaned From', 'Contract Valid Until',
       'Height', 'Weight', 'LS', 'ST', 'RS', 'LW', 'LF', 'CF', 'RF', 'RW',
       'LAM', 'CAM', 'RAM', 'LM', 'LCM', 'CM', 'RCM', 'RM', 'LWB', 'LDM',
       'CDM', 'RDM', 'RWB', 'LB', 'LCB', 'CB', 'RCB', 'RB', 'Crossing',
       'Finishing', 'HeadingAccuracy', 'ShortPassing', 'Volleys', 'Dribbling',
       'Curve', 'FKAccuracy', 'LongPassing', 'BallControl', 'Acceleration',
       'SprintSpeed', 'Agility', 'Reactions', 'Balance', 'ShotPower',
       'Jumping', 'Stamina', 'Strength', 'LongShots', 'Aggression',
       'Interceptions', 'Positioning', 'Vision', 'Penalties', 'Composure',
       'Marking', 'StandingTackle', 'SlidingTackle', 'GKDiving', 'GKHandling',
       'GKKicking', 'GKPositioning', 'GKReflexes', 'Release Clause'],
      dtype='object')
```

➥ Obter informações referente aos **indexes** do nosso **DataFrame**, como por exemplo qual o valor do index inicial e qual valor último index, e qual o valor incrementado de cada index.

```
[21]: dados.index
```

```
[21]: RangeIndex(start=0, stop=18207, step=1)
```

➥ Listar todos os dados específico de uma coluna, **dados[nome_coluna]**

```
[22]: dados['Name']
```

```
[22]: 0                 L. Messi
      1         Cristiano Ronaldo
      2                Neymar Jr
      3                   De Gea
      4             K. De Bruyne
                       ...
      18202          J. Lundstram
      18203    N. Christoffersson
      18204             B. Worman
      18205        D. Walker-Rice
      18206             G. Nugent
      Name: Name, Length: 18207, dtype: object
```

Vale destacar que ao utilizar **dados['Name']**, aonde retorna dados de apenas uma coluna, deixamos de trabalhar com **DataFrame**, e sim com objeto do tipo **Series**!! Que basicamente é como fosse um objeto **List** do **Pandas**!!

➥ Listar todos os dados específico de duas colunas ou mais: **dados[[nome_coluna1, nome_coluna2]]**

```
[23]: dados[['Name', 'Nationality']]
```

[23]:

	Name	Nationality
0	L. Messi	Argentina
1	Cristiano Ronaldo	Portugal
2	Neymar Jr	Brazil
3	De Gea	Spain
4	K. De Bruyne	Belgium
...	...	...
18202	J. Lundstram	England
18203	N. Christoffersson	Sweden
18204	B. Worman	England
18205	D. Walker-Rice	England
18206	G. Nugent	England

18207 rows × 2 columns

Obs: Vale destacar que como estamos trabalhando com mais de 1 coluna, dados[['Name', 'Nationality']], continuamos trabalhando com DataFrame!

Utilizando a propriedade de indexação ILOC

Agora vamos utilizar a propriedade **iloc[]**, para que possamos obter os dados específicos do nosso **DataFrame** através de **indexes**!

E para obter os dados de um registro(linha) específico, usamos a propriedade **iloc[index]**.

```
[25]: dados.iloc[0]
```

```
[25]: Unnamed: 0                                                      0
      ID                                                         158023
      Name                                                     L. Messi
      Age                                                            31
      Photo            https://cdn.sofifa.org/players/4/19/158023.png
                                                                   ...
      GKHandling                                                  11.0
      GKKicking                                                   15.0
      GKPositioning                                               14.0
      GKReflexes                                                   8.0
      Release Clause                                           €226.5M
      Name: 0, Length: 89, dtype: object
```

Podemos também definir um **range (intervalo)** de registros que queremos listar utilizando o recurso de **array slice,** fazendo: **dados.iloc[index_inicial : index_final]**

```
[29]: dados.iloc[0:3]
```

[29]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flag
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flag
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flag

Além disso podemos obter o dado **de uma linha** e **de uma coluna** específico! E para isso utilizamos **dados.iloc[index_column, index_row]**

```
[30]: dados.iloc[0, 2]
```

```
[30]: 'L. Messi'
```

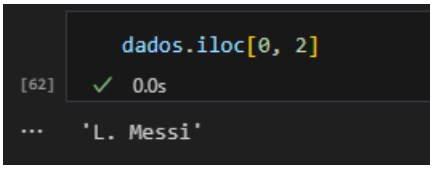
Veja que foi obtido o dado específico da **primeira linha** com o da **terceira coluna**, que nesse caso foi o nome do jogador específico!

Utilizando propriedade de indexação LOC

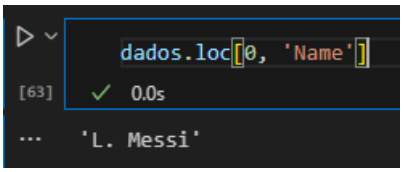
Já vimos como utilizar a propriedade de indexação **iloc[]**, e agora veremos outra propriedade de indexação **loc[]**!!

Mas afinal qual a diferença deles??

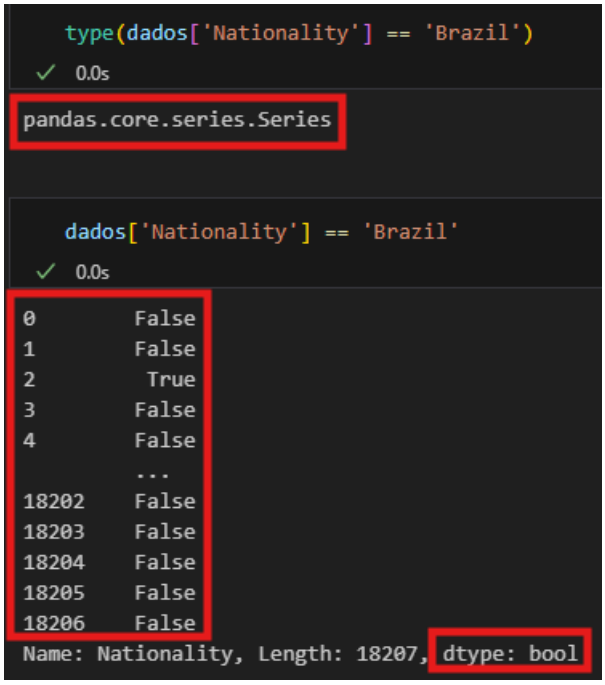
- **iloc[]**: utilizado para trabalhar com indexações numéricos.



- **loc[]**: utilizado para trabalhar com indexações alfanuméricos (string).



Além de poder trabalhar com rótulos em suas indexações, o **loc[]** também pode ser utilizado para aplicar **filtros**!! Pois quando utilizamos filtros **boolean** no **Pandas**, como resposta acaba sendo gerado um objeto **Series Boolean**, como podemos ver a seguir:



E essa estrutura de **Series Boolean** acaba sendo compatível com o uso de indexação com o **loc[]**!! Onde será retornado todos os registros dos **index** que tiver como resultado **True**!!

```
dados.loc[dados['Nationality'] == 'Brazil']
```
✓ 0.0s

	Unnamed: 0	ID	Name	Age	Photo	Nationality	
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofi
27	27	200145	Casemiro	26	https://cdn.sofifa.org/players/4/19/200145.png	Brazil	https://cdn.sofi
32	32	189242	Coutinho	26	https://cdn.sofifa.org/players/4/19/189242.png	Brazil	https://cdn.sofi
35	35	176676	Marcelo	30	https://cdn.sofifa.org/players/4/19/176676.png	Brazil	https://cdn.sofi
39	39	164240	Thiago Silva	33	https://cdn.sofifa.org/players/4/19/164240.png	Brazil	https://cdn.sofi
...	...	...	...	...	...	...	
17059	17059	243750	Gianluca Zanette	21	https://cdn.sofifa.org/players/4/19/243750.png	Brazil	https://cdn.sofi
17446	17446	235949	Strefezza	21	https://cdn.sofifa.org/players/4/19/235949.png	Brazil	https://cdn.sofi
17588	17588	244311	Lucas	19	https://cdn.sofifa.org/players/4/19/244311.png	Brazil	https://cdn.sofi

Utilizando a função de Sorted

Um outro recurso muito importante quando trabalhamos com dados é a utilização da função **sort_values('coluna')**, onde definimos a ordenação do DataFrame pela coluna espedifica.

```
dados.sort_values('Name')
```
✓ 0.1s

	Unnamed: 0	ID	Name	Age	Photo	Nationali
13632	13632	228006	A. Abang	21	https://cdn.sofifa.org/players/4/19/228006.png	Cameroo
15665	15665	243896	A. Abdellaoui	25	https://cdn.sofifa.org/players/4/19/243896.png	Alger
3055	3055	198076	A. Abdennour	28	https://cdn.sofifa.org/players/4/19/198076.png	Tunis
6388	6388	138698	A. Abdi	31	https://cdn.sofifa.org/players/4/19/138698.png	Switzerlan
12816	12816	245438	A. Abdu	21	https://cdn.sofifa.org/players/4/19/245438.png	Britr

Claro que podemos também definir não apenas uma coluna, mas **N** colunas que queremos ordenar, basta definir as colunas em um **array []**!

```
dados.sort_values(['Name', 'Nationality'])
```
✓ 0.2s

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Ov
13632	13632	228006	A. Abang	21	https://cdn.sofifa.org/players/4/19/228006.png	Cameroon	https://cdn.sofifa.org/flags/103.png	
15665	15665	243896	A. Abdellaoui	25	https://cdn.sofifa.org/players/4/19/243896.png	Algeria	https://cdn.sofifa.org/flags/97.png	
3055	3055	198076	A. Abdennour	28	https://cdn.sofifa.org/players/4/19/198076.png	Tunisia	https://cdn.sofifa.org/flags/145.png	
6388	6388	138698	A. Abdi	31	https://cdn.sofifa.org/players/4/19/138698.png	Switzerland	https://cdn.sofifa.org/flags/47.png	

E para finalizar o **Sorted**, podemos ordenar também de forma **decrescente**!! Basta definir o argumento **ascending** como **False**!

```
dados.sort_values('Name', ascending=False)
```

✓ 0.1s

	Unnamed: 0	ID	Name	Age	Photo	Nat
10002	10002	220433	Óscar Whalley	24	https://cdn.sofifa.org/players/4/19/220433.png	
10353	10353	243558	Óscar Valentín	23	https://cdn.sofifa.org/players/4/19/243558.png	
2564	2564	208621	Óscar Plano	27	https://cdn.sofifa.org/players/4/19/208621.png	
9803	9803	229865	Óscar	22	https://cdn.sofifa.org/players/4/19/229865.png	

Adicionando uma coluna no DataFrame

Chegou a hora em manipular o nosso **DataFrame**!! E nesse caso vamos adicionar uma coluna nele!!

E para isso, queremos criar essa coluna com base na somatória dos valores das colunas '**Acceleration**', '**Agility**' e '**Reactions**'.

```
dados['Total'] = dados['Acceleration'] + dados['Agility'] + dados['Reactions']
dados
```

✓ 0.0s Python

Marking	StandingTackle	SlidingTackle	GKDiving	GKHandling	GKKicking	GKPositioning	GKReflexes	Release Clause	Total
33.0	28.0	26.0	6.0	11.0	15.0	14.0	8.0	€226.5M	277.0
28.0	31.0	23.0	7.0	11.0	15.0	14.0	11.0	€127.1M	272.0
27.0	24.0	33.0	9.0	9.0	15.0	15.0	11.0	€228.1M	284.0

Veja como é simples criar colunas no **DataFrame**!! Basta definir o nome da coluna no objeto para atualizar os valores da coluna, como fizemos acima, e caso a coluna não exista será criado automaticamente!

E para melhorar um pouco a visualização dos dados, armazenar um novo **DataFrame** em uma outra variável, e ordenar os seus valores do maior para o menor!

```
dados_total = dados[['Name', 'Total']].sort_values('Total', ascending=False)
dados_total
```

✓ 0.0s

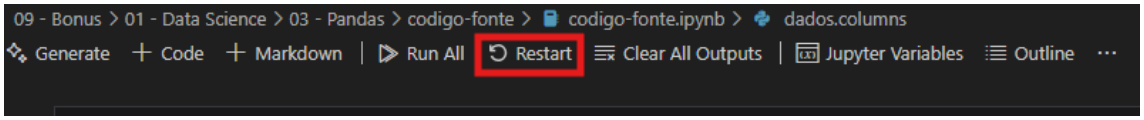
	Name	Total
2	Neymar Jr	284.0
5	E. Hazard	279.0
0	L. Messi	277.0
26	M. Salah	276.0
50	D. Mertens	275.0
...	...	...
13279	P. Mazzocchi	NaN
13280	Y. Ammour	NaN
13281	Jwa Joon Hyeop	NaN
13282	O. Marrufo	NaN
13283	Han Pengfei	NaN

18207 rows × 2 columns

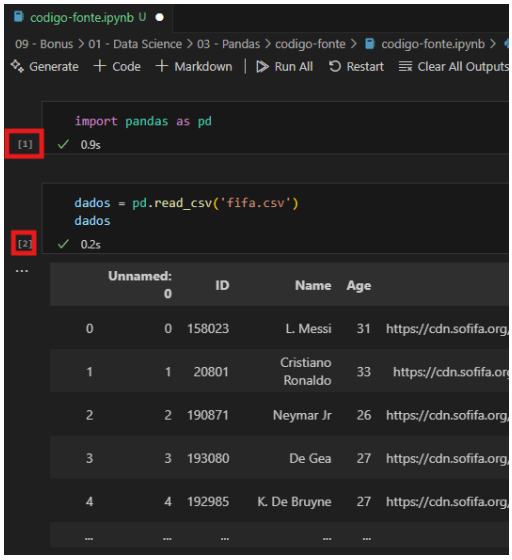
Exportando DataFrame

Enfim vamos exportar o nosso resultado do **DataFrame** em um outro arquivo separado, que no nosso caso será um **.csv**!

Mas inicialmente, vamos reiniciar o **kernel** do **Jupyter**, para seja limpa toda a memória **(variáveis, objetos, funções, etc)**! Pois como mexemos bastante no objeto dados, pode ser que tenha alguma informação desatualizada!



Após reiniciar o **kernel** do **Jupyter**, vamos executar novamente o script do **Notebook Python**, e veremos que tudo foi reiniciado corretamente, tanto que os valores das **células** foram reiniciados também!



Agora sim podemos exportar o nosso resultado em um arquivo **.csv**, e para isso usaremos a função do **DataFrame** chamado **to_csv('caminho_arquivo')**.

```
dados_total.to_csv('result.csv')
```
✓ 0.0s

E pronto, como podemos ver o arquivo foi gerado corretamente!

```
|-------|-----------------------|-------|
|       | Name                  | Total |
|-------|-----------------------|-------|
|     2 | Neymar Jr             | 284.0 |
|-------|-----------------------|-------|
|     5 | E. Hazard             | 279.0 |
|-------|-----------------------|-------|
|     0 | L. Messi              | 277.0 |
|-------|-----------------------|-------|
|    26 | M. Salah              | 276.0 |
|-------|-----------------------|-------|
|    50 | D. Mertens            | 275.0 |
|-------|-----------------------|-------|
```

Porém veja que foi criado uma coluna a mais no início do relatório!! Isso porque automaticamente o Pandas criar com uma coluna a mais referente ao index do **DataFrame** original!!

Dito isso, queremos ignorar a criação dessa coluna automática, e para isso bastar definir o argumento **index** como **false**!

```
dados_total.to_csv('result.csv', index=False)
```
✓ 0.0s

```
|-----------------------|-------|
| Name                  | Total |
|-----------------------|-------|
| Neymar Jr             | 284.0 |
|-----------------------|-------|
| E. Hazard             | 279.0 |
|-----------------------|-------|
| L. Messi              | 277.0 |
```