



DataHackers |



STATE OF DATA
BRAZIL 2025

AULA 01 – PANDAS

ORGANIZAÇÃO DO DATASET

APRESENTAÇÃO

QUEM É ESSE GORDIN CARECA?!

- ✓ Leon Sólon, muito prazer!! 😊
- ✓ Mestre em Computação Aplicada, Doutorando em IA pela Federal de Goiás (pequi power!), pesquisador no CEIA
- ✓ Auditor-Fiscal da RFB (trabalhando com dados)
- ✓ Instrutor e Host do Let's Data
- ✓ Cantor de rock nas horas que sobram...

AO FINAL DESSA AULA VOCÊ VAI SABER

AGENDA

- ✓ O que é pandas e porque ele é o mais usado na área de dados!
- ✓ Enumerar as características de séries e dataframes pandas
- ✓ Importar dados em formato csv no pandas!
- ✓ Organizar o Dataset do State of Data pra começar sua análise VENCEDORA!

O QUE É O PANDAS

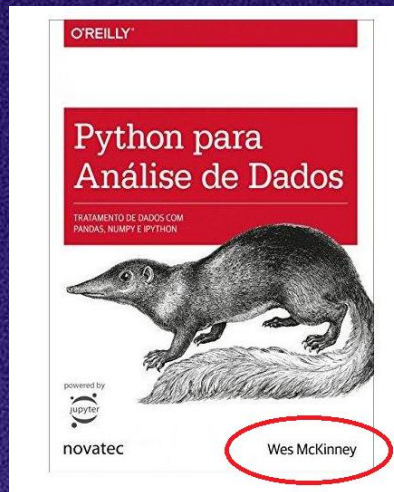
O QUE É PANDAS



O QUE É PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Biblioteca Python criada por Wes McKinney para manipulação de dados



O QUE É PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Estruturas tabulares chamadas dataframe (parece uma planilha excel)
- ✓ Principais estruturas:
 - ✓ Series: unidimensionais
 - ✓ DataFrame: dados bidimensionais
- ✓ Facilita a leitura de uma variedade de formatos de arquivo, como CSV, JSON e Excel
- ✓ Permite limpeza, transformação, visualização e análise de dados.

O QUE É PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Saber bem o pandas é muito importante!
- ✓ Manipulação de dados é uma grande parte do tempo investido em qualquer carreira de dados
- ✓ Bora aprender!!

POR QUE PANDAS?

POR QUE PANDAS?

Dados

Você



POR QUE PANDAS?

CONCEITOS FUNDAMENTAIS

- ✓ Biblioteca mais utilizada para manipulação de dados
DISPARADO
- ✓ Possui alguns “concorrentes”, como o Polars, mas o uso no mercado é escandalosamente maior
- ✓ Comando simples, mas poderosos, para tirar insights e tratar os dados
- ✓ Muitas funcionalidades relativamente simples

POR QUE PANDAS?

CONCEITOS FUNDAMENTAIS

- ✓ Fornece de pronto muitas funcionalidades para preparação de dados: merging (join/procv), ordenação, seleção de linhas e colunas, agregação etc.
- ✓ Permite a criação de novas colunas derivadas (importante para machine learning)
- ✓ Permite o tratamento de missing data (essencial para machine learning)

POR QUE PANDAS?

CONCEITOS FUNDAMENTAIS

- ✓ Carreiras de dados requerem soft skills
- ✓ MAS NÃO SE ENGANE, PADAWAN!
- ✓ Saber manipular dados de forma “automática” é um diferencial GIGANTE!
- ✓ Ou seja, pratique bastante até ficar “automático”

SÉRIES PANDAS

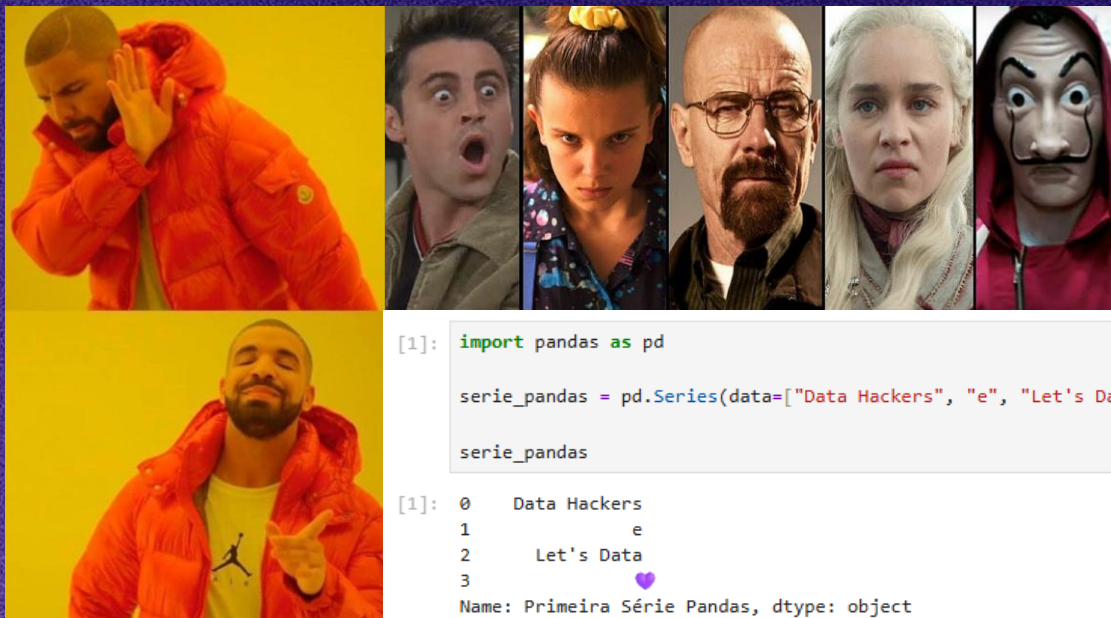
SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Duas estruturas principais: Series e DataFrames
- ✓ Series: estrutura unidimensional
- ✓ DataFrames: estrutura bidimensional, uma coleção de Series

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS



SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Séries são muito parecidas com listas comuns de Python...



SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ ... mas com rodas de liga leve



SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

✓ Lista:

```
x = [1,2,3,4,5]
```

✓ Série pandas:

```
y = pd.Series([1,2,3,4,5])
```


SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Estrutura de dados bem similar a listas, mas com umas coisinhas a mais!
- ✓ Possui QUATRO elementos principais:
 - ✓ Valores (não diga!?)
 - ✓ Índices
 - ✓ Tipo
 - ✓ Nome

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Valores podem ser de qualquer tipo
- ✓ Quando valores são do mesmo tipo, facilita transformação interna para Numpy arrays (mais performance)
- ✓ Ou seja, vamos tentar manter o mesmo tipo!
- ✓ Na criação da Series, passar como primeiro parâmetro ou no named parameter "data"

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Criar séries pandas a partir de lista de valores

```
import pandas as pd

s1 = pd.Series(data=["Data Hackers", "e", "Let's Data", "💜"])
s2 = pd.Series(["Data Hackers", "e", "Let's Data", "💜"])

display(s1)
display(s2)
```

```
0    Data Hackers
1              e
2    Let's Data
3             💜
dtype: object
0    Data Hackers
1              e
2    Let's Data
3             💜
dtype: object
```


SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Índices são uma parte ESSENCIAL de pandas
- ✓ Diferente das listas, em que acessamos de forma somente posicional (começando do 0)
- ✓ Os índices nas Series (e Dataframes) pandas podem ser de qualquer tipo, desde que sejam “hasheáveis”, ou seja, com acesso direto
- ✓ Parece estranho, mas... os índices não precisam ser únicos

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Diferente das listas, em que acessamos de forma somente posicional (começando do 0)

```
import pandas as pd

valores_lista = [100, 200, 300]
pd.Series(data=valores_lista)
```

```
0    100
1    200
2    300
dtype: int64
```


SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ “Uai, Leon, mas isso não é igual às listas comuns no Python??? Começou do zero e é posicional”

```
import pandas as pd
```

```
valores_lista = [100, 200, 300]
```

```
pd.Series(data=valores_lista)
```

```
0    100
```

```
1    200
```

```
2    300
```

```
dtype: int64
```


SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Muito bem observado, padawan! A diferença da série pandas para uma lista é que, nas séries, podemos criar algo que não remeta à posição!

```
import pandas as pd

valores_lista = [100, 200, 300]
valores_indices = ['indice1', 'indice2', 'indice3']
pd.Series(data=valores_lista, index=valores_indices)
```

```
indice1    100
indice2    200
indice3    300
dtype: int64
```


SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Mais ainda! Podemos definir índices numéricos, mas que NÃO REMETEM À POSIÇÃO DOS ELEMENTOS numa série pandas

```
In [1]: lista = ["A","B","C","D"]  
        lista[0]
```

```
Out[1]: 'A'
```

```
In [2]: import pandas as pd  
  
        s = pd.Series(["A","B","C","D"], index=[1,4,12341,0] )  
        s[0]
```

```
Out[2]: 'D'
```



SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Mais ainda! Podemos definir índices numéricos, mas que NÃO REMETEM À POSIÇÃO DOS ELEMENTOS numa série pandas

```
In [13]: import pandas as pd
```

```
s = pd.Series(["A","B","C","D"], index=[1,4,12341,0] )  
s[0]
```

```
Out[13]: 'D'
```

```
In [14]: s
```

```
Out[14]: 1      A  
         4      B  
        12341   C  
         0      D  
         dtype: object
```


SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Lembrando que os índices não precisam ser chaves (não precisam ser únicos)!
- ✓ Podemos ter valores de índices iguais para mais de uma linha.
- ✓ Quando buscamos pelo índice, ele traz todos os elementos “apontados” por aquele índice (todas as linhas que tem referência daquele índice)

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Quando buscamos pelo índice, ele traz todos os elementos “apontados” por aquele índice (todas as linhas que tem referência daquele índice)

```
import pandas as pd

serie = pd.Series(data=["Data Hackers", "e", "Let's Data", "❤️"], index=[0,1,0,2])
serie[0]
```

```
0    Data Hackers
0      Let's Data
dtype: object
```



SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ As séries pandas tem um TIPO
- ✓ Se não definirmos o tipo, o pandas infere a partir dos dados
- ✓ Inferir = bom e velho CHUTÔMETRO
- ✓ Mas é um chutômetro esertinho, se só tem valores inteiros, por exemplo, o tipo será int

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

```
import pandas as pd

s1 = pd.Series(["Data Hackers", "e", "Let's Data", "❤️"])
s2 = pd.Series([1, 2, 3])
s3 = pd.Series([.1, .2, .3])

print(s1.dtype)
print(s2.dtype)
print(s3.dtype)
```

object
int64
float64

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Mas a gente pode sugerir um tipo com o parâmetro dtype

```
In [6]: s3 = pd.Series([1., 2., 3.], dtype=int)
```

```
s3
```

```
Out[6]: 0    1  
        1    2  
        2    3  
        dtype: int32
```

* cuidado pra ter certeza que a conversão pode ser feita

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Mesmo depois de um tipo definido, tem como mudar com `astype`

```
In [9]: s2 = pd.Series([1,2,3])
```

```
print(s2.dtype)
```

```
s2 = s2.astype(float)
```

```
print(s2.dtype)
```

```
int64
```

```
float64
```

* cuidado pra ter certeza que a conversão pode ser feita

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Finalmente, podemos especificar um nome para a série!*

```
import pandas as pd

s1 = pd.Series(["Data Hackers", "e", "Let's Data", "💜"], name="Para de pensar no Switch 2")

s1
```

0	Data Hackers
1	e
2	Let's Data
3	💜

Name: Para de pensar no Switch 2, dtype: object

* Isso será muito útil nos dataframes pandas, pois cada coluna é uma série pandas com um.... nome

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Pra acessar os valores, índices, nome e tipos de uma série pandas “s”:
 - ✓ s.values
 - ✓ s.index
 - ✓ s.name
 - ✓ s.dtype

SÉRIES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Pra acessar os valores, índices, nome e tipos de uma série pandas:

```
# Série pandas:
```

```
s1 = pd.Series([1,2,3,4],  
               name="primeira serie pandas",  
               index=["um", "dois", "três", "quatro"])
```

```
print(s1.values)  
print(s1.index)  
print(s1.name)  
print(s1.dtype)
```

```
[1 2 3 4]  
Index(['um', 'dois', 'três', 'quatro'], dtype='object')  
primeira serie pandas  
int64
```


DATAFRAMES PANDAS

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Duas estruturas principais: Series e DataFrames
- ✓ Series: estrutura unidimensional
- ✓ DataFrames: estrutura bidimensional, uma coleção de Series

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Imaginem uma planilha Excel...

	A	B	C	D
1	Secteurs	Clients	CA 2014	CA 2015
2	Agriculture	Agro	21000	35000
3	Industrie	SBFM	34000	27000
4	Service	Veolia	56000	26300
5	Commerce	STTE	82000	64800
6	Transport	Bernard	64000	94050
7	Pêche	Thonier	78000	63400
8	Construction	Bouygues	63000	95600
9	Logistique	Geodis	44000	58700

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

✓ ... com superpoderes



DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Lista bidimensional (lista de listas):

```
x = [[1,2],[3,4],[5,6]]
```

- ✓ Dataframe pandas:

```
y = pd.DataFrame ([[1,2],[3,4],[5,6]]) # 3 linhas, 2 colunas
```


DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Principais características

```
pd.DataFrame(  
DADOS  ➡ data=None,  
ÍNDICES ➡ index: Union[Collection, NoneType] = None,  
COLUNAS* ➡ columns: Union[Collection, NoneType] = None,  
TIPOS  ➡ dtype: Union[ForwardRef('ExtensionDtype'), str, numpy.dtype, Type[Union[str, float,  
int, complex, bool]]], NoneType] = None,  
        copy: bool = False,  
)
```

* Na verdade, só os nomes das colunas, os dados mesmo estão no “data”

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Principais características

```
import pandas as pd

df = pd.DataFrame(data=[['costela', 30.], ['camarão', 60.]],
                  index=['linha um', 'linha dois'],
                  columns=['prato', 'preço'])

df
```

	prato	preço
linha um	costela	30.0
linha dois	camarão	60.0

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Valores da “tabela”
- ✓ Pode ser importados ou incluídos na criação do DataFrame
- ✓ Importação: suporta uma grande variedade de formatos
- ✓ Inclusão: listas, conjuntos, dicionários

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Criando dataframe a partir de uma lista (list)

```
lista = ["Leon", "só", "pensa", "em", "comida"]  
pd.DataFrame(lista)
```

	0
0	Leon
1	só
2	pensa
3	em
4	comida

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Criando dataframe a partir de um conjunto (set)

```
lista = ["Leon", "só", "pensa", "em", "comida"]  
conjunto = set(lista)  
  
pd.DataFrame(conjunto)
```

	0
0	em
1	pensa
2	comida
3	só
4	Leon

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Criando dataframe a partir de um dicionário (dict)

```
dicionario = {'prato': ['costela', 'camarão'], 'preço': [30., 60.]}  
  
df = pd.DataFrame(dicionario)  
df
```

	prato	preço
0	costela	30.0
1	camarão	60.0

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Criando dataframe a partir de arquivos

```
pd.read_csv('teste.csv')  
pd.read_table('teste.csv', sep=',')  
pd.read_excel('arquivo.xlsx', sheet_name='Sheet1')  
pd.read_stata('myfile.dta')  
pd.read_sas('sas_ainda existe.sas7bdat')  
pd.read_hdf('arquivo_hadoop.h5', 'df')  
pd.read_parquet('arquivao.parquet')  
pd.read_html(resultado_request.text)
```


DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

Importando um DataFrame da Web

```
df_medalhas = pd.read_csv("https://raw.githubusercontent.com/letsdata/arquivos-jcd/main/medals.csv")  
df_medalhas.head()
```

	Year	City	Sport	Discipline	NOC	Event	Event gender	Medal
0	1924	Chamonix	Skating	Figure skating	AUT	individual	M	Silver
1	1924	Chamonix	Skating	Figure skating	AUT	individual	W	Gold
2	1924	Chamonix	Skating	Figure skating	AUT	pairs	X	Gold
3	1924	Chamonix	Bobsleigh	Bobsleigh	BEL	four-man	M	Bronze
4	1924	Chamonix	Ice Hockey	Ice Hockey	CAN	ice hockey	M	Gold

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Para acessar os dados do DataFrame:

```
df.values
```

```
array([[ 'costela', 30.0],  
       [ 'camarão', 60.0]], dtype=object)
```


DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Para acessar os índices do DataFrame

```
df.index
```

```
Index(['linha1', 'linha2'], dtype='object')
```


DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Para acessar os nomes das colunas do DataFrame

```
df.columns
```

```
Index(['prato', 'preço'], dtype='object')
```


DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Acessar as colunas do DataFrame

```
df['prato']
```

```
0    costela  
1    camarão  
Name: prato, dtype: object
```

```
df.prato
```

```
0    costela  
1    camarão  
Name: prato, dtype: object
```

```
df['preço']
```

```
0    30.0  
1    60.0  
Name: preço, dtype: float64
```

```
df.preço
```

```
0    30.0  
1    60.0  
Name: preço, dtype: float64
```


DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Colunas no pandas DataFrames nada mais são que.... séries pandas!

```
print(type(df['prato']))  
print(type(df['preço']))
```

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


DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ Colunas no pandas DataFrames nada mais são que.... séries pandas!

ÍNDICES



```
df['preço']
```

linha1 30.0



DADOS

linha2 60.0

Name: preço, dtype: float64



NOME



TIPO

DATAFRAMES PANDAS

CONCEITOS FUNDAMENTAIS

- ✓ MUITO IMPORTANTE ENTENDER!!!
- ✓ Índices não são necessariamente números posicionais!!!
- ✓ Podem ser de muitos tipos diferentes
- ✓ As colunas tem nomes, porque as linhas não teriam?!
- ✓ É como se pudéssemos criar nomes para as linhas numa planilha Excel

DATAFRAMES PANDAS

ÍNDICES

```
dicionario = {'prato': ['costela', 'camarão'], 'preço': [30., 60.] }  
  
df = pd.DataFrame(dicionario)  
df
```

	prato	preço
0	costela	30.0
1	camarão	60.0

DATAFRAMES PANDAS

ÍNDICES

```
import pandas as pd

df = pd.DataFrame(data=[['costela', 30.], ['camarão', 60.]],
                  index=['linha um', 'linha dois'],
                  columns=['prato', 'preço'])

df
```

	prato	preço
linha um	costela	30.0
linha dois	camarão	60.0

DATAFRAMES PANDAS

ÍNDICES

```
pd.DataFrame([1, 2], index=[37, 'chocolate'])
```

	0
37	1
chocolate	2

```
pd.DataFrame([1, 2], index=[{'eu': 'só'}, {'quero': 'chocolate'}])
```

	0
{'eu': 'só'}	1
{'quero': 'chocolate'}	2

DATAFRAMES PANDAS

ÍNDICES

- ✓ Se cada coluna é uma Série pandas, cada coluna pode ter um tipo diferente
- ✓ Elas são derivadas dos tipos do Numpy (dtypes) ou tipos básicos do python
- ✓ Se não definirmos o tipo, o pandas infere a partir dos dados

DATAFRAMES PANDAS

ÍNDICES

- ✓ Para acessar os tipos das colunas (séries) de um DataFrame:

```
df.dtypes
```

```
prato      object  
preço     float64  
dtype: object
```

```
df_medalhas.dtypes
```

```
Year          int64  
City          object  
Sport         object  
Discipline    object  
NOC           object  
Event         object  
Event gender  object  
Medal         object  
dtype: object
```

DATAFRAMES PANDAS

ÍNDICES

- ✓ Para converter os tipos das colunas (séries) de um DataFrame:

```
df['preço'].astype(int)
```

```
0    30  
1    60
```

```
Name: preço, dtype: int32
```

```
df['preço'].astype(float)
```

```
0    30.0  
1    60.0
```

```
Name: preço, dtype: float64
```

```
df['preço'].astype(str)
```

```
0    30.0  
1    60.0
```

```
Name: preço, dtype: object
```


FILTROS LOC E ILOC

MANIPULAÇÃO DE DADOS

- ✓ Pra que serve pandas???
- ✓ Pra manipular dados!!!
- ✓ Pra isso precisamos ficar CRAQUES em acessar, filtrar, alterar os dados com muita destreza
- ✓ Aprender bem as formas de filtragem de dados, tratamento e criação de novas colunas é um BAITA diferencial em qualquer carreira de dados

FILTROS LOC E ILOC

MANIPULAÇÃO DE DADOS

- ✓ “Uai, Leon, e o negócio?”
- ✓ É o nosso norte, o nosso foco, mas ciência de dados ainda precisa muito de “sujar as mãos” com dados
- ✓ Quanto mais prática tiver em pandas, mais rápido vai conseguir chegar em bons resultados de negócio
- ✓ Essa aula portanto é ESSENCIAL!

FILTROS LOC E ILOC

FILTROS LOC E ILOC

- ✓ Acesso direto pelo índice com colchetes: `df['coluna1']` ou `df.coluna1`
- ✓ `loc`: acessa conteúdo pelo índice: `df.loc['indice1']`
- ✓ `iloc`: acessa conteúdo pela posição: `df.iloc[0]`

FILTROS LOC E ILOC

FILTROS LOC E ILOC

- ✓ Acesso direto pelo nome da coluna: retorna a coluna inteira*

```
df_state_of_data['1.b_genero']
```

```
0      Masculino
1      Masculino
2      Masculino
3      Masculino
4      Masculino
```

...

```
5212    Feminino
5213    Masculino
5214    Feminino
5215    Masculino
5216    Feminino
```

Name: 1.b_genero, Length: 5217, dtype: object

```
df_state_of_data['1.c_cor/raca/etnia']
```

```
0      Branca
1      Branca
2      Parda
3      Branca
4      Branca
```

...

```
5212    Branca
5213    Branca
5214    Parda
5215    Parda
5216    Parda
```

Name: 1.c_cor/raca/etnia, Length: 5217, dtype: object

FILTROS LOC E ILOC

FILTROS LOC E ILOC

- ✓ loc: acesso pelo índice (retorna a LINHA inteira)

```
df.loc['linha1'] ← ÍNDICE
```

```
prato    costela  
preço      30.0  
Name: linha1, dtype: object
```

ÍNDICE →

```
df = pd.DataFrame(data=[['costela', 30.], ['camarão', 60.]],  
                  index=['linha1', 'linha2'],  
                  columns=['prato', 'preço'])  
display(df)
```

	prato	preço
linha1	costela	30.0
linha2	camarão	60.0

FILTROS LOC E ILOC

FILTROS LOC E ILOC

- ✓ `iloc`: acesso pela POSIÇÃO! (retorna a LINHA inteira)

```
df.iloc[0] ← POSIÇÃO
```

```
prato    costela  
preço      30.0  
Name: linha1, dtype: object
```

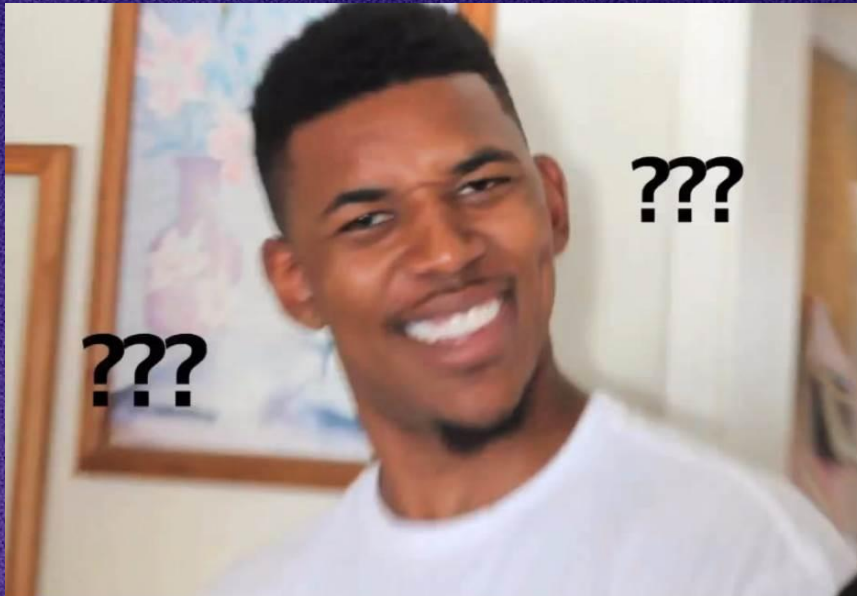
1º LINHA →

```
df = pd.DataFrame(data=[['costela', 30.], ['camarão', 60.]],  
                  index=['linha1', 'linha2'],  
                  columns=['prato', 'preço'])  
display(df)
```

	prato	preço
linha1	costela	30.0
linha2	camarão	60.0

FILTROS LOC E ILOC

FILTROS LOC E ILOC




FILTROS LOC E ILOC

FILTROS LOC E ILOC

- ✓ Filtrros!!! Usar o loc nos permite utilizar muitos filtros, serão MUITO úteis no nosso dia a dia

```
df_state_of_data.loc[df_state_of_data['1.b_genero'] == 'Feminino']
```



	0.a_token	1.d_data/hora_envio	1.a_idade	1.a.1_faixa_idade	1.b_genero	1.c_cor/raca/etnia
15	qlirs7btxom1i11ytqlijz5bmfsyb9vg	14/10/2024 16:49:44	19	17-21	Feminino	Branca
21	t15okkgf7aekvsit15s9yrf5ohmkbvawi	17/10/2024 06:27:08	19	17-21	Feminino	Branca
26	6b0uy326ywo4e9rq51lj6b0uy326u8v5	22/10/2024 11:45:51	19	17-21	Feminino	Branca
30	jrwjq4ryrap990q80jrwjqizv8439do4	23/10/2024 01:54:32	19	17-21	Feminino	Parda
38	7v48by2r1cafeq0g7v4ew0slx6obelxe	30/10/2024 17:07:39	19	17-21	Feminino	Parda

FILTROS LOC E ILOC

FILTROS LOC E ILOC

- ✓ Mais de um filtro! Cuidado: usar parênteses e “e comercial”

```
df_state_of_data.loc[(df_state_of_data['1.b_genero'] == 'Feminino') & (df_state_of_data['1.c_cor/raca/etnia'] == 'Parda')]
```



	0.a_token	0.d_data/hora_envio	1.a_idade	1.a.1_faixa_idade	1.b_genero	1.c_cor/raca/etnia
30	jrwjq4ryrap990q80jrwjqizv8439do4	23/10/2024 01:54:32	19	17-21	Feminino	Parda
38	7v48by2r1cafeq0g7v4ew0slx6obelxe	30/10/2024 17:07:39	19	17-21	Feminino	Parda
41	1w30ys1cmbi7q7u1we228h9otcmnaoh2	05/11/2024 11:51:01	19	17-21	Feminino	Parda
64	p7kspgp8s5yggjp70uadletdcr7uddv0u	17/10/2024 01:51:07	20	17-21	Feminino	Parda
77	ucbc1vzyiaawl6uu559jbucbc1vw9rkW	01/11/2024 18:09:51	20	17-21	Feminino	Parda

FILTROS LOC E ILOC

FILTROS LOC E ILOC

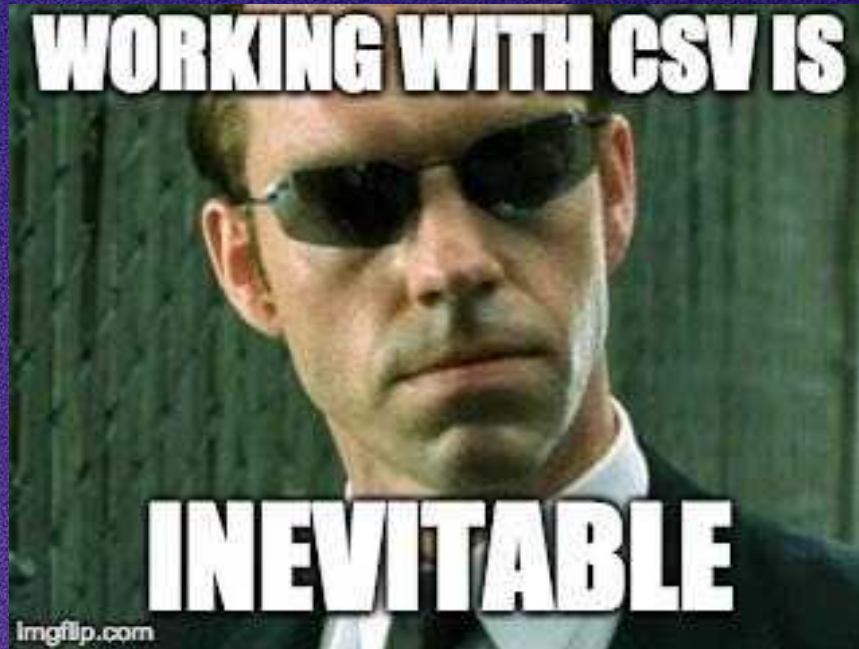
- ✓ Com o iloc temos os acessos pelas POSIÇÕES!

```
df_state_of_data.iloc[0:3]
```



	0.a_token	0.d_data/hora_envio	1.a_idade	1.a.1_faixa_idade	1.b_genero	1.c_cor/raca/etnia
0	reb94rv0msth7q4nreb94riaq80iz3yi	16/10/2024 11:19:17	18	17-21	Masculino	Branca
1	1zc66g69jtt49y32l1zc66g8wqj79m4e	16/10/2024 20:45:31	18	17-21	Masculino	Branca
2	uu99wmam4n5kc2uu99wmydf0rk7l58f7	17/10/2024 18:10:59	18	17-21	Masculino	Parda

ARQUIVOS CSV



ARQUIVOS CSV

CONCEITOS BÁSICOS

- ✓ Porque desde o noob padawan iniciante até o megablaster Kaggle Grandmaster utiliza arquivos CSV no seu dia a dia
- ✓ Porque é o mais simples de todos, mas também pode dar umas dorezinhas de cabeça para quem está começando
- ✓ Acho que já entendeu que é importante, né?

ARQUIVOS CSV

CONCEITOS BÁSICOS

- ✓ Nada mais que um arquivo texto que tem suas “colunas” separadas por... Vírgula
- ✓ Mas também pode ser ponto e vírgula, tabulação, dois pontos, pipe (a gente encontra cada loucura....)
- ✓ Uma das formas mais simples de representar dados tabulares
- ✓ Geralmente a primeira linha define o “cabeçalho” com os nomes das colunas

ARQUIVOS CSV

CONCEITOS BÁSICOS

- ✓ Talvez por conta do Excel que acaba sendo o programa no Windows mais utilizado, você não saiba que o CSV é um mero arquivo texto
- ✓ Tente abrir um arquivo com qualquer editor de textos (bloco de notas, por exemplo)
- ✓ Não tem nada de muito complexo, portanto

ARQUIVOS CSV

CONCEITOS BÁSICOS

✓ Exemplo:

banda,musica,album,estilo

Radiohead,Creep,Pablo Honey,indie rock

The Cure,Inbetween Days,The Head on the Door,pós-punk

The Beatles,Tomorrow Never Knows,Revolver,rock psicodélico

ARQUIVOS CSV

CONCEITOS BÁSICOS

✓ Exemplo:

banda;musica;álbum;estilo

Radiohead;Creep;Pablo Honey;indie rock

The Cure;Inbetween Days;The Head on the Door;pós-punk

The Beatles;Tomorrow Never Knows;Revolver;rock psicodélico

ARQUIVOS CSV

CONCEITOS BÁSICOS

✓ “Mas se é tão simples, qual a dificuldade?”



ARQUIVOS CSV

CONCEITOS BÁSICOS

- ✓ Separador de colunas
- ✓ Codificação
- ✓ Casas decimais
- ✓ Separadores de milhar
- ✓ Formatos de Datas
- ✓ Tamanho!

E O DATASET, MALUCO?!

BORA PRO MÃO NA MASSA!