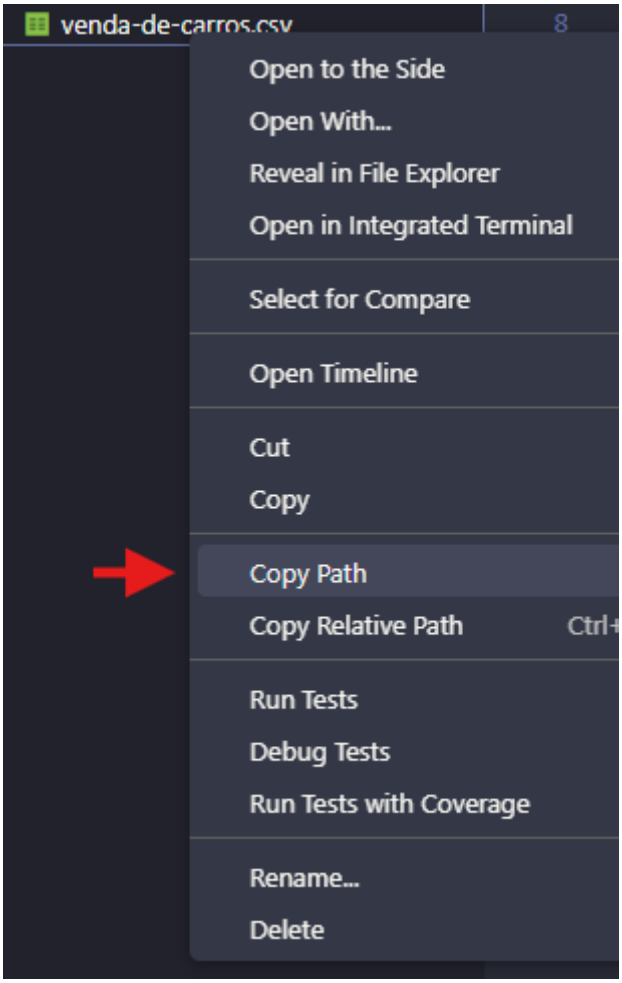
Resumo e exemplos baseados nos vídeos 4 ao 22 do curso de pandas:

#03 Importando um base de dados

Método **.read_csv()** serve para importar uma base de dados.

Para saber o caminho do arquivo, no VS CODE basta clicar com botão direito e copiar o caminho:



Comando para importar, substitua caminho da sua base de dados pelo caminho que copiou acima:

```
import pandas as pd
dados = pd.read_csv(r"caminho da sua base de dados")
```

[ ]

Após colocar o caminho, para exigir a base de dados importada basta escrever o nome dela abaixo (aqui funciona como um print):

```
import pandas as pd
dados = pd.read_csv(r"C:\Users\Youx\Documents\Lab\venda-de-carros.csv")
dados
```

[13] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco
0	Toyota	Branco	150043	4	R$ 24,000.00
1	Honda	Vermelho	87899	4	R$ 25,000.00
2	Toyota	Azul	32549	3	R$ 27,000.00
3	BMW	Preto	11179	5	R$ 122,000.00
4	Nissan	Branco	213095	4	R$ 13,500.00
5	Toyota	Verde	99213	4	R$ 14,500.00
6	Honda	Azul	45698	4	R$ 17,500.00
7	Honda	Azul	54738	4	R$ 27,000.00
8	Toyota	Branco	60000	4	R$ 26,250.00
9	Nissan	Branco	31600	4	R$ 19,700.00

Método .**head()** exibe as 5 primeiras linhas da base de dados por padrão, ou quantas linhas eu quiser passando o valor dentro do head:

```
dados.head()
```

[14] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco
0	Toyota	Branco	150043	4	R$ 24,000.00
1	Honda	Vermelho	87899	4	R$ 25,000.00
2	Toyota	Azul	32549	3	R$ 27,000.00
3	BMW	Preto	11179	5	R$ 122,000.00
4	Nissan	Branco	213095	4	R$ 13,500.00

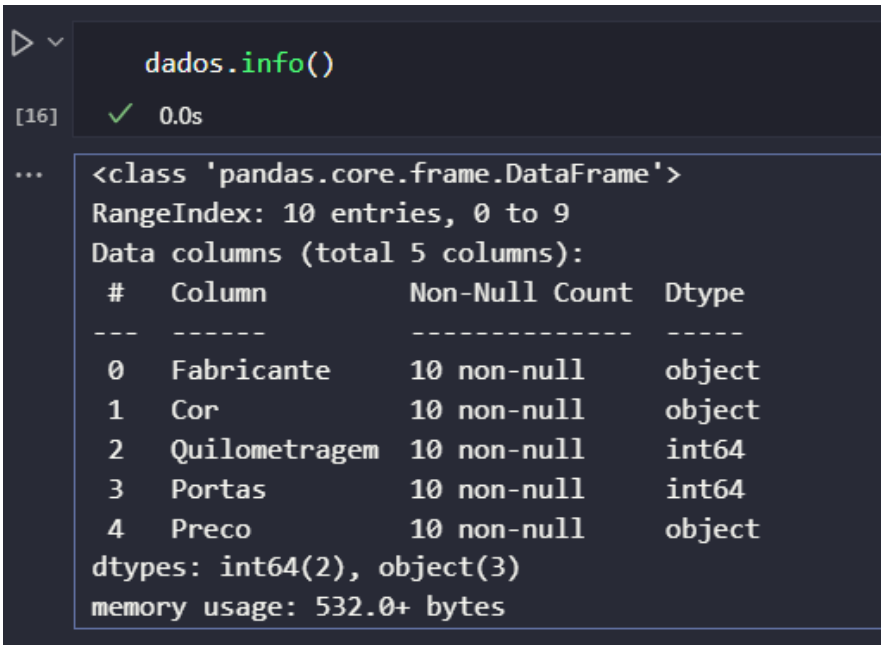
```
dados.head(6)
```

[15] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco
0	Toyota	Branco	150043	4	R$ 24,000.00
1	Honda	Vermelho	87899	4	R$ 25,000.00
2	Toyota	Azul	32549	3	R$ 27,000.00
3	BMW	Preto	11179	5	R$ 122,000.00
4	Nissan	Branco	213095	4	R$ 13,500.00
5	Toyota	Verde	99213	4	R$ 14,500.00

#4 Informações da base de dados

Método **.info()** exibe as informações da base:



<class 'pandas.core.frame.DataFrame'> informa que a base é um DataFrame (formato de base de dados no python).

RangeIndex: 10 entries, 0 to 9: informa que tem 10 linhas, e o index vai de 0 até 9.

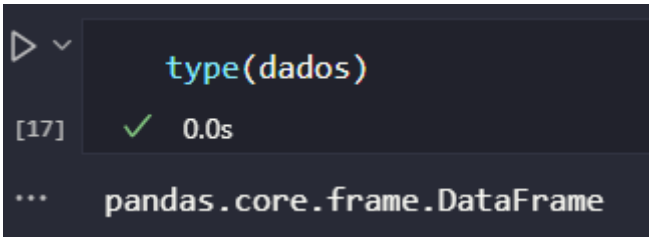
Non-Null Count: informa se possui valores nulos em cada coluna.

Dtype: informa o tipo de dado de cada coluna.

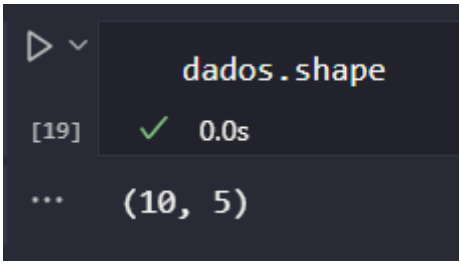
#05 DataFrame

Toda base de dados importada pro python, é um DataFrame (tabela bi-dimensional)

Função **type()** mostra o tipo:



O atributo **.shape** retorna quantas linhas e colunas tem no DataFrame:



Criando um DataFrame:

Função **.DataFrame()** do pandas cria um DataFrame, podemos passar um dicionário para criar:

```
personagens_df = pd.DataFrame({
    'nome': ['Luke Skywalker', 'Yoda', 'Palpatine'],
    'idade': [16, 1000, 70],
    'peso': [70.5, 15.2, 60.1],
    'eh jedi': [True, True, False]
})

personagens_df
```

[21] ✓ 0.0s

	nome	idade	peso	eh jedi
0	Luke Skywalker	16	70.5	True
1	Yoda	1000	15.2	True
2	Palpatine	70	60.1	False

Renomeando colunas:

Atributo **.columns** retorna uma “lista” com nome de todas as colunas, também é possível armazenar em uma lista:

```
personagens_df.columns
```

[22] ✓ 0.0s

```
Index(['nome', 'idade', 'peso', 'eh jedi'], dtype='object')
```

```
lista_personagens = list(personagens_df.columns)
lista_personagens
```

[24] ✓ 0.0s

```
['nome', 'idade', 'peso', 'eh jedi']
```

O método **.rename()** serve para renomear as colunas passando um dicionário, no exemplo abaixo estamos criando um novo DataFrame com as colunas renomeadas:

```
personagens_df_renomeado = personagens_df.rename(columns={
    'nome': 'Nome Completo',  # renomeia a coluna de nome 'nome' para 'Nome Completo'
    'idade': 'Idade'
})

personagens_df_renomeado
```

[25] ✓ 0.0s

	Nome Completo	Idade	peso	eh jedi
0	Luke Skywalker	16	70.5	True
1	Yoda	1000	15.2	True
2	Palpatine	70	60.1	False

É possível também renomear o DataFrame original, nesse caso personagens_df, e continuar utilizando-o, basta colocar **inplace = True**, veja o exemplo:

```
personagens_df.rename(columns={
    'nome': 'Nome Completo',
    'idade': 'Idade'},
    inplace=True)

personagens_df
```

[26] ✓ 0.0s

	Nome Completo	Idade	peso	eh jedi
0	Luke Skywalker	16	70.5	True
1	Yoda	1000	15.2	True
2	Palpatine	70	60.1	False

Outra forma de renomear é utilizando **.columns** e passando uma lista com os novos nomes:

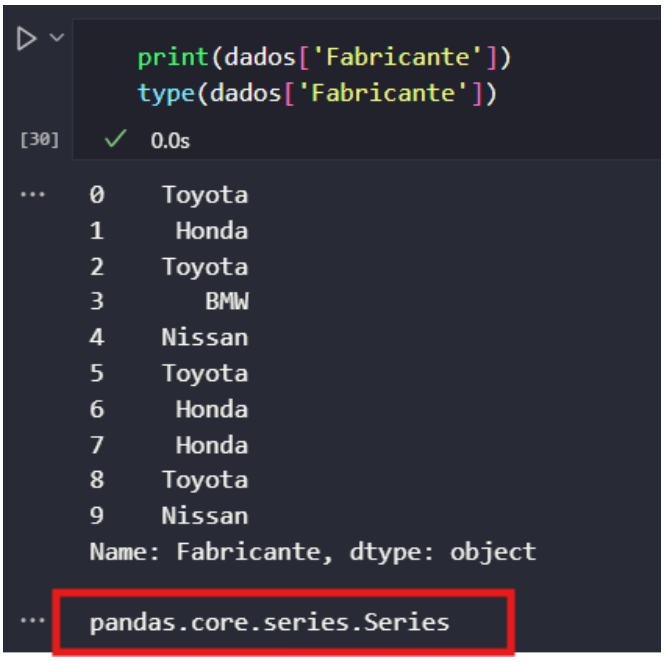
```
personagens_df.columns = ['NOME', 'IDADE', 'PESO', 'EH_JEDI']

personagens_df
```

[27] ✓ 0.0s

	NOME	IDADE	PESO	EH_JEDI
0	Luke Skywalker	16	70.5	True
1	Yoda	1000	15.2	True
2	Palpatine	70	60.1	False

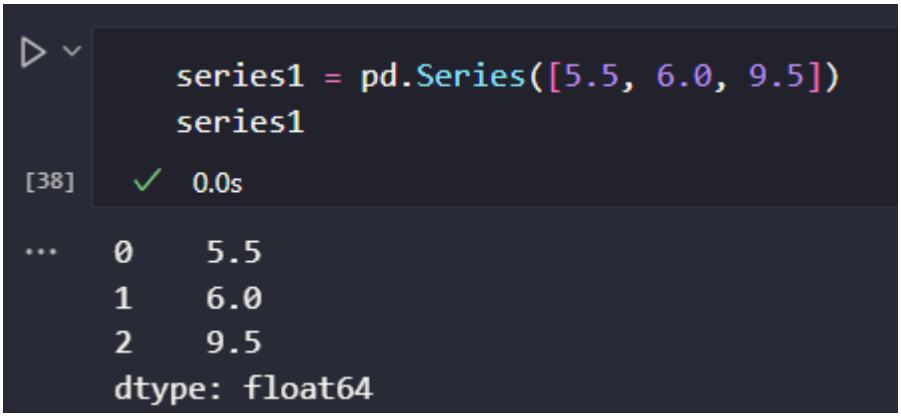
#06 Series

Array uni-dimensional com os dados e rótulos de um eixo. Exemplo: uma coluna da base de dados (vamos voltar pra base que chama dados):



Criando uma Series:

O método **.Series()** cria uma series, basta passar uma lista, ex:



Também é possível modificar o index e o nome:

Para criar uma cópia, precisamos utilizar o **.copy()**

```
cor_copia = dados['Cor'].copy
cor_copia
```

[42] ✓ 0.0s

```
<bound method NDFrame.copy of 0      Branco
1    Vermelho
2        Azul
3       Preto
4      Branco
5       Verde
6        Azul
7        Azul
8      Branco
9      Branco
Name: Cor, dtype: object>
```

Para gerar DataFrame a partir de series, podemos usar o **.concat()**:

```
series1 = pd.Series([5.5, 6.0, 9.5], index=['prova 1', 'prova 2', 'projeto'], name='Notas dos Luke Skywalker')
series2 = pd.Series([8.2, 6.5, 12.5], index=['prova 1', 'prova 2', 'projeto'], name='Notas dos Lakers ')

dados_series = pd.concat([series1, series2], axis=1)
dados_series
```

[132] ✓ 0.0s

	Notas dos Luke Skywalker	Notas dos Lakers
prova 1	5.5	8.2
prova 2	6.0	6.5
projeto	9.5	12.5

#08 Criando colunas

Podemos criar uma nova coluna a nossa base passando o nome dela, e o que queremos para ela, ex:

```
dados['Nova coluna'] = 'Testando'
dados
```

[60] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco	Nova coluna
0	Toyota	Branco	150043	4	R$ 24,000.00	Testando
1	Honda	Vermelho	87899	4	R$ 25,000.00	Testando
2	Toyota	Azul	32549	3	R$ 27,000.00	Testando
3	BMW	Preto	11179	5	R$ 122,000.00	Testando
4	Nissan	Branco	213095	4	R$ 13,500.00	Testando
5	Toyota	Verde	99213	4	R$ 14,500.00	Testando
6	Honda	Azul	45698	4	R$ 17,500.00	Testando
7	Honda	Azul	54738	4	R$ 27,000.00	Testando
8	Toyota	Branco	60000	4	R$ 26,250.00	Testando
9	Nissan	Branco	31600	4	R$ 19,700.00	Testando

Podemos criar também utilizando outra coluna, como por exemplo, criar uma coluna multiplicando a Quilometragem por 2:

```
dados['Quilometragem2'] = dados['Quilometragem'] * 2
dados
```

[66] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco	Quilometragem2
0	Toyota	Branco	150043	4	R$ 24,000.00	300086
1	Honda	Vermelho	87899	4	R$ 25,000.00	175798
2	Toyota	Azul	32549	3	R$ 27,000.00	65098
3	BMW	Preto	11179	5	R$ 122,000.00	22358
4	Nissan	Branco	213095	4	R$ 13,500.00	426190
5	Toyota	Verde	99213	4	R$ 14,500.00	198426
6	Honda	Azul	45698	4	R$ 17,500.00	91396
7	Honda	Azul	54738	4	R$ 27,000.00	109476
8	Toyota	Branco	60000	4	R$ 26,250.00	120000
9	Nissan	Branco	31600	4	R$ 19,700.00	63200

10 Seleção por índices

O **.iloc[]** retorna todas as observações indexadas na posição (posição começa do 0) que escolher, ex:

```
print(dados)

dados.iloc[1]
```

[33] ✓ 0.0s

```
   Fabricante       Cor  Quilometragem  Portas          Preco
0      Toyota    Branco         150043       4   R$ 24,000.00
1       Honda  Vermelho          87899       4   R$ 25,000.00
2      Toyota      Azul          32549       3   R$ 27,000.00
3         BMW     Preto          11179       5  R$ 122,000.00
4      Nissan    Branco         213095       4   R$ 13,500.00
5      Toyota     Verde          99213       4   R$ 14,500.00
6       Honda      Azul          45698       4   R$ 17,500.00
7       Honda      Azul          54738       4   R$ 27,000.00
8      Toyota    Branco          60000       4   R$ 26,250.00
9      Nissan    Branco          31600       4   R$ 19,700.00

Fabricante              Honda
Cor                   Vermelho
Quilometragem            87899
Portas                       4
Preco            R$ 25,000.00
Name: 1, dtype: object
```

```
print(dados)

dados.iloc[4:6]
```

[70] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco	Quilometragem2
0	Toyota	Branco	150043	4	R$ 24,000.00	300086
1	Honda	Vermelho	87899	4	R$ 25,000.00	175798
2	Toyota	Azul	32549	3	R$ 27,000.00	65098
3	BMW	Preto	11179	5	R$ 122,000.00	22358
4	Nissan	Branco	213095	4	R$ 13,500.00	426190
5	Toyota	Verde	99213	4	R$ 14,500.00	198426
6	Honda	Azul	45698	4	R$ 17,500.00	91396
7	Honda	Azul	54738	4	R$ 27,000.00	109476
8	Toyota	Branco	60000	4	R$ 26,250.00	120000
9	Nissan	Branco	31600	4	R$ 19,700.00	63200

	Fabricante	Cor	Quilometragem	Portas	Preco	Quilometragem2
4	Nissan	Branco	213095	4	R$ 13,500.00	426190
5	Toyota	Verde	99213	4	R$ 14,500.00	198426

Também é possível acessar linhas e colunas, colocando uma vírgula:

```
print(dados)

dados.iloc[4:6, 2]
```

[71] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco	Quilometragem2
0	Toyota	Branco	150043	4	R$ 24,000.00	300086
1	Honda	Vermelho	87899	4	R$ 25,000.00	175798
2	Toyota	Azul	32549	3	R$ 27,000.00	65098
3	BMW	Preto	11179	5	R$ 122,000.00	22358
4	Nissan	Branco	213095	4	R$ 13,500.00	426190
5	Toyota	Verde	99213	4	R$ 14,500.00	198426
6	Honda	Azul	45698	4	R$ 17,500.00	91396
7	Honda	Azul	54738	4	R$ 27,000.00	109476
8	Toyota	Branco	60000	4	R$ 26,250.00	120000
9	Nissan	Branco	31600	4	R$ 19,700.00	63200

```
4    213095
5     99213
Name: Quilometragem, dtype: int64
```

O **.loc[]** faz o index acessa os rótulos dos index, se eu acessar o rótulo 4 do index:

```
print(dados)

dados.loc[4]
```

[35] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco
0	Toyota	Branco	150043	4	R$ 24,000.00
1	Honda	Vermelho	87899	4	R$ 25,000.00
2	Toyota	Azul	32549	3	R$ 27,000.00
3	BMW	Preto	11179	5	R$ 122,000.00
4	Nissan	Branco	213095	4	R$ 13,500.00
5	Toyota	Verde	99213	4	R$ 14,500.00
6	Honda	Azul	45698	4	R$ 17,500.00
7	Honda	Azul	54738	4	R$ 27,000.00
8	Toyota	Branco	60000	4	R$ 26,250.00
9	Nissan	Branco	31600	4	R$ 19,700.00

```
Fabricante             Nissan
Cor                    Branco
Quilometragem          213095
Portas                      4
Preco           R$ 13,500.00
Name: 4, dtype: object
```

```
print(dados)

dados.loc[4:6]
```

[69] ✓ 0.0s

```
  Fabricante       Cor  Quilometragem  Portas          Preco  Quilometragem2
0     Toyota    Branco         150043       4   R$ 24,000.00          300086
1      Honda  Vermelho          87899       4   R$ 25,000.00          175798
2     Toyota      Azul          32549       3   R$ 27,000.00           65098
3        BMW     Preto          11179       5  R$ 122,000.00           22358
4     Nissan    Branco         213095       4   R$ 13,500.00          426190
5     Toyota     Verde          99213       4   R$ 14,500.00          198426
6      Honda      Azul          45698       4   R$ 17,500.00           91396
7      Honda      Azul          54738       4   R$ 27,000.00          109476
8     Toyota    Branco          60000       4   R$ 26,250.00          120000
9     Nissan    Branco          31600       4   R$ 19,700.00           63200
```

	Fabricante	Cor	Quilometragem	Portas	Preco	Quilometragem2
4	Nissan	Branco	213095	4	R$ 13,500.00	426190
5	Toyota	Verde	99213	4	R$ 14,500.00	198426
6	Honda	Azul	45698	4	R$ 17,500.00	91396

Também é possível acessar linhas e colunas, colocando uma vírgula:

```
print(dados)

dados.loc[4:6, 'Quilometragem']
```

[75] ✓ 0.0s

```
  Fabricante       Cor  Quilometragem  Portas          Preco
0     Toyota    Branco         150043       4   R$ 24,000.00
1      Honda  Vermelho          87899       4   R$ 25,000.00
2     Toyota      Azul          32549       3   R$ 27,000.00
3        BMW     Preto          11179       5  R$ 122,000.00
4     Nissan    Branco         213095       4   R$ 13,500.00
5     Toyota     Verde          99213       4   R$ 14,500.00
6      Honda      Azul          45698       4   R$ 17,500.00
7      Honda      Azul          54738       4   R$ 27,000.00
8     Toyota    Branco          60000       4   R$ 26,250.00
9     Nissan    Branco          31600       4   R$ 19,700.00

4    213095
5     99213
6     45698
Name: Quilometragem, dtype: int64
```

Seleção de múltiplas colunas:

```
dados[['Fabricante', 'Cor', 'Portas']]
```

[85] ✓ 0.0s

	Fabricante	Cor	Portas
0	Toyota	Azul	4
1	Honda	Azul	4
2	Toyota	Azul	3
3	BMW	Azul	5
4	Nissan	Azul	4
5	Toyota	Azul	4
6	Honda	Azul	4
7	Honda	Azul	4
8	Toyota	Azul	4
9	Nissan	Azul	4

Para deletar uma coluna use o **del**, veja o ex:

```
del dados['Quilometragem2']
dados
```

[97] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco
0	Toyota	Branco	150043	4	R$ 24,000.00
1	Honda	Vermelho	87899	4	R$ 25,000.00
2	Toyota	Azul	32549	3	R$ 27,000.00
3	BMW	Preto	11179	5	R$ 122,000.00
4	Nissan	Branco	213095	4	R$ 13,500.00
5	Toyota	Verde	99213	4	R$ 14,500.00

 # 14, 15, 16, 17 Filtragem de dados

O **.unique()** retorna os dados únicos de uma coluna:

```
dados['Fabricante'].unique()
```

[98] ✓ 0.0s

```
array(['Toyota', 'Honda', 'BMW', 'Nissan'], dtype=object)
```

Para filtrar por condições, utilizamos **.query()** e no final do filtro usamos **.reset_index()** para os dados filtrados voltarem o index começando do 0.

```
dados_filtrados = dados.query('Fabricante == "Honda"').reset_index()
dados_filtrados
```

[106] ✓ 0.0s

	index	Fabricante	Cor	Quilometragem	Portas	Preco
0	1	Honda	Vermelho	87899	4	R$ 25,000.00
1	6	Honda	Azul	45698	4	R$ 17,500.00
2	7	Honda	Azul	54738	4	R$ 27,000.00

```
dados_filtrados2 = dados.query('Fabricante == "Honda" and Cor == "Azul"').reset_index()
dados_filtrados2
```

[107] ✓ 0.0s

	index	Fabricante	Cor	Quilometragem	Portas	Preco
0	6	Honda	Azul	45698	4	R$ 17,500.00
1	7	Honda	Azul	54738	4	R$ 27,000.00

Outra forma de filtrar

```
dados_filtrados3 = dados[dados['Fabricante'] == "Honda"]
dados_filtrados3
```

[111] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco
1	Honda	Vermelho	87899	4	R$ 25,000.00
6	Honda	Azul	45698	4	R$ 17,500.00
7	Honda	Azul	54738	4	R$ 27,000.00

```
dados_filtrados4 = dados[(dados['Fabricante'] == "Honda") & (dados['Cor'] == "Azul")]
dados_filtrados4
```

[112] ✓ 0.0s

	Fabricante	Cor	Quilometragem	Portas	Preco
6	Honda	Azul	45698	4	R$ 17,500.00
7	Honda	Azul	54738	4	R$ 27,000.00

Filtrando a partir de uma lista:

```
lista_cor = ['Azul', 'Vermelho']
dados_filtrados5 = dados.query('Cor in @lista_cor')
dados_filtrados5
```

[116] ✓ 0.0s

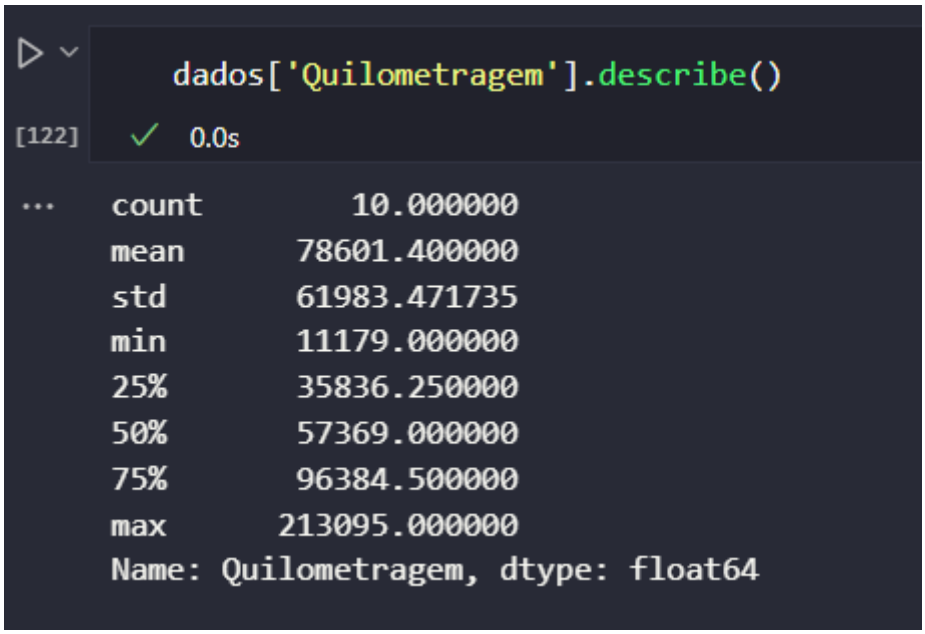
	Fabricante	Cor	Quilometragem	Portas	Preco
1	Honda	Vermelho	87899	4	R$ 25,000.00
2	Toyota	Azul	32549	3	R$ 27,000.00
6	Honda	Azul	45698	4	R$ 17,500.00
7	Honda	Azul	54738	4	R$ 27,000.00

O **.dropna()** remove todas as linhas do DataFrame que são vazias.

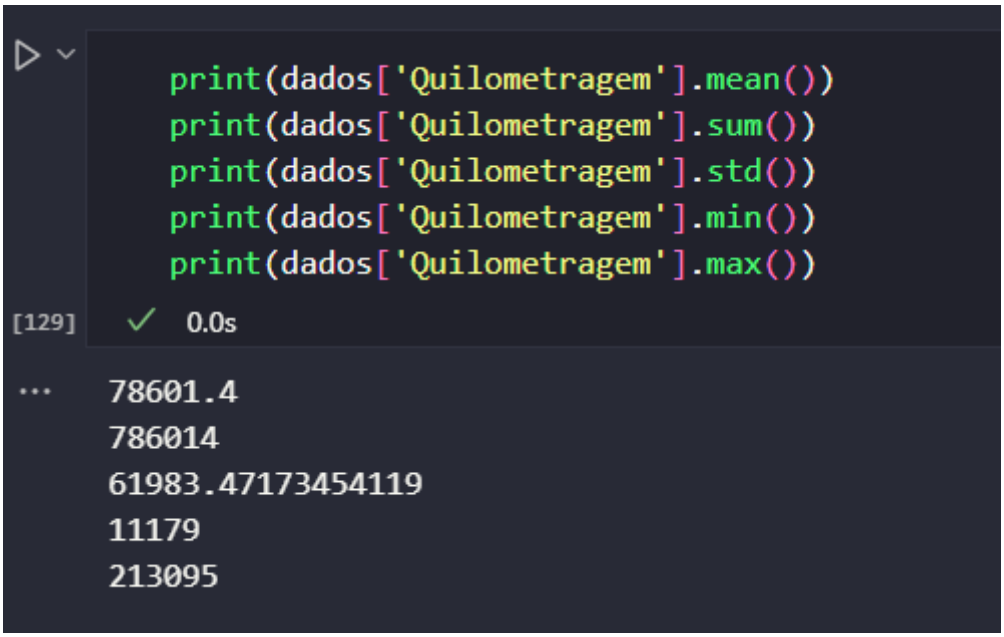
O **.fillna()** preenche todos valores vazios de um DataFrame com 0.

#21 Estatística descritiva

O **.describle()** nos retorna a estatística descritiva como contagem, média, etc.:



Para calcular a média, soma, desvio padrão, min, max:



Para contagem usamos **.value_counts()**:

```
dados['Fabricante'].value_counts()
```

[133] ✓ 0.0s

```
Fabricante
Toyota    4
Honda     3
Nissan    2
BMW       1
Name: count, dtype: int64
```

Podemos também criar um novo DataFrame usando **.to_frame()**:

```
dados_fabricante = dados['Fabricante'].value_counts().to_frame().reset_index()
dados_fabricante
```

[137] ✓ 0.0s

	Fabricante	count
0	Toyota	4
1	Honda	3
2	Nissan	2
3	BMW	1