02 - Pandas (Pt1)

Docupedia Export

Author:Lima Queila (CtP/ETS)
Date:27-Jan-2026 17:03

Table of Contents

1 TREINAMENTO DE PYTHON (PANDAS)

Pandas (https://pandas.pydata.org) é uma biblioteca Python utilizada para manipulação e análise de dados. Oferece estruturas de dados rápidas, flexíveis e expressivas, projetadas para tornar o trabalho com dados “relacionais” ou “rotulados” fáceis e intuitivos.
As duas principais estruturas de dados dos pandas, Series (unidimensional, ou seja, uma só coluna) e DataFrame (multi-dimensional, feito com uma coleção de Series), lidam com a grande maioria dos casos de uso típico em finanças, estatística, ciências sociais e muitas áreas de engenharia.

```
import pandas as pd
```

1.1 Vamos começar lendo um CSV (Comma-separated values) com o pandas

```
titanic = pd.read_csv("data/titanic.csv")
```

Apesar de CSV significar "valores separados por vírgulas", algumas vezes os dados podem vir separados por outros caracteres, por exemplo ";".
Por isso podemos usar o argumento "sep" dentro da nossa função.

```
titanic = pd.read_csv("data/titanic.csv", sep=",")
```

Para dar uma olhada nas primeiras linhas do dataset:

```
titanic.head() # mostra as 5 primeiras linhas
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

```
titanic.tail() # mostra as 5 últimas linhas
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

```
print(type(titanic))
# pandas.core.frame.DataFrame

# DataFrame é uma classe do pandas que representa os dados de forma tabular, parecendo muito com uma tabela do Excel.
```

5 linhas podem não ser o suficiente

```
titanic.head(10)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C

1.2 O que significa cada coluna?

Nosso dataset é baseado em um dado público, tem bastante informações sobre e sua descrição no Kaggle
https://www.kaggle.com/c/titanic/data

- survived: Sobreviventes
    - (0 = Não; 1 = Sim)
- pclass: Classe do passageiro
    - (1 = Primeira; 2 = Segunda; 3 = Terceira)
- name: Nome
- sex: Sexo
- age: Idade
- sibsp: Número de irmão/conjuges a bordo
- parch: Número de filhos/pais a bordo
- ticket: Número do ticket
- fare: Preço da Passagem
- cabin: Cabine
- embarked: Porto de embarque
    - [C = Cherbourg (França); Q = Queenstown (Irlanda); S = Southampton (Inglaterra)]

1.3 Indexação

Podemos indexar nosso Dataframe para vermos os dados de somente uma coluna.

```
titanic["Name"].head() # parece um dicionário de listas!
```

```
0                              Braund, Mr. Owen Harris
1    Cumings, Mrs. John Bradley (Florence Briggs Th...
2                               Heikkinen, Miss. Laina
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)
4                             Allen, Mr. William Henry
Name: Name, dtype: object
```

Podemos acessar pelo ponto também!

```
titanic.Name.head()
```

```
0                              Braund, Mr. Owen Harris
1    Cumings, Mrs. John Bradley (Florence Briggs Th...
2                               Heikkinen, Miss. Laina
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)
4                             Allen, Mr. William Henry
Name: Name, dtype: object
```

1.4 Informações sobre o dataset

Dimensão dos dados:

```
titanic.shape
# (892, 12)
```

Ao importar o CSV, o Dataframe inclui uma coluna index, que não está no arquivo. Essa coluna é tipo um índice de linha do Excel.

```
titanic.index
# RangeIndex(start=0, stop=891, step=1)
```

Quais são as colunas do Dataframe?

```
titanic.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

E os formatos dos dados?

```
titanic.dtypes
```

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

E se quisermos só as colunas de um certo tipo?

```
filtro_tipo_numerico = (titanic.dtypes == "object")
print(filtro_tipo_numerico)
```

```
PassengerId    False
Survived       False
Pclass         False
Name            True
Sex             True
Age            False
SibSp          False
Parch          False
Ticket          True
Fare           False
Cabin           True
Embarked        True
dtype: bool
```

```
colunas_numericas = titanic.dtypes[filtro_tipo_numerico].index
print(colunas_numericas)
# Index(['PassengerId', 'Survived', 'Pclass', 'SibSp', 'Parch'], dtype='object')
```

```
titanic[colunas_numericas].head()
```

	PassengerId	Survived	Pclass	SibSp	Parch
0	1	0	3	1	0
1	2	1	1	1	0
2	3	1	3	0	0
3	4	1	1	1	0
4	5	0	3	0	0

Posso limitar as colunas que quero selecionar:

```
colunas_selecionadas = ["PassengerId","SibSp","Parch"]
titanic[colunas_selecionadas].head()
```

	PassengerId	SibSp	Parch
0	1	1	0
1	2	1	0
2	3	0	0
3	4	1	0
4	5	0	0

1.5 DESAFIO

Como retornamos um DataFrame com todas as colunas menos a última?

Resultado

```
todas_colunas_menos_ultima = titanic.columns[:-1]
print(todas_colunas_menos_ultima)
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin'],
      dtype='object')
```

```
titanic[todas_colunas_menos_ultima].head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN

1.5.1 OU

```
titanic.iloc[:,:-1].head() # iloc seleciona os índices por linha e coluna
titanic.iloc[15:, :3].head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN

```
titanic.loc[:,["Survived"].head() # loc seleciona pelo nome das colunas
```

```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

1.6 Informações estatísticas do dataset

```
titanic.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
titanic.Age.describe()
```

```
count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
75%       38.000000
max       80.000000
Name: Age, dtype: float64
```

1.6.1 E se quisermos ver tudo junto (Obs: Fica bagunçado)?

```
titanic.describe(include='all')
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
count	891.000000	891.000000	891.000000	891	891	714.000000	891.000000	891.000000	891	891.000000	204	889
unique	NaN	NaN	NaN	891	2	NaN	NaN	NaN	681	NaN	147	3
top	NaN	NaN	NaN	Herman, Mrs. Samuel (Jane Laver)	male	NaN	NaN	NaN	1601	NaN	B96 B98	S
freq	NaN	NaN	NaN	1	577	NaN	NaN	NaN	7	NaN	4	644
mean	446.000000	0.383838	2.308642	NaN	NaN	29.699118	0.523008	0.381594	NaN	32.204208	NaN	NaN

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
std	257.353842	0.486592	0.836071	NaN	NaN	14.526497	1.102743	0.806057	NaN	49.693429	NaN	NaN
min	1.000000	0.000000	1.000000	NaN	NaN	0.420000	0.000000	0.000000	NaN	0.000000	NaN	NaN
25%	223.500000	0.000000	2.000000	NaN	NaN	20.125000	0.000000	0.000000	NaN	7.910400	NaN	NaN
50%	446.000000	0.000000	3.000000	NaN	NaN	28.000000	0.000000	0.000000	NaN	14.454200	NaN	NaN
75%	668.500000	1.000000	3.000000	NaN	NaN	38.000000	1.000000	0.000000	NaN	31.000000	NaN	NaN
max	891.000000	1.000000	3.000000	NaN	NaN	80.000000	8.000000	6.000000	NaN	512.329200	NaN	NaN

1.7 DESAFIO

E se quisermos um Dataframe com a descrição apenas das colunas AGE e PCLASS?

Resultado

```
colunas = ['Age','Pclass']
titanic[['Age','Pclass']].describe()
```

	Age	Pclass
count	714.000000	891.000000
mean	29.699118	2.308642
std	14.526497	0.836071
min	0.420000	1.000000
25%	20.125000	2.000000
50%	28.000000	3.000000
75%	38.000000	3.000000
max	80.000000	3.000000

Se eu quiser apenas um valor entre as informações estatísticas?

```
titanic['Age'].mean() # média
# 29.69911764705882
```

```
titanic.Age.std() # desvio padrão
.count()
# 14.526497332334044
```

1.8 DESAFIO

Importe as colunas **'crim'** e **'medv'** do dataset **BostonHousing**.
Mostre somente as primeiras 14 linhas.

OUTPUT

	crim	medv
0	0.00632	24.0
1	0.02731	21.6
2	0.02729	34.7
3	0.03237	33.4
4	0.06905	36.2
5	0.02985	28.7
6	0.08829	22.9
7	0.14455	27.1
8	0.21124	16.5
9	0.17004	18.9
10	0.22489	15.0
11	0.11747	18.9
12	0.09378	21.7
13	0.62976	20.4

Resultado

```
import pandas as pd
df= pd.read_csv("data/BostonHousing.csv", sep=",").loc[:13,["crim", "medv"]]
print(df)
```

	crim	medv
0	0.00632	24.0
1	0.02731	21.6
2	0.02729	34.7
3	0.03237	33.4
4	0.06905	36.2
5	0.02985	28.7
6	0.08829	22.9
7	0.14455	27.1
8	0.21124	16.5
9	0.17004	18.9
10	0.22489	15.0
11	0.11747	18.9
12	0.09378	21.7
13	0.62976	20.4

```
titanic["Embarked"].unique() #Podemos verificar todos os valores únicos
```

array(['S', 'C', 'Q', nan], dtype=object)

```
titanic["Age"].value_counts() #Contagem de quantas vezes eles aparecem, normalize
```

24.00 30
22.00 27
18.00 26
19.00 25
30.00 25
..
55.50 1

```
70.50    1
66.00    1
23.50    1
0.42     1
Name: Age, Length: 88, dtype: int64
```

```
#print(titanic.isna().head(15))
x=titanic.dropna() #excluir todas as linhas com not a number
print(x.isna().head(15))
```

```
     PassengerId  Survived  Pclass   Name    Sex    Age  SibSp  Parch  Ticket  \
1          False     False   False  False  False  False  False  False   False
3          False     False   False  False  False  False  False  False   False
6          False     False   False  False  False  False  False  False   False
10         False     False   False  False  False  False  False  False   False
11         False     False   False  False  False  False  False  False   False
21         False     False   False  False  False  False  False  False   False
23         False     False   False  False  False  False  False  False   False
27         False     False   False  False  False  False  False  False   False
52         False     False   False  False  False  False  False  False   False
54         False     False   False  False  False  False  False  False   False
62         False     False   False  False  False  False  False  False   False
66         False     False   False  False  False  False  False  False   False
75         False     False   False  False  False  False  False  False   False
88         False     False   False  False  False  False  False  False   False
92         False     False   False  False  False  False  False  False   False
```

	Fare	Cabin	Embarked
1	False	False	False
3	False	False	False
6	False	False	False
10	False	False	False
11	False	False	False
21	False	False	False
23	False	False	False
27	False	False	False
52	False	False	False
54	False	False	False
62	False	False	False
66	False	False	False
75	False	False	False
88	False	False	False
92	False	False	False

```
y=titanic["Age"].fillna(titanic["Age"].mean())#Preencheer onde é encontrado not a number
print(y.head(15))
z=titanic.fillna(200)
```

```
0     22.000000
1     38.000000
2     26.000000
3     35.000000
4     35.000000
5     29.699118
6     54.000000
7      2.000000
8     27.000000
9     14.000000
10     4.000000
11    58.000000
12    20.000000
13    39.000000
14    14.000000
Name: Age, dtype: float64
```

Podemos também agrupar dados baseados em determinados critérios, utilizando o método .groupby().
Esse método pode ser usado para resolver os mais amplos dos problemas.

```
print(round(titanic.groupby("Embarked").mean()["Age"].sort_values()["S"], 5))
#titanic.sort_values(by="Age")
```

29.4454

1.9 DESAFIO

Uma pessoa quer comprar um carro, e deseja ver as opções mais econômicas. Mostre os dados em um Dataframe com as seguintes condições:

- Deve ser um carro com 5 lugares; (Passengers)
- Selecione os 10 carros com maior MPG(Miles Per Gallon) na cidade;
- Dos 10 mais econômicos, mostre os 5 modelos mais baratos; (Price)
- Mostre somente as colunas 'Manufacturer','Make','Price','MPG.city','Type','Passengers'

Tempo estimado: 50 min.

Resultado

```
import pandas as pd
ap=pd.read_csv("data/Cars93_miss.csv", sep=",").loc[:, ["Manufacturer", "Make", "Price", "MPG.city", "Type", "Passengers"]].dropna()
ap=ap[ap["Passengers"]==5]
ap=ap.sort_values(by="MPG.city", ascending=False).head(10)
ap=ap.sort_values(by="Price").head(5)
print(ap.head())
```

	Manufacturer	Make	Price	MPG.city	Type	Passengers
43	Hyundai	Hyundai Excel	8.0	29.0	Small	5.0
22	Dodge	Dodge Colt	9.2	29.0	Small	5.0
31	Ford	Ford Escort	10.1	23.0	Small	5.0
61	Mitsubishi	Mitsubishi Mirage	10.3	29.0	Small	5.0
80	Subaru	Subaru Loyale	10.9	25.0	Small	5.0

1.10 Filtrando o DataFrame

O Filtro sempre é um conjunto de booleanos informando se o índice foi incluído ou não no filtro
Filtrando as linhas onde Pclass == 1 (os mais ricos)

```
filtro_apenas_primeira_classe = titanic["Pclass"] == 1
filtro_apenas_primeira_classe.head()
```

0 False
1 True
2 False
3 True
4 False
Name: Pclass, dtype: bool

```
filtro_apenas_primeira_classe = titanic["Pclass"] == 1
titanic[ filtro_apenas_primeira_classe ].head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C103	S
23	24	1	1	Sloper, Mr. William Thompson	male	28.0	0	0	113788	35.5000	A6	S

O resultado do filtro é outro dataframe, então podemos atribuir o resultado do filtro em uma variável e trabalhar com ela sem medo de alterar o dataframe original.

Podemos fazer o mesmo para pegar os da terceira classe.

```
terceira_classe = titanic[titanic["Pclass"] == 3]
type(terceira_classe)
```

pandas.core.frame.DataFrame

```
# fazer um filtro por sexo e aplicá-lo!
filtro_apenas_mulheres = titanic["Sex"] != "male"
mulheres_titanic = titanic[filtro_apenas_mulheres]
mulheres_titanic["Age"].describe()
```

count 261.000000
mean 27.915709
std 14.110146
min 0.750000
25% 18.000000
50% 27.000000
75% 37.000000
max 63.000000
Name: Age, dtype: float64

1.11 Filtrando com mais de uma condição

Pessoas que são da primeira classe e sobreviveram

```
# AND
# Filtrando as linhas onde Pclass == 1 E Survived == 1


rico = titanic["Pclass"] == 1
```

```
sobreviveu = titanic["Survived"] == 1

#print(titanic[ rico & sobreviveu ].head()) # tem que usar o bitwise ('and' não funciona)
titanic[ rico & sobreviveu ].describe()
# O mesmo filtro, em uma linha apenas
# titanic[ (titanic["Pclass"] == 1) & (titanic["Survived"] == 1) ].head()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	136.000000	136.0	136.0	122.000000	136.000000	136.000000	136.000000
mean	491.772059	1.0	1.0	35.368197	0.492647	0.389706	95.608029
std	239.006988	0.0	0.0	13.760017	0.632412	0.690387	85.286820
min	2.000000	1.0	1.0	0.920000	0.000000	0.000000	25.929200
25%	307.750000	1.0	1.0	24.250000	0.000000	0.000000	50.985450
50%	510.500000	1.0	1.0	35.000000	0.000000	0.000000	77.958300
75%	693.500000	1.0	1.0	45.000000	1.000000	1.000000	111.481225
max	890.000000	1.0	1.0	80.000000	3.000000	2.000000	512.329200

Para filtrar pessoas que são da primeira ou segunda classe

```
# OR
# Filtrando as linhas onde Pclass==1 OU Pclass == 2

titanic[(titanic["Pclass"] == 1) | (titanic["Pclass"] == 2)].head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C103	S

Informações estatísticas do Dataframe com o filtro

```
titanic[((titanic["Pclass"] == 1) & (titanic["Survived"] == 1)) |
        ((titanic["Pclass"] == 3) & (titanic["Survived"]==1))]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	G6	S
...	...	...	...	...	...	...	...	...	...	...	...	...
871	872	1	1	Beckwith, Mrs. Richard Leonard (Sallie Monypeny)	female	47.0	1	1	11751	52.5542	D35	S
875	876	1	3	Najib, Miss. Adele Kiamie "Jane"	female	15.0	0	0	2667	7.2250	NaN	C
879	880	1	1	Potter, Mrs. Thomas Jr (Lily Alexenia Wilson)	female	56.0	0	1	11767	83.1583	C50	C
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C

Comparando a diferença entre a Idade antes e depois do filtro

```
media = titanic.Age.mean()

media_filtro = titanic[(titanic["Pclass"] == 1) & (titanic["Survived"] == 1)].Age.mean()
print("Média de idade: {}\nMédia de idade dos sobreviventes da primeira classe: {}""".format(media, media_filtro))
```

Média de idade: 29.69911764705882
Média de idade dos sobreviventes da primeira classe: 35.36819672131148

1.12 E se quisermos um filtro que retorne os registros com os passageiros da primeira classe que sobreviveram
1.13 e os passageiros da terceira classe que não sobreviveram?

Tempo estimado: 10 min

Resultado

```
filtro = (
    (titanic["Pclass"] == 1) & (titanic["Survived"] == 1) |
    (titanic["Pclass"] == 3) & (titanic["Survived"] == 0)
)
titanic[filtro].head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q

1.14 DESAFIO

Robin está procurando por uma mulher que conheceu durante um evento no Titanic, poucas horas antes da tragédia.

Ele gostaria de saber se ela sobreviveu e nós podemos ajudar ele através do nosso banco de dados.

As informações que ele sabe sobre ela são:

- Ela embarcou em Southampton (Inglaterra)
- Ela era da segunda classe
- Ela tinha 29.0 anos
- E no nome completo dela tinha Anne, mas ele não sabe se era nome ou sobrenome.
- Lembre sempre de retirar os valores NaN das colunas (aplique outro valor no lugar deles)

Tempo estimado: 60 min

```
A mulher que ele está procurando é Faunthorpe, Mrs. Lizzie (Elizabeth Anne
Wilkinson), ela sobreviveu
```

Resultado

```python
import pandas as pd
titanic = pd.read_csv("C:/Users/veh_o/OneDrive/Documentos/Mod2/arquivos/Pandas/titanic.csv", sep=",")
# print(titanic)
media=titanic["Age"].mean()
titanic["Age"]=titanic["Age"].fillna(media)

titanic["Cabin"]=titanic["Cabin"].fillna("Desconhecido")
filtro_embarque= (titanic["Embarked"]=="S") & (titanic["Pclass"] == 2)
filtro_sex=titanic["Sex"]=="female"
titanic=titanic[filtro_embarque & filtro_sex]




print(titanic.index)

lista=[]
for i in titanic.index:
    print(i)
    titanic.loc[i,"Age"]=round(titanic.loc[i,"Age"])
    if "Anne" in titanic.loc[i,"Name"]:
        lista.append(titanic.loc[i,"Name"])
```

```
#lista2=[]
for i in titanic.index:
    for j in range(len(lista)):
        if titanic.loc[i,"Name"]=="lista[j]:
            lista2.append([titanic.loc[i,"Age"], titanic.loc[i,"Name"], titanic.loc[i,"Survived"]])
for k in range(len(lista2)):
    if lista2[k][0]==29.0:
        if lista2[k][2]==1:
            print("A mulher que ele está procurando é {}, ela sobreviveu".format(lista2[k][1]))
        else:
             print("A mulher que ele está procurando é {}, ela nao sobreviveu".format(lista2[k][1]))
```

1.15 Transformação dos dados

Supondo que queremos criar uma coluna onde o valor é a soma entre o **sibsp** e **parch**
Precisamos definir uma função

```
import pandas as pd
titanic = pd.read_csv("data/titanic.csv", sep=",")
def soma_sibsp_parch(linha, reg):
    return linha["SibSp"] + linha["Parch"] # ou seja, todos os parentes dentro do navio
```

Depois Aplicar a função linha a linha no DataFrame.
É como se fosse um `for` automático!

```
nova_coluna = titanic.apply(soma_sibsp_parch, axis=1)
```

axis = 1 representa que a função é aplicada linha a linha.
axis = 0 é o valor padrão, representa que a função é aplicada coluna a coluna.

Vamos ver o que ele retornou:

```
nova_coluna.head()
```

```
0  1
1  1
2  0
3  1
4  0
dtype: int64
```

```
type(nova_coluna)
```

```
pandas.core.series.Series
```

1.15.1 Como incluo essa coluna no meu DataFrame?

É muito igual a um Dicionário do Python (dict)

```
titanic["Relatives"] = nova_coluna
```

Vamos verificar que a coluna foi criada!

```
titanic.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked', 'Relatives'], dtype='object')
```

```
titanic.head()
print(titanic["SibSp"])
```

```
0    1
1    1
2    0
3    1
4    0
    ..
886  0
887  0
888  1
889  0
890  0
Name: SibSp, Length: 891, dtype: int64
```

E se eu quiser deletar essa coluna?

```
titanic.pop('Relatives')
titanic.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

1.15.2 E agora como faço para ter esse novo DataFrame salvo em csv?

```
titanic["Relatives"] = nova_coluna
```

```
titanic.to_csv("data/titanic_1_aula.csv", index=False)
```

Pode salvar em vários formatos!!

```
#titanic.to_ # use o tab
```

2

Chega de pandas por hoje