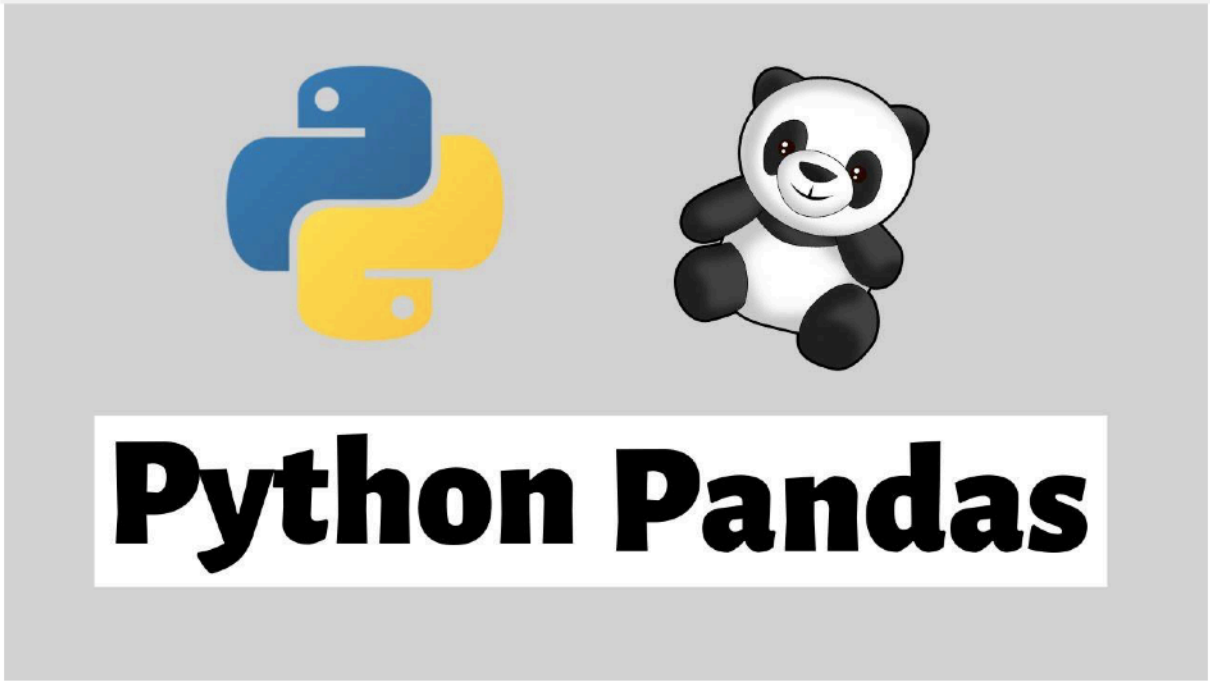
Python Pandas from Basics to Advance




```
df = pd.DataFrame({

 "Name": ["Braund, Mr. Owen Harris","Allen, Mr. William Henry","Bonnell, Miss. Elizabeth"],

 "Age": [22, 35, 58],

 "Sex": ["male", "male", "female"]

})

df
```

	Name	Age	Sex
0	Braund, Mr. Owen Harris	22	male
1	Allen, Mr. William Henry	35	male
2	Bonnell, Miss. Elizabeth	58	female

Ehsan Paydar
EPSOFT · he/him

EPSOFT
Everywhere
epsoft98@gmail.com
www.epsoft.ir
@epsoft98
in/ehsanpaydar
epsoft
epsoft98

Pandas toolkit Part 1

Syed Afroz Ali

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
df = pd.DataFrame({

 "Name": ["Braund, Mr. Owen Harris","Allen, Mr. William Henry","Bonnell, Miss.

 "Age": [22, 35, 58],

 "Sex": ["male", "male", "female"]

})

df
```

Out[2]:

	Name	Age	Sex
0	Braund, Mr. Owen Harris	22	male
1	Allen, Mr. William Henry	35	male
2	Bonnell, Miss. Elizabeth	58	female

In [3]:
```
df["Age"]
```

Out[3]:
```
0    22
1    35
2    58
Name: Age, dtype: int64
```

In [4]:
```
ages = pd.Series([22, 35, 58], name="Age")
ages
```

Out[4]:
```
0    22
1    35
2    58
Name: Age, dtype: int64
```

In [5]:
```
df["Age"].max()
```

Out[5]:
```
58
```

In [6]:
```
ages.max()
```

Out[6]:
```
58
```

In [7]:
```
df.describe()
```

Out[7]:

	Age
count	3.000000
mean	38.333333
std	18.230012
min	22.000000
25%	28.500000
50%	35.000000
75%	46.500000
max	58.000000

In [8]:
```
titanic = pd.read_csv("train_titanic.csv")
titanic.head()
```

Out[8]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cal
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	N
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	N
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	N

In [9]:
```
titanic.dtypes
```

Out[9]:
```
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
```

In [10]:
```
titanic.to_excel("titanic.xlsx", sheet_name="passengers", index=False)
```

In [11]:
```
titanic = pd.read_excel("titanic.xlsx", sheet_name="passengers")
```

In [12]:
```
titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [13]:
```
ages = titanic["Age"]
ages.head()
```

Out[13]:
```
0    22.0
1    38.0
2    26.0
3    35.0
4    35.0
Name: Age, dtype: float64
```

In [14]:
```
type(titanic["Age"])
```

Out[14]:
```
pandas.core.series.Series
```

In [15]:
```
titanic["Age"].shape
```

Out[15]:
```
(891,)
```

In [16]:
```
titanic["Age"].shape
```

Out[16]:
```
(891,)
```

In [17]:
```
age_sex = titanic[["Age", "Sex"]]
age_sex.head()
```

Out[17]:

	Age	Sex
0	22.0	male
1	38.0	female
2	26.0	female
3	35.0	female
4	35.0	male

In [18]:
```
titanic[["Age", "Sex"]].shape
```

Out[18]:
```
(891, 2)
```

In [19]:
```
above_35 = titanic[titanic["Age"] > 35]
above_35.head()
```

Out[19]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cab
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C8
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E4
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C10
13	14	0	3	Andersson, Mr. Anders Johan	male	39.0	1	5	347082	31.2750	Na
15	16	1	2	Hewlett, Mrs. (Mary D Kingcome)	female	55.0	0	0	248706	16.0000	Na

In [20]:
```
class_23 = titanic[titanic["Pclass"].isin([2, 3])]
class_23.head()
```

Out[20]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cal
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	N
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	N
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	N
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	N
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	N

In [21]:
```
class_23 = titanic[(titanic["Pclass"] == 2) | (titanic["Pclass"] == 3)]
class_23.head()
```

Out[21]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cal
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	N
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	N
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	N
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	N
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	N

In [22]:
```
age_no_na = titanic[titanic["Age"].notna()]
age_no_na.head()
```

Out[22]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cal
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	N
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	N
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	N

In [23]:
```
adult_names = titanic.loc[titanic["Age"] > 35]
adult_names.head()
```

Out[23]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cab
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C8
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E4
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C10
13	14	0	3	Andersson, Mr. Anders Johan	male	39.0	1	5	347082	31.2750	Na
15	16	1	2	Hewlett, Mrs. (Mary D Kingcome)	female	55.0	0	0	248706	16.0000	Na

In [24]:
```
adult_names = titanic.loc[titanic["Age"] > 35, "Name"]
adult_names.head()
```

Out[24]:
```
1     Cumings, Mrs. John Bradley (Florence Briggs Th...
6                               McCarthy, Mr. Timothy J
11                             Bonnell, Miss. Elizabeth
13                          Andersson, Mr. Anders Johan
15                    Hewlett, Mrs. (Mary D Kingcome)
Name: Name, dtype: object
```

In [25]:
```
titanic.iloc[9:25, 2:5]
```

Out[25]:

	Pclass	Name	Sex
9	2	Nasser, Mrs. Nicholas (Adele Achem)	female
10	3	Sandstrom, Miss. Marguerite Rut	female
11	1	Bonnell, Miss. Elizabeth	female
12	3	Saundercock, Mr. William Henry	male
13	3	Andersson, Mr. Anders Johan	male
14	3	Vestrom, Miss. Hulda Amanda Adolfina	female
15	2	Hewlett, Mrs. (Mary D Kingcome)	female
16	3	Rice, Master. Eugene	male
17	2	Williams, Mr. Charles Eugene	male
18	3	Vander Planke, Mrs. Julius (Emelia Maria Vande...	female
19	3	Masselmani, Mrs. Fatima	female
20	2	Fynney, Mr. Joseph J	male
21	2	Beesley, Mr. Lawrence	male
22	3	McGowan, Miss. Anna "Annie"	female
23	1	Sloper, Mr. William Thompson	male
24	3	Palsson, Miss. Torborg Danira	female

In [26]:
```
anon = titanic.iloc[0:3, 3] = "anonymous"
anon
```

Out[26]:
```
'anonymous'
```

In [27]:
```
titanic.head()
```

Out[27]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
0	1	0	3	anonymous	male	22.0	1	0	A/5 21171	7.2500	I
1	2	1	1	anonymous	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	anonymous	female	26.0	0	0	STON/O2. 3101282	7.9250	I
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	I

In [28]:
```
titanic["Age"].mean()
```

Out[28]:
```
29.69911764705882
```

In [29]:
```
titanic[["Age", "Fare"]].median()
```

Out[29]:
```
Age     28.0000
Fare    14.4542
dtype: float64
```

In [30]:
```
titanic[["Age", "Fare"]].describe()
```

Out[30]:

	Age	Fare
count	714.000000	891.000000
mean	29.699118	32.204208
std	14.526497	49.693429
min	0.420000	0.000000
25%	20.125000	7.910400
50%	28.000000	14.454200
75%	38.000000	31.000000
max	80.000000	512.329200

In [31]:
```
titanic.agg({

"Age": ["min", "max", "median", "skew"],
"Fare": ["min", "max", "median", "mean"]

})
```

Out[31]:

	Age	Fare
min	0.420000	0.000000
max	80.000000	512.329200
median	28.000000	14.454200
skew	0.389108	NaN
mean	NaN	32.204208

In [32]:
```
titanic[["Sex", "Age"]].groupby("Sex").mean()
```

Out[32]:

	Age
Sex	
female	27.915709
male	30.726645

In [33]:
```
titanic[["Sex", "Age"]].groupby("Sex").max()
```

Out[33]:

	Age
Sex	
female	63.0
male	80.0

In [34]:
```
titanic[["Sex", "Age"]].groupby("Sex").first()
```

Out[34]:

	Age
Sex	
female	38.0
male	22.0

In [35]:
```
titanic.head(2)
```

Out[35]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	anonymous	male	22.0	1	0	A/5 21171	7.2500	NaN
1	2	1	1	anonymous	female	38.0	1	0	PC 17599	71.2833	C85

In [37]:
```
titanic.groupby("Sex")["Age"].mean()
```

Out[37]:
```
Sex
female    27.915709
male      30.726645
Name: Age, dtype: float64
```

In [38]:
```
titanic.groupby(["Sex", "Pclass"])["Fare"].mean()
```

Out[38]:
```
Sex     Pclass
female  1         106.125798
        2          21.970121
        3          16.118810
male    1          67.226127
        2          19.741782
        3          12.661633
Name: Fare, dtype: float64
```

In [39]:
```
titanic["Pclass"].value_counts()
```

Out[39]:
```
3    491
1    216
2    184
Name: Pclass, dtype: int64
```

In [40]:
```
titanic.groupby("Pclass")["Pclass"].count()
```

Out[40]:
```
Pclass
1    216
2    184
3    491
Name: Pclass, dtype: int64
```

In [41]:
```
titanic.sort_values(by="Age",ascending=False).head()
```

Out[41]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
630	631	1	1	Barkworth, Mr. Algernon Henry Wilson	male	80.0	0	0	27042	30.0000	A
851	852	0	3	Svensson, Mr. Johan	male	74.0	0	0	347060	7.7750	N
493	494	0	1	Artagaveytia, Mr. Ramon	male	71.0	0	0	PC 17609	49.5042	N
96	97	0	1	Goldschmidt, Mr. George B	male	71.0	0	0	PC 17754	34.6542	
116	117	0	3	Connors, Mr. Patrick	male	70.5	0	0	370369	7.7500	N

In [42]:
```
titanic.sort_values(by=['Pclass', 'Age'], ascending=False).head()
```

Out[42]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
851	852	0	3	Svensson, Mr. Johan	male	74.0	0	0	347060	7.7750	Nal
116	117	0	3	Connors, Mr. Patrick	male	70.5	0	0	370369	7.7500	Nal
280	281	0	3	Duane, Mr. Frank	male	65.0	0	0	336439	7.7500	Nal
483	484	1	3	Turkula, Mrs. (Hedwig)	female	63.0	0	0	4134	9.5875	Nal
326	327	0	3	Nysveen, Mr. Johan Hansen	male	61.0	0	0	345364	6.2375	Nal

In [43]:
```
titanic.dtypes
```

Out[43]:
```
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
```

In [44]:
```
titanic["Name"].str.lower()
```

Out[44]:
```
0                                    anonymous
1                                    anonymous
2                                    anonymous
3      futrelle, mrs. jacques heath (lily may peel)
4                           allen, mr. william henry
                           ...
886                            montvila, rev. juozas
887                     graham, miss. margaret edith
888         johnston, miss. catherine helen "carrie"
889                            behr, mr. karl howell
890                              dooley, mr. patrick
Name: Name, Length: 891, dtype: object
```

In [45]:
```
titanic["Name"].str.split(",")
```

Out[45]:
```
0                                          [anonymous]
1                                          [anonymous]
2                                          [anonymous]
3      [Futrelle,  Mrs. Jacques Heath (Lily May Peel)]
4                           [Allen,  Mr. William Henry]
                             ...
886                           [Montvila,  Rev. Juozas]
887                    [Graham,  Miss. Margaret Edith]
888        [Johnston,  Miss. Catherine Helen "Carrie"]
889                           [Behr,  Mr. Karl Howell]
890                             [Dooley,  Mr. Patrick]
Name: Name, Length: 891, dtype: object
```

In [46]:
```
titanic["Surname"] = titanic["Name"].str.split(",").str.get(0)
titanic["Surname"]
```

Out[46]:
```
0      anonymous
1      anonymous
2      anonymous
3       Futrelle
4          Allen
          ...
886     Montvila
887       Graham
888     Johnston
889         Behr
890       Dooley
Name: Surname, Length: 891, dtype: object
```

In [47]:
```
titanic["Name_main"] = titanic["Name"].str.split(",").str.get(1)
titanic["Name_main"]
```

Out[47]:
```
0                                      NaN
1                                      NaN
2                                      NaN
3       Mrs. Jacques Heath (Lily May Peel)
4                           Mr. William Henry
                        ...
886                             Rev. Juozas
887                    Miss. Margaret Edith
888         Miss. Catherine Helen "Carrie"
889                           Mr. Karl Howell
890                               Mr. Patrick
Name: Name_main, Length: 891, dtype: object
```

In [48]:
```
titanic["Name"].str.split(",")
```

Out[48]:
```
0                                          [anonymous]
1                                          [anonymous]
2                                          [anonymous]
3      [Futrelle,  Mrs. Jacques Heath (Lily May Peel)]
4                           [Allen,  Mr. William Henry]
                            ...
886                           [Montvila,  Rev. Juozas]
887                    [Graham,  Miss. Margaret Edith]
888        [Johnston,  Miss. Catherine Helen "Carrie"]
889                           [Behr,  Mr. Karl Howell]
890                             [Dooley,  Mr. Patrick]
Name: Name, Length: 891, dtype: object
```

In [49]:
```
titanic['Real_Name'] = titanic["Name"].str.split(",").str.get(0)
titanic.head()
```

Out[49]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
0	1	0	3	anonymous	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	anonymous	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	anonymous	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [50]:
```
titanic['Surname'] = titanic["Name"].str.split(",").str.get(1)
titanic.head()
```

Out[50]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
0	1	0	3	anonymous	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	anonymous	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	anonymous	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [51]:
```
titanic['Salutation'] = titanic['Surname'].str.split(".").str.get(0)
titanic.head()
```

Out[51]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
0	1	0	3	anonymous	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	anonymous	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	anonymous	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [52]:
```
titanic["Name"].str.contains("Mr")
```

Out[52]:
```
0      False
1      False
2      False
3       True
4       True
       ...
886    False
887    False
888    False
889     True
890     True
Name: Name, Length: 891, dtype: bool
```

In [53]:
```
titanic[titanic["Name"].str.contains("Countess")]
```

Out[53]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
759	760	1	1	Rothes, the Countess. of (Lucy Noel Martha Dye...	female	33.0	0	0	110152	86.5	B77

In [54]:
```
titanic["Name"].str.len()
```

Out[54]:
```
0       9
1       9
2       9
3      44
4      24
       ..
886    21
887    28
888    40
889    21
890    19
Name: Name, Length: 891, dtype: int64
```

In [55]:
```
titanic["Name"].str.len().idxmax()
```

Out[55]:
```
307
```

In [56]:
```
titanic.loc[titanic["Name"].str.len().idxmax(), "Name"]
```

Out[56]:
```
'Penasco y Castellana, Mrs. Victor de Satode (Maria Josefa Perez de Soto y Va
llejo)'
```

In [57]:
```
titanic.loc[titanic["Name"].str.len().idxmin(), "Name"]
```

Out[57]:
```
'anonymous'
```

In [58]:
```
titanic["Sex_short"] = titanic["Sex"].replace({"male": "M", "female": "F"})
titanic["Sex_short"]
```

Out[58]:
```
0      M
1      F
2      F
3      F
4      M
      ..
886    M
887    F
888    F
889    M
890    M
Name: Sex_short, Length: 891, dtype: object
```

In [59]:
```
titanic["Sex_short"] = titanic["Sex"].str.replace("female", "F")
titanic["Sex_short"] = titanic["Sex_short"].str.replace("male", "M")
```

In [170]:
```
import numpy as np
df = pd.DataFrame(np.random.randn(10, 3), columns=list("abc"))
df[["a", "c","b"]]
```

Out[170]:

	a	c	b
0	0.971377	-0.762178	-0.305884
1	0.412251	0.588495	0.096369
2	1.801618	-0.597973	1.489147
3	-0.359858	-0.878680	-1.461579
4	-0.455795	0.681250	0.973445
5	0.852787	-0.544525	-0.295961
6	-1.098355	-1.421945	-0.417816
7	-0.133820	-0.183852	1.228257
8	-0.495825	-1.226723	-0.318924
9	-0.064218	-0.306832	-0.345931

In [171]:
```
df.loc[:, ["a", "c"]]
```

Out[171]:

	a	c
0	0.971377	-0.762178
1	0.412251	0.588495
2	1.801618	-0.597973
3	-0.359858	-0.878680
4	-0.455795	0.681250
5	0.852787	-0.544525
6	-1.098355	-1.421945
7	-0.133820	-0.183852
8	-0.495825	-1.226723
9	-0.064218	-0.306832

Good Code

In [ ]:
```
named = list("abcdefg")
n = 30
columns = named + np.arange(len(named), n).tolist()
df = pd.DataFrame(np.random.randn(n, n), columns=columns)
df.iloc[:, np.r_[:10, 24:30]]
```

In [ ]:
```
df = pd.DataFrame({

"v1": [1, 3, 5, 7, 8, 3, 5, np.nan, 4, 5, 7, 9],
"v2": [11, 33, 55, 77, 88, 33, 55, np.nan, 44, 55, 77, 99],
"by1": ["red", "blue", 1, 2, np.nan, "big", 1, 2, "red", 1, np.nan, 12],
"by2": ["wet","dry",99,95,np.nan,"damp",95,99,"red",99,np.nan,np.nan,]

})

df
```

In [ ]:
```
g = df.groupby(["by1", "by2"])
g[["v1", "v2"]].mean()
```

In [63]:
```
import numpy as np
s = pd.Series(np.arange(5), dtype=np.float32)
s
```

Out[63]:
```
0    0.0
1    1.0
2    2.0
3    3.0
4    4.0
dtype: float32
```

In [64]:
```
s.isin([2, 4])
```

Out[64]:
```
0    False
1    False
2     True
3    False
4     True
dtype: bool
```

Data genetarion code

In [65]:
```
# Data genetarion code

import random
import string

baseball = pd.DataFrame({
"team": ["team %d" % (x + 1) for x in range(5)] * 5,
"player": random.sample(list(string.ascii_lowercase), 25),
"batting avg": np.random.uniform(0.200, 0.400, 25),
    }
)

baseball
```

Out[65]:

	team	player	batting avg
0	team 1	b	0.311944
1	team 2	w	0.300678
2	team 3	c	0.271453
3	team 4	p	0.301531
4	team 5	a	0.257927
5	team 1	q	0.384259
6	team 2	d	0.279827
7	team 3	s	0.200344
8	team 4	f	0.269042
9	team 5	k	0.363716
10	team 1	u	0.355087
11	team 2	v	0.276580
12	team 3	z	0.381452
13	team 4	g	0.264230
14	team 5	e	0.397186
15	team 1	n	0.249416
16	team 2	i	0.245684
17	team 3	y	0.316917
18	team 4	o	0.206810
19	team 5	m	0.272293
20	team 1	h	0.328023
21	team 2	r	0.352936
22	team 3	t	0.350134
23	team 4	x	0.368002
24	team 5	j	0.282423

In [66]:
```
baseball.pivot_table(values="batting avg", columns="team", aggfunc=np.max)
```

Out[66]:

team	team 1	team 2	team 3	team 4	team 5
batting avg	0.384259	0.352936	0.381452	0.368002	0.397186

In [67]:
```
df = pd.DataFrame({"a": np.random.randn(10), "b": np.random.randn(10)})
df.head()
```

Out[67]:

	a	b
0	0.848214	1.528596
1	-1.363807	-1.321466
2	-0.525568	1.252385
3	-0.351879	-0.315065
4	-0.700257	0.328759

In [68]:
```
df.query("a <= b")
```

Out[68]:

	a	b
0	0.848214	1.528596
1	-1.363807	-1.321466
2	-0.525568	1.252385
3	-0.351879	-0.315065
4	-0.700257	0.328759
5	-0.927895	-0.516451
6	-1.526683	-0.259954
7	-2.127308	0.531293
8	-0.206562	0.237485
9	0.413750	0.770686

In [69]:
```
df[df["a"] <= df["b"]]
```

Out[69]:

	a	b
0	0.848214	1.528596
1	-1.363807	-1.321466
2	-0.525568	1.252385
3	-0.351879	-0.315065
4	-0.700257	0.328759
5	-0.927895	-0.516451
6	-1.526683	-0.259954
7	-2.127308	0.531293
8	-0.206562	0.237485
9	0.413750	0.770686

In [70]:
```
df.loc[df["a"] <= df["b"]]
```

Out[70]:

	a	b
0	0.848214	1.528596
1	-1.363807	-1.321466
2	-0.525568	1.252385
3	-0.351879	-0.315065
4	-0.700257	0.328759
5	-0.927895	-0.516451
6	-1.526683	-0.259954
7	-2.127308	0.531293
8	-0.206562	0.237485
9	0.413750	0.770686

In [71]:
```
df[df["a"] >= df["b"]]
```

Out[71]:

	a	b

In [72]:
```
df = pd.DataFrame({"a": np.random.randn(10), "b": np.random.randn(10)})
df.head()
```

Out[72]:

	a	b
0	0.057788	-0.548498
1	-0.150495	-1.303927
2	0.391174	-0.383887
3	-0.486376	0.660384
4	-0.149571	0.048288

In [73]:
```
df.eval("a + b")
```

Out[73]:
```
0   -0.490711
1   -1.454421
2    0.007287
3    0.174008
4   -0.101283
5   -0.636180
6    0.540169
7   -0.429076
8    0.185766
9   -1.048530
dtype: float64
```

In [74]:
```
df["a"] + df["b"]
```

Out[74]:
```
0   -0.490711
1   -1.454421
2    0.007287
3    0.174008
4   -0.101283
5   -0.636180
6    0.540169
7   -0.429076
8    0.185766
9   -1.048530
dtype: float64
```

In [75]:
```
df = pd.DataFrame({

"x": np.random.uniform(1.0, 168.0, 120),
"y": np.random.uniform(7.0, 334.0, 120),
"z": np.random.uniform(1.7, 20.7, 120),
"month": [5, 6, 7, 8] * 30,
"week": np.random.randint(1, 4, 120)

})

df.head()
```

Out[75]:

	x	y	z	month	week
0	41.516759	33.530167	18.106587	5	1
1	35.505487	252.682232	14.136898	6	3
2	91.041404	170.303748	7.498431	7	2
3	26.488195	332.594130	5.038641	8	3
4	105.767124	107.686286	15.308504	5	2

In [76]:
```
grouped = df.groupby(["month", "week"])
grouped["x"].agg([np.mean, np.std])
```

Out[76]:

		mean	std
month	**week**		
	1	87.949769	52.340418
5	**2**	80.472147	51.547185
	3	83.919926	38.707154
	1	62.640842	45.863364
6	**2**	76.669235	49.722219
	3	74.546488	45.781976
	1	83.765432	35.884936
7	**2**	120.548997	36.888050
	3	90.085512	49.915038
	1	85.325932	60.874941
8	**2**	50.293565	37.083210
	3	59.143017	41.149256

In [77]:
```
a = np.array(list(range(1, 24)) + [np.NAN]).reshape(2, 3, 4)
a
```

Out[77]:
```
array([[[ 1.,  2.,  3.,  4.],
        [ 5.,  6.,  7.,  8.],
        [ 9., 10., 11., 12.]],

       [[13., 14., 15., 16.],
        [17., 18., 19., 20.],
        [21., 22., 23., nan]]])
```

In [78]:
```
pd.DataFrame([tuple(list(x) + [val]) for x, val in np.ndenumerate(a)])
```

Out[78]:

	0	1	2	3
0	0	0	0	1.0
1	0	0	1	2.0
2	0	0	2	3.0
3	0	0	3	4.0
4	0	1	0	5.0
5	0	1	1	6.0
6	0	1	2	7.0
7	0	1	3	8.0
8	0	2	0	9.0
9	0	2	1	10.0
10	0	2	2	11.0
11	0	2	3	12.0
12	1	0	0	13.0
13	1	0	1	14.0
14	1	0	2	15.0
15	1	0	3	16.0
16	1	1	0	17.0
17	1	1	1	18.0
18	1	1	2	19.0
19	1	1	3	20.0
20	1	2	0	21.0
21	1	2	1	22.0
22	1	2	2	23.0
23	1	2	3	NaN

In [79]:
```
a = list(enumerate(list(range(1, 5)) + [np.NAN]))
a
```

Out[79]:
```
[(0, 1), (1, 2), (2, 3), (3, 4), (4, nan)]
```

In [80]:
```
pd.DataFrame(a)
```

Out[80]:

	0	1
0	0	1.0
1	1	2.0
2	2	3.0
3	3	4.0
4	4	NaN

In [81]:
```
cheese = pd.DataFrame({

"first": ["John", "Mary"],
"last": ["Doe", "Bo"],
"height": [5.5, 6.0],
"weight": [130, 150]

})

cheese
```

Out[81]:

	first	last	height	weight
0	John	Doe	5.5	130
1	Mary	Bo	6.0	150

In [82]:
```
pd.melt(cheese, id_vars=["first", "last"])
```

Out[82]:

	first	last	variable	value
0	John	Doe	height	5.5
1	Mary	Bo	height	6.0
2	John	Doe	weight	130.0
3	Mary	Bo	weight	150.0

In [83]:
```
cheese.set_index(["first", "last"]).stack() # alternative
```

Out[83]:
```
first  last
John   Doe   height      5.5
              weight    130.0
Mary   Bo    height      6.0
              weight    150.0
dtype: float64
```

In [84]:
```
df = pd.DataFrame({

"x": np.random.uniform(1.0, 168.0, 12),
"y": np.random.uniform(7.0, 334.0, 12),
"z": np.random.uniform(1.7, 20.7, 12),
"month": [5, 6, 7] * 4,
"week": [1, 2] * 6

})

mdf = pd.melt(df, id_vars=["month", "week"])

pd.pivot_table(mdf,values="value",index=["variable", "week"],columns=["month"],
```

Out[84]:

	month	5	6	7
variable	week			
x	1	69.688604	58.926280	50.639441
	2	67.470350	117.676001	118.517232
y	1	160.009684	21.384183	177.609046
	2	203.316298	197.839213	184.583499
z	1	10.060922	4.185807	8.656566
	2	8.009302	11.105621	11.506984

In [85]:
```
df = pd.DataFrame({

"Animal": ["Animal1","Animal2","Animal3","Animal2","Animal1","Animal2","Animal3
"FeedType": ["A", "B", "A", "A", "B", "B", "A"],
"Amount": [10, 7, 4, 2, 5, 6, 2]

})

df.pivot_table(values="Amount", index="Animal", columns="FeedType", aggfunc="su
```

Out[85]:

FeedType	A	B
Animal		
Animal1	10.0	5.0
Animal2	2.0	13.0
Animal3	6.0	NaN

In [86]:
```
df.groupby(["Animal", "FeedType"])["Amount"].sum()
```

Out[86]:
```
Animal   FeedType
Animal1  A           10
         B            5
Animal2  A            2
         B           13
Animal3  A            6
Name: Amount, dtype: int64
```

In [87]:
```
pd.cut(pd.Series([1, 2, 3, 4, 5, 6]), 3)
```

Out[87]:
```
0    (0.995, 2.667]
1    (0.995, 2.667]
2    (2.667, 4.333]
3    (2.667, 4.333]
4      (4.333, 6.0]
5      (4.333, 6.0]
dtype: category
Categories (3, interval[float64, right]): [(0.995, 2.667] < (2.667, 4.333] <
(4.333, 6.0]]
```

In [88]:
```
pd.Series([1, 2, 3, 2, 2, 3]).astype("category")
```

Out[88]:
```
0    1
1    2
2    3
3    2
4    2
5    3
dtype: category
Categories (3, int64): [1, 2, 3]
```

In [89]:
```
frame = pd.DataFrame({"col1": ["A", "B", np.NaN, "C", "D"], "col2": ["F", np.Na

frame
```

Out[89]:

	col1	col2
0	A	F
1	B	NaN
2	NaN	G
3	C	H
4	D	I

In [90]:
```
frame[frame["col2"].isna()]
```

Out[90]:

	col1	col2
1	B	NaN

In [91]:
```
frame[frame["col1"].notna()]
```

Out[91]:

	col1	col2
0	A	F
1	B	NaN
3	C	H
4	D	I

In [92]:
```
df1 = pd.DataFrame({"key": ["A", "B", "C", "D"], "value": np.random.randn(4)})
df2 = pd.DataFrame({"key": ["B", "D", "D", "E"], "value": np.random.randn(4)})
```

In [93]:
```
pd.merge(df1, df2, on="key")
```

Out[93]:

	key	value_x	value_y
0	B	-0.335446	1.794026
1	D	1.224740	1.418379
2	D	1.224740	0.425891

In [94]:
```
indexed_df2 = df2.set_index("key")
pd.merge(df1, indexed_df2, left_on="key", right_index=True)
```

Out[94]:

	key	value_x	value_y
1	B	-0.335446	1.794026
3	D	1.224740	1.418379
3	D	1.224740	0.425891

In [95]:
```
pd.merge(df1, df2, on="key", how="left")
```

Out[95]:

	key	value_x	value_y
0	A	0.429288	NaN
1	B	-0.335446	1.794026
2	C	-0.685751	NaN
3	D	1.224740	1.418379
4	D	1.224740	0.425891

In [96]:
```
pd.merge(df1, df2, on="key", how="right")
```

Out[96]:

	key	value_x	value_y
0	B	-0.335446	1.794026
1	D	1.224740	1.418379
2	D	1.224740	0.425891
3	E	NaN	0.828731

In [97]:
```
pd.merge(df1, df2, on="key", how="outer")
```

Out[97]:

	key	value_x	value_y
0	A	0.429288	NaN
1	B	-0.335446	1.794026
2	C	-0.685751	NaN
3	D	1.224740	1.418379
4	D	1.224740	0.425891
5	E	NaN	0.828731

In [98]:
```
df1 = pd.DataFrame({"city": ["Chicago", "San Francisco", "New York City"], "rar

df2 = pd.DataFrame({"city": ["Chicago", "Boston", "Los Angeles"], "rank": [1, 4

pd.concat([df1, df2])
```

Out[98]:

	city	rank
0	Chicago	1
1	San Francisco	2
2	New York City	3
0	Chicago	1
1	Boston	4
2	Los Angeles	5

In [99]:
```
pd.concat([df1, df2]).drop_duplicates()
```

Out[99]:

	city	rank
0	Chicago	1
1	San Francisco	2
2	New York City	3
1	Boston	4
2	Los Angeles	5

In [100]:
```
df = pd.DataFrame({"x": [1, 3, 5], "y": [2, 4, 6]})
df
```

Out[100]:

	x	y
0	1	2
1	3	4
2	5	6

In [101]:
```
firstlast = pd.DataFrame({"String": ["John Smith", "Jane Cook"]})
firstlast["First_Name"] = firstlast["String"].str.split(" ", expand=True)[0]
firstlast["Last_Name"] = firstlast["String"].str.rsplit(" ", expand=True)[1]
firstlast
```

Out[101]:

	String	First_Name	Last_Name
0	John Smith	John	Smith
1	Jane Cook	Jane	Cook

In [102]:
```
firstlast = pd.DataFrame({"string": ["John Smith", "Jane Cook"]})
firstlast["upper"] = firstlast["string"].str.upper()
firstlast["lower"] = firstlast["string"].str.lower()
firstlast["title"] = firstlast["string"].str.title()
firstlast
```

Out[102]:

	string	upper	lower	title
0	John Smith	JOHN SMITH	john smith	John Smith
1	Jane Cook	JANE COOK	jane cook	Jane Cook

In [103]:
```
df1 = pd.DataFrame({"key": ["A", "B", "C", "D"], "value": np.random.randn(4)})
df1
```

Out[103]:

	key	value
0	A	2.710180
1	B	-0.184712
2	C	-0.268376
3	D	1.136070

In [104]:
```
df2 = pd.DataFrame({"key": ["B", "D", "D", "E"], "value": np.random.randn(4)})
df2
```

Out[104]:

	key	value
0	B	-1.961649
1	D	0.885771
2	D	0.695118
3	E	-0.265280

In [105]:
```
inner_join = df1.merge(df2, on=["key"], how="inner")
inner_join
```

Out[105]:

	key	value_x	value_y
0	B	-0.184712	-1.961649
1	D	1.136070	0.885771
2	D	1.136070	0.695118

In [106]:
```
left_join = df1.merge(df2, on=["key"], how="left")
left_join
```

Out[106]:

	key	value_x	value_y
0	A	2.710180	NaN
1	B	-0.184712	-1.961649
2	C	-0.268376	NaN
3	D	1.136070	0.885771
4	D	1.136070	0.695118

In [107]:
```
right_join = df1.merge(df2, on=["key"], how="right")
right_join
```

Out[107]:

	key	value_x	value_y
0	B	-0.184712	-1.961649
1	D	1.136070	0.885771
2	D	1.136070	0.695118
3	E	NaN	-0.265280

In [108]:
```
outer_join = df1.merge(df2, on=["key"], how="outer")
outer_join
```

Out[108]:

	key	value_x	value_y
0	A	2.710180	NaN
1	B	-0.184712	-1.961649
2	C	-0.268376	NaN
3	D	1.136070	0.885771
4	D	1.136070	0.695118
5	E	NaN	-0.265280

In [109]:
```
df = pd.DataFrame({"AAA": [1] * 8, "BBB": list(range(0, 8))})
df
```

Out[109]:

	AAA	BBB
0	1	0
1	1	1
2	1	2
3	1	3
4	1	4
5	1	5
6	1	6
7	1	7

In [110]:
```
series = list(range(1, 5))
series
```

Out[110]:
```
[1, 2, 3, 4]
```

In [111]:
```
df.loc[2:5, "AAA"] = series
df
```

Out[111]:

	AAA	BBB
0	1	0
1	1	1
2	1	2
3	2	3
4	3	4
5	4	5
6	1	6
7	1	7

In [112]:
```
df = pd.DataFrame({

"class": ["A", "A", "A", "B", "C", "D"],
"student_count": [42, 35, 42, 50, 47, 45],
"all_pass": ["Yes", "Yes", "Yes", "No", "No", "Yes"]

})

df.drop_duplicates()
```

Out[112]:

	class	student_count	all_pass
0	A	42	Yes
1	A	35	Yes
3	B	50	No
4	C	47	No
5	D	45	Yes

In [113]:
```
df.drop_duplicates(["class", "student_count"])
```

Out[113]:

	class	student_count	all_pass
0	A	42	Yes
1	A	35	Yes
3	B	50	No
4	C	47	No
5	D	45	Yes

In [114]:
```
new_row = pd.DataFrame([["E", 51, True]],columns=["class", "student_count", "a
pd.concat([df, new_row])
```

Out[114]:

	class	student_count	all_pass
0	A	42	Yes
1	A	35	Yes
2	A	42	Yes
3	B	50	No
4	C	47	No
5	D	45	Yes
0	E	51	True

In [115]:
```
df = pd.DataFrame({"x": [1, 3, 5], "y": [2, 4, 6]})
df
```

Out[115]:

	x	y
0	1	2
1	3	4
2	5	6

In [116]:
```
df1 = pd.DataFrame({"key": ["A", "B", "C", "D"], "value": np.random.randn(4)})
df1
```

Out[116]:

	key	value
0	A	-1.402688
1	B	-0.545334
2	C	-1.455278
3	D	0.697387

In [117]:
```
df2 = pd.DataFrame({"key": ["B", "D", "D", "E"], "value": np.random.randn(4)})
df2
```

Out[117]:

	key	value
0	B	-0.665418
1	D	0.008734
2	D	-0.719310
3	E	-0.507211

In [118]:
```
inner_join = df1.merge(df2, on=["key"], how="inner")
inner_join
```

Out[118]:

	key	value_x	value_y
0	B	-0.545334	-0.665418
1	D	0.697387	0.008734
2	D	0.697387	-0.719310