02_pandas-tips

November 16, 2023

[ ]:
```python
import pandas as pd
import numpy as np
```

[ ]:
```python
# check the pandas version
print(pd.__version__)
```

```
2.1.3
```

[ ]:
```python
# second method to see the pandas version
pd.show_versions()
```

```
c:\Users\adnan\miniconda3\envs\python_eda\Lib\site-
packages\_distutils_hack\__init__.py:33: UserWarning: Setuptools is replacing
distutils.
  warnings.warn("Setuptools is replacing distutils.")


INSTALLED VERSIONS
------------------
commit              : 2a953cf80b77e4348bf50ed724f8abc0d814d9dd
python              : 3.11.5.final.0
python-bits         : 64
OS                  : Windows
OS-release          : 10
Version             : 10.0.22623
machine             : AMD64
processor           : Intel64 Family 6 Model 142 Stepping 11, GenuineIntel
byteorder           : little
LC_ALL              : None
LANG                : None
LOCALE              : English_Pakistan.1252

pandas              : 2.1.3
numpy               : 1.26.2
pytz                : 2023.3.post1
dateutil            : 2.8.2
setuptools          : 68.0.0
pip                 : 23.3
Cython              : None
```

```
pytest              : None
hypothesis          : None
sphinx              : None
blosc               : None
feather             : None
xlsxwriter          : None
lxml.etree          : None
html5lib            : None
pymysql             : None
psycopg2            : None
jinja2              : 3.1.2
IPython             : 8.15.0
pandas_datareader   : None
bs4                 : 4.12.2
bottleneck          : None
dataframe-api-compat: None
fastparquet         : None
fsspec              : None
gcsfs               : None
matplotlib          : 3.8.1
numba               : None
numexpr             : None
odfpy               : None
openpyxl            : None
pandas_gbq          : None
pyarrow             : None
pyreadstat          : None
pyxlsb              : None
s3fs                : None
scipy               : None
sqlalchemy          : None
tables              : None
tabulate            : None
xarray              : None
xlrd                : None
zstandard           : None
tzdata              : 2023.3
qtpy                : None
pyqt5               : None
```

0.0.1 2- Make a dataframe

[ ]:
```
df = pd.DataFrame({'Adnan': [1, 2, 3,], 'Haider': [4,5,6]})
df
```

[ ]:
```
   Adnan  Haider
0      1       4
1      2       5
```

```
2      3       6
```

```
[ ]: # use numpy array to make data-frame
     import numpy as np
     arr = np.array([[1,2,3], [4,5,6], [7,8,9]])
     arr
     # convert array into df
     df = pd.DataFrame(arr)
     df
```

```
[ ]:    0  1  2
     0  1  2  3
     1  4  5  6
     2  7  8  9
```

```
[ ]: # numpy array to dataframe
     df = pd.DataFrame(np.random.rand(3,6))
     df
```

```
[ ]:           0         1         2         3         4         5
     0  0.031042  0.844721  0.219745  0.867777  0.177176  0.438917
     1  0.538490  0.533392  0.844861  0.100089  0.821783  0.856601
     2  0.437989  0.824156  0.924164  0.320594  0.180362  0.733545
```

```
[ ]: # change the column name
     df = pd.DataFrame(np.random.rand(3, 6), columns=list('ABCDEF'))
     df
```

```
[ ]:           A         B         C         D         E         F
     0  0.863279  0.312047  0.686569  0.104209  0.613657  0.752895
     1  0.864922  0.172354  0.976212  0.489770  0.738529  0.710735
     2  0.533605  0.453174  0.865075  0.611988  0.918953  0.458909
```

0.0.2 3-Rename column names

```
[ ]: # Create df by using panda
     df = pd.DataFrame({'Adnan': [1, 2, 3,], 'Haider': [4,5,6]})
     df

     # Change the column name of the df
     df.rename(columns={'Adnan': "Aadi", 'Haider': "Ali"}, inplace=True)
     df
```

```
[ ]:    Aadi  Ali
     0     1    4
     1     2    5
     2     3    6
```

```
[ ]: # rename columns
     df.columns=['Adnan_1', "Haider_2"]
     df
```

```
[ ]:    Adnan_1  Haider_2
     0        1         4
     1        2         5
     2        3         6
```

```
[ ]: # to replace any character ,string
     df.columns = df.columns.str.replace("_", "*")
     df
```

```
[ ]:    Adnan*1  Haider*2
     0        1         4
     1        2         5
     2        3         6
```

```
[ ]: # adding prefix to columns
     df = df.add_prefix("A_")
     df
```

```
[ ]:    A_Adnan*1  A_Haider*2
     0          1           4
     1          2           5
     2          3           6
```

```
[ ]: # add suffix to the columns
     df = df.add_suffix("_H")
     df
```

```
[ ]:    A_Adnan*1_H  A_Haider*2_H
     0            1             4
     1            2             5
     2            3             6
```

```
[ ]: # again change to the first one
     df.columns=['Adnan', "Haider"]
     df
```

```
[ ]:    Adnan  Haider
     0      1       4
     1      2       5
     2      3       6
```

0.0.3 4- Using template data

[ ]:
```
import pandas as pd
import numpy as np
import seaborn as sns

# load dataset
tips = sns.load_dataset("tips")
tips.head()
```

[ ]:
```
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4
```

[ ]:
```
tips.describe()
```

[ ]:
```
       total_bill         tip        size
count  244.000000  244.000000  244.000000
mean    19.785943    2.998279    2.569672
std      8.902412    1.383638    0.951100
min      3.070000    1.000000    1.000000
25%     13.347500    2.000000    2.000000
50%     17.795000    2.900000    2.000000
75%     24.127500    3.562500    3.000000
max     50.810000   10.000000    6.000000
```

[ ]:
```
# column names
tips.columns
```

[ ]:
```
Index(['total_bill', 'tip', 'sex', 'smoker', 'day', 'time', 'size'],
dtype='object')
```

[ ]:
```
pip install openpyxl
```

```
Collecting openpyxl
  Downloading openpyxl-3.1.2-py2.py3-none-any.whl (249 kB)
     ---------------------------------------- 0.0/250.0 kB ? eta -:--:--
     ---------------------------------------- 0.0/250.0 kB ? eta -:--:--
     - -------------------------------------- 10.2/250.0 kB ? eta -:--:--
     - -------------------------------------- 10.2/250.0 kB ? eta -:--:--
     ---- ----------------------------------- 30.7/250.0 kB 262.6 kB/s eta 0:00:01
     --------- ------------------------------ 61.4/250.0 kB 297.7 kB/s eta 0:00:01
     -------------- ------------------------- 92.2/250.0 kB 403.5 kB/s eta 0:00:01
     -------------------- ------------------- 143.4/250.0 kB 502.3 kB/s eta 0:00:01
     ------------------------- -------------- 174.1/250.0 kB 525.1 kB/s eta 0:00:01
     ----------------------------- ------ 204.8/250.0 kB 541.9 kB/s eta 0:00:01
     ----------------------------- ------ 204.8/250.0 kB 541.9 kB/s eta 0:00:01
     -------------------------------- --- 225.3/250.0 kB 474.7 kB/s eta 0:00:01
     -------------------------------- --- 225.3/250.0 kB 474.7 kB/s eta 0:00:01
     -----------------------------------  245.8/250.0 kB 419.1 kB/s eta 0:00:01
     ------------------------------------ 250.0/250.0 kB 393.7 kB/s eta 0:00:00
Collecting et-xmlfile (from openpyxl)
  Downloading et_xmlfile-1.1.0-py3-none-any.whl (4.7 kB)
Installing collected packages: et-xmlfile, openpyxl
Successfully installed et-xmlfile-1.1.0 openpyxl-3.1.2
Note: you may need to restart the kernel to use updated packages.
```

[ ]:
```
# saving a template dataset
tips.to_csv('tips_save.csv')
tips.to_excel('tips.save.xlsx')
```

0.0.4 5-Reverse Row order

[ ]:
```
import seaborn as sns
import pandas as pd

df = sns.load_dataset("titanic")
df.head()
```

[ ]:
```
   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  22.0      1      0   7.2500        S  Third
1         1       1  female  38.0      1      0  71.2833        C  First
2         1       3  female  26.0      0      0   7.9250        S  Third
3         1       1  female  35.0      1      0  53.1000        S  First
4         0       3    male  35.0      0      0   8.0500        S  Third

     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no  False
1  woman       False    C    Cherbourg   yes  False
2  woman       False  NaN  Southampton   yes   True
3  woman       False    C  Southampton   yes  False
4    man        True  NaN  Southampton    no   True
```

[ ]:
```
# reverse the row
df.loc[::-1].head()
```

[ ]:
```
     survived  pclass     sex   age  sibsp  parch   fare embarked   class  \
890         0       3    male  32.0      0      0   7.75        Q   Third
889         1       1    male  26.0      0      0  30.00        C   First
888         0       3  female   NaN      1      2  23.45        S   Third
887         1       1  female  19.0      0      0  30.00        S   First
886         0       2    male  27.0      0      0  13.00        S  Second

       who  adult_male deck  embark_town alive  alone
```

```
890    man        True  NaN  Queenstown   no   True
889    man        True    C   Cherbourg  yes   True
888  woman       False  NaN Southampton   no  False
887  woman       False    B Southampton  yes   True
886    man        True  NaN Southampton   no   True
```

```
[ ]: # reset the row order
     df.loc[::-1].reset_index(drop=True).head()
```

```
[ ]:    survived  pclass     sex   age  sibsp  parch   fare embarked   class  \
     0         0       3    male  32.0      0      0   7.75        Q   Third
     1         1       1    male  26.0      0      0  30.00        C   First
     2         0       3  female   NaN      1      2  23.45        S   Third
     3         1       1  female  19.0      0      0  30.00        S   First
     4         0       2    male  27.0      0      0  13.00        S  Second

          who  adult_male deck  embark_town alive  alone
     0    man        True  NaN   Queenstown    no   True
     1    man        True    C    Cherbourg   yes   True
     2  woman       False  NaN  Southampton    no  False
     3  woman       False    B  Southampton   yes   True
     4    man        True  NaN  Southampton    no   True
```

0.0.5 7-Reverse Column order

```
[ ]: # reverse column order
     df_reverse = df.loc[:, ::-1].head()
     df_reverse
```

```
[ ]:    alone alive  embark_town deck  adult_male    who  class embarked     fare  \
     0  False    no  Southampton  NaN        True    man  Third        S   7.2500
     1  False   yes    Cherbourg    C       False  woman  First        C  71.2833
     2   True   yes  Southampton  NaN       False  woman  Third        S   7.9250
     3  False   yes  Southampton    C       False  woman  First        S  53.1000
     4   True    no  Southampton  NaN        True    man  Third        S   8.0500

        parch  sibsp   age     sex  pclass  survived
     0      0      1  22.0    male       3         0
     1      0      1  38.0  female       1         1
     2      0      0  26.0  female       3         1
     3      0      1  35.0  female       1         1
     4      0      0  35.0    male       3         0
```

```
[ ]: # reset the column order
     df_reset = df_reverse.reindex(columns=df.columns)
     df_reset
```

```
[ ]:    survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
     0         0       3    male  22.0      1      0   7.2500        S  Third
     1         1       1  female  38.0      1      0  71.2833        C  First
     2         1       3  female  26.0      0      0   7.9250        S  Third
     3         1       1  female  35.0      1      0  53.1000        S  First
     4         0       3    male  35.0      0      0   8.0500        S  Third

          who  adult_male deck  embark_town alive  alone
     0    man        True  NaN  Southampton    no  False
     1  woman       False    C    Cherbourg   yes  False
     2  woman       False  NaN  Southampton   yes   True
     3  woman       False    C  Southampton   yes  False
     4    man        True  NaN  Southampton    no   True
```

0.0.6 8- Select a column by dtype

```
[ ]: df.dtypes
```

```
[ ]: survived          int64
     pclass            int64
     sex              object
     age             float64
     sibsp             int64
     parch             int64
     fare            float64
     embarked         object
     class          category
     who              object
     adult_male         bool
     deck           category
     embark_town      object
     alive            object
     alone              bool
     dtype: object
```

```
[ ]: # select only those columns which have numeric values
     df.select_dtypes(include='number').head()
```

```
[ ]:    survived  pclass   age  sibsp  parch     fare
     0         0       3  22.0      1      0   7.2500
     1         1       1  38.0      1      0  71.2833
     2         1       3  26.0      0      0   7.9250
     3         1       1  35.0      1      0  53.1000
     4         0       3  35.0      0      0   8.0500
```

```
[ ]: # select only those columns which have object
     df.select_dtypes(include='object').head()
```

```
[ ]:       sex embarked    who  embark_town alive
     0    male        S    man  Southampton    no
     1  female        C  woman    Cherbourg   yes
     2  female        S  woman  Southampton   yes
     3  female        S  woman  Southampton   yes
     4    male        S    man  Southampton    no
```

```
[ ]: # select multi-types dataset
     df.select_dtypes(include=['object', 'category', 'number', 'bool']).head()
```

```
[ ]:    survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
     0         0       3    male  22.0      1      0   7.2500        S  Third
     1         1       1  female  38.0      1      0  71.2833        C  First
     2         1       3  female  26.0      0      0   7.9250        S  Third
     3         1       1  female  35.0      1      0  53.1000        S  First
     4         0       3    male  35.0      0      0   8.0500        S  Third

          who  adult_male deck  embark_town alive  alone
     0    man        True  NaN  Southampton    no  False
     1  woman       False    C    Cherbourg   yes  False
     2  woman       False  NaN  Southampton   yes   True
     3  woman       False    C  Southampton   yes  False
     4    man        True  NaN  Southampton    no   True
```

```
[ ]: # select all columns excepts
     df.select_dtypes(exclude=['number']).head()
```

```
[ ]:       sex embarked  class    who  adult_male deck  embark_town alive  alone
     0    male        S  Third    man        True  NaN  Southampton    no  False
     1  female        C  First  woman       False    C    Cherbourg   yes  False
     2  female        S  Third  woman       False  NaN  Southampton   yes   True
     3  female        S  First  woman       False    C  Southampton   yes  False
     4    male        S  Third    man        True  NaN  Southampton    no   True
```

0.0.7 9-Convert string to number

```
[ ]: # create df
     df = pd.DataFrame({'col-A': [1.2,2.2,3,4,5], 'col-B': ['6','7','8','9','10']})
     df
```

```
[ ]:    col-A col-B
     0    1.2     6
     1    2.2     7
     2    3.0     8
     3    4.0     9
     4    5.0    10
```

```
[ ]: df.dtypes
```

[ ]:
```
col-A    float64
col-B     object
dtype: object
```

[ ]:
```
pd.to_numeric(df['col-A'], errors='coerce')
pd.to_numeric(df['col-B'], errors='coerce')
```

[ ]:
```
0     6
1     7
2     8
3     9
4    10
Name: col-B, dtype: int64
```

0.0.8 10-Reduce dataframe size

[ ]:
```
df = sns.load_dataset('titanic')
df.shape
```

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

[ ]:
```
# take sample data
df.sample(frac=0.2).shape
```

[ ]:
```
(178, 15)
```

[ ]:
```
# take some sample
df.sample(n=5).shape
```

[ ]:
```
(5, 15)
```

0.0.9 11- Copy data from clip board

[ ]:
```
# dataset
df = sns.load_dataset('titanic')
df.head(2)
```

[ ]:
```
   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  22.0      1      0   7.2500        S  Third
1         1       1  female  38.0      1      0  71.2833        C  First

     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no  False
1  woman       False    C    Cherbourg   yes  False
```

[ ]:
```
# # read data from clipboard
# df_clipboard = pd.read_clipboard()
# df_clipboard.head(4)
# # write the clipboard data into pc
# df_clipboard.to_csv("clipboard_tips.csv")
```

0.0.10 12-Split dataframe into two subsets

[ ]:
```
import pandas as pd
import numpy as np
import seaborn as sns
# load data
df = sns.load_dataset('titanic')
df.head(2)
```

[ ]:
```
   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  22.0      1      0   7.2500        S  Third
1         1       1  female  38.0      1      0  71.2833        C  First

     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no  False
1  woman       False    C    Cherbourg   yes  False
```

[ ]:
```
# split the data into two parts
split_1  = df.sample(frac=0.5, ignore_index=True)
split_2 = df.drop(split_1.index)
print('split_1:', split_1.shape)
print('split_2:', split_2.shape)
```

```
split_1: (446, 15)
split_2: (445, 15)
```

[ ]:
```
len(df)
```

[ ]:
```
891
```

[ ]:
```
split_1.head()
```

[ ]:
```
   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  28.0      0      0   9.5000        S  Third
1         0       3    male   NaN      0      0   7.7333        Q  Third
2         0       3    male  22.0      0      0   7.7958        S  Third
3         1       3  female  26.0      0      0   7.9250        S  Third
4         1       1  female  30.0      0      0  56.9292        C  First

     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no   True
1    man        True  NaN   Queenstown    no   True
2    man        True  NaN  Southampton    no   True
3  woman       False  NaN  Southampton   yes   True
4  woman       False    E    Cherbourg   yes   True
```

```
[ ]: split_2.head()
```

```
[ ]:     survived  pclass     sex   age  sibsp  parch     fare embarked   class  \
446         1       2  female  13.0      0      1  19.5000        S  Second
447         1       1    male  34.0      0      0  26.5500        S   First
448         1       3  female   5.0      2      1  19.2583        C   Third
449         1       1    male  52.0      0      0  30.5000        S   First
450         0       2    male  36.0      1      2  27.7500        S  Second

       who  adult_male deck  embark_town alive  alone
446  child       False  NaN  Southampton   yes  False
447    man        True  NaN  Southampton   yes   True
448  child       False  NaN    Cherbourg   yes  False
449    man        True    C  Southampton   yes   True
450    man        True  NaN  Southampton    no  False
```

```
[ ]: len(split_1)+len(split_2)
```

```
[ ]: 891
```

0.0.11 13-Join two datasets

```
[ ]: # now combine two datasets
     df1 = pd.concat([split_1, split_2], ignore_index=True)
     df1.head()
```

```
[ ]:   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  28.0      0      0   9.5000        S  Third
1         0       3    male   NaN      0      0   7.7333        Q  Third
2         0       3    male  22.0      0      0   7.7958        S  Third
3         1       3  female  26.0      0      0   7.9250        S  Third
4         1       1  female  30.0      0      0  56.9292        C  First

     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no   True
1    man        True  NaN   Queenstown    no   True
2    man        True  NaN  Southampton    no   True
3  woman       False  NaN  Southampton   yes   True
4  woman       False    E    Cherbourg   yes   True
```

0.0.12 14-Filtering a dataset

```
[ ]: df.head(3)
```

```
[ ]:   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  22.0      1      0   7.2500        S  Third
1         1       1  female  38.0      1      0  71.2833        C  First
2         1       3  female  26.0      0      0   7.9250        S  Third
```

```
     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no  False
1  woman       False    C    Cherbourg   yes  False
2  woman       False  NaN  Southampton   yes   True
```

[ ]:
```
# find unique values in a column
df.sex.unique()
```

[ ]:
```
array(['male', 'female'], dtype=object)
```

[ ]:
```
# take only female data from the main data
df[(df.sex=="male")].head()
```

[ ]:
```
   survived  pclass   sex   age  sibsp  parch     fare embarked  class    who  \
0         0       3  male  22.0      1      0   7.2500        S  Third    man
4         0       3  male  35.0      0      0   8.0500        S  Third    man
5         0       3  male   NaN      0      0   8.4583        Q  Third    man
6         0       1  male  54.0      0      0  51.8625        S  First    man
7         0       3  male   2.0      3      1  21.0750        S  Third  child

   adult_male deck  embark_town alive  alone
0        True  NaN  Southampton    no  False
4        True  NaN  Southampton    no   True
5        True  NaN   Queenstown    no   True
6        True    E  Southampton    no   True
7       False  NaN  Southampton    no  False
```

[ ]:
```
df.columns
```

[ ]:
```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'deck', 'embark_town',
       'alive', 'alone'],
      dtype='object')
```

[ ]:
```
# class object is not working because it matches to pyhton function, so we need␣
↪t rename the class column
df.rename(columns={'class': 'Class'}, inplace=True)
```

[ ]:
```
df.Class.value_counts()
```

[ ]:
```
Class
Third     491
First     216
Second    184
Name: count, dtype: int64
```

```
[ ]: # take specific class data
     df[(df.Class=="First")].head(3)
```

```
[ ]:    survived  pclass     sex   age  sibsp  parch     fare embarked  Class  \
     1         1       1  female  38.0      1      0  71.2833        C  First
     3         1       1  female  35.0      1      0  53.1000        S  First
     6         0       1    male  54.0      0      0  51.8625        S  First

          who  adult_male deck  embark_town alive  alone
     1  woman       False    C    Cherbourg   yes  False
     3  woman       False    C  Southampton   yes  False
     6    man        True    E  Southampton    no   True
```

```
[ ]: # take only those who traveled from southampton
     df[(df.embark_town=="Southampton")].shape
```

```
[ ]: (644, 15)
```

```
[ ]: # select only male data who traveled from southampton data
     df[(df.embark_town=='Southampton') & (df.sex=="male")].head()
```

```
[ ]:     survived  pclass   sex   age  sibsp  parch     fare embarked  Class  \
     0          0       3  male  22.0      1      0   7.2500        S  Third
     4          0       3  male  35.0      0      0   8.0500        S  Third
     6          0       1  male  54.0      0      0  51.8625        S  First
     7          0       3  male   2.0      3      1  21.0750        S  Third
     12         0       3  male  20.0      0      0   8.0500        S  Third

           who  adult_male deck  embark_town alive  alone
     0     man        True  NaN  Southampton    no  False
     4     man        True  NaN  Southampton    no   True
     6     man        True    E  Southampton    no   True
     7   child       False  NaN  Southampton    no  False
     12    man        True  NaN  Southampton    no   True
```

```
[ ]: # if you want only male data who travelled from southampton or queenstown then␣
      ↪show the data
     df[((df.embark_town=="Southampton") | (df.embark_town=='Queenstown')) & (df.
      ↪sex=='male')].head()
```

```
[ ]:    survived  pclass   sex   age  sibsp  parch     fare embarked  Class    who  \
     0         0       3  male  22.0      1      0   7.2500        S  Third    man
     4         0       3  male  35.0      0      0   8.0500        S  Third    man
     5         0       3  male   NaN      0      0   8.4583        Q  Third    man
     6         0       1  male  54.0      0      0  51.8625        S  First    man
     7         0       3  male   2.0      3      1  21.0750        S  Third  child

        adult_male deck  embark_town alive  alone
```

```
0        True  NaN  Southampton    no  False
4        True  NaN  Southampton    no   True
5        True  NaN   Queenstown    no   True
6        True    E  Southampton    no   True
7       False  NaN  Southampton    no  False
```

[ ]:
```
# second method to select the specific category from the column
df[df.embark_town.isin(['Southampton', 'Queenstown']) & (df.sex=='male')].head()
```

[ ]:
```
   survived  pclass   sex   age  sibsp  parch     fare embarked  Class    who  \
0         0       3  male  22.0      1      0   7.2500        S  Third    man   
4         0       3  male  35.0      0      0   8.0500        S  Third    man   
5         0       3  male   NaN      0      0   8.4583        Q  Third    man   
6         0       1  male  54.0      0      0  51.8625        S  First    man   
7         0       3  male   2.0      3      1  21.0750        S  Third  child   

   adult_male deck  embark_town alive  alone  
0        True  NaN  Southampton    no  False  
4        True  NaN  Southampton    no   True  
5        True  NaN   Queenstown    no   True  
6        True    E  Southampton    no   True  
7       False  NaN  Southampton    no  False  
```

[ ]:
```
# select the specific age data
print("greater than 30 age shape is:",df[df.age > 30].shape)
print("less than 30 age shape is:",df[df.age < 30].shape)
```

```
greater than 30 age shape is: (305, 15)
less than 30 age shape is: (384, 15)
```

0.1 15-Filtering by large categories

[ ]:
```
# how many people from different embark town people travelled?
df.embark_town.value_counts()
```

[ ]:
```
embark_town
Southampton    644
Cherbourg      168
Queenstown      77
Name: count, dtype: int64
```

[ ]:
```
df.age.value_counts().nlargest(3)
```

[ ]:
```
age
24.0    30
22.0    27
18.0    26
Name: count, dtype: int64
```

[ ]:
```
df.fare.value_counts().nlargest(3)
```

[ ]:
```
fare
8.0500     43
13.0000    42
7.8958     38
Name: count, dtype: int64
```

[ ]:
```
df.Class.value_counts().nlargest()
```

[ ]:
```
Class
Third     491
First     216
Second    184
Name: count, dtype: int64
```

[ ]:
```
df.who.value_counts().nlargest()
```

[ ]:
```
who
man      537
woman    271
child     83
Name: count, dtype: int64
```

0.2 16-Splitting a string into multiple columns

[ ]:
```
# import libraries
import pandas as pd

# Create dataframe
df = pd.DataFrame({'name':['Muhammad Adnan', 'Haider ali', 'Mubashir hussain',␣
 ↪'Muzammil hussain'],
                   'location':['Dera, Pakistan', 'Faislabad, Pakistan', 'Dera,␣
 ↪Pakistan', 'Dera, Pakistan']})
df
```

[ ]:
```
               name             location
0    Muhammad Adnan       Dera, Pakistan
1        Haider ali  Faislabad, Pakistan
2  Mubashir hussain       Dera, Pakistan
3  Muzammil hussain       Dera, Pakistan
```

[ ]:
```
# Split the name column by using space between two names
df[['First_name', 'Last_name']] = df.name.str.split(' ', expand=True)
df
```

```
[ ]:                name             location First_name Last_name
     0    Muhammad Adnan       Dera, Pakistan   Muhammad     Adnan
     1        Haider ali  Faislabad, Pakistan     Haider       ali
     2  Mubashir hussain       Dera, Pakistan   Mubashir   hussain
     3  Muzammil hussain       Dera, Pakistan   Muzammil   hussain
```

```
[ ]: # split the location column into two
     df[['city', 'country']] = df.location.str.split(expand=True)
     df
```

```
[ ]:                name             location First_name Last_name        city  \
     0    Muhammad Adnan       Dera, Pakistan   Muhammad     Adnan       Dera,
     1        Haider ali  Faislabad, Pakistan     Haider       ali  Faislabad,
     2  Mubashir hussain       Dera, Pakistan   Mubashir   hussain       Dera,
     3  Muzammil hussain       Dera, Pakistan   Muzammil   hussain       Dera,

         country
     0  Pakistan
     1  Pakistan
     2  Pakistan
     3  Pakistan
```

```
[ ]: # drop location column
     df.drop(columns=['location'], inplace=True)
     df.drop(columns=['name'], inplace=True)
```

```
[ ]: # Refined dataframe
     df
```

```
[ ]:   First_name Last_name        city   country
     0   Muhammad     Adnan       Dera,  Pakistan
     1     Haider       ali  Faislabad,  Pakistan
     2   Mubashir   hussain       Dera,  Pakistan
     3   Muzammil   hussain       Dera,  Pakistan
```

0.3 17-Aggregate by multiple groups/function

```
[ ]: # Import libraries
     import pandas as pd
     import seaborn as sns

     # load dataset
     df = sns.load_dataset('titanic')
     df.head(3)
```

```
[ ]:    survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
     0         0       3    male  22.0      1      0   7.2500        S  Third
```

```
1         1       1  female  38.0      1      0  71.2833         C  First
2         1       3  female  26.0      0      0   7.9250         S  Third

     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no  False
1  woman       False    C    Cherbourg   yes  False
2  woman       False  NaN  Southampton   yes   True
```

[ ]:
```
# group by function
df.groupby('who').count()
```

[ ]:
```
       survived  pclass  sex  age  sibsp  parch  fare  embarked  class  \
who
child        83      83   83   83     83     83    83        83     83
man         537     537  537  413    537    537   537       537    537
woman       271     271  271  218    271    271   271       269    271

       adult_male  deck  embark_town  alive  alone
who
child          83    13           83     83     83
man           537    99          537    537    537
woman         271    91          269    271    271
```

[ ]:
```
df['survived'].value_counts()
```

[ ]:
```
survived
0    549
1    342
Name: count, dtype: int64
```

[ ]:
```
df.groupby(['sex', 'who', 'class']).count()
```

```
C:\Users\adnan\AppData\Local\Temp\ipykernel_8564\2509122198.py:1: FutureWarning:
The default of observed=False is deprecated and will be changed to True in a
future version of pandas. Pass observed=False to retain current behavior or
observed=True to adopt the future default and silence this warning.
  df.groupby(['sex', 'who', 'class']).count()
```

[ ]:
```
                     survived  pclass  age  sibsp  parch  fare  embarked  \
sex    who   class
female child First          3       3    3      3      3     3         3
             Second        10      10   10     10     10    10        10
             Third         30      30   30     30     30    30        30
       man   First          0       0    0      0      0     0         0
             Second         0       0    0      0      0     0         0
             Third          0       0    0      0      0     0         0
       woman First         91      91   82     91     91    91        89
             Second        66      66   64     66     66    66        66
             Third         114      114   72     114     114    114         114
male   child First           3        3    3       3       3      3           3
             Second          9        9    9       9       9      9           9
             Third          28       28   28      28      28     28          28
       man   First         119      119   98     119     119    119         119
             Second         99       99   90      99      99     99          99
             Third         319      319  225     319     319    319         319
       woman First           0        0    0       0       0      0           0
             Second          0        0    0       0       0      0           0
             Third           0        0    0       0       0      0           0

                     adult_male  deck  embark_town  alive  alone
sex    who   class
female child First            3     3            3      3      3
             Second          10     1           10     10     10
             Third           30     2           30     30     30
       man   First            0     0            0      0      0
             Second           0     0            0      0      0
             Third            0     0            0      0      0
       woman First           91    78           89     91     91
             Second          66     9           66     66     66
             Third          114     4          114    114    114
male   child First            3     3            3      3      3
             Second           9     3            9      9      9
             Third           28     1           28     28     28
       man   First          119    91          119    119    119
             Second          99     3           99     99     99
             Third          319     5          319    319    319
       woman First            0     0            0      0      0
             Second           0     0            0      0      0
             Third            0     0            0      0      0
```

0.4 18-Select specific rows and columns

[ ]:
```
# select specifi columns
df[['sex', 'class', 'deck']].head()
```

[ ]:
```
      sex  class deck
0    male  Third  NaN
1  female  First    C
2  female  Third  NaN
3  female  First    C
4    male  Third  NaN
```

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

```
[ ]:         survived      pclass         age       sibsp       parch        fare
     count  891.000000  891.000000  714.000000  891.000000  891.000000  891.000000
     mean     0.383838    2.308642   29.699118    0.523008    0.381594   32.204208
     std      0.486592    0.836071   14.526497    1.102743    0.806057   49.693429
     min      0.000000    1.000000    0.420000    0.000000    0.000000    0.000000
     25%      0.000000    2.000000   20.125000    0.000000    0.000000    7.910400
     50%      0.000000    3.000000   28.000000    0.000000    0.000000   14.454200
     75%      1.000000    3.000000   38.000000    1.000000    0.000000   31.000000
     max      1.000000    3.000000   80.000000    8.000000    6.000000  512.329200
```

```
[ ]: # select specific rows
     df.describe().loc[['min', '25%', '50%', '75%', 'max']]
```

```
[ ]:      survived  pclass     age  sibsp  parch      fare
     min       0.0     1.0   0.420    0.0    0.0    0.0000
     25%       0.0     2.0  20.125    0.0    0.0    7.9104
     50%       0.0     3.0  28.000    0.0    0.0   14.4542
     75%       1.0     3.0  38.000    1.0    0.0   31.0000
     max       1.0     3.0  80.000    8.0    6.0  512.3292
```

```
[ ]: # select specific rows with specific columns
     df.describe().loc['min': 'max',['age', 'fare']]
```

```
[ ]:         age      fare
     min   0.420    0.0000
     25%  20.125    7.9104
     50%  28.000   14.4542
     75%  38.000   31.0000
     max  80.000  512.3292
```

0.5 19-Reshape Multi-index Series

```
[ ]: # calculate mean of any column
     df.age.mean()
```

```
[ ]: 29.69911764705882
```

```
[ ]: # calculate multiple column mean
     df[['age', 'fare']].mean()
```

```
[ ]: age     29.699118
     fare    32.204208
     dtype: float64
```

```
[ ]: # calculate mean by using groupby function
     df.groupby('sex').age.mean()
```

[ ]:
```
sex
female    27.915709
male      30.726645
Name: age, dtype: float64
```

[ ]:
```
df.groupby(['sex', 'class']).survived.mean().unstack()
```

```
C:\Users\adnan\AppData\Local\Temp\ipykernel_8564\1574924337.py:1: FutureWarning:
The default of observed=False is deprecated and will be changed to True in a
future version of pandas. Pass observed=False to retain current behavior or
observed=True to adopt the future default and silence this warning.
  df.groupby(['sex', 'class']).survived.mean().unstack()
```

[ ]:
```
class      First    Second     Third
sex
female  0.968085  0.921053  0.500000
male    0.368852  0.157407  0.135447
```

0.6 20-Continous to categorical data conversion

[ ]:
```
# creating bins
df['age_bin'] = pd.cut(df.age, bins=[0, 18, 25, 90], labels=['child',␣
↪'young_adults', 'adults'])
df.head()
```

[ ]:
```
   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  22.0      1      0   7.2500        S  Third
1         1       1  female  38.0      1      0  71.2833        C  First
2         1       3  female  26.0      0      0   7.9250        S  Third
3         1       1  female  35.0      1      0  53.1000        S  First
4         0       3    male  35.0      0      0   8.0500        S  Third

     who  adult_male deck  embark_town alive  alone       age_bin
0    man        True  NaN  Southampton    no  False  young_adults
1  woman       False    C    Cherbourg   yes  False        adults
2  woman       False  NaN  Southampton   yes   True        adults
3  woman       False    C  Southampton   yes  False        adults
4    man        True  NaN  Southampton    no   True        adults
```

[ ]:
```
df['who'].value_counts()
```

[ ]:
```
who
man      537
woman    271
child     83
Name: count, dtype: int64
```

```
[ ]: # maximum child age in the data
     df[df.who=='child'].age.max()
```

```
[ ]: 15.0
```

0.7 Convert one set of values into another one

```
[ ]: # convert categorical column into numericals
     df['sex_num'] = df.sex.map({'male':0, 'female':1})
     df.head()
```

```
[ ]:    survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
     0         0       3    male  22.0      1      0   7.2500        S  Third
     1         1       1  female  38.0      1      0  71.2833        C  First
     2         1       3  female  26.0      0      0   7.9250        S  Third
     3         1       1  female  35.0      1      0  53.1000        S  First
     4         0       3    male  35.0      0      0   8.0500        S  Third

          who  adult_male deck  embark_town alive  alone       age_bin  sex_num
     0    man        True  NaN  Southampton    no  False  young_adults        0
     1  woman       False    C    Cherbourg   yes  False        adults        1
     2  woman       False  NaN  Southampton   yes   True        adults        1
     3  woman       False    C  Southampton   yes  False        adults        1
     4    man        True  NaN  Southampton    no   True        adults        0
```

```
[ ]: # use factorize function to encode
     df['class_num'] = pd.factorize(df['class'])[0]
     df.head()
```

```
[ ]:    survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
     0         0       3    male  22.0      1      0   7.2500        S  Third
     1         1       1  female  38.0      1      0  71.2833        C  First
     2         1       3  female  26.0      0      0   7.9250        S  Third
     3         1       1  female  35.0      1      0  53.1000        S  First
     4         0       3    male  35.0      0      0   8.0500        S  Third

          who  adult_male deck  embark_town alive  alone       age_bin  sex_num  \
     0    man        True  NaN  Southampton    no  False  young_adults        0
     1  woman       False    C    Cherbourg   yes  False        adults        1
     2  woman       False  NaN  Southampton   yes   True        adults        1
     3  woman       False    C  Southampton   yes  False        adults        1
     4    man        True  NaN  Southampton    no   True        adults        0

        class_num
     0          0
     1          1
     2          0
```

```
3          1
4          0
```

0.8 22- Transpose a wide dataframe

[ ]:
```
# Transpose dataset
df.head(10).T
```

[ ]:
```
                        0           1            2            3            4  \
survived                0           1            1            1            0
pclass                  3           1            3            1            3
sex                  male      female       female       female         male
age                  22.0        38.0         26.0         35.0         35.0
sibsp                   1           1            0            1            0
parch                   0           0            0            0            0
fare                 7.25     71.2833        7.925         53.1         8.05
embarked                S           C            S            S            S
class               Third       First        Third        First        Third
who                   man       woman        woman        woman          man
adult_male           True       False        False        False         True
deck                  NaN           C          NaN            C          NaN
embark_town   Southampton   Cherbourg  Southampton  Southampton  Southampton
alive                  no         yes          yes          yes           no
alone               False       False         True        False         True
age_bin      young_adults      adults       adults       adults       adults
sex_num                 0           1            1            1            0
class_num               0           1            0            1            0

                      5            6            7            8          9
survived              0            0            0            1          1
pclass                3            1            3            3          2
sex                male         male         male       female     female
age                 NaN         54.0          2.0         27.0       14.0
sibsp                 0            0            3            0          1
parch                 0            0            1            2          0
fare             8.4583      51.8625       21.075      11.1333    30.0708
embarked              Q            S            S            S          C
class             Third        First        Third        Third     Second
who                 man          man        child        woman      child
adult_male         True         True        False        False      False
deck                NaN            E          NaN          NaN        NaN
embark_town  Queenstown  Southampton  Southampton  Southampton  Cherbourg
alive                no           no           no          yes        yes
alone              True         True        False        False      False
age_bin             NaN       adults        child       adults      child
sex_num               0            0            0            1          1
class_num             0            1            0            0          2
```

```
[ ]: df.describe().T
```

```
[ ]:            count       mean        std   min      25%      50%   75%       max
     survived   891.0   0.383838   0.486592  0.00   0.0000   0.0000   1.0    1.0000
     pclass     891.0   2.308642   0.836071  1.00   2.0000   3.0000   3.0    3.0000
     age        714.0  29.699118  14.526497  0.42  20.1250  28.0000  38.0   80.0000
     sibsp      891.0   0.523008   1.102743  0.00   0.0000   0.0000   1.0    8.0000
     parch      891.0   0.381594   0.806057  0.00   0.0000   0.0000   0.0    6.0000
     fare       891.0  32.204208  49.693429  0.00   7.9104  14.4542  31.0  512.3292
     sex_num    891.0   0.352413   0.477990  0.00   0.0000   0.0000   1.0    1.0000
     class_num  891.0   0.655443   0.799734  0.00   0.0000   0.0000   1.0    2.0000
```

0.9 23-Reshaping a dataframe