1) Pandas Series

Create a Series

```
import numpy as np
import pandas as pd
# 1) creating pandas series from a list
my_data = [10,20,30,40,50]
labels = ['a','b','c','d','e']
pd.Series(data=my_data)
          10
          20
     1
          30
     3
          40
          50
     dtype: int64
pd.Series(data=my_data, index=labels)
          10
          30
     С
     d
          40
          50
     dtype: int64
pd.Series(my_data, labels)
          10
          40
     d
          50
     dtype: int64
# 2) Pandas series using numpy array
arr = np.array(my_data)
     array([10, 20, 30, 40, 50])
pd.Series(data=arr)
     0
          10
     1
          20
     2
          30
          40
          50
     dtype: int64
pd.Series(data=arr,index=labels)
          10
     а
     b
          20
          30
          40
          50
     dtype: int64
# 3) pandas series using a dictionary
d = {'a':100,'b':200, 'c':300, 'd':400}
     {'a': 100, 'b': 200, 'c': 300, 'd': 400}
```

```
pd.Series(d)
     b
          200
          300
     С
          400
     d
     dtype: int64
pd.Series(data=labels)
     0
     1
          b
     2
     3
          d
     dtype: object
pd.Series(data=[sum, len])
          <built-in function sum>
          <bul><built-in function len>
     dtype: object
```

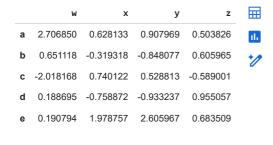
Grab Info from Pandas Series

```
ser1 = pd.Series(data=[1,2,3,4], index=['Red','Green','Blue','Orange'])
ser1
     Red
               1
     Green
               2
    Blue
               3
    Orange
    dtype: int64
ser2 = pd.Series(data=[1,2,5,4], index=['Red','Green','Yellow','Orange'])
ser2
    Red
               1
    Green
               2
    Yellow
               5
    Orange
    dtype: int64
ser2['Green']
     2
ser1['Blue']
     3
ser3 = pd.Series(data=['a','b','c','d','e'])
ser3
     0
         b
     1
     3
         d
     dtype: object
ser3[1]
     'b'
ser3[0:3]
     0
         b
    dtype: object
```

```
ser1
     Red
               1
               2
     Green
     Blue
               3
     Orange
               4
     dtype: int64
ser2
     Red
               1
     Green
               2
     Yellow
               5
     Orange
               4
     dtype: int64
ser1 + ser2
     Blue
               4.0
     Green
     Orange
               8.0
     Red
               2.0
     Yellow
               NaN
     dtype: float64
```

v 2) DataFrames

```
import numpy as np
import pandas as pd
from numpy.random import randn
np.random.seed(101)
df
                                           \blacksquare
                                       z
                               У
        2.706850 0.628133 0.907969
                                 0.503826
                                           th
       0.651118 -0.319318 -0.848077
                                 0.605965
     c -2.018168 0.740122 0.528813 -0.589001
       0.188695 -0.758872 -0.933237 0.955057
       0.190794 1.978757 2.605967 0.683509
# 1) Indexing and Selection
df['w']
        2.706850
    а
        0.651118
    b
       -2.018168
    C
        0.188695
    d
        0.190794
    Name: w, dtype: float64
type(df['w'])
    pandas.core.series.Series
type(df)
    pandas.core.frame.DataFrame
df
```



df[['x','y']]



- **b** -0.319318 -0.848077
- **c** 0.740122 0.528813
- d -0.758872 -0.933237
- e 1.978757 2.605967

type(df[['x','y']])

pandas.core.frame.DataFrame

2) add a new column

df

	W	x	у	z	#
а	2.706850	0.628133	0.907969	0.503826	ıl.
b	0.651118	-0.319318	-0.848077	0.605965	+/
С	-2.018168	0.740122	0.528813	-0.589001	
d	0.188695	-0.758872	-0.933237	0.955057	
е	0.190794	1.978757	2.605967	0.683509	

$$df['new'] = df['w'] + df['x']$$

df

	W	х	у	Z	new	\blacksquare
а	2.706850	0.628133	0.907969	0.503826	3.334983	ıl.
b	0.651118	-0.319318	-0.848077	0.605965	0.331800	+/
С	-2.018168	0.740122	0.528813	-0.589001	-1.278046	_
d	0.188695	-0.758872	-0.933237	0.955057	-0.570177	
е	0.190794	1.978757	2.605967	0.683509	2.169552	

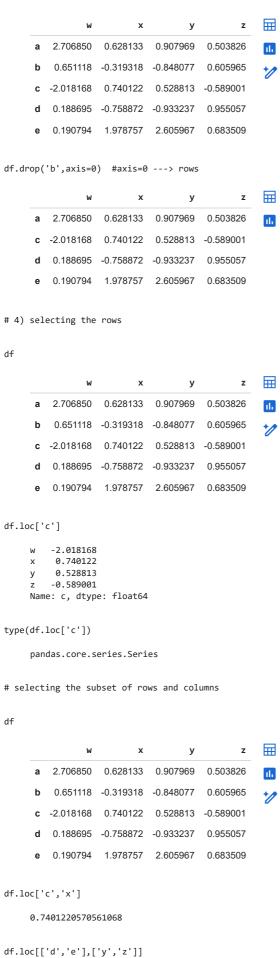
3) removing columns and rows

df.drop(labels='new',axis=1) #axis=1 is for columns and axis=0 is for rows.

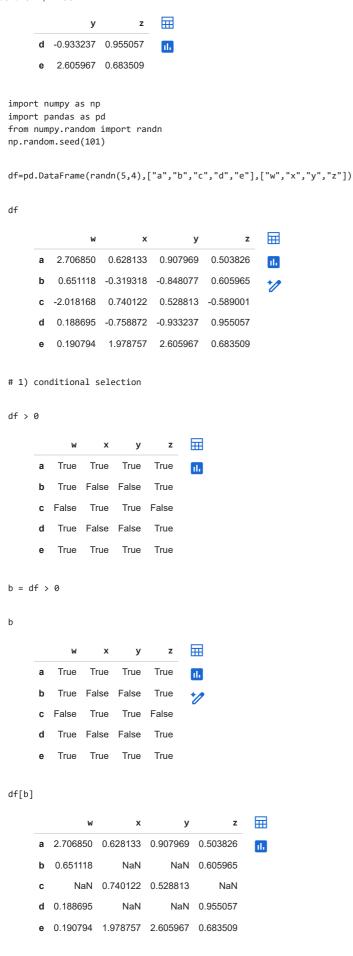
	W	x	У	z	
а	2.706850	0.628133	0.907969	0.503826	ıl.
b	0.651118	-0.319318	-0.848077	0.605965	
С	-2.018168	0.740122	0.528813	-0.589001	
d	0.188695	-0.758872	-0.933237	0.955057	
е	0.190794	1.978757	2.605967	0.683509	

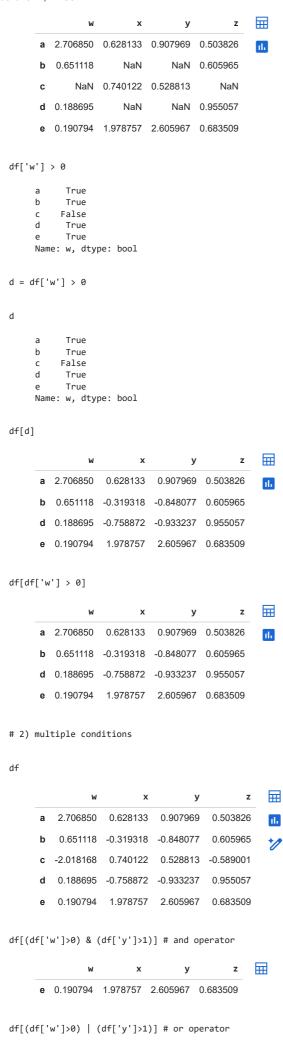
df.drop(labels='new',axis=1,inplace=True) #permanent deletion

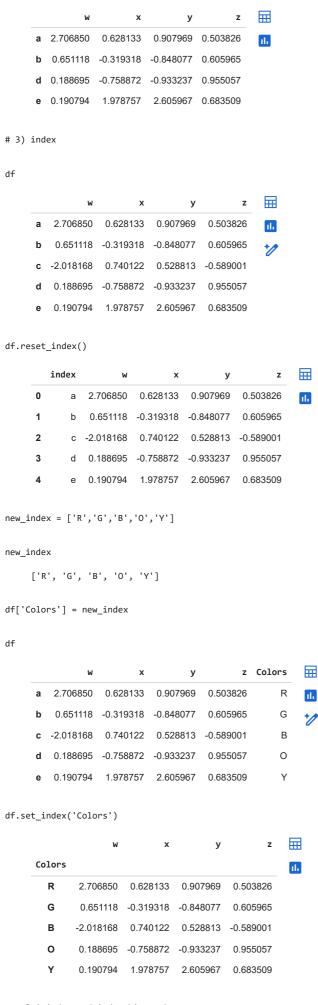
df



df[df > 0]







[#] Multi-index and index hierarchy

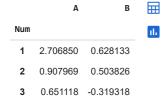
```
import numpy as np
import pandas as pd
from numpy.random import randn
np.random.seed(101)
outside = ['Red','Red','Red','Green','Green','Green']
inside = [1,2,3,1,2,3]
hier_index = list(zip(outside,inside))
hier_index
     [('Red', 1), ('Red', 2), ('Red', 3), ('Green', 1), ('Green', 2), ('Green', 3)]
multi_index = pd.MultiIndex.from_tuples(hier_index, names=['Colors','Numbers'])
multi\_index
     MultiIndex([( 'Red', 1),
                    'Red', 2),
                 ('Green', 1),
                 ('Green', 2),
                 ('Green', 3)],
                names=['Colors', 'Numbers'])
df = pd.DataFrame(data=randn(6,2), index=multi_index, columns=['A','B'])
df
                                             \blacksquare
                              Α
                                        В
      Colors Numbers
       Red
                 1
                       2.706850
                                 0.628133
                       0.907969
                                 0.503826
                 2
                 3
                        0.651118 -0.319318
                       -0.848077 0.605965
      Green
                 1
                 2
                       -2.018168 0.740122
                 3
                       0.528813 -0.589001
df.loc['Red'].loc[1]['A']
     2.706849839399938
df.loc['Green'].loc[2]['B']
     0.7401220570561068
df.index.names
     FrozenList(['Colors', 'Numbers'])
df.index.names = ['Col','Num']
df
                                        Col Num
                                        2.706850 0.628133
       Red
              1
              2
                  0.907969
                             0.503826
              3
                   0.651118 -0.319318
      Green
                  -0.848077
                             0.605965
              2
                  -2.018168 0.740122
              3
                  0.528813 -0.589001
```

cross section

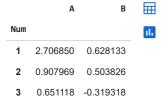
df



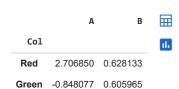
df.loc['Red']



df.xs('Red')



df.xs(1, level='Num')



Missing Data

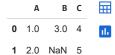
dropna method

df.dropna()

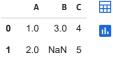
df.dropna(axis=1)



df.dropna(thresh=2, axis=0)



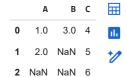
df.dropna(thresh=1, axis=1)



2 NaN NaN 6

fillna method

df



df.fillna(value='New Value')

	Α	В	c	=
0	1.0	3.0	4	ılı
1	2.0	New Value	5	

df['A'].fillna(value=df['A'].mean())

2 New Value New Value 6

```
0 1.0
1 2.0
2 1.5
Name: A, dtype: float64
```

Groupby Method

```
team_data
     {'Company': ['Apple', 'Apple', 'FB', 'FB', 'Google', 'Google'],
'Person': ['Mark', 'Tom', 'John', 'Sara', 'Mia', 'Emma'],
       'Sales': [200, 150, 350, 125, 260, 180]}
df = pd.DataFrame(team_data)
df
         Company Person Sales
                                    \blacksquare
      0
           Apple
                     Mark
                             200
                                    ıl.
      1
                     Tom
                             150
            Apple
      2
              FB
                     John
                             350
      3
              FΒ
                             125
                     Sara
      4
          Google
                      Mia
                             260
      5
          Google
                   Emma
                             180
df.groupby(by='Company')
     <pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f7023e03cd0>
b = df.groupby(by='Company')
b.mean()
     <ipython-input-123-781b9fa94bde>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a fι
       b.mean()
               Sales
                        Company
                         th
                175.0
       Apple
        FΒ
                237.5
                220.0
      Google
b.sum()
     <ipython-input-124-5625bf30d0a3>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a fut
       b.sum()
                Sales
                        Company
                         ılı.
                 350
       Apple
        FB
                 475
                 440
      Google
b.std()
     <ipython-input-125-be35c3b18507>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.std is deprecated. In a fut
       b.std()
                    Sales
                             Company
                              ılı.
                35.355339
       Apple
        FΒ
                159.099026
                56 568542
       Google
b.sum().loc['Apple']
     <ipython-input-126-b28e268a74b0>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a fut
       b.sum().loc['Apple']
     Sales
              350
     Name: Apple, dtype: int64
```

```
df.groupby(by='Company').sum().loc['FB']
     <ipython-input-127-b4cc53764e09>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a fut
       df.groupby(by='Company').sum().loc['FB']
     Sales
             475
     Name: FB, dtype: int64
b.count()
               Person Sales
                                Ħ
      Company
                                th
                    2
                           2
       Apple
        FB
                    2
                           2
                    2
                           2
      Google
b.max()
                                \blacksquare
               Person Sales
      Company
                                ılı.
       Apple
                  Tom
                         200
        FΒ
                 Sara
                         350
      Google
                  Mia
                         260
b.min()
               Person Sales
                                \blacksquare
      Company
                                th
       Apple
                 Mark
                         150
        FΒ
                 John
                         125
                         180
      Google
                Emma
# groupby method with describe function
df.groupby(by='Company').describe()
                                                                              \blacksquare
               Sales
                                               25%
                                                       50%
                                                             75%
               count mean std
                                                                     max
                                                                              11.
      Company
       Apple
                 2.0 175.0
                             35.355339
                                        150.0 162.50 175.0 187.50 200.0
        FΒ
                 2.0 237.5 159.099026 125.0 181.25 237.5 293.75 350.0
                 2.0 220.0
                             56.568542 180.0 200.00 220.0 240.00 260.0
      Google
df.groupby(by='Company').describe().transpose()
                                          FΒ
                                                           \blacksquare
             Company
                          Apple
                                                 Google
      Sales
             count
                        2.000000
                                   2.000000
                                               2.000000
                                                           ıl.
              mean
                      175.000000 237.500000 220.000000
               std
                       35.355339
                                 159.099026
                                              56.568542
               min
                      150.000000
                                 125.000000
                                             180.000000
              25%
                      162.500000
                                 181.250000 200.000000
              50%
                      175.000000
                                 237.500000 220.000000
```

187.500000 293.750000 240.000000

200.000000 350.000000 260.000000

75%

max

```
df.groupby(by='Company').describe().loc['FB']
                       2.000000
     Sales count
                     237.500000
            mean
            std
                     159.099026
                     125.000000
            min
            25%
                     181.250000
            50%
                     237.500000
            75%
                     293.750000
            max
                     350.000000
     Name: FB, dtype: float64
```

Merging, Joining and Concatenating

```
import pandas as pd
```

DataFrames Concatenation

```
df1=pd.DataFrame({'A':['A0','A1','A2','A3'],
                     'B':['B0','B1','B2','B3'],
'C':['C0','C1','C2','C3'],
'D': ['D0','D1','D2','D3']},
                      index = [0, 1, 2, 3])
df2=pd.DataFrame({'A':['A4','A5','A6','A7'],
                     'B':['B4','B5','B6','B7'],
                     'C':['C4','C5','C6','C7'],
'D':['D4','D5','D6','D7']},
                      index = [4, 5, 6, 7])
df3=pd.DataFrame({'A':['A8','A9','A10','A11'],
                     'B':['B8','B9','B10','B11'],
                     'C':['C8','C9','C10','C11'],
'D':['D8','D9','D10','D11']},
                      index = [8, 9, 10, 11])
df1
               В
                    C
                         D
                               \blacksquare
       0 A0 B0 C0 D0
                               ıl.
       1 A1 B1 C1 D1
       2 A2 B2 C2 D2
       3 A3 B3 C3 D3
df2
                               \blacksquare
               В
                    C
                         D
       4 A4 B4 C4 D4
       5 A5 B5 C5 D5
       6 A6 B6 C6 D6
       7 A7 B7 C7 D7
df3
                          c
                                     Ħ
            A8
                  B8
                        C8
                              D8
            Α9
                  B9
                        C9
                              D9
```

10 A10 B10 C10 D1011 A11 B11 C11 D11

pd.concat(objs=[df1,df2,df3])

```
\blacksquare
     Α
           В
                c
                     D
              C0
0
    A0
         B0
                   D0
                          d.
    Α1
         В1
              C1
                    D1
2
    Α2
         B2
              C2
                    D2
3
    АЗ
         В3
              C3
                   D3
    A4
         В4
              C4
                    D4
5
    A5
         B5
              C5
                    D5
6
    Α6
         В6
              C6
                    D6
7
    Α7
         B7
              C7
                    D7
    A8
         B8
              C8
                   D8
8
    Α9
         В9
              C9
                   D9
  A10 B10 C10 D10
10
11 A11 B11
             C11 D11
```

pd.concat(objs=[df1,df2,df3],axis=1)

```
Α
         В
                   D
                       Α
                            В
                                 C
                                     D
                                          Α
                                               В
                                                   c
                                                        D
                                                            n
    A0
        RΩ
             C<sub>0</sub>
                  DO NaN NaN NaN NaN NaN NaN NaN
                                                     NaN
                                                            ıl.
    Α1
        B1
             C1
                  D1 NaN
1
                         NaN NaN NaN NaN NaN
                                                     NaN
2
    A2
        B2
             C2
                  D2
                     NaN
                          NaN
                              NaN
                                   NaN
                                        NaN
                                            NaN
                                                 NaN
                                                     NaN
    А3
        ВЗ
             C3
                  D3 NaN
                          NaN
                              NaN NaN
                                        NaN NaN
                                                NaN
                                                     NaN
   NaN
       NaN NaN NaN
                      A4
                           В4
                                C4
                                    D4
                                        NaN
                                           NaN
                                                NaN
                                                     NaN
                                C5
       NaN NaN
                NaN
                      A5
                           B5
                                    D5
   NaN
                                        NaN
                                            NaN NaN
                                                     NaN
                               C6
   NaN
       NaN
           NaN NaN
                      A6
                           В6
                                    D6
                                        NaN
                                            NaN NaN
                                                     NaN
                               C7
   NaN
       NaN
           NaN
                NaN
                      A7
                           B7
                                    D7
                                       NaN
                                            NaN
                                                 NaN
                                                     NaN
   NaN
       NaN
           NaN
                NaN NaN
                         NaN NaN NaN
                                         A8
                                              B8
                                                  C8
                                                       D8
   NaN
       NaN
            NaN
                 NaN
                     NaN
                         NaN
                              NaN
                                   NaN
                                         A9
                                              В9
                                                  C9
                                                       D9
                                        A10
                                             B10
                                                 C10
                                                      D10
10 NaN NaN
            NaN NaN NaN
                         NaN
                              NaN
                                   NaN
11 NaN NaN NaN NaN NaN
                         NaN NaN NaN
                                        A11
                                             B11
                                                 C11
                                                      D11
```

DataFrames Merging

```
df4=pd.DataFrame({'Key':['K0','K1','K2','K3'],
                    'A':['A0','A1','A2','A3'],
'B':['B0','B1','B2','B3']})
df5=pd.DataFrame({'Key':['K0','K1','K2','K3'],
                      'C':['C0','C1','C2','C3'],
                      'D':['D0','D1','D2','D3']})
df4
         Key
                Α
                    В
                         ⊞
      0
          K0 A0 B0
                         ıl.
          K1 A1 B1
      2
          K2
             A2
                  B2
          K3 A3 B3
```

df5

```
Key C D
     0 K0 C0 D0
        K1 C1 D1
     2 K2 C2 D2
     3 K3 C3 D3
pd.merge(left=df4, right=df5, on='Key')
        Key A B C D
                            Ħ
        K0 A0 B0 C0 D0
        K1 A1 B1 C1 D1
     2 K2 A2 B2 C2 D2
     3 K3 A3 B3 C3 D3
df6 = pd.DataFrame({'key1': ['K0', 'K0', 'K1', 'K2'],
                   'key2': ['K0', 'K1', 'K0', 'K1'],
'A': ['A0', 'A1', 'A2', 'A3'],
'B': ['B0', 'B1', 'B2', 'B3']})
'D': ['D0', 'D1', 'D2', 'D3']})
df6
        key1 key2 A
                           \blacksquare
     0
         K0
              K0 A0 B0
                           ili
         K0
     1
              K1 A1 B1
                           +1
     2
         K1
              K0 A2 B2
     3
         K2
              K1 A3 B3
```

df7



pd.merge(left=df6, right=df7, on=['key1','key2'])

```
        key1
        key2
        A
        B
        C
        D

        0
        K0
        K0
        A0
        B0
        C0
        D0

        1
        K1
        K0
        A2
        B2
        C1
        D1

        2
        K1
        K0
        A2
        B2
        C2
        D2
```

DataFrames Join

df8

```
A B
     K0 A0 B0
                  ılı
     K1 A1 B1
     K2 A2 B2
     K3 A3 B3
df9
                  \blacksquare
          C D
     K0 C0 D0
                  ıl.
     K1 C1 D1
     K2 C2 D2
     K3 C3 D3
df8.join(df9)
          A B C D
     KO AO BO CO DO
     K1 A1 B1 C1 D1
     K2 A2 B2 C2 D2
     K3 A3 B3 C3 D3
#OPERATIONS
d = {'col1':[1,2,3,4],
     'col2':[45,55,65,45],
     'col3':['asd','jkl','qwe','xyz']}
df=pd.DataFrame(d)
                         col1 col2 col3
     0
               45
                   asd
     1
          2
               55
                    jkl
     2
          3
              65
                   qwe
     3
          4
               45
                   xyz
df['col2'].unique()
    array([45, 55, 65])
len(df['col2'].unique())
    3
df['col2'].nunique()
    3
df['col2'].value_counts()
    45
          2
    55
          1
    Name: col2, dtype: int64
#apply method
def times5(x):
 return x*5
```

```
20/01/2024, 11:50
   times5(10)
        50
   df['col1'].sum()
        10
   df
           col1 col2 col3
                             45
                              ıl.
         1
              2
                   55
                         jkl
         2
              3
                   65
                       qwe
         3
              4
                   45
                       xyz
   df['col1'].apply(times5)
             5
        0
            10
        1
        2
            15
        3
            20
        Name: col1, dtype: int64
   df['col3'].apply(len)
        0
            3
        1
            3
        2
            3
            3
        Name: col3, dtype: int64
   df.drop('col1',axis=1)
           col2 col3
                        45
                        th
                   jkl
         1
             55
         2
             65
                 qwe
         3
             45 xyz
   df.columns
        Index(['col1', 'col2', 'col3'], dtype='object')
   df.index
        RangeIndex(start=0, stop=4, step=1)
   df.sort_values('col2')
                             col1 col2 col3
                   45
              1
                       asd
         3
              4
                   45
                        xyz
```

2

55

ikl