■ Pandas

- ● 데이터를 분석 할 때 가장 많이 쓰는 라이브러리
- ● Series 와 Series의 묶음인 DataFrame(Table) 형태
- ● Series
  - 1차원 배열과 비슷하지만 각 데이터의 의미를 표시하는 인덱스 사용 가능

```
series1 = pd.Series(data=[25, 5, 5, 15, 2])
series1
```

```
0    25
1     5
2     5
3    15
4     2
dtype: int64
```

```
series2 = pd.Series([25, 5, 5, 15, 2], index=['서울', '대전', '광주', '부산', '제주'])
series2
```

```
서울    25
대전     5
광주     5
부산    15
제주     2
dtype: int64
```

■ Pandas

● DataFrame

- Series를 묶어놓은 형태로 2차원 행렬 데이터에 인덱스를 붙인 것과 비슷
- 2차원 이므로 Series의 행 방향 인덱스 뿐만 아니라 열 방향 인덱스도 가능

```
data_frame = pd.DataFrame(
    data=[
        {'구':25, '국번':'02'},
        {'구':5, '국번':'042'},
        {'구':5, '국번':'062'},
        {'구':15, '국번':'051'},
        {'구':2, '국번':'064'}
    ], index=['서울', '대전', '광주', '부산', '제주'])
data_frame
```

	구	국번
서울	25	02
대전	5	042
광주	5	062
부산	15	051
제주	2	064

columns

index

```
pd.Series([25, 5, 5, 15, 2],
          index=['서울', '대전', '광주', '부산', '제주'])
```

```
서울    25
대전     5
광주     5
부산    15
제주     2
dtype: int64
```

```
pd.Series(data=['02', '042', '062', '051', '064'],
          index=['서울', '대전', '광주', '부산', '제주'])
```

```
서울     02
대전    042
광주    062
부산    051
제주    064
dtype: object
```

■ Series

● 기본 생성

```
s1 = pd.Series(['100', '90', '80'])
s1
```

```
0    100
1     90
2     80
dtype: object
```

● index 지정

```
s1.index = ['kor', 'math', 'eng']
s1
```

```
kor     100
math     90
eng      80
dtype: object
```

```
s1 = pd.Series(['100', '90', '80'], index=['kor', 'math', 'eng'])
s1
```

```
kor     100
math     90
eng      80
dtype: object
```

■ Series

● Series를 이용하여 DataFrame 생성

```
s1 = pd.Series(['100', '90', '80'],
               index=['kor', 'math', 'eng'])
s1
```

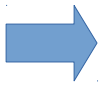
```
kor     100
math     90
eng      80
dtype: object
```

```
s2 = pd.Series(['80', '90', '100'],
               index=['kor', 'math', 'eng'])
s2
```

```
kor      80
math     90
eng     100
dtype: object
```

```
s3 = pd.Series(['90', '100', '80'],
               index=['kor', 'math', 'eng'])
s3
```

```
kor      90
math    100
eng      80
dtype: object
```



```
data_frame = pd.DataFrame(
    data={
        '1번':s1, '2번':s2, '3번':s3
    }
)
data_frame
```

	1번	2번	3번
kor	100	80	90
math	90	90	100
eng	80	100	80

■ DataFrame

● Python Array 사용

```
data1 = [
    [1, 2, 3, 4],
    [5, 6, 7, 8],
    [9, 10, 11, 12],
    [13, 14, 15, 16],
    [17, 18, 19, 20]
]
data_frame1 = pd.DataFrame(data1)
data_frame1
```

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12
3	13	14	15	16
4	17	18	19	20

■ DataFrame

● Numpy Array 사용

```
data2 = np.arange(1, 21).reshape(5, 4)
data_frame2 = pd.DataFrame(data=data2)
data_frame2
```

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12
3	13	14	15	16
4	17	18	19	20

■ DataFrame

● Series 사용 – 1

```
s1 = pd.Series(data=[1, 2, 3], index=['a', 'b', 'c'])
s2 = pd.Series(data=[4, 5, 6], index=['a', 'b', 'c'])
s3 = pd.Series(data=[7, 8, 9], index=['a', 'b', 'c'])

data_frame3 = pd.DataFrame(data = { '가' : s1, '나' : s2, '다' : s3 } )

data_frame3
```

	가	나	다
a	1	4	7
b	2	5	8
c	3	6	9

■ DataFrame

● Series 사용 – 2

```
s1 = pd.Series(data=[1, 2, 3], index=['a', 'b', 'c'])
s2 = pd.Series(data=[4, 5, 6], index=['a', 'b', 'c'])
s3 = pd.Series(data=[7, 8, 9], index=['a', 'b', 'c'])

data_frame3 = pd.DataFrame(data = [s1, s2, s3] )

data_frame3
```

	a	b	c
0	1	2	3
1	4	5	6
2	7	8	9

```
s1 = pd.Series(data=[1, 2, 3], index=['a', 'b', 'c'])
s2 = pd.Series(data=[4, 5, 6], index=['a', 'b', 'c'])
s3 = pd.Series(data=[7, 8, 9], index=['a', 'b', 'c'])

data_frame3 = pd.DataFrame(data=[s1, s2, s3], index=['a', 'b', 'c'])

data_frame3
```

동일한 결과

```
s1 = pd.Series(data=[1, 2, 3], index=['a', 'b', 'c'])
s2 = pd.Series(data=[4, 5, 6], index=['a', 'b', 'c'])
s3 = pd.Series(data=[7, 8, 9], index=['a', 'b', 'c'])

data_frame3 = pd.DataFrame(data=[s1, s2, s3], columns=['a', 'b', 'c'])

data_frame3
```

동일한 결과

■ DataFrame

● Series 사용 – 3

```
s1 = pd.Series(data=[1, 4, 7], index=['a', 'b', 'c'])
s2 = pd.Series(data=[2, 5, 8], index=['a', 'b', 'c'])
s3 = pd.Series(data=[3, 6, 9], index=['a', 'b', 'c'])

data_frame3 = pd.DataFrame(data=[s1, s2, s3],
                           columns=['a', 'b', 'c'], index=['가', '나', '다'])

data_frame3
```

	a	b	c
가	1	4	7
나	2	5	8
다	3	6	9

columns

index

■ DataFrame

● List – Dictionary 사용

```
list = [
    {'name' : 'kim', 'age' : 20, 'job' : 'designer'},
    {'name' : 'lee', 'age' : 21, 'job' : 'programmer'},
    {'name' : 'park', 'age' : 22, 'job' : 'dba'}
]
df = pd.DataFrame(list)
df
```

	age	job	name
0	20	designer	kim
1	21	programmer	lee
2	22	dba	park

생성 시 지정한 Series 순서와 다름

Series 확인

```
df['name']
```

```
0     kim
1     lee
2    park
Name: name, dtype: object
```

여러개의 Series 확인

```
df[['name', 'age']]
```

	name	age
0	kim	20
1	lee	21
2	park	22

Series 순서를 지정하여 확인 후 대입 – 순서 재지정

```
df = df[['name', 'age', 'job']]
df
```

	name	age	job
0	kim	20	designer
1	lee	21	programmer
2	park	22	dba

■ DataFrame

● List – List 사용

```
list = [
    ['kim', 20, 'designer'],
    ['lee', 21, 'programmer'],
    ['park', 22, 'dba']
]
column_name = ['name', 'age', 'job']
pd.DataFrame(list, columns=column_name)
```

	name	age	job
0	kim	20	designer
1	lee	21	programmer
2	park	22	dba

■ DataFrame

● Ordered Dictionary 사용

```
from collections import OrderedDict
```

-List<List<List>>

```
ordered_dict = OrderedDict(
    [
        ['name', ['kim', 'lee', 'park']],
        ['age', [20, 21, 22]],
        ['job', ['designer', 'programmer', 'dba']]
    ]
)
pd.DataFrame(ordered_dict)
```

	name	age	job
0	kim	20	designer
1	lee	21	programmer
2	park	22	dba

■ DataFrame

● Ordered Dictionary 사용

```
from collections import OrderedDict
```

-List<List<Tuple>>

```
ordered_dict = OrderedDict(
    [
        ['name', ('kim', 'lee', 'park')],
        ['age', (20, 21, 22)],
        ['job', ('designer', 'programmer', 'dba')]
    ]
)
pd.DataFrame(ordered_dict)
```

	name	age	job
0	kim	20	designer
1	lee	21	programmer
2	park	22	dba

■ DataFrame

● Ordered Dictionary 사용

```
from collections import OrderedDict
```

-List<Tuple<Tuple>>

```
ordered_dict = OrderedDict(
    [
        ('name',('kim', 'lee', 'park')),
        ('age', (20, 21, 22)),
        ('job', ('designer', 'programmer', 'dba'))
    ]
)
pd.DataFrame(ordered_dict)
```

	name	age	job
0	kim	20	designer
1	lee	21	programmer
2	park	22	dba

■ DataFrame

● Ordered Dictionary 사용

```
from collections import OrderedDict
```

-Tuple<Tuple<Tuple>>

```
ordered_dict = OrderedDict(
    (
        ('name',('kim', 'lee', 'park')),
        ('age', (20, 21, 22)),
        ('job', ('designer', 'programmer', 'dba'))
    )
)
pd.DataFrame(ordered_dict)
```

	name	age	job
0	kim	20	designer
1	lee	21	programmer
2	park	22	dba

■ DataFrame

● 의도하지 않은 결과 예

```
list = [
    {'name' : ['kim', 'lee', 'park']},
    {'age' : [20, 21, 22]},
    {'job' : ['designer', 'programmer', 'dba']},
]
pd.DataFrame(list)
```

	age	job	name
0	NaN	NaN	[kim, lee, park]
1	[20, 21, 22]	NaN	NaN
2	NaN	[designer, programmer, dba]	NaN

```
list = [
    ['name', ['kim', 'lee', 'park']],
    ['age', [20, 21, 22]],
    ['job', ['designer', 'programmer', 'dba']],
]
pd.DataFrame(list)
```

	0	1
0	name	[kim, lee, park]
1	age	[20, 21, 22]
2	job	[designer, programmer, dba]

■ DataFrame

● 생성된 데이터프레임 정보 확인 – info()

```
import pandas as pd

data_frame = pd.DataFrame(
    data=[
        {'구':25, '국번':'02'},
        {'구':5, '국번':'042'},
        {'구':5, '국번':'062'},
        {'구':15, '국번':'051'},
        {'구':2, '국번':'064'}
    ], index=['서울', '대전', '광주', '부산', '제주'])
data_frame
```

	구	국번
서울	25	02
대전	5	042
광주	5	062
부산	15	051
제주	2	064

```
data_frame.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 5 entries, 서울 to 제주
Data columns (total 2 columns):
구     5 non-null int64
국번    5 non-null object
dtypes: int64(1), object(1)
memory usage: 120.0+ bytes
```