### 2.1 [확인학습 1]

```
import pandas as pd
import numpy as np

ex_df = pd.DataFrame(np.arange(12).reshape((3, 4)),
                     index=[['c', 'd', 'd'], [1, 2, 1]],
                     columns=[['Daegu', 'Daejeon', 'Gangneung', 'Daegu'],
                              ['Yellow', 'Yellow', 'Red', 'Blue']])
```

executed in 1.91s, finished 10:40:50 2021-05-14

1. 인덱스에 이름을 지정하시오. 첫번째 레벨은 key1, 두번째 레벨은 key2

```
ex_df.index.names=['key1', 'key2']
```

executed in 15ms, finished 01:09:26 2021-05-12

2. 열 이름을 지정하시오. 첫 번째 열은 city, 두번째 열은 color

```
ex_df.columns.names=['city', 'color']
```

executed in 11ms, finished 01:09:56 2021-05-12

3. Daegu 열만 추출하세요

```
ex_df['Daegu']
```

executed in 24ms, finished 01:11:38 2021-05-12

[7]:

| | color | Yellow | Blue |
|---|---|---|---|
| key1 | key2 | | |
| c | 1 | 0 | 3 |
| d | 2 | 4 | 7 |
| | 1 | 8 | 11 |

4. city 별 평균값을 구하시오.

```
ex_df.mean().unstack()
```

executed in 33ms, finished 09:29:47 2021-05-12

22] :

| color | Blue | Red | Yellow |
|---|---|---|---|
| city | | | |
| Daegu | 7.0 | NaN | 4.0 |
| Daejeon | NaN | NaN | 5.0 |
| Gangneung | NaN | 6.0 | NaN |

5. key2 별로 각 행의 합계를 구하시오.

```
ex_df.sum(level='key2', axis=0)
```

executed in 23ms, finished 01:16:31 2021-05-12

16] :

| city | Daegu | Daejeon | Gangneung | Daegu |
|---|---|---|---|---|
| color | Yellow | Yellow | Red | Blue |
| key2 | | | | |
| 1 | 8 | 10 | 12 | 14 |
| 2 | 4 | 5 | 6 | 7 |

### 4.6 [확인학습 2]

```
data1_dic = {
    'id':['0','1','2','3','4','6','8','11','12','13'],
    'city':['Seoul','Pusan','Daegu','Gangneung','Seoul',
            'Seoul','Pusan','Daegu','Gangneung','Seoul'],
    'birth_year':[1990,1989,1992,1997,1982,1991,1988,1990,1995,1981],
    'name':['Junho','Heejin','Mijung','Minho','Steeve',
            'Mina','Sumi','Minsu','Jinhee','Daeho']
}
data2_dic = {
    'id':['70','80','90','120','150'],
    'city':['Ilsan','Gunpo','Seoul','Changwon','Jeju'],
    'birth_year':[1980,1999,1995,1994,1994],
    'name':['Jinhee','Yeongho','Jongho','Yeonghee','Hyejin']
}
data1=pd.DataFrame(data1_dic)
data2=pd.DataFrame(data2_dic)
```

executed in 19ms, finished 15:59:28 2021-05-14

1. 두 데이터를 INNER JOIN 하시오.

```
pd.merge(data1, data2, how='inner', on='city')
```

executed in 18ms, finished 15:59:40 2021-05-14

]6]:

| | id_x | city | birth_year_x | name_x | id_y | birth_year_y | name_y |
|---|---|---|---|---|---|---|---|
| 0 | 0 | Seoul | 1990 | Junho | 90 | 1995 | Jongho |
| 1 | 4 | Seoul | 1982 | Steeve | 90 | 1995 | Jongho |
| 2 | 6 | Seoul | 1991 | Mina | 90 | 1995 | Jongho |
| 3 | 13 | Seoul | 1981 | Daeho | 90 | 1995 | Jongho |

2. 두 데이터를 FULL JOIN 하시오.

```
pd.merge(data1, data2, how='outer')
```

executed in 44ms, finished 09:40:05 2021-05-12

27]:

| | id | city | birth_year | name |
|---|---|---|---|---|
| 0 | 0 | Seoul | 1990 | Junho |
| 1 | 1 | Pusan | 1989 | Heejin |
| 2 | 2 | Daegu | 1992 | Mijung |
| 3 | 3 | Gangneung | 1997 | Minho |
| 4 | 4 | Seoul | 1982 | Steeve |
| 5 | 6 | Seoul | 1991 | Mina |
| 6 | 8 | Pusan | 1988 | Sumi |
| 7 | 11 | Daegu | 1990 | Minsu |
| 8 | 12 | Gangneung | 1995 | Jinhee |
| 9 | 13 | Seoul | 1981 | Daeho |
| 10 | 70 | Ilsan | 1980 | Jinhee |
| 11 | 80 | Gunpo | 1999 | Yeongho |
| 12 | 90 | Seoul | 1995 | Jongho |
| 13 | 120 | Changwon | 1994 | Yeonghee |
| 14 | 150 | Jeju | 1994 | Hyejin |

3. 두 데이터를 수직방향으로 결합하시오.

```
pd.concat([data1, data2], ignore_index=True, axis=0)
```

executed in 29ms, finished 09:49:25 2021-05-12

[3]:

| | id | city | birth_year | name |
|---|---|---|---|---|
| 0 | 0 | Seoul | 1990 | Junho |
| 1 | 1 | Pusan | 1989 | Heejin |
| 2 | 2 | Daegu | 1992 | Mijung |
| 3 | 3 | Gangneung | 1997 | Minho |
| 4 | 4 | Seoul | 1982 | Steeve |
| 5 | 6 | Seoul | 1991 | Mina |
| 6 | 8 | Pusan | 1988 | Sumi |
| 7 | 11 | Daegu | 1990 | Minsu |
| 8 | 12 | Gangneung | 1995 | Jinhee |
| 9 | 13 | Seoul | 1981 | Daeho |
| 10 | 70 | Ilsan | 1980 | Jinhee |

### 6.1 [확인학습 3]

1. 수학성적 데이터 student-mat.csv를 읽어 들여, 연령(age)에 2를 곱한 새로운 컬럼을 마지막 열에 추가하시오.

```
ds = pd.read_csv('student-mat.csv', sep=';')
tmp = pd.DataFrame(ds['age'].values * 2, columns=['age*2'])
pd.merge(ds, tmp, left_index=True, right_index=True)
```

executed in 108ms, finished 11:06:06 2021-05-14

[5]:

| | freetime | goout | Dalc | Walc | health | absences | G1 | G2 | G3 | age*2 |
|---|---|---|---|---|---|---|---|---|---|---|
| ... | 3 | 4 | 1 | 1 | 3 | 6 | 5 | 6 | 6 | 36 |
| ... | 3 | 3 | 1 | 1 | 3 | 4 | 5 | 5 | 6 | 34 |
| ... | 3 | 2 | 2 | 3 | 3 | 10 | 7 | 8 | 10 | 30 |
| ... | 2 | 2 | 1 | 1 | 5 | 2 | 15 | 14 | 15 | 30 |
| ... | 3 | 2 | 1 | 2 | 5 | 4 | 6 | 10 | 10 | 32 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| ... | 5 | 4 | 4 | 5 | 4 | 11 | 9 | 9 | 9 | 40 |
| ... | 4 | 5 | 3 | 4 | 2 | 3 | 14 | 16 | 16 | 34 |
| ... | 5 | 3 | 3 | 3 | 3 | 3 | 10 | 8 | 7 | 42 |
| ... | 4 | 1 | 3 | 4 | 5 | 0 | 11 | 12 | 10 | 36 |
| ... | 2 | 3 | 3 | 3 | 5 | 5 | 8 | 9 | 9 | 38 |

2. absences 컬럼을 세 개의 구간으로 나누고 각 구간별 학생수를 계산하시오.(구간 분할 간격 absences_bin = [0,1,5,100] )

```
absences_bin = pd.cut(ds['absences'], [0, 1, 5, 100])
ds.groupby(absences_bin)[['absences']].count()
```

executed in 48ms, finished 11:39:02 2021-05-14

:3]:

| | absences |
|---|---|
| absences | |
| (0, 1] | 3 |
| (1, 5] | 131 |
| (5, 100] | 146 |

3. absences 컬럼을 qcut 함수로 세 개의 구간으로 분할하시오.

```
absences_bin = pd.qcut(ds['absences'], 3)
ds.groupby(absences_bin)[['absences']].count()
```

executed in 35ms, finished 15:50:28 2021-05-14

25]:

| absences | absences |
|---|---|
| (-0.001, 2.0] | 183 |
| (2.0, 6.0] | 97 |
| (6.0, 75.0] | 115 |

4. 학교(school) 변수를 기준으로 각 학교의 G1 평균 점수를 구하시오.

```
# ds.groupby(['school', 'G1']).mean().unstack()
ds.pivot_table(index = ['school'], aggfunc = {'G1' : 'mean'})
```

executed in 20ms, finished 15:51:33 2021-05-14

26]:

| school | G1 |
|---|---|
| GP | 10.939828 |
| MS | 10.673913 |

5. 학교(school)와 성별(sex)를 기준으로 G1,G2,G3 평균 점수를 구하시오.

```
ds.pivot_table(index=['school', 'sex'],
               aggfunc={'G1':'mean',
                        'G2':'mean',
                        'G3':'mean'})
```

executed in 47ms, finished 15:53:23 2021-05-14

32]:

| school | sex | G1 | G2 | G3 |
|---|---|---|---|---|
| GP | F | 10.579235 | 10.398907 | 9.972678 |
| | M | 11.337349 | 11.204819 | 11.060241 |
| MS | F | 10.920000 | 10.320000 | 9.920000 |
| | M | 10.380952 | 10.047619 | 9.761905 |

6. 학교(school)와 성별(sex)를 기준으로 G1,G2,G3 최댓값을 구하시오.

```
ds.pivot_table(index = ['school', 'sex'],
               aggfunc = {'G1' : 'max',
                          'G2' : 'max',
                          'G3' : 'max'})
```

executed in 27ms, finished 15:53:15 2021-05-14

31]:

| | | G1 | G2 | G3 |
|---|---|---|---|---|
| school | sex | | | |
| GP | F | 18 | 18 | 19 |
| | M | 19 | 19 | 20 |
| MS | F | 19 | 18 | 19 |
| | M | 15 | 16 | 16 |

### 8.5 [확인학습 4]

수학성적 데이터 student-mat.csv를 이용해 다음 질문에 답하시오.

1. 연령(age)X성별(sex) 기준으로 G1 평균을 계산하고 세로축이 연령(age), 가로축이 성별(sex)인 표를 만드시오.

```
ds = pd.read_csv('student-mat.csv', sep=';')
gph = ds.groupby(['age', 'sex'])['G1'].mean().unstack()
gph
```

executed in 25ms, finished 14:16:32 2021-05-12

08]:

| sex | F | M |
|---|---|---|
| age | | |
| 15 | 10.052632 | 12.250000 |
| 16 | 10.203704 | 11.740000 |
| 17 | 11.103448 | 10.600000 |
| 18 | 10.883721 | 10.538462 |
| 19 | 10.642857 | 9.700000 |
| 20 | 15.000000 | 13.000000 |
| 21 | NaN | 10.000000 |
| 22 | NaN | 6.000000 |

2. 1번에서 만든 표에서 NaN인 행을 모두 제거한 결과를 출력하시오.

```
gph.dropna()
```

executed in 24ms, finished 14:16:34 2021-05-12

09]:

| sex | F | M |
|---|---|---|
| age | | |
| 15 | 10.052632 | 12.250000 |
| 16 | 10.203704 | 11.740000 |
| 17 | 11.103448 | 10.600000 |
| 18 | 10.883721 | 10.538462 |
| 19 | 10.642857 | 9.700000 |
| 20 | 15.000000 | 13.000000 |