10 주차 Homework			
학번	32151671	이름	박민혁

수업자료 10 주차에 있는 확인학습을 모두 풀어서 제출바랍니다.

확인학습에 필요한 데이터셋은 수업자료에 포함되어 있습니다.

[확인 학습 1]

1. 인덱스에 이름을 지정하시오. 첫번째 레벨은 key1, 두번째 레벨은 key2
2. 열 이름을 지정하시오. 첫번째 열은 city, 두번째 열은 color

```
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']])
ex_df.index.names = ['key1', 'key2']
ex_df.columns.names = ['city', 'color']
```

[14]:

city		Daegu	Daejeon	Gangneung	Daegu
color		Yellow	Yellow	Red	Blue
key1	key2				
c	1	0	1	2	3
d	2	4	5	6	7
	1	8	9	10	11

3. Daegu 열만 추출하세요.

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

[15]:

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

4. city 별 평균값을 구하시오.

```
ex_df.mean(level = 'city', axis = 1)
```

[26]:

key1	key2	city	Daegu	Daejeon	Gangneung
c	1		1.5	1.0	2.0
d	2		5.5	5.0	6.0
	1		9.5	9.0	10.0

5. key2 별로 각 행의 합계를 구하시오.

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

[29]:

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

[확인 학습 2]

1. 두 데이터를 INNER JOIN 하시오.

```
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)

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

[61]:

	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')
```

[62]:

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

[89]:

	id	city_x	birth_year_x	name_x	city_y	birth_year_y	name_y
0	0	Seoul	1990.0	Junho	NaN	NaN	NaN
1	1	Pusan	1989.0	Heejin	NaN	NaN	NaN
2	2	Daegu	1992.0	Mijung	NaN	NaN	NaN
3	3	Gangneung	1997.0	Minho	NaN	NaN	NaN
4	4	Seoul	1982.0	Steeve	NaN	NaN	NaN
5	6	Seoul	1991.0	Mina	NaN	NaN	NaN
6	8	Pusan	1988.0	Sumi	NaN	NaN	NaN
7	11	Daegu	1990.0	Minsu	NaN	NaN	NaN
8	12	Gangneung	1995.0	Jinhee	NaN	NaN	NaN
9	13	Seoul	1981.0	Daeho	NaN	NaN	NaN
10	70	NaN	NaN	NaN	Ilsan	1980.0	Jinhee
11	80	NaN	NaN	NaN	Gunpo	1999.0	Yeongho
12	90	NaN	NaN	NaN	Seoul	1995.0	Jongho
13	120	NaN	NaN	NaN	Changwon	1994.0	Yeonghee
14	150	NaN	NaN	NaN	Jeju	1994.0	Hyejin

[확인 학습 3]

1. 수학 성적 데이터 student-mat.csv 를 읽어 들여, 연력(age)에 2 를 곱한 새로운 컬럼을 마지막 열에 추가 하시오.

```
import numpy as np
import pandas as pd
import seaborn as sns

data = pd.read_csv('student-mat.csv', sep = ';')
data['age2'] = data['age'] * 2
```

[111]:

atus	Medu	Fedu	Mjob	Fjob	...	freetime	goout	Dalc	Walc	health	absences	G1	G2	G3	age2
A	4	4	at_home	teacher	...	3	4	1	1	3	6	5	6	6	36
T	1	1	at_home	other	...	3	3	1	1	3	4	5	5	6	34
T	1	1	at_home	other	...	3	2	2	3	3	10	7	8	10	30
T	4	2	health	services	...	2	2	1	1	5	2	15	14	15	30
T	3	3	other	other	...	3	2	1	2	5	4	6	10	10	32

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

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

```
data.groupby(absences_bin)[['absences']].count()
```

[163]:

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

3. absences 컬럼을 qcut 함수로 세 개의 구간으로 분할하시오.

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

[165]:

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.

```
data.pivot_table(index = ['school'], aggfunc = {'G1' : 'mean'})
```

[130]:

school	G1
GP	10.939828
MS	10.673913

5. 학교(school)와 성별(sex)를 기준으로 각 학교의 G1 평균 점수를 구하시오.

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

[157]:

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.

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

[162]:

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

[확인 학습 4]

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

```
data.pivot_table(index = 'age', columns = 'sex', aggfunc = {'G1' : 'mean', 'G2' : 'mean',
'G3' : 'mean'})
```

[176]:

	G1		G2		G3	
sex	F	M	F	M	F	M
age						
15	10.052632	12.250000	9.789474	12.727273	9.552632	12.727273
16	10.203704	11.740000	10.722222	11.680000	10.537037	11.560000
17	11.103448	10.600000	10.879310	9.900000	10.482759	9.975000
18	10.883721	10.538462	9.976744	10.307692	9.325581	9.794872
19	10.642857	9.700000	9.714286	8.600000	8.357143	8.000000
20	15.000000	13.000000	14.000000	13.500000	15.000000	13.500000
21	NaN	10.000000	NaN	8.000000	NaN	7.000000
22	NaN	6.000000	NaN	8.000000	NaN	8.000000

2. 1 번에서 만든 표에서 NaN 인 행을 모두 제거한 결과를 출력하시오.

```
data.pivot_table(index = 'age', columns = 'sex', aggfunc = {'G1' : 'mean', 'G2' : 'mean', 'G3' : 'mean'}).dropna()
```

[181]:

	G1		G2		G3	
sex	F	M	F	M	F	M
age						
15	10.052632	12.250000	9.789474	12.727273	9.552632	12.727273
16	10.203704	11.740000	10.722222	11.680000	10.537037	11.560000
17	11.103448	10.600000	10.879310	9.900000	10.482759	9.975000
18	10.883721	10.538462	9.976744	10.307692	9.325581	9.794872
19	10.642857	9.700000	9.714286	8.600000	8.357143	8.000000
20	15.000000	13.000000	14.000000	13.500000	15.000000	13.500000