1. 필요한 모듈 불러오기

In [1]:
```
import pandas as pd
import numpy as np
```

2. Case, PatientInfo, PatientRoute csv 파일 불러오기

Case.csv는 case / PatientInfo.csv는 patient_info / PatientRoute.csv는 patient_route로 저장해주세요

In [2]:
```
case = pd.read_csv('COVID19/Case.csv')
patient_info = pd.read_csv('COVID19/PatientInfo.csv')
patient_route = pd.read_csv('COVID19/PatientRoute.csv')
```

3. 각 파일의 행과 열의 개수를 구해주는 함수를 아래와 같이 만들어주세요

In [2]:
```
# 아래 셀에서 작성해주세요
check_shape(case)
check_shape(patient_info)
check_shape(patient_route)
```

```
해당 데이터는 81행 8 열입니다
해당 데이터는 2243행 18 열입니다
해당 데이터는 175행 7 열입니다
```

In [3]:
```
def check_shape(data):
    print('해당 데이터는', len(data), '행 ', len(data.columns),'열 입니다.')

check_shape(case)
check_shape(patient_info)
check_shape(patient_route)
```

```
해당 데이터는 81 행  8 열 입니다.
해당 데이터는 2243 행  18 열 입니다.
해당 데이터는 175 행  7 열 입니다.
```

4. patient_info에서 birth_year를 이용하며 age column에 정확한 나이 숫자로 바꿔주세요

In [4]:
```
patient_info.head()
```

Out[4]:

	patient_id	global_num	sex	birth_year	age	country	province	city	disease	infection_cas
0	1000000001	2.0	male	1964.0	50s	Korea	Seoul	Gangseo-gu	NaN	oversea inflo
1	1000000002	5.0	male	1987.0	30s	Korea	Seoul	Jungnang-gu	NaN	oversea inflo
2	1000000003	6.0	male	1964.0	50s	Korea	Seoul	Jongno-gu	NaN	contact wi patie
3	1000000004	7.0	male	1991.0	20s	Korea	Seoul	Mapo-gu	NaN	oversea inflo
4	1000000005	9.0	female	1992.0	20s	Korea	Seoul	Seongbuk-gu	NaN	contact wi patie

In [10]:
```
import datetime
def calculate_difference(difference):
    difference = 2020 - input_integer + 1
    return difference

def calculate_difference():
    print()
```

In [6]:
```
patient_info["age"] = 2020 - patient_info["birth_year"] + 1
patient_info.head()
```

Out[6]:

	patient_id	global_num	sex	birth_year	age	country	province	city	disease	infection_ca
0	1000000001	2.0	male	1964.0	57.0	Korea	Seoul	Gangseo-gu	NaN	overse inflc
1	1000000002	5.0	male	1987.0	34.0	Korea	Seoul	Jungnang-gu	NaN	overse inflc
2	1000000003	6.0	male	1964.0	57.0	Korea	Seoul	Jongno-gu	NaN	contact w patie
3	1000000004	7.0	male	1991.0	30.0	Korea	Seoul	Mapo-gu	NaN	overse inflc
4	1000000005	9.0	female	1992.0	29.0	Korea	Seoul	Seongbuk-gu	NaN	contact w patie

5. 서울과 제주도의 확진환자 성별 평균 나이를 구해주세요

In [3]:
```
# 아래 셀에서 작성해주세요
seoul_jeju_mean_age
```

Out[3]:

		age
province	sex	
Jeju-do	female	36.000000
	male	35.000000
Seoul	female	45.400000
	male	43.090278

In [9]:
```
seoul_jeju_mean_age = patient_info.pivot_table(index = ['province', 'sex'], ag
seoul_jeju_mean_age
```

Out[9]:

		age
province	**sex**	
Busan	female	44.944444
	male	39.680851
Chungcheongbuk-do	female	61.000000
	male	40.250000
Chungcheongnam-do	female	42.740741
	male	35.189189
Daegu	female	53.054054
	male	62.346154
Gangwon-do	female	52.571429
	male	58.200000
Gyeonggi-do	female	45.726708
	male	42.884892
Gyeongsangbuk-do	female	50.700935
	male	43.865672
Gyeongsangnam-do	female	46.238095
	male	41.425000
Incheon	female	43.285714
	male	46.833333
Jeju-do	female	36.000000
	male	35.000000
Jeollabuk-do	female	53.666667
	male	48.250000
Jeollanam-do	female	31.500000
	male	35.000000
Sejong	male	33.000000
Seoul	female	45.400000
	male	43.090278
Ulsan	female	44.529412
	male	40.615385

In [10]:
```
seoul_jeju_mean_age.loc[['Jeju-do', 'Seoul']]
```

Out[10]:

province	sex	age
Jeju-do	female	36.000000
	male	35.000000
Seoul	female	45.400000
	male	43.090278

6. patient_info에서 접촉자수(contact_number)와 나이(age)가 NaN인 값은 지우기

In [12]:
```
# 결측값 있는 행, 열 제거하기
# Delete row with NaN : df.dropna(axis = 0)
# Delete column with NaN : df.dropna(axis = 1)

patient_info.dropna(subset = ['contact_number', 'age'])
patient_info
```

Out[12]:

	patient_id	global_num	sex	birth_year	age	country	province	city	disease
0	1000000001	2.0	male	1964.0	57.0	Korea	Seoul	Gangseo-gu	NaN
1	1000000002	5.0	male	1987.0	34.0	Korea	Seoul	Jungnang-gu	NaN
2	1000000003	6.0	male	1964.0	57.0	Korea	Seoul	Jongno-gu	NaN
3	1000000004	7.0	male	1991.0	30.0	Korea	Seoul	Mapo-gu	NaN
4	1000000005	9.0	female	1992.0	29.0	Korea	Seoul	Seongbuk-gu	NaN
...	...	...	...	...	...	...	...	...	...
2238	6100000085	NaN	male	1990.0	31.0	Korea	Gyeongsangnam-do	Changwon-si	NaN
2239	7000000001	139.0	male	1998.0	23.0	Korea	Jeju-do	Jeju-do	NaN
2240	7000000002	222.0	female	1998.0	23.0	Korea	Jeju-do	Jeju-do	NaN
2241	7000000003	4345.0	female	1972.0	49.0	Korea	Jeju-do	etc	NaN
2242	7000000004	5534.0	male	1974.0	47.0	Korea	Jeju-do	Jeju-do	NaN

2243 rows × 18 columns

7. 위에서 NaN값을 제거한 데이터프레임에서 province와 city를 기준으로 contact_number와 age의 평균을 구해주세요

In [13]:
```
patient_avg = patient_info.pivot_table(index = ['province', 'city'], aggfunc =
patient_avg
```

Out[13]:

		age	contact_number
province	**city**		
Busan	Buk-gu	32.000000	12.000000
	Busanjin-gu	56.727273	16.142857
	Dongnae-gu	37.172414	46.586207
	Gangseo-gu	26.666667	14.000000
	Geumjeong-gu	32.500000	15.250000
...	...	...	...
Ulsan	Buk-gu	36.200000	NaN
	Dong-gu	55.800000	NaN
	Jung-gu	35.666667	NaN
	Nam-gu	44.333333	NaN
	Ulju-gun	26.500000	NaN

139 rows × 2 columns

8. 7번의 결과 데이터 프레임에서 각 province별 평균 감염자수와 평균 나이를 column에 추가하기

In [4]:
```
# 아래 셀에서 작성해주세요
patient_group
```

Out[4]:

		contact_number	age	province_mean_contact_number	province_mea
province	**city**				
Busan	Buk-gu	12.000000	32.000000	26.038454	43.6
	Busanjin-gu	16.142857	51.000000	26.038454	43.6
	Dongnae-gu	46.586207	37.172414	26.038454	43.6
	Gangseo-gu	14.000000	26.666667	26.038454	43.6
	Geumjeong-gu	15.250000	32.500000	26.038454	43.6
	Haeundae-gu	24.437500	43.062500	26.038454	43.6
	Nam-gu	148.500000	50.000000	26.038454	43.6
	Saha-gu	8.000000	38.714286	26.038454	43.6
	Sasang-gu	22.500000	42.000000	26.038454	43.6
	Seo-gu	12.833333	49.333333	26.038454	43.6

In [22]:
```
patient_avg_col.columns = patient_avg.loc['province_mean_age', 'province_mean_c
patient_avg_col
```

```
   2898             except KeyError:
-> 2899                 return self._engine.get_loc(self._maybe_cast_indexer(
key))
   2900         indexer = self.get_indexer([key], method=method, tolerance=to
lerance)
   2901         if indexer.ndim > 1 or indexer.size > 1:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHas
hTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHas
hTable.get_item()

KeyError: 'province_mean_age'
```

9. 경기도 시흥시의 contact_number와 age를 추출해주세요

In [26]:
```
# patient_avg.loc -> patient_group.loc

patient_avg.loc[('Gyeonggi-do', 'Siheung-si'), ('contact_number', 'age')]
```

Out[26]:
```
contact_number    17.333333
age               47.800000
Name: (Gyeonggi-do, Siheung-si), dtype: float64
```

10. 7번의 결과 데이터프레임을 활용해서 province를 입력하면 contact_number가 가장 높은 5개의 city를 출력해주는 함수를 만들어주세요

In [5]:
```
# 아래 셀에서 작성해주세요
print_city('Seoul')
```

```
1 번째 : Gangseo-gu
2 번째 : Songpa-gu
3 번째 : Jongno-gu
4 번째 : Jungnang-gu
5 번째 : Seodaemun-gu
```

In [6]:
```
# 아래 셀에서 작성해주세요
print_city('Busan')
```

```
1 번째 : Nam-gu
2 번째 : Dongnae-gu
3 번째 : Haeundae-gu
4 번째 : Sasang-gu
5 번째 : Busanjin-gu
```

In [27]:
```
patient_avg.sort_values(by = 'contact_number', ascending = False).loc['Seoul'].
```

Out[27]:
```
'Songpa-gu'
```

In [29]:
```
def city_output(city):
    for x in range(5):
        city_name = patient_avg.sort_values(by = 'contact_number', ascending =
        print("%d 번째 : %s" %(x + 1, city_name))
```

```
1 번째 : Gangseo-gu
2 번째 : Songpa-gu
3 번째 : Jongno-gu
4 번째 : Jungnang-gu
5 번째 : Seodaemun-gu
```

In [33]:
```
print('[Seoul]')
city_output('Seoul')
print('\n')
print('[Busan]')
city_output('Busan')
```

```
[Seoul]
1 번째 : Gangseo-gu
2 번째 : Songpa-gu
3 번째 : Jongno-gu
4 번째 : Jungnang-gu
5 번째 : Seodaemun-gu


[Busan]
1 번째 : Nam-gu
2 번째 : Dongnae-gu
3 번째 : Haeundae-gu
4 번째 : Sasang-gu
5 번째 : Busanjin-gu
```

11. patientInfo와 PatientRoute를 patient_id를 기준으로 inner join 해주세요

In [35]:
```
pat_sort_info = patient_info.sort_values(by = 'patient_id')
pat_sort_route = patient_route.sort_values(by = 'patient_id')
patient_inner_id = pd.merge(pat_sort_info, pat_sort_route, how = 'inner')

patient_inner_id
```

Out[35]:

	patient_id	global_num	sex	birth_year	age	country	province	city	disease	infection_
0	1000000002	5.0	male	1987.0	34.0	Korea	Seoul	Jungnang-gu	NaN	over ir
1	1000000002	5.0	male	1987.0	34.0	Korea	Seoul	Jungnang-gu	NaN	over ir
2	1000000002	5.0	male	1987.0	34.0	Korea	Seoul	Jungnang-gu	NaN	over ir
3	1000000002	5.0	male	1987.0	34.0	Korea	Seoul	Jungnang-gu	NaN	over ir
4	1000000003	6.0	male	1964.0	57.0	Korea	Seoul	Jongno-gu	NaN	contact pa
...	...	...	...	...	...	...	...	...	...	
66	2000000011	28.0	female	1989.0	32.0	China	Gyeonggi-do	Goyang-si	NaN	contact pa
67	5000000001	8.0	female	1958.0	63.0	Korea	Jeollabuk-do	Gunsan-si	NaN	over ir
68	5000000001	8.0	female	1958.0	63.0	Korea	Jeollabuk-do	Gunsan-si	NaN	over ir
69	5000000001	8.0	female	1958.0	63.0	Korea	Jeollabuk-do	Gunsan-si	NaN	over ir
70	5000000001	8.0	female	1958.0	63.0	Korea	Jeollabuk-do	Gunsan-si	NaN	over ir

71 rows × 21 columns

12. 위에서 merge한 파일에서 각 column마다 NaN의 개수와 비율을 구하기 위해 아래와 같은 data frame을 만들어주세요

In [7]:
```
# 아래 셀에서 작성해주세요
na
```

Out[7]:

	patient_id	global_num	sex	birth_year	age	country	province	city	disease	infecti
index	0	1	2	3	4	5	6	7	8	
column type	int64	float64	object	float64	float64	object	object	object	object	
null values(num)	0	0	0	0	0	0	0	0	71	
null values(%)	0	0	0	0	0	0	0	0	100	

4 rows × 21 columns

In [ ]:
```
patient_inner_id
```

13. null values(%)가 100인 column은 제거해주세요

```
In [ ]:
```

14.

1: patient_info에서 감염 경우가 contact with patient인 경우

2: case에서 group(집단 감염 여부)가 true인 경우

1,2번 두 csv파일을 province, city를 기준으로 inner join해서

sex(성별)을 기준으로 state(완치(released), 자가격리(isolated) 명수 파악

```
In [8]: # 아래 셀에서 작성해주세요
        state_df
```

Out[8]:

	isolated	released
male	55	20
female	82	22

```
In [ ]: state_df = pd.merge()
```