```
import pandas as pd
pd.set_option('mode.chained_assignment', None)
import numpy as np
data = pd.read_csv("공공보건의료기관현황.csv", index_col = 0, encoding = 'CP949', engine = 'python')
data.head()

addr = pd.DataFrame(data['주소'].apply(lambda v: v.split()[:2]).tolist(), columns =('시도','군구'))
addr.head()

addr['시도'].unique()

addr[addr['시도'] == '창원시']

addr.iloc[27] = ['경상남도', '창원시']
addr.iloc[31] = ['경상남도', '창원시']

addr.iloc[27]
addr.iloc[31]

addr[addr['시도'] == '경산시']
addr.iloc[47] = ['경상북도', '경산시']
addr.iloc[47]

addr[addr['시도'] == '천안시']
addr.iloc[209] = ['충청남도', '천안시']
addr.iloc[210] = ['충청남도', '천안시']
addr.iloc[209]
addr.iloc[210]

addr['시도'].unique()
```

```
addr_aliases = {'경기':'경기도', '경남':'경상남도', '경북':'경상북도',
                    '충북':'충청북도', '서울시':'서울특별시', '부산특별시':
                    '부산광역시', '대전시':'대전광역시', '충남':'충청남도',
                    '전남':'전라남도', '전북':'전라북도'}

addr['시도'] = addr['시도'].apply(lambda v: addr_aliases.get(v, v))
addr['시도'].unique()

addr['군구'].unique()
addr[addr['군구'] == '아란13길']

addr.iloc[75] = ['제주특별자치도', '제주시']

addr['군구'].unique()

addr['시도군구'] = addr.apply(lambda r: r['시도'] + ' ' + r['군구'], axis = 1)
addr.head()   #작업 확인용 출력

addr['count'] = 0
addr.head() #작업 확인용 출력

addr_group = pd.DataFrame(addr.groupby(['시도','군구','시도군구'], as_index = False).count())
addr_group.head()

addr_group = addr_group.set_index("시도군구")
addr_group.head() #작업 확인용 출력

population = pd.read_excel('행정구역_시군구_별__성별_인구수_2.xlsx')
population.head() #작업 확인용 출력

population = population.rename(columns = {'행정구역(시군구)별(1)': '시도', '행정구역(시군구)별(2)': '군구'})
population.head() #작업 확인용 출력
```

```
for element in range(0,len(population)):
    population['군구'][element] = population['군구'][element].strip()

population['시도군구'] = population.apply(lambda r: r['시도'] + ' ' + r['군구'], axis = 1)
population.head()    #작업 확인용 출력

population = population[population.군구 != '소계']

population = population.set_index("시도군구")
population.head() #작업 확인용 출력

addr_population_merge = pd.merge(addr_group,population, how = 'inner', left_index = True, right_index = True)
addr_population_merge.head() #작업 확인용 출력

local_MC_Population = addr_population_merge[['시도_x', '군구_x', 'count', '총인구수 (명)']]
local_MC_Population.head() #작업 확인용 출력

local_MC_Population = local_MC_Population.rename(columns = {'시도_x': '시도', '군구_x': '군구','총인구수 (명)': '인구수'})
MC_count = local_MC_Population['count']
local_MC_Population['MC_ratio'] = MC_count.div(local_MC_Population['인구수'], axis = 0)*100000
local_MC_Population.head() #작업 확인용 출력
```

```
from matplotlib import pyplot as plt
from matplotlib import rcParams, style
style.use('ggplot')
from matplotlib import font_manager, rc
font_name = font_manager.FontProperties(fname = "c:/Windows/Fonts/malgun.ttf").get_name()
rc('font', family = font_name)

MC_ratio = local_MC_Population[['count']]
MC_ratio = MC_ratio.sort_values('count', ascending = False)
plt.rcParams["figure.figsize"] = (25, 5)
MC_ratio.plot(kind = 'bar', rot = 90)
plt.show()

MC_ratio = local_MC_Population[['MC_ratio']]
MC_ratio = MC_ratio.sort_values('MC_ratio', ascending = False)
plt.rcParams["figure.figsize"] = (25, 5)
MC_ratio.plot(kind = 'bar', rot = 90)
plt.show()


import os
path = os.getcwd()

data_draw_korea = pd.read_csv(path+'\\data_draw_korea.csv', index_col = 0, encoding = 'UTF-8', engine = 'python')
data_draw_korea.head() #작업 확인용 출력

data_draw_korea['시도군구'] = data_draw_korea.apply(lambda r: r['광역시도'] + ' ' + r['행정구역'], axis = 1)

data_draw_korea = data_draw_korea.set_index("시도군구")
data_draw_korea.head()

data_draw_korea_MC_Population_all = pd.merge(data_draw_korea,local_MC_Population, how = 'outer', left_index = True, right_index = True)
data_draw_korea_MC_Population_all.head()
```

```
BORDER_LINES = [
 [(3, 2), (5, 2), (5, 3), (9, 3), (9, 1)], # 인천
 [(2, 5), (3, 5), (3, 4), (8, 4), (8, 7), (7, 7), (7, 9), (4, 9),(4, 7), (1, 7)], # 서울
 [(1, 6), (1, 9), (3, 9), (3, 10), (8, 10), (8, 9),(9, 9), (9, 8), (10, 8), (10, 5), (9, 5), (9, 3)], # 경기도
 [(9, 12), (9, 10), (8, 10)], # 강원도
 [(10, 5), (11, 5), (11, 4), (12, 4), (12, 5), (13, 5), (13, 4), (14, 4), (14, 2)], #충청남도
 [(11, 5), (12, 5), (12, 6), (15, 6), (15, 7), (13, 7), (13, 8), (11, 8), (11, 9), (10, 9), (10, 8)], # 충청북도
 [(14, 4), (15, 4), (15, 6)], # 대전시
 [(14, 7), (14, 9), (13, 9), (13, 11), (13, 13)], # 경상북도
 [(14, 8), (16, 8), (16, 10), (15, 10), (15, 11), (14, 11), (14, 12), (13, 12)], # 대구시
 [(15, 11), (16, 11), (16, 13)], # 울산시
 [(17, 1), (17, 3), (18, 3), (18, 6), (15, 6)], # 전라북도
 [(19, 2), (19, 4), (21, 4), (21, 3), (22, 3), (22, 2), (19, 2)], #광주시
 [(18, 5), (20, 5), (20, 6)], # 전라남도
 [(16, 9), (18, 9), (18, 8), (19, 8), (19, 9), (20, 9), (20, 10)], #부산시
]
```

```
def draw_blockMap(blockedMap, targetData, title, color ):
    whitelabelmin = (max(blockedMap[targetData]) - min(blockedMap[targetData])) * 0.25 + min(blockedMap[targetData])
    datalabel = targetData
    vmin = min(blockedMap[targetData])
    vmax = max(blockedMap[targetData])
    mapdata = blockedMap.pivot(index = 'y', columns = 'x', values = targetData)
    masked_mapdata = np.ma.masked_where(np.isnan(mapdata), mapdata)
    plt.figure(figsize = (8, 13))
    plt.title(title)
    plt.pcolor(masked_mapdata, vmin = vmin, vmax = vmax, cmap = color, edgecolor = '#aaaaaa', linewidth = 0.5)
    #지역 이름 표시
    for idx, row in blockedMap.iterrows():
        annocolor = 'white' if row[targetData] > whitelabelmin else 'black'
        #광역시는 구 이름이 겹치는 경우가 많아서 시단위 이름도 같이 표시
        if row['광역시도'].endswith('시') and not row['광역시도'].startswith('세종'):
            dispname = '{}\n{}'.format(row['광역시도'][:2], row['행정구역'][:-1])
            if len(row['행정구역']) <= 2:
                dispname += row['행정구역'][-1]
        else:
            dispname = row['행정구역'][:-1]
        #서대문구, 서귀포시 같이 이름이 3자 이상이면 작은 글자로 표시
        if len(dispname.splitlines()[-1]) >= 3:
            fontsize, linespacing = 9.5, 1.5
        else:
            fontsize, linespacing = 11, 1.2
        plt.annotate(dispname, (row['x']+0.5, row['y']+0.5), weight = 'bold', fontsize = fontsize, ha = 'center', va = 'center', color = annocolor, linespacing = linespacing)

    #시도 경계를 그린다.
    for path in BORDER_LINES:
        ys, xs = zip(*path)
        plt.plot(xs, ys, c = 'black', lw = 4)

    plt.gca().invert_yaxis()
    #plt.gca().set_aspect(1)
    plt.axis('off')

    cb = plt.colorbar(shrink = 1, aspect = 10)
    cb.set_label(datalabel)
    plt.tight_layout()
    plt.savefig('blockMap_' + targetData + '.png')
    plt.show()
```

```
draw_blockMap(data_draw_korea_MC_Population_all, 'count', '행정구역별 공공보건의료기관 수', 'Blues')
draw_blockMap(data_draw_korea_MC_Population_all, 'MC_ratio', '행정구역별 인구수 대비 공공보건의료기관 비율', 'Reds')
```