4 장 공급을 통해 부동산 살펴보기

1. 공급량이 부동산 가격에 미치는 영향

수요가 시장에서 특정 재화를 사람들이 원하는 정도라고 정의 한다면, 공급은 시장에 제공되는 재화의 양이라고 정의할 수 있다.

2. 부동산 공급 측정을 위한 데이터

살펴볼 데이터는 '인허가' 데이터이다. 인허가 데이터는 '부동산 통계정보 시스템 (https://www.r-one.co.kr/)'이라는 사이트에서 내려 받을 수 있다.

'주택건설인허가'로 검색

다운 로드 받은 파일 '주택건설인허가실적.xlsx'을 데이터 폴더에 복사한다.

3. 파이썬으로 공급 데이터 분석

① 인허가 데이터를 읽어와 데이터프레임으로 정리

```
# [예제 3.21] 인허가 엑셀 데이터를 read_excel 로 읽어오기

import pandas as pd

permission_path = r'데이터₩주택건설인허가실적.xlsx'
pd.read_excel(permission_path)
```

	* 통계표명 :	주택건설인허가실적	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	...	Unnamed: 153	Unnamed: 154	Unnamed: 155	Un
0	* 수록기간 :	2007년 01월 ~ 2020년 06월	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	
1	* 조회기간 :	2007년 01월 ~ 2020년 06월	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	
2	* 출처 :	국토교통부	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	

	지역	2007년 01월	2007년 02월	2007년 03월	2007년 04월	2007년 05월	2007년 06월	2007년 07월	2007년 08월	2007년 09월	...	2019년 09월	2019년 10월	2019년 11월	2019
9															
10	전국	12038	17751	20038	19186	30593	30830	45848	48461	55745	...	31271	39757	33368	1
11	서울	1530	794	1888	1963	1866	1893	7338	11719	13182	...	2661	2987	4275	
12	부산	455	1099	2121	364	897	235	9678	6688	5079	...	1692	1383	2145	
13	대구	818	2731	1047	514	575	887	439	754	2305	...	2152	3186	4148	
14	인천	69	116	930	3337	211	5349	1207	1401	4464	...	1485	3732	1326	
15	광주	1022	1178	1347	226	2071	139	1559	478	1117	...	1066	431	1526	
16	대전	55	34	65	81	511	427	54	51	46	...	39	2391	2	
17	울산	183	1173	685	2281	244	1787	4248	1506	4007	...	27	57	154	

데이터는 11 행부터 시작한다. read_excel 함수에 skip_row=10 이라는 옵션을 추가한다.

```
# 인허가 엑셀 데이터를 read_excel 로 읽어오기

permission_raw = pd.read_excel(permission_path, skiprows=10, index_col=0)
```

데이터프레임에서 날짜는 인덱스에 설정하기 위해 칼럼과 행을 바꾼다. permission_raw 변수의 T 명령어가 칼럼과 행을 바꾼다.

```
# permission_df 의 행과 열을 바꾸기

transposed_permission = permission_raw.T
```

```
transposed_permission
```

지 역	전국	서울	부산	대구	인천	광주	대전	울산	세종	경기	강원	충북	충남	전북	전남	경북	경남	제주
2007년 01월	12038	1530	455	818	69	1022	55	183	-	2647	452	188	1004	888	206	698	1691	132
2007년 02월	17751	794	1099	2731	116	1178	34	1173	-	2870	190	305	1306	602	494	2244	2515	100
2007년 03월	20038	1888	2121	1047	930	1347	65	685	-	3188	389	694	1890	663	391	2314	2321	105
2007년 04월	19186	1963	364	514	3337	226	81	2281	-	3987	1154	464	701	474	1219	1224	1049	148
2007년 05월	30593	1866	897	575	211	2071	511	244	-	13168	513	680	3087	311	353	850	5178	78
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2020년 02월	37980	5651	394	1222	2764	404	926	28	230	17801	502	229	3321	869	439	385	2520	295
2020년 03월	33648	4617	1091	2185	457	1195	2172	1023	28	11368	1062	370	2668	555	2116	1065	929	747
2020년 04월	31884	4340	1194	3206	368	1205	475	73	1024	9423	1611	290	1014	1050	941	2010	3303	357
2020년 05월	28279	4124	717	824	1479	1439	72	47	69	10135	1834	326	2190	274	3354	486	511	398
2020년 06월	33079	3659	4793	3042	1945	760	3198	683	37	9651	1404	290	918	842	670	506	405	276

Pandas 는 년이나 월을 인식하지 못한다. 2007 년 01 월을 알지 못한다. 그러나 '2007.1.1'이나 '2007.1'은 년.월.일로 인식한다. 년도 4 글자와 월 2 글자 사이에 '.'을 추가한다.

```
# 인덱스를 '연도.날짜' 형식으로 바꾸기

new_index = []

for old_date in transposed_permission.index:
    temp_list = old_date.split(' ')
    new_index.append(temp_list[0][:4] + '.' + temp_list[1][:2])
```

```
# 인덱스를 새로 설정하고 데이터프레임 완성하기

transposed_permission.index = pd.to_datetime(new_index)
transposed_permission.columns.name = None
```

```
transposed_permission
```

	전국	서울	부산	대구	인천	광주	대전	울산	세종	경기	강원	충북	충남	전북	전남	경북	경남	제주
2007-01-01	12038	1530	455	818	69	1022	55	183	-	2647	452	188	1004	888	206	698	1691	132
2007-02-01	17751	794	1099	2731	116	1178	34	1173	-	2870	190	305	1306	602	494	2244	2515	100
2007-03-01	20038	1888	2121	1047	930	1347	65	685	-	3188	389	694	1890	663	391	2314	2321	105
2007-04-01	19186	1963	364	514	3337	226	81	2281	-	3987	1154	464	701	474	1219	1224	1049	148
2007-05-01	30593	1866	897	575	211	2071	511	244	-	13168	513	680	3087	311	353	850	5178	78
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2020-02-01	37980	5651	394	1222	2764	404	926	28	230	17801	502	229	3321	869	439	385	2520	295
2020-03-01	33648	4617	1091	2185	457	1195	2172	1023	28	11368	1062	370	2668	555	2116	1065	929	747
2020-04-01	31884	4340	1194	3206	368	1205	475	73	1024	9423	1611	290	1014	1050	941	2010	3303	357
2020-05-01	28279	4124	717	824	1479	1439	72	47	69	10135	1834	326	2190	274	3354	486	511	398
2020-06-01	33079	3659	4793	3042	1945	760	3198	683	37	9651	1404	290	918	842	670	506	405	276

입력으로 인허가 데이터의 위치와 파일 이름 정보가 담긴 문자열 데이터를 path 로 받은 다음 데이터프레임으로 완성해서 반환하는 함수를 만든다.

```
# 인허가 데이터를 데이터프레임으로 변환하는 함수 정의

def permission_preprocessing(path):
    permission_raw = pd.read_excel(path, skiprows=10, index_col=0)
    transposed_permission = permission_raw.T
    new_index = []

    for old_date in transposed_permission.index:
        temp_list = old_date.split(' ')
        new_index.append(temp_list[0][:4] + '.' + temp_list[1][:2])

    transposed_permission.index = pd.to_datetime(new_index)
    transposed_permission.columns.name = None

    return transposed_permission
```

② 함수들을 활용해서 인허가, 매매가 지수, 전세가 지수 데이터프레임 만들기

매매가, 전세가 지수를 데이터프레임으로 가져온다. KBpriceindex_preprocessing() 함수를 활용한다.

```
# KBpriceindex_preprocessing 함수 가져오기

import xlwings as xw

def KBpriceindex_preprocessing(path, data_type):
    # path : KB 데이터 엑셀 파일의 디렉토리 (문자열)
```

```
    # data_type : '매매종합', '매매 APT', '매매연립', '매매단독', '전세종합', '전세 APT', '전세연립', '전세단독' 중 하나

    wb = xw.Book(path)
    sheet = wb.sheets[data_type]
    row_num = sheet.range(1,1).end('down').end('down').end('down').row
    data_range = 'A2:GE' + str(row_num)
    raw_data = sheet[data_range].options(pd.DataFrame, index=False, header=True).value

    bignames = '서울 대구 부산 대전 광주 인천 울산 세종 경기 강원 충북 충남 전북 전남 경북 경남 제주도 6 개광역시 5 개광역시 수도권 기타지방 구분 전국'
    bigname_list = bignames.split(' ')
    big_col = list(raw_data.columns)
    small_col = list(raw_data.iloc[0])

    for num, gu_data in enumerate(small_col):
        if gu_data == None:
            small_col[num] = big_col[num]

        check = num
        while True:
            if big_col[check] in bigname_list:
                big_col[num] = big_col[check]
                break
            else:
                check = check - 1

    big_col[129] = '경기'
    big_col[130] = '경기'
    small_col[185] = '서귀포'

    raw_data.columns = [big_col, small_col]
    new_col_data = raw_data.drop([0,1])

    index_list = list(new_col_data['구분']['구분'])

    new_index = []

    for num, raw_index in enumerate(index_list):
        temp = str(raw_index).split('.')
        if int(temp[0]) > 12 :
            if len(temp[0]) == 2:
```

```
                new_index.append('19' + temp[0] + '.' + temp[1])
            else:
                new_index.append(temp[0] + '.' + temp[1])
        else:
            new_index.append(new_index[num-1].split('.')[0] + '.' + temp[0])

    new_col_data.set_index(pd.to_datetime(new_index), inplace=True)
    cleaned_data  = new_col_data.drop(('구분', '구분'), axis=1)
    return cleaned_data
```

```
# 앞에서 정의한 함수들을 이용해 데이터 전처리하고 데이터프레임으로 가져오기

permission_path  = r'데이터₩주택건설인허가실적.xlsx'
permission = permission_preprocessing(permission_path)
kb_path = r'데이터₩★(월간)KB 주택가격동향_시계열(2020.07).xlsx'
price_index = KBpriceindex_preprocessing(kb_path, '매매종합')
jun_index   = KBpriceindex_preprocessing(kb_path, '전세종합')
```

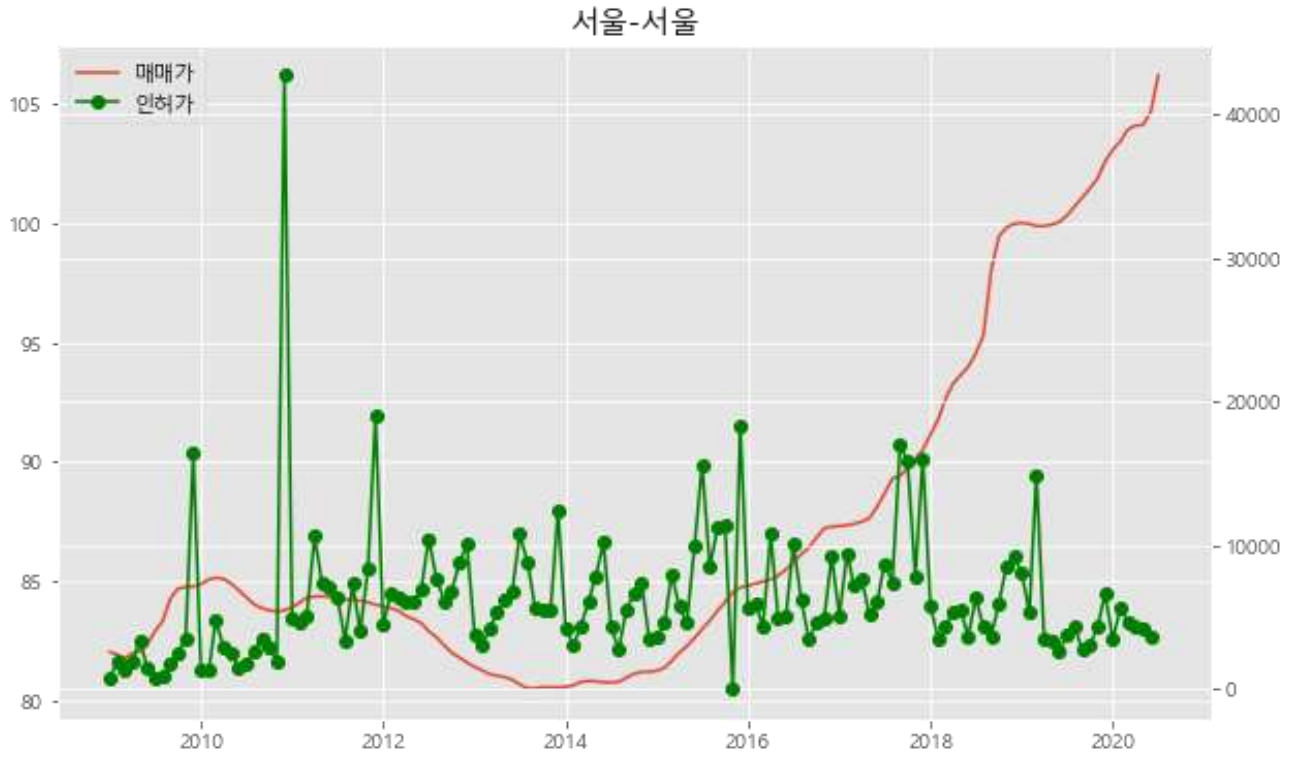
```
# 그래프를 위한 설정

import matplotlib.pyplot as plt
from matplotlib import font_manager, rc
from matplotlib import style
style.use('ggplot')
%matplotlib inline

font_name = font_manager.FontProperties(fname="c:/Windows/Fonts/malgun.ttf").get_name()
rc('font', family=font_name)
# 맥 OS 인 경우 위 두 줄을 입력하지 말고 아래 코드를 입력하세요
# rc('font', family='AppleGothic')
plt.rcParams['axes.unicode_minus'] = False
```

③ 매매가 지수와 인허가 데이터를 함께 그래프로 나타낸다.

매매가 지수와 인허가 데이터의 그래프를 함께 그려야 한다. 매매가 지수와 인허가 데이터가 하나의 y 축을 공유하면 두 데이터 간의 단위 차이가 크게 나므로 올바른 그래프가 그려지지 못한다. 이럴 때 2 개의 y 축을 만들어 각각 다른 축에 매매가 지수의 데이터와 인허가 데이터의 값을 표시해야 한다. 이렇게 두 개의 y 축을 가진 그래프를 그린다.

```
# 서울의 인허가와 매매가 지수의 움직임을 그래프로 나타내기
```

```
plt.figure(figsize=(10,6))
ax = plt.subplot()
ax2 = ax.twinx()

si = '서울'
gu = '서울'

plt.title(si + '-' + gu)
ln1 = ax.plot(price_index[si][gu]['2009-1':], label='매매가')
ln2 = ax2.plot(permission[si]['2009-1':], label='인허가', color='green',marker="o")
lns = ln1 +ln2
labs = [l.get_label() for l in lns]
ax.legend(lns, labs, loc='upper left')

plt.show()
```



④ 인허가 데이터를 1 년 단위로 묶어서 2 년 앞으로 옮기기

인허가 데이터를 월별 선형 그래프로 나타냈더니 상승이나 하락의 추세를 파악하기 어렵다. 이를 년으로 묶어서 살펴본다. 그리고 인허가 데이터가 실제로 부동산 시장에 반영되는 시점은 2 년에서 3 년뒤이다. 그래서 그래프를 2 년 정도 옮긴다.

Pandas 의 groupby() 함수가 데이터프레임을 그룹별로 분류하는 역할을 한다. 연도별 총 인허가량을 보기 위해 sum() 함수를 사용한다.

```
# 인허가 데이터 연도별로

year_permission = permission.groupby(permission.index.year).sum()
```

	전국	서울	부산	대구	인천	광주	대전	울산	경기	강원	충북	충남	전북	전남	경북	경남	제주
2007	555792	62842	41254	18174	41571	13088	11180	24507	198138	10677	19983	29317	11842	15255	24285	31503	2176
2008	371285	48417	13594	22880	33632	3945	14556	5897	115531	13235	10014	21657	12063	10849	15881	24375	4759
2009	381787	36090	6506	6645	59519	5024	1849	6728	159549	12312	11537	22860	11634	8984	13316	17119	2115
2010	386542	69190	18331	4724	37477	4487	4034	4904	143551	9312	8504	15331	12299	17599	13684	18234	4881
2011	549594	88060	37256	12462	35905	16059	19736	13146	148191	12989	18010	46794	16117	15576	16936	39339	13018
2012	586884	86123	42333	13012	32132	19584	6708	9751	151035	12156	24773	44450	24288	22222	25713	44760	10256
2013	440116	77621	29922	18078	18907	8454	5180	5344	96082	12964	19267	32343	13179	20061	23878	34683	6309
2014	515251	65249	17210	19079	13583	11056	5073	12502	163057	12977	16391	35564	13768	17628	41438	49424	8805
2015	765328	101235	33535	27118	30590	14673	7987	12459	276948	18868	31125	40311	22552	15631	53046	45325	18690
2016	726048	74739	36664	23169	22186	22796	13509	16325	244237	29489	29516	31800	28737	20983	36551	61124	21596
2017	653441	113131	47159	31378	22689	20326	9953	12747	185582	29497	30463	25301	17224	20439	25105	38952	14163
2018	554136	65751	34352	35444	39375	14999	6520	12759	174971	26297	27895	26131	13019	16070	25428	25691	7372
2019	487975	62272	17237	27725	44530	19174	17523	5919	165424	19366	11463	26951	10352	19406	11727	17887	5722
2020	188848	25808	8445	11405	8239	5388	8283	4589	65948	7158	2249	10392	4610	9485	4704	7931	2454

2 년 뒤로 옮기기 위해 year_permission 데이터프레임 전체를 두 행씩 밑으로 내리기 위해 shift() 함수를 사용한다. 인덱스를 날짜로 만드는 코드는 단순히 연도만 2008, 2009, ...과 같이 존재하면 파이썬이 시간 형식으로 인식하지 못해서 이를 '2008-6-1'과 같이 시간 데이터로 바꿔줘야 한다.

```
# 인허가 데이터를 2 년 뒤로 옮기기

modified_permission = year_permission.shift(2)
temp = []
for year in modified_permission.index:
    temp.append(str(year) + '-6-1')
modified_permission.index = pd.to_datetime(temp)
```

```
modified_permission
```

	전국	서울	부산	대구	인천	광주	대전	울산	경기	강원	충북	충남	전북	전남	경북	경남
2007-06-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2008-06-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2009-06-01	555792.0	62842.0	41254.0	18174.0	41571.0	13088.0	11180.0	24507.0	198138.0	10677.0	19983.0	29317.0	11842.0	15255.0	24285.0	31503.0
2010-06-01	371285.0	48417.0	13594.0	22880.0	33632.0	3945.0	14556.0	5897.0	115531.0	13235.0	10014.0	21657.0	12063.0	10849.0	15881.0	24375.0
2011-06-01	381787.0	36090.0	6506.0	6645.0	59519.0	5024.0	1849.0	6728.0	159549.0	12312.0	11537.0	22860.0	11634.0	8984.0	13316.0	17119.0

⑤ 수정한 인허가 데이터와 매매가 지수를 함께 그래프로 나타낸다.

```
# 수정한 인허가 데이터와 매매가 지수 그래프

plt.figure(figsize=(10,6))
ax = plt.subplot()
ax2 = ax.twinx()

si = '서울'
gu = '서울'

plt.title(si + '-' + gu)
ln1 = ax.plot(price_index[si][gu]['2009-1':], label='매매가')
ln2 = ax2.plot(modified_permission[si]['2009':], label='인허가', color='green',marker="o")
lns = ln1 +ln2
labs = [l.get_label() for l in lns]
ax.legend(lns, labs, loc='upper left')

plt.show()
```

⑥ 전세가 지수 데이터를 추가해서 그래프로 나타낸다.

한 그래프에 여러 개의 데이터가 들어가 인허가 그래프는 ls='--' 옵션을 추가해 선을 점선으로 한다. 데이터 값 자체가 아니라 변화율을 보고 싶다면 pandas 의 pct_change() 함수를 이용한다. 연간 변화율을 계산하기 위해 pct_change(12)라는 옵션을 추가한다.

```
# 변화율로 살펴보는 그래프

plt.figure(figsize=(10,6))
ax = plt.subplot()
ax2 = ax.twinx()

si = '서울'
gu = '서울'

plt.title(si + '-' + gu)
ln1 = ax.plot(price_index[si][gu]['2009-1':].pct_change(12), label='매매가')
ln2 = ax.plot(jun_index[si][gu]['2009-1':].pct_change(12), label='전세가')
ln3 = ax2.plot(modified_permission[si]['2009':].pct_change(), label='인허가', color='lightslategray', ls='--')
lns = ln1 +ln2 + ln3
labs = [l.get_label() for l in lns]
ax.legend(lns, labs, loc='upper left')

plt.show()
```

⑦ 수요 전략에 인허가 데이터를 추가한다.

```
# demand 함수를 실행해 결과 저장

from datetime import datetime
```

```
from dateutil.relativedelta import relativedelta

def demand(price_index, jeonse_index, index_date, time_range):

    prev_date = index_date - relativedelta(months=time_range)
    prev_date2 = index_date - relativedelta(months=time_range*3)

    demand_df = pd.DataFrame()
    demand_df['매매증감률'] = (price_index.loc[index_date] - price_index.loc[prev_date])/
price_index.loc[prev_date].replace(0,None)
    demand_df['전세증감률'] = (jeonse_index.loc[index_date] -
jeonse_index.loc[prev_date])/jeonse_index.loc[prev_date].replace(0,None)
    demand_df['이전최대값'] = price_index[prev_date2:index_date][:-1].max()
    demand_df['최댓값대비증감률'] = (price_index.loc[index_date] - demand_df['이전최대값'])
/demand_df['이전최대값'].replace(0,None)

    demand_df['매매가상승'] = demand_df['매매증감률'] > 0.01
    demand_df['전세가상승'] = demand_df['전세증감률'] > 0.01
    demand_df['더빠른전세상승'] = demand_df['전세증감률'] > demand_df['매매증감률']
    demand_df['최댓값대비상승'] = demand_df['최댓값대비증감률'] > 0
    demand_df['수요총합'] =
demand_df[['매매가상승','전세가상승','더빠른전세상승','최댓값대비상승']].sum(axis=1)

    demand_df = demand_df[demand_df['수요총합'] == 4]

    seleted_index = []

    for name in demand_df.index:
        if name[0] is not name[1]:
            seleted_index.append((name[0], name[1]))

    demand_df = demand_df.loc[seleted_index]

    return demand_df

index_date = datetime(2013, 1, 1)
time_range = 12
demand_1 = demand(price_index, jun_index, index_date, time_range)
```

demand_1

		매매증감률	전세증감률	이전최대값	최댓값대비증감률	매매가상승	전세가상승	더빠른전세상승	최댓값대비상승	수요총합
부산	영도구	0.0105916	0.0231865	96.094907	0.000679802	True	True	True	True	4
	연제구	0.0271495	0.0438877	90.282601	0.000141418	True	True	True	True	4
대구	중구	0.0381799	0.0473526	80.969141	0.00302007	True	True	True	True	4
	동구	0.059901	0.0832191	76.263379	0.00435544	True	True	True	True	4
	서구	0.0391477	0.0481874	80.862977	0.0015462	True	True	True	True	4
	북구	0.0808392	0.121241	79.444851	0.0037033	True	True	True	True	4
	수성구	0.0309687	0.0603635	69.659600	0.00202815	True	True	True	True	4
	달서구	0.0766036	0.118091	75.490414	0.00597216	True	True	True	True	4
	달성군	0.0913725	0.115203	80.869389	0.00694921	True	True	True	True	4
광주	동구	0.0314413	0.0402683	89.370661	0.001956	True	True	True	True	4
	서구	0.0401655	0.0459345	77.529566	0.000947088	True	True	True	True	4
	남구	0.0338606	0.0587107	85.682467	8.45907e-05	True	True	True	True	4
	북구	0.0332272	0.0484564	86.939627	0.000925296	True	True	True	True	4
	광산구	0.0449257	0.0757373	81.358856	0.00132932	True	True	True	True	4
울산	남구	0.0516848	0.0608639	93.759913	0.000527829	True	True	True	True	4
	북구	0.0732078	0.0853805	102.278373	0.000840243	True	True	True	True	4
경기	이천	0.0208333	0.041492	93.588009	0.00196	True	True	True	True	4
충북	청주	0.0529109	0.0737586	97.060707	0.00290533	True	True	True	True	4
	상당구	0.0466149	0.0653927	100.530977	0.00568305	True	True	True	True	4
	흥덕구	0.0569288	0.0790754	95.297107	0.00115787	True	True	True	True	4
	충주	0.0283333	0.0462394	98.505446	0.00197854	True	True	True	True	4
충남	천안	0.0834931	0.122239	98.335947	0.00347807	True	True	True	True	4
	동남구	0.0917818	0.123461	96.962410	0.00639117	True	True	True	True	4
	서북구	0.0775694	0.121375	99.455152	0.0013792	True	True	True	True	4
	공주	0.0179989	0.0315939	100.954720	0.000267288	True	True	True	True	4
	아산	0.0779389	0.13861	96.174140	0.00208491	True	True	True	True	4
경북	포항	0.0532139	0.0775433	94.206634	0.00355497	True	True	True	True	4
	북구	0.0713002	0.0974405	94.379928	0.00657981	True	True	True	True	4
	구미	0.096102	0.118327	96.038163	0.0101963	True	True	True	True	4
	경산	0.0907691	0.122536	83.889568	0.00309716	True	True	True	True	4

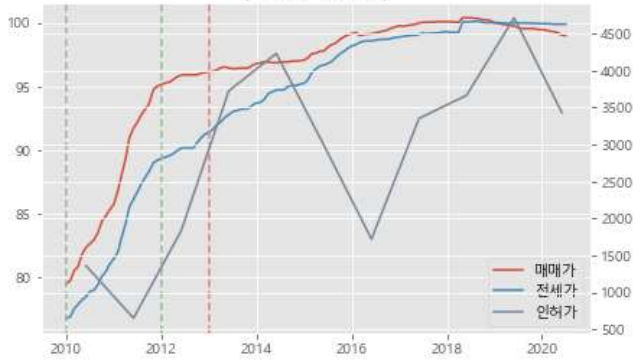
매매가와 전세 지수는 지역이 [시도][시군구]와 같이 이중으로 나뉘어 있는데 인허가 데이터는 [시도] 지역만 표시하고 있다. demand 함수의 결과는 위 그림처럼 데이터프레임인데 인덱스가 지역명이다. 지역명은 (시도, 시군구)와 같은 형식으로 돼 있어서 매매가와 전세가 지수 데이터는 시도와 시군구를 다 활용해서 데이터를 선택하지만 인허가 데이터는 시도만 활용해 데이터를 선택한다.

```
# demand 함수 결과를 인허가 데이터와 함께 보기

prev_date = index_date - relativedelta(months=time_range)
prev_date2 = index_date - relativedelta(months=time_range * 3)
graph_start = index_date - relativedelta(months=time_range * 3)
```

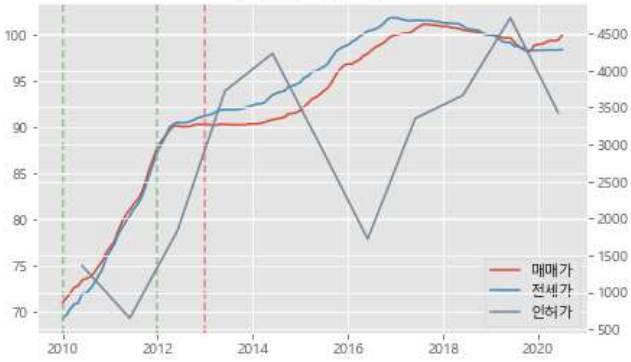
```
num_row = int((len(demand_1.index)-1)/2)+1

plt.figure(figsize=(15, num_row*5))
for i, spot in enumerate(demand_1.index):
    ax = plt.subplot(num_row, 2, i+1)
    si = spot[0]
    gu = spot[1]
    plt.title(spot)
    ax2 = ax.twinx()
    ln1 = ax.plot(price_index[si][gu][graph_start:], label='매매가')
    ln2 = ax.plot(jun_index[si][gu][graph_start:], label='전세가')
    ln3 = ax2.plot(modified_permission[si][graph_start:]/10,  color='lightslategray', label='인허가')

    ax.axvline(x=index_date, color='lightcoral', linestyle='--')
    ax.axvline(x=prev_date, color='darkseagreen', linestyle='--')
    ax.axvline(x=prev_date2, color='darkseagreen', linestyle='--')
    lns = ln1 +ln2 + ln3
    labs = [l.get_label() for l in lns]
    ax.legend(lns, labs, loc='lower right')

plt.show()
```

demand 함수를 실행해서 매매가와 전세가 지수 뿐만아니라 인허가의 공급 요소도 포함되어 있어 다양한 정보를 확인할 수 있다.

('부산', '영도구')



('부산', '연제구')

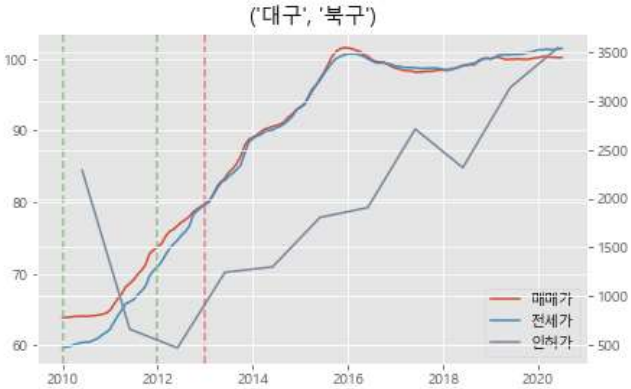


('대구', '중구')

('대구', '동구')

('대구', '서구')

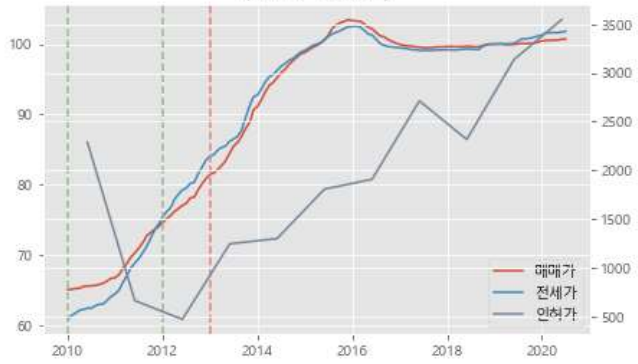
('대구', '북구')



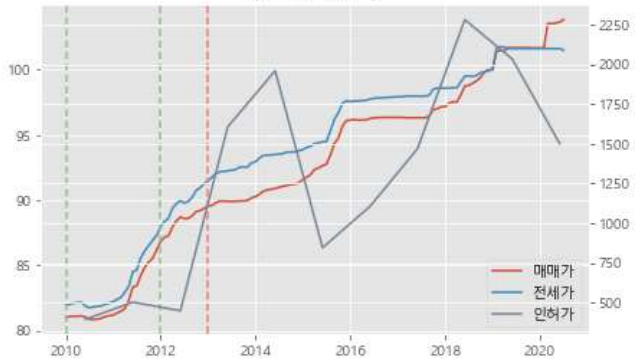
('대구', '수성구')

('대구', '달서구')

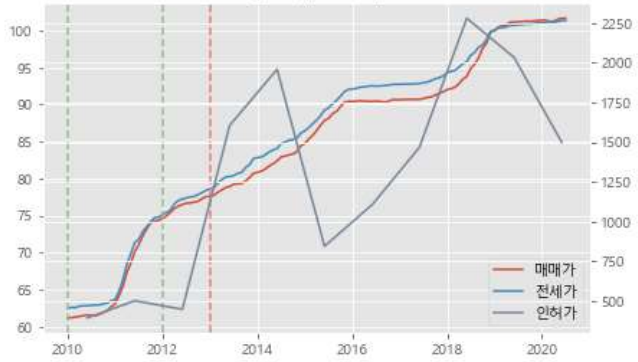
('대구', '달성군')



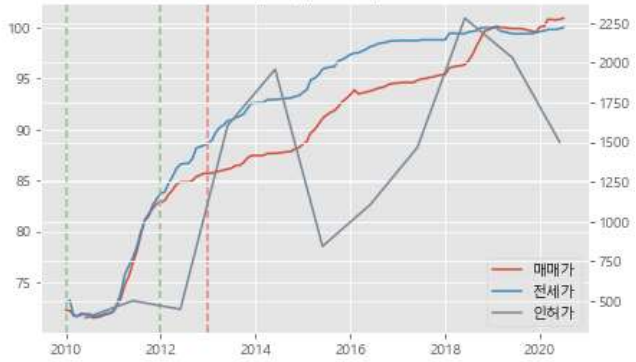
('광주', '동구')



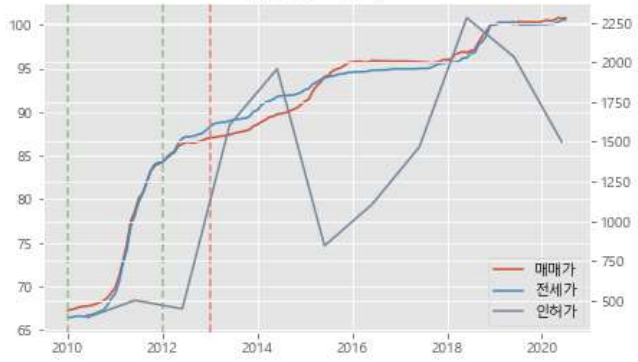
('광주', '서구')



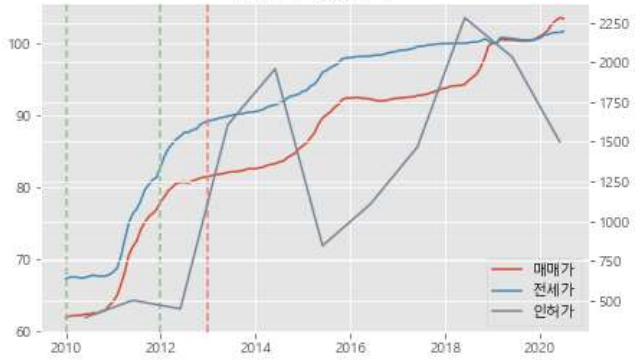
('광주', '남구')



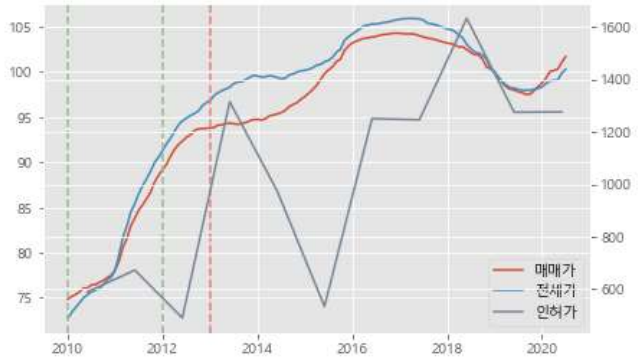
('광주', '북구')



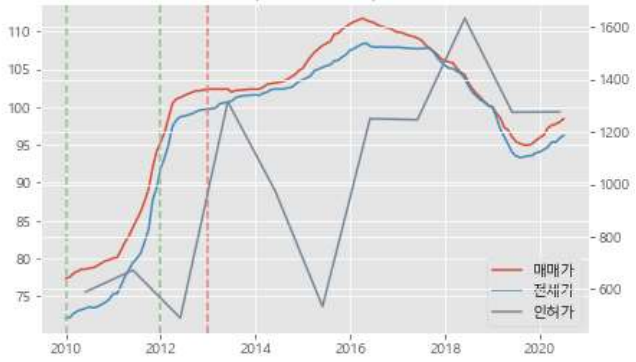
('광주', '광산구')



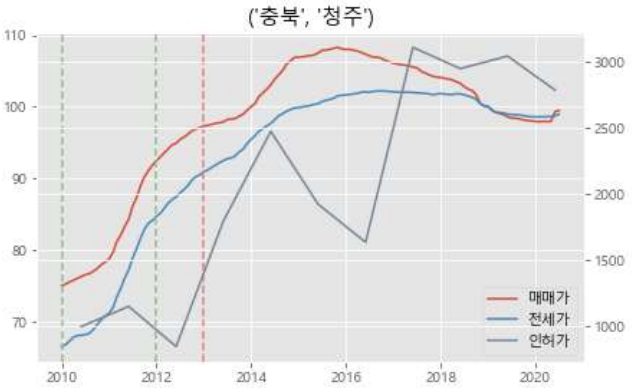
('울산', '남구')



('울산', '북구')



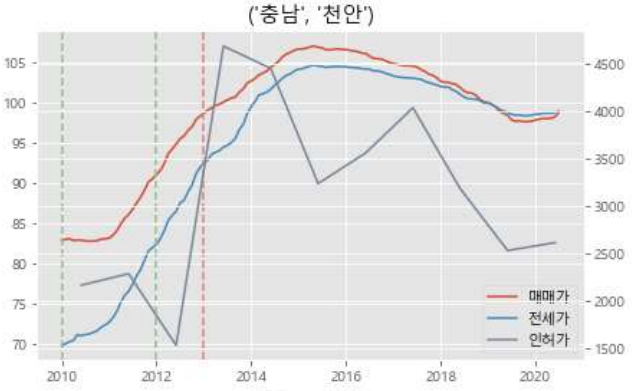

('경기', '이천')
매매가
전세가
인허가

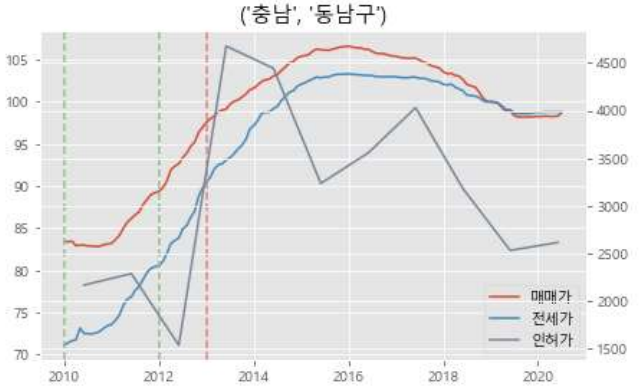

('충북', '청주')
매매가
전세가
인허가

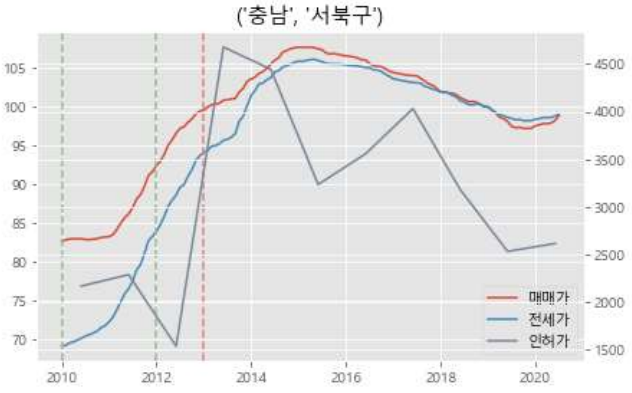

('충북', '상당구')
매매가
전세가
인허가


('충북', '흥덕구')
매매가
전세가
인허가

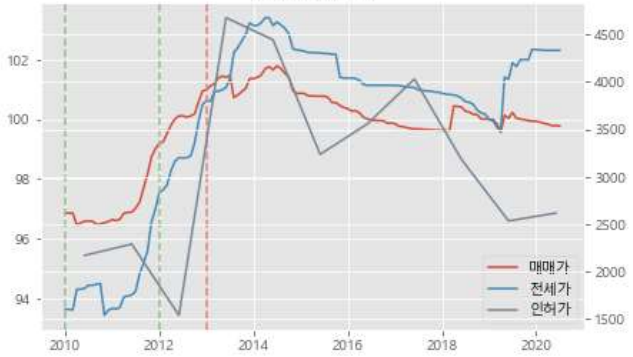

('충북', '충주')
매매가
전세가
인허가


('충남', '천안')
매매가
전세가
인허가

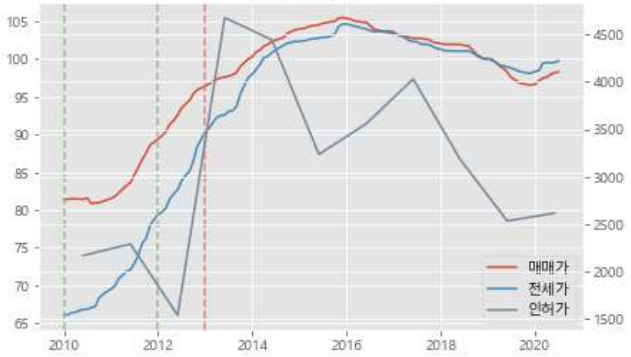

('충남', '동남구')
매매가
전세가
인허가


('충남', '서북구')
매매가
전세가
인허가

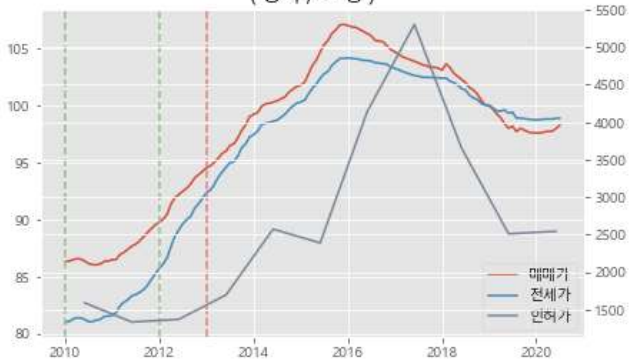
('충남', '공주')



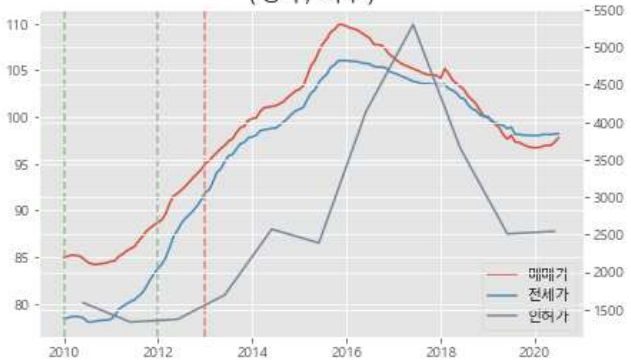
('충남', '아산')



('경북', '포항')



('경북', '북구')



('경북', '구미')

('경북', '경산')