전국 신규 민간 아파트 분양가격 분석

전국의 신규 민간 아파트의 분양가격을 분석하도록 하겠다.

In [191]:

```
import warnings
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd
import re
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import font_manager, rc
font_path = 'C:/Windows/Fonts/H2GTRE.TTF'
font = font_manager.FontProperties(fname=font_path).get_name()
rc('font', family=font)
```

In [192]:

```
df = pd.read_csv("주택도시보증공사_전국 신규 민간 아파트 분양가격 동향_20200331.csv", engine='python')
df.shape
```

Out[192]:

(4590, 5)

In [193]:

```
df.head()
```

Out[193]:

	지역명	규모구분	연도	월	분양가격(㎡)
0	서울	전체	2015	10	5841
1	서울	전용면적 60㎡이하	2015	10	5652
2	서울	전용면적 60㎡초과 85㎡이하	2015	10	5882
3	서울	전용면적 85㎡초과 102㎡이하	2015	10	5721
4	서울	전용면적 102㎡초과	2015	10	5879

In [194]:

```
df.tail()
```

Out[194]:

	지역명	규모구분	연도	월	분양가격(㎡)
4585	제주	전체	2020	3	3955
4586	제주	전용면적 60㎡이하	2020	3	4039
4587	제주	전용면적 60㎡초과 85㎡이하	2020	3	3962
4588	제주	전용면적 85㎡초과 102㎡이하	2020	3	NaN
4589	제주	전용면적 102㎡초과	2020	3	3601

In [195]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4590 entries, 0 to 4589
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   지역명       4590 non-null   object
 1   규모구분      4590 non-null   object
 2   연도        4590 non-null   int64 
 3   월         4590 non-null   int64 
 4   분양가격(㎡)  4276 non-null   object
dtypes: int64(2), object(3)
memory usage: 179.4+ KB
```

In [196]:

```
df.isnull().sum()
```

Out[196]:

```
지역명           0
규모구분          0
연도            0
월             0
분양가격(㎡)     314
dtype: int64
```

In [197]:

```
df.isna().sum()
```

Out[197]:

```
지역명           0
규모구분          0
연도            0
월             0
분양가격(㎡)     314
dtype: int64
```

각 칼럼에 Null값은 분양가격에만 존재한다. isnull().sum()을 이용해 Null 값을 체크해줬다. True는 1, False는 0이다. isnull() 메소드를 사용하면 null이냐 아니냐에 따라 Ture, False 값이 나타나고 이를 sum해주면 true값들만 컬럼을 기준으로 다 더한다. 314개의 true값이 있음을 확인할 수 있다.

In [198]:

```
pd.to_numeric(df["분양가격(㎡)"], errors='coerce')
```

Out[198]:

```
0       5841.0
1       5652.0
2       5882.0
3       5721.0
4       5879.0
         ...
4585    3955.0
4586    4039.0
4587    3962.0
4588       NaN
4589    3601.0
Name: 분양가격(㎡), Length: 4590, dtype: float64
```

In [199]:

```
df["분양가격"] = pd.to_numeric(df["분양가격(㎡)"], errors='coerce')
df["분양가격"].head(1)
```

Out[199]:

```
0    5841.0
Name: 분양가격, dtype: float64
```

In [200]:

```
type(pd.np.nan)
```

Out[200]:

```
float
```

to_numeric에서 errors 옵션을 사용했다. coerce의 경우는 무시하고 공백도 인트로 바꿔준다. 위 결과에서는 nan값이 float타입이기 때문에 전체가 float형태로 바뀌었다.

In [201]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4590 entries, 0 to 4589
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   지역명       4590 non-null   object
 1   규모구분      4590 non-null   object
 2   연도        4590 non-null   int64
 3   월         4590 non-null   int64
 4   분양가격(㎡)  4276 non-null   object
 5   분양가격      4260 non-null   float64
dtypes: float64(1), int64(2), object(3)
memory usage: 215.3+ KB
```

In [202]:

```
# 오브젝트 타입의 요약
df["분양가격(㎡)"].describe()
```

Out[202]:

```
count     4276
unique    1766
top       3395
freq        18
Name: 분양가격(㎡), dtype: object
```

3395라는 숫자가 17번 등장했음을 확인

In [203]:

```
# 수치데이터로 변경된 분양가격 컬럼 요약
df["분양가격"].describe()
```

Out[203]:

```
count     4260.000000
mean      3270.160798
std       1300.362742
min       1868.000000
25%       2454.750000
50%       2890.000000
75%       3601.000000
max      13835.000000
Name: 분양가격, dtype: float64
```

In [204]:

```
df["규모구분"].unique()
```

Out[204]:

```
array(['전체', '전용면적 60㎡이하', '전용면적 60㎡초과 85㎡이하', '전용면적 85㎡초
과 102㎡이하',
       '전용면적 102㎡초과'], dtype=object)
```

In [205]:

```
df["전용면적"] = df["규모구분"].str.replace("전용면적", "")
#df["규모구분"].str.replace("초과", "")
df["전용면적"]
```

Out[205]:

```
0                    전체
1                60㎡이하
2          60㎡초과 85㎡이하
3          85㎡초과 102㎡이하
4               102㎡초과
             ...
4585                 전체
4586             60㎡이하
4587       60㎡초과 85㎡이하
4588       85㎡초과 102㎡이하
4589            102㎡초과
Name: 전용면적, Length: 4590, dtype: object
```

In [206]:

```
df["전용면적"] = df["규모구분"].str.replace("전용면적", "")
df["전용면적"] = df["전용면적"].str.replace("초과", "~")
df["전용면적"] = df["전용면적"].str.replace("이하", "")
df["전용면적"] = df["전용면적"].str.replace(" ", "").str.strip()
df["전용면적"]
```

Out[206]:

```
0                 전체
1               60㎡
2          60㎡~85㎡
3         85㎡~102㎡
4             102㎡~
            ...
4585              전체
4586            60㎡
4587       60㎡~85㎡
4588      85㎡~102㎡
4589          102㎡~
Name: 전용면적, Length: 4590, dtype: object
```

전용면적의 데이터 값을 보기 좋은 문자열로 replace해줬다.

In [207]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4590 entries, 0 to 4589
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   지역명       4590 non-null   object
 1   규모구분      4590 non-null   object
 2   연도        4590 non-null   int64 
 3   월         4590 non-null   int64 
 4   분양가격(㎡)  4276 non-null   object
 5   분양가격      4260 non-null   float64
 6   전용면적      4590 non-null   object
dtypes: float64(1), int64(2), object(4)
memory usage: 251.1+ KB
```

In [208]:

```
# axis 0: 행, 1: 열
df = df.drop(["규모구분", "분양가격(㎡)"], axis=1)
df.head(1)
```

Out[208]:

	지역명	연도	월	분양가격	전용면적
0	서울	2015	10	5841.0	전체

불필요한 컬럼(규모구분, 분양가격(㎡))을 제거했다.

In [209]:

```
df
```

Out[209]:

	지역명	연도	월	분양가격	전용면적
0	서울	2015	10	5841.0	전체
1	서울	2015	10	5652.0	60㎡
2	서울	2015	10	5882.0	60㎡~85㎡
3	서울	2015	10	5721.0	85㎡~102㎡
4	서울	2015	10	5879.0	102㎡~
...	...	...	...	...	...
4585	제주	2020	3	3955.0	전체
4586	제주	2020	3	4039.0	60㎡
4587	제주	2020	3	3962.0	60㎡~85㎡
4588	제주	2020	3	NaN	85㎡~102㎡
4589	제주	2020	3	3601.0	102㎡~

4590 rows × 5 columns

In [210]:

```
df.groupby(["지역명"])["분양가격"].mean()
```

Out[210]:

```
지역명
강원     2419.072797
경기     4087.385185
경남     2817.060606
경북     2548.451362
광주     3066.281106
대구     3679.222222
대전     3132.873239
부산     3687.429630
서울     7268.218519
세종     2993.287402
울산     3046.746914
인천     3645.875940
전남     2316.270677
전북     2352.917603
제주     3440.598291
충남     2506.089147
충북     2320.633333
Name: 분양가격, dtype: float64
```

In [211]:

```
df.groupby(["전용면적"])["분양가격"].mean()
```

Out[211]:

```
전용면적
102㎡~          3524.597619
60㎡             3172.022196
60㎡~85㎡        3137.927393
85㎡~102㎡       3414.938338
전체              3140.871287
Name: 분양가격, dtype: float64
```

In [212]:

```
df.groupby(["지역명", "전용면적"])["분양가격"].mean()
```

Out[212]:

```
지역명  전용면적
강원    102㎡~          2599.703704
        60㎡             2306.148148
        60㎡~85㎡        2280.185185
        85㎡~102㎡       2671.977778
        전체              2279.500000
                          ...
충북    102㎡~          2490.666667
        60㎡             2166.481481
        60㎡~85㎡        2207.685185
        85㎡~102㎡       2542.851852
        전체              2195.481481
Name: 분양가격, Length: 85, dtype: float64
```

In [213]:

```
df.groupby(["전용면적", "지역명"])["분양가격"].mean().unstack()
```

Out[213]:

지역명	강원	경기	경남	경북	광주	대구	
전용면적							
102㎡~	2599.703704	4506.000000	3132.880000	2790.759259	3404.731707	4026.314815	4507.6
60㎡	2306.148148	4060.370370	2643.230769	2399.685185	2921.510204	3668.129630	2826.0
60㎡~85㎡	2280.185185	3823.333333	2634.592593	2457.111111	3048.759259	3619.462963	2971.0
85㎡~102㎡	2671.977778	4207.685185	3045.129630	2658.731707	2817.000000	3421.711111	2750.7
전체	2279.500000	3839.537037	2646.425926	2462.518519	3045.907407	3617.574074	2994.8

In [214]:

```
# 연도, 지역명으로 분양가격의 평균을 구한다.
g = df.groupby(["연도", "지역명"])["분양가격"].mean()
g
```

Out[214]:

```
연도    지역명
2015  강원      2178.200000
      경기      3351.800000
      경남      2563.400000
      경북      2261.866667
      광주      2399.000000
                  ...
2020  전남      2744.538462
      전북      2576.600000
      제주      3889.250000
      충남      2736.000000
      충북      2470.466667
Name: 분양가격, Length: 102, dtype: float64
```

In [215]:

```
df.groupby(["연도", "지역명"])["분양가격"].mean().unstack()
```

Out[215]:

지역명	강원	경기	경남	경북	광주	대구	
연도							
2015	2178.200000	3351.800000	2563.400000	2261.866667	2399.000000	2733.000000	2482.0
2016	2170.576923	3540.900000	2574.766667	2349.516667	2785.055556	3115.766667	2700.2
2017	2217.833333	3726.116667	2668.666667	2511.534483	2909.000000	3694.537037	3003.9
2018	2490.683333	4320.733333	2826.566667	2630.538462	2886.955556	3678.561404	3101.2
2019	2707.416667	4747.133333	3241.701754	2742.500000	3670.204545	4267.166667	3824.0
2020	2919.928571	4881.600000	3023.000000	2753.833333	3985.000000	4474.866667	3708.8

In [216]:

```
df.groupby(["연도", "지역명"])["분양가격"].mean().unstack().T
```

Out[216]:

연도	2015	2016	2017	2018	2019	2020
지역명						
강원	2178.200000	2170.576923	2217.833333	2490.683333	2707.416667	2919.928571
경기	3351.800000	3540.900000	3726.116667	4320.733333	4747.133333	4881.600000
경남	2563.400000	2574.766667	2668.666667	2826.566667	3241.701754	3023.000000
경북	2261.866667	2349.516667	2511.534483	2630.538462	2742.500000	2753.833333
광주	2399.000000	2785.055556	2909.000000	2886.955556	3670.204545	3985.000000
대구	2733.000000	3115.766667	3694.537037	3678.561404	4267.166667	4474.866667
대전	2482.000000	2700.222222	3003.912281	3101.244444	3824.000000	3708.818182
부산	3144.666667	3255.616667	3526.550000	3906.050000	4102.383333	4066.666667
서울	6156.266667	6591.950000	6625.483333	7030.983333	8571.766667	9390.933333
세종	2656.066667	2684.183333	2767.559322	3133.473684	3424.058824	3573.500000
울산	2838.666667	2903.810345	3221.260870	3103.454545	3095.833333	3218.666667
인천	3326.066667	3363.350000	3537.100000	3600.464286	4015.083333	4223.600000
전남	2060.266667	2102.000000	2246.433333	2402.983333	2490.689655	2744.538462
전북	2154.666667	2092.916667	2244.280702	2477.150000	2585.533333	2576.600000
제주	2409.416667	2899.236364	3826.830189	3616.960000	3584.384615	3889.250000
충남	2330.266667	2411.583333	2471.913793	2485.400000	2651.163636	2736.000000
충북	2069.333333	2161.616667	2261.383333	2469.483333	2415.416667	2470.466667

groupby를 이용해 데이터 집계를 하여 그룹별 분양가격의 평균을 확인했다.

In [217]:

```
pd.pivot_table(df, index=["지역명"], values=["분양가격"], aggfunc="mean")
```

Out[217]:

지역명	분양가격
강원	2419.072797
경기	4087.385185
경남	2817.060606
경북	2548.451362
광주	3066.281106
대구	3679.222222
대전	3132.873239
부산	3687.429630
서울	7268.218519
세종	2993.287402
울산	3046.746914
인천	3645.875940
전남	2316.270677
전북	2352.917603
제주	3440.598291
충남	2506.089147
충북	2320.633333

In [218]:

```
df.groupby(["전용면적"])["분양가격"].mean()
```

Out[218]:

```
전용면적
102㎡~          3524.597619
60㎡             3172.022196
60㎡~85㎡        3137.927393
85㎡~102㎡       3414.938338
전체              3140.871287
Name: 분양가격, dtype: float64
```

In [219]:

```
pd.pivot_table(df, index="전용면적", values="분양가격")
```

Out[219]:

전용면적	분양가격
102㎡~	3524.597619
60㎡	3172.022196
60㎡~85㎡	3137.927393
85㎡~102㎡	3414.938338
전체	3140.871287

pivot_table은 더 명시적으로 인자를 준다는 것과 결과가 데이터프레임으로 나온다는 점이 다르다. groupby는 시리즈 형태로, pivot_table은 데이터프레임 형태로 결과를 출력한다.

In [220]:

```
# 지역명, 전용면적으로 분양가격의 평균을 구한다.
df.groupby(["전용면적", "지역명"])["분양가격"].mean().unstack().round()
```

Out[220]:

지역명	강원	경기	경남	경북	광주	대구	대전	부산	서울	세종	울산
전용면적											
102㎡~	2600.0	4506.0	3133.0	2791.0	3405.0	4026.0	4508.0	4018.0	7203.0	3106.0	3012.0
60㎡	2306.0	4060.0	2643.0	2400.0	2922.0	3668.0	2826.0	3463.0	7105.0	2824.0	2848.0
60㎡~85㎡	2280.0	3823.0	2635.0	2457.0	3049.0	3619.0	2971.0	3618.0	6955.0	2982.0	3182.0
85㎡~102㎡	2672.0	4208.0	3045.0	2659.0	2817.0	3422.0	2751.0	3701.0	8163.0	3010.0	2685.0
전체	2280.0	3840.0	2646.0	2463.0	3046.0	3618.0	2995.0	3638.0	6914.0	2994.0	3181.0

In [221]:

```
df.pivot_table(index=["전용면적", "지역명"], values="분양가격")
```

Out[221]:

전용면적	지역명	분양가격
102㎡~	강원	2599.703704
	경기	4506.000000
	경남	3132.880000
	경북	2790.759259
	광주	3404.731707
...	...	...
전체	전남	2228.018519
	전북	2224.037037
	제주	3311.740741
	충남	2380.074074
	충북	2195.481481

85 rows × 1 columns

In [222]:

```
df.pivot_table(index='전용면적', columns='지역명', values='분양가격').round()
```

Out[222]:

지역명	강원	경기	경남	경북	광주	대구	대전	부산	서울	세종	울산
전용면적											
102㎡~	2600.0	4506.0	3133.0	2791.0	3405.0	4026.0	4508.0	4018.0	7203.0	3106.0	3012.0
60㎡	2306.0	4060.0	2643.0	2400.0	2922.0	3668.0	2826.0	3463.0	7105.0	2824.0	2848.0
60㎡~85㎡	2280.0	3823.0	2635.0	2457.0	3049.0	3619.0	2971.0	3618.0	6955.0	2982.0	3182.0
85㎡~102㎡	2672.0	4208.0	3045.0	2659.0	2817.0	3422.0	2751.0	3701.0	8163.0	3010.0	2685.0
전체	2280.0	3840.0	2646.0	2463.0	3046.0	3618.0	2995.0	3638.0	6914.0	2994.0	3181.0

In [223]:

```
# 연도, 지역명으로 분양가격의 평균 구하기
# g = df.groupby(['연도', '지역명'])['분양가격'].mean()
p = pd.pivot_table(df, index=['연도', '지역명'], values='분양가격')
p.loc[2017]
```

Out[223]:

지역명	분양가격
강원	2217.833333
경기	3726.116667
경남	2668.666667
경북	2511.534483
광주	2909.000000
대구	3694.537037
대전	3003.912281
부산	3526.550000
서울	6625.483333
세종	2767.559322
울산	3221.260870
인천	3537.100000
전남	2246.433333
전북	2244.280702
제주	3826.830189
충남	2471.913793
충북	2261.383333

loc 메서드로 인덱스로 조회해 원하는 결과만을 출력할 수 있다.

In [224]:

```
df.columns.to_list
```

Out[224]:

```
<bound method IndexOpsMixin.tolist of Index(['지역명', '연도', '월', '분양가격',
'전용면적'], dtype='object')>
```

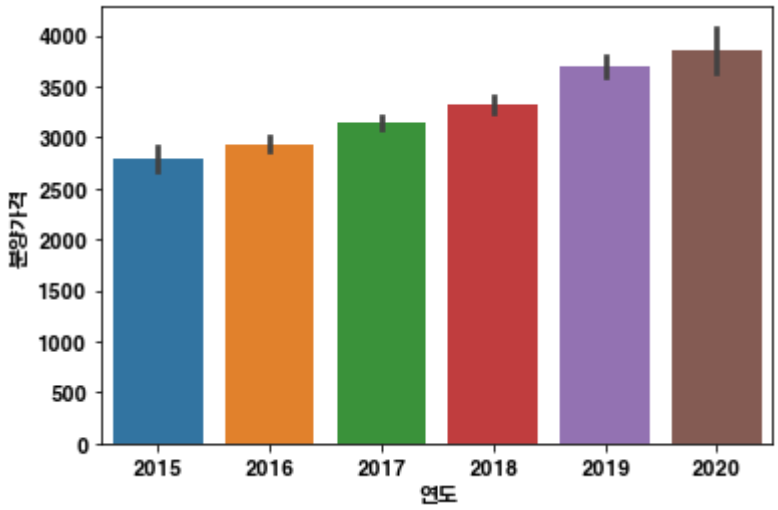
연도별 분양가격 평균 그래프

In [225]:

```
sns.barplot(data=df, x='연도', y='분양가격')
```

Out[225]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x155c5719208>
```

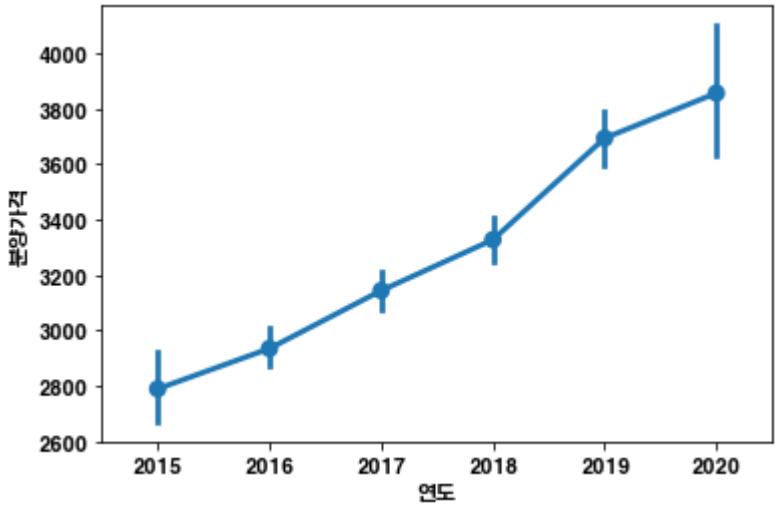


In [226]:

```
sns.pointplot(data=df, x='연도', y='분양가격')
```

Out[226]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x155c5aa9c08>
```

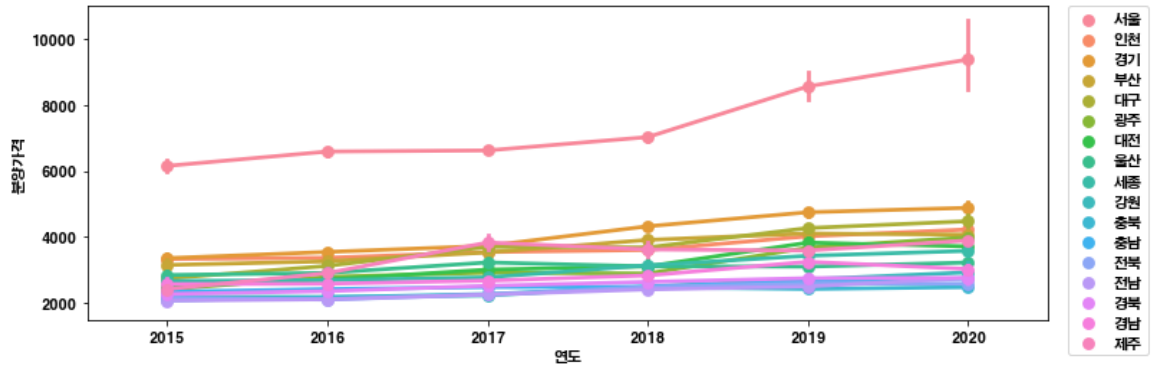


In [227]:

```
plt.figure(figsize=(12, 4))
sns.pointplot(data=df, x='연도', y='분양가격', hue='지역명')
plt.legend(bbox_to_anchor=(1.02, 1), loc=2, borderaxespad=0.)
```

Out[227]:

<matplotlib.legend.Legend at 0x155c5ab92c8>



서울만 barplot으로 그리기

In [228]:

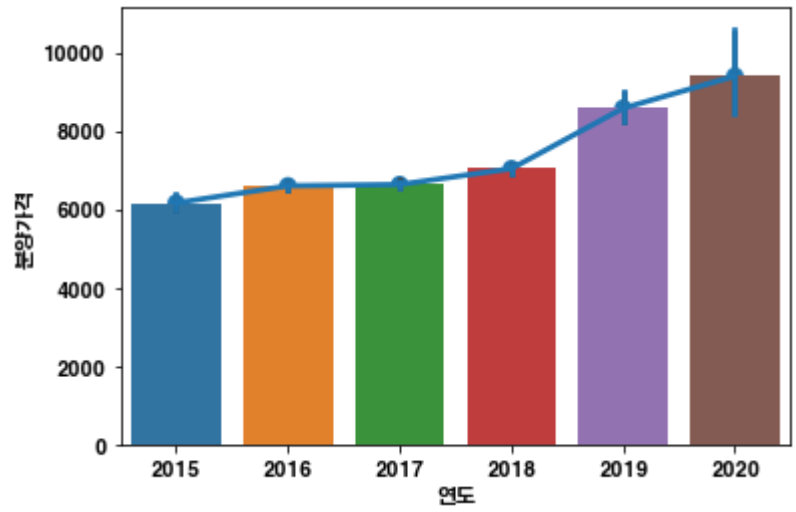
```
df_seoul = df[df["지역명"] == '서울'].copy()
print(df_seoul.shape)

sns.barplot(data=df_seoul, x='연도', y='분양가격')
sns.pointplot(data=df_seoul, x='연도', y='분양가격')
```

(270, 5)

Out[228]:

<matplotlib.axes._subplots.AxesSubplot at 0x155c57bdfc8>



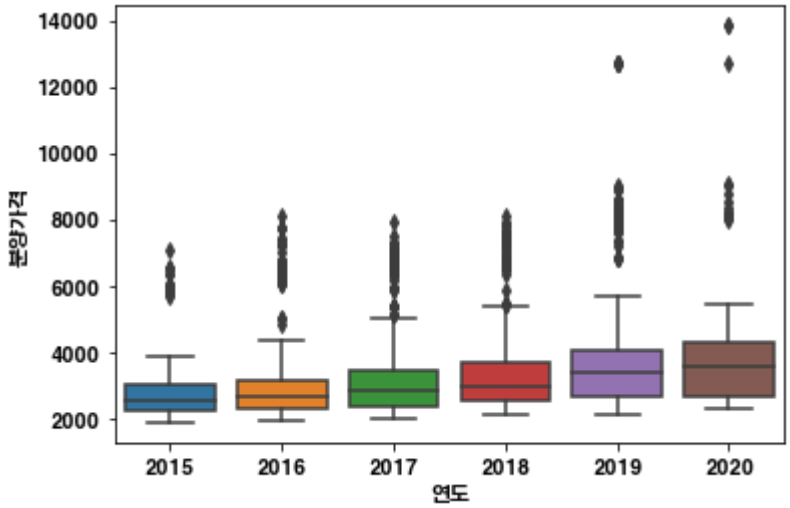
연도별 분양가격 boxplot 그리기

In [229]:

```
sns.boxplot(data=df, x='연도', y='분양가격')
```

Out[229]:

<matplotlib.axes._subplots.AxesSubplot at 0x155c57e0f88>

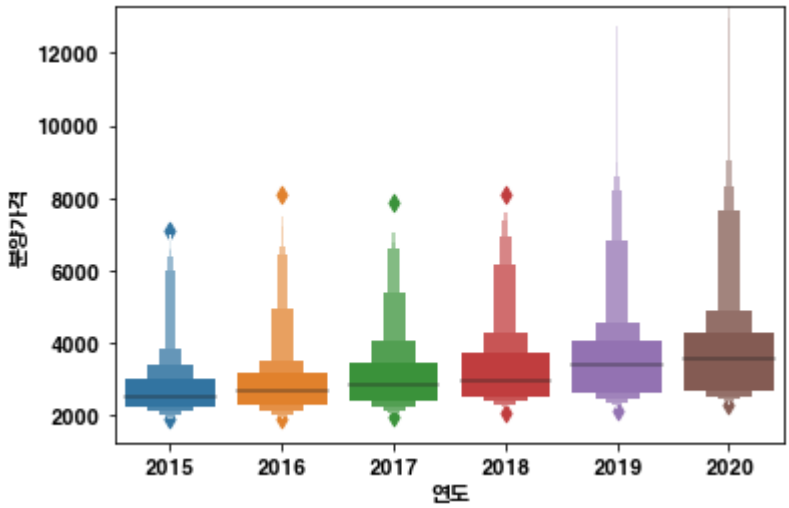


In [230]:

```
sns.boxenplot(data=df, x='연도', y='분양가격')
```

Out[230]:

<matplotlib.axes._subplots.AxesSubplot at 0x155c581c4c8>

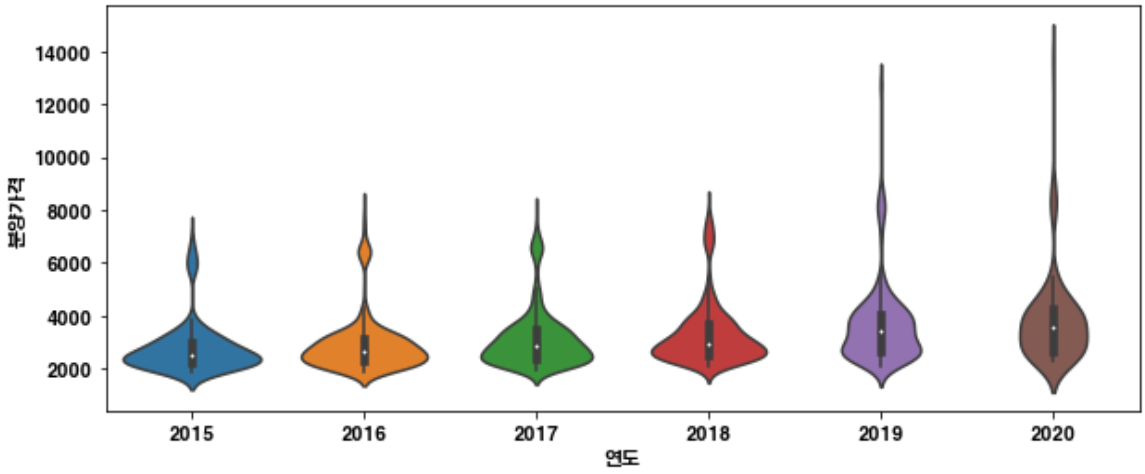


In [231]:

```
plt.figure(figsize=(10, 4))
sns.violinplot(data=df, x='연도', y='분양가격')
```

Out[231]:

<matplotlib.axes._subplots.AxesSubplot at 0x155c5875a48>

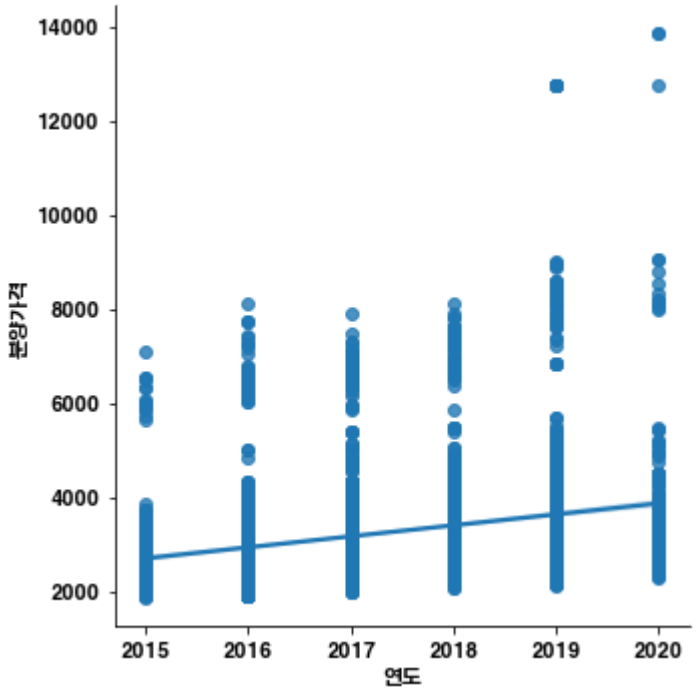


In [232]:

```
sns.lmplot(data=df, x='연도', y='분양가격')
```

Out[232]:

<seaborn.axisgrid.FacetGrid at 0x155c587c808>



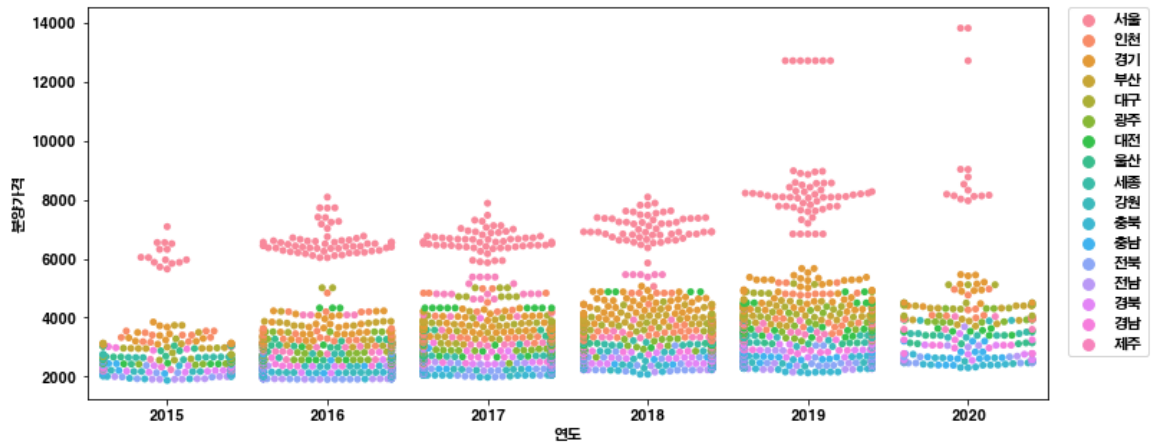
연도별 분양가격 swarmplot 그리기

In [233]:

```
plt.figure(figsize=(12, 5))
sns.swarmplot(data=df, x='연도', y='분양가격', hue='지역명')
plt.legend(bbox_to_anchor=(1.02, 1), loc=2, borderaxespad=0.)
```

Out[233]:

<matplotlib.legend.Legend at 0x155c5f4cc48>

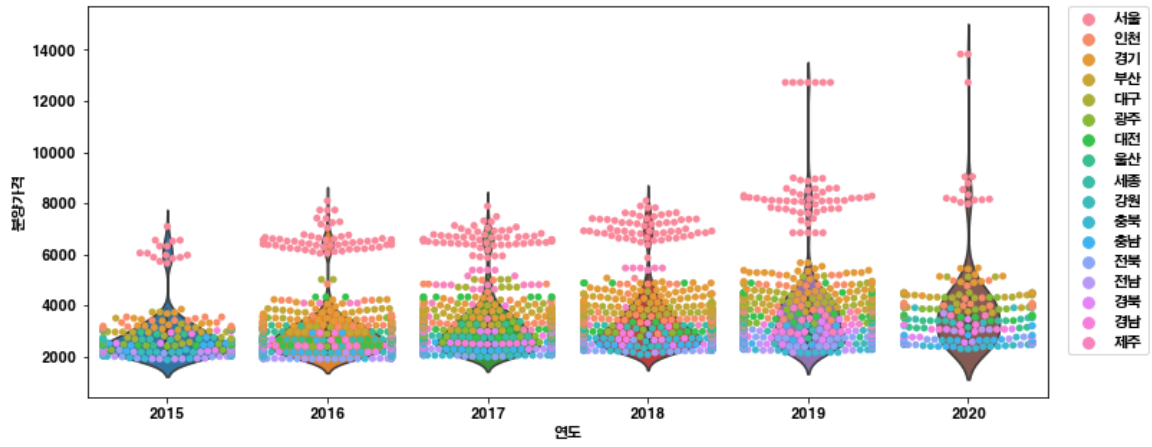


In [234]:

```
plt.figure(figsize=(12, 5))
sns.violinplot(data=df, x='연도', y='분양가격')
sns.swarmplot(data=df, x='연도', y='분양가격', hue='지역명')
plt.legend(bbox_to_anchor=(1.02, 1), loc=2, borderaxespad=0.)
```

Out[234]:

<matplotlib.legend.Legend at 0x155c6fdf908>



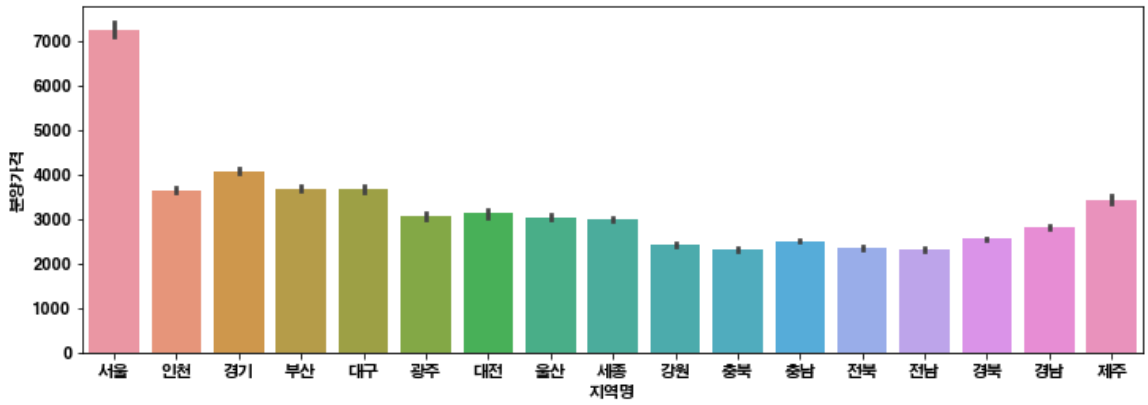
지역별 분양가격

In [235]:

```
# barplot으로 지역별 분양가격 그리기
plt.figure(figsize=(12, 4))
sns.barplot(data=df, x='지역명', y='분양가격')
```

Out[235]:

<matplotlib.axes._subplots.AxesSubplot at 0x155c701e748>

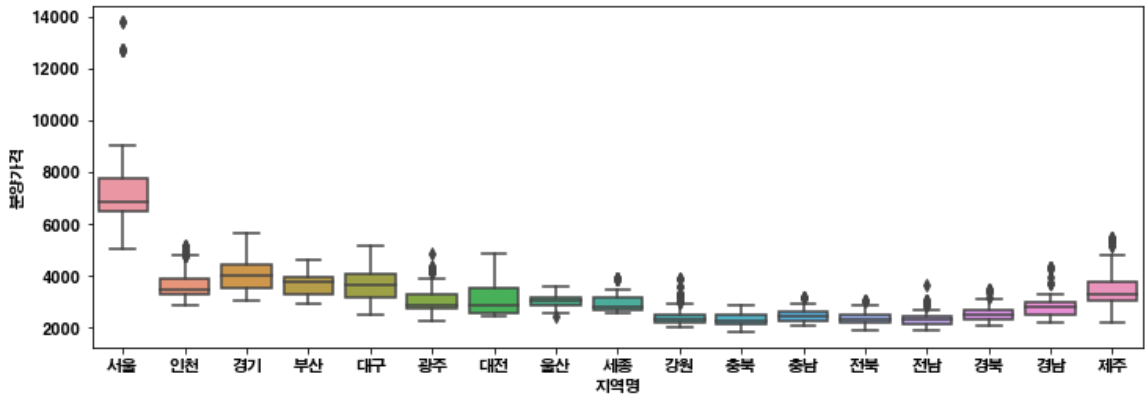


In [236]:

```
# boxplot으로 지역별 분양가격 그리기
plt.figure(figsize=(12, 4))
sns.boxplot(data=df, x='지역명', y='분양가격')
```

Out[236]:

<matplotlib.axes._subplots.AxesSubplot at 0x155c71379c8>

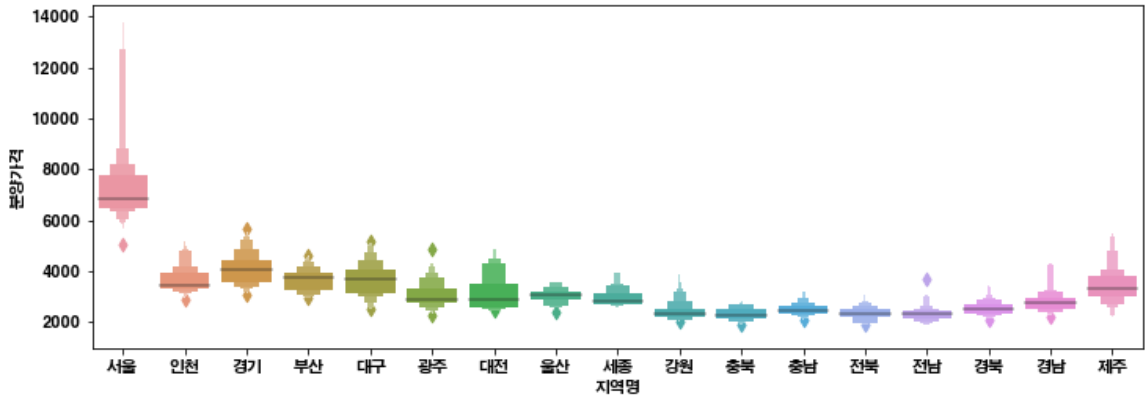


In [237]:

```
plt.figure(figsize=(12, 4))
sns.boxenplot(data=df, x='지역명', y='분양가격')
```

Out[237]:

<matplotlib.axes._subplots.AxesSubplot at 0x155c70f5f48>

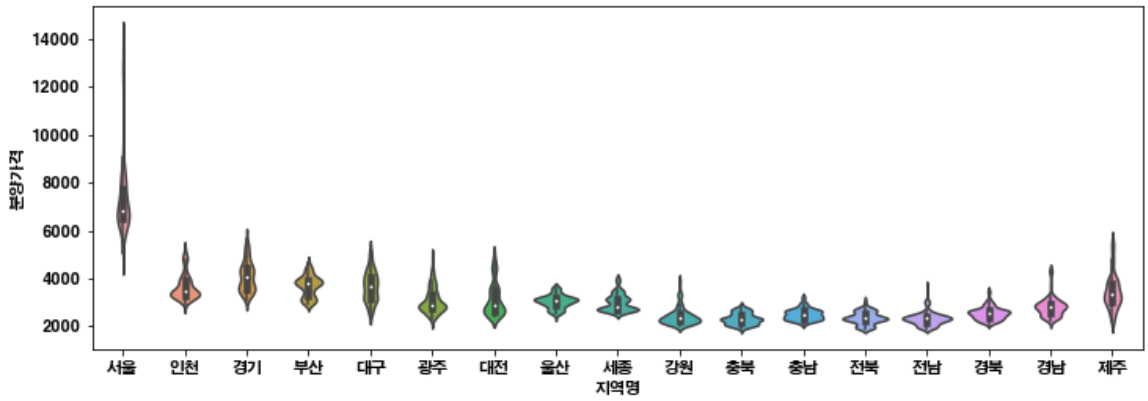


In [238]:

```
# violinplot으로 지역별 분양가격 그리기
plt.figure(figsize=(12, 4))
sns.violinplot(data=df, x='지역명', y='분양가격')
```

Out[238]:

<matplotlib.axes._subplots.AxesSubplot at 0x155c73a1e08>

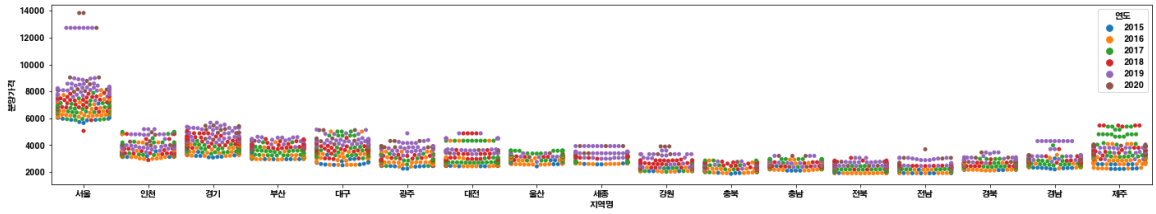


In [239]:

```
# swarmplot으로 지역별 분양가격 그리기
plt.figure(figsize=(24, 4))
sns.swarmplot(data=df, x='지역명', y='분양가격', hue='연도')
```

Out[239]:

<matplotlib.axes._subplots.AxesSubplot at 0x155c7038c08>



이렇게, 신축 아파트 분양가격을 분석해봤다. 그래프에서 보이 듯이 서울과 경기권의 아파트가 분양가가 훨씬 높았다. 시간 순차적으로 보면 2017년 이 후, 분양가가 급격히 올랐음을 확인할 수 있었다. 실제 뉴스에서 접하는 아파트 분양가 상승과 똑같은 흐름을 확인했다.

어려웠던 부분은 서칭을 통해 해결할 수 있었으며 이번 과제를 통해 데이터 분석의 재미를 한 층 더 깊게 알아갈 수 있었다.