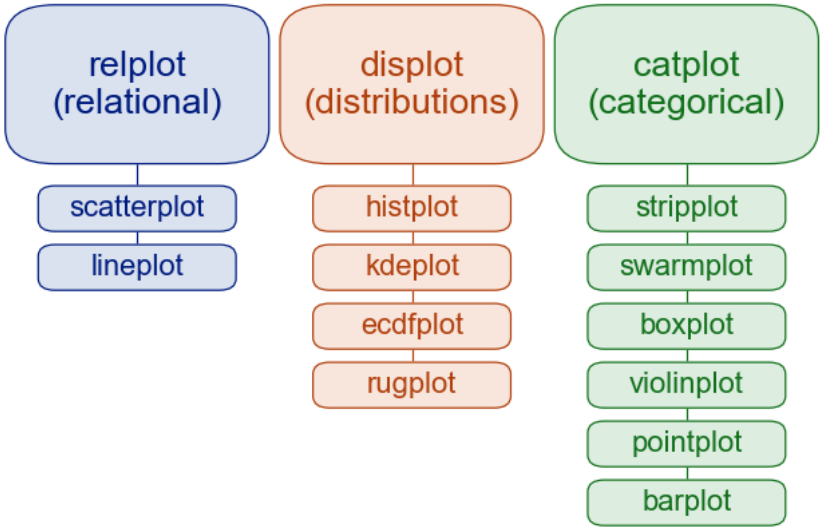
Seaborn 으로 시각화하기

Pandas를 이용해 데이터 처리를 한 후 시각화를 해봤습니다. 그에 비해 Seaborn 이라는 라이브러리는 별도의 내부 연산을 통해서 데이터를 시각화 하는 기능이 있으니 보다 쉽고, 편리하게 데이터를 처리해 볼 수 있습니다.

※ Seaborn 공식문서 [외부링크↗]



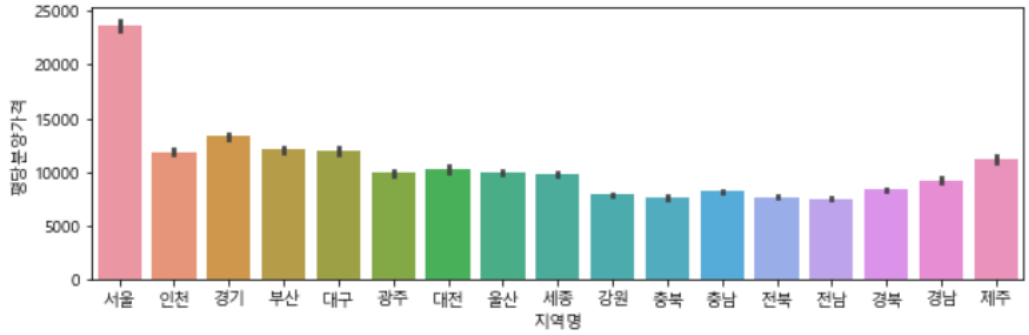
1. 라이브러리 불러오기

```
import seaborn as sns

# 최신 버전의 주피터 노트북 사용자는 아래 코드가 불필요, 구버전의 경우 추가
# %matplotlib inline

# 공식문서
# https://seaborn.pydata.org/index.html
```

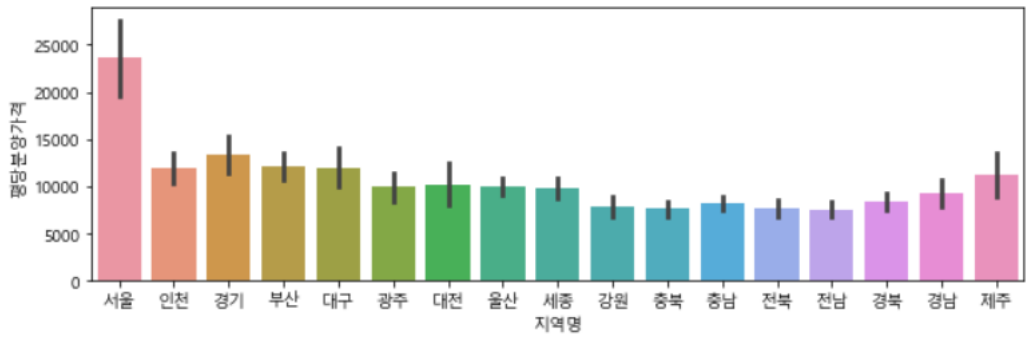
2. barplot 으로 지역별 평당 분양가격을 시각화하기

```
# 이전의 pandas 시각화하기에 이어서 작성
# docstring 확인하기
# sns.barplot? 연산의 기본은 mean(평균)
# seaborn을 사용할 경우 groupby나 pivot_table로 통계를 구하지 않고도 시각화 가능
sns.barplot(data=df_last, x='지역명', y='평당분양가격')

# 그래프 사이즈 (matplotlib 기반)
plt.figure(figsize=(10,3))

# confidence interval (신뢰구간) - estimate value(추정치, 연산의 기본은 mean)
# ci : 95 (신뢰구간 95%, 나머지 5% 이상치, 기본값)
# ci : sd (Standard Deviation, 관측값, 표준편차)
# ci : None
```

<AxesSubplot:xlabel='지역명', ylabel='평당분양가격'>



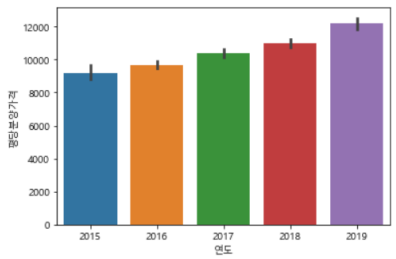

```
# 표준편차
sns.barplot(data=df_last, x='지역명', y='평당분양가격', ci='sd')
```



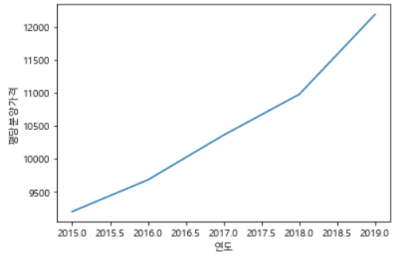
3. 연도별 평당 분양가격을 시각화하기

```
# 평균값
sns.barplot(data=df_last, x='연도', y='평당분양가격')

# barplot --> lineplot --> boxplot 등으로 변경해서 확인해보세요!
# 대부분의 사용법이 비슷함.
```

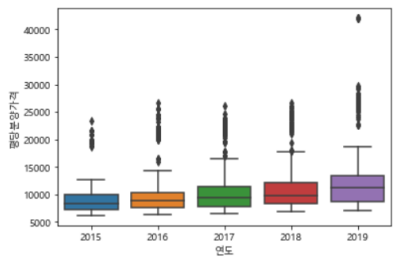
<AxesSubplot:xlabel='연도', ylabel='평당분양가격'>



<AxesSubplot:xlabel='연도', ylabel='평당분양가격'>

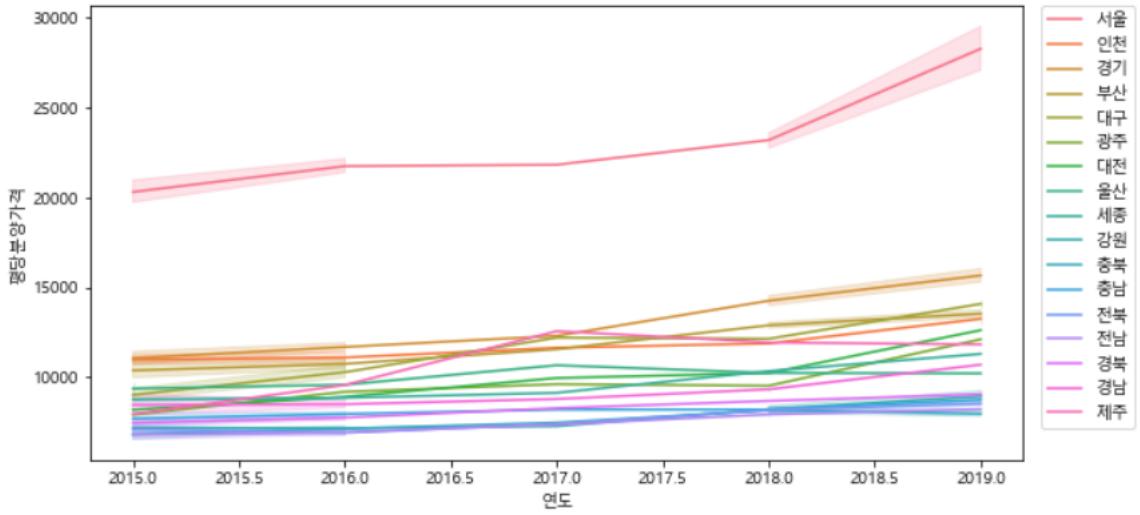


<AxesSubplot:xlabel='연도', ylabel='평당분양가격'>



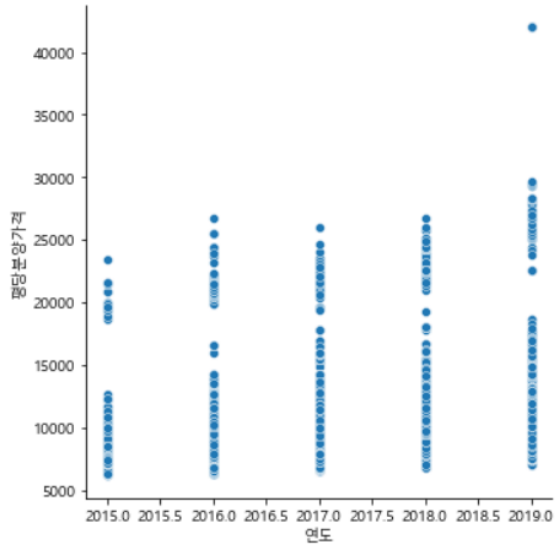

```
# 그래프 사이즈 조정하기
plt.figure(figsize=(10, 5))

# 라인그래프 그리기
# hue : 그룹화하여 다르게 보여줄 카테고리 데이터
sns.lineplot(data=df_last, x='연도', y='평당분양가격', hue='지역명')

# legend 위치 조정하기 (stackoverflow 링크 참조)
plt.legend(bbox_to_anchor=(1.02, 1), loc=2, borderaxespad=0.)
```

```
<matplotlib.legend.Legend at 0x23dcd8b8ac0>
```

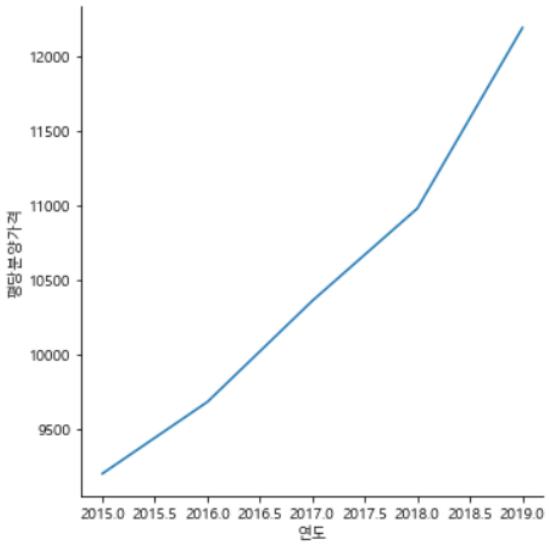



```
# relplot 그려보기
sns.relplot(data=df_last, x='연도', y='평당분양가격', hue='지역명')

# relplot --> kind=scatter, line을 선택가능
# (좌)scatter,  (우)line
```

```
<seaborn.axisgrid.FacetGrid at 0x15e56f07250>
```

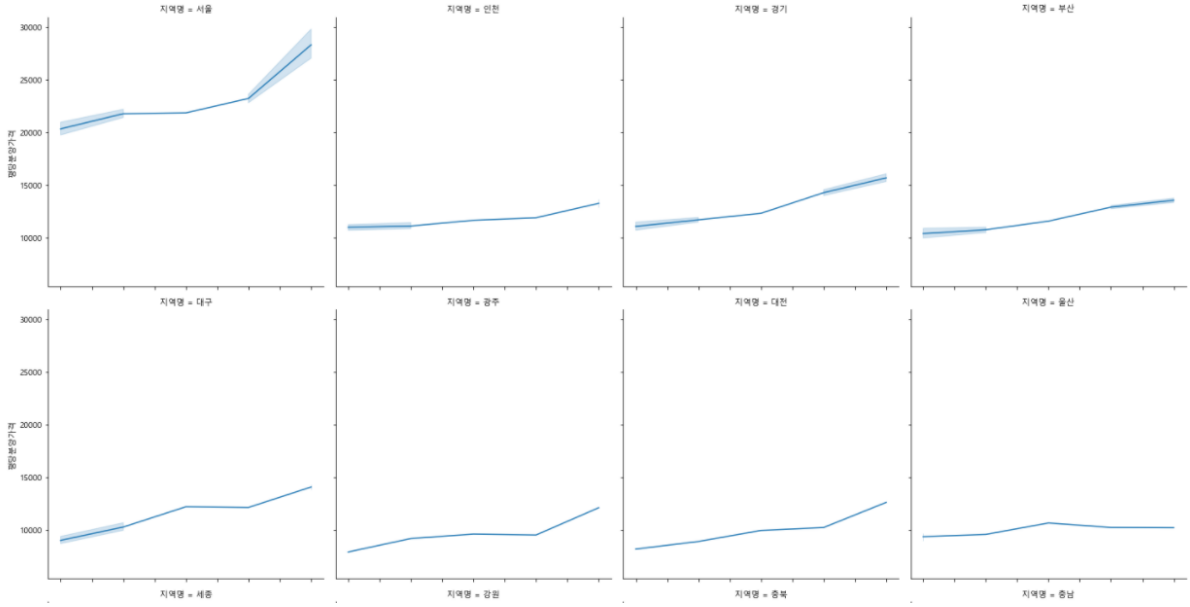



```
<seaborn.axisgrid.FacetGrid at 0x15e56ef4ca0>
```

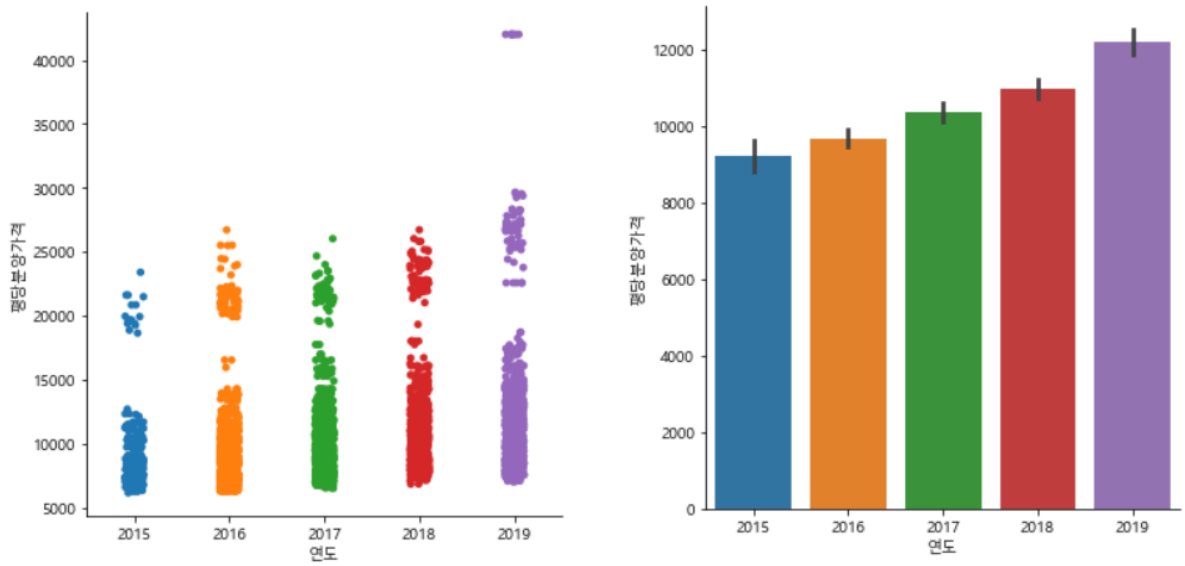



```
# 서브플롯 그리기, 갯수 조절하기
sns.relplot(data=df_last, x='연도', y='평당분양가격', kind='line', col='지역명', col_wrap=4)

# relplot --> lineplot을 subplot(플롯을 여러개)로 그린 것
# 연산을 조금 더 빠르게 하기 위해 ci=None을 추가해보세요
```

```
<seaborn.axisgrid.FacetGrid at 0x15e585fea90>
```

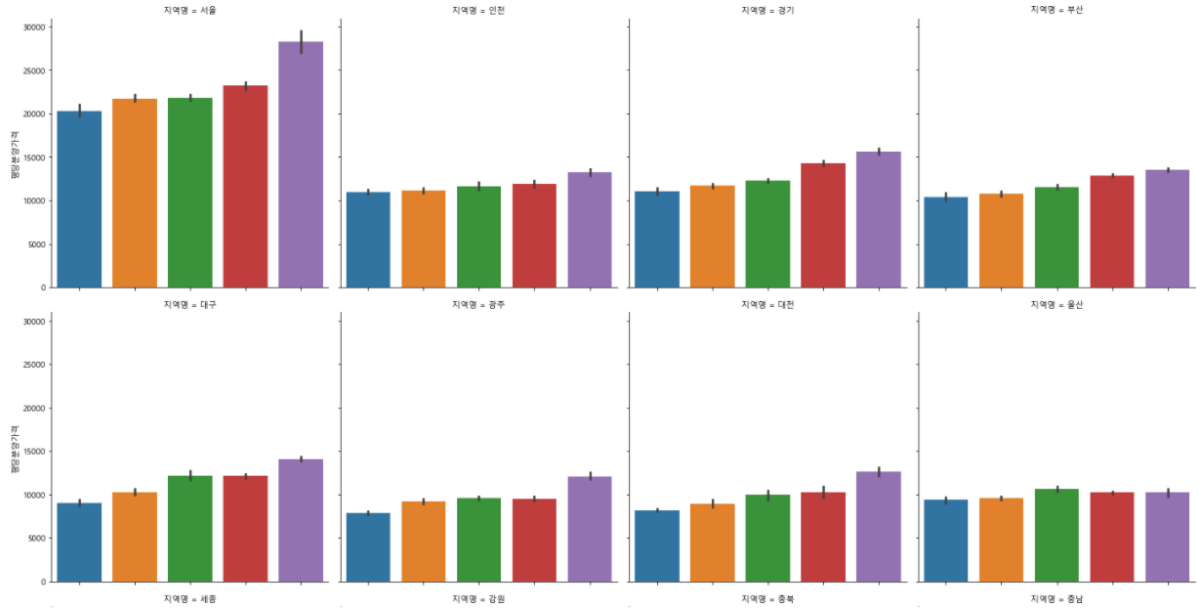



```
# catplot 으로 시각화해보기
# catplot --> barplot을 subplot으로 그린것
# kind는 scatter가 기본
sns.catplot(data=df_last, x='연도', y='평당분양가격')
# (좌)scatter   (우) kind='line'
```




```
# catplot으로 bar형태의 subplot 그려보기
# barplot과 비교해보기 : 1개 vs 4개
sns.catplot(data=df_last, x='연도', y='평당분양가격', kind='bar', col='지역명', col_wrap=4)
```

```
<seaborn.axisgrid.FacetGrid at 0x15e51c1bbe0>
```



Seaborn의 장점 : 데이터 통계처리 없이 손쉽게 시각화 할 수 있고 이를 내부적인 연산(mean, numpy를 활용하여 추가적인 연산을 진행가능)을 해주기 때문에 Pandas를 이용하여 시각화 하는 것보다 쉽습니다. Pandas의 경우 groupby, pivot_table 을 이용해 통계값을 구한 뒤 시각화 해야하고, 서브플롯의 경우 반복문과 명령어를 사용해야만 작성할 수 있습니다.

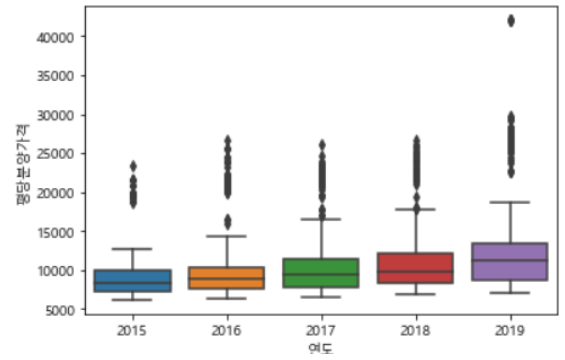
※ Pandas Subplot [외부링크↗]

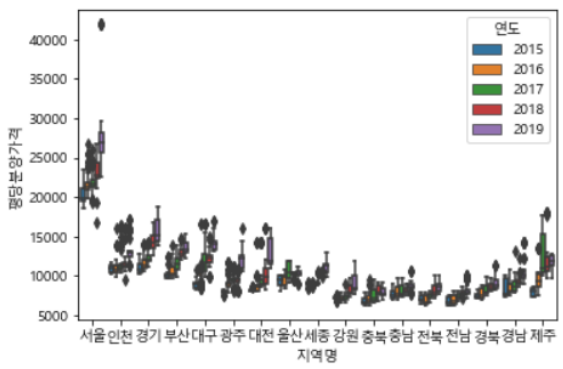
4. boxplot과 lineplot 그려보기

※ Seaborn Example Gallery [외부링크↗]

4-1. boxplot 그리기

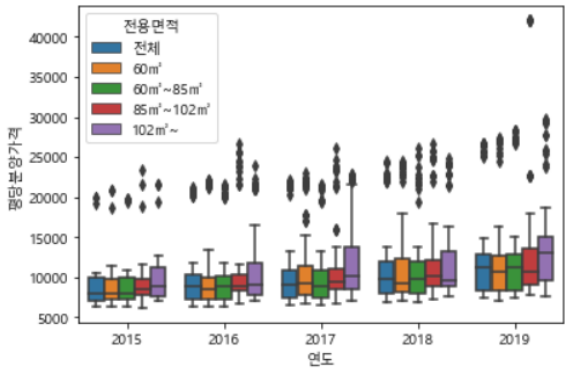
```
# 연도별 평당분양가격을 boxplot으로 그려봅니다.
sns.boxplot(data=df_last, x='연도', y='평당분양가격')

# hue : 카테고리가 적을때 사용하도록 (많으면 읽기 어려움)
# sns.boxplot(data=df_last, x='지역명', y='평당분양가격', hue='연도')
# boxplot에서 점(dot)은 이상치를 표현함 (상자수염그림 링크 참조)
```



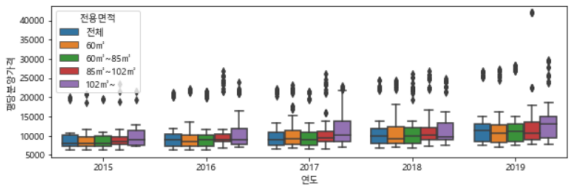


```
# 연도별 평당분양가격을 전용면적별로 보기
sns.boxplot(data=df_last, x='연도', y='평당분양가격', hue='전용면적')

# plt.figure(figsize=(10,3))
# (우) figsize 조정
```

<AxesSubplot:xlabel='연도', ylabel='평당분양가격'>

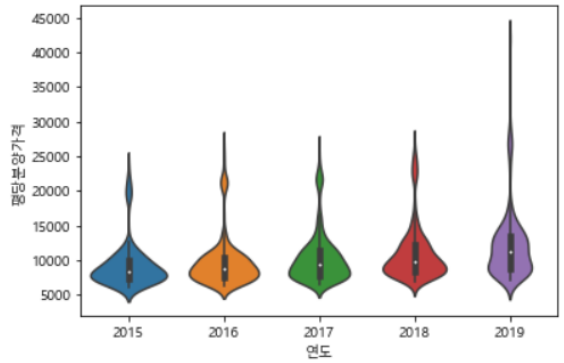


<AxesSubplot:xlabel='연도', ylabel='평당분양가격'>

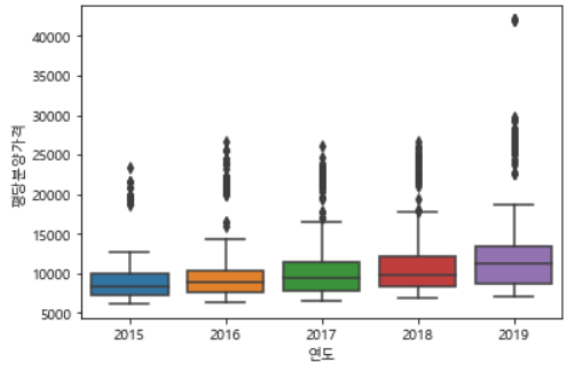


4-2. violinplot 그리기

```
# 바이올린 모양을 이용한 시각화 (boxplot의 단점을 보완)
# barplot vs boxplot vs violinplot
# 높낮이 vs 이상치 vs 높낮이, 이상치
sns.violinplot(data=df_last, x='연도', y='평당분양가격')

# (좌) violinpot    (우) boxplot
```



<AxesSubplot:xlabel='연도', ylabel='평당분양가격'>



※ boxplot의 단점 [외부링크↗]

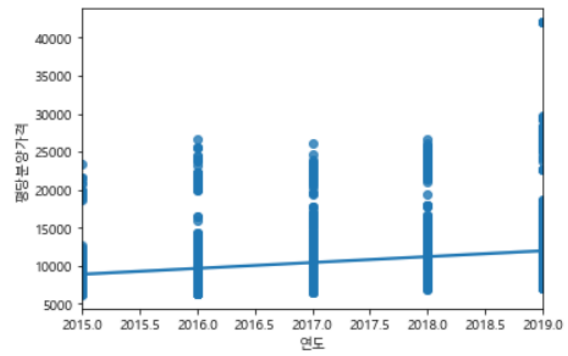
※ anscombe's quartet [외부링크↗]

5. lmplot과 swarmplot 그려보기

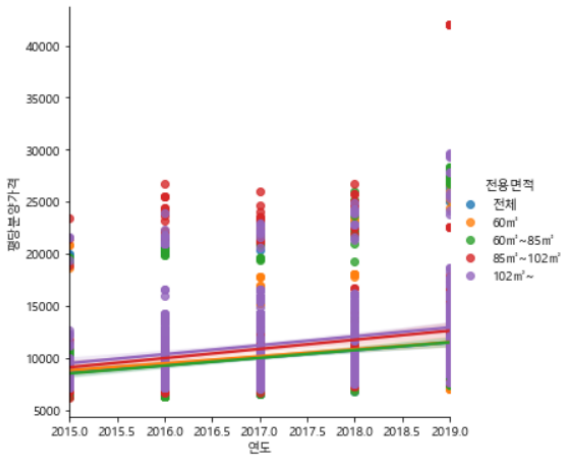
5-1. lmplot 그리기

```
# 연도별 평당 분양가격 그려보기
# regplot - 회귀선(regression)

# (좌) regplot - hue가 없음
# sns.regplot(data=df_last, x='연도', y='평당분양가격')

# (우) lmplot
sns.lmplot(data=df_last, x='연도', y='평당분양가격', hue='전용면적')
```

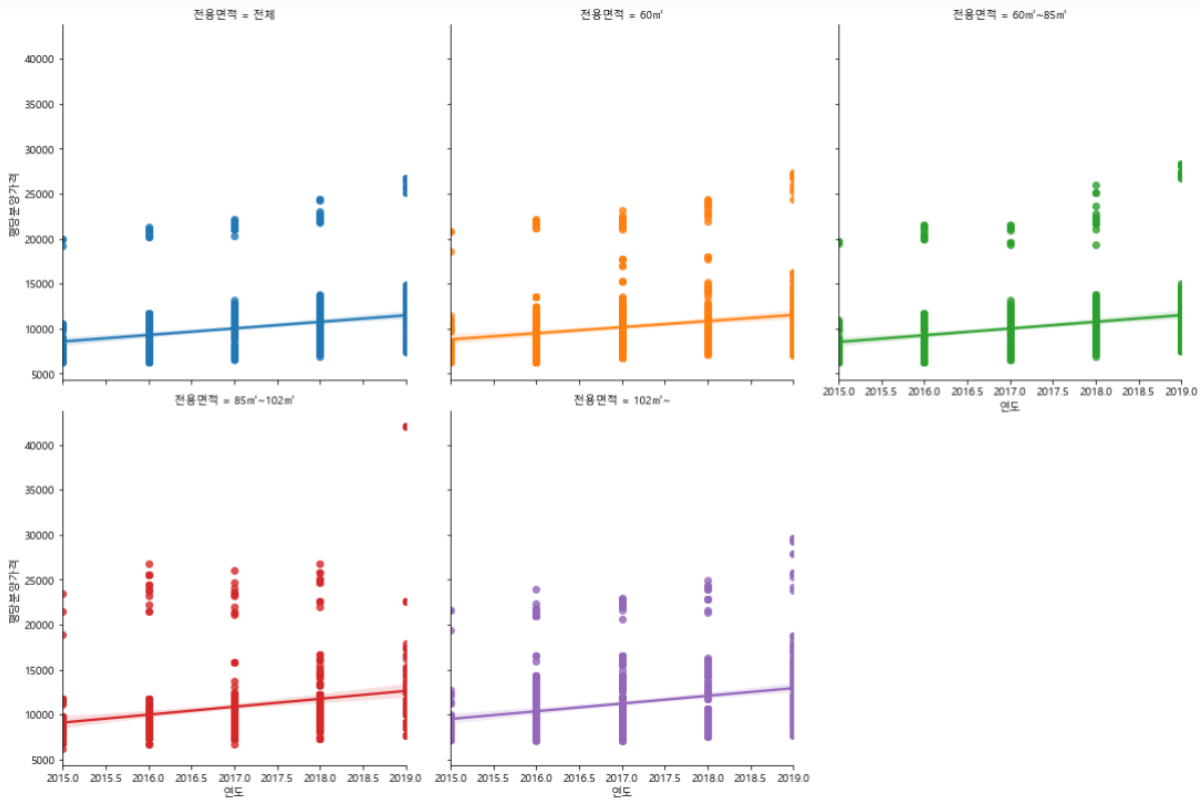
```
<AxesSubplot:xlabel='연도', ylabel='평당분양가격'>
```




```
<seaborn.axisgrid.FacetGrid at 0x15e66a58b20>
```




```
# 점이 겹쳐져서 보이면 파악이 어려우므로, 서브플롯을 그려보기
sns.lmplot(data=df_last, x='연도', y='평당분양가격',hue='전용면적', col='전용면적', col_wrap=3)

# x,y 축 둘다 수치형 데이터 일때 적합 but 연도는 수치보다 카테고리형 데이터로 봐야함
```

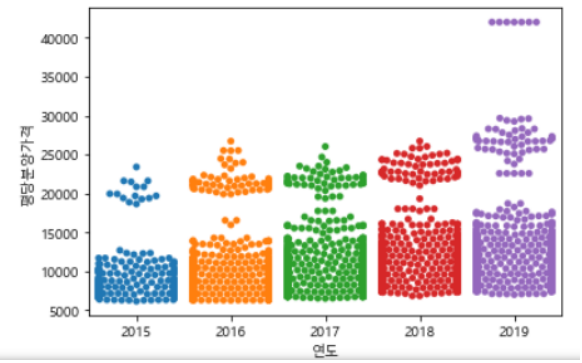


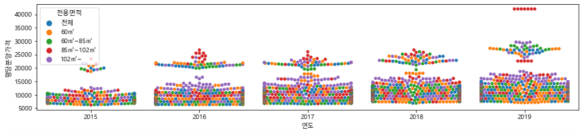
5-2. swarmplot 그리기

```
# 연도별 평당분양가격을 swarmplot 그리기
# swarmplot은 범주형(카테고리) 데이터의 산점도를 표현하기에 적합
sns.swarmplot(data=df_last, x='연도', y='평당분양가격')
```

```
# (우) 그래프 간격 조정, 전용면적별로 색상표현
# sns.swarmplot(data=df_last, x='연도', y='평당분양가격', hue='전용면적')
# 많은 데이터를 swarmplot으로 그리기에는 적합하지 않음 (적은 데이터)
```

```
<AxesSubplot:xlabel='연도', ylabel='평당분양가격'>
```





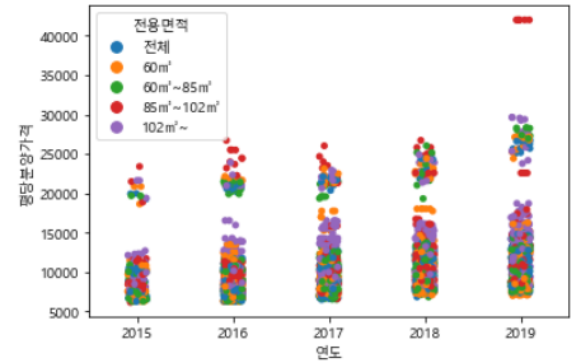
※ 산점도를 표현하기 위해 점을 하나씩 찍어주기때문에 시간이 오래걸리고 처리속도가 떨어지며 아래와 같은 경고가 나타날 수 있음.

```
C:\Users\seon\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 60.4% of the points cannot be placed; yo
u may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\seon\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 82.5% of the points cannot be placed; yo
u may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\seon\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 72.9% of the points cannot be placed; yo
u may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\seon\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 73.4% of the points cannot be placed; yo
u may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\seon\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 71.7% of the points cannot be placed; yo
u may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

5-3. swarmplot을 stripplot으로 변경해서 그리기

```
# sns.stripplot(data=df_last, x='연도', y='평당분양가격', hue='전용면적')
```

```
<AxesSubplot:xlabel='연도', ylabel='평당분양가격'>
```



데이터 분석 후 시각화를 통해 데이터의 이상치, 경향등을 파악하고 가설을 세우고 검증할때 필요합니다. Pandas와 Seaborn의 차이에 대해서도 다시한번 생각해보세요

번외. 수치화된 데이터에서 이상치 찾아보기

```
# 평당 분양가격을 요약해봅니다
# df_last['평당분양가격']
df_last['평당분양가격'].describe()

# (좌) 평당분양가격 컬럼    (우) 평당분양가격 컬럼을 요약한 결과

# 중앙값, 3사분위값, 평균값 vs 최대값
```

```
0       19275.3
1       18651.6
2       19410.6
3       18879.3
4       19400.7
         ...
4330    12810.6
4331        NaN
4332    12863.4
4333        NaN
4334    11883.3
Name: 평당분양가격, Length: 4335, dtype: float64
```

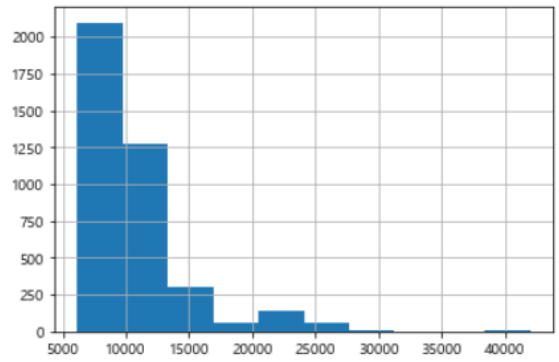
```
count     3957.000000
mean     10685.824488
std       4172.222780
min       6164.400000
25%       8055.300000
50%       9484.200000
75%      11751.300000
max      42002.400000
Name: 평당분양가격, dtype: float64
```

```
# 최대값을 max_price라는 변수로 지정
max_price = df_last['평당분양가격'].max()
df_last[df_last['평당분양가격'] == max_price]
```

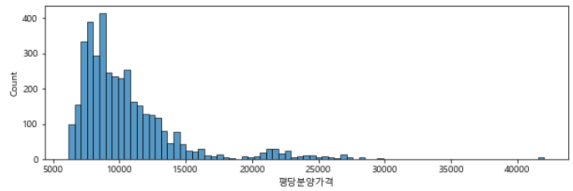
	지역명	연도	월	분양가격	평당분양가격	전용면적
3743	서울	2019	6	12728.0	42002.4	85m²~102m²
3828	서울	2019	7	12728.0	42002.4	85m²~102m²
3913	서울	2019	8	12728.0	42002.4	85m²~102m²
3998	서울	2019	9	12728.0	42002.4	85m²~102m²
4083	서울	2019	10	12728.0	42002.4	85m²~102m²
4168	서울	2019	11	12728.0	42002.4	85m²~102m²
4253	서울	2019	12	12728.0	42002.4	85m²~102m²

```
# 수치형 데이터 히스토그램 그리기 / 평당분양가격
# df_last['평당분양가격'].hist()

sns.distplot(df_last['평당분양가격'])

# distplot은 다음 버전 업데이트시 제거 예정, histplot을 사용하길 권고하는 오류
'''
C:\Users\seon\anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function and
  warnings.warn(msg, FutureWarning)
'''

# 결측치가 없는 데이터에서 평당분양가격만 가져옵니다. 그리고 price라는 변수에 담습니다.
# .loc[행]
# .loc[행, 열]
price = df_last.loc[df_last['평당분양가격'].notnull(), '평당분양가격']
plt.figure(figsize=(10,3))
sns.histplot(price)

# (좌) 히스토그램    (우) histplot
```

<AxesSubplot:>



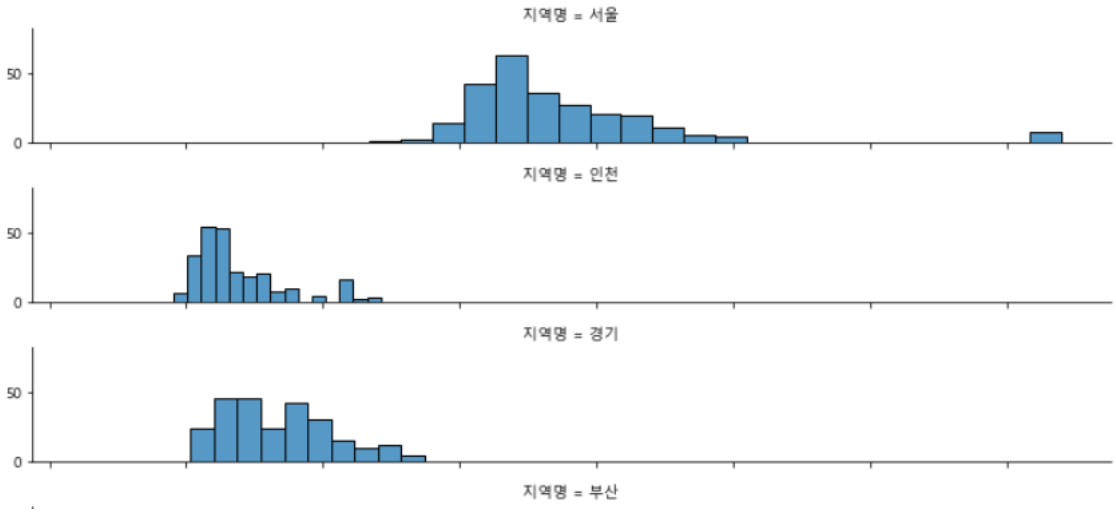
<AxesSubplot:xlabel='평당분양가격', ylabel='Count'>




```
# histplot을 서브플롯 그리기 (여러개)
g = sns.FacetGrid(df_last, row='지역명', height=1.5, aspect=5)
g.map(sns.histplot, '평당분양가격')
```

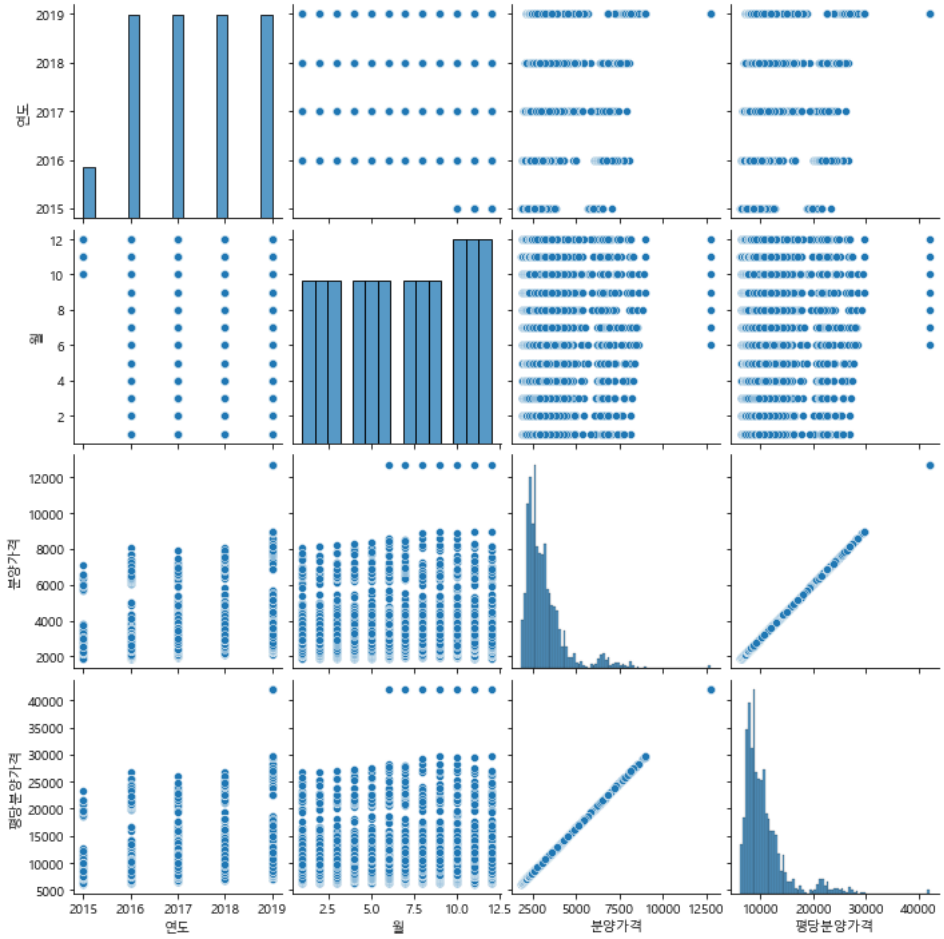
<seaborn.axisgrid.FacetGrid at 0x15e76436550>

<Figure size 720x216 with 0 Axes>



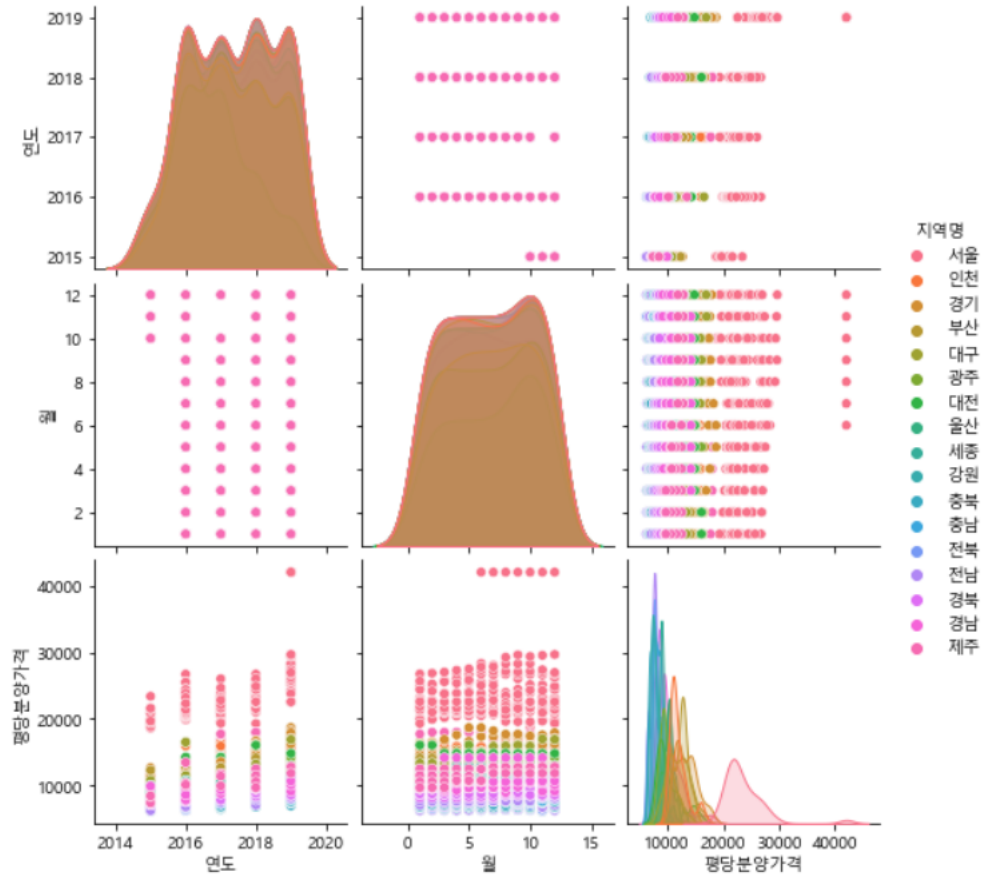

```
# pairplot 그려보기
sns.pairplot(df_last)
```

```
<seaborn.axisgrid.PairGrid at 0x15e7f2626d0>
```




```
# loc[행]
# loc[행, 열]
df_last_notnull = df_last.loc[df_last['평당분양가격'].notnull(), ['연도', '월','평당분양가격','지역명','전용면적']]
sns.pairplot(df_last_notnull, hue='지역명')
```

```
<seaborn.axisgrid.PairGrid at 0x15e02a74e80>
```




```
# 전용 면적별 value_counts를 사용해 데이터를 집계하기
df_last['전용면적'].value_counts()

'''
60㎡          867
전체           867
102㎡~        867
60㎡~85㎡      867
85㎡~102㎡     867
Name: 전용면적, dtype: int64
'''

# 2015년 8월 이전 데이터 df_first에는 전용면적이 없음.
```