```
!git clone https://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@github.com/a2ran/yonsei_bigdata_comp.
```

```
Cloning into 'yonsei_bigdata_comp'...
remote: Enumerating objects: 250, done.
remote: Counting objects: 100% (49/49), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 250 (delta 20), reused 29 (delta 6), pack-reused 201
Receiving objects: 100% (250/250), 119.85 MiB | 7.13 MiB/s, done.
Resolving deltas: 100% (38/38), done.
Updating files: 100% (368/368), done.
```

```
!ls ./yonsei_bigdata_comp/전처리데이터
```

```
'회원정보(CRM.CR_CUSTOMER).csv'      data1.csv   KICC_매출내역.csv
'포인트적립(CRM.CR_POINT_ADD).csv'   data2.csv   가맹점관리_KICC.xlsx
 월별_매출건수.csv                   data3.csv   readme.md
 일별_매출건수.csv                   data4.csv  '월별매출속보(SANL104).csv'
 일별_순매출.csv                     data5.csv  '매출코드(SCON002).csv'
 월별_순매출.csv                     data6.csv   임대계약관리_후지쯔.xlsx
 일별_총매출.csv                     data7.csv
 월별_총매출.csv                     data8.csv
```

```
!apt-get -qq install fonts-nanum
```

```
Selecting previously unselected package fonts-nanum.
(Reading database ... 120511 files and directories currently installed.)
Preparing to unpack .../fonts-nanum_20200506-1_all.deb ...
Unpacking fonts-nanum (20200506-1) ...
Setting up fonts-nanum (20200506-1) ...
Processing triggers for fontconfig (2.13.1-4.2ubuntu5) ...
```

```
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.pylab as pylab
import seaborn as sns

%matplotlib inline
mpl.style.use('ggplot')
sns.set_style('white')
pylab.rcParams['figure.figsize'] = 6,4

import matplotlib.font_manager as fm
!apt-get -qq install fonts-nanum
fe = fm.FontEntry(
    fname=r'/usr/share/fonts/truetype/nanum/NanumGothic.ttf',
    name='NanumGothic')
fm.fontManager.ttflist.insert(0, fe)
plt.rcParams.update({'font.size': 18, 'font.family': 'NanumGothic'})

import warnings
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd

df_1 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data1.csv')
df_2 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data2.csv')
df_3 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data3.csv')
df_4 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data4.csv')
df_5 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data5.csv')
df_6 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data6.csv')
df_7 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data7.csv')
df_8 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data8.csv')

df = pd.concat([df_1, df_2, df_3, df_4,
                df_5, df_6, df_7, df_8], ignore_index=True)

df = df.drop(['적립포인트', '포인트적립대상금액'], axis = 1)
df = df.dropna(axis = 0)
df.shape
```

```
(500540, 24)
```

```
# df['시간'] = pd.to_datetime(df['시간'])
# df = df[(df['시간'].dt.year == 2021) & (df['시간'].dt.month == 5)]

# df['시간'] = df['시간'].astype(np.int64)
# df.shape
```

```
(11947, 24)
```

```
quantitative = [f for f in df.columns if df.dtypes[f] != 'object']

qualitative = [f for f in df.columns if df.dtypes[f] == 'object']

quantitative
```

```
['카드번호', '매장코드', '총거래금액', '전용면적(m)', '공용면적(m)']
```

```
qualitative
```

```
['포인트적립ID',
 '승인ID',
 '회원ID',
 '매장명',
 '회원명',
 '성별',
 '생일',
 '회원자택주소1',
 '도',
 '구',
 '시간',
 '명판주소',
 '카테고리명',
 '카테고리명.1',
 '카테고리명.2',
 '층',
 '공간호실',
 '매장위치=동구분',
 '계약면적']
```

None of the quantitative variables has normal distribution

```
from scipy import stats

test_normality = lambda x: stats.shapiro(x.fillna(0))[1] < 0.01
normal = pd.DataFrame(df[quantitative])
normal = normal.apply(test_normality)
print(not normal.any())
```

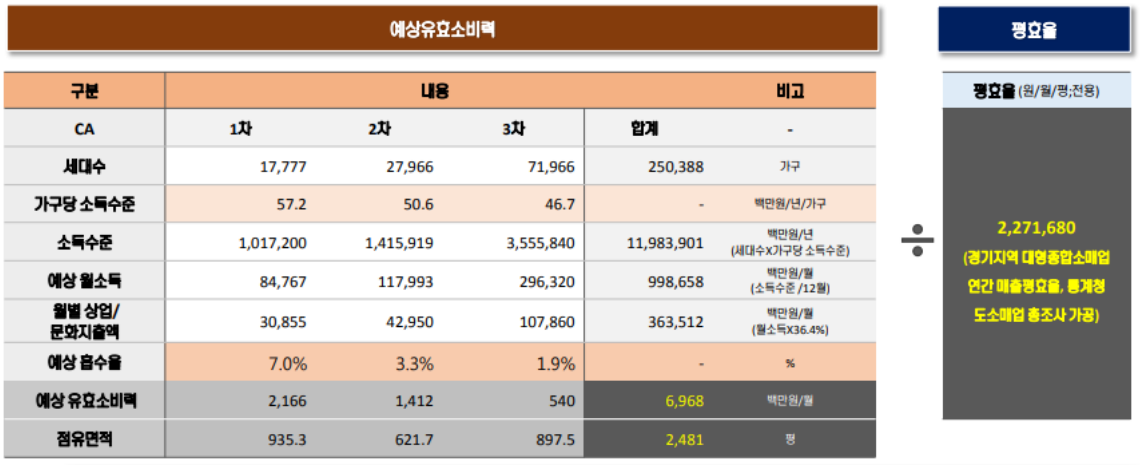
```
False
```

1. CA (Catchment Area)

- 1차 CA : 광교신도시 남부지역 (인구 비율 : 15%)
- 2차 CA : 광교신도시, 매탄동, 원천동 (인구 비율 : 24%)
- 3차 CA : 동수원, 기흥구 (인구 비율 : 61%)

산정식 : (CA 내 총 소비력 (총 소득액 x 상업/문화 지출 비율) * 흡수율)/평효율 = 개발 적정 면적

사업지의 적정 개발규모 추정을 위해 먼저 CA 내 총 소비력을 추정한 결과 월별 상업/문화관련 지출액이 약 1,820억원/월 수준임



예상유효소비력

구분	내용				비고
CA	1차	2차	3차	합계	-
세대수	17,777	27,966	71,966	250,388	가구
가구당 소득수준	57.2	50.6	46.7	-	백만원/년/가구
소득수준	1,017,200	1,415,919	3,555,840	11,983,901	백만원/년 (세대수x가구당 소득수준)
예상 월소득	84,767	117,993	296,320	998,658	백만원/월 (소득수준/12월)
월별 상업/문화지출액	30,855	42,950	107,860	363,512	백만원/월 (월소득X36.4%)
예상 흡수율	7.0%	3.3%	1.9%	-	%
예상 유효소비력	2,166	1,412	540	6,968	백만원/월
점유면적	935.3	621.7	897.5	2,481	평

÷

평효율

평효율 (원/월/평;전용)
2,271,680 (경기지역 대형종합소매업 연간 매출평효율, 통계청 도소매업 총조사 가공)

```
## 총매출 & 순매출

df_chong = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/일별_총매출.csv')
df_soon = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/일별_순매출.csv')
```

```
df_chong.shape
```

```
(3646, 37)
```

```
df_chong = df_chong.fillna(0)
df_soon = df_soon.fillna(0)
df_chong = df_chong.groupby(['년', '월']).agg({'합계' : 'sum'}).reset_index()
df_soon = df_soon.groupby(['년', '월']).agg({'합계' : 'sum'}).reset_index()
```

유효 소비력 추정

- 사업지 면적은 전용 3,700평으로 가정
- 경쟁시설 및 사업지의 상업시설 면적만을 기준으로 흡수율 가정
- 흡수율 = 사업지 공급면적(3,700평)÷ CA내 경쟁시설 공급면적

```
# CA 내 예상 월별 상업/문화지출액 = 6,968,000,000

print("월 총매출 median : {:,}원".format(df_chong['합계'].median()))
print("월 순매출 median : {:,}원".format(df_soon['합계'].median()))
```

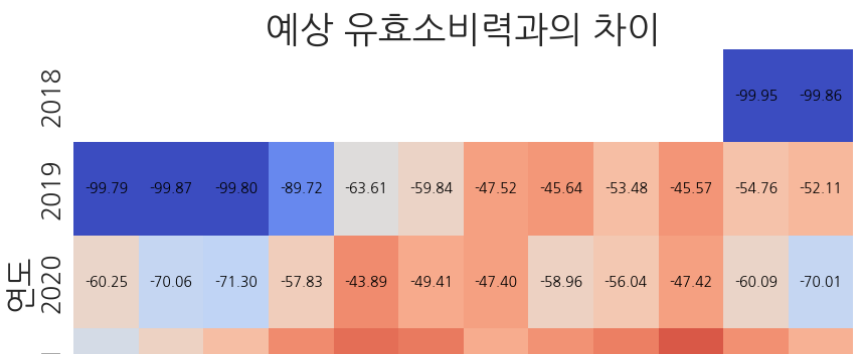
```
월 총매출 median : 3,385,706,152.5원
월 순매출 median : 3,089,063,189.0원
```

```
target_value = 6_968_000_000
df_chong['diff'] = (df_chong['합계'] - target_value) / target_value * 100

def color_map(val):
    if val < 0:
        color = 'blue'
        alpha = min(0.5, 1 - abs(val) / 100)
    else:
        color = 'red'
        alpha = min(0.5, val / 100)
    return color + f", {alpha:.2f}"

df_chong['color'] = df_chong['diff'].apply(color_map)
```

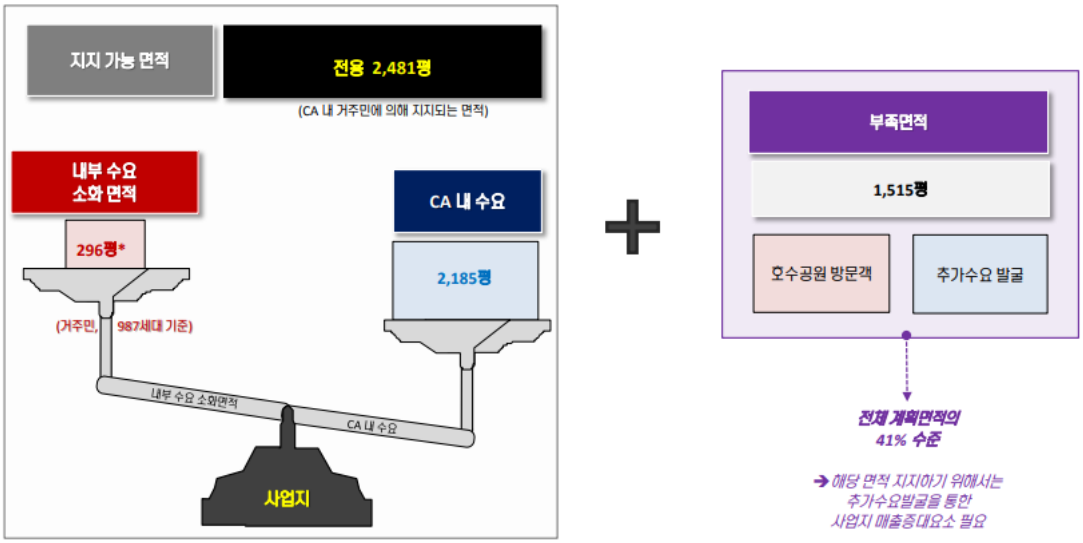
```
df_chong.head()
```

	년	월	합계	diff	color
0	2018	11	3340000	-99.952067	blue, 0.00
1	2018	12	9900700	-99.857912	blue, 0.00
2	2019	1	14932000	-99.785706	blue, 0.00
3	2019	2	9115000	-99.869188	blue, 0.00
4	2019	3	13636200	-99.804303	blue, 0.00

```
pivot_table = pd.crosstab(df_chong['년'], df_chong['월'], values=df_chong['diff'], aggfunc='sum')

plt.figure(figsize=(10, 6))
sns.heatmap(pivot_table, cmap='coolwarm', annot=True, fmt=".2f", cbar=False,
            annot_kws={"size": 10, "weight": "bold", "color": "black", "ha": "center", "va": "center"})
plt.title("예상 유효소비력과의 차이")
plt.xlabel("월")
plt.ylabel("연도")
plt.show()
```



사업지의 상권내 위계등을 고려할 때, 약 1,500평을 지지할 수 있는 수요가 부족한 것으로 도출되었으나, 일부 수요는 호수공원 방문객으로 충당할 수 있을 것으로 추정됨




```
print("월 지지가능 면적 : 전용 {:,.2f}평".format(df_chong['합계'].median()/2_261_680))
```

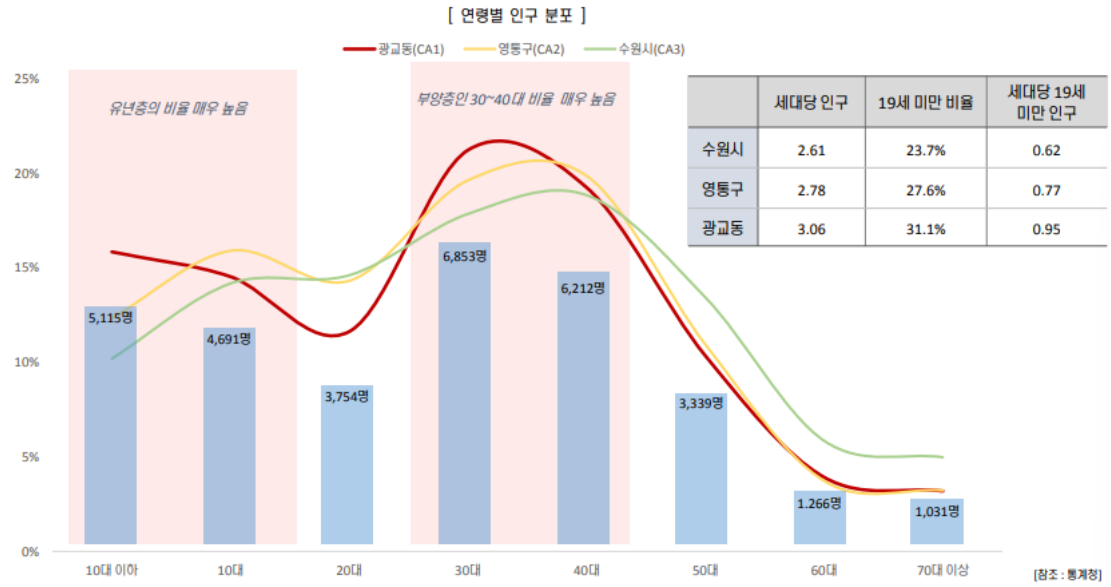
```
월 지지가능 면적 : 전용 1,496.99평
```

```
print("부족면적 : 전용 {:,.2f}평".format(3700-df_chong['합계'].median()/2_261_680))
```

```
부족면적 : 전용 2,203.01평
```

2. 전제 : "광교 신도시는 CA범위 내에서 가장 높은 생활 수준을 가질 것이다"

- 아파트 및 인근도시 소득을 통해 추정된 사업지 인근의 연소득은 약 6,600만원(550만원/월)
- CA1인 광교신도시의 세대당 인구는 3.06명, 10대 자녀가 있는 30~40대 부부로 구성된 가구의 비율이 높음
- 30대 비율과 연령대 기준으로 보면 광교(CA1), 영통(CA2), 수원시(CA3) 순서로 맞벌이 비율이 낮을 것으로 예상되며...
- 평형대를 감안하더라도 CA1, CA2, CA3 비율로 맞벌이 비율이 낮을 것으로 추정됨
- 사업지 가구소득 분석 결과 광교신도시(CA1)는 예비부유층으로 도시고소득의 라이프스타일을 추종하고, 영통구(CA2)는 전형적 중산층 및 수원시(CA3)는 무관심형 중산층으로 CA1 대비 상대적으로 보수적인 성향을 보일 것임
- 1차 CA인 광교신도시는 보다 트렌디하고 높은 Grade의 상품을 선호할 것이며, 상권이 커질수록 Moderate를 중심으로 선호도가 분포하게 됨 CA1,2,3이 모두 겹치는 부분은 평균보다 약간 높은 수준의 Grade와 Trendy 수준임



	세대당 인구	19세 미만 비율	세대당 19세 미만 인구
수원시	2.61	23.7%	0.62
영통구	2.78	27.6%	0.77
광교동	3.06	31.1%	0.95

3. 상업시설 적합성

- 우리 사업지가 추구하는 '라이프스타일 센터'란 부유한 근린주거지역에 가깝게 위치하여 쇼핑과 엔터테인먼트를 제공하면서 문화와 지역커뮤니티 기능을 제공하는 공간임
- 즉, 광교신도시 고소득 거주민의 소비행태에 적합한 매장과 필수 여가시설을 배치하여 높은 수익성을 추구하는 동시에, 이용객에서 우리 사업지가 의도하는 라이프스타일을 제안해 줄 수 있는 공간이 될 것임

지역 내 수요확보

- Retail : 상품구성은 단순화하되 가격 다양성을 확보
- F&B : 가격이나 브랜드 중심이 아닌 컨텐츠 또는 테마 중심 예) 안정성을 살린 맛집(전체 식재료의 유기농 및 무항생제 사용)
- Service : 일상 이용이 편리한 호수공원 공연장과 시너지 효과가 가능한 문화시설 지역 내 부족한 경제성 있는 문화소비공간 전략적 배치

세부사항

- 외식에서 선택하는 메뉴는 한식이 큰 비중을 차지하지만, 실제로 광교 아브뉴프랑에 도입된 한식의 비율은 상대적으로 매우 낮음 빈번한 방문을 유도하는 사업지의 특성상 한식 비율을 높이는 도입 방향이 필요함
- 사람들이 주로 이용하는 외식업종에서 한식의 경우 일반적인 날 68%, 특별한 날 56.4%의 이용률을 보임
- 메인 타깃인 30~40대 중에서 주요 소비층이 필요로 하는 잠재된 수요를 파악
- 주력 업종 외 키즈, F&B, 공방 등 적절히 mix하여 빈번한 방문 유도

업종	구성 비율	비고
Retail	40~60%	Anchor 비율 포함
F&B	25~35%	효율 감안하여 증가 가능
Ent.	5~10%	상황에 따라 F&B 등으로 전환 (예. 영화관 불가시)
Service & etc.	5~15%	-

```
df_chong = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/일별_총매출.csv')
df_soon = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/일별_순매출.csv')
```

```
df.columns
```

```
Index(['포인트적립ID', '카드번호', '승인ID', '매장코드', '회원ID', '총거래금액', '매장명', '회원명', '성별',
       '생일', '회원자택주소1', '도', '구', '시간', '명판주소', '카테고리명', '카테고리명.1', '카테고리명.2',
       '층', '공간호실', '매장위치=동구분', '계약면적', '전용면적(m)', '공용면적(m)'],
      dtype='object')
```

1. 구성비율

```
## 근생, 단기, 락, 주 == Service & etc.

np.unique(df['카테고리명.1'])
```

```
array(['근생(Service)', '단기(Popup)', '락(Wants)', '식(F&B)', '의(Retail)',
       '주(Lifestyle)'], dtype=object)
```

```
np.unique(df['카테고리명.2'])
```

```
array(['가구', '근생', '마트', '베이커리', '분식', '뷰티', '스튜디오', '양식', '여성', '일식',
       '잡화', '주류/기타', '중식', '카페', '키즈', '판매', '펫', '편집샵', '한식'],
      dtype=object)
```

```
gooseong = df.groupby(['매장명', '카테고리명.1']).agg({'포인트적립ID' : 'count'}).reset_index()
gooseong = gooseong.groupby(['카테고리명.1']).agg({'매장명' : 'count'}).reset_index()
gooseong
```

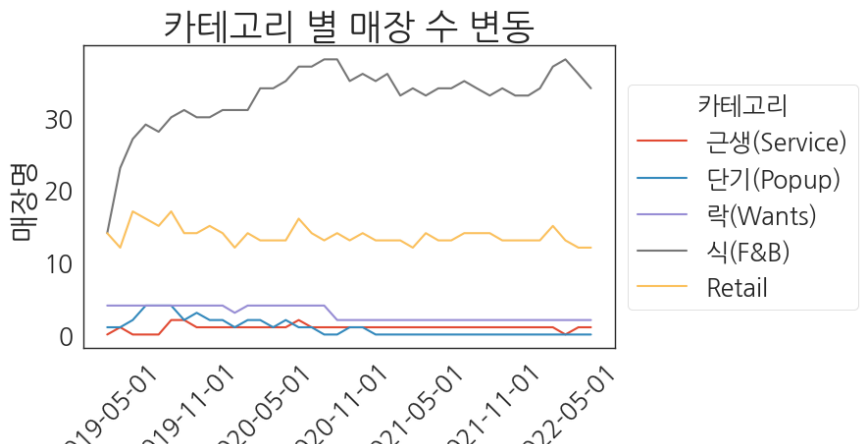
	카테고리명.1	매장명
0	근생(Service)	3
1	단기(Popup)	9
2	락(Wants)	6
3	식(F&B)	62
4	의(Retail)	13
5	주(Lifestyle)	15

- 식(F&B) : 62 = 60%
- 의(Retail) + 주(Lifestyle) :28 = 30%
- 근생(Service) : 3 = 3%

하지만 시간에 따라서 폐업 + 개장하는 매점도 존재.

월별로 각 카테고리의 증감 계산

```
df['시간'] = pd.to_datetime(df['시간'])
df['시간'] = df['시간'].apply(lambda x: x.strftime('%Y-%m'))
gooseong_1 = df.groupby(['매장명', '카테고리명.1', '시간']).agg({'포인트적립ID' : 'count'}).reset_index()
gooseong_1 = gooseong_1.groupby(['카테고리명.1', '시간']).agg({'매장명' : 'count'}).reset_index()


gooseong_1['시간'] = pd.to_datetime(gooseong_1['시간'])

cross_tab = pd.crosstab(gooseong_1['카테고리명.1'], gooseong_1['시간'], values=gooseong_1['매장명'], aggfunc='sum')
cross_tab_filled = cross_tab.fillna(0)

retail_row = cross_tab_filled.loc[['의(Retail)', '주(Lifestyle)']].sum()
cross_tab_filled.drop(index=['의(Retail)', '주(Lifestyle)'], inplace=True)
cross_tab_filled.loc['Retail'] = retail_row

plt.figure(figsize=(7, 4))
for idx, row in cross_tab_filled.iloc[:,:-1].iterrows():
    sns.lineplot(x=cross_tab_filled.columns[:-1], y=row, label=idx)

plt.xlabel('시간')
plt.ylabel('매장명')
plt.xticks(cross_tab_filled.columns[::6], rotation=45)
plt.title('카테고리 별 매장 수 변동')
plt.legend(title='카테고리', loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()
```



변동성이 높은 식(F%B) 심층분석

- 외식에서 선택하는 메뉴는 한식이 큰 비중을 차지하지만, 실제로 광교 아브뉴프랑에 도입된 한식의 비율은 상대적으로 매우 낮음 빈번한 방문을 유도하는 사업지의 특성상 한식 비율을 높이는 도입 방향이 필요함

```
FB = df[df['카테고리명.1'] == '식(F&B)']
FB.shape
```

```
(435766, 24)
```

```
FB.columns
```

```
Index(['포인트적립ID', '카드번호', '승인ID', '매장코드', '회원ID', '총거래금액', '매장명', '회원명', '성별',
       '생일', '회원자택주소1', '도', '구', '시간', '명판주소', '카테고리명', '카테고리명.1', '카테고리명.2',
       '층', '공간호실', '매장위치=동구분', '계약면적', '전용면적(m)', '공용면적(m)'],
      dtype='object')
```

```
FB = FB.groupby(['매장명', '카테고리명.2']).agg({'회원ID' : 'count', '총거래금액' : 'sum'}).reset_index()
```

```
FB
```

	매장명	카테고리명.2	회원ID	총거래금액
0	게방식당	한식	3316	111259800
1	게방찬	판매	1311	38829287
2	겐짱타코야키	일식	231	1838950
3	겟올라잇	양식	80	13346000
4	굿데이메이트	카페	1112	13272140
...	...	...	...	...
58	커피반점	주류/기타	4287	66136400
59	타이거슈가	카페	16397	134627284
60	평양일미	한식	77	3478000
61	풍류랑	판매	7329	165013848
62	핑거스시	일식	850	15102700

63 rows × 4 columns

```
## 한식이 제일 많다
```

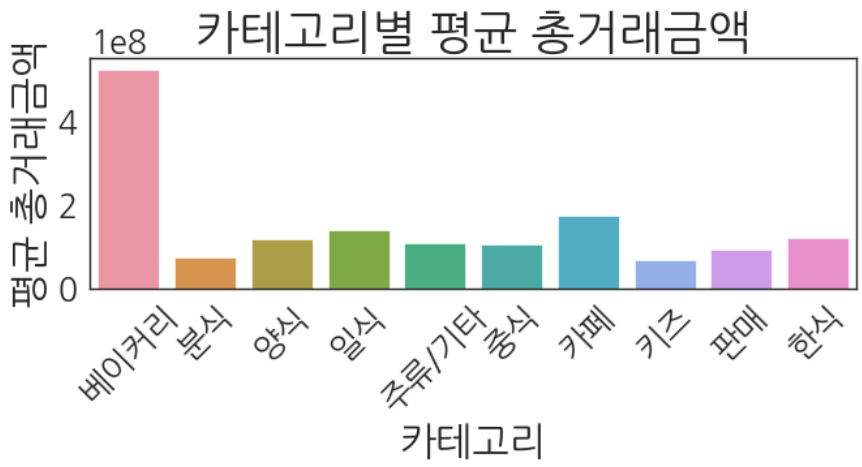
```
FB['카테고리명.2'].value_counts()
```

```
한식      19
일식      10
카페      10
양식       7
주류/기타    7
판매       3
베이커리     3
분식       2
키즈       1
중식       1
Name: 카테고리명.2, dtype: int64
```

```
## 그러면 매점 하나당 매출 정도는?

category_counts = FB['카테고리명.2'].value_counts()
FB_profit = FB.groupby(['카테고리명.2']).agg({'총거래금액' : 'sum'}).reset_index()
FB_profit = FB_profit.merge(category_counts, left_on='카테고리명.2', right_index=True, suffixes=('_sum', '_count'))
FB_profit['총거래금액_평균'] = FB_profit['총거래금액'] / FB_profit['카테고리명.2_count']
FB_profit.drop(['카테고리명.2_sum', '총거래금액', '카테고리명.2_count'], axis=1, inplace=True)

FB_profit
```

	카테고리명.2	총거래금액_평균
0	베이커리	5.239021e+08
1	분식	7.509273e+07
2	양식	1.181933e+08
3	일식	1.405512e+08
4	주류/기타	1.066665e+08
5	중식	1.061642e+08
6	카페	1.727803e+08
7	키즈	6.793380e+07
8	판매	9.342906e+07
9	한식	1.198071e+08

```
plt.figure(figsize=(7, 4))
sns.barplot(x='카테고리명.2', y='총거래금액_평균', data=FB_profit)

plt.xlabel('카테고리')
plt.ylabel('평균 총거래금액')
plt.title('카테고리별 평균 총거래금액')
plt.xticks(rotation=45)

plt.tight_layout()
plt.show()
```



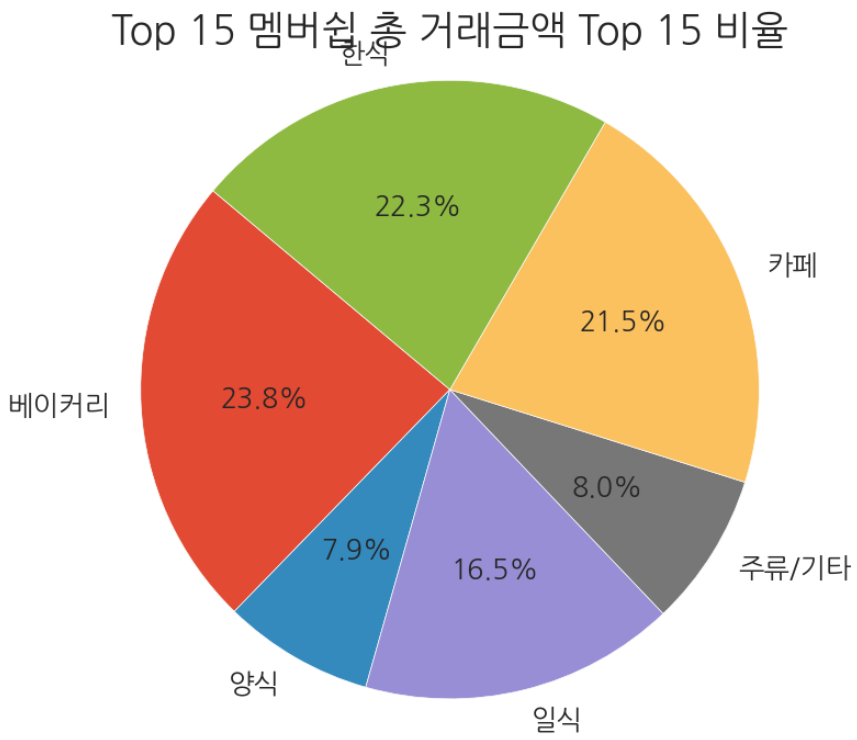
- 높은 기대를 받은 한식이 생각이상으로 부진중이다.

  사실상 빵집 원툴인가.?

```
## 멤버쉽 순위 top 15
## 아우어베이커리가 카페로 분류됨..!

FB_top = FB.sort_values(by = ['총거래금액'], ascending = False)
FB_top.head(15)
```

	매장명	카테고리명.2	회원ID	총거래금액
21	밀도	베이커리	95918	1412226655
42	아우어베이커리	카페	57416	845152837
11	도쿄등심	일식	2018	486638961
14	동트는농가	한식	14084	454899320
56	책발전소	카페	31636	426693748
37	스시덴고쿠	일식	5295	315584930
55	지로나	양식	2835	290742220
6	낙찌하다	한식	7611	278302640
7	더부쓰	주류/기타	10919	267768580
22	바오닦	한식	19612	219707170

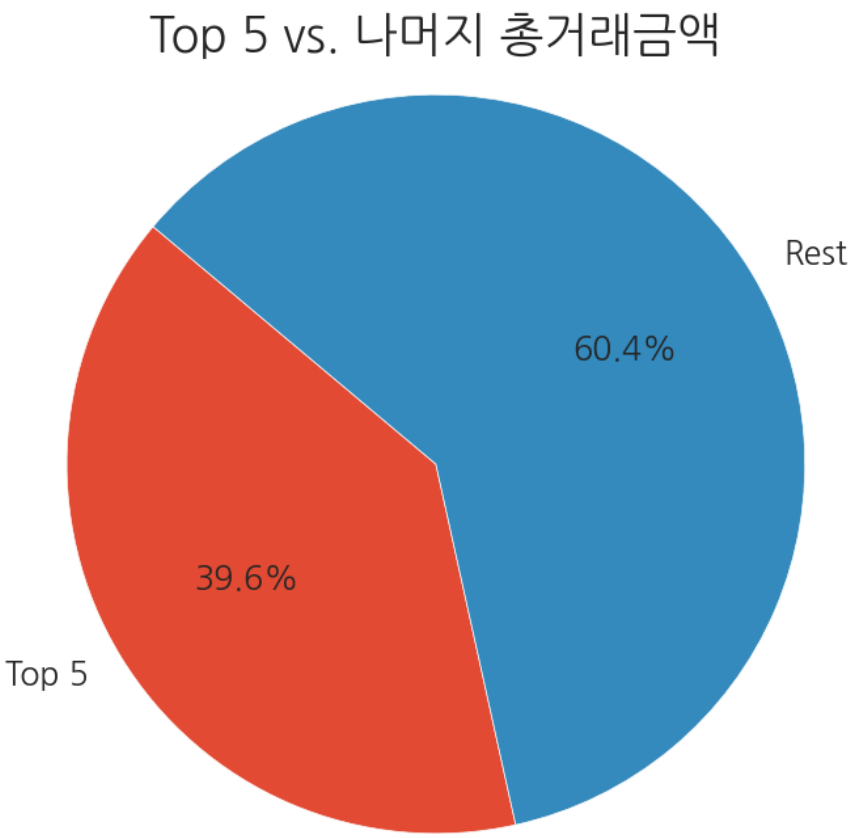
```
top_categories_sum = FB_top.head(15).groupby('카테고리명.2')['총거래금액'].sum()

plt.figure(figsize=(8, 8))
plt.pie(top_categories_sum, labels=top_categories_sum.index, autopct='%1.1f%%', startangle=140)
plt.title('Top 15 멤버쉽 총 거래금액 Top 15 비율')
plt.axis('equal')

plt.show()
```




```
FB_top.tail(10)
```

	매장명	카테고리명.2	회원ID	총거래금액
53	이유의계절	카페	824	13906830

```
top_5_total = FB_top.head(5)['총거래금액'].sum()

rest_total = FB_top.iloc[5:]['총거래금액'].sum()

plt.figure(figsize=(8, 8))
plt.pie([top_5_total, rest_total], labels=['Top 5', 'Rest'], autopct='%1.1f%%', startangle=140)
plt.title('Top 5 vs. 나머지 총거래금액')
plt.axis('equal')

plt.show()
```



멤버쉽이 아닌 전체 총매출 기준으로는..

```
df = df[df['카테고리명.1'] == '식(F&B)']
all = df.groupby(['매장명', '카테고리명.2']).agg({'회원ID' : 'count', '총거래금액' : 'sum'}).reset_index()
all = all[['매장명', '카테고리명.2']]
all = all.rename(columns = {'매장명' : '명', '카테고리명.2' : '서브_카테고리'})
df_chong = df_chong.fillna(0)
df_chong = df_chong.merge(all, on = '명', how = 'left')
df_chong = df_chong.dropna(axis = 0)


df_chong = df_chong.groupby(['명', '서브_카테고리']).agg({'매장코드' : 'count', '합계' : 'sum'}).reset_index()


df_chong['서브_카테고리'].value_counts()
```

```
한식       19
일식       10
카페       10
양식        7
주류/기타     7
판매        3
베이커리      3
분식        2
키즈        1
중식        1
Name: 서브_카테고리, dtype: int64
```

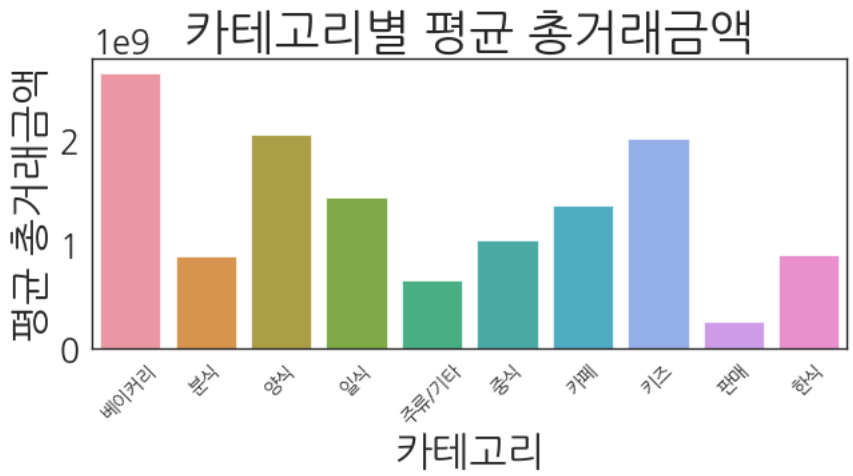
```
## 매장 하나당 매출 정도는?
```

```
category_counts = df_chong['서브_카테고리'].value_counts()
df_profit = df_chong.groupby(['서브_카테고리']).agg({'합계' : 'sum'}).reset_index()
df_profit = df_profit.merge(category_counts, left_on='서브_카테고리', right_index=True, suffixes=('_sum', '_count'))
df_profit['총거래금액_평균'] = df_profit['합계'] / df_profit['서브_카테고리_count']
df_profit.drop(['서브_카테고리_sum', '합계', '서브_카테고리_count'], axis=1, inplace=True)
```

```
df_profit
```

	서브_카테고리	총거래금액_평균
0	베이커리	2.661653e+09
1	분식	8.938765e+08
2	양식	2.059584e+09
3	일식	1.463180e+09
4	주류/기타	6.605222e+08
5	중식	1.051063e+09
6	카페	1.386806e+09
7	키즈	2.028753e+09
8	판매	2.534971e+08
9	한식	9.037115e+08

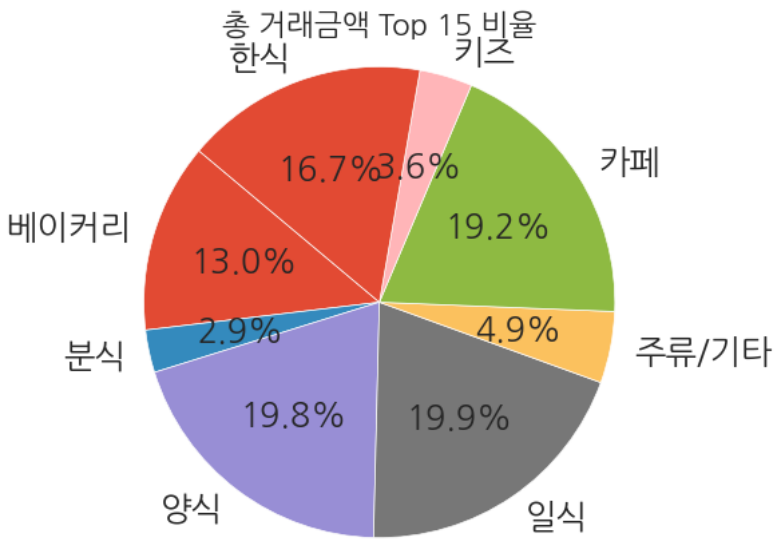
```
plt.figure(figsize=(7, 4))
sns.barplot(x='서브_카테고리', y='총거래금액_평균', data=df_profit)

plt.xlabel('카테고리')
plt.ylabel('평균 총거래금액')
plt.title('카테고리별 평균 총거래금액')
plt.xticks(rotation=45, fontsize = 10)

plt.tight_layout()
plt.show()
```




```
## 매출 순위 top 15
## 아우어베이커리가 카페로 분류됨..!

df_top = df_chong.sort_values(by = ['합계'], ascending = False)
df_top.head(15)
```

	명	서브_카테고리	매장코드	합계
11	도쿄등심	일식	39	9338555653
34	세상의모든아침	양식	39	8428564355
42	아우어베이커리	카페	43	7429618568
21	밀도	베이커리	42	7282005472
56	책발전소	카페	41	3349489677
14	돔트는농가	한식	40	3117679895

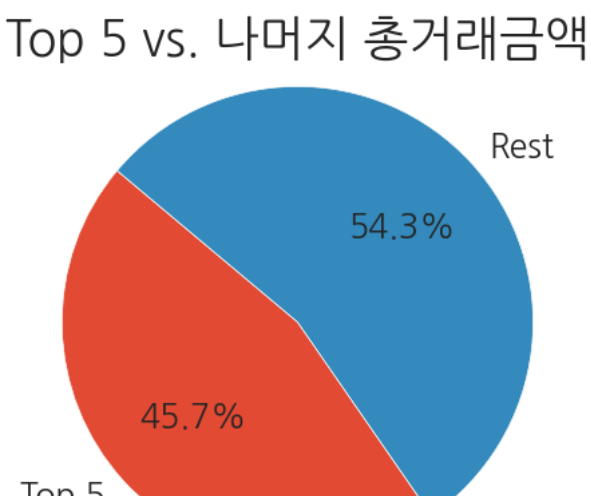
```
top_categories_sum = df_top.head(15).groupby('서브_카테고리')['합계'].sum()

plt.figure(figsize=(5, 5))
plt.pie(top_categories_sum, labels=top_categories_sum.index, autopct='%1.1f%%', startangle=140)
plt.title('총 거래금액 Top 15 비율', fontsize = 16)
plt.axis('equal')

plt.show()
```




```
FB_top.tail(10)
```

	매장명	카테고리명.2	회원ID	총거래금액
53	이유의계절	카페	824	13906830
3	겟올라잇	양식	80	13346000
4	굿데이메이트	카페	1112	13272140
29	빗소리	주류/기타	292	11733400
28	비아티하우스	카페	873	10628850
49	우마이야	일식	362	10265500
35	셋째집	한식	63	5504000
19	램브란트	한식	7	5370000
60	평양일미	한식	77	3478000
2	겐짱타코야키	일식	231	1838950

```
top_5_total = df_top.head(5)['합계'].sum()

rest_total = df_top.iloc[5:]['합계'].sum()

plt.figure(figsize=(5, 5))
plt.pie([top_5_total, rest_total], labels=['Top 5', 'Rest'], autopct='%1.1f%%', startangle=140)
plt.title('Top 5 vs. 나머지 총거래금액')
plt.axis('equal')

plt.show()
```



결론 :

사람들이 주로 이용하는 외식업종에서 한식의 경우 일반적인 날 68%, 특별한 날 56.4%의 이용률을 보임

--> 한식이 멤버쉽 & 총매출 기준으로 anchor 역할 수행에 실패함.

명제 2: 메인 타깃인 30~40대 중에서 주요 소비층이 필요로 하는 잠재된 수요를 파악

```
## 총매출 X 나이 변수가 있는 멤버십 데이터 사용

df_1 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data1.csv')
df_2 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data2.csv')
df_3 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data3.csv')
df_4 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data4.csv')
df_5 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data5.csv')
df_6 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data6.csv')
df_7 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data7.csv')
df_8 = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/data8.csv')

df = pd.concat([df_1, df_2, df_3, df_4,
                df_5, df_6, df_7, df_8], ignore_index=True)


df.columns
```

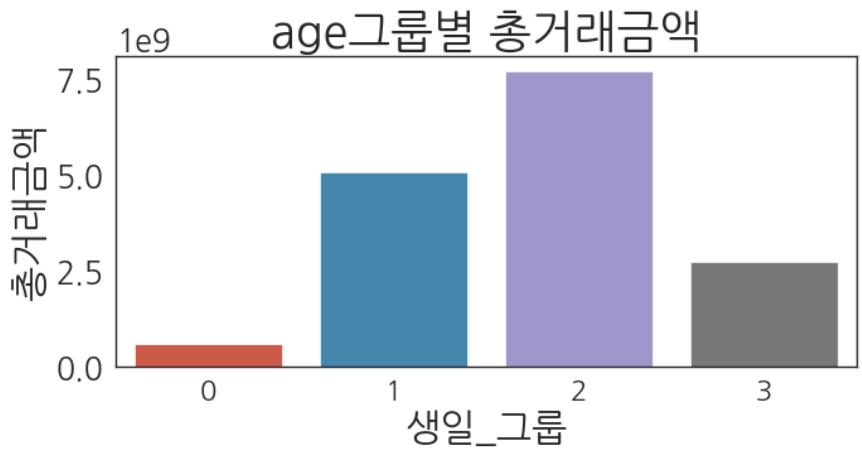
```
Index(['포인트적립ID', '카드번호', '적립포인트', '승인ID', '매장코드', '회원ID', '총거래금액',
       '포인트적립대상금액', '매장명', '회원명', '성별', '생일', '회원자택주소1', '도', '구', '시간',
       '명판주소', '카테고리명', '카테고리명.1', '카테고리명.2', '층', '공간호실', '매장위치=동구분', '계약면적',
       '전용면적(m)', '공용면적(m)'],
      dtype='object')
```

```
df['생일'] = df['생일'].str.slice(0, 4)


np.unique(df['생일'].tolist())
```

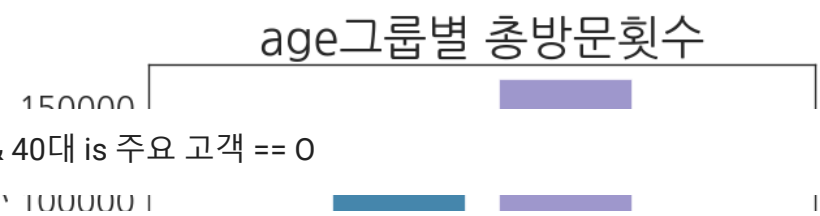
```
array(['1686', '1905', '1922', '1927', '1931', '1933', '1935', '1938',
       '1940', '1941', '1942', '1943', '1944', '1945', '1946', '1947',
       '1948', '1949', '1950', '1951', '1952', '1953', '1954', '1955',
       '1956', '1957', '1958', '1959', '1960', '1961', '1962', '1963',
       '1964', '1965', '1966', '1967', '1968', '1969', '1970', '1971',
       '1972', '1973', '1974', '1975', '1976', '1977', '1978', '1979',
       '1980', '1981', '1982', '1983', '1984', '1985', '1986', '1987',
       '1988', '1989', '1990', '1991', '1992', '1993', '1994', '1995',
       '1996', '1997', '1998', '1999', '2000', '2001', '2002', '2003',
       '2004', '2005', '2006', '2007'], dtype='<U4')
```

```
## 20대 밑 : 1994 ~ : 0
## 30대 : 1984 ~ 1994 : 1
## 40대 : 1974 ~ 1984 : 2
## 40대 위 : ~ 1974 : 3

encode_year = {
    range(1984, 1994): 1,
    range(1974, 1984): 2,
    range(0, 1974): 3
}

encode_year = {year: label for rng, label in encode_year.items() for year in rng}
```

```
decode_year = {v: k for k, v in encode_year.items()}

df['생일'] = df['생일'].fillna('0').astype(str)
df['생일_그룹'] = df['생일'].astype(int).map(encode_year).fillna(0).astype(int)


df['생일_그룹'].unique()

      array([3, 2, 1, 0])


def asdf(st):
  return st[:10]

df['시간'] = df['시간'].apply(asdf)
df['시간'][0]

      '2019-05-01'


age1 = df.groupby(['생일_그룹']).agg({'총거래금액' : 'sum'}).reset_index()

plt.figure(figsize=(7, 4))
sns.barplot(x='생일_그룹', y='총거래금액', data=age1)

plt.xlabel('생일_그룹')
plt.ylabel('총거래금액')
plt.title('age그룹별 총거래금액')
plt.xticks(fontsize = 16)

plt.tight_layout()
plt.show()
```




```
age2 = df.groupby(['생일_그룹', '회원ID', '시간']).agg({'회원명' : 'count'}).reset_index()
age2 = age2.groupby(['생일_그룹']).agg({'회원ID' : 'count'}).reset_index()

plt.figure(figsize=(7, 4))
sns.barplot(x='생일_그룹', y='회원ID', data=age2)

plt.xlabel('생일_그룹')
plt.ylabel('총방문횟수')
plt.title('age그룹별 총방문횟수')
plt.xticks(fontsize = 16)

plt.tight_layout()
plt.show()
```



30대 & 40대 is 주요 고객 == O

30대 & 40대 재방문 하나?

1. 주소가 영통구 X (앨리웨이 주변 거주 X) 사람들이 재방문하는지 여부
2. 영통구 거주중인 인원의 방문 순위


```
# 30대

df_30 = df[df['생일_그룹'] == 1]


df_30.shape
```

```
(158674, 25)
```

```
df.columns
```

```
Index(['포인트적립ID', '카드번호', '승인ID', '매장코드', '회원ID', '총거래금액', '매장명', '회원명', '성별',
       '생일', '회원자택주소1', '도', '구', '시간', '명판주소', '카테고리명', '카테고리명.1', '카테고리명.2',
       '층', '공간호실', '매장위치=동구분', '계약면적', '전용면적(m)', '공용면적(m)', '생일_그룹'],
      dtype='object')
```

```
df_30['case1'] = np.where(df_30['구'] == '영통구', 0, 1)


## 전체 30대 멤버쉽 사람 수

df_case1 = df_30.groupby(['case1', '회원ID', '시간']).agg({'회원명' : 'count'}).reset_index()
df_case1 = df_case1.groupby(['case1', '회원ID']).agg({'시간' : 'count'}).reset_index()
df_case1 = df_case1.groupby(['case1']).agg({'회원ID' : 'count'}).reset_index()
df_case1
```

	case1	회원ID
0	0	3128
1	1	3161

```
## 재방문한 30대 멤버쉽 사람 수

df_case1 = df_30.groupby(['case1', '회원ID', '시간']).agg({'회원명' : 'count'}).reset_index()
df_case1 = df_case1.groupby(['case1', '회원ID']).agg({'시간' : 'count'}).reset_index()
df_case1 = df_case1[df_case1['시간'] > 1]
df_case1 = df_case1.groupby(['case1']).agg({'회원ID' : 'count'}).reset_index()
df_case1
```

	case1	회원ID
0	0	2697
1	1	1887

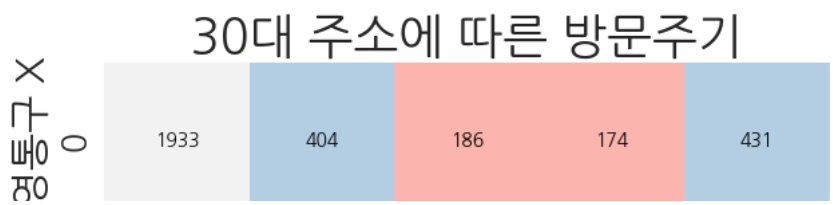
```
def classify_period(period):
    if pd.isnull(period): # 방문기록이 한 번뿐인 경우
        return 4
    elif (period <= 7) & (period > 0): # 일주일 이내
        return 0
    elif (period > 7) & (period <= 30): # 한 달 이내
        return 1
    elif (period > 30) & (period <= 90): # 세 달 이내
        return 2
    else: # 세 달 이상
        return 3


## 재방문주기

df_v = df_30.drop_duplicates(['회원ID', '시간'])
df_v['시간'] = pd.to_datetime(df_v['시간'])
df_v = df_v.sort_values(['회원ID', '시간'])
```

```
df_v['시간_차이'] = df_v.groupby('회원ID')['시간'].diff()

df_v['시간_차이'] = df_v['시간_차이'].dt.days
df_v['주기'] = df_v['시간_차이'].apply(classify_period)


df_v = df_v.groupby(['회원ID', 'case1', '성별', '주기']).agg({'회원명' : 'count'}).reset_index()
df_v = df_v.groupby(['회원ID', 'case1', '성별'])['주기'].min().reset_index()


def create_label(row):
    if row['case1'] == 0 and row['주기'] == 0:
        return 0
    elif row['case1'] == 0 and row['주기'] == 1:
        return 1
    elif row['case1'] == 0 and row['주기'] == 2:
        return 2
    elif row['case1'] == 0 and row['주기'] == 3:
        return 3
    elif row['case1'] == 0 and row['주기'] == 4:
        return 4
    elif row['case1'] == 1 and row['주기'] == 0:
        return 5
    elif row['case1'] == 1 and row['주기'] == 1:
        return 6
    elif row['case1'] == 1 and row['주기'] == 2:
        return 7
    elif row['case1'] == 1 and row['주기'] == 3:
        return 8
    elif row['case1'] == 1 and row['주기'] == 4:
        return 8
    else:
        return 9


df_v['label'] = df_v.apply(create_label, axis=1)
df_v
```

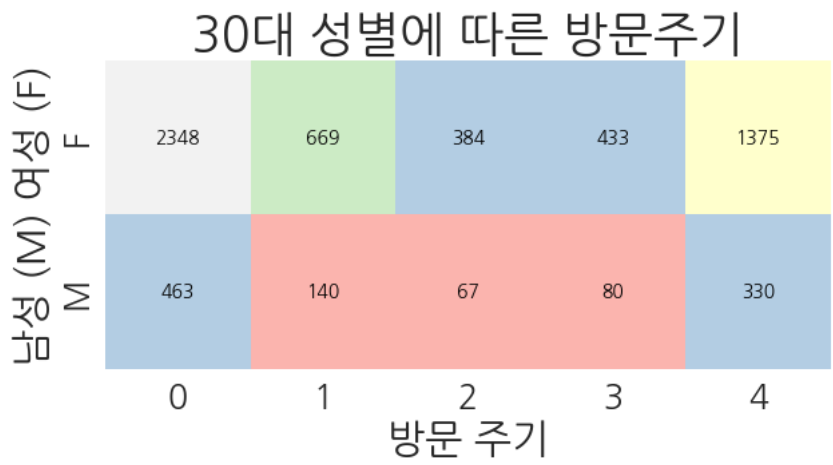
	회원ID	case1	성별	주기	label
0	CM0120190000000000057	1	F	0	5
1	CM0120190000000000058	0	F	0	0
2	CM0120190000000000059	1	F	0	5
3	CM0120190000000000062	0	F	1	1
4	CM0120190000000000073	0	F	0	0
...	...	...	...	...	...
6284	CM0120220000000025285	1	F	4	8
6285	CM0120220000000025316	0	F	4	4
6286	CM0120220000000025323	1	F	4	8
6287	CM0120220000000025329	1	F	4	8
6288	CM0120220000000025334	1	M	4	8

6289 rows × 5 columns

```
pivot_table = pd.crosstab(df_v['case1'], df_v['주기'], values=df_v['회원ID'], aggfunc='count')

plt.figure(figsize=(7, 3))
sns.heatmap(pivot_table, cmap='Pastel1', annot=True, fmt=".0f", cbar=False,
            annot_kws={"size": 10, "weight": "bold", "color": "black", "ha": "center", "va": "center"})

plt.title("30대 주소에 따른 방문주기")
plt.xlabel("방문 주기")
plt.ylabel("영통구 vs 영통구 X")
plt.show()
```




```
pivot_table = pd.crosstab(df_v['성별'], df_v['주기'], values=df_v['회원ID'], aggfunc='count')

plt.figure(figsize=(7, 3))
sns.heatmap(pivot_table, cmap='Pastel1', annot=True, fmt=".0f", cbar=False,
            annot_kws={"size": 10, "weight": "bold", "color": "black", "ha": "center", "va": "center"})

plt.title("30대 성별에 따른 방문주기")
plt.xlabel("방문 주기")
plt.ylabel("남성 (M) 여성 (F)")
plt.show()
```



재방문했다면, 어디를 가장 많이 방문했을까?

```
df_r = df_30[['회원ID', '시간', '매장명', '카테고리명.1', '카테고리명.2', '매장위치=동구분']]
df_r = df_r.merge(df_v[['회원ID', 'label']], on = '회원ID', how = 'left')
df_r
```

	회원ID	시간	매장명	카테고리명.1	카테고리명.2	매장위치=동구분	label
0	CM0120190000000000855	2019-05-01	아우어베이커리	식(F&B)	카페	B	0
1	CM0120190000000001580	2019-05-02	아우어베이커리	식(F&B)	카페	B	0
2	CM0120190000000001472	2019-05-02	밀도	식(F&B)	베이커리	B	0
3	CM0120190000000001357	2019-05-02	아오로	식(F&B)	양식	C	0
4	CM0120190000000001563	2019-05-02	아우어베이커리	식(F&B)	카페	B	8
...	...	...	...	...	...	...	...
		2022-	굿데이				

```
for _ in range(9):
    print(f'index # : {_}')
    print('매장명', df_r[df_r['label'] == _]['매장명'].value_counts().index[:7])
    print('방문횟수', df_r[df_r['label'] == _]['매장명'].value_counts().values[:7])
    print()
```

```
index # : 0
매장명 Index(['밀도', '아우어베이커리', '책발전소', '바오담', '삐삐네', '범산목장', '타이거슈가'], dtype='object')
방문횟수 [23063 14661  7258  5161  3837  3826  3826]

index # : 1
매장명 Index(['밀도', '아우어베이커리', '책발전소', '핌언더그라운드', '동트는농가', '도산분식', '더부쓰'], dtype='object')
```

```
방문횟수 [667 391 200 177  93  84  78]

index # : 2
매장명 Index(['밀도', '아우어베이커리', '핌언더그라운드', '책발전소', '동트는농가', '식물원', '더부쓰'], dtype='object')
방문횟수 [153 123  64  48  31  28  25]

index # : 3
매장명 Index(['밀도', '아우어베이커리', '책발전소', '핌언더그라운드', '피그먼트', '지로나', '동트는농가'], dtype='object')
방문횟수 [85 66 49 34 19 15 15]

index # : 4
매장명 Index(['아우어베이커리', '밀도', '책발전소', '핌언더그라운드', '안데르센', '신시아', '도산분식'], dtype='object')
방문횟수 [86 70 48 44 23 20 18]

index # : 5
매장명 Index(['밀도', '아우어베이커리', '책발전소', '식물원', '게방식당', '핌언더그라운드', '범산목장'], dtype='object')
방문횟수 [4658 2876 2558 1427  804  686  621]

index # : 6
매장명 Index(['밀도', '아우어베이커리', '책발전소', '핌언더그라운드', '동트는농가', '연남방앗간', '도산분식'], dtype='object')
방문횟수 [588 387 257 222  85  75  73]

index # : 7
매장명 Index(['밀도', '아우어베이커리', '책발전소', '핌언더그라운드', '식물원', '동트는농가', '도산분식'], dtype='object')
방문횟수 [233 202 121 105  47  46  36]

index # : 8
매장명 Index(['아우어베이커리', '밀도', '핌언더그라운드', '책발전소', '안데르센', '도산분식', '동트는농가'], dtype='object')
방문횟수 [473 399 319 278 224 109 100]
```

한번 방문시 두개 이상의 매장에 들렀는지 여부

```
def make_list(group):
    return list(set([item for item in group]))

df_r = df_r.groupby(['회원ID', '시간']).agg({'매장명': make_list, '매장위치=동구분' : make_list}).reset_index()
df_r
```

	회원ID	시간	매장명	매장위치=동구분
0	CM0120190000000000057	2019-05-03	[밀도, 아우어베이커리]	[B]
1	CM0120190000000000057	2019-05-04	[신시아]	[A]
2	CM0120190000000000057	2019-05-05	[밀도]	[B]
3	CM0120190000000000057	2019-05-16	[신시아]	[A]
4	CM0120190000000000057	2019-05-20	[밀도]	[B]
...	...	...	...	...
114654	CM0120220000000025285	2022-07-15	[아우어베이커리]	[B]
114655	CM0120220000000025316	2022-07-15	[세상의모든아침]	[B]
114656	CM0120220000000025323	2022-07-17	[신시아, 책발전소]	[B, A]
114657	CM0120220000000025329	2022-07-20	[핌언더그라운드]	[A]
114658	CM0120220000000025334	2022-07-22	[핌언더그라운드]	[A]

114659 rows × 4 columns

```
df_r['다른매장방문'] = np.where(df_r['매장명'].apply(len) >= 2, 0, 1)
df_r['다른매장방문'].value_counts()
```

```
1    88616
0    26043
Name: 다른매장방문, dtype: int64
```

```
## 가장 많이 방문한 조합

df_r0 = df_r[df_r['다른매장방문'] == 0]
df_r0['매장명'].value_counts()
```

```
[밀도, 아우어베이커리]                1117
[밀도, 책발전소]                     541
[밀도, 식물원]                       403
[아우어베이커리, 책발전소]                395
[밀도, 동트는농가]                    331
                              ...
[바오담, 만두언니네부엌, 식물원, 봉주르하와이]       1
[동네정미소, 삐삐네, 커피반점]                1
```

```
[밀도, 식물원, 아오로]                    1
[아우어베이커리, 책발전소, 식물원, 낙찌하다]       1
[연남방앗간, 핌언더그라운드, 동트는농가]          1
Name: 매장명, Length: 5151, dtype: int64
```

```
df_chong = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/일별_총매출.csv')
df_chong = df_chong.fillna(0)
```

```
df_chong.shape
```

```
(3646, 37)
```

```
df_chong = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/일별_총매출.csv')
df_chong = df_chong.fillna(0)
df_매장 = df_chong.drop_duplicates(['명', '매장코드'])
df_chong = df_chong.groupby(['매장코드']).agg({'합계' : 'median'}).reset_index()
df_chong = df_chong.sort_values(by = ['합계'], ascending = False)
df_chong = df_chong.merge(df_매장[['매장코드', '명']], on = '매장코드', how = 'left')
df_chong = df_chong.drop_duplicates(['매장코드'])
df_chong = df_chong.drop_duplicates(['명'])
df_chong.head(10)
```

	매장코드	합계	명
0	100640	242935385.0	도쿄등심
2	102310	211203158.5	세상의모든아침
5	100310	209042020.0	다곳
7	100240	201393845.0	밀도
9	100210	193305355.0	아우어베이커리
15	100600	130766070.0	준오헤어
17	101160	129123776.0	올리브영
19	100020	116244090.0	CU편의점
21	102220	115330500.0	셋째집
22	100500	106573500.0	안데르센

매출 Top 5: 도쿄등심, 세상의모든아침, 다곳, 밀도, 아우어베이커리

F&B Top 5 : 밀도, 아우어베이커리, 도쿄등심, 동트는농가, 책발전소

Anchor 계산 : Top5에 방문한 고객은 Top5가 아닌 매점에도 들리는가?

```
def anchor_list(lst):
    anchor_keywords = ['도쿄등심', '세상의모든아침', '다곳', '밀도', '아우어베이커리', '동트는농가', '책발전소']
    for keyword in anchor_keywords:
        if keyword in lst:
            return 0
    return 1

def anchor_list1(lst):
    cnt = 0
    anchor_keywords = ['도쿄등심', '세상의모든아침', '다곳', '밀도', '아우어베이커리', '동트는농가', '책발전소']
    for keyword in lst:
        if keyword not in anchor_keywords:
            cnt += 1
    return cnt

df_r0['anchor'] = df_r0['매장명'].apply(anchor_list)
df_r0['anchor_effect'] = df_r0['매장명'].apply(anchor_list1)
```

```
df_r0
```

	회원ID	시간	매장명	매장위치=동구분	다른매장방문	anchor	anchor_effect
0	CM012019000000000057	2019-05-03	[밀도, 아우어베이커리]	[B]	0	0	0
9	CM012019000000000057	2019-06-14	[아벡데프리츠, 신시아, 바오담]	[A, C]	0	1	3

```
df_r0['anchor'].value_counts()
```

```
0    17029
1     9014
Name: anchor, dtype: int64
```

18	CM012019000000000057	2019-	[낙찌아다,	[B, C]	0	1	2

```
df_r0[df_r0['anchor'] == 0]['매장명'].value_counts()[:10]
```

```
[밀도, 아우어베이커리]        1117
[밀도, 책발전소]             541
[밀도, 식물원]              403
[아우어베이커리, 책발전소]       395
[밀도, 동트는농가]            331
[밀도, 바오담]              254
[아우어베이커리, 동트는농가]      236
[밀도, 낙찌하다]             216
[밀도, 타이거슈가]            211
[밀도, 삐삐네]              210
Name: 매장명, dtype: int64
```

```
df_r0[df_r0['anchor'] == 0]['anchor_effect'].value_counts()
```

```
1    11491
0     3004
2     2084
3      374
4       66
5        9
6        1
Name: anchor_effect, dtype: int64
```

anchor 효과는 있음.

그러면 혹시 매점이 있는 동(A,B,C)에도 상관이 있을까?

```
df_an = df_r0[(df_r0['anchor'] == 0) & (df_r0['anchor_effect'] != 0)]
```

```
df_an['매장위치=동구분']
```

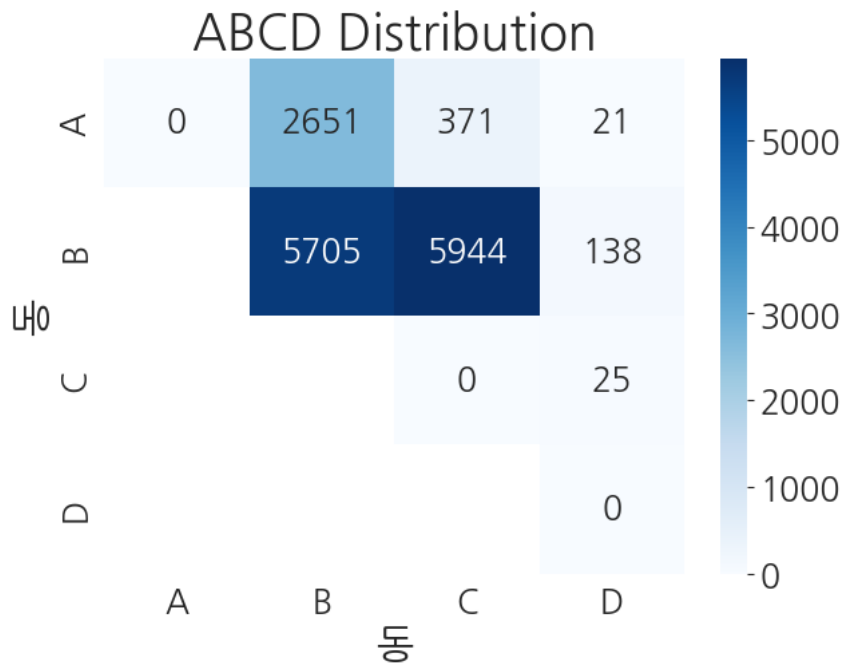
```
13        [B, C]
19        [B, C]
24           [B]
29           [B]
32           [B]
           ...
114603       [B]
114618    [B, C]
114628    [B, A]
114651    [B, A]
114656    [B, A]
Name: 매장위치=동구분, Length: 14025, dtype: object
```

```
np.unique(df_an['매장위치=동구분'])
```

```
array([list(['B']), list(['B', 'A']), list(['B', 'A', 'C']),
       list(['B', 'A', 'D']), list(['B', 'A', 'D', 'C']),
       list(['B', 'C']), list(['B', 'D']), list(['B', 'D', 'C']),
       list(['D', 'B', 'A']), list(['D', 'B', 'A', 'C']),
       list(['D', 'B', 'C'])], dtype=object)
```

```
classes = ['A', 'B', 'C', 'D']

matrix = np.zeros((len(classes), len(classes)), dtype=int)

for _, class_combinations in df_an['매장위치=동구분'].items():
    if len(class_combinations) == 1:
        matrix[1][1] += 1
```

```
            matrix[i][i] += 1
        else:
            for c1 in class_combinations:
                idx1 = classes.index(c1)
                for c2 in class_combinations:
                    if c2 != c1:
                      idx2 = classes.index(c2)
                      matrix[idx1][idx2] += 1

    plt.figure(figsize=(7, 5))
    sns.heatmap(matrix, annot=True, fmt='d', cmap='Blues',
                xticklabels=classes, yticklabels=classes, mask=np.tri(len(classes), k=-1))
    plt.xlabel('동')
    plt.ylabel('동')
    plt.title('ABCD Distribution')
    plt.show()
```



킬러 매장의 anchor 효과에는 문제가 없는 것으로 판단,

- 사업지의 상권내 위계등을 고려할 때, 약 1,500평을 지지할 수 있는 수요가 부족한 것으로 도출되었으나, 일부 수요는 호수공원 방문객으로 충당할 수 있을 것으로 추정됨

호수공원 방문객으로 충당하는데서 문제점 발생했나?

1. CA (Catchment Area)

- 1차 CA : 광교신도시 남부지역 (인구 비율 : 15%)
- 2차 CA : 광교신도시, 매탄동, 원천동 (인구 비율 : 24%)
- 3차 CA : 동수원, 기흥구 (인구 비율 : 61%)

산정식 : (CA 내 총 소비력 (총 소득액 x 상업/문화 지출 비율) * 흡수율)/평효율 = 개발 적정 면적

사업지의 적정 개발규모 추정을 위해 먼저 CA 내 총 소비력을 추정한 결과 월별 상업/문화관련 지출액이 약 1,820억원/월 수준임

예상유효소비력

구분	내용				비고
CA	1차	2차	3차	합계	-
세대수	17,777	27,966	71,966	250,388	가구
가구당 소득수준	57.2	50.6	46.7	-	백만원/년/가구
소득수준	1,017,200	1,415,919	3,555,840	11,983,901	백만원/년 (세대수X가구당 소득수준)
예상 월소득	84,767	117,993	296,320	998,658	백만원/월 (소득수준 /12월)
월별 상업/문화지출액	30,855	42,950	107,860	363,512	백만원/월 (월소득X36.4%)
예상 흡수율	7.0%	3.3%	1.9%	-	%
예상 유효소비력	2,166	1,412	540	6,968	백만원/월
점유면적	935.3	621.7	897.5	2,481	평

÷

평효율

평효율 (원/월/평;전용)
2,271,680 (경기지역 대형종합소매업 연간 매출평효율, 통계청 도소매업 총조사 가공)

CA1. 광교신도시

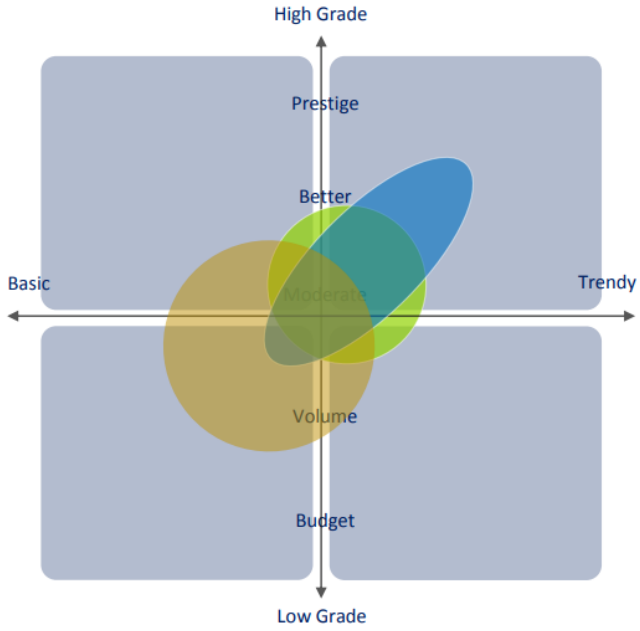
주 소비층	30~40대 및 유년 자녀 가정
평균소득	550만원/월
NEEDS	나만의 가치와 상품의 희소가치 중시 비중높음

CA2. 영통구

주 소비층	30~40대 및 10대 자녀 가정
평균소득	450만원/월
NEEDS	가족과 함께 즐길수 있는 소비추구 비중높음

CA3. 수원시

주 소비층	40대를 기준으로 다양하게 분포
평균소득	430만원/월
NEEDS	가족과 건강에 대한 소비가 상대적으로 높음



구분	판교 아브뉴프랑	사업지	산정 방법	비고
전용면적	3,700평	3,500평	-	-
기준 임대가	35만원/평	25만원/평	1층 Prime NI	사업지NI 가정
평균 임대가	25만원/평	18만원/평	전체 평균 NI	판교 사례 비율 반영 (Anchor 제외)
월임대료	92,500만원	63,000만원	(평균 임대가 X 전용면적)	-
매출수수료	12%	12%	전체 테넌트 임대조건 추정	판교 사례 수수료율 반영
월 매출액	77억원	53억원	(월임대료 X 매출수수료)	추정치
1인당 지출액	15,000원	15,000원	담당자 인터뷰	사업지는 사례와 동일하게 가정
월 방문객수	51만명	35만명	(월 매출액 ÷ 1인당 지출액)	실질 구매자
일 방문객수	1.7만명	1.2만명	(월 방문객수 ÷ 30일)	실질 구매자
MD 방향성	✓ Anchor 도입 → 집객력 시설 도입으로 방문객수 증가(인근 매장과 매출 연동) ✓ 라이프스타일샵 도입 → 다양한 가격대 및 상품군으로 방문객 범위 확대			

```
## 멤버십 매출 & 총 매출 퍼센트 비교

df_chong = pd.read_csv('/content/yonsei_bigdata_comp/전처리데이터/일별_총매출.csv')
df_chong = df_chong.fillna(0)


df.head(2)
```

	포인트적립ID	카드번호	승인ID	매장코드	회
0	A2019050000000120	2849010590003100	A2019050000000118	100210	CM01201900000000

```
df_chong.head(2)
```

	년	월	NO	매장코드	명	합계	1	2	3	4	...	22
					편사우							

```
df['시간'] = pd.to_datetime(df['시간'], errors='coerce')
df['년'] = df['시간'].dt.year
df['월'] = df['시간'].dt.month

df_ = df.groupby(['년', '월']).agg({'총거래금액' : 'sum'}).reset_index()


df_chong_ = df_chong.groupby(['년', '월']).agg({'합계' : 'sum'}).reset_index()


df_ = df_.merge(df_chong_, on = ['년', '월'], how = 'left')


df_['차이'] = df_['합계']/df_['총거래금액']


print(np.median(df_['차이']))
print(sum(df_['차이'])/len(df_['차이']))
```

```
7.84033638906152
161.48148949765618
```

```
df['회원자택주소1'].value_counts()
```

```
경기도          491110
서울특별시         7389
경기              758
인천광역시          357
충청남도            229
경상남도            198
부산광역시          116
세종특별자치시          75
광주광역시           54
충청북도            50
전라북도            49
경상북도            44
강원도             38
대전광역시           37
전라남도            17
제주특별자치도           9
대구광역시             8
울산광역시             2
Name: 회원자택주소1, dtype: int64
```

```
df['도'].value_counts()
```

```
수원시      430743
용인시       46312
```

```
화성시       6011
성남시       3801
강남구       1621
          ...
북구           1
시청대로         1
동두천시         1
경주시          1
계룡시          1
Name: 도, Length: 111, dtype: int64
```

```
df['구'].value_counts()[:10]
```

```
영통구    407915
기흥구     26555
수지구     18344
팔달구      8909
권선구      7588
장안구      6331
분당구      3229
처인구      1413
동안구       841
봉담읍       619
Name: 구, dtype: int64
```

```
# 1차 CA : 광교신도시 남부지역 (인구 비율 : 15%)
# 2차 CA : 광교신도시, 매탄동, 원천동 (인구 비율 : 24%)
# 3차 CA : 동수원, 기흥구 (인구 비율 : 61%)

# 1차 CA + 2차 CA = 영통구
# 3차 CA : 기흥구 + 수지구 + 팔달구 + 권선구
# 나머지 : 외부


def classify_CA(location):
    if location == '영통구':
        return 0
    elif location in ['기흥구', '수지구', '팔달구', '권선구']:
        return 1
    else:
        return 2


df['_loc'] = df['구'].apply(classify_CA)


df['_loc'].value_counts()
```

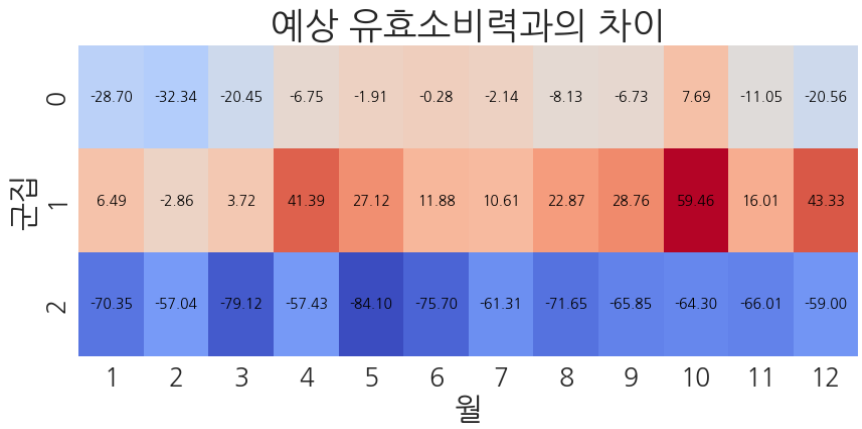
```
0    407915
1     61396
2     31229
Name: _loc, dtype: int64
```

```
## 세대수

## 0 : 17777 + 27966 = 45743
## 1 : 71966
## 2 : 250388-17777-27966-71966 = 132679

## 멤버쉽 결재내역 분포

## 0 : 407915
## 1 : 61396
## 2 : 31229

# 407915/45743 = 9 : 1
# 61396/71966 = 0.9 : 1
# 31229/132679 = 0.25 : 1

# 멤버쉽 방문 기준 : 36 : 4 : 1


df_0 = df[df['_loc'] == 0]
df_0 = df_0.groupby(['년', '월']).agg({'총거래금액' : 'sum'}).reset_index()
df_0 = df_0.merge(df_chong_, on = ['년', '월'], how = 'left')
df_0['차이'] = df_0['합계']/df_0['총거래금액']
print(np.median(df_0['차이']))
print(sum(df_0['차이'])/len(df_0['차이']))
```

```
12.058666740355388
12.484875998255426
```

```
df_ = df.groupby(['_loc', '년', '월']).agg({'총거래금액' : 'sum'}).reset_index()
```

```
## 군집 0 유효소비력 : 2166 + 1412 = 3578
## 군집 1 유효소비력 : 540
## 군집 2 유효소비력 : 6968 - 540 - 2166 - 1412 = 2850

def purchase_CA(_loc):
    if _loc == 0:
        return 3_578_000_000
    elif _loc == 1:
        return 540_000_000
    else:
        return 2_850_000_000

df_['purchase'] = df_['_loc'].apply(purchase_CA)
df_['총거래금액'] = df_['총거래금액'] * 12


df_['diff'] = (df_['총거래금액'] - df_['purchase']) / df_['purchase'] * 100

def color_map(val):
    if val < 0:
        color = 'blue'
        alpha = min(0.5, 1 - abs(val) / 100)
    else:
        color = 'red'
        alpha = min(0.5, val / 100)
    return color + f", {alpha:.2f}"

df_['color'] = df_['diff'].apply(color_map)

pivot_table = pd.crosstab(df_['_loc'], df_['월'], values=df_['diff'], aggfunc='median')

plt.figure(figsize=(10, 4))
sns.heatmap(pivot_table, cmap='coolwarm', annot=True, fmt=".2f", cbar=False,
            annot_kws={"size": 10, "weight": "bold", "color": "black", "ha": "center", "va": "center"})
plt.title("예상 유효소비력과의 차이")
plt.xlabel("월")
plt.ylabel("군집")
plt.show()
```



당초 예상한 유효소비력과 큰 차이 X,

하지만 호수공원 방문객의 역량을 과대평가함 (-50~80%)

멀리서 방문한 방문객은 멤버쉽에 가입하지 않는 경향성이 있다는 점은 고려해야 하지만, 앞선 -40 ~ -30%에 해당하는 손실은 외부 호수공원 방문객 유입 인구의 영향을 너무 크게 잡았다고 생각함

Final

solution

1. CA 1 + 2 (앨리웨이 근처 영통구 주민)의 유입을 더 활성화

2. 당초 계획했던 호수공원 유입 인원 확대

how?

1. 잘나가는 킬러 매장 (아우어베이커리, 밀도 등...)은 동 (마슬마켓, 앨리웨이) 장소에 관계없이 사람들을 타 매장에도 유입시키는 앵커효과가 있음. ==> 전체적인 유입을 견인하는 킬러 매장을 유입하면 방문객이 늘어남
2. 한식 특출나지 않음. 주 고객층이 30대~40대 특화해서 새로운 입점 매장 받아들이기 ==> but 기존 매장과 경쟁구도도 생각해야함
3. 호수공원 방문객 유입. But 보통 광교역 통해서 롯데백화점 쪽으로 유입되는데 가능할까? 호수공원 방문객보다 기존 파이를 늘리는게 더 좋을거 같음.

- 호수공원 방문하려고 광교오는가 앨리웨이 내 킬러매장 오려고 광교오는가?

1.

잘나가는 매점 --> 평일에도 사람들 많아서 자리 없음.

킬러매점이 얼마만큼 더 많아야 본전치나?

✓ 0초 오후 7:13에 완료됨