In [1]:
```
import pandas as pd
import numpy as np
file_path ='./cafedata/gyeongnam-pricedata.csv'
df = pd.read_csv(file_path)
df.head()
```

Out[1]:

	뚜레쥬르 지점	뚜레쥬르 거제수월	뚜레쥬르 거창스카이	뚜레쥬르 장유내덕	뚜레쥬르 장유석봉	뚜레쥬르 장유신문	뚜레쥬르 진영이진	뚜레쥬르 사천카이	뚜레쥬르 사천정동	뚜레쥬르 양산석산	...	뚜레쥬르 해운대수비교차로	뚜레쥬르 울산신정현대	뚜레쥬르 울산무거	뚜레쥬르 울산아음롯데	뚜레쥬르 울산전하동	뚜레쥬르 울산방어	뚜레쥬르 울산지웰자이	뚜레쥬르 울산명촌	뚜레쥬르 울산반구감변	뚜레쥬르 울산유곡혁신
0	마늘 단짝 고구마	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	...	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0
1	깊은 밤 행 스위스	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	...	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0
2	BELT 샌드위치	NaN	6900.0	7300.0	6900.0	6900.0	6900.0	6900.0	6900.0	NaN	...	7000.0	6900.0	6900.0	NaN	6900.0	6900.0	6900.0	6900.0	6900.0	6900.0
3	BLT콤 샐러드	NaN	NaN	NaN	NaN	NaN	8500.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	쉬림프 에그 샐러드	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

5 rows × 43 columns

In [2]:
```
print(df.columns)
```

```
Index(['뚜레쥬르 지점', '뚜레쥬르 거제수월', '뚜레쥬르 거창스카이', '뚜레쥬르 장유내덕', '뚜레쥬르 장유석봉',
       '뚜레쥬르 장유신문', '뚜레쥬르 진영이진', '뚜레쥬르 사천카이', '뚜레쥬르 사천정동', '뚜레쥬르 양산석산',
       '뚜레쥬르 양산물금신도시', '뚜레쥬르 양산터미널', '뚜레쥬르 진주호탄', '뚜레쥬르 진주망경', '뚜레쥬르 통영무전',
       '뚜레쥬르 경남함안', '뚜레쥬르 함안칠원', '뚜레쥬르 창원메트로시티', '뚜레쥬르 마산산호마린', '뚜레쥬르 부산중앙대로',
       '뚜레쥬르 부산대학교', '뚜레쥬르 부산남산', '뚜레쥬르 기장교리', '뚜레쥬르 부산일광신도시', '뚜레쥬르 부산대연',
       '뚜레쥬르 동래허브스카이', '뚜레쥬르 부산동의대역', '뚜레쥬르 부산개금백병원', '뚜레쥬르 부산학장본점',
       '뚜레쥬르 부산서대신역', '뚜레쥬르 부산동아대', '뚜레쥬르 부산남천', '뚜레쥬르 부산대청로', '뚜레쥬르 해운대수비교차로',
       '뚜레쥬르 울산신정현대', '뚜레쥬르 울산무거', '뚜레쥬르 울산아음롯데', '뚜레쥬르 울산전하동', '뚜레쥬르 울산방어',
       '뚜레쥬르 울산지웰자이', '뚜레쥬르 울산명촌', '뚜레쥬르 울산반구감변', '뚜레쥬르 울산유곡혁신'],
      dtype='object')
```

In [3]:
```
import re

def categorize_menu(df):
    # 키워드 기반 카테고리 매핑 딕셔너리
    category_keywords = {
        '샌드위치류': ['샌드위치', 'BELT', 'BLT', 'V.E.L.T'],
        '샐러드류': ['샐러드'],
        '식빵류': ['식빵', '우유롤', '우유 브레드', '소버식빵'],
        '크림빵': ['크림가득 메론빵','마당 얼그레이 크림번','순진우유크림빵','겹겹이 연유 크림 데니쉬','사르르 고구마케이크빵','사르르 우유크림빵','빵속에리얼초코','카페모카크림빵','까까웨뜨'],
        '피자빵,고로케': ['고로케', '소시지브레드','피자토스트','NEW어니언소시지포카치아'],
        '파이/패스트리': ['바통쉬크레','크라상','애플파이','유자파이'],
        '간식빵': ['소금버터롤','치즈방앗간','깨찰빵','소보로빵','오리지널 커피번','카페모카빵','꽈배기','옛날 단팥 도넛',r'^단팥빵$','단팥소보로빵'],
        '신제품': ['마구마구', '단짝', '뺑스위스']
    }

    # 새로운 카테고리 컬럼 생성
    df['카테고리'] = '기타'  # 기본값

    # 각 메뉴명에 대해 카테고리 매핑
    for idx, menu_name in enumerate(df['뚜레쥬르 지점']):
        if pd.isna(menu_name):  # null 체크
            continue

        menu_name = str(menu_name).lower()  # 소문자 변환

        # 각 카테고리의 키워드 체크
        for category, keywords in category_keywords.items():
            if any(keyword.lower() in menu_name for keyword in keywords):
                df.loc[idx, '카테고리'] = category
                break

    return df

def analyze_categories_by_store(df):
    # 매장별 카테고리별 기본 통계
    stores = df.columns[1:-1]  # 첫 번째 열(메뉴명)과 마지막 열(카테고리) 제외

    # 카테고리별 기본 통계
    category_stats = pd.DataFrame()

    for store in stores:
        # 매장별 데이터 숫자로 변환 (오류 방지)
        df[store] = pd.to_numeric(df[store], errors='coerce')

        temp = df.groupby('카테고리').agg({store: 'mean'})
        temp.reset_index(inplace=True)
        temp.rename(columns={store: '평균 가격'}, inplace=True)
        temp['매장명'] = store
        category_stats = pd.concat([category_stats, temp], axis=0)

    return category_stats

def pivot_store_category(stats):
    # 피벗 테이블 생성
    pivot_table = stats.pivot_table(index='매장명', columns='카테고리', values='평균 가격', aggfunc='mean')
    pivot_table=pivot_table.round(1)
    pivot_table.reset_index(inplace=True)
    return pivot_table

# 데이터 로드 및 처리
def process_bakery_data(filepath):
    # CSV 파일 읽기
    df = pd.read_csv(filepath)

    # 카테고리 지정
    df = categorize_menu(df)

    # 매장별 카테고리별 분석
    stats = analyze_categories_by_store(df)

    # 피벗 테이블 생성
    pivot_table = pivot_store_category(stats)

    return df, pivot_table

# 파일 처리 및 결과 생성
df, pivot_table = process_bakery_data(file_path)

# 카테고리화된 데이터 및 매장별 통계 표시
from IPython.display import display

# print("카테고리화된 가격 데이터 (처음 5개 행)")
# display(df.head())
#########################################################################################################################

storeinfo_filepath='./adress_process/gyeongnam_adress.csv'

def process_address(address):
    try:
        # 수동 수정
        if address == '경기도 동탄지성로469번길 60 5단지 상가1동107호,108호,109호':
            return '경기도 화성시'

        # 정규표현식으로 '충청남도 XX시' 추출
        match = re.match(r'경상남도 \w+시', address) or re.match(r'경상남도 \w+군', address) or re.match(r'부산광역시\s+\w+구', address) or re.match(r'부산광역시\s+\w+군', address) or re.match(r'울산광역시\s+\w+구', address) or re.match(r'울

        if match:
            return match.group()

        # 기본값 반환
        return address
    except Exception as e:
        print(f"주소 처리 중 오류 발생: {address}, {e}")
        return address

def load_store_info(storeinfo_filepath):
    store_info = pd.read_csv(storeinfo_filepath)
    # 주소 컬럼 처리
    store_info['주소'] = store_info['주소'].apply(process_address)
    return store_info



def process_bakery_data(price_filepath, store_info_filepath):
    # 가격 데이터 로드
    df = pd.read_csv(price_filepath)

    # 매장 정보 데이터 로드
    store_info = load_store_info(store_info_filepath)

    # 카테고리 지정
    df = categorize_menu(df)
```

```
    # 매장별 카테고리별 분석
    stats = analyze_categories_by_store(df)

    # 피벗 테이블 생성 후 매장 정보 병합
    pivot_table = pivot_store_category(stats)
    result = pd.merge(pivot_table, store_info,
                      left_on='매장명',
                      right_on='매장',
                      how='left')

    # 컬럼 순서 재정렬
    columns = ['매장명', '주소', '지역'] + [col for col in result.columns
              if col not in ['매장명', '매장', '주소', '지역']]
    result = result[columns]

    return df, result

# 실제 파일 경로로 호출
df, result = process_bakery_data('./cafedata/gyeongnam-pricedata.csv',
                                 './adress_process/gyeongnam_adress.csv')

# 결과 출력
print("\n매장별 카테고리별 평균 가격 (주소 정보 포함)")
display(result)
```

매장별 카테고리별 평균 가격 (주소 정보 포함)

	매장명	주소	지역	간식빵	기타	샌드위치류	샐러드류	식빵류	신제품	크림빵	파이/패스트리	피자빵,고로케
0	뚜레쥬르 거제수월	경상남도 거제시	경남	3050.0	4202.0	7166.7	NaN	5027.3	4600.0	3450.0	2933.3	3116.7
1	뚜레쥬르 거창스카이	경상남도 거창군	경남	3250.0	4339.4	7040.0	8400.0	4977.8	4600.0	3233.3	3100.0	2800.0
2	뚜레쥬르 경남함안	경상남도 함안군	경남	2671.4	4502.8	7160.0	NaN	5000.0	4600.0	3240.0	2933.3	2850.0
3	뚜레쥬르 기장교리	부산광역시 기장군	경남	2671.4	3825.0	6950.0	NaN	4666.7	4600.0	2350.0	NaN	2750.0
4	뚜레쥬르 동래허브스카이	부산광역시 동래구	경남	3571.4	4145.9	7166.7	NaN	5245.5	4600.0	3240.0	2933.3	3650.0
5	뚜레쥬르 마산산호마린	경상남도 창원시	경남	2700.0	4032.7	7142.9	NaN	5111.1	4933.3	3675.0	3100.0	3116.7
6	뚜레쥬르 부산개금백병원	부산광역시 부산진구	경남	2810.0	4300.0	7150.0	NaN	4212.5	4300.0	3360.0	2933.3	3200.0
7	뚜레쥬르 부산남산	부산광역시 금정구	경남	2722.2	4171.4	NaN	NaN	5222.2	4600.0	3675.0	2933.3	3180.0
8	뚜레쥬르 부산남천	부산광역시 수영구	경남	2885.7	4111.5	7240.0	7400.0	4900.0	4600.0	3760.0	2400.0	2750.0
9	뚜레쥬르 부산대연	부산광역시 남구	경남	3354.5	4183.3	7166.7	NaN	5090.0	4600.0	3525.0	3100.0	3180.0
10	뚜레쥬르 부산대청로	부산광역시 중구	경남	3100.0	4354.8	7075.0	8350.0	5087.5	4600.0	3314.3	2933.3	3116.7
11	뚜레쥬르 부산대학교	부산광역시 금정구	경남	3042.9	3789.2	7055.6	NaN	5000.0	4600.0	2975.0	2900.0	2950.0
12	뚜레쥬르 부산동아대	부산광역시 사하구	경남	2755.6	4200.0	7260.0	NaN	5050.0	4600.0	3700.0	2933.3	3100.0
13	뚜레쥬르 부산동의대역	부산광역시 부산진구	경남	2971.4	4471.9	7100.0	8000.0	5025.0	4600.0	2800.0	2933.3	2866.7
14	뚜레쥬르 부산서대신역	부산광역시 서구	경남	2836.4	4438.5	7160.0	8300.0	4972.7	4600.0	3337.5	2933.3	3180.0
15	뚜레쥬르 부산일광신도시	부산광역시 기장군	경남	2900.0	4008.8	7080.0	NaN	5081.8	4300.0	3314.3	2933.3	3180.0
16	뚜레쥬르 부산중앙대로	부산광역시 중구	경남	2800.0	4156.0	7100.0	8350.0	5000.0	4600.0	3066.7	2800.0	3100.0
17	뚜레쥬르 부산학장본점	부산광역시 사상구	경남	2480.0	4271.4	7133.3	NaN	4981.8	4600.0	3700.0	3200.0	3025.0
18	뚜레쥬르 사천정동	경상남도 사천시	경남	2930.0	4461.1	7075.0	NaN	5340.0	4600.0	3720.0	2900.0	3340.0
19	뚜레쥬르 사천카이	경상남도 사천시	경남	2866.7	3953.1	7114.3	NaN	4888.9	4600.0	3675.0	2900.0	3340.0
20	뚜레쥬르 양산물금신도시	경상남도 양산시	경남	2900.0	4246.3	7128.6	8700.0	4981.8	4600.0	3720.0	2900.0	3366.7
21	뚜레쥬르 양산석산	경상남도 양산시	경남	2533.3	4390.9	7250.0	NaN	5040.0	4600.0	3333.3	2933.3	3100.0
22	뚜레쥬르 양산터미널	경상남도 양산시	경남	2333.3	4235.0	7133.3	NaN	5000.0	4600.0	3100.0	2400.0	3200.0
23	뚜레쥬르 울산명촌	울산광역시 북구	경남	2920.0	4421.4	7066.7	NaN	5514.3	4600.0	3366.7	3000.0	3800.0
24	뚜레쥬르 울산무거	울산광역시 남구	경남	3077.8	4568.8	7160.0	8066.7	5300.0	4600.0	3566.7	2800.0	3180.0
25	뚜레쥬르 울산반구강변	울산광역시 중구	경남	3100.0	4351.5	7160.0	8300.0	5277.8	4600.0	3483.3	2933.3	3100.0
26	뚜레쥬르 울산방어	울산광역시 동구	경남	2414.3	4055.6	7150.0	8300.0	5181.8	4600.0	3233.3	2933.3	2980.0
27	뚜레쥬르 울산신정현대	울산광역시 남구	경남	2800.0	3902.5	7125.0	8350.0	4876.9	4600.0	3333.3	2933.3	3180.0
28	뚜레쥬르 울산야음롯데	울산광역시 남구	경남	2783.3	4200.0	7460.0	NaN	5058.3	4600.0	3514.3	3066.7	3400.0
29	뚜레쥬르 울산유곡혁신	울산광역시 중구	경남	3125.0	4122.2	7120.0	8300.0	4975.0	4933.3	3357.1	2933.3	3025.0
30	뚜레쥬르 울산전하동	울산광역시 동구	경남	3000.0	4415.0	7183.3	8350.0	5100.0	4600.0	3525.0	2933.3	3025.0
31	뚜레쥬르 울산지웰자이	울산광역시 동구	경남	2671.4	4337.0	7133.3	8400.0	4340.0	4600.0	3314.3	2933.3	3083.3
32	뚜레쥬르 장유내덕	경상남도 김해시	경남	2985.7	3909.5	7450.0	NaN	5050.0	4933.3	3600.0	2933.3	3025.0
33	뚜레쥬르 장유석봉	경상남도 김해시	경남	3625.0	4446.4	7250.0	NaN	4875.0	4600.0	3833.3	3066.7	2700.0
34	뚜레쥬르 장유신문	경상남도 김해시	경남	2800.0	4352.8	7085.7	NaN	4872.7	4600.0	3233.3	2800.0	3120.0
35	뚜레쥬르 진영이진	경상남도 김해시	경남	2550.0	4144.2	7300.0	8500.0	5066.7	4600.0	3337.5	2933.3	3116.7
36	뚜레쥬르 진주망경	경상남도 진주시	경남	3072.7	4298.3	7140.0	NaN	5470.0	4933.3	3520.0	3100.0	3080.0
37	뚜레쥬르 진주호탄	경상남도 진주시	경남	2787.5	4188.2	7050.0	NaN	5214.3	4600.0	3466.7	2933.3	3650.0
38	뚜레쥬르 창원메트로시티	경상남도 창원시	경남	3040.0	4314.0	7233.3	8300.0	5114.3	4600.0	3400.0	3100.0	3180.0
39	뚜레쥬르 통영무전	경상남도 통영시	경남	2900.0	4151.2	7050.0	8300.0	4925.0	4600.0	3314.3	2933.3	3040.0
40	뚜레쥬르 함안칠원	경상남도 함안군	경남	3077.8	3765.0	6940.0	NaN	5133.3	4933.3	3333.3	3000.0	3150.0
41	뚜레쥬르 해운대수비교차로	부산광역시 해운대구	경남	2990.0	3842.9	7166.7	8500.0	4888.9	4600.0	3283.3	3300.0	3325.0

In [4]:
```
grouped_data = result.groupby('주소')[['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']].mean().mean(axis=1).sort_values(ascending=False)

# groupby 결과를 데이터프레임으로 변환
grouped_df = pd.DataFrame(grouped_data).reset_index()

# 컬럼명 변경
grouped_df.columns = ['주소', '평균가격']

# CSV 파일로 저장
grouped_df.to_csv('./anal_gyeongsang/average_allbread_gn.csv', index=False, encoding='utf-8-sig')
grouped_df
```

Out[4]:

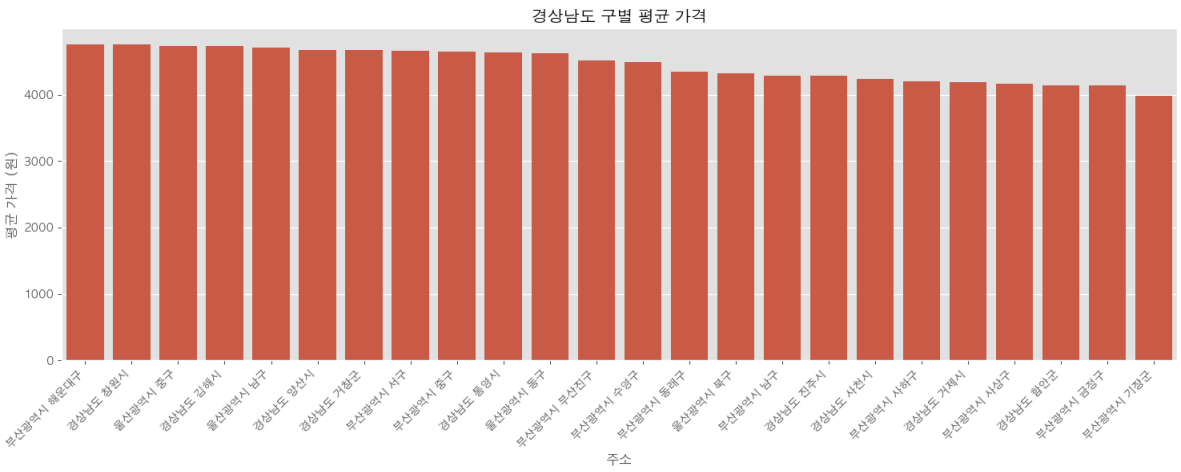
	주소	평균가격
0	부산광역시 해운대구	4756.737500
1	경상남도 창원시	4752.912500
2	울산광역시 중구	4732.693750
3	경상남도 김해시	4729.475000
4	울산광역시 남구	4710.027083
5	경상남도 양산시	4677.233333
6	경상남도 거창군	4675.137500
7	부산광역시 서구	4664.987500
8	부산광역시 중구	4649.593750
9	경상남도 통영시	4632.825000
10	울산광역시 동구	4624.370833
11	부산광역시 부산진구	4516.387500
12	부산광역시 수영구	4491.962500
13	부산광역시 동래구	4343.842857
14	울산광역시 북구	4323.957143
15	부산광역시 남구	4288.028571
16	경상남도 진주시	4286.985714
17	경상남도 사천시	4234.992857
18	부산광역시 사하구	4199.842857
19	경상남도 거제시	4192.000000
20	부산광역시 사상구	4160.014286
21	경상남도 함안군	4144.457143
22	부산광역시 금정구	4136.557143
23	부산광역시 기장군	3979.342857

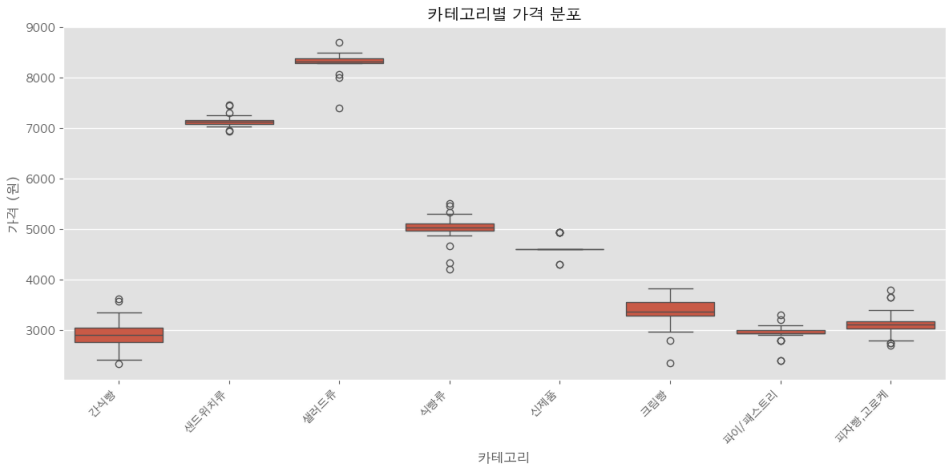
In [5]:
```
categories = ['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']

# 각 카테고리별로 구의 평균 가격 계산
grouped_data = {}
for category in categories:
    grouped_data[category] = result.groupby('주소')[category].mean().round(2)

# 데이터프레임 생성
grouped_df = pd.DataFrame(grouped_data)

# CSV 파일로 저장
grouped_df.to_csv('anal_gyeongsang/average_categorized_shopgn.csv', encoding='utf-8-sig')
grouped_df
```

Out[5]:

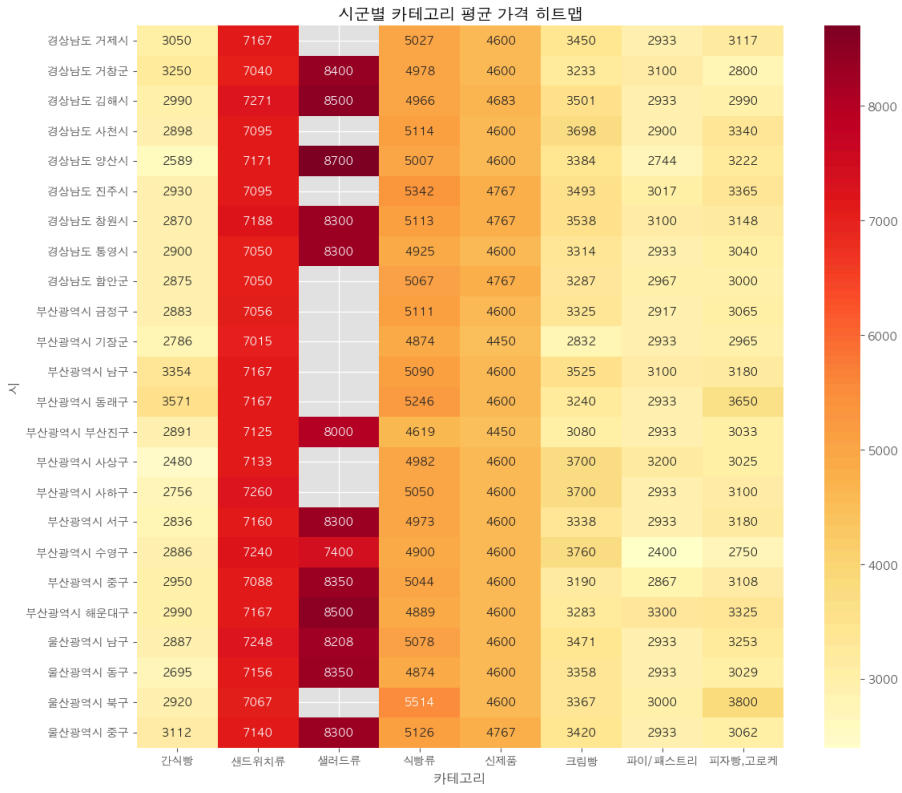
주소	간식빵	샌드위치류	샐러드류	식빵류	신제품	크림빵	파이/패스트리	피자빵,고로케
경상남도 거제시	3050.00	7166.70	NaN	5027.30	4600.00	3450.00	2933.30	3116.70
경상남도 거창군	3250.00	7040.00	8400.00	4977.80	4600.00	3233.30	3100.00	2800.00
경상남도 김해시	2990.18	7271.42	8500.00	4966.10	4683.32	3501.02	2933.32	2990.42
경상남도 사천시	2898.35	7094.65	NaN	5114.45	4600.00	3697.50	2900.00	3340.00
경상남도 양산시	2588.87	7170.63	8700.00	5007.27	4600.00	3384.43	2744.43	3222.23
경상남도 진주시	2930.10	7095.00	NaN	5342.15	4766.65	3493.35	3016.65	3365.00
경상남도 창원시	2870.00	7188.10	8300.00	5112.70	4766.65	3537.50	3100.00	3148.35
경상남도 통영시	2900.00	7050.00	8300.00	4925.00	4600.00	3314.30	2933.30	3040.00
경상남도 함안군	2874.60	7050.00	NaN	5066.65	4766.65	3286.65	2966.65	3000.00
부산광역시 금정구	2882.55	7055.60	NaN	5111.10	4600.00	3325.00	2916.65	3065.00
부산광역시 기장군	2785.70	7015.00	NaN	4874.25	4450.00	2832.15	2933.30	2965.00
부산광역시 남구	3354.50	7166.70	NaN	5090.00	4600.00	3525.00	3100.00	3180.00
부산광역시 동래구	3571.40	7166.70	NaN	5245.50	4600.00	3240.00	2933.30	3650.00
부산광역시 부산진구	2890.70	7125.00	8000.00	4618.75	4450.00	3080.00	2933.30	3033.35
부산광역시 사상구	2480.00	7133.30	NaN	4981.80	4600.00	3700.00	3200.00	3025.00
부산광역시 사하구	2755.60	7260.00	NaN	5050.00	4600.00	3700.00	2933.30	3100.00
부산광역시 서구	2836.40	7160.00	8300.00	4972.70	4600.00	3337.50	2933.30	3180.00
부산광역시 수영구	2885.70	7240.00	7400.00	4900.00	4600.00	3760.00	2400.00	2750.00
부산광역시 중구	2950.00	7087.50	8350.00	5043.75	4600.00	3190.50	2866.65	3108.35
부산광역시 해운대구	2990.00	7166.70	8500.00	4888.90	4600.00	3283.30	3300.00	3325.00
울산광역시 남구	2887.03	7248.33	8208.35	5078.40	4600.00	3471.43	2933.33	3253.33
울산광역시 동구	2695.23	7155.53	8350.00	4873.93	4600.00	3357.53	2933.30	3029.43
울산광역시 북구	2920.00	7066.70	NaN	5514.30	4600.00	3366.70	3000.00	3800.00
울산광역시 중구	3112.50	7140.00	8300.00	5126.40	4766.65	3420.20	2933.30	3062.50

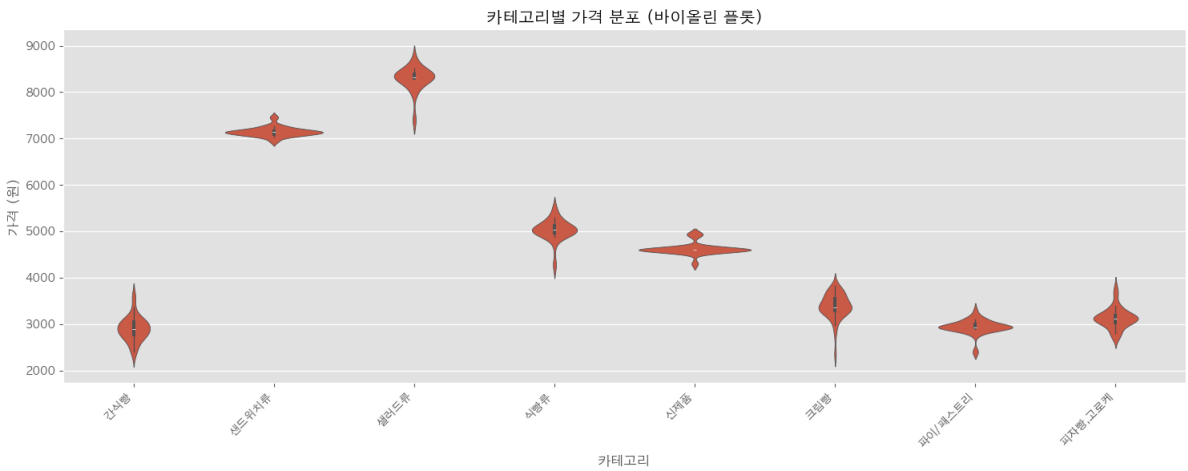
In [6]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
from matplotlib import font_manager, rc

# Mac OS 용 폰트 설정
plt.rc('font', family='AppleGothic') # 맥용 폰트 설정

# 그래프 기본 설정
plt.rcParams['axes.unicode_minus'] = False
plt.style.use('ggplot')


# 1. 구별 전체 평균 가격 분석
plt.figure(figsize=(15, 6))
grouped_data = result.groupby('주소')[['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']].mean().mean(axis=1).sort_values(ascending=False)

sns.barplot(x=grouped_data.index, y=grouped_data.values)
plt.title('경상남도 구별 평균 가격')
plt.xticks(rotation=45, ha='right')
plt.ylabel('평균 가격 (원)')
plt.tight_layout()
plt.show()

# 2. 카테고리별 가격 분포 (박스플롯)
plt.figure(figsize=(12, 6))
categories = ['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']
data_melted = pd.melt(result, value_vars=categories)

sns.boxplot(x='variable', y='value', data=data_melted)
plt.title('카테고리별 가격 분포')
plt.xticks(rotation=45, ha='right')
plt.xlabel('카테고리')
plt.ylabel('가격 (원)')
plt.tight_layout()
plt.show()
```





In [7]:
```
# 3. 구별/카테고리별 평균 가격 히트맵
plt.figure(figsize=(12, 10))
pivot_data = result.groupby('주소')[categories].mean()
sns.heatmap(pivot_data, annot=True, fmt='.0f', cmap='YlOrRd')
plt.title('시군별 카테고리 평균 가격 히트맵')
plt.ylabel('시')
plt.xlabel('카테고리')
plt.tight_layout()
plt.show()

# 5. 카테고리별 가격 분포 (바이올린 플롯)
plt.figure(figsize=(15, 6))
sns.violinplot(x='variable', y='value', data=data_melted)
plt.title('카테고리별 가격 분포 (바이올린 플롯)')
plt.xticks(rotation=45, ha='right')
plt.xlabel('카테고리')
plt.ylabel('가격 (원)')
plt.tight_layout()
plt.show()
```

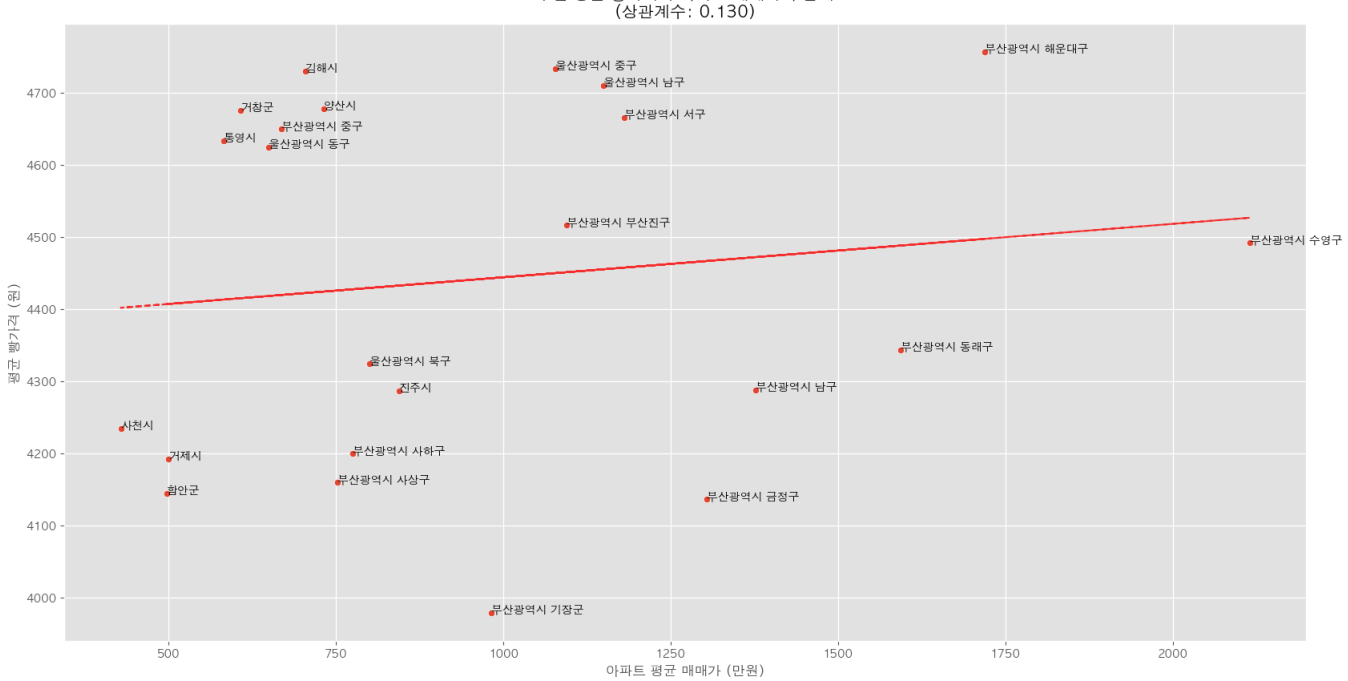




In [8]:
```
# 1. 구별 평균 빵가격 계산
categories = ['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']
bread_price_by_district = result.groupby('주소')[categories].mean().mean(axis=1).reset_index()
bread_price_by_district.columns = ['구분', '평균_빵가격']
# '경상남도'제거
bread_price_by_district['구분'] = bread_price_by_district['구분'].str.replace('경상남도', '').str.strip()

# 아파트 가격 데이터 전처리
apt_price = pd.read_csv('anal_gyeongsang/gyeongnam_apt_price.csv')
# '경상남도'와 '구' 제거
apt_price['구분'] = apt_price['구분'].str.replace('경상남도', '').str.strip()

apt_price['매매'] = pd.to_numeric(apt_price['매매'].str.replace(',', ''), errors='coerce')
```

```
apt_price = apt_price.dropna()  # 결측치 제거


# 데이터 확인
print("전처리 후 구별 빵가격 데이터:")
print(bread_price_by_district)
print("\n전처리 후 아파트 가격 데이터:")
print(apt_price)

# 데이터 병합
merged_df = pd.merge(bread_price_by_district, apt_price[['구분', '매매']], on='구분', how='inner')
print("\n병합된 데이터:")
print(merged_df)

# 시각화
if not merged_df.empty:
    plt.figure(figsize=(20, 10))
    sns.scatterplot(data=merged_df, x='매매', y='평균_빵가격')

    # 추세선 추가
    x = merged_df['매매'].values
    y = merged_df['평균_빵가격'].values
    z = np.polyfit(x, y, 1)
    p = np.poly1d(z)
    plt.plot(x, p(x), "r--", alpha=0.8)

    # 각 점에 구 이름 표시
    for idx, row in merged_df.iterrows():
        plt.annotate(row['구분'], (row['매매'], row['평균_빵가격']))

    correlation = merged_df['평균_빵가격'].corr(merged_df['매매'])
    plt.title(f'구별 평균 빵가격과 아파트 매매가의 관계\n(상관계수: {correlation:.3f})')
    plt.xlabel('아파트 평균 매매가 (만원)')
    plt.ylabel('평균 빵가격 (원)')

    print(f"\n상관계수: {correlation:.3f}")
    if correlation > 0:
        print("양의 상관관계가 있습니다: 아파트 가격이 높은 구일수록 빵 가격도 높은 경향이 있습니다.")
    else:
        print("음의 상관관계가 있습니다: 아파트 가격이 높은 구일수록 빵 가격이 낮은 경향이 있습니다.")
```

```
전처리 후 구별 빵가격 데이터:
              구분        평균_빵가격
0           거제시  4192.000000
1           거창군  4675.137500
2           김해시  4729.475000
3           사천시  4234.992857
4           양산시  4677.233333
5           진주시  4286.985714
6           창원시  4752.912500
7           통영시  4632.825000
8           함안군  4144.457143
9    부산광역시 금정구  4136.557143
10   부산광역시 기장군  3979.342857
11    부산광역시 남구  4288.028571
12   부산광역시 동래구  4343.842857
13  부산광역시 부산진구  4516.387500
14   부산광역시 사상구  4160.014286
15   부산광역시 사하구  4199.842857
16    부산광역시 서구  4664.987500
17   부산광역시 수영구  4491.962500
18    부산광역시 중구  4649.593750
19  부산광역시 해운대구  4756.737500
20    울산광역시 남구  4710.027083
21    울산광역시 동구  4624.370833
22    울산광역시 북구  4323.957143
23    울산광역시 중구  4732.693750

전처리 후 아파트 가격 데이터:
              구분    매매   전세
0           거제시   500  367
1           거창군   608  391
2           고성군   327  217
3           김해시   704  555
4           남해군   375  220
5           밀양시   533  337
6           사천시   429  359
7           산청군   571  353
8           양산시   732  496
9           의령군   672  448
10          진주시   844  654
11          창녕군   488  364
12   창원시 마산합포구   660  537
13   창원시 마산회원구   722  588
14    창원시 성산구  1042  786
15    창원시 의창구  1280  820
16    창원시 진해구   729  572
17          통영시   582  453
18          하동군   338  224
19          함안군   497  420
20          함양군   576  415
21          합천군   336  229
22   부산광역시 강서구  1331  850
23   부산광역시 금정구  1304  808
24   부산광역시 기장군   982  677
25    부산광역시 남구  1377  822
26    부산광역시 동구  1316  735
27   부산광역시 동래구  1593  867
28  부산광역시 부산진구  1095  690
29    부산광역시 북구   971  641
30   부산광역시 사상구   752  533
31   부산광역시 사하구   775  522
32    부산광역시 서구  1181  728
33   부산광역시 수영구  2115  870
34   부산광역시 연제구  1554  859
35   부산광역시 영도구   722  479
36    부산광역시 중구   669  387
37  부산광역시 해운대구  1719  843
38    울산광역시 남구  1150  789
39    울산광역시 동구   650  490
40    울산광역시 북구   800  585
41   울산광역시 울주군   775  594
42    울산광역시 중구  1078  768

병합된 데이터:
              구분        평균_빵가격    매매
0           거제시  4192.000000   500
1           거창군  4675.137500   608
2           김해시  4729.475000   704
3           사천시  4234.992857   429
4           양산시  4677.233333   732
5           진주시  4286.985714   844
6           통영시  4632.825000   582
7           함안군  4144.457143   497
8    부산광역시 금정구  4136.557143  1304
9    부산광역시 기장군  3979.342857   982
10    부산광역시 남구  4288.028571  1377
11   부산광역시 동래구  4343.842857  1593
12  부산광역시 부산진구  4516.387500  1095
13   부산광역시 사상구  4160.014286   752
14   부산광역시 사하구  4199.842857   775
15    부산광역시 서구  4664.987500  1181
16   부산광역시 수영구  4491.962500  2115
17    부산광역시 중구  4649.593750   669
18  부산광역시 해운대구  4756.737500  1719
19    울산광역시 남구  4710.027083  1150
20    울산광역시 동구  4624.370833   650
21    울산광역시 북구  4323.957143   800
22    울산광역시 중구  4732.693750  1078

상관계수: 0.130
양의 상관관계가 있습니다: 아파트 가격이 높은 구일수록 빵 가격도 높은 경향이 있습니다.
```



In [9]:
```python
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# 시각화 기본 설정
plt.style.use('default')  # 기본 스타일 사용
plt.rcParams['figure.figsize'] = (15, 10)
plt.rcParams['font.family'] = 'AppleGothic'

if not merged_df.empty:
    # 데이터 정렬
    sorted_df = merged_df.sort_values(by='매매')

    # 메인 그래프 생성
    fig, ax1 = plt.subplots(figsize=(30, 13))

    # 기본 그리드 제거
    ax1.grid(False)

    # 바차트 배경 (매매가)
    ax1.bar(range(len(sorted_df)), sorted_df['매매'], alpha=0.5, color='pink', label='아파트 매매가')

    # 산점도를 위한 두 번째 y축 생성
    ax2 = ax1.twinx()
    ax2.grid(False)

    # 산점도 그리기
    scatter = ax2.scatter(range(len(sorted_df)), sorted_df['평균_빵가격'],
                          s=100, alpha=0.6, color='red', label='빵 가격')

    # 추세선 추가
    z = np.polyfit(range(len(sorted_df)), sorted_df['평균_빵가격'], 1)
    p = np.poly1d(z)
    ax2.plot(range(len(sorted_df)), p(range(len(sorted_df))), "g--",
             linewidth=2, alpha=0.8, label='빵 가격 추세선')

    # x축 레이블 설정 (45도 회전)
    ax1.set_xticks(range(len(sorted_df)))
    ax1.set_xticklabels(sorted_df['구분'], rotation=45, ha='right')

    # 각 점에 구 이름 표시
    for i, row in enumerate(sorted_df.itertuples()):
        ax2.annotate(row.구분,
                     (i, row.평균_빵가격),
                     xytext=(6, 6),
                     textcoords='offset points',
                     fontsize=10,
                     bbox=dict(boxstyle='round,pad=0.3', fc='yellow', alpha=0.3, ec='none'))
                     # arrowprops=dict(arrowstyle='->', connectionstyle='arc3,rad=0'))

    # 평균선 추가
    # ax2.axhline(y=sorted_df['평균_빵가격'].mean(), color='g', linestyle='--', alpha=0.3, label='평균 빵가격')
    # ax1.axhline(y=sorted_df['매매'].mean(), color='b', linestyle='--', alpha=0.3, label='평균 매매가')

    # y축 그리드만 추가
    ax1.grid(True, axis='y', alpha=0.3, linestyle='-', color='gray')

    # 축 레이블 설정
    ax1.set_ylabel('아파트 평균 매매가 (만원)', fontsize=12)
    ax2.set_ylabel('평균 빵가격 (원)', fontsize=12)

    # 상관계수 계산
    correlation = sorted_df['평균_빵가격'].corr(sorted_df['매매'])

    # 그래프 제목 설정
    plt.title('경남 지역 구별 아파트 매매가와 평균 빵가격의 관계', fontsize=16, pad=20)

    # 통계 정보 추가
    stats_text = f'상관계수: {correlation:.3f}\n'
    stats_text += f'평균 빵가격: {sorted_df["평균_빵가격"].mean():.0f}원\n'
    stats_text += f'평균 매매가: {sorted_df["매매"].mean():.0f}만원'
    ax1.text(0.02, 0.98, stats_text,
             transform=ax1.transAxes,
             verticalalignment='top',
             bbox=dict(boxstyle='round', facecolor='white', alpha=0.8))

    # 범례 추가
    lines1, labels1 = ax1.get_legend_handles_labels()
    lines2, labels2 = ax2.get_legend_handles_labels()
    ax1.legend(lines1 + lines2, labels1 + labels2, loc='upper right')

    # 여백 조정
    plt.tight_layout()

    # 그래프 표시
    plt.show()

    # 추가 분석 출력
    print("\n=== 지역별 상세 데이터 ===")
    analysis_df = merged_df.copy()
    analysis_df['가격_차이_순위'] = analysis_df['매매'].rank(ascending=False)
    analysis_df['빵가격_순위'] = analysis_df['평균_빵가격'].rank(ascending=False)
    analysis_df['순위_차이'] = abs(analysis_df['가격_차이_순위'] - analysis_df['빵가격_순위'])

    print("\n아파트 가격 상위 5개 지역:")
    print(analysis_df.nlargest(5, '매매')[['구분', '매매', '평균_빵가격']])
    print("\n빵 가격 상위 5개 지역:")
    print(analysis_df.nlargest(5, '평균_빵가격')[['구분', '매매', '평균_빵가격']])
    print("\n순위 차이가 가장 큰 5개 지역 (불일치도가 높은 지역):")
    print(analysis_df.nlargest(5, '순위_차이')[['구분', '매매', '평균_빵가격', '순위_차이']])

    print(f"\n상관계수: {correlation:.3f}")
    if correlation > 0:
        print("양의 상관관계가 있습니다: 아파트 가격이 높은 지역일수록 빵 가격도 높은 경향이 있습니다.")
    else:
        print("음의 상관관계가 있습니다: 아파트 가격이 높은 지역일수록 빵 가격이 낮은 경향이 있습니다.")
```


```
=== 지역별 상세 데이터 ===

아파트 가격 상위 5개 지역:
          구분    매매      평균_빵가격
16   부산광역시 수영구  2115  4491.962500
18  부산광역시 해운대구  1719  4756.737500
11   부산광역시 동래구  1593  4343.842857
10    부산광역시 남구  1377  4288.028571
8     부산광역시 금정구  1304  4136.557143

빵 가격 상위 5개 지역:
          구분    매매      평균_빵가격
18  부산광역시 해운대구  1719  4756.737500
22    울산광역시 중구  1078  4732.693750
2           김해시   704  4729.475000
19    울산광역시 남구  1150  4710.027083
4           양산시   732  4677.233333

순위 차이가 가장 큰 5개 지역 (불일치도가 높은 지역):
          구분    매매      평균_빵가격  순위_차이
8   부산광역시 금정구  1304  4136.557143   17.0
1         거창군   608  4675.137500   13.0
2          김해시   704  4729.475000   13.0
9   부산광역시 기장군   982  3979.342857   13.0
6         통영시   582  4632.825000   11.0

상관계수: 0.130
양의 상관관계가 있습니다: 아파트 가격이 높은 지역일수록 빵 가격도 높은 경향이 있습니다.
```

In [10]:
```
import pandas as pd
import numpy as np
gbile_path ='./cafedata/gyeongbuk-pricedata.csv'
gbdf = pd.read_csv(gbile_path)
gbdf.head()
```

Out[10]:

	뚜레쥬르 지점	뚜레쥬르 경산옥곡	뚜레쥬르 경산평	뚜레쥬르 경주중앙	뚜레쥬르 경주총효서라벌중앙	뚜레쥬르 구미형곡중앙	뚜레쥬르 김천부곡센트럴	뚜레쥬르 문경중앙	뚜레쥬르 안동정하	뚜레쥬르 옥동호반	...	뚜레쥬르 월배점정	뚜레쥬르 현풍테크노	뚜레쥬르 동대구해모로	뚜레쥬르 동대구더샵디어엘로	뚜레쥬르 북현GS	뚜레쥬르 침산명성푸르지오	뚜레쥬르 서대구센트럴자이	뚜레쥬르 대구신평리	뚜레쥬르 수성3가캐슬	뚜레쥬르 대구달성공원역
0	마늘 단짝 고구마	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	...	5300.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	4900.0	NaN	4900.0
1	깊은 밤 뺑스위스	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	...	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	4300.0	NaN	4300.0
2	BELT 샌드위치	NaN	6900.0	NaN	6900.0	6900.0	6900.0	6900.0	6900.0	6900.0	...	7300.0	6900.0	NaN	6900.0	7100.0	8000.0	6900.0	6900.0	NaN	6900.0
3	BLT콤 샐러드	NaN	NaN	8500.0	8500.0	NaN	NaN	8500.0	NaN	8500.0	...	NaN	NaN	NaN	NaN	NaN	NaN	8500.0	NaN	NaN	8500.0
4	쉬림프 에그 샐러드	NaN	NaN	NaN	NaN	NaN	NaN	10500.0	NaN	NaN	...	11000.0	NaN	NaN	NaN	10500.0	NaN	10500.0	NaN	NaN	NaN

5 rows × 32 columns

In [11]:
```
import re

def categorize_menu(gbdf):
    # 키워드 기반 카테고리 매핑 딕셔너리
    category_keywords = {
        '샌드위치류': ['샌드위치', 'BELT', 'BLT', 'V.E.L.T'],
        '샐러드류': ['샐러드'],
        '식빵류': ['식빵', '우유롤', '우유 브레드', '소버식빵'],
        '크림빵': ['크림가득 메론빵','마담 엘그레이 크림빵','숯진유유크림빵','겹겹이 연유 크림 데니쉬','사르르 고구마케이크빵','사르르 우유크림빵','빵속에리얼초코','카페모카크림빵','까까웨뜨'],
        '피자빵,고로케': ['고로케', '소시지브레드','피자토스트','NEW어니언소시지포카치아'],
        '파이/패스트리': ['바통쉬크레','크라상','애플파이','유자파이'],
        '간식빵': ['소금버터롤','치즈방앗간','깨찰빵','소보로빵','오리지널 커피번','카페모카빵','꽈배기','옛날 단팥 도넛',r'^단팥빵$','단팥소보로빵'],
        '신제품': ['아구마구', '단짝', '뺑스위스']
    }

    # 새로운 카테고리 컬럼 생성
    gbdf['카테고리'] = '기타'  # 기본값

    # 각 메뉴명에 대해 카테고리 매핑
    for idx, menu_name in enumerate(gbdf['뚜레쥬르 지점']):
        if pd.isna(menu_name):  # null 체크
            continue

        menu_name = str(menu_name).lower()  # 소문자 변환

        # 각 카테고리의 키워드 체크
        for category, keywords in category_keywords.items():
            if any(keyword.lower() in menu_name for keyword in keywords):
                gbdf.loc[idx, '카테고리'] = category
                break

    return gbdf

def analyze_categories_by_store(gbdf):
    # 매장별 카테고리별 기본 통계
    stores = gbdf.columns[1:-1]  # 첫 번째 열(메뉴명)과 마지막 열(카테고리) 제외

    # 카테고리별 기본 통계
    category_stats = pd.DataFrame()

    for store in stores:
        # 매장별 데이터 숫자로 변환 (오류 방지)
        gbdf[store] = pd.to_numeric(gbdf[store], errors='coerce')

        temp = gbdf.groupby('카테고리').agg({store: 'mean'})
        temp.reset_index(inplace=True)
        temp.rename(columns={store: '평균 가격'}, inplace=True)
        temp['매장명'] = store
        category_stats = pd.concat([category_stats, temp], axis=0)

    return category_stats

def pivot_store_category(stats):
    # 피벗 테이블 생성
    pivot_table = stats.pivot_table(index='매장명', columns='카테고리', values='평균 가격', aggfunc='mean')
    pivot_table=pivot_table.round(1)
    pivot_table.reset_index(inplace=True)
    return pivot_table

# 데이터 로드 및 처리
def process_bakery_data(gbile_path):
    # CSV 파일 읽기
```

```
    gbdf = pd.read_csv(gbile_path)

    # 카테고리 지정
    gbdf = categorize_menu(gbdf)

    # 매장별 카테고리별 분석
    stats = analyze_categories_by_store(gbdf)

    # 피벗 테이블 생성
    pivot_table = pivot_store_category(stats)

    return gbdf, pivot_table

# 파일 처리 및 결과 생성
gbdf, pivot_table = process_bakery_data(gbile_path)

# 카테고리화된 데이터 및 매장별 통계 표시
from IPython.display import display

# print("카테고리화된 가격 데이터 (처음 5개 행)")
# display(gbdf.head())
#########################################################################################################################

storeinfo_filepath='./adress_process/gyeongbuk_adress.csv'

def process_address(address):
    try:
        # 수동 수정
        if address == '경기도 동탄지성로469번길 60 5단지 상가1동107호,108호,109호':
            return '경기도 화성시'

        # 정규표현식으로 '충청북도 XX시' 추출
        match = (
            re.match(r'경상북도\s+\w+시', address) or
            re.match(r'경상북도\s+\w+군', address) or
            re.match(r'대구광역시\s+\w+구', address) or
            re.match(r'대구광역시\s+\w+군', address)
        )
        if match:
            return match.group()

        # 기본값 반환
        return address
    except Exception as e:
        print(f"주소 처리 중 오류 발생: {address}, {e}")
        return address

def load_store_info(storeinfo_filepath):
    store_info = pd.read_csv(storeinfo_filepath)
    # 주소 컬럼 처리
    store_info['주소'] = store_info['주소'].apply(process_address)
    return store_info



def process_bakery_data(price_filepath, store_info_filepath):
    # 가격 데이터 로드
    gbdf = pd.read_csv(price_filepath)

    # 매장 정보 데이터 로드
    store_info = load_store_info(store_info_filepath)

    # 카테고리 지정
    gbdf = categorize_menu(gbdf)

    # 매장별 카테고리별 분석
    stats = analyze_categories_by_store(gbdf)

    # 피벗 테이블 생성 후 매장 정보 병합
    pivot_table = pivot_store_category(stats)
    result = pd.merge(pivot_table, store_info,
                      left_on='매장명',
                      right_on='매장',
                      how='left')

    # 컬럼 순서 재정렬
    columns = ['매장명', '주소', '지역'] + [col for col in result.columns
              if col not in ['매장명', '매장', '주소', '지역']]
    result = result[columns]

    return gbdf, result

# 실제 파일 경로로 호출
gbdf, result = process_bakery_data('./cafedata/gyeongbuk-pricedata.csv',
                                   './adress_process/gyeongbuk_adress.csv')

# 결과 출력
print("\n매장별 카테고리별 평균 가격 (주소 정보 포함)")
display(result)

# result.to_csv('./anal_gyeongsang/?시별_카테고리_평균가격.csv', encoding='utf-8-sig')
```

매장별 카테고리별 평균 가격 (주소 정보 포함)

	매장명	주소	지역	간식빵	기타	샌드위치류	샐러드류	식빵류	신제품	크림빵	파이/패스트리	피자빵,고로케
0	뚜레쥬르 경북울진	경상북도 울진군	경북	2642.9	4125.0	7100.0	NaN	5190.9	4600.0	3357.1	2933.3	3100.0
1	뚜레쥬르 경북청도	경상북도 청도군	경북	2987.5	4076.7	7187.5	9400.0	5055.6	4600.0	3814.3	3066.7	3025.0
2	뚜레쥬르 경산옥곡	경상북도 경산시	경북	2480.0	3921.6	7300.0	NaN	4942.9	4600.0	3675.0	3200.0	3025.0
3	뚜레쥬르 경산정평	경상북도 경산시	경북	3100.0	3862.8	7160.0	8333.3	5185.7	4600.0	3514.3	2933.3	3116.7
4	뚜레쥬르 경주중앙	경상북도 경주시	경북	2887.5	3963.6	8500.0	8500.0	5150.0	4600.0	3833.3	2966.7	3120.0
5	뚜레쥬르 경주충효서라벌중앙	경상북도 경주시	경북	2825.0	4314.6	7311.1	NaN	5162.5	4600.0	3500.0	2933.3	3650.0
6	뚜레쥬르 구미형곡중앙	경상북도 구미시	경북	2940.0	4329.4	7177.8	NaN	4976.9	4600.0	3580.0	2800.0	3300.0
7	뚜레쥬르 김천부곡센트럴	경상북도 김천시	경북	3100.0	4029.3	7128.6	NaN	4788.9	4600.0	3800.0	2933.3	3025.0
8	뚜레쥬르 대구달성공원역	대구광역시 중구	경북	2616.7	3950.0	7283.3	8500.0	4930.0	4600.0	3250.0	2933.3	3283.3
9	뚜레쥬르 대구대봉	대구광역시 남구	경북	3177.8	4261.9	7044.4	9966.7	5215.4	4600.0	3600.0	3233.3	3316.7
10	뚜레쥬르 대구본리	대구광역시 달서구	경북	2888.9	3858.8	7000.0	NaN	5075.0	4600.0	3700.0	3133.3	3420.0
11	뚜레쥬르 대구신평리	대구광역시 서구	경북	2680.0	4156.2	7133.3	8300.0	4714.3	4600.0	3360.0	2933.3	3080.0
12	뚜레쥬르 대구용산	대구광역시 달서구	경북	3100.0	4191.8	7100.0	8350.0	5109.1	4600.0	3233.3	2933.3	3040.0
13	뚜레쥬르 도청신도시	경상북도 예천군	경북	2620.0	4237.2	7066.7	NaN	4983.3	4600.0	3680.0	2800.0	3333.3
14	뚜레쥬르 동대구더샵디어엘로	대구광역시 동구	경북	2916.7	4252.8	7042.9	8300.0	4800.0	4600.0	4200.0	2933.3	3133.3
15	뚜레쥬르 동대구해모로	대구광역시 동구	경북	2944.4	4097.0	NaN	NaN	5050.0	4600.0	3514.3	3100.0	3200.0
16	뚜레쥬르 문경중앙	경상북도 문경시	경북	2872.7	4057.4	7320.0	8925.0	4860.0	4933.3	3825.0	3066.7	3180.0
17	뚜레쥬르 북현GS	대구광역시 북구	경북	3125.0	3975.0	7242.9	9100.0	5090.0	4600.0	3400.0	3166.7	3300.0
18	뚜레쥬르 서대구센트럴자이	대구광역시 서구	경북	2925.0	4010.5	7260.0	10500.0	5037.5	4933.3	3525.0	2800.0	3025.0
19	뚜레쥬르 수성3가캐슬	대구광역시 수성구	경북	3000.0	4053.5	NaN	NaN	5000.0	NaN	3637.5	3100.0	3116.7
20	뚜레쥬르 안동정하	경상북도 안동시	경북	3216.7	4170.8	6920.0	8025.0	5207.7	4933.3	3314.3	3100.0	3142.9
21	뚜레쥬르 영주가흥제일	경상북도 영주시	경북	3250.0	4128.6	7088.9	8500.0	4926.7	4933.3	3337.5	2933.3	3142.9
22	뚜레쥬르 영천망정	경상북도 영천시	경북	2657.1	4192.5	7320.0	8450.0	4908.3	4600.0	3857.1	3600.0	3040.0
23	뚜레쥬르 옥동호반	경상북도 안동시	경북	3090.9	4408.1	7225.0	8175.0	5122.2	4600.0	3557.1	3100.0	2950.0
24	뚜레쥬르 월배삼정	대구광역시 달서구	경북	3900.0	4716.9	6883.3	11000.0	5206.2	4800.0	4144.4	3100.0	3400.0
25	뚜레쥬르 침산명성푸르지오	대구광역시 북구	경북	3355.6	4324.0	7516.7	NaN	5011.1	4600.0	3300.0	3200.0	3100.0
26	뚜레쥬르 포항유강	경상북도 포항시	경북	2700.0	4289.2	7150.0	NaN	5090.9	4600.0	3350.0	3100.0	3200.0
27	뚜레쥬르 포항이동	경상북도 포항시	경북	2487.5	4194.6	7157.1	8350.0	5281.8	4933.3	3525.0	3066.7	3100.0
28	뚜레쥬르 포항자이	경상북도 포항시	경북	2987.5	4297.1	7066.7	8400.0	4620.0	4600.0	3233.3	2800.0	3333.3
29	뚜레쥬르 포항중앙	경상북도 포항시	경북	3142.9	4328.6	7080.0	7300.0	4955.6	4600.0	3900.0	2800.0	3200.0
30	뚜레쥬르 현풍테크노	대구광역시 달성군	경북	2885.7	3916.7	7111.1	NaN	5010.0	4600.0	3460.0	3100.0	3350.0

In [12]:
```
grouped_data = result.groupby('주소')[['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']].mean().mean(axis=1).sort_values(ascending=False)

# groupby 결과를 데이터프레임으로 변환
grouped_df = pd.DataFrame(grouped_data).reset_index()

# 컬럼명 변경
grouped_df.columns = ['주소', '평균가격']

# CSV 파일로 저장
grouped_df.to_csv('anal_gyeongsang/average_allbread_gb.csv', index=False, encoding='utf-8-sig')
grouped_df
```

Out[12]:

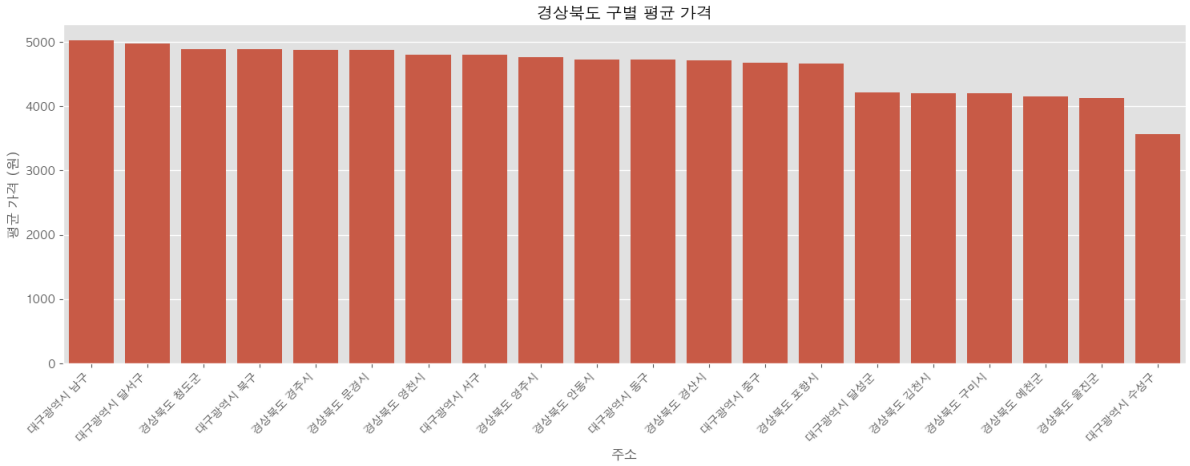
	주소	평균가격
0	대구광역시 남구	5019.287500
1	대구광역시 달서구	4974.658333
2	경상북도 청도군	4892.075000
3	대구광역시 북구	4888.000000
4	경상북도 경주시	4877.462500
5	경상북도 문경시	4872.837500
6	경상북도 영천시	4804.062500
7	대구광역시 서구	4800.418750
8	경상북도 영주시	4764.075000
9	경상북도 안동시	4730.006250
10	대구광역시 동구	4729.862500
11	경상북도 경산시	4718.718750
12	대구광역시 중구	4674.575000
13	경상북도 포항시	4660.258333
14	대구광역시 달성군	4216.685714
15	경상북도 김천시	4196.542857
16	경상북도 구미시	4196.385714
17	경상북도 예천군	4154.757143
18	경상북도 울진군	4132.028571
19	대구광역시 수성구	3570.840000

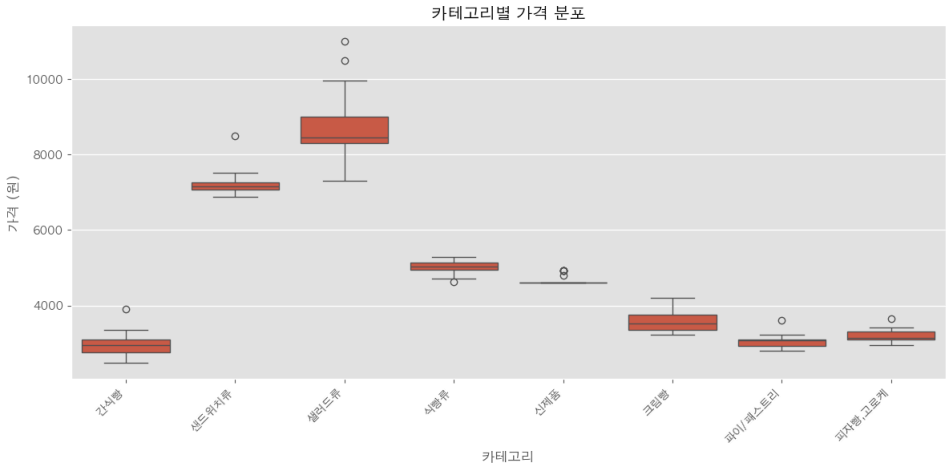
In [13]:
```
categories = ['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']

# 각 카테고리별로 구의 평균 가격 계산
grouped_data = {}
for category in categories:
    grouped_data[category] = result.groupby('주소')[category].mean().round(2)

# 데이터프레임 생성
grouped_df = pd.DataFrame(grouped_data)

# CSV 파일로 저장
grouped_df.to_csv('anal_gyeongsang/average_categorized_gbshop.csv', encoding='utf-8-sig')
grouped_df
```

Out[13]:

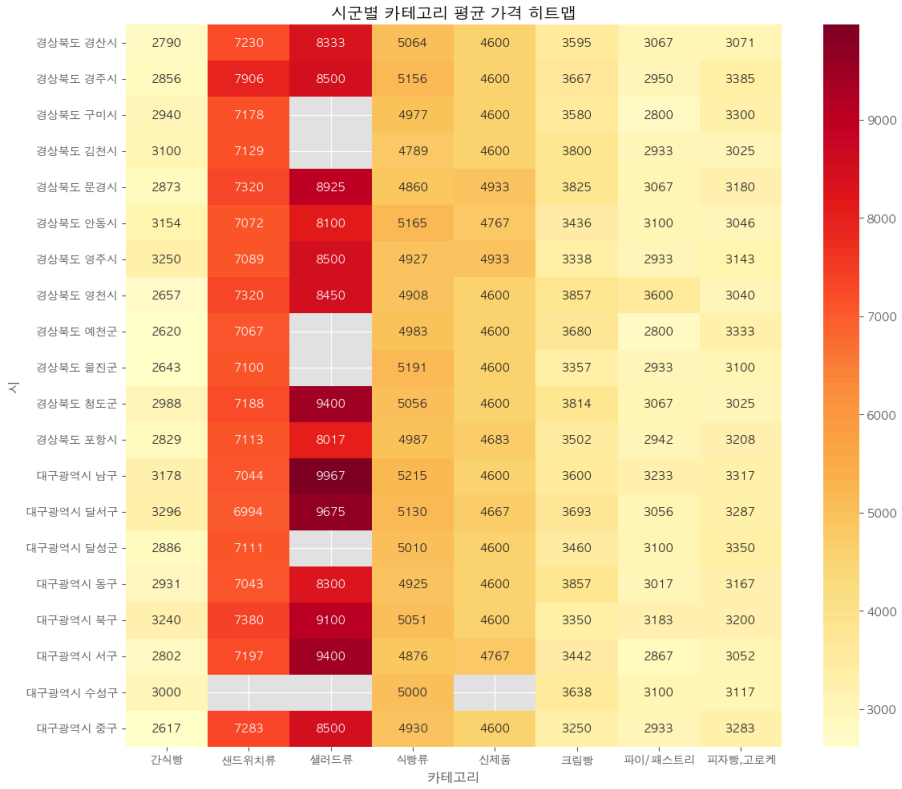
주소	간식빵	샌드위치류	샐러드류	식빵류	신제품	크림빵	파이/패스트리	피자빵,고로케
경상북도 경산시	2790.00	7230.00	8333.30	5064.30	4600.00	3594.65	3066.65	3070.85
경상북도 경주시	2856.25	7905.55	8500.00	5156.25	4600.00	3666.65	2950.00	3385.00
경상북도 구미시	2940.00	7177.80	NaN	4976.90	4600.00	3580.00	2800.00	3300.00
경상북도 김천시	3100.00	7128.60	NaN	4788.90	4600.00	3800.00	2933.30	3025.00
경상북도 문경시	2872.70	7320.00	8925.00	4860.00	4933.30	3825.00	3066.70	3180.00
경상북도 안동시	3153.80	7072.50	8100.00	5164.95	4766.65	3435.70	3100.00	3046.45
경상북도 영주시	3250.00	7088.90	8500.00	4926.70	4933.30	3337.50	2933.30	3142.90
경상북도 영천시	2657.10	7320.00	8450.00	4908.30	4600.00	3857.10	3600.00	3040.00
경상북도 예천군	2620.00	7066.70	NaN	4983.30	4600.00	3680.00	2800.00	3333.30
경상북도 울진군	2642.90	7100.00	NaN	5190.90	4600.00	3357.10	2933.30	3100.00
경상북도 청도군	2987.50	7187.50	9400.00	5055.60	4600.00	3814.30	3066.70	3025.00
경상북도 포항시	2829.48	7113.45	8016.67	4987.08	4683.32	3502.08	2941.68	3208.32
대구광역시 남구	3177.80	7044.40	9966.70	5215.40	4600.00	3600.00	3233.30	3316.70
대구광역시 달서구	3296.30	6994.43	9675.00	5130.10	4666.67	3692.57	3055.53	3286.67
대구광역시 달성군	2885.70	7111.10	NaN	5010.00	4600.00	3460.00	3100.00	3350.00
대구광역시 동구	2930.55	7042.90	8300.00	4925.00	4600.00	3857.15	3016.65	3166.65
대구광역시 북구	3240.30	7379.80	9100.00	5050.55	4600.00	3350.00	3183.35	3200.00
대구광역시 서구	2802.50	7196.65	9400.00	4875.90	4766.65	3442.50	2866.65	3052.50
대구광역시 수성구	3000.00	NaN	NaN	5000.00	NaN	3637.50	3100.00	3116.70
대구광역시 중구	2616.70	7283.30	8500.00	4930.00	4600.00	3250.00	2933.30	3283.30

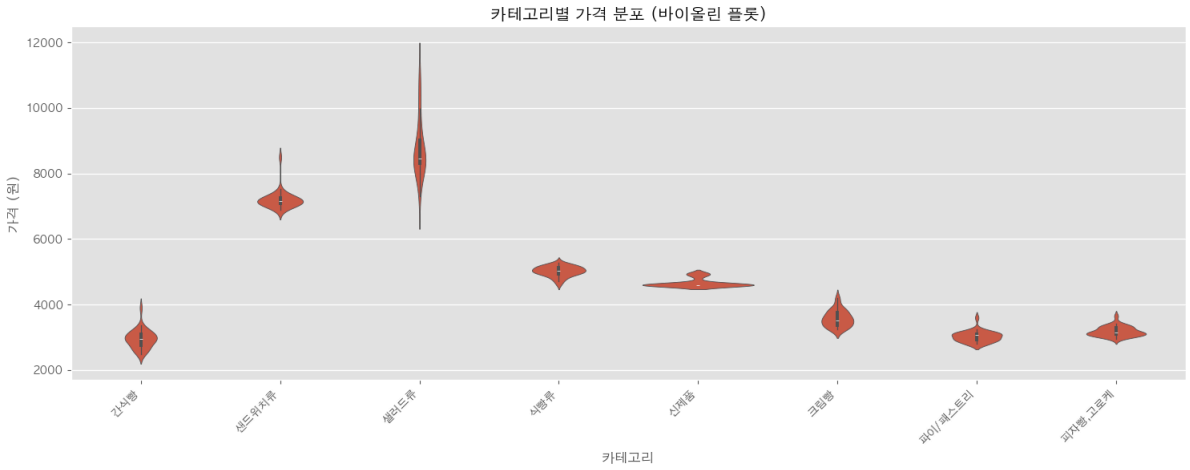
In [14]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
from matplotlib import font_manager, rc

# Mac OS 용 폰트 설정
plt.rc('font', family='AppleGothic') # 맥용 폰트 설정

# 그래프 기본 설정
plt.rcParams['axes.unicode_minus'] = False
plt.style.use('ggplot')


# 1. 구별 전체 평균 가격 분석
plt.figure(figsize=(15, 6))
grouped_data = result.groupby('주소')[['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']].mean().mean(axis=1).sort_values(ascending=False)

sns.barplot(x=grouped_data.index, y=grouped_data.values)
plt.title('경상북도 구별 평균 가격')
plt.xticks(rotation=45, ha='right')
plt.ylabel('평균 가격 (원)')
plt.tight_layout()
plt.show()

# 2. 카테고리별 가격 분포 (박스플롯)
plt.figure(figsize=(12, 6))
categories = ['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']
data_melted = pd.melt(result, value_vars=categories)

sns.boxplot(x='variable', y='value', data=data_melted)
plt.title('카테고리별 가격 분포')
plt.xticks(rotation=45, ha='right')
plt.xlabel('카테고리')
plt.ylabel('가격 (원)')
plt.tight_layout()
plt.show()
```





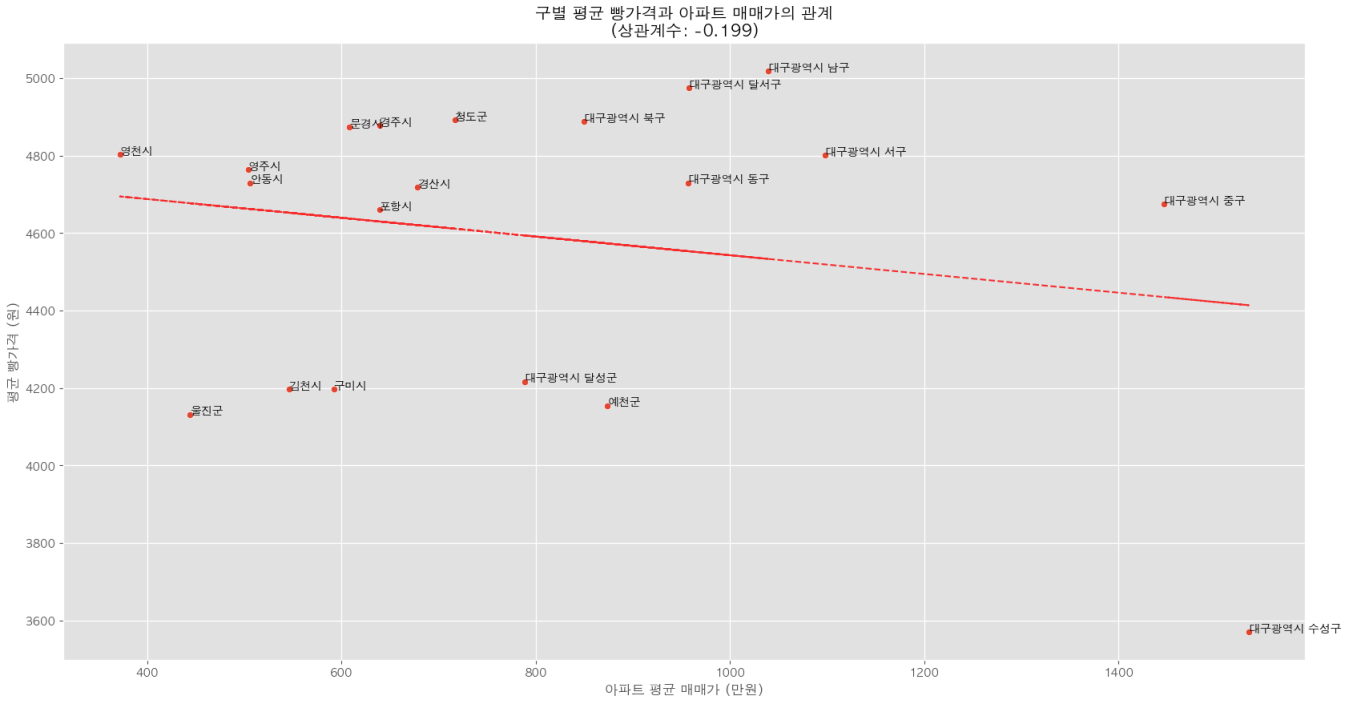
In [15]:
```python
# 3. 구별/카테고리별 평균 가격 히트맵
plt.figure(figsize=(12, 10))
pivot_data = result.groupby('주소')[categories].mean()
sns.heatmap(pivot_data, annot=True, fmt='.0f', cmap='YlOrRd')
plt.title('시군별 카테고리 평균 가격 히트맵')
plt.ylabel('시')
plt.xlabel('카테고리')
plt.tight_layout()
plt.show()

# 5. 카테고리별 가격 분포 (바이올린 플롯)
plt.figure(figsize=(15, 6))
sns.violinplot(x='variable', y='value', data=data_melted)
plt.title('카테고리별 가격 분포 (바이올린 플롯)')
plt.xticks(rotation=45, ha='right')
plt.xlabel('카테고리')
plt.ylabel('가격 (원)')
plt.tight_layout()
plt.show()
```





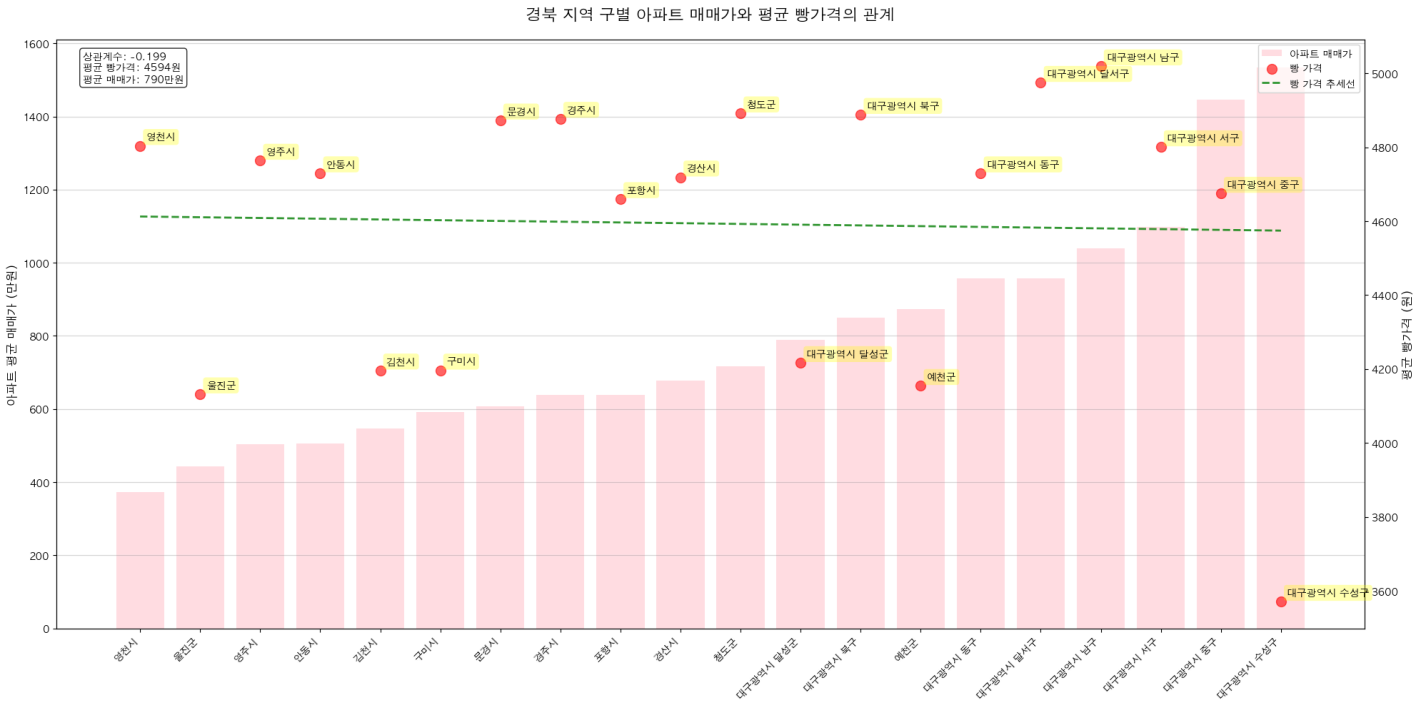
In [16]:
```python
# 1. 구별 평균 빵가격 계산
categories = ['간식빵', '샌드위치류', '샐러드류', '식빵류', '신제품', '크림빵', '파이/패스트리', '피자빵,고로케']
bread_price_by_district = result.groupby('주소')[categories].mean().mean(axis=1).reset_index()
bread_price_by_district.columns = ['구분', '평균_빵가격']
# '경상남도'제거
bread_price_by_district['구분'] = bread_price_by_district['구분'].str.replace('경상북도', '').str.strip()

# 아파트 가격 데이터 전처리
apt_price = pd.read_csv('anal_gyeongsang/gyeongbuk_apt_price.csv')
# '경상남도'와 '구' 제거
apt_price['구분'] = apt_price['구분'].str.replace('경상북도', '').str.strip()

apt_price['매매'] = pd.to_numeric(apt_price['매매'].str.replace(',', ''), errors='coerce')
apt_price = apt_price.dropna()  # 결측치 제거


# 데이터 확인
print("전처리 후 구별 빵가격 데이터:")
print(bread_price_by_district)
print("\n전처리 후 아파트 가격 데이터:")
print(apt_price)

# 데이터 병합
merged_df = pd.merge(bread_price_by_district, apt_price[['구분', '매매']], on='구분', how='inner')
print("\n병합된 데이터:")
print(merged_df)

# 시각화
if not merged_df.empty:
    plt.figure(figsize=(20, 10))
    sns.scatterplot(data=merged_df, x='매매', y='평균_빵가격')

    # 추세선 추가
    x = merged_df['매매'].values
    y = merged_df['평균_빵가격'].values
    z = np.polyfit(x, y, 1)
    p = np.poly1d(z)
    plt.plot(x, p(x), "r--", alpha=0.8)

    # 각 점에 구 이름 표시
    for idx, row in merged_df.iterrows():
        plt.annotate(row['구분'], (row['매매'], row['평균_빵가격']))

    correlation = merged_df['평균_빵가격'].corr(merged_df['매매'])
    plt.title(f'구별 평균 빵가격과 아파트 매매가의 관계\n(상관계수: {correlation:.3f})')
    plt.xlabel('아파트 평균 매매가 (만원)')
    plt.ylabel('평균 빵가격 (원)')

    print(f"\n상관계수: {correlation:.3f}")
    if correlation > 0:
        print("양의 상관관계가 있습니다: 아파트 가격이 높은 구일수록 빵 가격도 높은 경향이 있습니다.")
    else:
        print("음의 상관관계가 있습니다: 아파트 가격이 높은 구일수록 빵 가격이 낮은 경향이 있습니다.")
```

```
전처리 후 구별 빵가격 데이터:
             구분        평균_빵가격
0          경산시  4718.718750
1          경주시  4877.462500
2          구미시  4196.385714
3          김천시  4196.542857
4          문경시  4872.837500
5          안동시  4730.006250
6          영주시  4764.075000
7          영천시  4804.062500
8          예천군  4154.757143
9          울진군  4132.028571
10         청도군  4892.075000
11         포항시  4660.258333
12   대구광역시 남구  5019.287500
13  대구광역시 달서구  4974.658333
14  대구광역시 달성군  4216.685714
15   대구광역시 동구  4729.862500
16   대구광역시 북구  4888.000000
17   대구광역시 서구  4800.418750
18  대구광역시 수성구  3570.840000
19   대구광역시 중구  4674.575000

전처리 후 아파트 가격 데이터:
             구분    매매   전세
0          경산시   678  492
1          경주시   639  484
2          고령군   439  297
3          구미시   592  465
4          김천시   546  439
5          문경시   608  420
6          봉화군   407  294
7          상주시   451  286
8          성주군   415  288
9          안동시   506  368
10         영주시   504  340
11         영천시   372  273
12         예천군   874  644
13         울진군   444  308
14         청도군   717  515
15         칠곡군   414  319
16         포항시   639  540
17   대구광역시 남구  1039  678
18  대구광역시 달서구   958  640
19  대구광역시 달성군   789  544
20   대구광역시 동구   957  633
21   대구광역시 북구   850  624
22   대구광역시 서구  1098  609
23  대구광역시 수성구  1534  852
24   대구광역시 중구  1447  847

병합된 데이터:
             구분        평균_빵가격    매매
0          경산시  4718.718750   678
1          경주시  4877.462500   639
2          구미시  4196.385714   592
3          김천시  4196.542857   546
4          문경시  4872.837500   608
5          안동시  4730.006250   506
6          영주시  4764.075000   504
7          영천시  4804.062500   372
8          예천군  4154.757143   874
9          울진군  4132.028571   444
10         청도군  4892.075000   717
11         포항시  4660.258333   639
12   대구광역시 남구  5019.287500  1039
13  대구광역시 달서구  4974.658333   958
14  대구광역시 달성군  4216.685714   789
15   대구광역시 동구  4729.862500   957
16   대구광역시 북구  4888.000000   850
17   대구광역시 서구  4800.418750  1098
18  대구광역시 수성구  3570.840000  1534
19   대구광역시 중구  4674.575000  1447

상관계수: -0.199
음의 상관관계가 있습니다: 아파트 가격이 높은 구일수록 빵 가격이 낮은 경향이 있습니다.
```



In [17]:
```python
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# 시각화 기본 설정
plt.style.use('default')  # 기본 스타일 사용
plt.rcParams['figure.figsize'] = (15, 10)
plt.rcParams['font.family'] = 'AppleGothic'

if not merged_df.empty:
    # 데이터 정렬
    sorted_df = merged_df.sort_values(by='매매')

    # 메인 그래프 생성
    fig, ax1 = plt.subplots(figsize=(20, 10))

    # 기본 그리드 제거
    ax1.grid(False)

    # 바차트 배경 (매매가)
    ax1.bar(range(len(sorted_df)), sorted_df['매매'], alpha=0.5, color='pink', label='아파트 매매가')

    # 산점도를 위한 두 번째 y축 생성
    ax2 = ax1.twinx()
    ax2.grid(False)

    # 산점도 그리기
    scatter = ax2.scatter(range(len(sorted_df)), sorted_df['평균_빵가격'],
                          s=100, alpha=0.6, color='red', label='빵 가격')

    # 추세선 추가
    z = np.polyfit(range(len(sorted_df)), sorted_df['평균_빵가격'], 1)
    p = np.poly1d(z)
    ax2.plot(range(len(sorted_df)), p(range(len(sorted_df))), "g--",
```

```
            linewidth=2, alpha=0.8, label='빵 가격 추세선')

# x축 레이블 설정 (45도 회전)
ax1.set_xticks(range(len(sorted_df)))
ax1.set_xticklabels(sorted_df['구분'], rotation=45, ha='right')

# 각 점에 구 이름 표시
for i, row in enumerate(sorted_df.itertuples()):
    ax2.annotate(row.구분,
                 (i, row.평균_빵가격),
                 xytext=(6, 6),
                 textcoords='offset points',
                 fontsize=10,
                 bbox=dict(boxstyle='round,pad=0.3', fc='yellow', alpha=0.3, ec='none'))
                 # arrowprops=dict(arrowstyle='->', connectionstyle='arc3,rad=0'))

# 평균선 추가
# ax2.axhline(y=sorted_df['평균_빵가격'].mean(), color='g', linestyle='--', alpha=0.3, label='평균 빵가격')
# ax1.axhline(y=sorted_df['매매'].mean(), color='b', linestyle='--', alpha=0.3, label='평균 매매가')

# y축 그리드만 추가
ax1.grid(True, axis='y', alpha=0.3, linestyle='-', color='gray')

# 축 레이블 설정
ax1.set_ylabel('아파트 평균 매매가 (만원)', fontsize=12)
ax2.set_ylabel('평균 빵가격 (원)', fontsize=12)

# 상관계수 계산
correlation = sorted_df['평균_빵가격'].corr(sorted_df['매매'])

# 그래프 제목 설정
plt.title('경북 지역 구별 아파트 매매가와 평균 빵가격의 관계', fontsize=16, pad=20)

# 통계 정보 추가
stats_text = f'상관계수: {correlation:.3f}\n'
stats_text += f'평균 빵가격: {sorted_df["평균_빵가격"].mean():.0f}원\n'
stats_text += f'평균 매매가: {sorted_df["매매"].mean():.0f}만원'
ax1.text(0.02, 0.98, stats_text,
         transform=ax1.transAxes,
         verticalalignment='top',
         bbox=dict(boxstyle='round', facecolor='white', alpha=0.8))

# 범례 추가
lines1, labels1 = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()
ax1.legend(lines1 + lines2, labels1 + labels2, loc='upper right')

# 여백 조정
plt.tight_layout()

# 그래프 표시
plt.show()

# 추가 분석 출력
print("\n=== 지역별 상세 데이터 ===")
analysis_df = merged_df.copy()
analysis_df['가격_차이_순위'] = analysis_df['매매'].rank(ascending=False)
analysis_df['빵가격_순위'] = analysis_df['평균_빵가격'].rank(ascending=False)
analysis_df['순위_차이'] = abs(analysis_df['가격_차이_순위'] - analysis_df['빵가격_순위'])

print("\n아파트 가격 상위 5개 지역:")
print(analysis_df.nlargest(5, '매매')[['구분', '매매', '평균_빵가격']])
print("\n빵 가격 상위 5개 지역:")
print(analysis_df.nlargest(5, '평균_빵가격')[['구분', '매매', '평균_빵가격']])
print("\n순위 차이가 가장 큰 5개 지역 (불일치도가 높은 지역):")
print(analysis_df.nlargest(5, '순위_차이')[['구분', '매매', '평균_빵가격', '순위_차이']])

print(f"\n상관계수: {correlation:.3f}")
if correlation > 0:
    print("양의 상관관계가 있습니다: 아파트 가격이 높은 지역일수록 빵 가격도 높은 경향이 있습니다.")
else:
    print("음의 상관관계가 있습니다: 아파트 가격이 높은 지역일수록 빵 가격이 낮은 경향이 있습니다.")
```




```
=== 지역별 상세 데이터 ===

아파트 가격 상위 5개 지역:
           구분     매매       평균_빵가격
18  대구광역시 수성구  1534  3570.840000
19   대구광역시 중구  1447  4674.575000
17   대구광역시 서구  1098  4800.418750
12   대구광역시 남구  1039  5019.287500
13  대구광역시 달서구   958  4974.658333

빵 가격 상위 5개 지역:
           구분     매매       평균_빵가격
12   대구광역시 남구  1039  5019.287500
13  대구광역시 달서구   958  4974.658333
10        청도군   717  4892.075000
16   대구광역시 북구   850  4888.000000
1         경주시   639  4877.462500

순위 차이가 가장 큰 5개 지역 (불일치도가 높은 지역):
           구분     매매       평균_빵가격  순위_차이
18  대구광역시 수성구  1534  3570.840000   19.0
7         영천시   372  4804.062500   13.0
8         예천군   874  4154.757143   11.0
19   대구광역시 중구  1447  4674.575000   11.0
6         영주시   504  4764.075000    9.0

상관계수: -0.199
음의 상관관계가 있습니다: 아파트 가격이 높은 지역일수록 빵 가격이 낮은 경향이 있습니다.
```

In [18]:
```
pip install adjustText
```

```
Requirement already satisfied: adjustText in /opt/anaconda3/lib/python3.12/site-packages (1.3.0)
Requirement already satisfied: numpy in /opt/anaconda3/lib/python3.12/site-packages (from adjustText) (1.26.4)
Requirement already satisfied: matplotlib in /opt/anaconda3/lib/python3.12/site-packages (from adjustText) (3.8.4)
Requirement already satisfied: scipy in /opt/anaconda3/lib/python3.12/site-packages (from adjustText) (1.13.1)
Requirement already satisfied: contourpy>=1.0.1 in /opt/anaconda3/lib/python3.12/site-packages (from matplotlib->adjustText) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /opt/anaconda3/lib/python3.12/site-packages (from matplotlib->adjustText) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in /opt/anaconda3/lib/python3.12/site-packages (from matplotlib->adjustText) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /opt/anaconda3/lib/python3.12/site-packages (from matplotlib->adjustText) (1.4.4)
Requirement already satisfied: packaging>=20.0 in /opt/anaconda3/lib/python3.12/site-packages (from matplotlib->adjustText) (23.2)
Requirement already satisfied: pillow>=8 in /opt/anaconda3/lib/python3.12/site-packages (from matplotlib->adjustText) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /opt/anaconda3/lib/python3.12/site-packages (from matplotlib->adjustText) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in /opt/anaconda3/lib/python3.12/site-packages (from matplotlib->adjustText) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in /opt/anaconda3/lib/python3.12/site-packages (from python-dateutil>=2.7->matplotlib->adjustText) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [19]:

```python
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

# 데이터 로드
people_df = pd.read_csv('./data_people/worked_data_people/people_daegu.csv')
people_df['유동인구 수'] = people_df['유동인구 수'].str.replace(',', '').astype(float)
people_df = people_df.rename(columns={'지역': '지역'})  # '지역' 열 이름 확인 및 정리

# 가상의 merged_df 생성 (샘플 데이터)
# 실제 데이터가 없을 경우 아래 데이터로 테스트 가능
# merged_df = pd.DataFrame({
#     '구분': ['대구광역시 남구', '대구광역시 동구', '대구광역시 북구', '대구광역시 서구'],
#     '평균_빵가격': [4500, 4700, 4300, 4600]
# })

# '구분'에서 대구 지역 데이터 추출 및 '지역'으로 열 이름 변경
daegu_df = merged_df[merged_df['구분'].str.contains('대구광역시')].copy()
daegu_df['구분'] = daegu_df['구분'].str.replace('대구광역시 ', '').str.strip()
daegu_df = daegu_df.rename(columns={'구분': '지역'})  # 병합 키에 맞게 열 이름 변경

# 병합
merged_df = pd.merge(
    daegu_df,
    people_df,
    on='지역',  # 병합 키로 '지역' 열 사용
    how='inner',  # 교집합 병합
    indicator=True
)

# 병합 결과 확인
if merged_df.empty:
    raise ValueError("병합된 데이터프레임이 비어 있습니다. 데이터가 일치하지 않습니다.")
merged_df = merged_df.drop(columns=['_merge'])  # 병합 상태 열 삭제

# 데이터 정렬
sorted_df = merged_df.sort_values(by='유동인구 수').dropna()

# 시각화
plt.figure(figsize=(15, 10))
fig, ax1 = plt.subplots()

# 바차트: 유동인구
bars = ax1.bar(range(len(sorted_df)), sorted_df['유동인구 수'], color='pink', alpha=0.5, label='유동인구 수')

# 산점도: 빵 가격
ax2 = ax1.twinx()
scatter = ax2.scatter(range(len(sorted_df)), sorted_df['평균_빵가격'], color='red', s=100, alpha=0.6, label='빵 가격')

# 추세선 추가 (빵 가격)
if len(sorted_df) > 1:
    z = np.polyfit(range(len(sorted_df)), sorted_df['평균_빵가격'], 1)
    p = np.poly1d(z)
    ax2.plot(range(len(sorted_df)), p(range(len(sorted_df))), "g--", alpha=0.8, label='빵 가격 추세선')

from adjustText import adjust_text



# x축 설정
ax1.set_xticks(range(len(sorted_df)))
ax1.set_xticklabels(sorted_df['지역'], rotation=45)
ax1.set_ylabel('유동인구 수 (명)')
ax2.set_ylabel('평균 빵가격 (원)')

# 제목 설정
plt.title('대구광역시 구별 유동인구와 평균 빵가격의 관계', fontsize=16)

# 그래프 표시
plt.tight_layout()
plt.show()
```

```
<Figure size 1500x1000 with 0 Axes>
```