# t1\_p1

```
import pandas as pd
import numpy as np
```

# 파일 로드

```
# 엑셀 파일 읽기

df_raw = pd.read_excel("data/basic1.xlsx", sheet_name='basic1')

# 첫 번째 열을 쉼표 기준으로 분리

df = df_raw.iloc[:, 0].str.split(",", expand=True)

# 컬럼명 설정

df.columns = ['id', 'age', 'city', 'f1', 'f2', 'f3', 'f4', 'f5']

# 타입 변환 (숫자 컬럼)

cols_to_numeric = ['age', 'f1', 'f2', 'f5']

df[cols_to_numeric] = df[cols_to_numeric].apply(pd.to_numeric, errors='coerce')
```

#### df.head()

	id	age	city	f1	f2	f3	f4	f5
0	id01	2.0	서울	NaN	0		ENFJ	91.297791
1	id02	9.0	서울	70.0	1		ENFJ	60.339826
2	id03	27.0	서울	61.0	1		ISTJ	17.252986
3	id04	75.0	서울	NaN	2		INFP	52.667078
4	id05	24.0	서울	85.0	2		ISFJ	29.269869

### df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 8 columns):
# Column Non-Null Count Dtype
0 id 100 non-null object
1 age 100 non-null float64
2 city 100 non-null object
3 f1 69 non-null float64
4 f2 100 non-null int64
       100 non-null object
   f3
6 f4
       100 non-null object
7 f5
        100 non-null float64
dtypes: float64(3), int64(1), object(4)
memory usage: 6.4+ KB
```

# 작업형 1유형 - 문제 1

### ★ 문제 설명

f1 컬럼의 결측치를 city 별 평균값으로 대체하고, 조건에 맞는 평균을 계산하시오.

### ◆ 조건

- 1. f1 컬럼의 결측치는 city 별 평균값으로 대체할 것
- 2. 결측치가 처리된 이후, 아래 조건을 만족하는 데이터만 추출:
  - city == '서울'

• f2 >= 1

3. 위 조건을 만족하는 행들의 f1 평균값을 소수점 둘째 자리까지 반올림하여 출력하시오.

#### ☑ 출력 형식

```
print(result) # 예시: 65.71
```

# 나의 풀이

```
df1= df.copy()
df1.groupby(['city'])['f1'].transform('mean')
     67.933333
0
1
     67.933333
2
     67.933333
     67.933333
3
     67.933333
95
     62.551724
96
     62.551724
97
     62.551724
98
     62.551724
     62.551724
Name: f1, Length: 100, dtype: float64
```

```
# f1 컬럼의 결측치는 city별 평균값으로 대체할 것

df1.loc[:,'f1']=df1['f1'].fillna(df1.groupby(['city'])['f1'].transform('mean'))
```

```
round(df1[(df1['city']=='서울')&(df1['f2']>=1)]['f1'].mean(),2)
```

67.45

# → 작업형 1유형 - 문제 2

### ★ 문제 설명

f5 컬럼에서 이상치를 적절한 기준에 따라 탐지한 뒤, 이를 처리하고 평균값을 구하시오.

#### ◆ 조건

- 1. 이상치 기준은 IQR 방식을 따른다:
  - Q1 = 1사분위수, Q3 = 3사분위수
  - IQR = Q3 Q1
  - 이상치 범위: f5 < Q1 1.5×IQR 또는 f5 > Q3 + 1.5×IQR
- 2. 이상치에 해당하는 값을 전체 f5 의 중앙값으로 대체한다.
- 3. 이상치가 처리된 f5 컬럼의 평균값을 소수점 둘째 자리까지 반올림하여 출력하시오.

### ☑ 출력 형식

```
print(result) # 예시: 51.27
```

# 나의 풀이

```
df2=df.copy()
```

```
med=df2['f5'].median()
q3= df2['f5'].quantile(.75)
q1= df2['f5'].quantile(.25)
iqr=q3-q1
over= df2['f5']>(q3+1.5*iqr)
below = df2['f5'] < (q1-1.5*iqr)
df2[over|below]= med
round(df2['f5'].mean(),2)</pre>
```

56.73

# 정답 풀이



# 



특정 조건을 만족하는 데이터에 대해 정렬 및 부분 통계를 수행하시오.

### ◆ 조건

- 1. f4 컬럼의 값이 'ENFJ' 인 데이터만 선택하시오.
- 2. 위 데이터에서 f5 컬럼을 기준으로 내림차순 정렬하시오.
- 3. 정렬된 데이터 중 상위 3개의 f5 평균값을 소수 둘째 자리까지 반올림하여 출력하시오.

round(df3[cond].sort\_values(['f5'],ascending=False).head(3)['f5'].mean(),2)

### ☑ 출력 형식

```
print(result) # 예시: 73.41
df3=df.copy()
df3.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 8 columns):
   Column Non-Null Count Dtype
0 id 100 non-null object
    age
           100 non-null
2 city 100 non-null object
3 f1
       69 non-null float64
4 f2 100 non-null int64
        100 non-null
   f3
                        object
        100 non-null
   f4
6
                         object
   f5
         100 non-null
                         float64
dtypes: float64(3), int64(1), object(4)
memory usage: 6.4+ KB
cond= df3['f4']=='ENFJ'
```

# 🥓 작업형 1유형 - 문제 4

#### ★ 문제 설명

f1 점수를 기준으로 새로운 파생 변수를 만들고, 그룹별 통계량을 확인하시오.

### ◆ 조건

- 1. f1 컬럼의 값이 70 이상이면 'high', 그 미만이면 'low' 로 구분하여 새로운 컬럼 f1\_grade 를 생성하시오.
  - 조건 기반 파생 변수 생성 방식 사용
- 2. 생성된 f1\_grade 그룹별로 f2 컬럼의 최빈값(가장 자주 등장하는 값)을 출력하시오.
  - value\_counts() 또는 mode() 활용 가능

#### ☑ 출력 예시

```
# 예시 출력 (실제 결과는 다를 수 있음)
high 그룹 → 2
low 그룹 → 1
```

출력 형식은 자유롭게 딕셔너리나 개별 출력으로 구성해도 무방함.

```
df4=df.copy()

df4['f1_grade']= np.where(df4['f1']>=70, 'high', 'low')

# df4.groupby(['f1_grade'])['f2'].mode()

df4[df4['f1_grade'] == 'high']['f2'].mode()[0]
```

```
df4[df4['f1_grade'] == 'low']['f2'].mode()[0]
```

나의 풀이

# 개념 정리

- **☑** agg()란?
- agg()는 집계 함수(aggregation)를 적용할 때 쓰는 메서드야.
- groupby() 뒤에 붙여서 각 그룹별로 원하는 통계 연산을 지정할 수 있어.

구분	agg()	transform()
목적	그룹별 요약값 반환	원래 데이터의 <mark>형태 유지</mark> 하며 변환값 반환
결과 형태	줄어듦 (group 수만큼)	원래 행 수 그대로 유지됨
예시 용도	그룹별 평균, 최댓값 등 <mark>통계 요약표</mark>	그룹 평균 등으로 <mark>결측치 대</mark> 체할 때
대표 예	<pre>df.groupby('city')['f1'].agg('mean')</pre>	<pre>df['f1'].fillna(df.groupby('city')['f1'].transform('mean'))</pre>

함수	역할	예시
<pre>groupby()</pre>	그룹 나누기	<pre>df.groupby('city')</pre>

함수	역할	예시
agg()	그룹별 요약값 계산	<pre>agg('mean'), agg(['mean','max'])</pre>
<pre>transform()</pre>	그룹별 통계값을 <mark>원본 형태로 반환</mark>	<pre>transform('mean')</pre>
lambda	사용자 정의 계산	agg(lambda x:)
idxmax()	최댓값의 <mark>인덱스(=위치)</mark> 반환	<pre>x.value_counts().idxmax()</pre>
idxmin()	최솟값의 인덱스 반환	<pre>x.value_counts().idxmin()</pre>

groupby, agg, transform, lambda는 실전에서 거의 세트처럼 등장하고, 여기에 idxmax, idxmin도 자주 붙어.

# 🥕 작업형 1유형 - 문제 5

### ★ 문제 설명

문자열 조건 필터링 후, 데이터 타입을 변환하고 통계값을 계산하시오.

#### ◆ 조건

- 1. f4 컬럼의 값 중 문자 'E' 를 포함하는 행만 필터링하시오.
- 2. 필터링된 데이터에서 f5 컬럼의 값을 정수형(int)으로 변환하시오.
- 3. 변환된 f5 컬럼의 평균값을 소수 둘째 자리까지 반올림하여 출력하시오.

#### ☑ 출력 형식

```
print(result) # 예시: 51.27

df5=df.copy()

round(df5[df5['f4'].str.contains('E')]['f5'].apply(lambda x : int(x)).mean(),2) # astype(int) 또는 apply(lambda x: int(x))

52.96
```

### 

### ★ 문제 설명

중복된 데이터를 제거하고, 재구조화를 통해 요약 통계를 계산하시오.

#### ◆ 조건

- 1. f4, f2 두 컬럼을 기준으로 중복된 행을 제거하시오.
- 2. 중복이 제거된 데이터에서 f4 를 행 인덱스로 하여, f2 의 평균을 계산하는 <mark>피벗 테이블</mark>을 생성하시오.
- 3. 결과 테이블을 출력하시오.

### ☑ 출력 예시

```
# 예시 (형태만 참고)
f2
f4
ENFJ 1.3
INFP 1.7
ISFJ 0.8
```

```
단, 출력 형식은 pivot_table() 결과 그대로 출력하면 충분함.
df6=df.copy()
df6.head(2)
df.shape
 (100, 8)
df.head(3)
      id
                   city
                         f1
                                f2
                                      f3
                                             f4
                                                   f5
             age
0
      id01
                                             ENFJ 91.297791
             2.0
                   서울
                                0
                         NaN
      id02
1
             9.0
                   서울
                         70.0
                                1
                                             ENFJ
                                                   60.339826
                                             ISTJ
2
      id03
            27.0
                  서울
                        61.0
                               1
                                                   17.252986
# help(df6.drop_duplicates)
df6 = df6.drop_duplicates(subset =['f4','f2'])
df6.shape
 (40, 8)
```

```
pivot=df6.pivot_table(index='f4',values='f2',aggfunc='mean')
pivot.shape
```

pivot.reset\_index(inplace=True)

pivot.shape

(16, 2)

(16, 1)