Pandas 합치기

4

서로 다른 데이터프레임 가로, 세로 합치기

가로합치기 concat()

예제 데이터

```
import pandas as pd
df1 = pd.DataFrame({'Name': ['Kim', 'Lee', 'Park'], 'Gender': ['M', 'M', 'F'],
                    'Mid': [90, 85, 95]})
df1
```

	Name	Gender	Mid
0	Kim	M	90
1	Lee	M	85
2	Park	F	95

```
df2 = pd.DataFrame({'Name': ['Wolfpack', 'Ray'],
                    'Mid': [90, 95]})
df2
```

	Name	Mid
0	Wolfpack	90
1	Ray	95

```
df3 = pd.DataFrame({'Name': ['Wolfpack', 'Ray'], 'Gender': ['M', 'F'],
                    'Mid': [90, 95]})
df3
```

	Name	Gender	Mid
0	Wolfpack	M	90
1	Ray	F	95

```
df4 = pd.DataFrame({'Name': ['Lee', 'Kim', 'Park', 'Wolfpack', 'Ray'],
                    'PopQuiz': [80, 85, 90, 95, 95]})
df4
```

	Name	PopQuiz
0	Lee	80
1	Kim	85
2	Park	90
3	Wolfpack	95
4	Ray	95

```
df5 = pd.DataFrame({'Name': ['Kim', 'Lee', 'Wolfpack', 'Ray'],
                    'Final': [90, 85, 95, 95]})
df5
```

	Name	Final
0	Kim	90
1	Lee	85
2	Wolfpack	95
3	Ray	95

가로 합치기 df.concat([df1, df2])

옵션이 설정된 것은 디폴트 옵션으로 사용하지 않으면 디폴트가

pd.concat([df1, df2], axis=0, join='outer')

- axis : 행 혹은 열 합치기 1=열 합치기
- join : outer=합집합, inner=교집합

두 데이터프레임이 공통변수가 정확하게 동일한 경우 pd.concat[[df1, df2]]

- **df1.append(df2) 동일함**

```
pd.concat([df1,df3])
```

	Name	Gender	Mid
0	Kim	M	90
1	Lee	M	85
2	Park	F	95
0	Wolfpack	M	90
1	Ray	F	95

```
df1.append(df3)
```

	Name	Gender	Mid
0	Kim	M	90
1	Lee	M	85
2	Park	F	95
0	Wolfpack	M	90
1	Ray	F	95

두 데이터프레임 변수가 다른 경우 - 없는 것은 결측치

두 데이터프레임 공통적으로 있는 변수 행 합치기, join='inner'

```
pd.concat([df1,df2])
```

	Gender	Mid	Name
0	M	90	Kim
1	M	85	Lee
2	F	95	Park
0	NaN	90	Wolfpack
1	NaN	95	Ray

```
pd.concat([df1,df2], join='inner')
```

	Name	Mid
0	Kim	90
1	Lee	85
2	Park	95
0	Wolfpack	90
1	Ray	95

세로 합치기 df.concat([df1, df2], axis=1)

두 데이터프레임 세로 병렬합치기

join='outer' (합집합 합치기)- 디폴트 : 행 개수가 적은 경우 NoN 결측값이 저장

join='inner' (교집합 합치기)- 행 개수가 적은 데이터 프레임 행 개수만 저장

```
pd.concat([df3,df1], axis=1)
```

	Name	Gender	Mid	Name	Gender	Mid
0	Wolfpack	M	90.0	Kim	M	90
1	Ray	F	95.0	Lee	M	85
2	NaN	NaN	NaN	Park	F	95

```
pd.concat([df1,df3], join='inner',axis=1)
```

	Name	Gender	Mid	Name	Gender	Mid
0	Kim	M	90	Wolfpack	M	90
1	Lee	M	85	Ray	F	95

세로합치기 merge

형식

pd.merge(left, right, how='inner', on=None)

- on=[['키변수1','키변수2']] 데이터 프레임 합치는 매칭 키 변수 지정, 없으면 병렬합치기
- how : inner(교집합), outer(합집합), left(왼쪽 데이터 프레임 키변수 사용), right(오른쪽 데이터프레임 키변수 사용)

```
df0=pd.concat([df1,df3])
df0
```

	Name	Gender	Mid
0	Kim	M	90
1	Lee	M	85
2	Park	F	95
0	Wolfpack	M	90
1	Ray	F	95

df4

	Name	PopQuiz
0	Lee	80
1	Kim	85
2	Park	90
3	Wolfpack	95
4	Ray	95

df5

	Name	Final
0	Kim	90
1	Lee	85
2	Wolfpack	95
3	Ray	95

병렬합치기

how='inner' (교집합) 디폴트

- df5에는 Park Final 데이터 값이 없어 df0, df5 merge 결과에는 Park 없음

```
pd.merge(df0,df4, on='Name')
```

	Name	Gender	Mid	PopQuiz
0	Kim	M	90	85
1	Lee	M	85	80
2	Park	F	95	90
3	Wolfpack	M	90	95
4	Ray	F	95	95

```
pd.merge(df0,df5, on='Name')
```

	Name	Gender	Mid	Final
0	Kim	M	90	90
1	Lee	M	85	85
2	Wolfpack	M	90	95
3	Ray	F	95	95

```
pd.merge(df0,df5, how='outer', on='Name')
```

	Name	Gender	Mid	Final
0	Kim	M	90	90.0
1	Lee	M	85	85.0
2	Park	F	95	NaN
3	Wolfpack	M	90	95.0
4	Ray	F	95	95.0

how='outer' 합집합 옵션으로

Final이 없는 Park에는 결측값 NaN 저장된다.

예제코딩

예제 데이터

기상청 기후 데이터 2018년

http://203.247.53.31/Stat_Notes/example_data/climate/2018climate.csv

```
import pandas as pd
url='http://203.247.53.31/Stat_Notes/example_data/climate/2018climate.csv'
df_c=pd.read_csv(url,encoding='ms949')
```

```
df_c.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34675 entries, 0 to 34674
Data columns (total 61 columns):
지점                  34675 non-null int64
일시                  34675 non-null object
평균기온(℃)            34630 non-null float64
최저기온(℃)            34672 non-null float64
최저기온 시각(hhmi)      34672 non-null float64
최고기온(℃)            34673 non-null float64
```

```
df_c.head(3)
```

	지점	일시	평균기온(℃)	최저기온(℃)	최저기온 시각(hhmi)	최고기온(℃)	최고기온 시각(hhmi)	강수계속시간(hr)	10분 최다 강수량(mm)
0	90	2018-01-01	1.0	-3.2	343.0	4.2	1443.0	NaN	NaN
1	90	2018-01-02	1.5	-2.1	2344.0	5.6	1358.0	NaN	NaN

관측지점 정보

```
import pandas as pd
url='http://203.247.53.31/Stat_Notes/example_data/climate/location.csv'
df_pos=pd.read_csv(url,encoding='ms949')
```

```
df_pos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 588 entries, 0 to 587
Data columns (total 2 columns):
관측지점    588 non-null int64
관측주소    588 non-null object
dtypes: int64(1), object(1)
memory usage: 9.3+ KB
```

```
df_pos.head(3)
```

	관측지점	관측주소
0	90	강원도 고성군 토성면 봉포5길 9 속초고층관측소
1	93	강원도 춘천시 신북읍 장본1길 12 춘천기상대
2	95	강원도 철원군 갈말읍 명성로179번길 26 철원자동기상관측소

관측지점 키변수 merge

- 매칭 키변수 지점-두 데이터프레임은 관측지점 번호를 가지고 있음
- df_c에는 (관측)지점 번호 정보만 있어 주소가 필요하다.
- 두 데이터프레임의 키변수의 이름이 상이 : 지점-관측지점

먼저 df_c 데이터프레임 지점-> 관측지점으로 변경

```
df_c.rename(columns={'지점':'관측지점'},inplace=True)
df_c.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34675 entries, 0 to 34674
Data columns (total 61 columns):
관측지점          34675 non-null int64
일시              34675 non-null object
평균기온(°C)        34630 non-null float64
```

pd.merge 활용 합치기

```
df_fin=pd.merge(df_pos, df_c, on='관측지점')
df_fin.head(3)
```

	관측지점	관측주소	일시	평균기온(°C)	최저기온(°C)	최저기온 시각(hhmi)	최고기온(°C)	최고기온 시각(hhmi)	강수계속시간(hr)
0	90	강원도 고성군 토성면 봉포5길 9 속초고층관측소	2018-01-01	1.0	-3.2	343.0	4.2	1443.0	NaN
1	90	강원도 고성군 토성면 봉포5길 9 속초	2018-	1.5	-2.1	2344.0	5.6	1358.0	NaN

관측주소 - 시도이름 만들기 str.split() / str.slice()

- str.split() 이용 관측주소 첫 문자열 시도명 가져오기
- str.slice() : 시도명 앞 2개 글자 가져오기

관측주소 빈도분석

- 2018년 기상 데이터이므로 지점마다 365개 관측치 있음

```
df_fin.관측주소.value_counts()
```

```
경상남도 함양군 함양읍 용평리 915-202 함양군자동기상관측소      365
강원도 영월군 영월읍 영월로 1894-25 영월자동기상관측소        365
경상북도 안동시 열루재1길 16 안동기상대                  365
충청북도 영동군 추풍령면 관리길 25-15 추풍령자동기상관측소      365
대구광역시 동구 효동로2길 10 대구기상지청                 365
강원도 강릉시 사천면 과학단지로 130 강원지방기상청           365
경상남도 의령군 의합대로 44-54 의령군자동기상관측소          365
Name: 관측주소, Length: 94, dtype: int64
```

' ' 공백, 1번 사용 첫 행 [0]이 시도명임 [1]은 고성군 토성면 ~

```
df_fin['시도이름']=df_fin.관측주소.str.split(' ',n=1,expand=True)[0]
```

```
df_fin.info()
```

```
안개 계속시간(hr)      580 non-null float64
시도이름             34310 non-null object
dtypes: float64(58), int64(1), object(4)
memory usage: 18.0+ MB
```

```
df_fin['시도이름'].head(3)
```

```
0    강원도
1    강원도
2    강원도
Name: 시도이름, dtype: object
```

시도이름의 앞 2 글자 시도명

```
df_fin['시도이름2']=df_fin['시도이름'].str.slice(0,2)
```

```
df_fin['시도이름2'].head(2)
```

```
0  강원
1  강원
Name: 시도이름2, dtype: object
```

```
기사                  5343 non-
안개 계속시간(hr)       580
시도이름              34310 n
시도이름2             34310 n
dtypes: float64(58), int64(1)
```

광역시 데이터만 가져오기

```
df_fin0=df_fin[(df_fin['시도이름2']=='대전') | (df_fin['시도이름2']=='대구')
    | (df_fin['시도이름2']=='울산') | (df_fin['시도이름2']=='부산')
    | (df_fin['시도이름2']=='광주') | (df_fin['시도이름2']=='인천')
    | (df_fin['시도이름2']=='서울')]
```

```
df_fin0.시도이름2.value_counts()
```

```
인천    1095
부산     365
대구     365
대전     365
서울     365
광주     365
울산     365
Name: 시도이름2, dtype: int64
```

```
df_fin0.shape
```

```
(3285, 64)
```

일별 대표 값 만들기

- df_max : 해당 시도 내 해당 일 최대 기온 관측지점 대표값
- df_p90 : 해당 시도 내 해당 일 상위 90% 관측지점 대표값

```
df_max=df_fin0.groupby(['시도이름2','일시']).max()
df_p90=df_fin0.groupby(['시도이름2','일시']).quantile(0.9)
```

인덱스 열 이름으로 설정

```
df_max.reset_index(inplace=True)
```

```
df_max.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2555 entries, 0 to 2554
Data columns (total 63 columns):
시도이름2              2555 non-null object
일시                   2555 non-null object
0.5m 지중온도(℃)       1456 non-null float64
1.0m 지중온도(℃)       1460 non-null float64
```

```
df_max.shape
```

```
(2555, 63)
```

(365일 * 7개 광역시) =2555 행 관측치

결측값 대체하기 df.fillna(값, inplace=True)

- 광역시 기후 데이터가 없는 경우 NaN가 저장되어 있음
- 기후 데이터의 경우 결측치는 0으로 대체하여 사용하면 된다.

```
df_max.iloc[:,[0,1,24,28]]
```

	시도이름2	일시	일강수량(mm)	최고기온(°C)
0	광주	2018-01-01	NaN	7.1
1	광주	2018-01-02	NaN	8.4
2	광주	2018-01-03	NaN	3.8
3	광주	2018-01-04	NaN	2.1
4	광주	2018-01-05	NaN	5.2
5	광주	2018-01-06	NaN	7.2

```
df_max.fillna(0, inplace=True)
df_max.iloc[:,[0,1,24,28]]
```

	시도이름2	일시	일강수량(mm)	최고기온(°C)
0	광주	2018-01-01	0.0	7.1
1	광주	2018-01-02	0.0	8.4
2	광주	2018-01-03	0.0	3.8

날짜 데이터 만들기, 활용 - pd.to_datetime(문자열변수. object)

- 만약 문자열변수가 아니고 int인 경우 **df['int변수'].astype(str)**

```
df_max.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2555 entries, 0 to 2554
Data columns (total 63 columns):
시도이름2                2555 non-null object
일시                   2555 non-null object
```

```
df_max['일시']=pd.to_datetime(df_max['일시'])
```

```
df_max.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2555 entries, 0 to 2554
Data columns (total 63 columns):
시도이름2                2555 non-null object
일시                   2555 non-null datetime64[ns]
0.5m 지중온도(°C)        2555 non-null float64
```

- DFSeries.dt.옵션 : weekday 0=월, 6=일요일 숫자임

```
df_max['월']=df_max['일시'].dt.month
df_max['요일']=df_max['일시'].dt.weekday
```

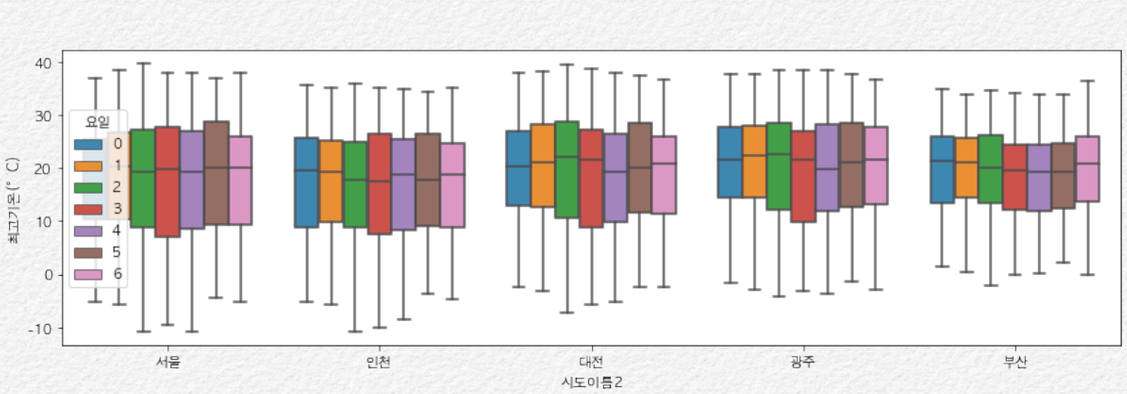
```
df_max[['일시','월', '요일']].head(3)
```

	일시	월	요일
0	2018-01-01	1	0
1	2018-01-02	1	1
2	2018-01-03	1	2

```python
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import rc
import seaborn as sns
%matplotlib inline

rc('font', family='AppleGothic')
plt.rcParams['axes.unicode_minus'] = False

plt.rcParams["figure.figsize"] = (14,4) #set size of Graph
ax = sns.boxplot(x="시도이름2", y="최고기온(°C)", hue="요일",
                 data=df_max, order=['서울', '인천', '대전', '광주', '부산'])
```




```
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import rc
import seaborn as sns
%matplotlib inline

rc('font', family='AppleGothic')
plt.rcParams['axes.unicode_minus'] = False

plt.rcParams["figure.figsize"] = (14,4) #set size of Graph
ax = sns.boxplot(x="시도이름2", y="최고기온(°C)", hue="요일",
                 data=df_max, order=['서울', '인천', '대전', '광주', '부산'])
```