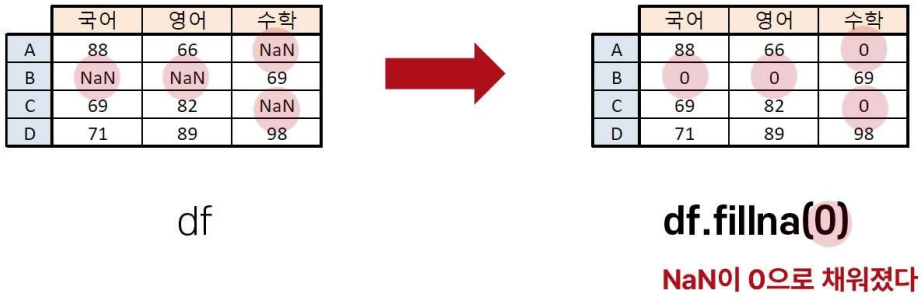
Ch08. OECD 국가의 GDP데이터로 실무의 데이터 다루기

1. 결측값 채우기 (fillna)

> pandas fillna

결측값 null(대표적으로 NaN)을 지정한 값으로 채우는 함수



value (scalar, 딕셔너리, 시리즈, 데이터 프레임)

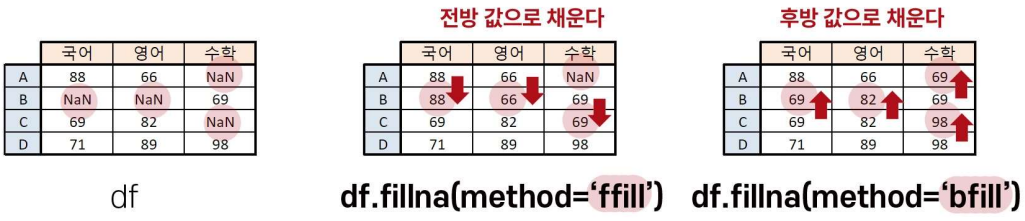
NaN을 채울 값

- scalar를 입력하면 동일한 값으로 채운다
- 딕셔너리나 시리즈를 입력하면 key에 맞는 열마다 다른 값을 채운다
- 데이터 프레임은 인덱스가 동일한 값을 채운다

method (인수는 'backfill', 'bfill', 'pad', 'ffill', None / 기본값은 None)

근접한 값으로 NaN을 채울 때 사용하는 인자

- ffill / pad : 이전 값으로 채운다
- bfill / backfill: 이후의 값으로 채운다



axis (인수는 0 또는 1 / 기본값은 0)

method로 근접한 값을 채울 때 축을 지정하는 인자, 0이면 열에서 근접한 값을 채운다

fillna 함수 설명 : https://kimpanda.tistory.com/100

fillna 공식문서 : https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.fillna.html

```
# 실습 준비 코드
import pandas as pd
data = [[88, 66, None], [None, None, 69], [69, 82, None], [71, 89, 98]]
data1 = [[0, 1, 2] , [3, 4, 5], [6, 7, 8], [9, 10, 11]]
df = pd.DataFrame(data, index=list('ABCD'), columns=['국어', '영어', '수학'])
df1 = pd.DataFrame(data1, index=df.index, columns=df.columns)
df
```

	국어	영어	수학
A	88.0	66.0	NaN
B	NaN	NaN	69.0
C	69.0	82.0	NaN
D	71.0	89.0	98.0

```
# 값으로 채우기 (scalar로 채우기)
df.fillna(0)
```

	국어	영어	수학
A	88.0	66.0	0.0
B	0.0	0.0	69.0
C	69.0	82.0	0.0
D	71.0	89.0	98.0

```
# 맵퍼로 채우기 (열마다 다른 값으로 채운다)
df.fillna({'국어': 0, '영어': 1, '수학': 2})
```

	국어	영어	수학
A	88.0	66.0	2.0
B	0.0	1.0	69.0
C	69.0	82.0	2.0
D	71.0	89.0	98.0

```
# 모든 열을 key에 지정할 필요는 없다
df.fillna({'국어': 0, '영어': 1})
```

	국어	영어	수학
A	88.0	66.0	NaN
B	0.0	1.0	69.0

```
# 열의 평균으로 채우기
df.fillna(df.mean())
```

	국어	영어	수학
A	88.0	66.0	83.5
B	76.0	79.0	69.0
C	69.0	82.0	83.5
D	71.0	89.0	98.0

```
# 행의 평균으로 채우는 것은 기본으로는 불가능하니 lambda 함수를 만들자
x = df.loc['A']
x.fillna(x.mean())
```

```
국어    88.0
영어    66.0
수학    77.0
Name: A, dtype: float64
```

```
# 시리즈의 평균값으로 시리즈를 채우는 lambda 함수를 apply로 행별(axis=1)로 적용
df.apply(lambda x: x.fillna(x.mean()), axis=1)
```

	국어	영어	수학
A	88.0	66.0	77.0
B	69.0	69.0	69.0
C	69.0	82.0	75.5
D	71.0	89.0	98.0

```
# 열의 전방값으로 채우기
df.fillna(method='ffill')
```

	국어	영어	수학
A	88.0	66.0	NaN
B	88.0	66.0	69.0
C	69.0	82.0	69.0
D	71.0	89.0	98.0

```
# 열의 전방값으로 채우는 함수 ffill
df.ffill()
```

	국어	영어	수학
A	88.0	66.0	NaN
B	88.0	66.0	69.0
C	69.0	82.0	69.0
D	71.0	89.0	98.0

```
# 열의 후방값으로 채우기
df.fillna(method='bfill')
```

	국어	영어	수학
A	88.0	66.0	69.0
B	69.0	82.0	69.0
C	69.0	82.0	98.0
D	71.0	89.0	98.0

```
# 열의 후방값으로 채우는 함수 bfill
df.bfill()
```

	국어	영어	수학
A	88.0	66.0	69.0
B	69.0	82.0	69.0
C	69.0	82.0	98.0
D	71.0	89.0	98.0

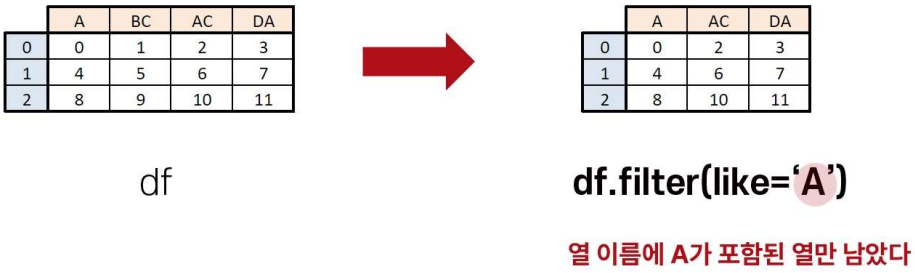
```
# 행의 전방값으로 채우기 (axis=1)
df.fillna(method='ffill', axis=1)
```

	국어	영어	수학
A	88.0	66.0	66.0
B	NaN	NaN	69.0
C	69.0	82.0	82.0
D	71.0	89.0	98.0

2. 열을 레이블로 필터링하기(filter)

> pandas filter

인덱스의 레이블로 데이터프레임이나 시리즈를 필터링하는 함수



items

필터링때 정확하게 일치하는 문자열을 지정하는 인자

like

필터링때 포함하는 문자열을 지정하는 인자

regex (regular expression)

필터링때 정규 표현식을 지정하는 인자

axis (0 or 1 / 기본값은 1)

index를 필터링하는지 columns를 필터링하는지 결정하는 인자

filter 함수 설명 : https://kimpanda.tistory.com/101

filter 판다스 공식문서 : https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.filter.html

```
# 실습 준비 코드
import pandas as pd
data = [[0, 1, 2, 3], [4, 5, 6, 7], [8, 9, 10, 11]]
df = pd.DataFrame(data, columns=['A', 'BC', 'AC', 'DA'])
df
```

	A	BC	AC	DA
0	0	1	2	3
1	4	5	6	7
2	8	9	10	11

```
# 레이블에 A가 포함되는 열 필터링
df.filter(like='A')
```

	A	AC	DA
0	0	2	3
1	4	6	7
2	8	10	11

```
# 레이블이 A로 시작되는 열 필터링
df.filter(regex=r'^A')
```

	A	AC
0	0	2
1	4	6
2	8	10

```
# 레이블이 B 또는 C를 포함하는 열 필터링
df.filter(regex=r'B|C')
```

	BC	AC
0	1	2
1	5	6
2	9	10

정규표현식을 배우지 않은 사람들도

- ^ : 시작
- $: 끝
- | : 또는

간단한 위 세가지만 활용해도 크게 도움이 된다

3. 실무 데이터의 문제점

OECD 국가 주요 지표 : https://kosis.kr/statHtml/statHtml.do?orgId=101&tblId=DT_2KAAG01

문제점

1. columns가 멀티인덱스인데도 열이 너무 많다. 이런 피벗테이블은 한눈에 알아볼 수 없다
2. 국가의 데이터에 대륙을 구분하기 위한 행이 존재한다

국가별	2019	
	국내총생산(GDP) (당해년 가격) (10억US$)	1인당 GDP(당해년 가격) (달러)
세계	-	-
아시아		
대한민국	1,651.0	31,929
이스라엘	394.7	43,589
일본	5,064.9	40,113
튀르키예	761.4	9,127
북아메리카		
캐나다	1,741.6	46,327
멕시코	1,268.9	9,946
미국	21,433.2	65,280
남아메리카		
칠레	279.4	14,742

대륙을 구분하기 위한 불필요한 행

→ 피벗테이블로서의 기능도 못하고 데이터를 집계할수 있도록 관리하는 기능도 못하고 있다

raw 데이터 처럼 집계를 할 수 있는 데이터로 만들어 필요한 요약은 피벗테이블로 집계해 제공할수 있으면 문제가 해결된다

4. OECD의 GDP 데이터 전처리하기

통계청 데이터는 전처리해야 될것이 지나치게 많아 약간의 전처리를 한 데이터를 사용하겠습니다

사용할 csv 파일 링크 : https://raw.githubusercontent.com/panda-kim/csv_files/main/OECD_GDP.csv

```
# 프로젝트 코드
import pandas as pd
pd.options.display.max_rows = 6 # 판다스 버전업에 따라 6행 출력의 바뀐 코드
url = 'https://raw.githubusercontent.com/panda-kim/csv_files/main/OECD_GDP.csv'
```

1. **데이터 소실 없이 연도 데이터를 열로 만들기 위해 csv파일에서 columns를 멀티인덱스로 index로는 국가열을 지정해서 데이터 프레임을 부른다**

	2019					2020				
국가	1인당 GDP	GDP 성장률	GDP	수입	수출	1인당 GDP	GDP 성장률	GDP	수입	수출
아시아	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
대한민국	31929	2.2	1651	503343	542233	31637	-0.9	1638.2	467633	512498
...	...	...	...	...	...	...	...	...	...	...
북아메리카	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
캐나다	46327	1.9	1741.6	462392	450805	43242	-5.4	1643.4	419688	392129
...	...	...	...	...	...	...	...	...	...	...
남아메리카	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...
....	...	...	...	...	...	...	...	...	...	...

```
df_ex1 = pd.read_csv(url, header=[0, 1], index_col=0)
df_ex1
```

	2019					2020				
	1인당 GDP	GDP 성장률	GDP	수입	수출	1인당 GDP	GDP 성장률	GDP	수입	
국가										
아시아	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
대한민국	31929.0	2.2	1651.0	503343	542233.0	31637.0	-0.9	1638.2	467633	
이스라엘	43589.0	3.5	394.7	75697	51899.0	43611.0	-2.4	402.0	69810	
...	...	...	...	...	...	...	...	...	...	
오세아니아	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
오스트레일리아	55057.0	2.2	1396.6	154043	271006.0	51812.0	-0.3	1330.9	146522	
뉴질랜드	41999.0	1.6	209.1	42362	39516.0	41792.0	1.0	212.5	37135	

43 rows × 10 columns

2. GDP에 관련되지 않은 열은 삭제하고
columns였던 연도 데이터와 index였던 국가 데이터를 모두 열로 만든다.

	국가	연도	1인당 GDP	GDP	GDP 성장률
0	아시아	2019	NaN	NaN	NaN
1	아시아	2020	NaN	NaN	NaN
2	대한민국	2019	31929	1651	2.2
3	대한민국	2020	31637	1638.2	-0.9
...	...	...	...	...	...
10	북아메리카	2019	NaN	NaN	NaN
11	북아메리카	2020	NaN	NaN	NaN
12	캐나다	2019	46327	1741.6	1.9
13	캐나다	2020	43242	1643.4	-5.4
...	...	...	...	...	...
18	남아메리카	2019	NaN	NaN	NaN
19	남아메리카	2020	NaN	NaN	NaN
...	...	...	...	...	...

```
# 필요한 GDP열만 남긴 뒤 연도와 국가를 열로 만들고 열 이름을 수정
df_ex2 = (df_ex1
          .stack(0, dropna=False)
          .filter(like='GDP')
          .reset_index()
          .rename(columns={'level_1':'연도'}))
df_ex2
```

	국가	연도	1인당 GDP	GDP	GDP 성장률
0	아시아	2019	NaN	NaN	NaN
1	아시아	2020	NaN	NaN	NaN
2	대한민국	2019	31929.0	1651.0	2.2
...	...	...	...	...	...
83	오스트레일리아	2020	51812.0	1330.9	-0.3

3. 국가 열에 불필요한 행으로 구분되던 대륙을 열로 만들고 불필요한 행은 삭제한다
그러면 raw data 처럼 집계가 가능한 데이터가 만들어진다. 전처리 완성!

	국가	연도	1인당 GDP	GDP	GDP 성장률	구분
2	대한민국	2019	31929	1651	2.2	아시아
3	대한민국	2020	31637	1638.2	-0.9	아시아
...	...	...	...	...	...	...
12	캐나다	2019	46327	1741.6	1.9	북아메리카
13	캐나다	2020	43242	1643.4	-5.4	북아메리카
...	...	...	...	...	...	...

```
# 대륙구분을 행에서 하는 것이 아니고 구분 열을 만들고 대륙 구분 행 삭제
cond1 = df_ex2['GDP'].isnull()
df_ex2['구분'] = df_ex2['국가'].mask(~cond1).fillna(method='ffill')
df_ex3 = df_ex2[~cond1].reset_index(drop=True)
df_ex3
```

	국가	연도	1인당 GDP	GDP	GDP 성장률	구분
0	대한민국	2019	31929.0	1651.0	2.2	아시아
1	대한민국	2020	31637.0	1638.2	-0.9	아시아
2	이스라엘	2019	43589.0	394.7	3.5	아시아
...	...	...	...	...	...	...
73	오스트레일리아	2020	51812.0	1330.9	-0.3	오세아니아
74	뉴질랜드	2019	41999.0	209.1	1.6	오세아니아
75	뉴질랜드	2020	41792.0	212.5	1.0	오세아니아

76 rows × 6 columns

df_ex3는 집계가 가능한 raw data와 마찬가지이므로 다양한 집계를 해보자

```
# 연도별 대륙별 GDP 성장률의 최대 피벗 테이블
df_ex3.pivot_table('GDP 성장률', index='구분', columns='연도',
                   aggfunc='max')
```

연도 구분	2019	2020
남아메리카	3.3	-4.5
북아메리카	2.2	-3.5
아시아	3.5	1.8
오세아니아	2.2	1.0
유럽	5.6	3.4

```
# 연도별 대륙별 GDP 성장률의 최대인 국가 피벗 테이블
df_ex3.pivot_table('GDP 성장률', index='구분', columns='연도',
                   aggfunc=lambda x: df_ex3.loc[x.idxmax(), '국가'])
```

연도 구분	2019	2020
남아메리카	콜롬비아	코스타리카
북아메리카	미국	미국
아시아	이스라엘	튀르키예
오세아니아	오스트레일리아	뉴질랜드
유럽	아일랜드	아일랜드

사용된 함수들의 설명이 필요하다면 링크를 참조

[함수 설명 블로그]