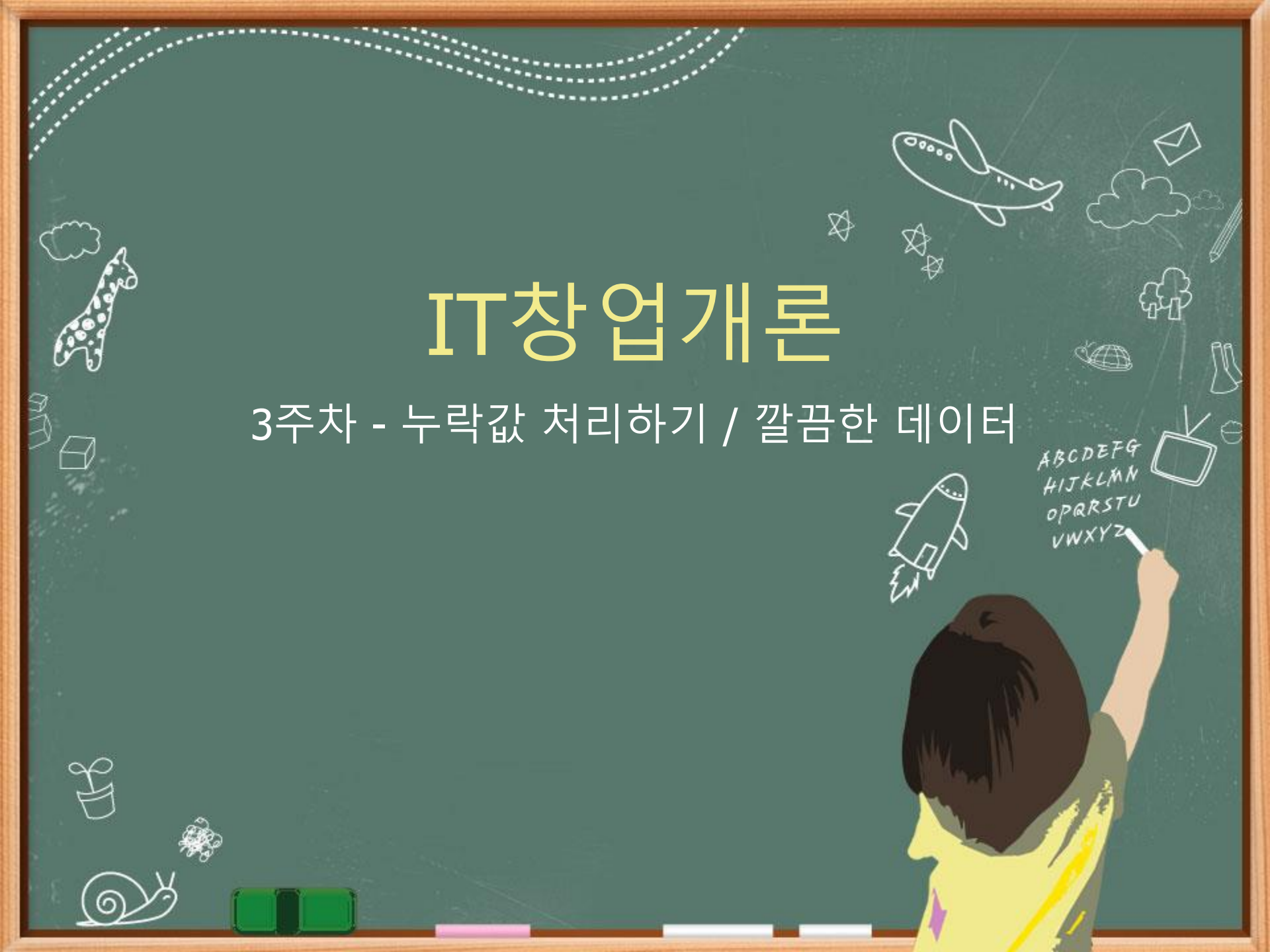
IT창업개론

3주차 - 누락값 처리하기 / 깔끔한 데이터



누락값이란?

- NaN, NAN, nan으로 표기
- numpy 라이브러리
  - 수학이나 과학 연산을 위해 만든 파이썬 라이브러리로 누락값을 사용하기 위해 필요
- 데이터 자체가 없다는 것을 의미
  - '같다' 라는 개념도 없음
  - 자기 자신과 비교해도 False

```
from numpy import NaN, NAN, nan
```

```
NaN == True
```
False

```
NaN == False
```
False

```
NaN == 0
```
False

```
NaN == ''
```
False

```
NaN == NaN
```
False

```
NaN == nan
```
False

```
NaN == NAN
```
False

```
nan == NAN
```
False

누락값이란?

- 누락값 확인 방법

```
import pandas as pd
pd.isnull(NaN)
```

```
True
```

```
pd.isnull(nan)
```

```
True
```

```
pd.isnull(NAN)
```

```
True
```

```
pd.notnull(NaN)
```

```
False
```

```
pd.notnull(42)
```

```
True
```

```
pd.notnull('missing')
```

```
True
```

누락값이 생기는 이유

- 누락값이 있는 데이터 집합을 연결할 때

```
visited = pd.read_csv('../data/survey_visited.csv')
visited
```

	ident	site	dated
0	619	DR-1	1927-02-08
1	622	DR-1	1927-02-10
2	734	DR-3	1939-01-07
3	735	DR-3	1930-01-12
4	751	DR-3	1930-02-26
5	752	DR-3	NaN
6	837	MSK-4	1932-01-14
7	844	DR-1	1932-03-22

```
survey = pd.read_csv('../data/survey_survey.csv')
survey
```

	taken	person	quant	reading
0	619	dyer	rad	9.82
1	619	dyer	sal	0.13
2	622	dyer	rad	7.80
3	622	dyer	sal	0.09
4	734	pb	rad	8.41
5	734	lake	sal	0.05
6	734	pb	temp	-21.50
7	735	pb	rad	7.22
8	735	NaN	sal	0.06
9	735	NaN	temp	-26.00
10	751	pb	rad	4.35
11	751	pb	temp	-18.50
12	751	lake	sal	0.10
13	752	lake	rad	2.19
14	752	lake	sal	0.09
15	752	lake	temp	-16.00
16	752	roe	sal	41.60
17	837	lake	rad	1.46
18	837	lake	sal	0.21
19	837	roe	sal	22.50
20	844	roe	rad	11.25

```
visited.merge(survey, left_on='ident', right_on='taken')
```

	ident	site	dated	taken	person	quant	reading
0	619	DR-1	1927-02-08	619	dyer	rad	9.82
1	619	DR-1	1927-02-08	619	dyer	sal	0.13
2	622	DR-1	1927-02-10	622	dyer	rad	7.80
3	622	DR-1	1927-02-10	622	dyer	sal	0.09
4	734	DR-3	1939-01-07	734	pb	rad	8.41
5	734	DR-3	1939-01-07	734	lake	sal	0.05
6	734	DR-3	1939-01-07	734	pb	temp	-21.50
7	735	DR-3	1930-01-12	735	pb	rad	7.22
8	735	DR-3	1930-01-12	735	NaN	sal	0.06
9	735	DR-3	1930-01-12	735	NaN	temp	-26.00
10	751	DR-3	1930-02-26	751	pb	rad	4.35
11	751	DR-3	1930-02-26	751	pb	temp	-18.50
12	751	DR-3	1930-02-26	751	lake	sal	0.10
13	752	DR-3	NaN	752	lake	rad	2.19
14	752	DR-3	NaN	752	lake	sal	0.09
15	752	DR-3	NaN	752	lake	temp	-16.00
16	752	DR-3	NaN	752	roe	sal	41.60
17	837	MSK-4	1932-01-14	837	lake	rad	1.46
18	837	MSK-4	1932-01-14	837	lake	sal	0.21
19	837	MSK-4	1932-01-14	837	roe	sal	22.50
20	844	DR-1	1932-03-22	844	roe	rad	11.25

```
result[pd.isnull(result['dated'])]
```

	ident	site	dated	taken	person	quant	reading
13	752	DR-3	NaN	752	lake	rad	2.19
14	752	DR-3	NaN	752	lake	sal	0.09
15	752	DR-3	NaN	752	lake	temp	-16.00
16	752	DR-3	NaN	752	roe	sal	41.60

```
result[pd.notnull(result['dated'])]
```

	ident	site	dated	taken	person	quant	reading
0	619	DR-1	1927-02-08	619	dyer	rad	9.82
1	619	DR-1	1927-02-08	619	dyer	sal	0.13
2	622	DR-1	1927-02-10	622	dyer	rad	7.80
3	622	DR-1	1927-02-10	622	dyer	sal	0.09
4	734	DR-3	1939-01-07	734	pb	rad	8.41
5	734	DR-3	1939-01-07	734	lake	sal	0.05
6	734	DR-3	1939-01-07	734	pb	temp	-21.50
7	735	DR-3	1930-01-12	735	pb	rad	7.22
8	735	DR-3	1930-01-12	735	NaN	sal	0.06
9	735	DR-3	1930-01-12	735	NaN	temp	-26.00
10	751	DR-3	1930-02-26	751	pb	rad	4.35
11	751	DR-3	1930-02-26	751	pb	temp	-18.50
12	751	DR-3	1930-02-26	751	lake	sal	0.10
17	837	MSK-4	1932-01-14	837	lake	rad	1.46
18	837	MSK-4	1932-01-14	837	lake	sal	0.21
19	837	MSK-4	1932-01-14	837	roe	sal	22.50
20	844	DR-1	1932-03-22	844	roe	rad	11.25

누락값이 생기는 이유

- 범위를 지정하여 데이터를 추출할 때
  - 데이터프레임에 존재하지 않는 데이터를 추출하면 누락값이 생김

```
gapminder = pd.read_csv('../data/gapminder.tsv', sep='₩t')
```

```
life_exp = gapminder.groupby(['year'])['lifeExp'].mean()
life_exp
```

```
<class 'pandas.core.series.Series'>

year
1952    49.057620
1957    51.507401
1962    53.609249
1967    55.678290
1972    57.647386
1977    59.570157
1982    61.533197
1987    63.212613
1992    64.160338
1997    65.014676
2002    65.694923
2007    67.007423
Name: lifeExp, dtype: float64
```

```
life_exp.loc[range(2000, 2010),]
```

```
year
2000          NaN
2001          NaN
2002    65.694923
2003          NaN
2004          NaN
2005          NaN
2006          NaN
2007    67.007423
2008          NaN
2009          NaN
```

```
life_exp.loc[2000:2010]
```

```
year
2002    65.694923
2007    67.007423
```

누락값의 개수

count 함수

- 누락값이 아닌 값의 개수를 구할 수 있음

```
ebola = pd.read_csv('../data/country_timeseries.csv')
ebola
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone	Cases_Nigeria
0	1/5/2015	289	2776.0	NaN	10030.0	NaN
1	1/4/2015	288	2775.0	NaN	9780.0	NaN
2	1/3/2015	287	2769.0	8166.0	9722.0	NaN
3	1/2/2015	286	NaN	8157.0	NaN	NaN
4	12/31/2014	284	2730.0	8115.0	9633.0	NaN
...	...	...	...	...	...	...
117	3/27/2014	5	103.0	8.0	6.0	NaN
118	3/26/2014	4	86.0	NaN	NaN	NaN
119	3/25/2014	3	86.0	NaN	NaN	NaN
120	3/24/2014	2	86.0	NaN	NaN	NaN
121	3/22/2014	0	49.0	NaN	NaN	NaN

122 rows × 18 columns

```
ebola.count()
```

```
Date                 122
Day                  122
Cases_Guinea          93
Cases_Liberia         83
Cases_SierraLeone     87
Cases_Nigeria         38
Cases_Senegal         25
Cases_UnitedStates    18
Cases_Spain           16
Cases_Mali            12
Deaths_Guinea         92
Deaths_Liberia        81
Deaths_SierraLeone    87
Deaths_Nigeria        38
Deaths_Senegal        22
Deaths_UnitedStates   18
Deaths_Spain          16
Deaths_Mali           12
```

누락값의 개수

```
num_rows = ebola.shape[0]
num_rows - ebola.count()
```

```
Date                    0
Day                     0
Cases_Guinea           29
Cases_Liberia          39
Cases_SierraLeone      35
Cases_Nigeria          84
Cases_Senegal          97
Cases_UnitedStates    104
Cases_Spain           106
Cases_Mali            110
Deaths_Guinea          30
Deaths_Liberia         41
Deaths_SierraLeone     35
Deaths_Nigeria         84
Deaths_Senegal        100
Deaths_UnitedStates   104
Deaths_Spain          106
Deaths_Mali           110
```

```
ebola.isnull()
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone	Cases_Nigeria
0	False	False	False	True	False	True
1	False	False	False	True	False	True
2	False	False	False	False	False	True
3	False	False	True	False	True	True
4	False	False	False	False	False	True
...	...	...	...	...	...	...
117	False	False	False	False	False	True
118	False	False	False	True	True	True
119	False	False	False	True	True	True
120	False	False	False	True	True	True
121	False	False	False	True	True	True

122 rows × 18 columns

누락값의 개수

- count_nonzero()
  - 배열에서 0이 아닌 값의 개수를 세는 함수
- value_counts()
  - 지정한 열의 값의 빈도를 구하는 함수

```
import numpy as np
np.count_nonzero(ebola.isnull())
```

```
1214
```

```
np.count_nonzero(ebola['Cases_Guinea'].isnull())
```

```
29
```

```
ebola.Cases_Guinea.value_counts(dropna=False)
```

```
NaN       29
86.0       3
495.0      2
112.0      2
390.0      2
          ..
235.0      1
231.0      1
226.0      1
224.0      1
2776.0     1
```

누락값 변경하기

- 누락값은 누락값을 임의의 값으로 변경하거나 데이터프레임에 이미 있는 값으로 대신 채우는 방법 등으로 처리할 수 있음
- fillna()
  - 누락값 변경 함수

```
ebola.iloc[0:10, 0:5]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone
0	1/5/2015	289	2776.0	NaN	10030.0
1	1/4/2015	288	2775.0	NaN	9780.0
2	1/3/2015	287	2769.0	8166.0	9722.0
3	1/2/2015	286	NaN	8157.0	NaN
4	12/31/2014	284	2730.0	8115.0	9633.0
5	12/28/2014	281	2706.0	8018.0	9446.0
6	12/27/2014	280	2695.0	NaN	9409.0
7	12/24/2014	277	2630.0	7977.0	9203.0
8	12/21/2014	273	2597.0	NaN	9004.0
9	12/20/2014	272	2571.0	7862.0	8939.0

```
ebola.fillna(0).iloc[0:10, 0:5]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone
0	1/5/2015	289	2776.0	0.0	10030.0
1	1/4/2015	288	2775.0	0.0	9780.0
2	1/3/2015	287	2769.0	8166.0	9722.0
3	1/2/2015	286	0.0	8157.0	0.0
4	12/31/2014	284	2730.0	8115.0	9633.0
5	12/28/2014	281	2706.0	8018.0	9446.0
6	12/27/2014	280	2695.0	0.0	9409.0
7	12/24/2014	277	2630.0	7977.0	9203.0
8	12/21/2014	273	2597.0	0.0	9004.0
9	12/20/2014	272	2571.0	7862.0	8939.0

누락값 변경하기

- fillna()
  - method 인자값을 **ffill**로 지정하면 누락값이 나타나기 전의 값으로 변경

```
ebola.iloc[0:10, 0:5]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone
0	1/5/2015	289	2776.0	NaN	10030.0
1	1/4/2015	288	2775.0	NaN	9780.0
2	1/3/2015	287	2769.0	8166.0	9722.0
3	1/2/2015	286	NaN	8157.0	NaN
4	12/31/2014	284	2730.0	8115.0	9633.0
5	12/28/2014	281	2706.0	8018.0	9446.0
6	12/27/2014	280	2695.0	NaN	9409.0
7	12/24/2014	277	2630.0	7977.0	9203.0
8	12/21/2014	273	2597.0	NaN	9004.0
9	12/20/2014	272	2571.0	7862.0	8939.0

```
ebola.fillna(method='ffill').iloc[0:10, 0:5]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone
0	1/5/2015	289	2776.0	NaN	10030.0
1	1/4/2015	288	2775.0	NaN	9780.0
2	1/3/2015	287	2769.0	8166.0	9722.0
3	1/2/2015	286	2769.0	8157.0	9722.0
4	12/31/2014	284	2730.0	8115.0	9633.0
5	12/28/2014	281	2706.0	8018.0	9446.0
6	12/27/2014	280	2695.0	8018.0	9409.0
7	12/24/2014	277	2630.0	7977.0	9203.0
8	12/21/2014	273	2597.0	7977.0	9004.0
9	12/20/2014	272	2571.0	7862.0	8939.0

누락값 변경하기

- fillna()
  - method 인자값을 **bfill**로 지정하면 누락값이 나타난 후의 첫 번째 값으로 앞쪽의 누락값이 모두 변경

```
ebola.iloc[0:10, 0:5]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone
0	1/5/2015	289	2776.0	NaN	10030.0
1	1/4/2015	288	2775.0	NaN	9780.0
2	1/3/2015	287	2769.0	8166.0	9722.0
3	1/2/2015	286	NaN	8157.0	NaN
4	12/31/2014	284	2730.0	8115.0	9633.0
5	12/28/2014	281	2706.0	8018.0	9446.0
6	12/27/2014	280	2695.0	NaN	9409.0
7	12/24/2014	277	2630.0	7977.0	9203.0
8	12/21/2014	273	2597.0	NaN	9004.0
9	12/20/2014	272	2571.0	7862.0	8939.0

```
ebola.fillna(method='bfill').iloc[0:10, 0:5]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone
0	1/5/2015	289	2776.0	8166.0	10030.0
1	1/4/2015	288	2775.0	8166.0	9780.0
2	1/3/2015	287	2769.0	8166.0	9722.0
3	1/2/2015	286	2730.0	8157.0	9633.0
4	12/31/2014	284	2730.0	8115.0	9633.0
5	12/28/2014	281	2706.0	8018.0	9446.0
6	12/27/2014	280	2695.0	7977.0	9409.0
7	12/24/2014	277	2630.0	7977.0	9203.0
8	12/21/2014	273	2597.0	7862.0	9004.0
9	12/20/2014	272	2571.0	7862.0	8939.0

누락값 변경하기

- interpolate()
  - 누락값 양쪽에 있는 값을 이용하여 중간 값을 구한 다음 누락값 처리

```
ebola.iloc[0:10, 0:5]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone
0	1/5/2015	289	2776.0	NaN	10030.0
1	1/4/2015	288	2775.0	NaN	9780.0
2	1/3/2015	287	2769.0	8166.0	9722.0
3	1/2/2015	286	NaN	8157.0	NaN
4	12/31/2014	284	2730.0	8115.0	9633.0
5	12/28/2014	281	2706.0	8018.0	9446.0
6	12/27/2014	280	2695.0	NaN	9409.0
7	12/24/2014	277	2630.0	7977.0	9203.0
8	12/21/2014	273	2597.0	NaN	9004.0
9	12/20/2014	272	2571.0	7862.0	8939.0

```
ebola.interpolate().iloc[0:10, 0:5]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone
0	1/5/2015	289	2776.0	NaN	10030.0
1	1/4/2015	288	2775.0	NaN	9780.0
2	1/3/2015	287	2769.0	8166.0	9722.0
3	1/2/2015	286	2749.5	8157.0	9677.5
4	12/31/2014	284	2730.0	8115.0	9633.0
5	12/28/2014	281	2706.0	8018.0	9446.0
6	12/27/2014	280	2695.0	7997.5	9409.0
7	12/24/2014	277	2630.0	7977.0	9203.0
8	12/21/2014	273	2597.0	7919.5	9004.0
9	12/20/2014	272	2571.0	7862.0	8939.0

누락값 삭제하기

- 누락값이 필요 없을 경우에는 삭제해도 되지만 데이터가 너무 편향되거나 데이터의 개수가 너무 적어질 수 있음

```
ebola.shape
```

```
(122, 18)
```

```
ebola_dropna = ebola.dropna()
ebola_dropna.shape
```

```
(1, 18)
```

```
ebola_dropna.iloc[:, 0:5]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone
19	11/18/2014	241	2047.0	7082.0	6190.0

누락값이 포함된 데이터 계산하기

- ebola 발병 수를 구한다면?

```
ebola['Cases_multiple'] = ebola['Cases_Guinea'] + ebola['Cases_Liberia'] + ebola['Cases_SierraLeone']
```

```
ebola_subset = ebola.loc[:, ['Cases_Guinea', 'Cases_Liberia', 'Cases_SierraLeone', 'Cases_multiple']]
ebola_subset.head(n=10)
```

	Cases_Guinea	Cases_Liberia	Cases_SierraLeone	Cases_multiple
0	2776.0	NaN	10030.0	NaN
1	2775.0	NaN	9780.0	NaN
2	2769.0	8166.0	9722.0	20657.0
3	NaN	8157.0	NaN	NaN
4	2730.0	8115.0	9633.0	20478.0
5	2706.0	8018.0	9446.0	20170.0
6	2695.0	NaN	9409.0	NaN
7	2630.0	7977.0	9203.0	19810.0
8	2597.0	NaN	9004.0	NaN
9	2571.0	7862.0	8939.0	19372.0

```
ebola.Cases_Guinea.sum(skipna = True)
```

```
84729.0
```

```
ebola.Cases_Guinea.sum(skipna = False)
```

```
nan
```

누락값이 포함된 데이터 계산하기

- ebola 발병 수를 구한다면?

```
ebola['Cases_multiple'] = ebola['Cases_Guinea'].fillna(0) + ebola['Cases_Liberia'].fillna(0) + ebola['Cases_SierraLeone'].fillna(0)
```

```
ebola_subset = ebola.loc[:, ['Cases_Guinea', 'Cases_Liberia', 'Cases_SierraLeone', 'Cases_multiple']]
ebola_subset.head(n=10)
```

	Cases_Guinea	Cases_Liberia	Cases_SierraLeone	Cases_multiple
0	2776.0	NaN	10030.0	12806.0
1	2775.0	NaN	9780.0	12555.0
2	2769.0	8166.0	9722.0	20657.0
3	NaN	8157.0	NaN	8157.0
4	2730.0	8115.0	9633.0	20478.0
5	2706.0	8018.0	9446.0	20170.0
6	2695.0	NaN	9409.0	12104.0
7	2630.0	7977.0	9203.0	19810.0
8	2597.0	NaN	9004.0	11601.0
9	2571.0	7862.0	8939.0	19372.0

열과 피벗

넓은 데이터

- 데이터프레임의 열은 열 자체가 어떤 값을 의미하므로 데이터프레임의 열이 길게 늘어선 형태의 데이터
  - 데이터의 열 이름이 어떤 값을 의미하면 열의 폭이 넓은 경우가 많음

'미국의 소득과 종교' 데이터, 퓨 리서치 센터 (Pew Research Center)

```
import pandas as pd
pew = pd.read_csv('../data/pew.csv')
pew.head()
```

	religion	<$10k	$10-20k	$20-30k	$30-40k	$40-50k	$50-75k	$75-100k	$100-150k	>150k	Don't know/refused
0	Agnostic	27	34	60	81	76	137	122	109	84	96
1	Atheist	12	27	37	52	35	70	73	59	74	76
2	Buddhist	27	21	30	34	33	58	62	39	53	54
3	Catholic	418	617	732	670	638	1116	949	792	633	1489
4	Don't know/refused	15	14	15	11	10	35	21	17	18	116

열과 피벗

- 6개의 열만 출력한 결과
  - 종교와 소득 정보가 출력됨
- 소득 정보 열을 행 데이터로 옮기고 싶다면?

```
pew.iloc[:, 0:6]
```

	religion	<$10k	$10-20k	$20-30k	$30-40k	$40-50k
0	Agnostic	27	34	60	81	76
1	Atheist	12	27	37	52	35
2	Buddhist	27	21	30	34	33
3	Catholic	418	617	732	670	638
4	Don't know/refused	15	14	15	11	10
5	Evangelical Prot	575	869	1064	982	881
6	Hindu	1	9	7	9	11
7	Historically Black Prot	228	244	236	238	197
8	Jehovah's Witness	20	27	24	24	21
9	Jewish	19	19	25	25	30
10	Mainline Prot	289	495	619	655	651
11	Mormon	29	40	48	51	56
12	Muslim	6	7	9	10	9
13	Orthodox	13	17	23	32	32
14	Other Christian	9	7	11	13	13
15	Other Faiths	20	33	40	46	49
16	Other World Religions	5	2	3	4	2
17	Unaffiliated	217	299	374	365	341

열과 피벗

- melt()
  - 지정한 열의 데이터를 모두 행으로 정리해 주는 함수

arguments	description
id_vars	위치를 그대로 유지할 열의 이름 지정
value_vars	행으로 위치를 변경할 열의 이름 지정
var_name	위치를 변경한 열의 이름 지정
value_name	위치를 변경한 열의 데이터를 저장한 열의 이름 지정

열과 피벗

- 1개의 열만 고정하고 나머지 열을 행으로 바꾸기

'religion 열을 고정하여 피벗(pivot)했다' 라고 표현

```
import pandas as pd
pew = pd.read_csv('../data/pew.csv')
```

```
pd.melt(pew, id_vars='religion').head()
```

	religion	variable	value
0	Agnostic	<$10k	27
1	Atheist	<$10k	12
2	Buddhist	<$10k	27
3	Catholic	<$10k	418
4	Don't know/refused	<$10k	15

열과 피벗

- variable, value 열의 이름 바꾸기

```
pd.melt(pew, id_vars='religion', var_name='income', value_name='count').head()
```

	religion	income	count
0	Agnostic	<$10k	27
1	Atheist	<$10k	12
2	Buddhist	<$10k	27
3	Catholic	<$10k	418
4	Don't know/refused	<$10k	15

열과 피벗

- 2개 이상의 열을 고정하고 나머지 열을 행으로 바꾸기

빌보드 차트 데이터

```
billboard = pd.read_csv('../data/billboard.csv')
billboard.iloc[0:5, 0:10]
```

	year	artist	track	time	date.entered	wk1	wk2	wk3	wk4	wk5
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	87	82.0	72.0	77.0	87.0
1	2000	2Ge+her	The Hardest Part Of ...	3:15	2000-09-02	91	87.0	92.0	NaN	NaN
2	2000	3 Doors Down	Kryptonite	3:53	2000-04-08	81	70.0	68.0	67.0	66.0
3	2000	3 Doors Down	Loser	4:24	2000-10-21	76	76.0	72.0	69.0	67.0
4	2000	504 Boyz	Wobble Wobble	3:35	2000-04-15	57	34.0	25.0	17.0	17.0

열과 피벗

- 2개 이상의 열을 고정하고 나머지 열을 행으로 바꾸기

```
pd.melt(billboard, id_vars=['year', 'artist', 'track', 'time', 'date.entered'], var_name='week', value_name='rating').head()
```

	year	artist	track	time	date.entered	week	rating
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	wk1	87.0
1	2000	2Ge+her	The Hardest Part Of ...	3:15	2000-09-02	wk1	91.0
2	2000	3 Doors Down	Kryptonite	3:53	2000-04-08	wk1	81.0
3	2000	3 Doors Down	Loser	4:24	2000-10-21	wk1	76.0
4	2000	504 Boyz	Wobble Wobble	3:35	2000-04-15	wk1	57.0

열 이름 관리하기

- 하나의 열이 여러 의미를 가지고 있는 경우

ebola 데이터 집합의 열 중 하나인 Deaths_Guinea는 '사망자 수'와 '나라 이름'을 합쳐 만든 이름

```
ebola.iloc[:5, [0, 1, 2, 3, 10, 11]]
```

	Date	Day	Cases_Guinea	Cases_Liberia	Deaths_Guinea	Deaths_Liberia
0	1/5/2015	289	2776.0	NaN	1786.0	NaN
1	1/4/2015	288	2775.0	NaN	1781.0	NaN
2	1/3/2015	287	2769.0	8166.0	1767.0	3496.0
3	1/2/2015	286	NaN	8157.0	NaN	3496.0
4	12/31/2014	284	2730.0	8115.0	1739.0	3471.0

열 이름 관리하기

- 하나의 열이 여러 의미를 가지고 있는 경우

Date와 Day를 고정하고 나머지를 행으로 피벗하면 각 나라별 발병자 수를 행으로 볼 수 있어 편리

```
ebola_long = pd.melt(ebola, id_vars=['Date', 'Day'])
ebola_long.head()
```

	Date	Day	variable	value
0	1/5/2015	289	Cases_Guinea	2776.0
1	1/4/2015	288	Cases_Guinea	2775.0
2	1/3/2015	287	Cases_Guinea	2769.0
3	1/2/2015	286	Cases_Guinea	NaN
4	12/31/2014	284	Cases_Guinea	2730.0

```
ebola_long.groupby('variable')['value'].sum()
```

```
variable
Cases_Guinea              84729.0
Cases_Liberia            193833.0
Cases_Mali                   42.0
Cases_Nigeria               636.0
Cases_Senegal                27.0
Cases_SierraLeone         211181.0
Cases_Spain                  16.0
Cases_UnitedStates           59.0
Deaths_Guinea             51818.0
Deaths_Liberia            89198.0
Deaths_Mali                  38.0
Deaths_Nigeria              233.0
Deaths_Senegal                0.0
Deaths_SierraLeone        60352.0
Deaths_Spain                  3.0
Deaths_UnitedStates          15.0
```

열 이름 관리하기

- Cases_Guinea와 같이 2개 이상의 의미를 가지고 있는 열 이름을 Cases, Guinea 분리하려면?

- split()로 열 이름 분리하기
  - 기본적으로 공백을 기준으로 문자열을 자름

```
variable_split = ebola_long.variable.str.split('_')
variable_split[:5]
```

```
0    [Cases, Guinea]
1    [Cases, Guinea]
2    [Cases, Guinea]
3    [Cases, Guinea]
4    [Cases, Guinea]
```

```
type(variable_split)
```

```
pandas.core.series.Series
```

```
type(variable_split[0])
```

```
list
```

열 이름 관리하기

- status, country 라는 열 이름으로 데이터프레임에 추가

```
ebola_long['status'] = status_values
ebola_long['country'] = country_values
ebola_long.head()
```

	Date	Day	variable	value	status	country
0	1/5/2015	289	Cases_Guinea	2776.0	Cases	Guinea
1	1/4/2015	288	Cases_Guinea	2775.0	Cases	Guinea
2	1/3/2015	287	Cases_Guinea	2769.0	Cases	Guinea
3	1/2/2015	286	Cases_Guinea	NaN	Cases	Guinea
4	12/31/2014	284	Cases_Guinea	2730.0	Cases	Guinea

```
status_values = variable_split.str.get(0)
country_values = variable_split.str.get(1)

status_values[:5]
```

```
0    Cases
1    Cases
2    Cases
3    Cases
4    Cases
Name: variable, dtype: object
```

```
status_values[-5:]
```

```
1947    Deaths
1948    Deaths
1949    Deaths
1950    Deaths
1951    Deaths
Name: variable, dtype: object
```

```
country_values[:5]
```

```
0    Guinea
1    Guinea
2    Guinea
3    Guinea
4    Guinea
Name: variable, dtype: object
```

```
country_values[-5:]
```

```
1947    Mali
1948    Mali
1949    Mali
1950    Mali
1951    Mali
Name: variable, dtype: object
```

열 이름 관리하기

```
variable_split = ebola_long.variable.str.split('_')
variable_split[:5]
```

```
0    [Cases, Guinea]
1    [Cases, Guinea]
2    [Cases, Guinea]
3    [Cases, Guinea]
4    [Cases, Guinea]
```

```
type(variable_split)
```

```
pandas.core.series.Series
```

```
type(variable_split[0])
```

```
list
```

```
variable_split = ebola_long.variable.str.split('_', expand=True)
variable_split[:5]
```

	0	1
0	Cases	Guinea
1	Cases	Guinea
2	Cases	Guinea
3	Cases	Guinea
4	Cases	Guinea

```
type(variable_split)
```

```
pandas.core.frame.DataFrame
```

```
type(variable_split[0])
```

```
pandas.core.series.Series
```

열 이름 관리하기

```
ebola_long = pd.melt(e
ebola_long.head()
```

	Date	Day	
0	1/5/2015	289	Cas
1	1/4/2015	288	Cas
2	1/3/2015	287	Cas
3	1/2/2015	286	Cas
4	12/31/2014	284	Cas

```
variable_split.columns = ['status', 'country']
pd.concat([ebola_long, variable_split], axis=1)
```

	Date	Day	variable	value	status	country
0	1/5/2015	289	Cases_Guinea	2776.0	Cases	Guinea
1	1/4/2015	288	Cases_Guinea	2775.0	Cases	Guinea
2	1/3/2015	287	Cases_Guinea	2769.0	Cases	Guinea
3	1/2/2015	286	Cases_Guinea	NaN	Cases	Guinea
4	12/31/2014	284	Cases_Guinea	2730.0	Cases	Guinea
...	...	...	...	...	...	...
1947	3/27/2014	5	Deaths_Mali	NaN	Deaths	Mali
1948	3/26/2014	4	Deaths_Mali	NaN	Deaths	Mali
1949	3/25/2014	3	Deaths_Mali	NaN	Deaths	Mali
1950	3/24/2014	2	Deaths_Mali	NaN	Deaths	Mali
1951	3/22/2014	0	Deaths_Mali	NaN	Deaths	Mali

1952 rows × 6 columns

```
'_', expand=True)
```

여러 열을 하나로 정리하기

- 기상 데이터의 열을 하나로 정리하기
  - 각 월별 최고, 최저 온도 데이터가 저장되어 있음
  - 날짜 열이 옆으로 길게 늘어져 있어 보기 불편함

```
weather = pd.read_csv('../data/weather.csv')
weather.iloc[:5, :11]
```

	id	year	month	element	d1	d2	d3	d4	d5	d6	d7
0	MX17004	2010	1	tmax	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	MX17004	2010	1	tmin	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	MX17004	2010	2	tmax	NaN	27.3	24.1	NaN	NaN	NaN	NaN
3	MX17004	2010	2	tmin	NaN	14.4	14.4	NaN	NaN	NaN	NaN
4	MX17004	2010	3	tmax	NaN	NaN	NaN	NaN	32.1	NaN	NaN

여러 열을 하나로 정리하기

- 기상 데이터의 열을 하나로 정리하기
  - 날짜 열을 행 데이터로 피벗

```
weather_melt = pd.melt(weather, id_vars=['id', 'year', 'month', 'element'], var_name='day', value_name='temp')
weather_melt.head()
```

	id	year	month	element	day	temp
0	MX17004	2010	1	tmax	d1	NaN
1	MX17004	2010	1	tmin	d1	NaN
2	MX17004	2010	2	tmax	d1	NaN
3	MX17004	2010	2	tmin	d1	NaN
4	MX17004	2010	3	tmax	d1	NaN

여러 열을 하나로 정리하기

기상 데이터의 열을 하나로 정리하기

- pivot_table()
  - 행과 열의 위치를 다시 바꿔 정리 해주는 함수

arguments	description
index	위치를 그대로 유지할 열 이름 지정
columns	피벗할 열 이름 지정
values	새로운 열의 데이터가 될 열 이름 지정

```
weather_tidy = weather_melt.pivot_table(
    index=['id', 'year', 'month', 'day'],
    columns='element',
    values='temp'
)

weather_tidy
```

	id	year	month	element	day	temp
0	MX17004	2010	1	tmax	d1	NaN
1	MX17004	2010	1	tmin	d1	NaN
2	MX17004	2010	2	tmax	d1	NaN
3	MX17004	2010	2	tmin	d1	NaN
4	MX17004	2010	3	tmax	d1	NaN

여러 열을 하나로 정리하기

```
weather_tidy = weather_melt.pivot_table(
    index=['id', 'year', 'month', 'day'],
    columns='element',
    values='temp'
)

weather_tidy
```

id	year	month	day	element tmax	element tmin
		1	d30	27.8	14.5
		2	d11	29.7	13.4
			d2	27.3	14.4
			d23	29.9	10.7
			d3	24.1	14.4
		3	d10	34.5	16.8
			d16	31.1	17.6
			d5	32.1	14.2
		4	d27	36.3	16.7
		5	d27	33.2	18.2
		6	d17	28.0	17.5
			d29	30.1	18.0
		7	d3	28.6	17.5
			d14	29.9	16.5
MX17004	2010	8	d23	26.4	15.0
			d5	29.6	15.8
			d29	28.0	15.3
			d13	29.8	16.5
			d25	29.7	15.6
			d31	25.4	15.4
			d8	29.0	17.3
		10	d5	27.0	14.0
			d14	29.5	13.0
			d15	28.7	10.5
			d28	31.2	15.0
			d7	28.1	12.9
		11	d2	31.3	16.3
			d5	26.3	7.9
			d27	27.7	14.2
			d26	28.1	12.1
			d4	27.2	12.0
		12	d1	29.9	13.8
			d6	27.8	10.5

여러 열을 하나로 정리하기

- reset_index()
  - 데이터프레임의 인덱스를 새로 지정하는 함수

			element	tmax	tmin
id	year	month	day		
		1	d30	27.8	14.5
		2	d11	29.7	13.4
			d2	27.3	14.4
			d23	29.9	10.7
			d3	24.1	14.4
		3	d10	34.5	16.8
			d16	31.1	17.6
			d5	32.1	14.2
		4	d27	36.3	16.7
		5	d27	33.2	18.2
		6	d17	28.0	17.5
			d29	30.1	18.0

```
weather_tidy_flat = weather_tidy.reset_index()
weather_tidy_flat.head()
```

	element	id	year	month	day	tmax	tmin
0	MX17004	2010	1	d30	27.8	14.5	
1	MX17004	2010	2	d11	29.7	13.4	
2	MX17004	2010	2	d2	27.3	14.4	
3	MX17004	2010	2	d23	29.9	10.7	
4	MX17004	2010	2	d3	24.1	14.4	

중복 데이터 처리하기

- 빌보드 차트 데이터는 artist, track, time, date.entered 열의 데이터가 반복됨

```
billboard = pd.read_csv('../data/billboard.csv')
billboard.iloc[0:5, 0:10]
```

	year	artist	track	time	date.entered	wk1	wk2	wk3	wk4	wk5
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	87	82.0	72.0	77.0	87.0
1	2000	2Ge+her	The Hardest Part	3:15	2000-09-02	91	87.0	92.0	NaN	NaN

```
billboard_long = pd.melt(billboard, id_vars=['year', 'artist', 'track', 'time', 'date.entered'],
                         var_name='week', value_name='rating')
billboard_long.head()
```

	year	artist	track	time	date.entered	week	rating
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	wk1	87.0
1	2000	2Ge+her	The Hardest Part Of ...	3:15	2000-09-02	wk1	91.0
2	2000	3 Doors Down	Kryptonite	3:53	2000-04-08	wk1	81.0
3	2000	3 Doors Down	Loser	4:24	2000-10-21	wk1	76.0
4	2000	504 Boyz	Wobble Wobble	3:35	2000-04-15	wk1	57.0

중복 데이터 처리하기

- 노래 제목 (track)이 Loser인 데이터만 따로 모아 살펴보면 중복 데이터가 꽤 많다는 것을 알 수 있음

```
billboard_long[billboard_long.track == 'Loser'].head()
```

	year	artist	track	time	date.entered	week	rating
3	2000	3 Doors Down	Loser	4:24	2000-10-21	wk1	76.0
320	2000	3 Doors Down	Loser	4:24	2000-10-21	wk2	76.0
637	2000	3 Doors Down	Loser	4:24	2000-10-21	wk3	72.0
954	2000	3 Doors Down	Loser	4:24	2000-10-21	wk4	69.0
1271	2000	3 Doors Down	Loser	4:24	2000-10-21	wk5	67.0

중복 데이터 처리하기

- 중복 데이터를 가지고 있는 열은 year, artist, track, time, date.entered
- 위의 열을 따로 모아 새로운 데이터프레임에 저장

```
billboard_long[billboard_long.track == 'Loser'].head()
```

	year	artist	track	time	date.entered	week	rating
3	2000	3 Doors Down	Loser	4:24	2000-10-21	wk1	76.0
320	2000	3 Doors Down	Loser	4:24	2000-10-21	wk2	76.0
637	2000	3 Doors Down	Loser	4:24			
954	2000	3 Doors Down	Loser	4:24			
1271	2000	3 Doors Down	Loser	4:24			

```
billboard_songs = billboard_long[['year', 'artist', 'track', 'time']]
billboard_songs.head()
```

	year	artist	track	time
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22
1	2000	2Ge+her	The Hardest Part Of ...	3:15
2	2000	3 Doors Down	Kryptonite	3:53
3	2000	3 Doors Down	Loser	4:24
4	2000	504 Boyz	Wobble Wobble	3:35

중복 데이터 처리하기

- drop_duplicates()로 데이터프레임의 중복 데이터를 제거

```
billboard_songs = billboard_long[['year', 'artist', 'track', 'time']]
print(billboard_songs.shape)
billboard_songs.head()
```

```
(24092, 4)
```

	year	artist	track	time
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22
1	2000	2Ge+her	The Hardest Part Of ...	3:15
2	2000	3 Doors Down	Kryptonite	3:53
3	2000	3 Doors Down	Loser	4:24
4	2000	504 Boyz	Wobble Wobble	3:35

```
billboard_songs = billboard_songs.drop_duplicates()
billboard_songs
```

	year	artist	track	time
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22
1	2000	2Ge+her	The Hardest Part Of ...	3:15
2	2000	3 Doors Down	Kryptonite	3:53
3	2000	3 Doors Down	Loser	4:24
4	2000	504 Boyz	Wobble Wobble	3:35
...	...	...	...	...
312	2000	Yankee Grey	Another Nine Minutes	3:10
313	2000	Yearwood, Trisha	Real Live Woman	3:55
314	2000	Ying Yang Twins	Whistle While You Tw...	4:19
315	2000	Zombie Nation	Kernkraft 400	3:30
316	2000	matchbox twenty	Bent	4:12

317 rows × 4 columns

중복 데이터 처리하기

- 중복을 제거한 데이터프레임에 아이디(id) 추가

```
billboard_songs['id'] = range(len(billb
billboard_songs.head(n=10)
```

```
C:₩Users₩wkimc₩Anaconda3₩lib₩site-packa
arning:
A value is trying to be set on a copy o
Try using .loc[row_indexer,col_indexer]

See the caveats in the documentation: h
er_guide/indexing.html#returning-a-view
  """Entry point for launching an IPyth
```

	year	artist	tr
0	2000	2 Pac	Baby Don't Cry (Kee
1	2000	2Ge+her	The Hardest Part O
2	2000	3 Doors Down	Krypto
3	2000	3 Doors Down	Lo
4	2000	504 Boyz	Wobble Wob
5	2000	98^0	Give Me Just One Ni
6	2000	A*Teens	Dancing Qu
7	2000	Aaliyah	I Don't Wa
8	2000	Aaliyah	Try Ag
9	2000	Adams, Yolanda	Open My He

```
billboard_songs.loc[:, 'id'] = range(len(billboard_songs))
billboard_songs.head(n=10)
```

	year	artist	track	time	id
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22	0
1	2000	2Ge+her	The Hardest Part Of ...	3:15	1
2	2000	3 Doors Down	Kryptonite	3:53	2
3	2000	3 Doors Down	Loser	4:24	3
4	2000	504 Boyz	Wobble Wobble	3:35	4
5	2000	98^0	Give Me Just One Nig...	3:24	5
6	2000	A*Teens	Dancing Queen	3:44	6
7	2000	Aaliyah	I Don't Wanna	4:15	7
8	2000	Aaliyah	Try Again	4:03	8
9	2000	Adams, Yolanda	Open My Heart	5:30	9

중복 데이터 처리하기

- merge()를 사용해 노래 정보와 주간 순위 데이터 병합

```
billboard_long = pd.melt(billboard, id_vars=['year', 'artist', 'track', 'time', 'date.entered'],
                         var_name='week', value_name='rating')
billboard_long.head()
```

```
billboard_ratings = billboard_long.merge( billboard_songs, on=['year', 'artist', 'track', 'time'])
print(billboard_ratings.shape)
billboard_ratings.head()
```

```
(24092, 8)
```

	year	artist	track	time	date.entered	week	rating	id
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	wk1	87.0	0
1	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	wk2	82.0	0
2	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	wk3	72.0	0
3	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	wk4	77.0	0
4	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	wk5	87.0	0
8	2000	Aaliyah	Try Again	4:03				8
9	2000	Adams, Yolanda	Open My Heart	5:30				9

Thank You!