깔끔한 데이터

- 열과 피벗
- 열 이름 관리하기
- 여러 열을 하나로 정리하기
- 중복 데이터 처리하기
- 대용량 데이터 처리하기

■ 열과 피벗

● 넓은 데이터

- 데이터프레임의 열이 옆으로 길게 늘어선 형태

ex) 열 자체가 어떤 값을 의미하는 경우

	10년간 변화	증감	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
경기	2,358	증가	8,934	9,219	9,544	9,927	10,207	10,463	10,697	10,906	11,106	11,292
계	2,205	증가	47,335	47,733	48,022	48,230	48,387	48,582	48,782	48,990	49,269	49,540
인천	184	증가	2,509	2,546	2,565	2,578	2,570	2,579	2,600	2,624	2,665	2,693
경남	145	증가	3,080	3,094	3,107	3,124	3,139	3,144	3,160	3,173	3,197	3,225
대전	117	증가	1,364	1,386	1,403	1,420	1,432	1,443	1,455	1,466	1,476	1,481
충남	100	증가	1,919	1,922	1,918	1,908	1,913	1,953	1,963	1,974	1,996	2,019
울산	88	증가	1,024	1,040	1,056	1,065	1,073	1,081	1,088	1,092	1,100	1,112
광주	66	증가	1,357	1,372	1,384	1,397	1,396	1,401	1,402	1,408	1,413	1,423
충북	28	증가	1,492	1,498	1,497	1,493	1,490	1,489	1,489	1,495	1,507	1,520
제주	22	증가	539	542	547	551	552	555	558	558	559	561
대구	-13	감소	2,506	2,524	2,525	2,526	2,530	2,525	2,511	2,496	2,493	2,493
강원	-48	감소	1,557	1,555	1,552	1,539	1,527	1,521	1,513	1,505	1,504	1,509
서울	-63	감소	10,264	10,311	10,263	10,207	10,174	10,173	10,167	10,181	10,193	10,201
경북	-135	감소	2,809	2,797	2,785	2,757	2,721	2,696	2,688	2,689	2,681	2,674
전북	-154	감소	2,010	1,999	2,006	1,954	1,954	1,907	1,885	1,868	1,862	1,856
전남	-236	감소	2,155	2,131	2,099	2,054	2,018	1,986	1,967	1,943	1,930	1,919
부산	-252	감소	3,817	3,797	3,771	3,730	3,691	3,666	3,638	3,612	3,587	3,565

이름	1. 1.	1. 2.	1. 3.
김길동	O	X	O
이길동	O	O	X

■ 열과 피벗

● 넓은 데이터

- 열의 데이터를 행으로 정리 : melt()

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

	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
5	Evangelical Prot	575	869	1064	982	881	1486	949	723	414	1529
6	Hindu	1	9	7	9	11	34	47	48	54	37
7	Historically Black Prot	228	244	236	238	197	223	131	81	78	339
8	Jehovah's Witness	20	27	24	24	21	30	15	11	6	37
9	Jewish	19	19	25	25	30	95	69	87	151	162
10	Mainline Prot	289	495	619	655	651	1107	939	753	634	1328
11	Mormon	29	40	48	51	56	112	85	49	42	69
12	Muslim	6	7	9	10	9	23	16	8	6	22
13	Orthodox	13	17	23	32	32	47	38	42	46	73
14	Other Christian	9	7	11	13	13	14	18	14	12	18
15	Other Faiths	20	33	40	46	49	63	46	40	41	71
16	Other World Religions	5	2	3	4	2	7	3	4	4	8
17	Unaffiliated	217	299	374	365	341	528	407	321	258	597

■ 열과 피벗

● 넓은 데이터

- 열의 데이터를 행으로 정리 : melt()

```
pew_long = pd.melt(pew, id_vars='religion')
print(pew_long[:20])
```

	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
5	Evangelical Prot	<$10k	575
6	Hindu	<$10k	1
7	Historically Black Prot	<$10k	228
8	Jehovah's Witness	<$10k	20
9	Jewish	<$10k	19
10	Mainline Prot	<$10k	289
11	Mormon	<$10k	29
12	Muslim	<$10k	6
13	Orthodox	<$10k	13
14	Other Christian	<$10k	9
15	Other Faiths	<$10k	20
16	Other World Religions	<$10k	5
17	Unaffiliated	<$10k	217
18	Agnostic	$10-20k	34
19	Atheist	$10-20k	27

id_vars 에 지정된 열 ➔ 제외

나머지 소득정보 열 이름 ➔ variable

소득정보 열의 행 데이터 ➔ value

■ 열과 피벗

● 넓은 데이터

- 열의 데이터를 행으로 정리 : melt()

```
pew_long = pd.melt(
    pew, id_vars='religion',
    var_name='income', value_name='count')
print(pew_long[:10])
```

	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
5	Evangelical Prot	<$10k	575
6	Hindu	<$10k	1
7	Historically Black Prot	<$10k	228
8	Jehovah's Witness	<$10k	20
9	Jewish	<$10k	19

var_name : variable → income

value_name : value → count

■ 열과 피벗

● 넓은 데이터

- 빌보드 차트 데이터 피벗 (2개 이상의 열 고정 / 나머지 행 변환)

```
billboard = pd.read_csv('data/billboard.csv')
print(billboard.shape)
print(billboard.columns)
```

```
(317, 81)
Index(['year', 'artist', 'track', 'time', 'date.entered', 'wk1', 'wk2', 'wk3',
       'wk4', 'wk5', 'wk6', 'wk7', 'wk8', 'wk9', 'wk10', 'wk11', 'wk12',
       'wk13', 'wk14', 'wk15', 'wk16', 'wk17', 'wk18', 'wk19', 'wk20', 'wk21',
       'wk22', 'wk23', 'wk24', 'wk25', 'wk26', 'wk27', 'wk28', 'wk29', 'wk30',
       'wk31', 'wk32', 'wk33', 'wk34', 'wk35', 'wk36', 'wk37', 'wk38', 'wk39',
       'wk40', 'wk41', 'wk42', 'wk43', 'wk44', 'wk45', 'wk46', 'wk47', 'wk48',
       'wk49', 'wk50', 'wk51', 'wk52', 'wk53', 'wk54', 'wk55', 'wk56', 'wk57',
       'wk58', 'wk59', 'wk60', 'wk61', 'wk62', 'wk63', 'wk64', 'wk65', 'wk66',
       'wk67', 'wk68', 'wk69', 'wk70', 'wk71', 'wk72', 'wk73', 'wk74', 'wk75',
       'wk76'],
      dtype='object')
```

■ 열과 피벗

● 넓은 데이터

- 빌보드 차트 데이터 피벗 (2개 이상의 열 고정 / 나머지 행 변환)

```
billboard.iloc[0:5, 0:7]
```

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

■ 열과 피벗

● 넓은 데이터

- 빌보드 차트 데이터 피벗 (2개 이상의 열 고정 / 나머지 행 변환)

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

print(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

■ 열 이름 관리하기

● 하나의 열이 여러 의미를 가지는 경우

- split()과 concat()으로 데이터 정리
- 에볼라 바이러스 감염 및 사망자 수

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

```
Index(['Date', 'Day', 'Cases_Guinea', 'Cases_Liberia', 'Cases_SierraLeone',
       'Cases_Nigeria', 'Cases_Senegal', 'Cases_UnitedStates', 'Cases_Spain',
       'Cases_Mali', 'Deaths_Guinea', 'Deaths_Liberia', 'Deaths_SierraLeone',
       'Deaths_Nigeria', 'Deaths_Senegal', 'Deaths_UnitedStates',
       'Deaths_Spain', 'Deaths_Mali'],
      dtype='object')
```

```
print(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 가지는 경우

- Date, Day 열 고정 / 나머지 열 피벗

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

- 'variable' 열의 데이터를 _ 기준으로 분리

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

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

■ 열 이름 관리하기

● 하나의 열이 여러 의미를 가지는 경우

- 분리된 데이터 확인

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

print(status_values[:5])
print(country_values[:5])
```

```
0    Cases                      1947    Deaths
1    Cases                      1948    Deaths
2    Cases                      1949    Deaths
3    Cases                      1950    Deaths
4    Cases                      1951    Deaths
Name: variable, dtype: object  Name: variable, dtype: object
0    Guinea                     1947    Mali
1    Guinea                     1948    Mali
2    Guinea                     1949    Mali
3    Guinea                     1950    Mali
4    Guinea                     1951    Mali
Name: variable, dtype: object  Name: variable, dtype: object
```

■ 열 이름 관리하기

● 하나의 열이 여러 의미를 가지는 경우

1. 분리된 데이터를 'ebola_long' 데이터프레임에 추가

```
ebola_long['status'] = status_values
ebola_long['country'] = country_values
print(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

■ 열 이름 관리하기

● 하나의 열이 여러 의미를 가지는 경우

2. 분리된 데이터를 concat() 으로 합치기

```
variable_split = ebola_long.variable.str.split('_', expand=True)
variable_split.columns = ['status', 'country']
ebola_parsed = pd.concat([ebola_long, variable_split], axis=1)

print(ebola_parsed.head())
```

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

■ 여러 열을 하나로 정리하기

● 기상 데이터의 여러 열을 하나로 정리하기

- 기상 데이터 크기 및 열 이름 확인하기

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

```
(22, 35)
Index(['id', 'year', 'month', 'element', 'd1', 'd2', 'd3', 'd4', 'd5', 'd6',
       'd7', 'd8', 'd9', 'd10', 'd11', 'd12', 'd13', 'd14', 'd15', 'd16',
       'd17', 'd18', 'd19', 'd20', 'd21', 'd22', 'd23', 'd24', 'd25', 'd26',
       'd27', 'd28', 'd29', 'd30', 'd31'],
      dtype='object')
```

	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 여러 열을 하나로 정리하기

- melt() 사용 (열 데이터를 행 데이터로)

```
weather_melt = pd.melt(weather,
    id_vars=['id', 'year', 'month', 'element'],
    var_name='day', value_name='temp')
weather_melt[100:110]
```

	id	year	month	element	day	temp
100	MX17004	2010	7	tmax	d5	NaN
101	MX17004	2010	7	tmin	d5	NaN
102	MX17004	2010	8	tmax	d5	29.6
103	MX17004	2010	8	tmin	d5	15.8
104	MX17004	2010	10	tmax	d5	27.0
105	MX17004	2010	10	tmin	d5	14.0
106	MX17004	2010	11	tmax	d5	26.3
107	MX17004	2010	11	tmin	d5	7.9
108	MX17004	2010	12	tmax	d5	NaN
109	MX17004	2010	12	tmin	d5	NaN

■ 여러 열을 하나로 정리하기

● 기상 데이터의 여러 열을 하나로 정리하기

- pivot_table() 사용 (행 데이터를 열 데이터로)

```
weather_tidy = weather_melt.pivot_table(
    index=['id', 'year', 'month', 'day'],
    columns='element', values='temp')
print(weather_tidy.head(10))
```

element 열의 데이터 tmax / tmin 를 열로 변

tmax / tmin 열의 데이터는 temp

			element	tmax	tmin
id	year	month	day		
MX17004	2010	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

피벗 전 weather 데이터프레임

	id	year	month	element	day	temp
638	MX17004	2010	1	tmax	d30	27.8
639	MX17004	2010	1	tmin	d30	14.5

	id	year	month	element	day	temp
222	MX17004	2010	2	tmax	d11	29.7
223	MX17004	2010	2	tmin	d11	13.4

■ 여러 열을 하나로 정리하기

● 기상 데이터의 여러 열을 하나로 정리하기

- 인덱스 초기화

```
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 데이터의 중복 데이터 처리하기 (정규화)

- 빌보드 차트 데이터 가져오기

```
billboard = pd.read_csv('data/billboard.csv')
billboard.head()
```

```
(24092, 7)
```

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

■ 중복 데이터 처리하기

● 빌보드 차트 데이터의 중복 데이터 처리하기 (정규화)

- 빌보드 차트 데이터 피벗

	year	artist	track	time	date.entered	week	wk2
0	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	wk1	87
1	2000	2Ge+her	The Hardest Part Of ...	3:15	2000-09-02	wk1	91
...							
317	2000	2 Pac	Baby Don't Cry (Keep...	4:22	2000-02-26	wk2	82
318	2000	2Ge+her	The Hardest Part Of ...	3:15	2000-09-02	wk2	87
...							

id	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
...				

date.entered	week	wk2	id
2000-02-26	wk1	87	0
2000-09-02	wk1	91	1
2000-02-26	wk2	82	0
2000-09-02	wk2	87	1
...			

■ 중복 데이터 처리하기

● 빌보드 차트 데이터의 중복 데이터 처리하기 (정규화)

- 빌보드 차트 데이터 피벗

```
billboard_long = pd.melt(
    billboard,
    id_vars=['year', 'artist', 'track', 'time', 'date.entered'],
    var_name='week', value_name='rating')
print(billboard_long.shape)
billboard_long.head(10)
```

```
(24092, 7)
```

	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
5	2000	98^0	Give Me Just One Nig...	3:24	2000-08-19	wk1	51.0
6	2000	A*Teens	Dancing Queen	3:44	2000-07-08	wk1	97.0
7	2000	Aaliyah	I Don't Wanna	4:15	2000-01-29	wk1	84.0
8	2000	Aaliyah	Try Again	4:03	2000-03-18	wk1	59.0
9	2000	Adams, Yolanda	Open My Heart	5:30	2000-08-26	wk1	76.0

■ 중복 데이터 처리하기

● 빌보드 차트 데이터의 중복 데이터 처리하기 (정규화)

- 노래 제목으로 중복 데이터 현황 확인

```
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

● 빌보드 차트 데이터의 중복 데이터 처리하기 (정규화)

- 'year', 'artist', 'track', 'time' 중복 데이터를 새로운 데이터프레임으로 저장

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

```
(24092, 4)
```

- 중복 데이터 제거 (drop_duplicates())

```
billboard_songs = billboard_songs.drop_duplicates()
print(billboard_songs.shape)
```

```
(317, 4)
```

- 순번(id) 추가

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

■ 중복 데이터 처리하기

● 빌보드 차트 데이터의 중복 데이터 처리하기 (정규화)

- 노래 정보와 주간 순위 데이터 합치기

```
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

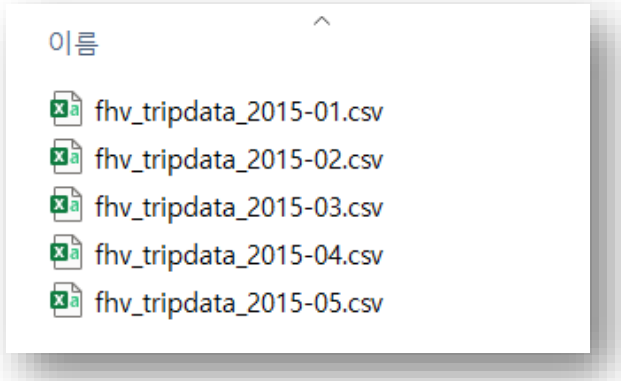
■ 대용량 데이터 처리하기

● 뉴욕 택시 데이터 가져오기

- 택시 데이터 5개 다운로드 (500MB 이상)

```
import os
import urllib.request

with open('data/raw_data_urls.txt', 'r') as data_urls:
    for line, url in enumerate(data_urls):
        if line == 5:
            break
        fn = url.split('/')[-1].strip()  # 파일명만 추출
        fp = os.path.join('', 'data', fn)  # data 폴더 저장 경로 지정
        print(url)
        print(fp)
        urllib.request.urlretrieve(url, fp)
```



■ 대용량 데이터 처리하기

● 뉴욕 택시 데이터 가져오기

- 다운로드 파일 목록

```
import glob
nyc_taxi_data = glob.glob('data/fhv_*')
print(nyc_taxi_data)
```

```
['data\\fhv_tripdata_2015-01.csv', 'data\\fhv_tripdata_2015-02.csv',
'data\\fhv_tripdata_2015-03.csv', 'data\\fhv_tripdata_2015-04.csv',
'data\\fhv_tripdata_2015-05.csv']
```

- 각각의 파일을 데이터프레임으로 저장

```
taxi1 = pd.read_csv(nyc_taxi_data[0])
taxi2 = pd.read_csv(nyc_taxi_data[1])
taxi3 = pd.read_csv(nyc_taxi_data[2])
taxi4 = pd.read_csv(nyc_taxi_data[3])
taxi5 = pd.read_csv(nyc_taxi_data[4])
print(taxi1.shape)
print(taxi2.shape)
print(taxi3.shape)
print(taxi4.shape)
print(taxi5.shape)
```

```
(2746033, 3)
(3126401, 3)
(3281427, 3)
(3917789, 3)
(4296067, 3)
```

■ 대용량 데이터 처리하기

● 뉴욕 택시 데이터 가져오기

- 데이터 확인

```
print(taxi1.head(2))
print(taxi2.head(2))
print(taxi3.head(2))
print(taxi4.head(2))
print(taxi5.head(2))
```

```
  Dispatching_base_num          Pickup_date  locationID
0               B00013  2015-01-01 00:30:00         NaN
1               B00013  2015-01-01 01:22:00         NaN
  Dispatching_base_num          Pickup_date  locationID
0               B00013  2015-02-01 00:00:00         NaN
1               B00013  2015-02-01 00:01:00         NaN
  Dispatching_base_num          Pickup_date  locationID
0               B00029  2015-03-01 00:02:00       213.0
1               B00029  2015-03-01 00:03:00        51.0
  Dispatching_base_num          Pickup_date  locationID
0               B00001  2015-04-01 04:30:00         NaN
1               B00001  2015-04-01 06:00:00         NaN
  Dispatching_base_num          Pickup_date  locationID
0               B00001  2015-05-01 04:30:00         NaN
(17367717, 3)   B00001  2015-05-01 05:00:00         NaN
```

■ 대용량 데이터 처리하기

● 뉴욕 택시 데이터 가져오기

- 데이터 처리를 위해 각 데이터프레임 연결

```
taxi = pd.concat([taxi1, taxi2, taxi3, taxi4, taxi5])
print(taxi.shape)
taxi[2746031:2746041]
```

```
(17367717, 3)
```

	Dispatching_base_num	Pickup_date	locationID
2746031	B02765	2015-01-31 23:59:40	181.0
2746032	B02765	2015-01-31 23:59:48	79.0
0	B00013	2015-02-01 00:00:00	NaN
1	B00013	2015-02-01 00:01:00	NaN
2	B00013	2015-02-01 00:21:00	NaN
3	B00013	2015-02-01 01:00:00	NaN
4	B00013	2015-02-01 02:10:00	NaN
5	B00013	2015-02-01 03:34:00	NaN
6	B00013	2015-02-01 03:37:00	NaN
7	B00013	2015-02-01 03:39:00	NaN

■ 대용량 데이터 처리하기

● 뉴욕 택시 데이터 가져오기

- 데이터 처리를 위해 각 데이터프레임 연결 (인덱스 정리)

```
taxi = pd.concat([taxi1, taxi2, taxi3, taxi4, taxi5], ignore_index=True)
print(taxi.shape)
taxi[2746031:2746041]
```

```
(17367717, 3)
```

	Dispatching_base_num	Pickup_date	locationID
2746031	B02765	2015-01-31 23:59:40	181.0
2746032	B02765	2015-01-31 23:59:48	79.0
2746033	B00013	2015-02-01 00:00:00	NaN
2746034	B00013	2015-02-01 00:01:00	NaN
2746035	B00013	2015-02-01 00:21:00	NaN
2746036	B00013	2015-02-01 01:00:00	NaN
2746037	B00013	2015-02-01 02:10:00	NaN
2746038	B00013	2015-02-01 03:34:00	NaN
2746039	B00013	2015-02-01 03:37:00	NaN
2746040	B00013	2015-02-01 03:39:00	NaN