

▼ Ch2. 엑셀의 VLOOKUP 및 여러기능 판다스로 해보기

▼ 1. 엑셀 파일에서 데이터프레임 불러오기(read_excel)

pandas read_excel

엑셀파일을 데이터프레임으로 불러오는 함수

io

파일의 경로명

sheet_name (인수는 문자열, 정수, 리스트 / 기본값은 0)

불러올 시트를 지정하는 인자

예)

- 지정하지 않을 때: 첫번째 시트를 불러온다
- 1: 2번째 시트를 불러온다
- "Sheet1": 문자열을 입력하면 해당 이름을 가진 시트를 불러온다. 여기서는 "Sheet1"이라는 이름의 시트를 불러온다
- [0, 1, "Sheet5"]: 첫번째 시트와 두번째 시트 그리고 "Sheet5"라는 이름의 시트 세개를 딕셔너리로 통합해 가져온다.
- None: 모든 시트를 딕셔너리로 통합해 가져온다.

header (인수는 정수, 정수의 리스트 / 기본값은 0)

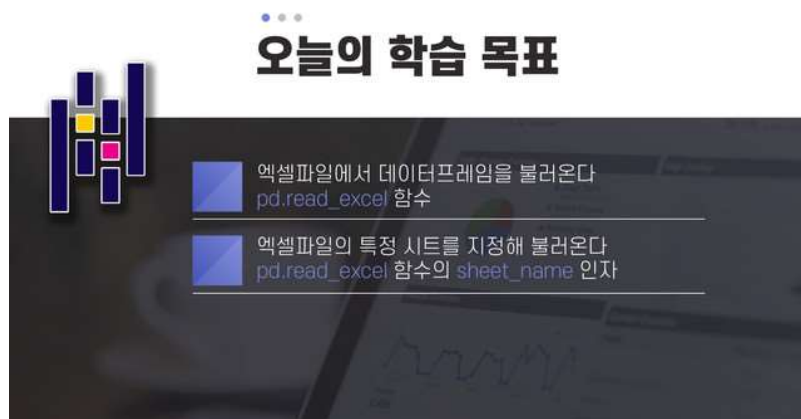
columns를 지정하는 인자. 지정하지 않으면 맨 윗줄이 columns가 된다. 리스트로 지정하면 멀티 인덱스인 columns가 된다.

index_col (인수는 정수, 정수의 리스트 / 기본값은 None)

index를 지정하는 인자. 지정하지 않으면 RangeIndex가 index로 부여된다. 리스트로 지정하면 멀티 인덱스인 index가 된다.

[read_excel 함수 설명 블로그](#)

[read_excel 판다스 공식 문서](#)



```
import pandas as pd
```

```
# 먼저 파일을 업로드 후 경로를 복사해서 변수(여기서는 url)로 지정한다
url = '/content/02_01_read_excel.xlsx'
```

```
# read_excel 함수로 데이터프레임을 불러온다. 기본값으로는 첫번째 시트
df = pd.read_excel(url)
df
```

	이름	국어	영어	수학
0	송중기	63	93	97
1	김나현	89	83	71
2	권보아	83	76	92

두번째 시트 불러오기

```
pd.read_excel(url, sheet_name=1)
```

	두 번째 시트
0	송중기 63 93 97
1	김나현 89 83 71
2	권보아 83 76 92
3	박효신 94 88 73

세번째 시트를 불러오되 세번째 행을 columns로 지정하기

```
pd.read_excel(url, sheet_name=2, header=2)
```

	이름	국어	영어	수학
0	송중기	63	93	97
1	김나현	89	83	71
2	권보아	83	76	92
3	박효신	94	88	73

네번째 시트를 불러오되 첫번째 열을 index로 지정하기

```
pd.read_excel(url, sheet_name=3, index_col=0)
```

	국어	영어	수학
송중기	63	93	97
김나현	89	83	71
권보아	83	76	92
박효신	94	88	73

2. 단일요건 vlookup (merge)

vlookup 이란?



- 두 데이터프레임을 기준 열의 내용에 따라 병합하는 엑셀 함수
- 엑셀에서 가장 유용한 함수로 꼽힌다

pandas merge

right (인수 데이터프레임 혹은 시리즈)

병합할 객체

how (인수는 'left', 'right', 'outer', 'inner', 'cross' / 기본값은 'inner')

병합할 방식을 결정하는 인자.

- left: 왼쪽 데이터프레임의 키(key)만을 병합에 사용한다.(엑셀의 vlookup과 유사)
- right: 오른쪽 데이터프레임의 키(key)만을 병합에 사용한다.
- outer: 양쪽 데이터프레임의 키(key)들의 합집합을 병합에 사용한다.
- inner: 양쪽 데이터프레임의 키(key)들의 교집합을 병합에 사용한다.
- cross: 양쪽 데이터프레임의 곱집합(cartesian product)을 생성한다.

on (인수는 열의 레이블 또는 리스트 / 기본값은 None)

병합의 기준이 되는 열을 지정하는 인자. 기본 값은 양쪽 데이터프레임에서 이름이 공통인 열들이 지정된다. 리스트로 입력하면 복수의 열을 기준으로 병합한다.

[merge 함수 설명 블로그](#)

[merge 판다스 공식 문서](#)



```
import pandas as pd
```

```
# 판다스 출력옵션 조절
pd.set_option('max_rows', 6)
```

```
url = '/content/02_02_merge01.xlsx'
df1 = pd.read_excel(url)
df1
```

	이름	제품
0	백철민	아이스티
1	박태인	레몬에이드
2	국지용	아메리카노
...
371	기주봉	아이스티
372	백승환	바닐라라떼
373	계승현	레몬에이드

374 rows × 2 columns

```
df2 = pd.read_excel(url, sheet_name=1)
df2
```

	제품	가격
0	아메리카노	3900
1	바닐라라떼	3900
2	까페라떼	3900
...
7	자몽티	4100

```
# df1과 df2를 vlookup 방식으로 병합
df3 = df1.merge(df2, how='left', on='제품')
df3
```

	이름	제품	가격
0	백철민	아이스티	4400
1	박태인	레몬에이드	4400
2	국지용	아메리카노	3900
...
371	기주봉	아이스티	4400
372	백승환	바닐라라떼	3900
373	계승현	레몬에이드	4400

374 rows × 3 columns

```
# 결과를 엑셀 파일로 저장하기
df3.to_excel('result_merge01.xlsx')
```

3. 다중요건 vlookup (merge)

다중 요건 vlookup 이란?

	이름	업체	음료
0	송중기	별다방	아이스티
1	김나현	콩다방	카페오레
2	권보아	별다방	카페오레
3	박효신	콩다방	아이스티
4	김병수	별다방	카페오레

df1

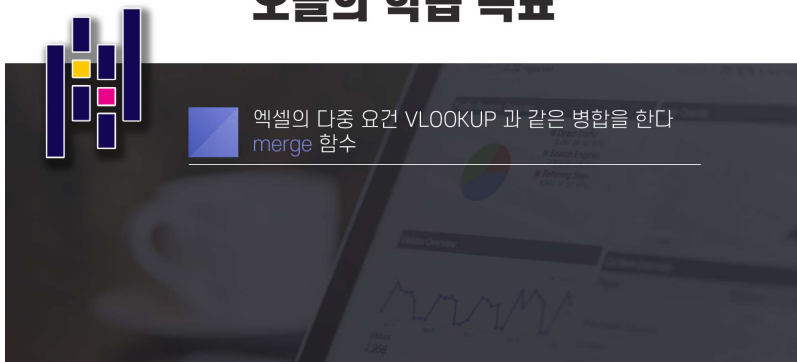
	업체	음료	가격
0	별다방	아이스티	4500
1	별다방	카페오레	4000
2	콩다방	아이스티	6000
3	콩다방	카페오레	5500

df2

	이름	업체	음료	가격
0	송중기	별다방	아이스티	4500
1	김나현	콩다방	카페오레	5500
2	권보아	별다방	카페오레	4000
3	박효신	콩다방	아이스티	6000
4	김병수	별다방	카페오레	4000

- 복수의 열을 기준으로 vlookup을 하는 것
- 위 그림의 경우 업체와 음료 모두 기준 열로 삼아 두 데이터프레임을 병합하였다

오늘의 학습 목표



```
import pandas as pd
pd.set_option('max_rows', 6)
data1 = [['송중기', '별다방', '아이스티'],
          ['김나현', '콩다방', '카페오레'],
          ['권보아', '별다방', '카페오레'],
          ['박효신', '콩다방', '아이스티'],
          ['김범수', '별다방', '카페오레']]
data2 = [['별다방', '아이스티', 4500],
          ['별다방', '카페오레', 4000],
          ['콩다방', '아이스티', 6000],
          ['콩다방', '카페오레', 5500]]
df1 = pd.DataFrame(data1, columns=['이름', '업체', '음료'])
df2 = pd.DataFrame(data2, columns=['업체', '음료', '가격'])
```

df1과 df2를 업체와 음료 두개의 열로 vlookup 하기
df1.merge(df2, how='left', on=['업체', '음료'])

	이름	업체	음료	가격
0	송중기	별다방	아이스티	4500
1	김나현	콩다방	카페오레	5500
2	권보아	별다방	카페오레	4000
3	박효신	콩다방	아이스티	6000
4	김범수	별다방	카페오레	4000

기본 값으로 양쪽 데이터프레임에서 공통인 열이 모두 on으로 지정된다
df1.merge(df2, how='left')

	이름	업체	음료	가격
0	송중기	별다방	아이스티	4500
1	김나현	콩다방	카페오레	5500
2	권보아	별다방	카페오레	4000
3	박효신	콩다방	아이스티	6000
4	김범수	별다방	카페오레	4000

엑셀파일로 한번 더 실습해보자
url = '/content/02_03_merge02.xlsx'
df3 = pd.read_excel(url)
df4 = pd.read_excel(url, sheet_name=1)

df3와 df4를 다중요건 vlookup을 한 뒤 엑셀파일로 저장하자
df3.merge(df4, how='left').to_excel('result_merge02.xlsx', index=False)

merge 함수에 대해서 더 공부할 필요한 분들은 아래 강의를 참고하세요

- [엑셀튜파이션 채널 merge 강의](#)

▼ 4. 행과 열의 이름 바꾸기(rename)

pandas rename

index (인수는 mapper)

index의 이름을 바꾸는 인자

- mapper란? 딕셔너리나 시리즈나 함수와 같이 매핑을 할수 있는 매개체

columns (인수는 mapper)

columns의 이름을 바꾸는 인자

level (인수는 **level**의 로케이션 혹은 레이블 / 기본값은 None)

멀티인덱스에서 이름을 바꿀 레벨을 지정하는 인자. 기본값은 모든 **level**에서 이름을 바꾼다.

[rename 함수 설명 블로그](#)

[rename 판다스 공식문서](#)

오늘의 학습 목표

- index나 columns의 레이블(이름)을 바꿔보자
`rename` 함수
- mapper가 무엇인지 이해한다
- 함수 적용 결과가 원본을 덮어쓰지 않는다(`inplace=False`)는 것을 알게된다

```
import pandas as pd
pd.set_option('max_rows', 6)
data1 = [['송중기', '별다방', '아이스티'],
          ['김나현', '콩다방', '카페오레'],
          ['권보아', '별다방', '카페오레'],
          ['박효신', '콩다방', '아이스티'],
          ['김범수', '별다방', '카페오레']]
data2 = [['별다방', '아이스티', 4500],
          ['별다방', '카페오레', 4000],
          ['콩다방', '아이스티', 6000],
          ['콩다방', '카페오레', 5500]]
df1 = pd.DataFrame(data1, columns=['이름', '업체', '음료'])
df2 = pd.DataFrame(data2, columns=['업체', '제품', '가격'])
```

제품 열을 음료 열로 열의 이름을 바꿔보자
df2.rename(columns={'제품': '음료'})

	업체	음료	가격
0	별다방	아이스티	4500
1	별다방	카페오레	4000
2	콩다방	아이스티	6000
3	콩다방	카페오레	5500

df2를 실행해보라 (함수의 결과는 원본을 덮어쓰지 않는다)
df2

	업체	제품	가격
0	별다방	아이스티	4500
1	별다방	카페오레	4000
2	콩다방	아이스티	6000
3	콩다방	카페오레	5500

판다스의 원본 덮어쓰기(inplace)

판다스에서는 대부분의 함수를 적용한 결과가
원본을 덮어쓰지 않는다 (극히 일부의 예외는 있다)
함수 적용 결과를 기존 변수에 다시 지정하거나
inplace=True를 사용해야 원본을 덮어쓰게 된다

```
# 함수의 적용결과가 원본을 덮어쓰지 않기 위해 새로운 변수로 지정해야 한다
df3 = df2.rename(columns={'제품':'음료'})
df1.merge(df3, how='left')
```

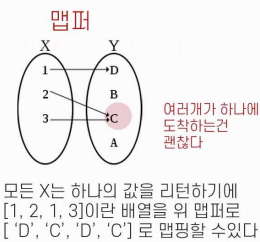
	이름	업체	음료	가격
0	송중기	별다방	아이스티	4500
1	김나현	콩다방	카페오레	5500
2	권보아	별다방	카페오레	4000
3	박효신	콩다방	아이스티	6000
4	김범수	별다방	카페오레	4000

```
# 혹은 함수를 적용 결과를 그대로 사용해도 된다
df1.merge(df2.rename(columns={'제품':'음료'}), how='left')
```

	이름	업체	음료	가격
0	송중기	별다방	아이스티	4500
1	김나현	콩다방	카페오레	5500
2	권보아	별다방	카페오레	4000
3	박효신	콩다방	아이스티	6000
4	김범수	별다방	카페오레	4000

맵퍼(mapper)란?

딕셔너리나 시리즈 함수와 같이
특정 값은 반드시 하나의 값을 리턴하기에
맵핑(mapping)을 할 수 있는 매개체



5. 정렬하기 (sort_values)

pandas sort_values

by (인수는 label 또는 label의 리스트)

정렬의 기준이 될 배열(주로 정렬의 기준인 열)을 지정하는 인자

ascending (인수는 bool 또는 bool의 리스트 / 기본값은 True)

정렬 순서를 오름차순으로 정렬할지 내림차순으로 정렬할지 지정하는 인자

[sort_values 함수 설명 블로그](#)

[sort_values 판다스 공식 문서](#)

오늘의 학습 목표

- 특정 열을 기준으로 정렬을 한다
`sort_values` 함수
- 오름차순과 내림차순으로 정렬한다
`sort_values` 함수의 `ascending` 인자
- 복수의 열을 기준으로
오름차순과 내림차순을 섞어서 정렬한다

```
import pandas as pd
data = [[60, 84, 80, 70, 19],
        [77, 62, 95, 85, 17],
        [61, 97, 72, 67, 15],
        [75, 65, 95, 51, 18]]
cols = ['국어', '영어', '수학', '과학', '나이']
df = pd.DataFrame(data, index=['A', 'B', 'C', 'D'], columns=cols)
df
```

	국어	영어	수학	과학	나이
A	60	84	80	70	19
B	77	62	95	85	17
C	61	97	72	67	15
D	75	65	95	51	18

```
# 국어열을 기준으로 오름차순으로 정렬하기
df.sort_values('국어')
```

	국어	영어	수학	과학	나이
A	60	84	80	70	19
C	61	97	72	67	15
D	75	65	95	51	18
B	77	62	95	85	17

```
# df는 바뀌지 않는다 (원본을 덮어쓰지 않음)
df
```

	국어	영어	수학	과학	나이
A	60	84	80	70	19
B	77	62	95	85	17
C	61	97	72	67	15
D	75	65	95	51	18

```
# 영어 열을 기준으로 내림차순으로 정렬하기
df.sort_values('영어', ascending=False)
```


	국어	영어	수학	과학	나이
C	61	97	72	67	15
A	60	84	80	70	19

수학으로 정렬하되 수학이 동점이면 과학으로 정렬 (모두 내림차순)
`df.sort_values(['수학', '과학'], ascending=False)`

	국어	영어	수학	과학	나이
B	77	62	95	85	17
D	75	65	95	51	18
A	60	84	80	70	19
C	61	97	72	67	15

수학으로 내림차순으로 정렬하되 수학이 동점이면 과학으로 오름차순 정렬
`df.sort_values(['수학', '과학'], ascending=[0, 1])`

	국어	영어	수학	과학	나이
D	75	65	95	51	18
B	77	62	95	85	17
A	60	84	80	70	19
C	61	97	72	67	15

▼ 6. 범위로 병합하기 (merge_asof)

pandas merge_asof

left (인수는 데이터프레임 혹은 시리즈)

병합할 객체1

right (인수는 데이터프레임 혹은 시리즈)

병합할 객체2

on (인수는 열의 레이블)

유사일치로 병합할 기준이 되는 열의 레이블을 지정한다.

- 반드시 하나의 열만 지정해야 한다.
- 숫자나 `datetime` 같이 유사일치가 가능한 자료형의 열이어야 한다.
- 오름차순으로 정렬이 되어 있어야 한다.

by (인수는 열의 레이블 또는 열의 레이블의 리스트)

정확히 일치시킬 열을 지정하는 인자.

direction (인수는 'backward', 'forward', 'nearest' / 기본값은 'backward')

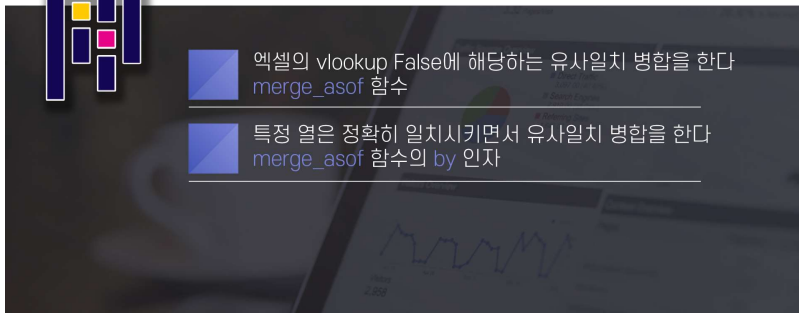
경계를 기준으로 유사일치시킬 방향을 결정하는 인자

- 'backward'는 작은 값중 최대를 찾아 병합한다. 기본값이다.
- 'forward'는 큰 값중 최소를 찾아 병합한다.
- 'nearest'는 가장 가까운 값을 찾아 병합한다.

[merge_asof 함수 설명 블로그](#)

[merge_asof 함수 판다스 공식문서](#)

오늘의 학습 목표



- 엑셀의 vlookup False에 해당하는 유사일치 병합을 한다
`merge_asof` 함수
- 특정 열은 정확히 일치시키면서 유사일치 병합을 한다
`merge_asof` 함수의 `by` 인자

```
import pandas as pd
pd.set_option('max_rows', 6)
data1 = [['가', 88], ['나', 72], ['다', 80], ['라', 60], ['마', 95]]
data2 = [[90, 'A'], [80, 'B'], [70, 'C'], [0, 'F']]
df1 = pd.DataFrame(data1, columns=['이름', '점수'])
df2 = pd.DataFrame(data2, columns=['점수', '학점'])
df2
```

	점수	학점
0	90	A
1	80	B
2	70	C
3	0	F

```
# 점수를 기준으로 범위로 병합한다 (반드시 오름차순으로 정렬되어 있어야함)
df1 = df1.sort_values('점수')
df2 = df2.sort_values('점수')
pd.merge_asof(df1, df2, on='점수')
```

	이름	점수	학점
0	라	60	F
1	나	72	C
2	다	80	B
3	가	88	B
4	마	95	A

```
# 엑셀파일로 merge_asof 실습하기
url = '/content/02_06_merge_asof.xlsx'
df3 = pd.read_excel(url)
df4 = pd.read_excel(url, sheet_name=1).sort_values('수량')

# 사이즈와 종류는 정확히 일치시키고 수량은 범위로 병합한다
df5 = pd.merge_asof(df3, df4, on='수량', by=['사이즈', '종류'])
df5
```

	사이즈	종류	수량	단가
0	A3	풀컬러	10	360
1	A4	컬러	10	90
2	A4	풀컬러	29	180
...

결과를 index는 제외하고 엑셀파일로 저장한다
`df5.to_excel('result_merge_asof.xlsx', index=False)`

99 A4 풀컬러 975 100

merge_asof 함수에 대해서 더 공부가 필요한 분들은 아래 강의를 참고하세요

[엑셀투파이썬 채널 merge_asof 강의](#)

▼ 7. 데이터 프레임 결합하기 (concat)

pandas concat

objs (인수는 시리즈 혹은 데이터프레임의 배열)

결합할 시리즈나 데이터프레임들을 리스트로 지정하는 인자

axis (인수는 0 또는 1 / 기본값은 0)

결합할 축 방향을 지정하는 인자. axis=0일때는 행이 증가하는 방향으로 결합하고, axis=1일때는 열이 증가하는 방향으로 결합한다.

join (인수는 'inner', 'outer' / 기본값은 'outer')

결합방식을 지정하는 인자.

- inner는 columns가 교집합으로 결합한다. (axis=0일때는 index가 교집합)
- outer는 columns가 합집합으로 결합한다. (axis=0일때는 index가 교집합)

keys (인수는 배열 / 기본값은 None)

결합하는 각 데이터프레임에 레벨을 부여하는 인자

[concat 함수 설명 블로그](#)

[concat 함수 판다스 공식문서](#)

오늘의 학습 목표

- 데이터 프레임을 결합한다
concat 함수
- 행방향과 열방향으로 결합한다
concat 함수의 axis 인자
- inner join과 outer join을 이해한다

```
import pandas as pd
```

```
data1 = [[80, 69, 83, 98], [71, 90, 69, 66], [74, 72, 72, 95]]
data2 = [[68, 70, 84, 70], [65, 91, 90, 66], [78, 94, 96, 64]]
data3 = [[84, 70, 70, 68], [90, 66, 91, 65], [96, 64, 94, 78]]
idx1 = ['송중기', '박보검', '김나현']
idx2 = ['권보아', '김범수', '박효신']
col = ['국어', '수학', '영어', '과학']
```

```
df1 = pd.DataFrame(data1, index=idx1, columns=col)
df2 = pd.DataFrame(data2, index=idx2, columns=col)
df3 = pd.DataFrame([[65, 82], [85, 60], [75, 78]],
                    index=idx1, columns=['사회', '음악'])
df4 = pd.DataFrame(data3, index=idx2,
                    columns=['영어', '과학', '수학', '사회'])
```

```
# 세로로 결합
pd.concat([df1, df2])
```

	국어	수학	영어	과학
송중기	80	69	83	98
박보검	71	90	69	66
김나현	74	72	72	95
권보아	68	70	84	70
김범수	65	91	90	66
박효신	78	94	96	64

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

	국어	수학	영어	과학	사회	음악
송중기	80	69	83	98	65	82
박보검	71	90	69	66	85	60
김나현	74	72	72	95	75	78

```
# outer join으로 결합
pd.concat([df1, df4])
```

	국어	수학	영어	과학	사회
송중기	80.0	69	83	98	NaN
박보검	71.0	90	69	66	NaN
김나현	74.0	72	72	95	NaN
권보아	NaN	70	84	70	68.0
김범수	NaN	91	90	66	65.0
박효신	NaN	94	96	64	78.0

```
# inner join으로 결합
pd.concat([df1, df4], join='inner')
```

	수학	영어	과학
송중기	69	83	98
박보검	90	69	66
김나현	72	72	95
권보아	70	84	70
김범수	91	90	66
박효신	94	96	64

```
# 엑셀파일에서 3개의 시트를 병합하기
url = '/content/02_07_concat.xlsx'
df1 = pd.read_excel(url)
```

```
df2 = pd.read_excel(url, sheet_name=1)
df3 = pd.read_excel(url, sheet_name=2)
pd.concat([df1, df2, df3])
```

	이름	국어	영어	소속
0	송중기	63	93	1반
1	김나현	89	83	1반
0	권보아	83	76	2반
1	박효신	94	88	2반
0	권혁수	58	91	3반
1	주현영	77	49	3반

```
# 딕셔너리에서 밸류(value)만 리스트에 담는 방법
dict_1 = {'key1': 'value1', 'key2': 'value2'}
dict_1.values()
```

```
dict_values(['value1', 'value2'])
```

```
# 엑셀파일의 모든 시트를 병합하는 코드
pd.concat(pd.read_excel(url, sheet_name=None).values())
```

	이름	국어	영어	소속
0	송중기	63	93	1반
1	김나현	89	83	1반
0	권보아	83	76	2반
1	박효신	94	88	2반
0	권혁수	58	91	3반
1	주현영	77	49	3반

concat 함수에 대해서 더 공부가 필요한 분들은 아래 강의를 참고하세요

[엑셀투파워 채널 concat 강의](#)