[Python] Pandas 사용법 - 다양한 인덱스 함수(reset_index,set_index,sort_index) — 나무늘보의 개발 블로그

노트북: 첫 번째 노트북
만든 날짜: 2020-11-06 오후 5:24
URL: https://continuous-development.tistory.com/134?category=736681

Python

[Python] Pandas 사용법 - 다양한 인덱스 함수(reset_index,set_index,sort_index)

2020. 10. 16. 00:45 수정 삭제 공개

reset_index()

기존 행 인덱스를 제거하고 인덱스를 데이터 열 추가

```
DataFrame.reset_index(inplace=True,drop=True)
=> 원본의 인덱스를 대체한다.
inplace는 생략 가능하다.

DataFrame.reset_index(inplace=False,drop=False)
=> 원본의 인덱스를 컬럼으로 만들고 인덱스를 생성한다.
```

새로운 인덱스를 할당하고, 기존 인덱스는 인덱스라는 새로운 칼럼명으로 추가

```
In [141]: titanic.head()
```

Out[141]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
5	0	3	male	NaN	0	0	8.4583	Q	Third	man	True	NaN	Queenstown	no	True
6	0	1	male	54.0	0	0	51.8625	S	First	man	True	E	Southampton	no	True
7	0	3	male	2.0	3	1	21.0750	S	Third	child	False	NaN	Southampton	no	False

```
In [142]: titanic_reset_index_df = titanic.reset_index(inplace=False)
          titanic_reset_index_df.head()
```

Out[142]:

	index	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
1	4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
2	5	0	3	male	NaN	0	0	8.4583	Q	Third	man	True	NaN	Queenstown	no	True
3	6	0	1	male	54.0	0	0	51.8625	S	First	man	True	E	Southampton	no	True
4	7	0	3	male	2.0	3	1	21.0750	S	Third	child	False	NaN	Southampton	no	False

이 같은식으로 쓴다 inplace가 False 기 때문에 받는 변수를 지정해줘야 한다.
만약 True로 한다면 원본데이터 변경되고 따로 지정은 안 해줘도 된다.

```
In [82]: index_df2.reset_index()
```

Out[82]:

	col01	col02	col03	col04
0	A	0.54	0.12	0.89
1	B	0.28	0.67	0.21
2	C	0.42	0.83	0.19
3	D	0.84	0.14	0.11
4	E	0.0	0.58	0.22

인덱스가 새로 생성 된 걸 볼 수 있다. 여기서 drop 옵션을 주면

```
In [83]: index_df2.reset_index(drop=True)
```

Out[83]:

	col02	col03	col04
0	0.54	0.12	0.89
1	0.28	0.67	0.21
2	0.42	0.83	0.19
3	0.84	0.14	0.11
4	0.0	0.58	0.22

기존의 인덱스였던 것을 drop 하고 새로 만들어진다.

```
In [147]: titanic_reset_index_df[['pclass','fare']].head()
```

Out[147]:

	pclass	fare
0	1	53.1000
1	3	8.0500
2	3	8.4583
3	1	51.8625
4	3	21.0750

리스트로 결과를 볼 수 있고

In [151]:
```
titanic_reset_index_df[titanic_reset_index_df['pclass'] == 3].head()
```

Out[151]:

	index	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
1	4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
2	5	0	3	male	NaN	0	0	8.4583	Q	Third	man	True	NaN	Queenstown	no	True
4	7	0	3	male	2.0	3	1	21.0750	S	Third	child	False	NaN	Southampton	no	False
5	8	1	3	female	27.0	0	2	11.1333	S	Third	woman	False	NaN	Southampton	yes	False
7	10	1	3	female	4.0	1	1	16.7000	S	Third	child	False	G	Southampton	yes	False

boolean indexing을 써서 해당하는 값을 가져올 수 도 있다.

In [152]:
```
titanic_reset_index_df.iloc[0:7,2:4]
```

Out[152]:

	pclass	sex
0	1	female
1	3	male
2	3	male
3	1	male
4	3	male
5	3	female
6	2	female

In [153]:
```
titanic_reset_index_df.iloc[[4,6,8],[2,4,6]]
```

Out[153]:

	pclass	age	parch
4	3	2.0	1
6	2	14.0	0
8	1	58.0	0

다양한 방법으로 추출 할 수 있다.

ex)

In [158]:
```
# age >60 이상인 정보만 추출하고 싶다면?
titanic_reset_index_df[titanic_reset_index_df['age']>60].head()
```

Out[158]:

	index	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
30	33	0	2	male	66.0	0	0	10.5000	S	Second	man	True	NaN	Southampton	no	True
51	54	0	1	male	65.0	0	1	61.9792	C	First	man	True	B	Cherbourg	no	False
93	96	0	1	male	71.0	0	0	34.6542	C	First	man	True	A	Cherbourg	no	True
113	116	0	3	male	70.5	0	0	7.7500	Q	Third	man	True	NaN	Queenstown	no	True
167	170	0	1	male	61.0	0	0	33.5000	S	First	man	True	B	Southampton	no	True

In [159]:
```
# age > 60 이상인 pclass, survived, who 만 추출하고 싶다면?
titanic_reset_index_df[titanic_reset_index_df['age']>60][['pclass','survived','who']].head()
```

Out[159]:

	pclass	survived	who
30	2	0	man
51	1	0	man
93	1	0	man
113	3	0	man
167	1	0	man

-set_index

DataFrame.set_index('컬럼')

기존 행 인덱스를 제거하고 데이터 열 중 하나를 인덱스로 설정

```
In [75]: np.random.seed(100)
         index_df = pd.DataFrame(np.vstack([list('ABCDE'), # 이 프레임과
                                            np.round(np.random.rand(3,5),2)]).T, # 이 프레임을 vstack으로 붙여 줬다.
                                 columns=['col01','col02','col03','col04'])
```

```
In [77]: index_df
```

Out[77]:

	col01	col02	col03	col04
0	A	0.54	0.12	0.89
1	B	0.28	0.67	0.21
2	C	0.42	0.83	0.19
3	D	0.84	0.14	0.11
4	E	0.0	0.58	0.22

위와 같은 프레임이 있었을때 col01을 인덱스 값으로 변경 하기 위해서는

```
In [78]: index_df2 = index_df.set_index('col01')
```

```
In [79]: index_df2
```

Out[79]:

col01	col02	col03	col04
A	0.54	0.12	0.89
B	0.28	0.67	0.21
C	0.42	0.83	0.19
D	0.84	0.14	0.11
E	0.0	0.58	0.22

데이터 프레임.set_index('컬럼명') 을 통해 변경할 수 있다.

```
In [81]: index_df3 = index_df2.set_index('col02')
         index_df3
```

Out[81]:

col02	col03	col04
0.54	0.12	0.89
0.28	0.67	0.21
0.42	0.83	0.19
0.84	0.14	0.11
0.0	0.58	0.22

이렇게 된다.

sort_index

index를 정렬한다.

정렬을 하기 전에 일단 랜덤으로 값을 뽑아서 가져온다.

In [183]:
```
np.random.seed(100)
sort_df = pd.DataFrame(np.random.randint(0,10,(6,4)))
sort_df
```

Out[183]:

	0	1	2	3
0	8	8	3	7
1	7	0	4	2
2	5	2	2	2
3	1	0	8	4
4	0	9	6	2
5	4	1	5	3

In [185]:
```
sort_df.columns = ['A','B','C','D']
sort_df.index = pd.date_range('20201014',periods=6) # periods를 통해 날짜를 하나씩 늘린다.
sort_df
```

Out[185]:

	A	B	C	D
2020-10-14	8	8	3	7
2020-10-15	7	0	4	2
2020-10-16	5	2	2	2
2020-10-17	1	0	8	4
2020-10-18	0	9	6	2
2020-10-19	4	1	5	3

In [187]:
```
# 순열 랜덤 치환
random_date = np.random.permutation(sort_df.index) # sort_df.index에서 랜덤으로 뽑아서 쓴다.
random_date
```

Out[187]:
```
array(['2020-10-18T00:00:00.000000000', '2020-10-19T00:00:00.000000000',
       '2020-10-16T00:00:00.000000000', '2020-10-14T00:00:00.000000000',
       '2020-10-17T00:00:00.000000000', '2020-10-15T00:00:00.000000000'],
      dtype='datetime64[ns]')
```

In [188]:
```
sort_df2 = sort_df.reindex(index=random_date, columns = ['B','A','D','C'])
sort_df2
```

Out[188]:

	B	A	D	C
2020-10-18	9	0	2	6
2020-10-19	1	4	3	5
2020-10-16	2	5	2	2
2020-10-14	8	8	7	3
2020-10-17	0	1	4	8
2020-10-15	0	7	2	4

여기까지 랜덤으로 데이터프레임을 만들었다.

In [194]:
```
# axis = 0 : row , axis = 1 : col
sort_df2.sort_index(axis=1,ascending=False) # 컬럼의 인덱스를 sort한다.
```

Out[194]:

	D	C	B	A
2020-10-18	2	6	9	0
2020-10-19	3	5	1	4
2020-10-16	2	2	2	5
2020-10-14	7	3	8	8
2020-10-17	4	8	0	1
2020-10-15	2	4	0	7

sort_index를 통해 해당 인덱스를 sort 한다. 여기서는 axis =1은 col 즉 칼럼을 sort 하고 ascending=False로 하면 내림차순이 된다.

In [195]:
```
# axis = 0 : row , axis = 1 : col
sort_df2.sort_index(axis=0,ascending=False) # 행의 인덱스를 sort한다.
```

Out[195]:

	B	A	D	C
2020-10-19	1	4	3	5
2020-10-18	9	0	2	6
2020-10-17	0	1	4	8
2020-10-16	2	5	2	2
2020-10-15	0	7	2	4
2020-10-14	8	8	7	3

이렇게 axis=0으로 할 경우에는 행에 대한 sort 를 해준다.

단 하나의 컬럼을 sort 하기 위해서는

In [198]:
```
sort_df2.sort_index(by='B') # B의 컬럼를 sort한다.
```

```
C:₩Users₩hwang in beom₩Anaconda3₩lib₩site-packages₩ipykernel_launcher.py:1: FutureWarning: by argument to sort_index is deprecated, please use .sort_values(by=...)
  """Entry point for launching an IPython kernel.
```

Out[198]:

	B	A	D	C
2020-10-17	0	1	4	8
2020-10-15	0	7	2	4
2020-10-19	1	4	3	5
2020-10-16	2	5	2	2
2020-10-14	8	8	7	3
2020-10-18	9	0	2	6

웨이와 같이 by라고 하고 해당 칼럼을 지정해준다.

만약 같은 값에 대한 우선순위를 정하고 싶다면

```
In [199]: sort_df2.sort_index(by=['B','A']) # B의 컬럼를 sort한다.
```

```
C:₩Users₩hwang in beom₩Anaconda3₩lib₩site-packages₩ipykernel_launcher.py:1: FutureWarning: by argument to sort_index is deprecated, please use .sort_values(by=...)
  """Entry point for launching an IPython kernel.
```

Out[199]:

	B	A	D	C
2020-10-17	0	1	4	8
2020-10-15	0	7	2	4
2020-10-19	1	4	3	5
2020-10-16	2	5	2	2
2020-10-14	8	8	7	3
2020-10-18	9	0	2	6

이런 식으로 값을 두 개를 넣는다.

'Python' 카테고리의 다른 글□

[Python] Pandas 사용법 - 그룹화 및 그룹 함수 (groupby, qcut, cut, transfrom ...

[Python] Pandas 사용법 - 두가지의 DataFrame 합치기 (merge, join)□

[Python] Pandas 사용법 - 다양한 인덱스 함수(reset_index,set_index,sort_in...

[Python] Pandas 사용법 - 인덱싱 접근,데이터 조작, 인덱스조작(loc,iloc)□

[Python] Pandas 사용법 - 다양한 함수 사용(데이터 입출력, 대소문자변환, 공백...

[Python] Pandas 사용법 - DataFrame 생성, 추가 , 수정, 삭제, indexing□

reset_index set_index sort_index

나무늘보스

혼자 끄적끄적하는 블로그 입니다.