レシピ37 重複のないインデックスとソートしたインデックスによる選択（P97）

- **インデックス選択の処理速度は「重複なし・ソートあり」でさらに劇的に向上する**
- 前回のレシピ36ではそれらを行っていなかったため遅かった

＜point＞

- 「Booleanインデックス法」と「インデックス選択」は後者が高速
- 「重複なし、ソートあり」でさらに高速
- そのためには、ユニークなカラムをインデックスにセットする
- または複数のカラムを結合し条件にあったユニークなカラムをつくる
- `df.index.values`　インデックスの要素を取得
- `df.is_monotonic`　インデックスがソートされているか
- `df.index.is_unique`　インデックスがユニークかどうか
- `df.sort_index()`　インデックスをソートする
- `df.loc['col_A']` インデックス選択
- 出力されるオブジェクト
  - Booleanインデックス法：DataFrame
  - インデックス選択：Series

- このレシピでは、collegeデータセットを使って、重複なし・ソートありでインデックス選択を実行してみる
- Booleanインデックス法と比較する

（1）collegeデータセットを読み込む。STABBRをインデックスにセットする。

- **ソートされているかを確認する `df.is_monotonic`**

In [40]:
```
import pandas as pd
college = pd.read_csv('college.csv')

college2 = college.set_index('STABBR')
college2.index.is_monotonic
```

Out[40]: False

（2）college2のインデックスをソートし、別のオブジェクトとして保存

- インデックスがソートされていると、**pandasは二分探索を用いることができる**

In [31]:
```
college3 = college2.sort_index()
college3.index.is_monotonic
```

Out[31]: True

（3）これら3つのdfで「TX」州の選択を計測し、比較する

- **インデックス法（ソートあり）は劇速！** Booleanインデックス法の10倍以上、ソート無しの7倍

In [6]:
```
# Booleanインデックス法
%timeit college[college['STABBR'] == 'TX']
```

798 µs ± 74.1 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)

In [7]:
```
# インデックス法（ソートなし）
%timeit college2.loc['TX']
```

342 µs ± 2.98 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)

In [8]:
```
# インデックス法（ソートあり）
%timeit college3.loc['TX']
```

58.1 µs ± 553 ns per loop (mean ± std. dev. of 7 runs, 10000 loops each)

（4）インデックスに重複がない場合を調べる。校名をインデックスに使う

- **重複がないカラムをインデックスに用いる。インデックス選択はさらに高速になる**→（7）のそれ

In [32]:
```
college_unique = college.set_index('INSTNM')
college_unique.index.is_unique
```

Out[32]: True

（5）Booleanインデックス法で Stanford Universeを選ぶ

In [12]:
```
college[college['INSTNM'] == 'Stanford University']
```

Out[12]:

	INSTNM	CITY	STABBR	HBCU	MENONLY	WOMENONLY	RELAFFIL	SATVRMID	SATMTMID	DISTANCEON
4217	Stanford University	Stanford	CA	0.0	0.0	0.0	0	730.0	745.0	0

1 rows × 27 columns

（6）インデックス選択でも同じようにStanfordを選ぶ

In [13]:
```
college_unique.loc['Stanford University']
```

Out[13]:
```
CITY                Stanford
STABBR                    CA
HBCU                     0.0
MENONLY                  0.0
WOMENONLY                0.0
RELAFFIL                   0
SATVRMID                 730.0
SATMTMID                 745.0
DISTANCEONLY             0.0
UGDS                  7018.0
UGDS_WHITE            0.3752
UGDS_BLACK            0.0591
UGDS_HISP             0.1607
UGDS_ASIAN            0.1979
UGDS_AIAN             0.0114
UGDS_NHPI             0.0038
UGDS_2MOR             0.1067
UGDS_NRA              0.0819
UGDS_UNKN             0.0031
PPTUG_EF                 0.0
CURROPER                   1
PCTPELL               0.1556
PCTFLOAN              0.1256
UG25ABV               0.0401
MD_EARN_WNE_P10        86000
```

```
GRAD_DEBT_MDN_SUPP      12782
Name: Stanford University, dtype: object
```

（7）両方の結果は同じだが、出力のオブジェクトが異なる。それぞれを計測する

In [15]:
```
print(type(college[college['INSTNM'] == 'Stanford University']))
print(type(college_unique.loc['Stanford University']))
```

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

In [16]:
```
# Booleanインデックス法
%timeit college[college['INSTNM'] == 'Stanford University']
```

```
658 µs ± 12.5 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)
```

In [19]:
```
# インデックス法
%timeit college_unique.loc['Stanford University']
```

```
116 µs ± 3.63 µs per loop (mean ± std. dev. of 7 runs, 10000 loops each)
```

（補足）

- Boolean選択では、条件にいくつでもカラムを使えるのが利点。インデックス選択よりも柔軟性に富む
- 一方、インデックス法においても**複数カラムを連結してインデックスにすることも出来る**ため、ある程度の柔軟性はある
- 次のコードではインデックスを都市と州のカラムを連結している

In [42]:
```
college.index = college['CITY'] + ', ' + college['STABBR']
college = college.sort_index()
college.head()
```

Out[42]:

	INSTNM	CITY	STABBR	HBCU	MENONLY	WOMENONLY	RELAFFIL	SATVRMID	SATMTMID	DIST
ARTESIA, CA	Angeles Institute	ARTESIA	CA	0.0	0.0	0.0	0	NaN	NaN	
Aberdeen, SD	Presentation College	Aberdeen	SD	0.0	0.0	0.0	1	440.0	480.0	
Aberdeen, SD	Northern State University	Aberdeen	SD	0.0	0.0	0.0	0	480.0	475.0	
Aberdeen, WA	Grays Harbor College	Aberdeen	WA	0.0	0.0	0.0	0	NaN	NaN	
Abilene, TX	Hardin-Simmons University	Abilene	TX	0.0	0.0	0.0	1	508.0	515.0	

5 rows × 27 columns

- これにより、**Booleanインデックス法を使わずに**、指定した都市と州の組み合わせで大学を選べる

In [43]:
```
college.loc['Miami, FL'].head()
```

Out[43]:

	INSTNM	CITY	STABBR	HBCU	MENONLY	WOMENONLY	RELAFFIL	SATVRMID	SATMTMID	DISTANCEC
Miami, FL	New Professions Technical Institute	Miami	FL	0.0	0.0	0.0	0	NaN	NaN	

	INSTNM	CITY	STABBR	HBCU	MENONLY	WOMENONLY	RELAFFIL	SATVRMID	SATMTMID	DISTANCEC
Miami, FL	Management Resources College	Miami	FL	0.0	0.0	0.0	0	NaN	NaN	
Miami, FL	Strayer University-Doral	Miami	FL	NaN	NaN	NaN	1	NaN	NaN	
Miami, FL	Keiser University-Miami	Miami	FL	NaN	NaN	NaN	1	NaN	NaN	
Miami, FL	George T Baker Aviation Technical College	Miami	FL	0.0	0.0	0.0	0	NaN	NaN	

5 rows × 27 columns

- 複合インデックス法（上）と、Booleanインデックス法（下）の速度を比較すると、20倍以上インデックス法が速い

In [28]:
```
%%timeit
college.loc['Miami, FL']
```

61.1 µs ± 3 µs per loop (mean ± std. dev. of 7 runs, 10000 loops each)

In [44]:
```
%%timeit
crit1 = college['CITY'] == 'Miami'
crit2 = college['STABBR'] == 'FL'
college[crit1 & crit2]
```

1.56 ms ± 241 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)
