

# 第4回 Pythonによるデータ加工処理の基礎 (Pandas)

---

2025/10/28

## 自己紹介

嵐 大樹(あらし ひろき)



所属

奈良県立医科大学医学部医学科

GCI 2023 winter

受講歴

DL基礎講座 2024, LLM講座 2024

AIエンジニアリング実践講座 2025 他

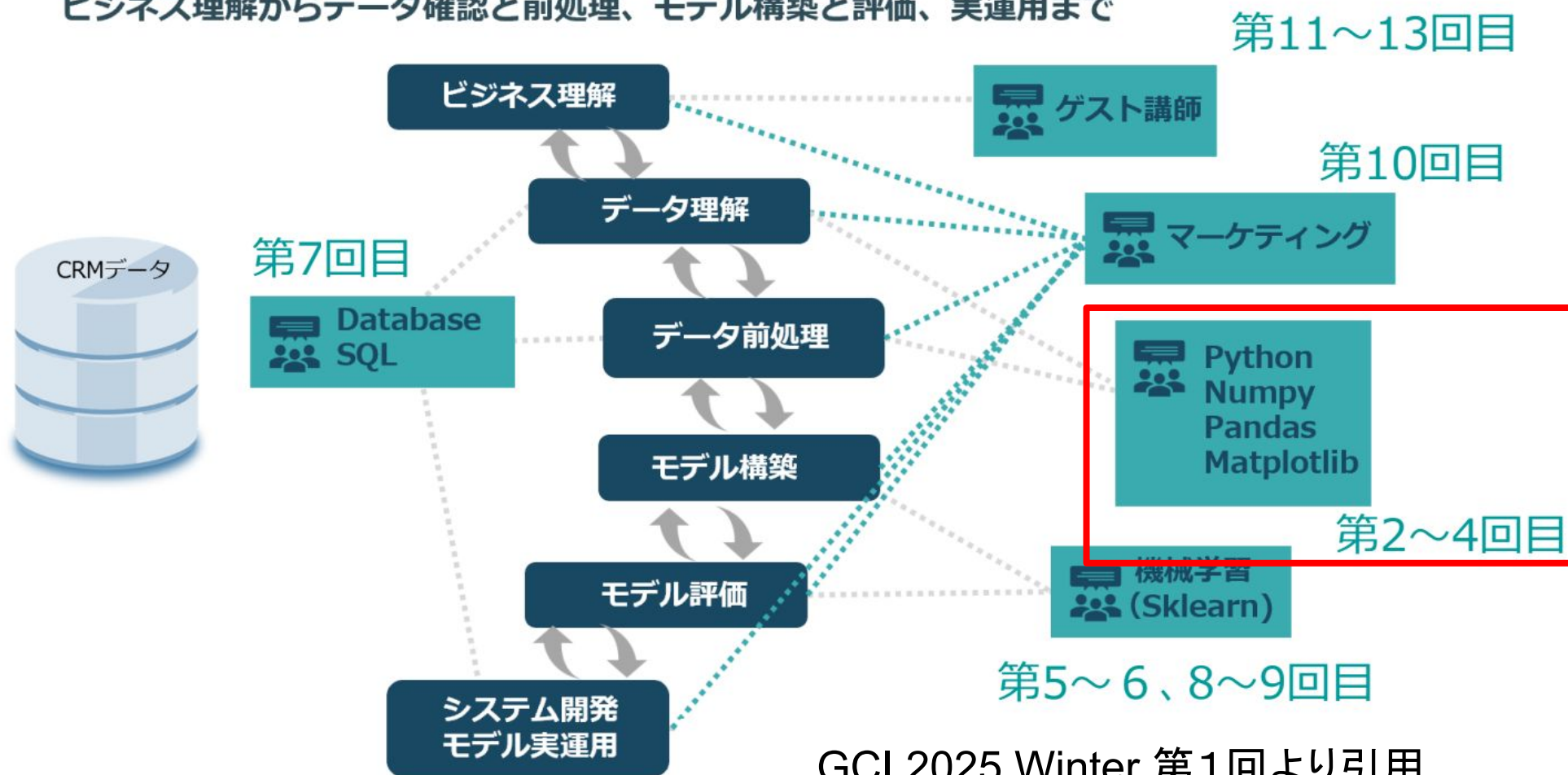
一言

私自身、受講生時代にこの回の講義で苦労した記憶があるので、その経験を活かした講義を心がけています。講義内容が身に付くまでには時間がかかりますが、一緒に頑張りましょう。

- イン트로ダクション
  - pandasの概要
- 演習形式の講義
  - データの読み込み
  - データの加工処理
- コンペ提出の操作デモ

## 実データサイエンスのプロジェクトと本講義の関係性

ビジネス理解からデータ確認と前処理、モデル構築と評価、実運用まで



GCI 2025 Winter 第1回より引用

- 主にDataFrame形式のデータの加工処理を行うためのライブラリ

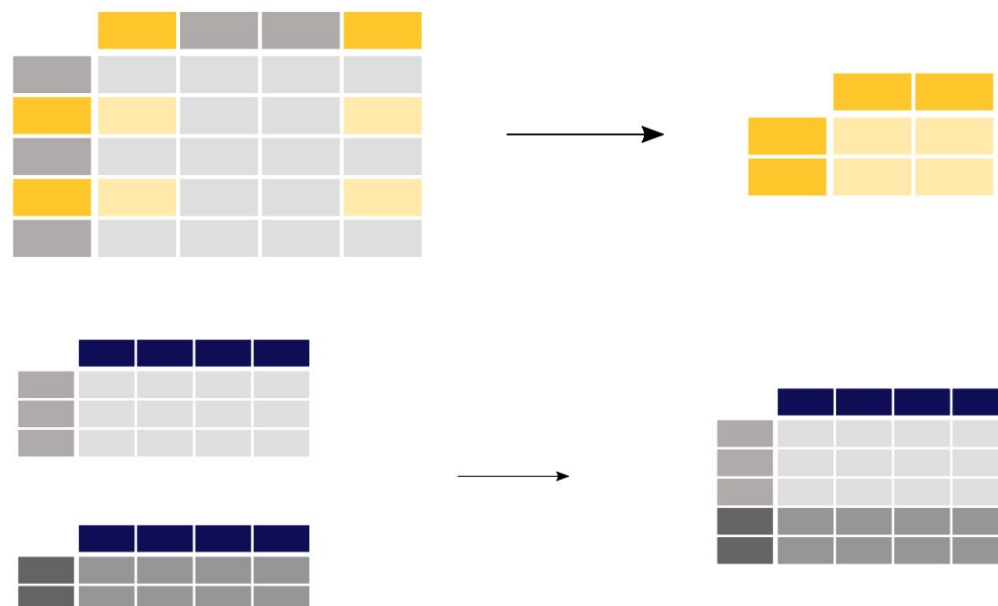


# DataFrame形式のデータとは？



- Excelのような二次元の表形式のデータ
- 例えば、様々な人について、それぞれの特徴をまとめたデータ

- 様々にあり、その一部を今回の講義で1つ1つ学びます
- 例えば、ある人たちのいくつかの特徴のデータだけを抜き出したり、いくつかのデータセットを結合したり、など

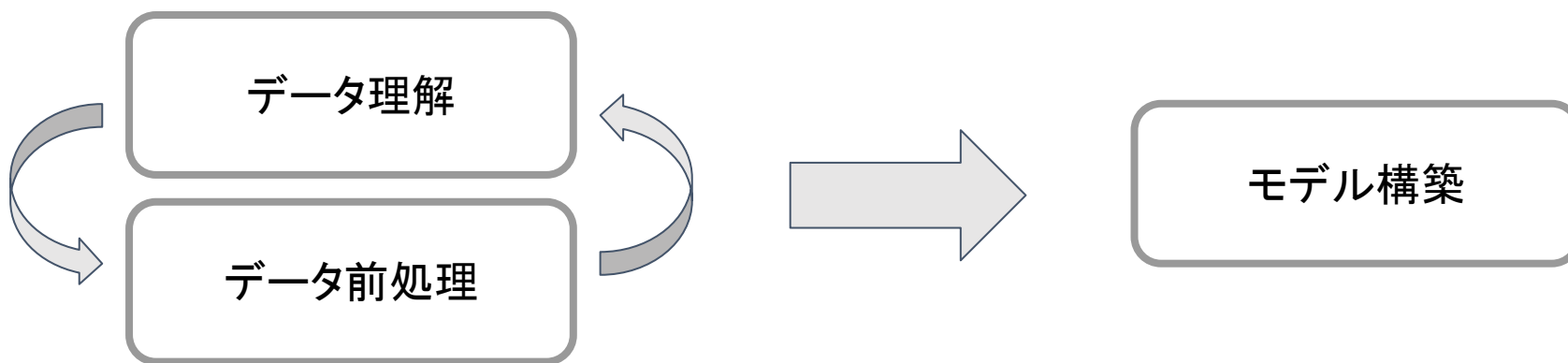


**DataFrame (2次元)**

|     |   | 1列目 | 2列目      | 3列目        | 4列目     |
|-----|---|-----|----------|------------|---------|
|     |   | ID  | City     | Birth_year | Name    |
| 1行目 | 0 | 100 | Tokyo    | 1990       | Hiroshi |
| 2行目 | 1 | 101 | Osaka    | 1989       | Akiko   |
| 3行目 | 2 | 102 | Kyoto    | 1992       | Yuki    |
| 4行目 | 3 | 103 | Hokkaido | 1997       | Satoru  |
| 5行目 | 4 | 104 | Tokyo    | 1982       | Steve   |

- AIモデルはデータを元に学習する
- AIモデルをより「賢く」するためには、よくデータを見て、よりよいデータに整えてから、AIモデルに与える必要がある

➡ よくデータを見る、よりよく整えるためには、加工処理が不可欠





| 加工処理の種類       | 実装のイメージ   | 重要度のイメージ |
|---------------|---|----------|
| ライブラリの読み込み    | <code>import ~~ as ~~</code>                                      | ☆☆☆      |
| データの読み込み      | <code>pd.read_csv()</code>  | ☆☆☆      |
| データの確認        | <code>df.head()</code>  | ☆☆☆      |
| データの概観        | <code>df.info()</code> , <code>df.describe()</code>               | ☆☆☆      |
| データの選択と代入     | <code>df.loc()</code> , <code>df.iloc</code> , <code>df.at</code> | ☆☆       |
| 特定の条件のデータの抽出  | <code>df[df['~~'] == ~~]</code>                                   | ☆☆☆      |
| 値のソート         | <code>df.sort_values()</code>                                     | ☆☆       |
| nan(null)の判定  | <code>df.isnull().sum()</code>                                    | ☆☆☆      |
| データの結合        | <code>pd.merge()</code> , <code>pd.concat()</code>                | ☆☆☆      |
| データの削除        | <code>df.drop()</code>  | ☆☆       |
| 重複データの除去      | <code>df.drop_duplicates()</code>                                 | ☆        |
| マッピング処理       | <code>df['~~'].map()</code>                                       | ☆☆       |
| ビン分割          | <code>pd.cut()</code>   | ☆☆       |
| データの集約とグループ演算 | <code>df.groupby('~~').agg()</code>                               | ☆☆       |
| 欠損データの扱い方     | <code>df.fillna()</code>  | ☆☆☆      |

- これだけの量を一度に全て理解して覚えることは(ほぼ)不可能です

➡ まずはざっくりと全体像を掴むことに注力してみてください

➡ 最終的に調べながら使えるようになることが目指すべきところです

- まずは一通りの情報を浴びるイメージで、一旦最後までついてきてください
- その上で、特によく使うものは「特に重要」(☆☆☆)と言うので、そこに関してはしっかり聞いてみてください(ただ、それも調べればOKです)

- pandasとは何かを説明できるようになる
- pandasを学ぶ意義について説明できるようになる
- pandasを用いて様々なデータの加工処理が可能となることを具体的な実感を持ってイメージできるようになる
- pandasを用いた特に重要な加工処理について、その実装方法を理解する
- コンペの予測結果を提出できるようになる

Pandasでよく使われるデータ構造は Series と DataFrame

## Series (1次元)

|     | ID    |
|-----|-------|
| 1行目 | 0 100 |
| 2行目 | 1 101 |
| 3行目 | 2 102 |
| 4行目 | 3 103 |
| 5行目 | 4 104 |

## DataFrame (2次元)

|     | 1列目   | 2列目      | 3列目        | 4列目     |
|-----|-------|----------|------------|---------|
|     | ID    | City     | Birth_year | Name    |
| 1行目 | 0 100 | Tokyo    | 1990       | Hiroshi |
| 2行目 | 1 101 | Osaka    | 1989       | Akiko   |
| 3行目 | 2 102 | Kyoto    | 1992       | Yuki    |
| 4行目 | 3 103 | Hokkaido | 1997       | Satoru  |
| 5行目 | 4 104 | Tokyo    | 1982       | Steve   |

DataFrameはNumPyの二次元配列に行ラベル・列ラベルがついたもの  
行ラベルを **index** ・列ラベルを **columns** という

## Series (1次元)

|              | ID            |
|--------------|---------------|
| 1行目          | 0 100         |
| 2行目          | 1 101         |
| 3行目          | 2 102         |
| 4行目          | 3 103         |
| 5行目          | 4 104         |
| <b>index</b> | <b>values</b> |

## DataFrame (2次元)

|              |   | 1列目 | 2列目      | 3列目        | 4列目     |                |
|--------------|---|-----|----------|------------|---------|----------------|
|              |   | ID  | City     | Birth_year | Name    | <b>columns</b> |
| 1行目          | 0 | 100 | Tokyo    | 1990       | Hiroshi |                |
| 2行目          | 1 | 101 | Osaka    | 1989       | Akiko   |                |
| 3行目          | 2 | 102 | Kyoto    | 1992       | Yuki    |                |
| 4行目          | 3 | 103 | Hokkaido | 1997       | Satoru  |                |
| 5行目          | 4 | 104 | Tokyo    | 1982       | Steve   |                |
| <b>index</b> |   |     |          |            |         | <b>values</b>  |

データを抽出する方法は、何を取得するか・何を指定するか によって使い分ける

Seriesや  
DataFrame  
を取得する

## ラベル指定

**loc** `df.loc[0:3, ['ID', 'Birth_year']]`

|   | ID  | City     | Birth_year | Name    | Score |
|---|-----|----------|------------|---------|-------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi | 0     |
| 1 | 101 | Osaka    | 1989       | Akiko   | 10    |
| 2 | 102 | Kyoto    | 1992       | Yuki    | 20    |
| 3 | 103 | Hokkaido | 1997       | Satoru  | 30    |
| 4 | 104 | Tokyo    | 1982       | Steve   | 40    |

## インデックス指定

**iloc** `df.iloc[0:4, [0,2]]`

|   | ID  | City     | Birth_year | Name    | Score |
|---|-----|----------|------------|---------|-------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi | 0     |
| 1 | 101 | Osaka    | 1989       | Akiko   | 10    |
| 2 | 102 | Kyoto    | 1992       | Yuki    | 20    |
| 3 | 103 | Hokkaido | 1997       | Satoru  | 30    |
| 4 | 104 | Tokyo    | 1982       | Steve   | 40    |

単独の要素  
を取得する

**at** `df.at[2, 'Birth_year']`

|   | ID  | City     | Birth_year | Name    | Score |
|---|-----|----------|------------|---------|-------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi | 0     |
| 1 | 101 | Osaka    | 1989       | Akiko   | 10    |
| 2 | 102 | Kyoto    | 1992       | Yuki    | 20    |
| 3 | 103 | Hokkaido | 1997       | Satoru  | 30    |
| 4 | 104 | Tokyo    | 1982       | Steve   | 40    |

**iat** `df.iat[2, 2]`

|   | ID  | City     | Birth_year | Name    | Score |
|---|-----|----------|------------|---------|-------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi | 0     |
| 1 | 101 | Osaka    | 1989       | Akiko   | 10    |
| 2 | 102 | Kyoto    | 1992       | Yuki    | 20    |
| 3 | 103 | Hokkaido | 1997       | Satoru  | 30    |
| 4 | 104 | Tokyo    | 1982       | Steve   | 40    |

# データの結合 (merge)

内部結合では、両方のデータにキーが存在する場合に結合 する

df1

|   | id  | city     | birth_year | name    |
|---|-----|----------|------------|---------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi |
| 1 | 101 | Osaka    | 1989       | Akiko   |
| 2 | 102 | Kyoto    | 1992       | Yuki    |
| 3 | 103 | Hokkaido | 1997       | Satoru  |
| 4 | 104 | Tokyo    | 1982       | Steeve  |
| 5 | 106 | Tokyo    | 1991       | Mituru  |
| 6 | 108 | Osaka    | 1988       | Aoi     |
| 7 | 110 | Kyoto    | 1990       | Tarou   |
| 8 | 111 | Hokkaido | 1995       | Suguru  |
| 9 | 113 | Tokyo    | 1981       | Mitsuo  |

df2

|   | id  | math | english | sex | index_num |
|---|-----|------|---------|-----|-----------|
| 0 | 100 | 50   | 90      | M   | 0         |
| 1 | 101 | 43   | 30      | F   | 1         |
| 2 | 102 | 33   | 20      | F   | 2         |
| 3 | 105 | 76   | 50      | M   | 3         |
| 4 | 107 | 98   | 30      | M   | 4         |

## 【キーの指定】

- on, left\_on, right\_on, left\_index, right\_index 引数で指定可能
- 指定なしの場合, 両方のデータに共通のカラムがキーとなる

id(キー)がdf1・df2ともに  
存在する行のみ結合される

pd.merge(df1, df2)

|   | id  | city  | birth_year | name    | math | english | sex | index_num |
|---|-----|-------|------------|---------|------|---------|-----|-----------|
| 0 | 100 | Tokyo | 1990       | Hiroshi | 50   | 90      | M   | 0         |
| 1 | 101 | Osaka | 1989       | Akiko   | 43   | 30      | F   | 1         |
| 2 | 102 | Kyoto | 1992       | Yuki    | 33   | 20      | F   | 2         |



# データの結合 (merge)

左外部結合では、左のデータのキーをもとに結合する

df1

|   | id  | city     | birth_year | name    |
|---|-----|----------|------------|---------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi |
| 1 | 101 | Osaka    | 1989       | Akiko   |
| 2 | 102 | Kyoto    | 1992       | Yuki    |
| 3 | 103 | Hokkaido | 1997       | Satoru  |
| 4 | 104 | Tokyo    | 1982       | Steeve  |
| 5 | 106 | Tokyo    | 1991       | Mituru  |
| 6 | 108 | Osaka    | 1988       | Aoi     |
| 7 | 110 | Kyoto    | 1990       | Tarou   |
| 8 | 111 | Hokkaido | 1995       | Suguru  |
| 9 | 113 | Tokyo    | 1981       | Mitsuo  |

df2

|   | id  | math | english | sex | index_num |
|---|-----|------|---------|-----|-----------|
| 0 | 100 | 50   | 90      | M   | 0         |
| 1 | 101 | 43   | 30      | F   | 1         |
| 2 | 102 | 33   | 20      | F   | 2         |
| 3 | 105 | 76   | 50      | M   | 3         |
| 4 | 107 | 98   | 30      | M   | 4         |

左のデータ(df1)の  
情報量は失われない

|   | id  | city     | birth_year | name    | math | english | sex | index_num |
|---|-----|----------|------------|---------|------|---------|-----|-----------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi | 50.0 | 90.0    | M   | 0.0       |
| 1 | 101 | Osaka    | 1989       | Akiko   | 43.0 | 30.0    | F   | 1.0       |
| 2 | 102 | Kyoto    | 1992       | Yuki    | 33.0 | 20.0    | F   | 2.0       |
| 3 | 103 | Hokkaido | 1997       | Satoru  | NaN  | NaN     | NaN | NaN       |
| 4 | 104 | Tokyo    | 1982       | Steeve  | NaN  | NaN     | NaN | NaN       |
| 5 | 106 | Tokyo    | 1991       | Mituru  | NaN  | NaN     | NaN | NaN       |
| 6 | 108 | Osaka    | 1988       | Aoi     | NaN  | NaN     | NaN | NaN       |
| 7 | 110 | Kyoto    | 1990       | Tarou   | NaN  | NaN     | NaN | NaN       |
| 8 | 111 | Hokkaido | 1995       | Suguru  | NaN  | NaN     | NaN | NaN       |
| 9 | 113 | Tokyo    | 1981       | Mitsuo  | NaN  | NaN     | NaN | NaN       |

`pd.merge(df1, df2, how = 'left')`



# データの結合 (merge)



完全外部結合 では, どちらかのデータにキーがあれば結合 する

|   | id  | city     | birth_year | name    |
|---|-----|----------|------------|---------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi |
| 1 | 101 | Osaka    | 1989       | Akiko   |
| 2 | 102 | Kyoto    | 1992       | Yuki    |
| 3 | 103 | Hokkaido | 1997       | Satoru  |
| 4 | 104 | Tokyo    | 1982       | Steeve  |
| 5 | 106 | Tokyo    | 1991       | Mituru  |
| 6 | 108 | Osaka    | 1988       | Aoi     |
| 7 | 110 | Kyoto    | 1990       | Tarou   |
| 8 | 111 | Hokkaido | 1995       | Suguru  |
| 9 | 113 | Tokyo    | 1981       | Mitsuo  |

|   | id  | math | english | sex | index_num |
|---|-----|------|---------|-----|-----------|
| 0 | 100 | 50   | 90      | M   | 0         |
| 1 | 101 | 43   | 30      | F   | 1         |
| 2 | 102 | 33   | 20      | F   | 2         |
| 3 | 105 | 76   | 50      | M   | 3         |
| 4 | 107 | 98   | 30      | M   | 4         |

どちらのデータも  
情報量は失われない

|    | id  | city     | birth_year | name    | math | english | sex | index_num |
|----|-----|----------|------------|---------|------|---------|-----|-----------|
| 0  | 100 | Tokyo    | 1990.0     | Hiroshi | 50.0 | 90.0    | M   | 0.0       |
| 1  | 101 | Osaka    | 1989.0     | Akiko   | 43.0 | 30.0    | F   | 1.0       |
| 2  | 102 | Kyoto    | 1992.0     | Yuki    | 33.0 | 20.0    | F   | 2.0       |
| 3  | 103 | Hokkaido | 1997.0     | Satoru  | NaN  | NaN     | NaN | NaN       |
| 4  | 104 | Tokyo    | 1982.0     | Steeve  | NaN  | NaN     | NaN | NaN       |
| 5  | 106 | Tokyo    | 1991.0     | Mituru  | NaN  | NaN     | NaN | NaN       |
| 6  | 108 | Osaka    | 1988.0     | Aoi     | NaN  | NaN     | NaN | NaN       |
| 7  | 110 | Kyoto    | 1990.0     | Tarou   | NaN  | NaN     | NaN | NaN       |
| 8  | 111 | Hokkaido | 1995.0     | Suguru  | NaN  | NaN     | NaN | NaN       |
| 9  | 113 | Tokyo    | 1981.0     | Mitsuo  | NaN  | NaN     | NaN | NaN       |
| 10 | 105 | NaN      | NaN        | NaN     | 76.0 | 50.0    | M   | 3.0       |
| 11 | 107 | NaN      | NaN        | NaN     | 98.0 | 30.0    | M   | 4.0       |

`pd.merge(df1, df2, how = 'outer')`

# データの結合 (concat)



縦結合では、キーを指定せずにデータを積み上げて結合 する

|   | id  | city     | birth_year | name    |
|---|-----|----------|------------|---------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi |
| 1 | 101 | Osaka    | 1989       | Akiko   |
| 2 | 102 | Kyoto    | 1992       | Yuki    |
| 3 | 103 | Hokkaido | 1997       | Satoru  |
| 4 | 104 | Tokyo    | 1982       | Steeve  |
| 5 | 106 | Tokyo    | 1991       | Mituru  |
| 6 | 108 | Osaka    | 1988       | Aoi     |
| 7 | 110 | Kyoto    | 1990       | Tarou   |
| 8 | 111 | Hokkaido | 1995       | Suguru  |
| 9 | 113 | Tokyo    | 1981       | Mitsuo  |

|   | id  | city     | birth_year | name    |
|---|-----|----------|------------|---------|
| 0 | 117 | Chiba    | 1990       | Suguru  |
| 1 | 118 | Kanagawa | 1989       | Kouichi |
| 2 | 119 | Tokyo    | 1992       | Satochi |
| 3 | 120 | Fukuoka  | 1997       | Yukie   |
| 4 | 125 | Okinawa  | 1982       | Akari   |

|   | id  | city     | birth_year | name    |
|---|-----|----------|------------|---------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi |
| 1 | 101 | Osaka    | 1989       | Akiko   |
| 2 | 102 | Kyoto    | 1992       | Yuki    |
| 3 | 103 | Hokkaido | 1997       | Satoru  |
| 4 | 104 | Tokyo    | 1982       | Steeve  |
| 5 | 106 | Tokyo    | 1991       | Mituru  |
| 6 | 108 | Osaka    | 1988       | Aoi     |
| 7 | 110 | Kyoto    | 1990       | Tarou   |
| 8 | 111 | Hokkaido | 1995       | Suguru  |
| 9 | 113 | Tokyo    | 1981       | Mitsuo  |
| 0 | 117 | Chiba    | 1990       | Suguru  |
| 1 | 118 | Kanagawa | 1989       | Kouichi |
| 2 | 119 | Tokyo    | 1992       | Satochi |
| 3 | 120 | Fukuoka  | 1997       | Yukie   |
| 4 | 125 | Okinawa  | 1982       | Akari   |

共通カラムで  
積み上げる

`pd.concat([df1,df3])`

# データの結合 (concat)



横結合では, カラムを無視してインデックスをもとに結合 する

|   | id  | city     | birth_year | name    |
|---|-----|----------|------------|---------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi |
| 1 | 101 | Osaka    | 1989       | Akiko   |
| 2 | 102 | Kyoto    | 1992       | Yuki    |
| 3 | 103 | Hokkaido | 1997       | Satoru  |
| 4 | 104 | Tokyo    | 1982       | Steeve  |
| 5 | 106 | Tokyo    | 1991       | Mituru  |
| 6 | 108 | Osaka    | 1988       | Aoi     |
| 7 | 110 | Kyoto    | 1990       | Tarou   |
| 8 | 111 | Hokkaido | 1995       | Suguru  |
| 9 | 113 | Tokyo    | 1981       | Mitsuo  |

|   | id  | math | english | sex | index_num |
|---|-----|------|---------|-----|-----------|
| 0 | 100 | 50   | 90      | M   | 0         |
| 1 | 101 | 43   | 30      | F   | 1         |
| 2 | 102 | 33   | 20      | F   | 2         |
| 3 | 105 | 76   | 50      | M   | 3         |
| 4 | 107 | 98   | 30      | M   | 4         |

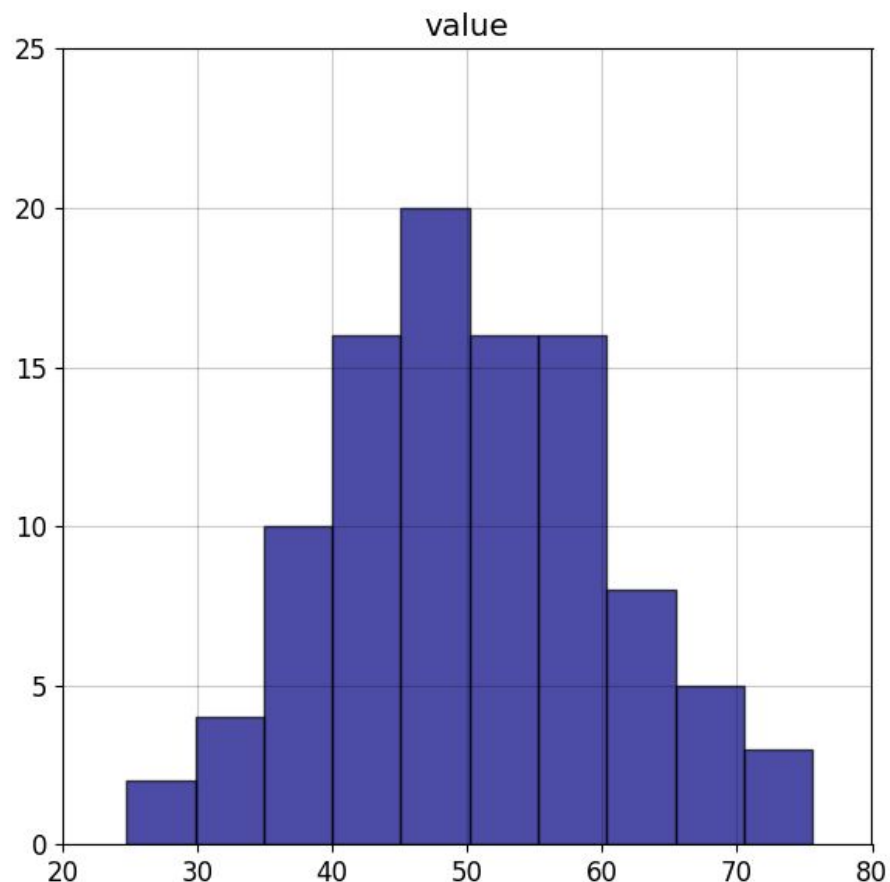
不足しているデータは  
欠損値となる

|   | id  | city     | birth_year | name    | id  | math | english | sex | index_num |
|---|-----|----------|------------|---------|-----|------|---------|-----|-----------|
| 0 | 100 | Tokyo    | 1990       | Hiroshi | 100 | 50.0 | 90.0    | M   | 0.0       |
| 1 | 101 | Osaka    | 1989       | Akiko   | 101 | 43.0 | 30.0    | F   | 1.0       |
| 2 | 102 | Kyoto    | 1992       | Yuki    | 102 | 33.0 | 20.0    | F   | 2.0       |
| 3 | 103 | Hokkaido | 1997       | Satoru  | 105 | 76.0 | 50.0    | M   | 3.0       |
| 4 | 104 | Tokyo    | 1982       | Steeve  | 107 | 98.0 | 30.0    | M   | 4.0       |
| 5 | 106 | Tokyo    | 1991       | Mituru  | NaN | NaN  | NaN     | NaN | NaN       |
| 6 | 108 | Osaka    | 1988       | Aoi     | NaN | NaN  | NaN     | NaN | NaN       |
| 7 | 110 | Kyoto    | 1990       | Tarou   | NaN | NaN  | NaN     | NaN | NaN       |
| 8 | 111 | Hokkaido | 1995       | Suguru  | NaN | NaN  | NaN     | NaN | NaN       |
| 9 | 113 | Tokyo    | 1981       | Mitsuo  | NaN | NaN  | NaN     | NaN | NaN       |

`pd.concat([df1, df2], axis=1)`

ビン分割を行うときは、データを等間隔または等個数で分割 する

```
data = DataFrame({"value":np.random.normal(50,10,100)}) # 正規分布に従う乱数
```



**cut(): データを 等間隔で分割**

```
pd.cut(data["value"], bins=5)
```

|       | (24. 702,<br>34. 927] | (34. 927,<br>45. 101] | (45. 101,<br>55. 275] | (55. 275,<br>65. 449] | (65. 449,<br>75. 624] |
|-------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|
| count | 6                     | 26                    | 36                    | 24                    | 8                     |

**qcut(): データを 等個数で分割**

```
pd.qcut(data["value"], q=5)
```

|       | (24. 752,<br>42. 199] | (42. 199,<br>46. 881] | (46. 881,<br>53. 138] | (53. 138,<br>58. 002] | (58. 002,<br>75. 624] |
|-------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|
| count | 20                    | 20                    | 20                    | 20                    | 20                    |

欠損値や異常値は、なんらかの方法で**処理する必要**がある

**欠損値**: 何らかの理由で欠損している値

- ・列や行ごと除去する
- ・別の値で置き換える

**異常値**: 他の値から大きくかけ離れた値

- ・列や行ごと除去する
- ・別の値で置き換える

|   | ID  | City  | Birth_year | Name    |
|---|-----|-------|------------|---------|
| 0 | 100 | Tokyo | 3000       | Hiroshi |
| 1 | 101 | Osaka | 1989       | Akiko   |
| 2 | 102 | Kyoto | 1992       | Yuki    |
| 3 | 103 | NaN   | 1997       | Satoru  |
| 4 | 104 | Tokyo | 1982       | Steve   |



松尾・岩澤研究室

MATSUO-IWASAWA LAB UTOKYO

# 休憩中

# 19:40に再開します