Lovely Pandas

Contents

- Datasets
  - Data representation
  - Types of variables
- Pandas for Data Analysis
  - The pandas.Series data structure
  - The pandas.DataFrame data structure
  - Read data from files
- Basics of Descriptive Analytics
  - Descriptive measures
  - Visualizing data

Datasets

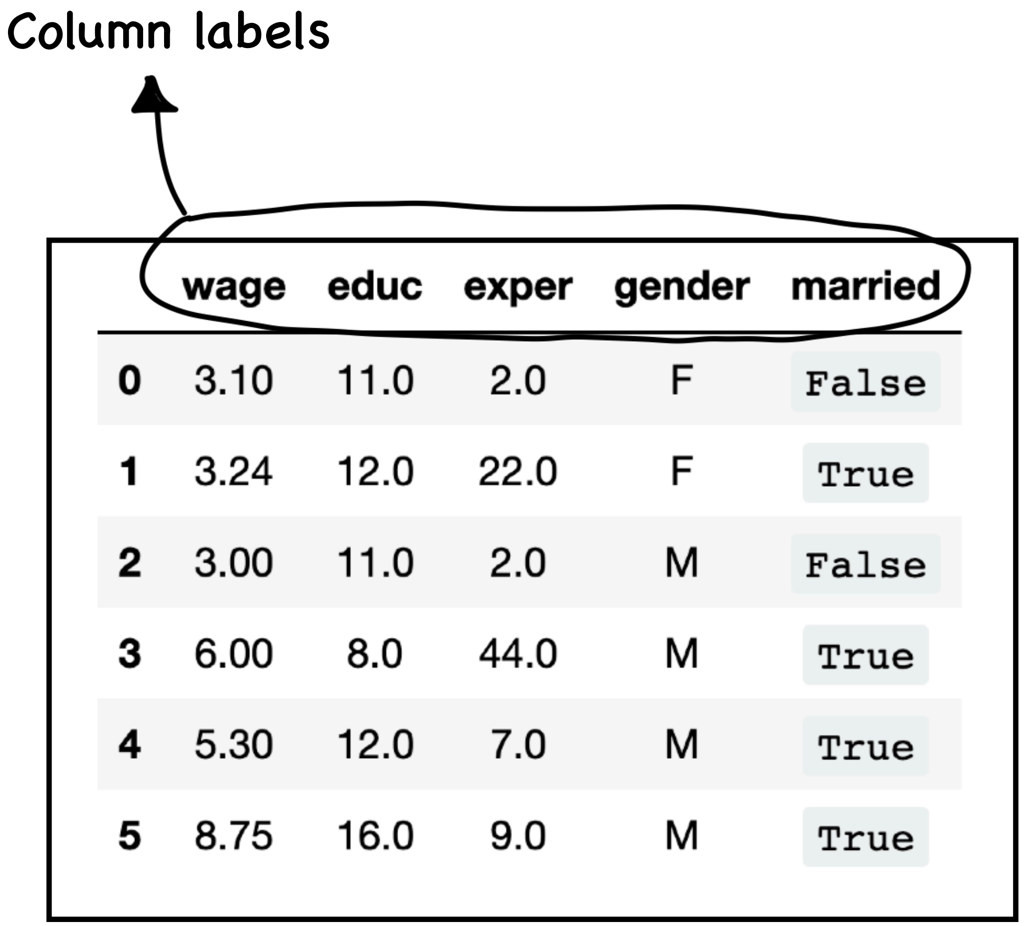
- Data representation

Example 1: The following table contains a cross-sectional dataset on a number of working individuals for the year 1976. Columns of the data table are summarized as follows.

- **wage**: average hourly earnings (in dollars)
- **educ**: years of education
- **exper**: years of potential experience
- **gender**: genders of these working individuals
- **married**: `True` if married, and `False` otherwise

Datasets

- Data representation
  - Variables (fields/attributes) as columns

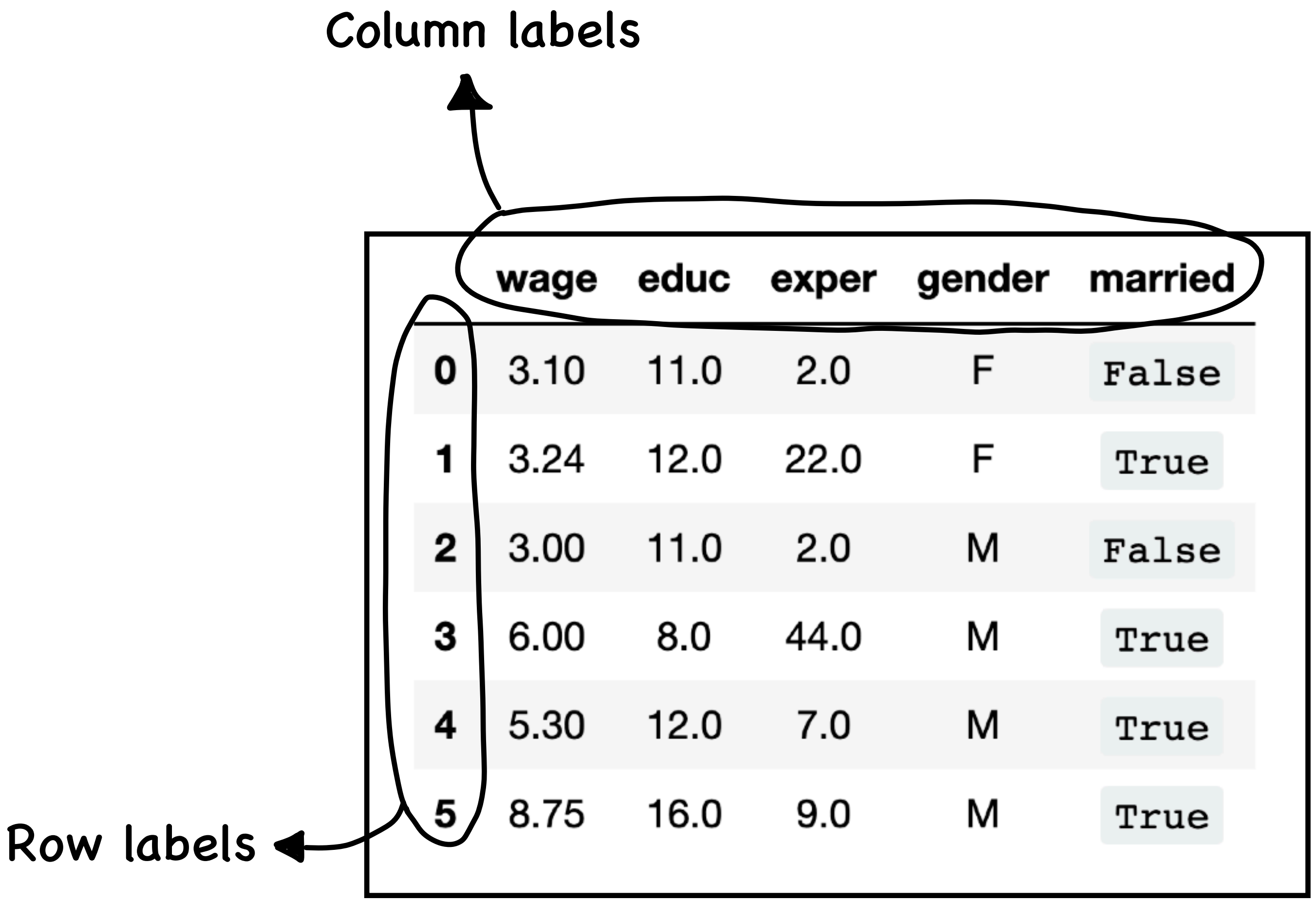


Column labels

	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Datasets

- Data representation
  - Variables (fields/attributes) as columns
  - Observations (cases/records) as rows



Column labels
Row labels

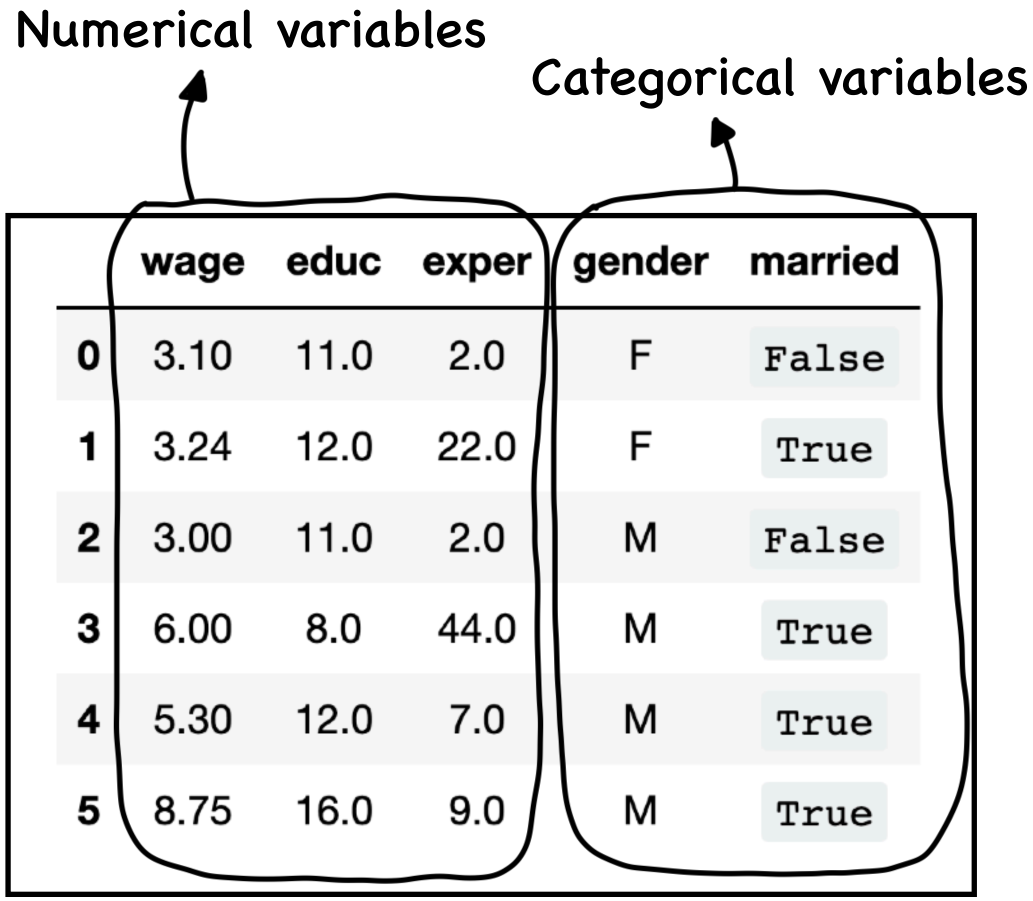
	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Datasets

- Data representation
  - Types of variables
    - ✓ Numerical (quantitative) variables
    - ✓ Categorical (qualitative) variables

```python
print(int(True))
print(int(False))
```

```
1
0
```



Numerical variables

Categorical variables

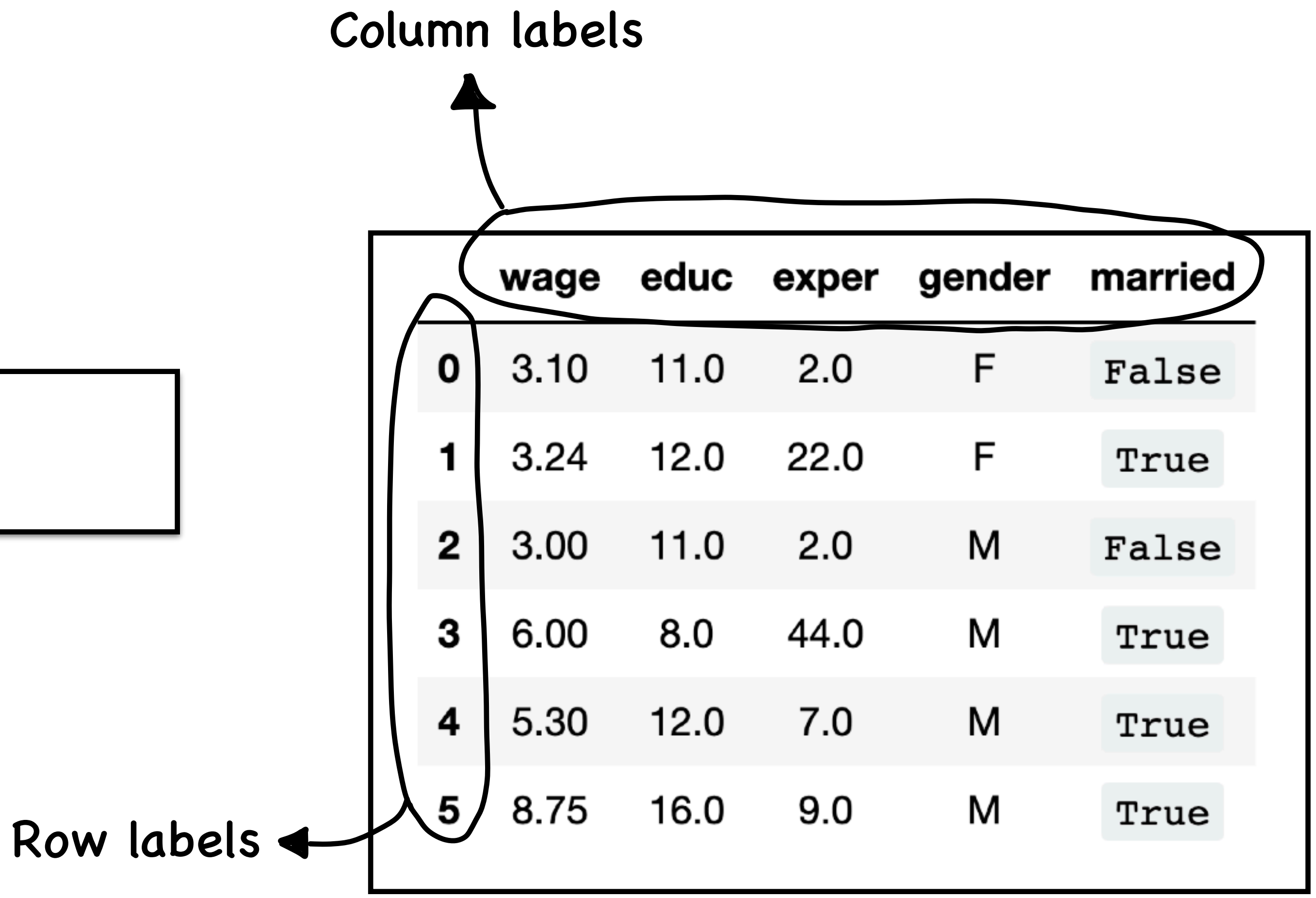
	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- Introduction to Pandas
  - Labeled data
  - Heterogenous data
  - Possible missing values

- Import Pandas

```python
import pandas as pd
```



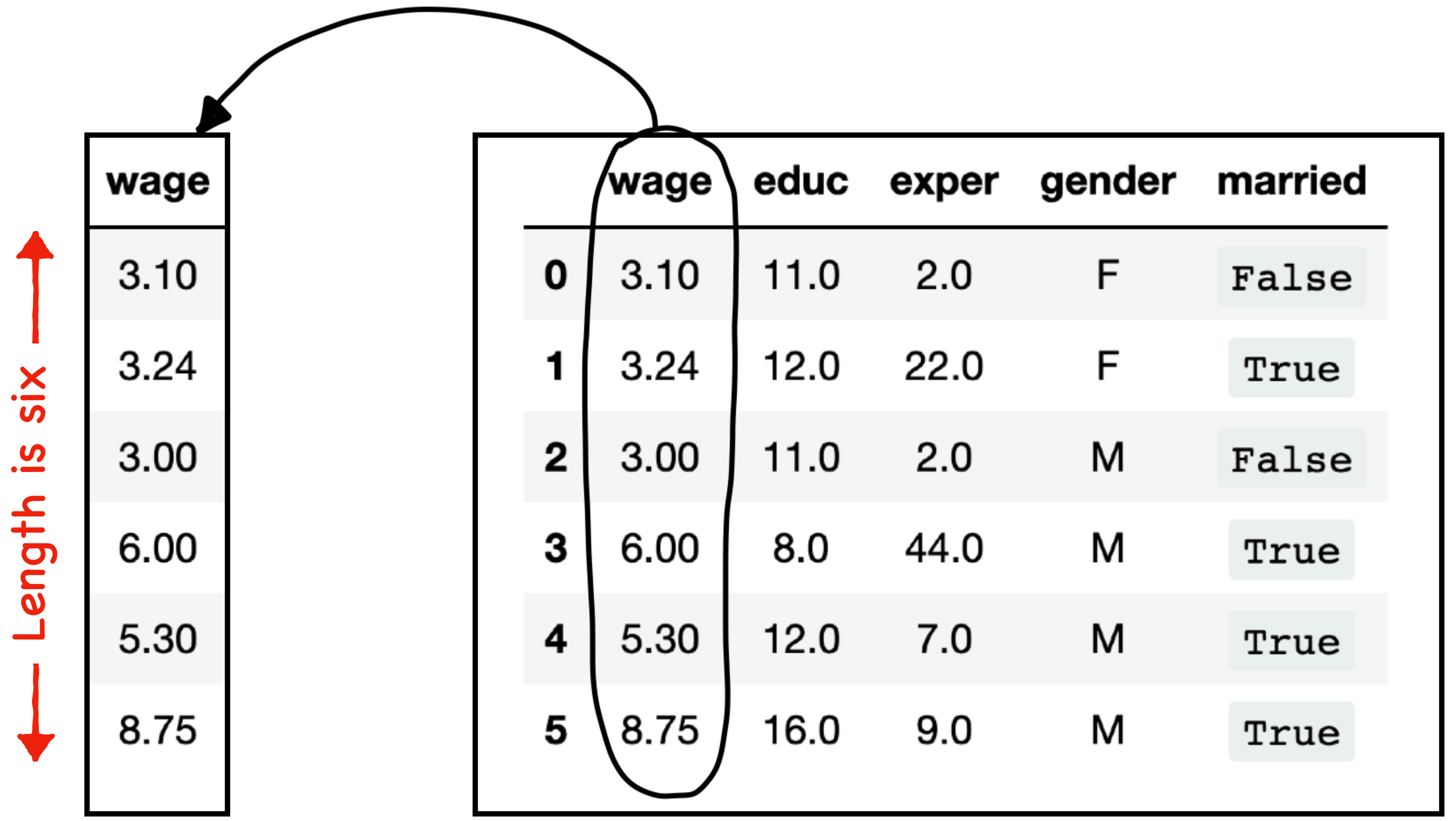
Column labels
Row labels

	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.Series` data structure
  - A one-dimensional array of indexed data

One-dimensional array

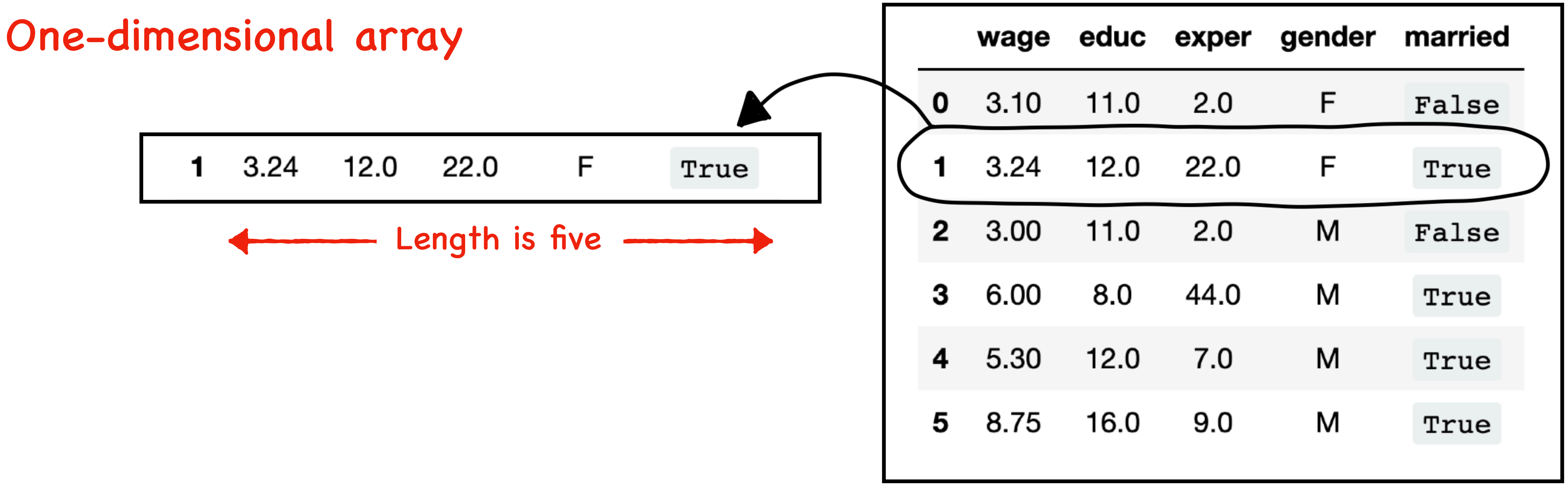


wage
3.10
3.24
3.00
6.00
5.30
8.75

	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.Series` data structure
  - A one-dimensional array of indexed data



One-dimensional array

1	3.24	12.0	22.0	F	True

Length is five

	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.Series` data structure
  - A one-dimensional array of indexed data
  - Create a `pandas.Series` type object

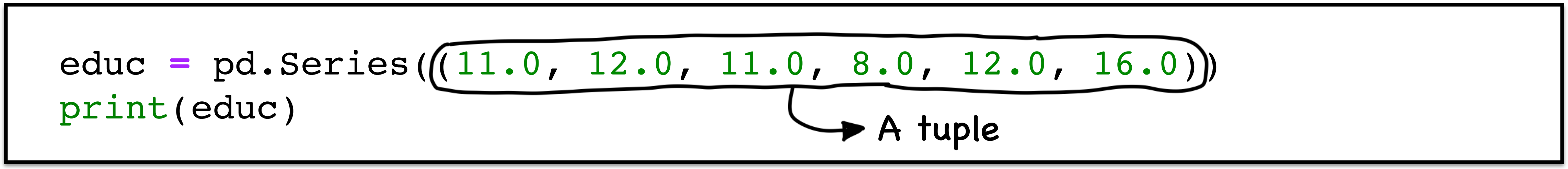
```
wage = pd.Series([3.10, 3.24, 3.00, 6.00, 5.30, 8.75])
print(wage)
```

A list

```
0     3.10
1     3.24
2     3.00
3     6.00
4     5.30
5     8.75
dtype: float64
```

Pandas for Data Analysis

- The `pandas.Series` data structure
  - A one-dimensional array of indexed data
  - Create a `pandas.Series` type object



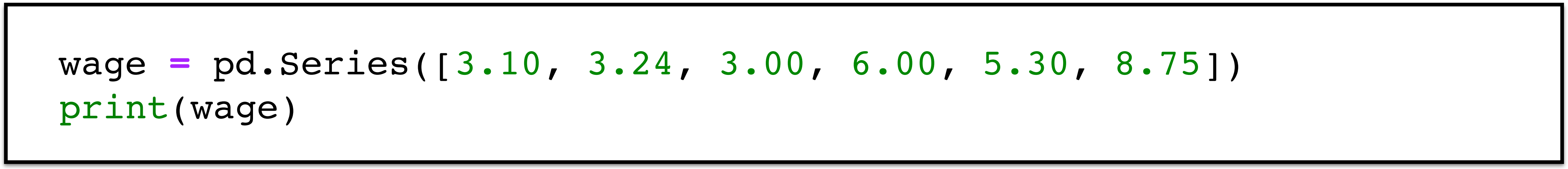
```
educ = pd.Series((11.0, 12.0, 11.0, 8.0, 12.0, 16.0))
print(educ)
```

A tuple

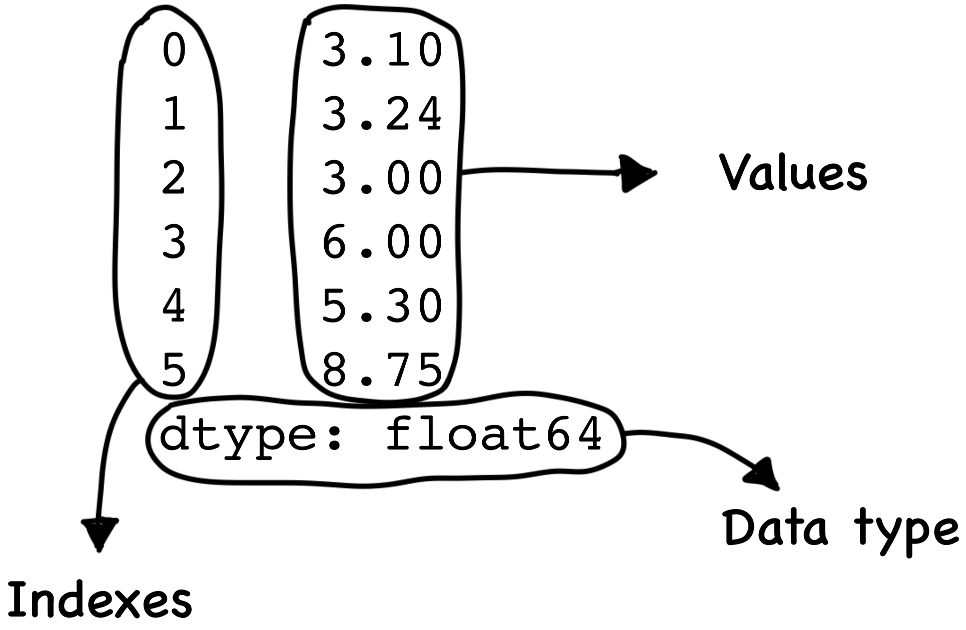
```
0     11.0
1     12.0
2     11.0
3      8.0
4     12.0
5     16.0
dtype: float64
```

Pandas for Data Analysis

- The `pandas.Series` data structure
  - Attributes of series



```
wage = pd.Series([3.10, 3.24, 3.00, 6.00, 5.30, 8.75])
print(wage)
```



```
0    3.10
1    3.24
2    3.00
3    6.00
4    5.30
5    8.75
dtype: float64
```

Attributes of Charmander

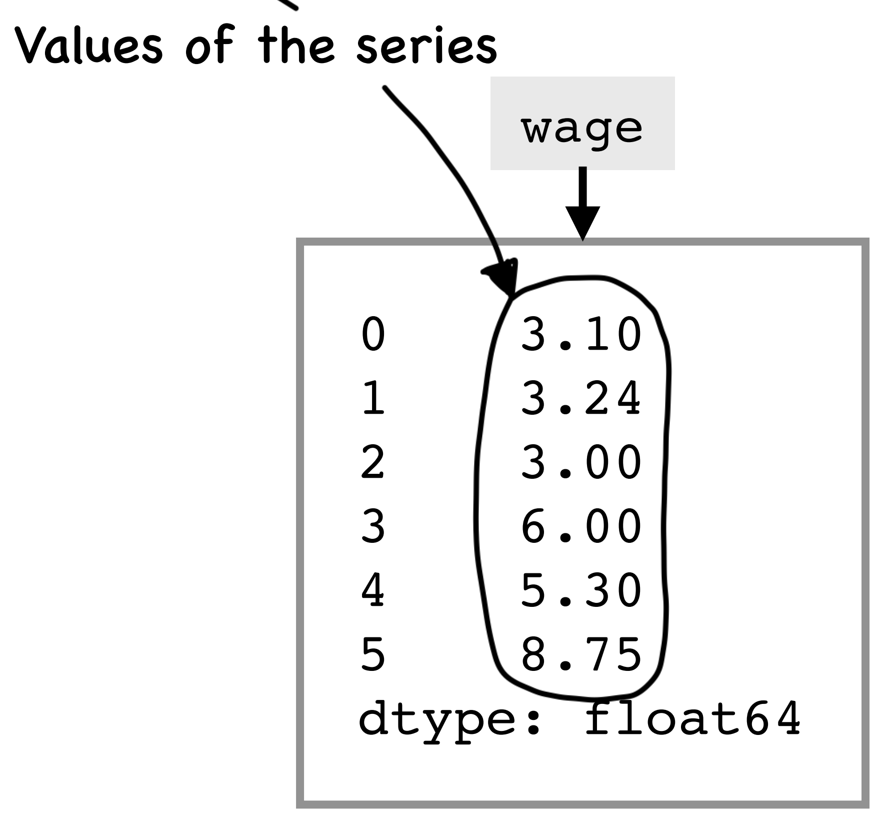
HP:	39	🟥
Attack:	52	🟥
Defense:	43	🟥
Sp. Atk:	50	🟥
Sp. Def:	50	🟥
Speed:	65	🟧

Pandas for Data Analysis

- The `pandas.Series` data structure
  - Attributes of series

```
wage.values
```

```
array([3.1 , 3.24, 3.  , 6.  , 5.3 , 8.75])
```



Values of the series

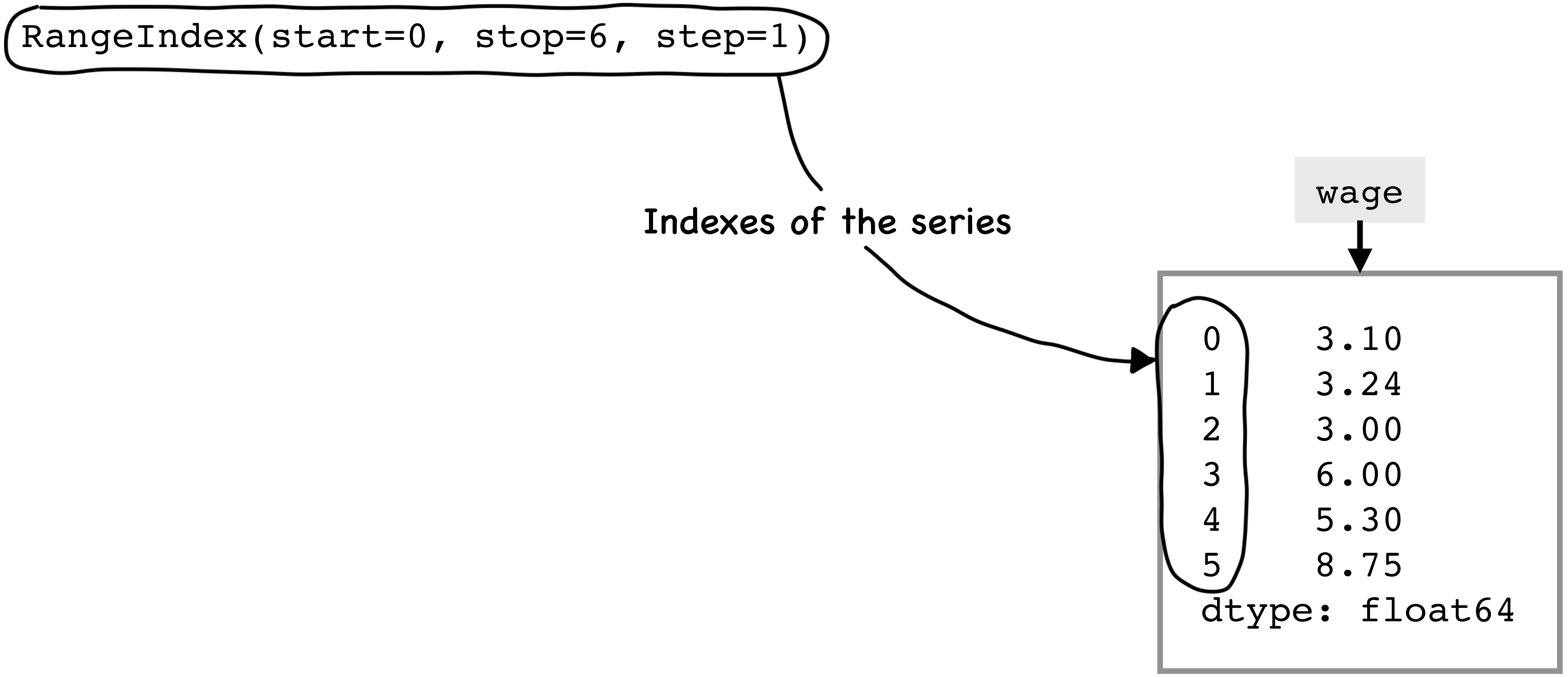
wage

```
0     3.10
1     3.24
2     3.00
3     6.00
4     5.30
5     8.75
dtype: float64
```

Pandas for Data Analysis

- The `pandas.Series` data structure
  - Attributes of series

```
wage.index
```



```
RangeIndex(start=0, stop=6, step=1)
```

Indexes of the series

wage

```
0     3.10
1     3.24
2     3.00
3     6.00
4     5.30
5     8.75
dtype: float64
```

Pandas for Data Analysis

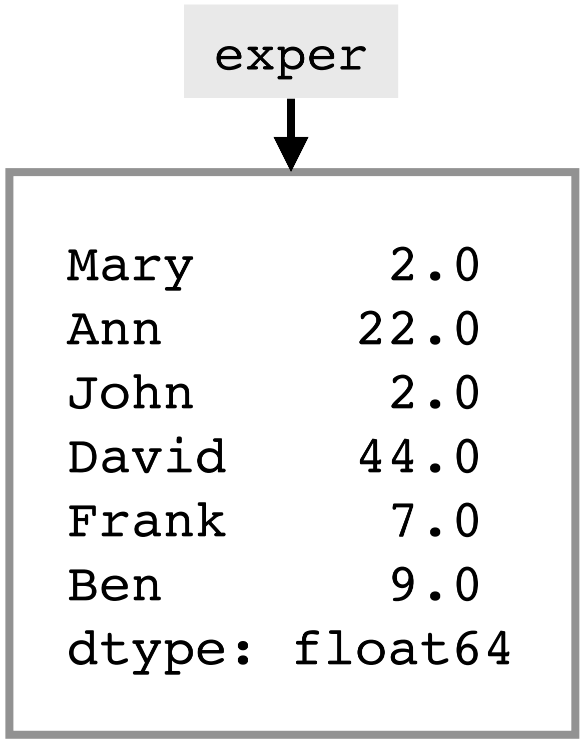
- The `pandas.Series` data structure
    - Attributes of series

```
index = ['Mary', 'Ann', 'John', 'David', 'Frank', 'Ben']
exper = pd.Series([2.0, 22.0, 2.0, 44.0, 7.0, 9.0],
                  index=index)

print(exper)
```

Specify indexes via the keyword argument

```
Mary      2.0
Ann      22.0
John      2.0
David    44.0
Frank     7.0
Ben       9.0
dtype: float64
```



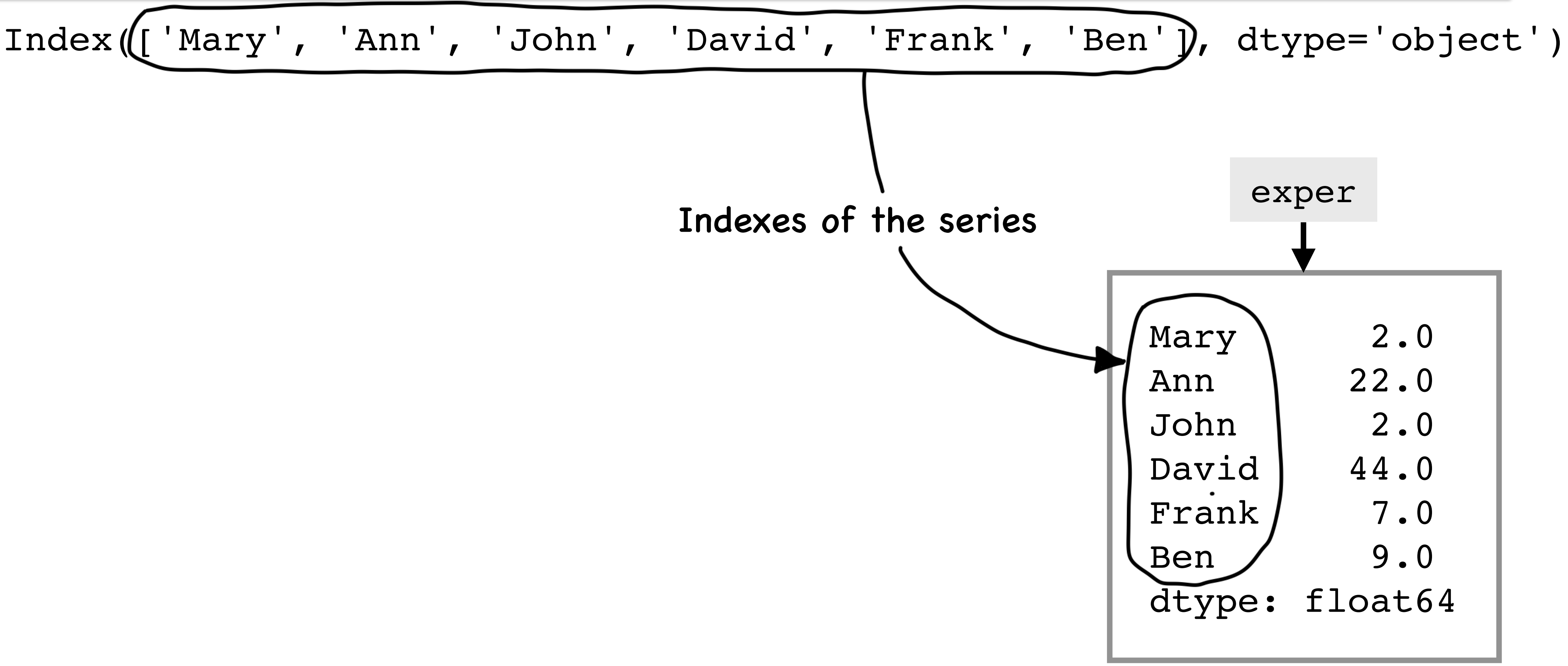
exper

```
Mary      2.0
Ann      22.0
John      2.0
David    44.0
Frank     7.0
Ben       9.0
dtype: float64
```

Pandas for Data Analysis

- The `pandas.Series` data structure
    - Attributes of series

```python
print(exper.index)
```



```
Index(['Mary', 'Ann', 'John', 'David', 'Frank', 'Ben'], dtype='object')
```

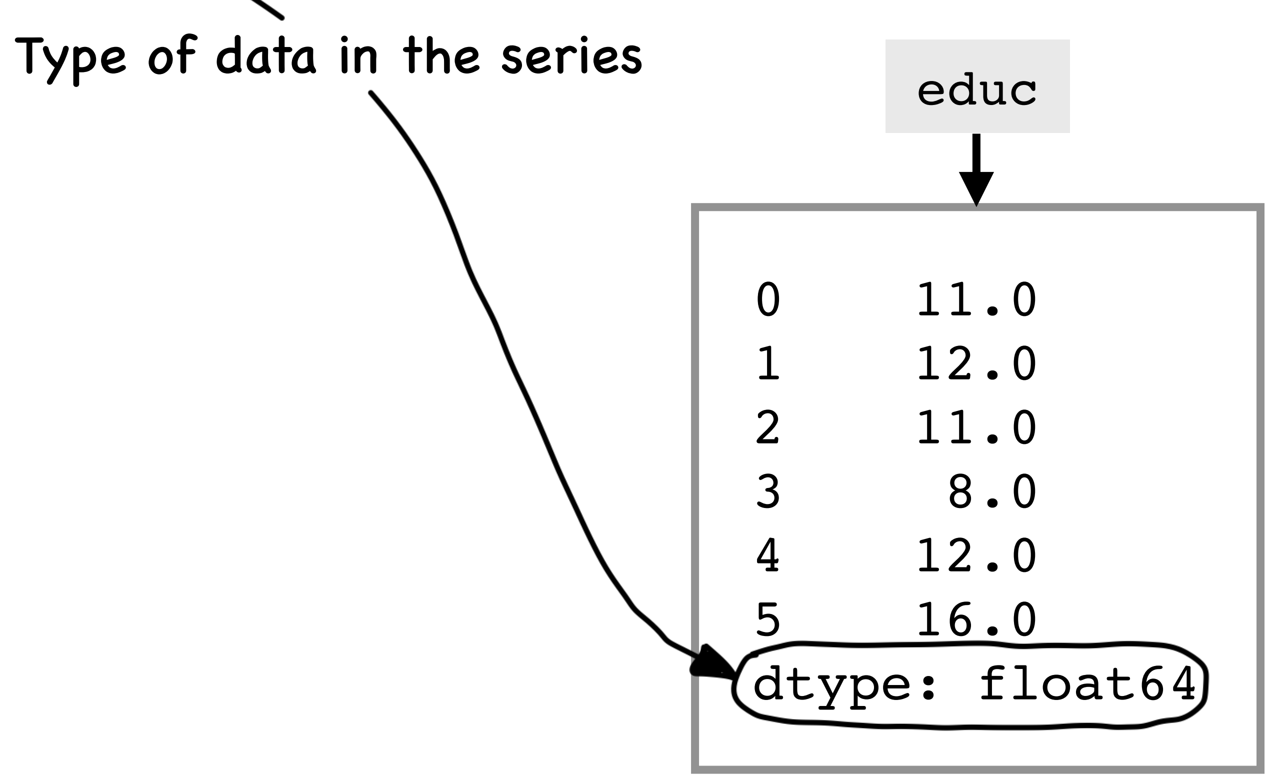
```
Mary      2.0
Ann      22.0
John      2.0
David    44.0
Frank     7.0
Ben       9.0
dtype: float64
```

Pandas for Data Analysis

- The `pandas.Series` data structure
  - Attributes of series

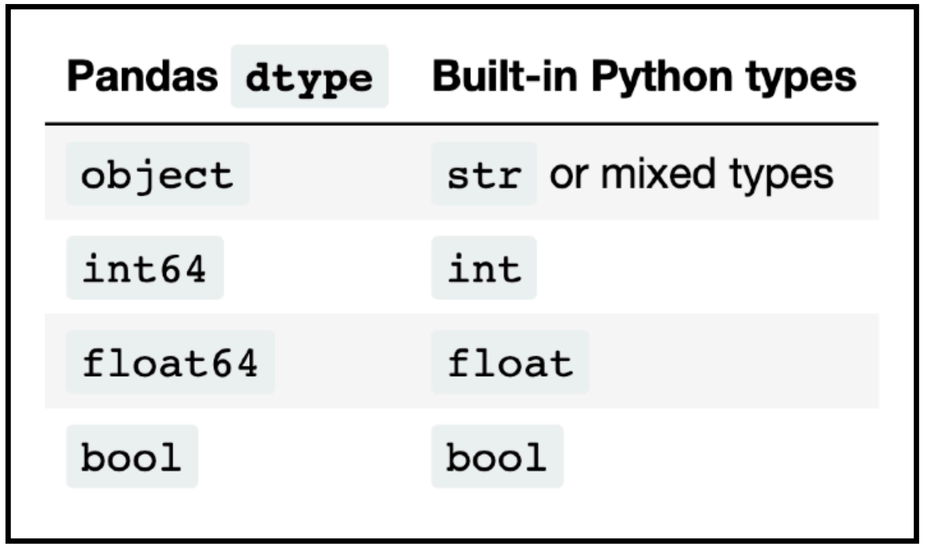
```python
print(educ.dtype)
```

```
float64
```



Type of data in the series

```
0      11.0
1      12.0
2      11.0
3       8.0
4      12.0
5      16.0
dtype: float64
```



Pandas `dtype`	Built-in Python types
`object`	`str` or mixed types
`int64`	`int`
`float64`	`float`
`bool`	`bool`

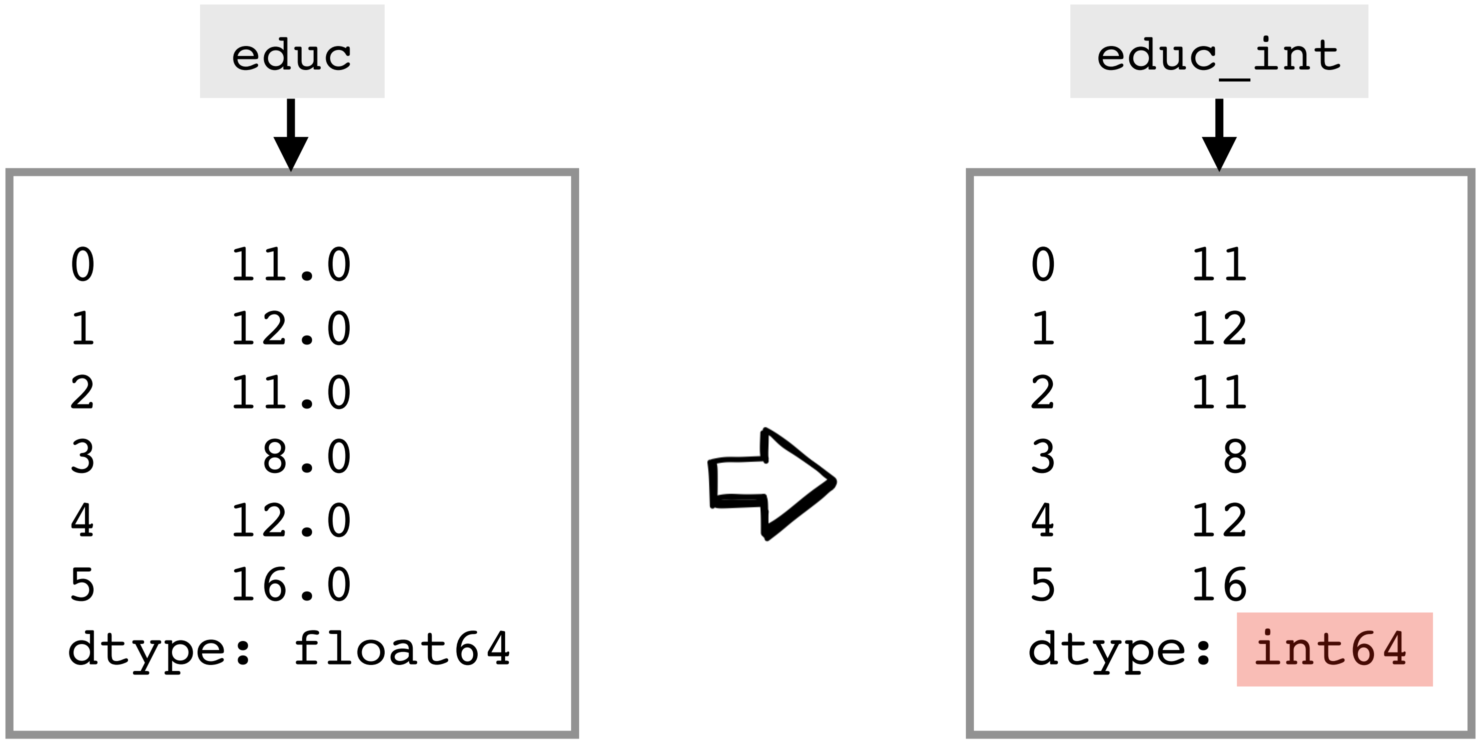
Pandas for Data Analysis

- The `pandas.Series` data structure
  - Attributes of series

```
educ_int = educ.astype(int)
print(educ_int)
```

Convert all data items to integers

```
0     11
1     12
2     11
3      8
4     12
5     16
dtype: int64
```



educ

```
0     11.0
1     12.0
2     11.0
3      8.0
4     12.0
5     16.0
dtype: float64
```

educ_int

```
0     11
1     12
2     11
3      8
4     12
5     16
dtype: int64
```

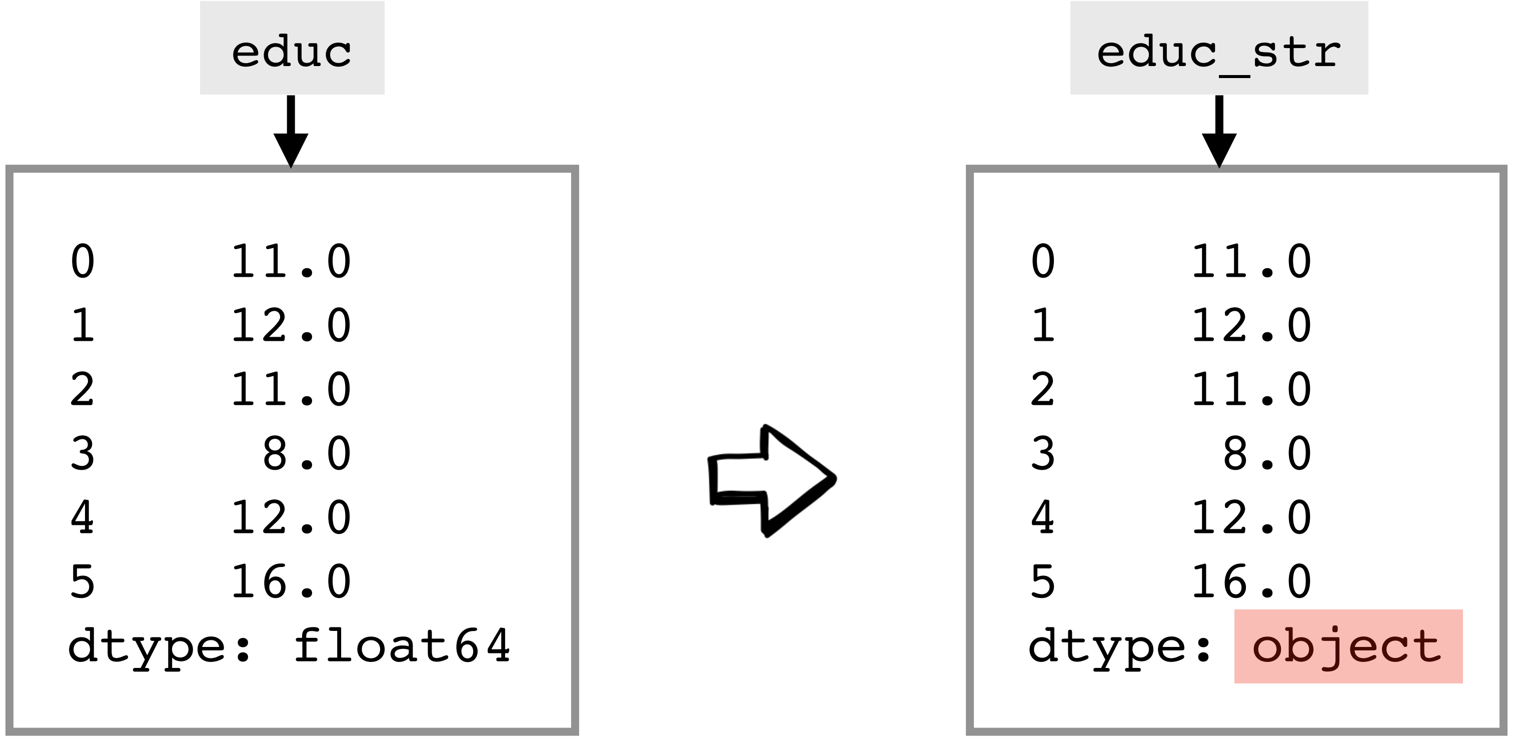
Pandas for Data Analysis

- The `pandas.Series` data structure
    - Attributes of series

```
educ_str = educ.astype(str)
print(educ_str)
```

Convert all data items to strings

```
0     11.0
1     12.0
2     11.0
3      8.0
4     12.0
5     16.0
dtype: object
```



`educ`

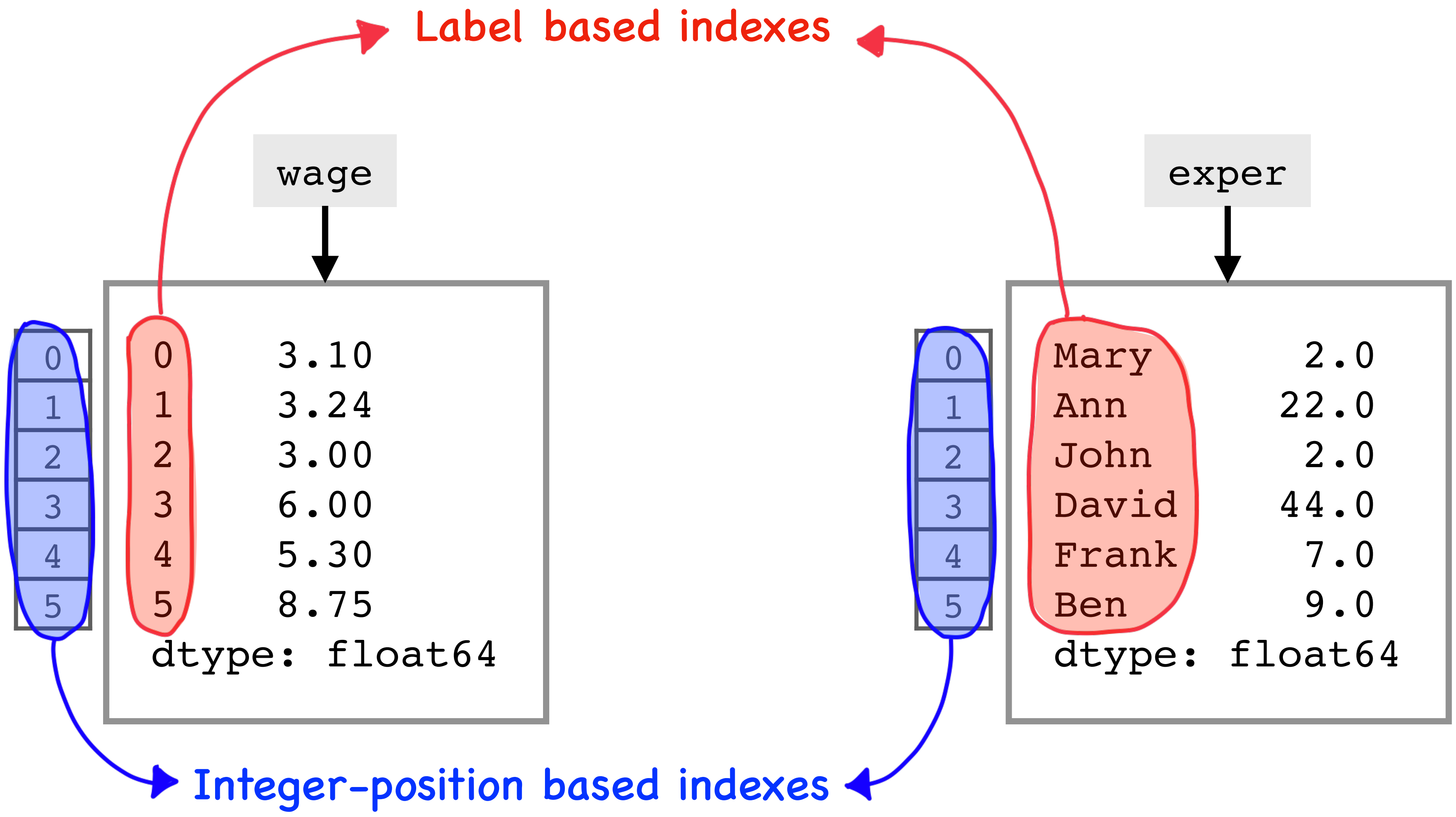
```
0     11.0
1     12.0
2     11.0
3      8.0
4     12.0
5     16.0
dtype: float64
```

`educ_str`

```
0     11.0
1     12.0
2     11.0
3      8.0
4     12.0
5     16.0
dtype: object
```

Pandas for Data Analysis

- The `pandas.Series` data structure
    - Indexing and slicing via `iloc[]` and `loc[]` indexers

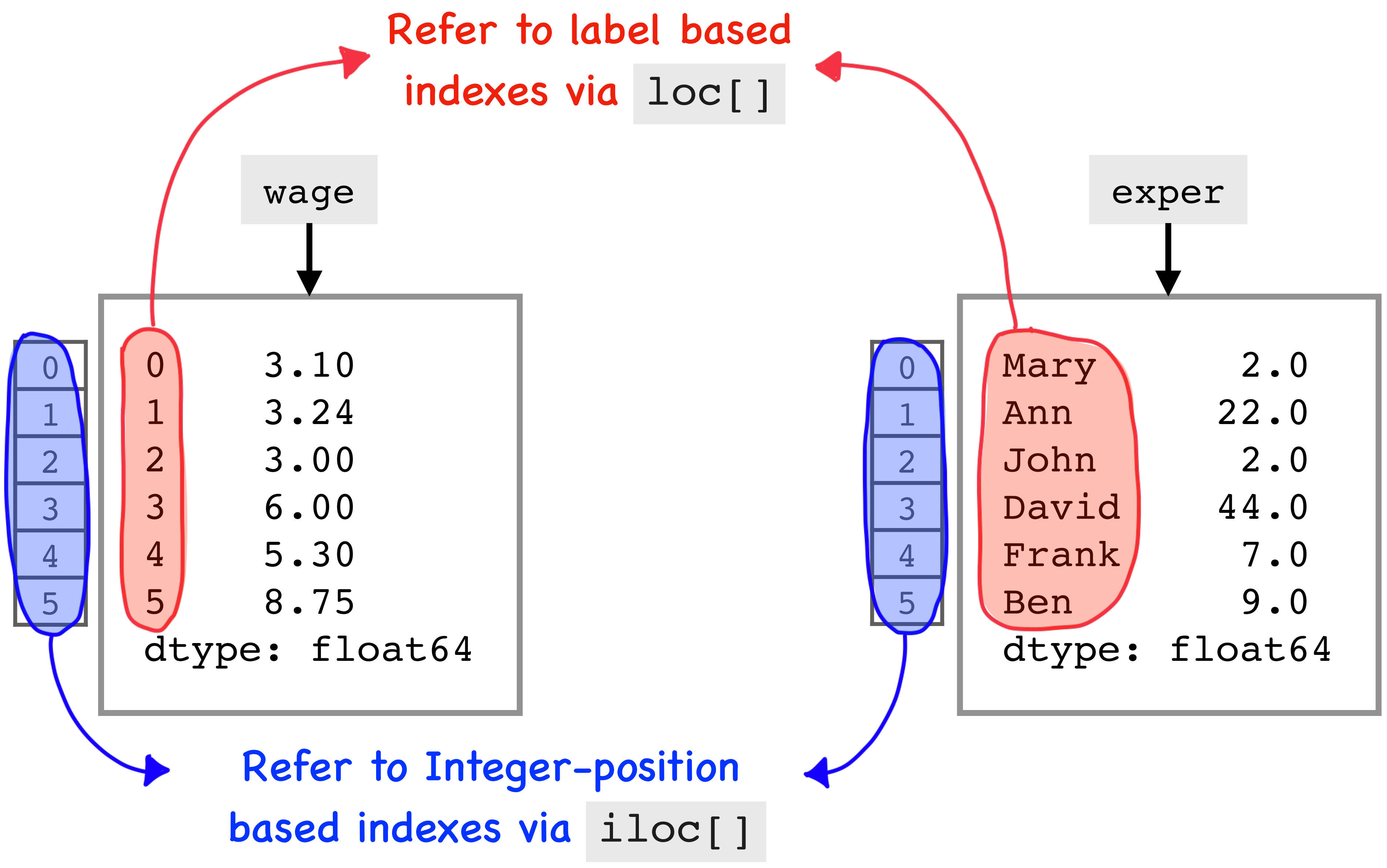


```
0      3.10
1      3.24
2      3.00
3      6.00
4      5.30
5      8.75
dtype: float64
```

```
Mary      2.0
Ann      22.0
John      2.0
David    44.0
Frank     7.0
Ben       9.0
dtype: float64
```

Pandas for Data Analysis

- The `pandas.Series` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers



Refer to label based indexes via `loc[]`

```
0        3.10
1        3.24
2        3.00
3        6.00
4        5.30
5        8.75
dtype: float64
```

```
Mary       2.0
Ann       22.0
John       2.0
David     44.0
Frank      7.0
Ben        9.0
dtype: float64
```

Refer to Integer-position based indexes via `iloc[]`

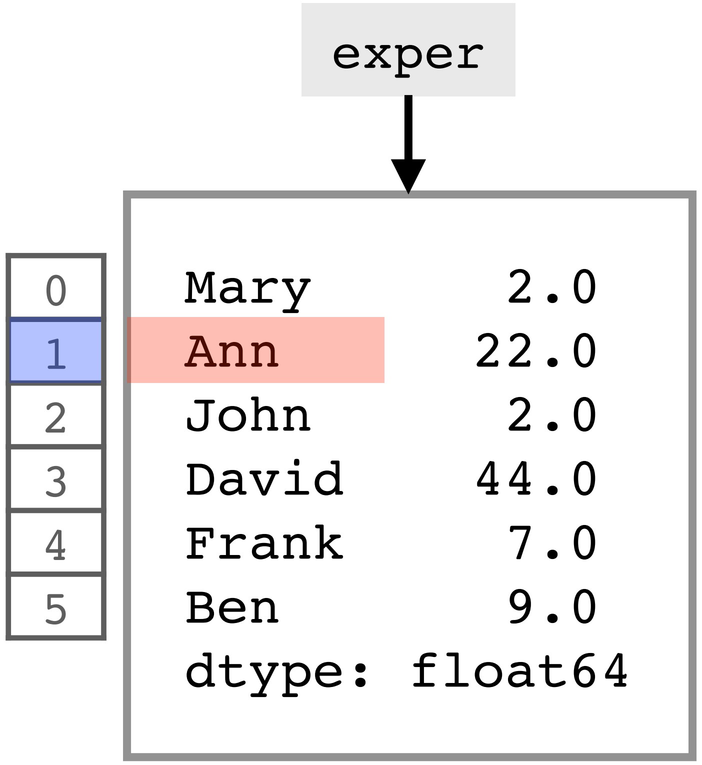
Pandas for Data Analysis

- The `pandas.Series` data structure
    - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
print(exper.iloc[1])
```

```
print(exper.loc['Ann'])
```

```
22.0
```



```
Mary      2.0
Ann      22.0
John      2.0
David    44.0
Frank     7.0
Ben       9.0
dtype: float64
```

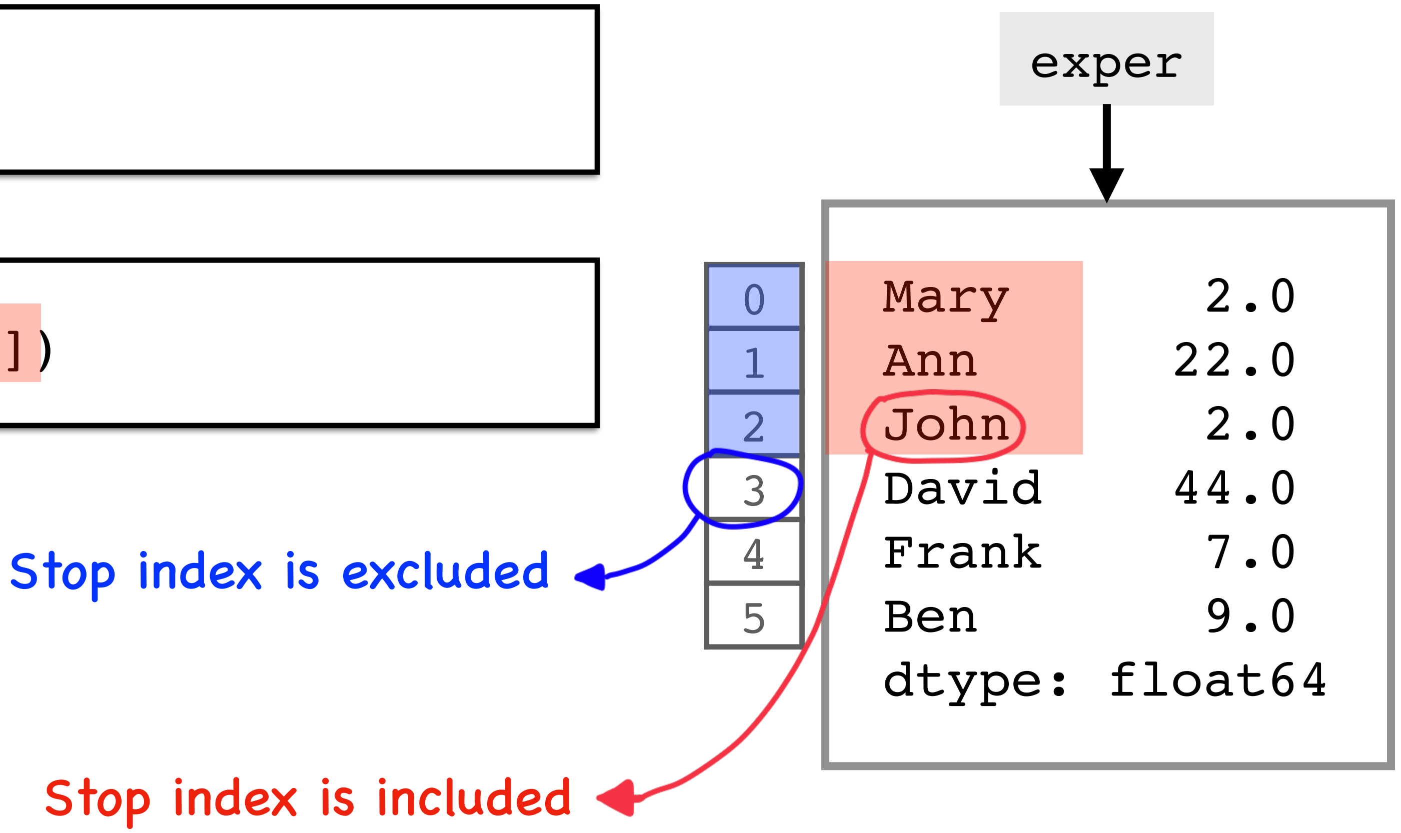
Pandas for Data Analysis

- The `pandas.Series` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
print(exper.iloc[:3])
```

```
print(exper.loc[:'John'])
```

```
Mary      2.0
Ann      22.0
John      2.0
dtype:  float64
```



exper

```
0    Mary       2.0
1    Ann       22.0
2    John       2.0
3    David     44.0
4    Frank      7.0
5    Ben        9.0
     dtype:  float64
```

Stop index is excluded

Stop index is included

Notes: In the slicing expressions for label based indexes, the item indexed by **stop** are included in the selection.

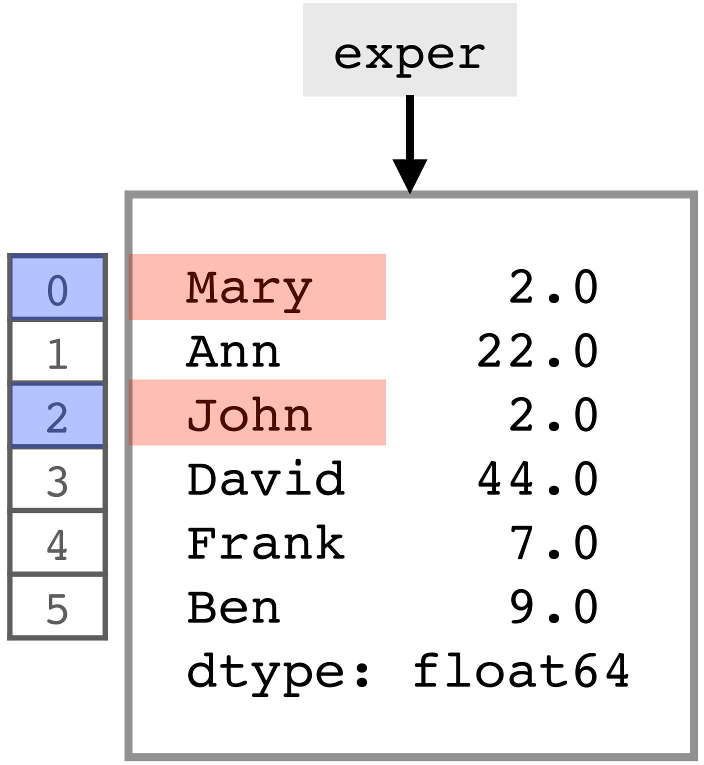
Pandas for Data Analysis

- The `pandas.Series` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
print(exper.iloc[[0, 2]])
```

```
print(exper.loc[['Mary', 'John']])
```

```
Mary     2.0
John     2.0
dtype:  float64
```



```
0   Mary       2.0
1   Ann       22.0
2   John       2.0
3   David     44.0
4   Frank      7.0
5   Ben        9.0
    dtype:  float64
```

Pandas for Data Analysis

- The `pandas.Series` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
print(wage.iloc[:2])
```

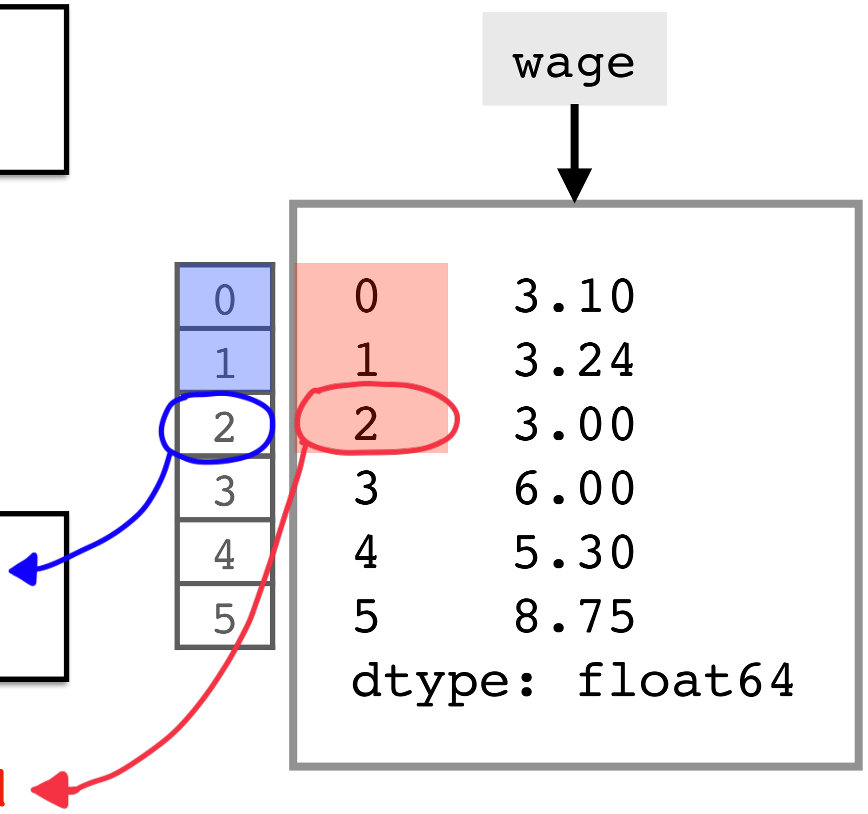
```
0     3.10
1     3.24
Name: wage, dtype: float64
```

```
print(wage.loc[:2])
```

Stop index is excluded

```
0     3.10
1     3.24
2     3.00
Name: wage, dtype: float64
```

Stop index is included



```
0     3.10
1     3.24
2     3.00
3     6.00
4     5.30
5     8.75
dtype: float64
```

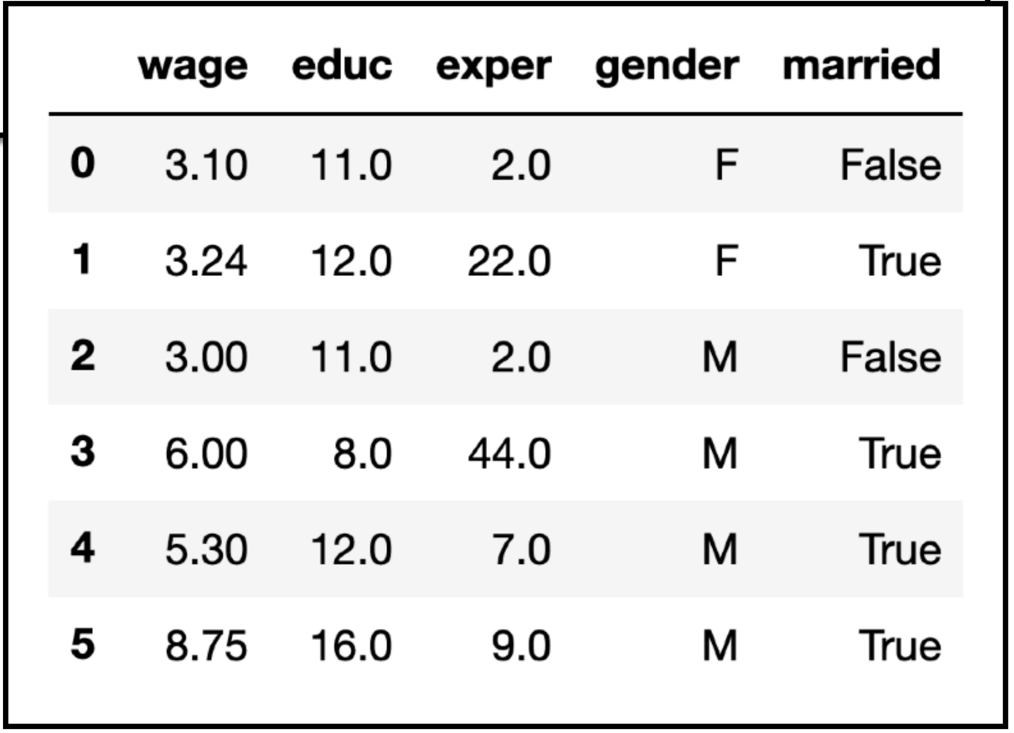
Pandas for Data Analysis

- The `pandas.DataFrame` data structure
    - Create a `pandas.DataFrame` object

```
data_dict = {'wage': [3.10, 3.24, 3.00, 6.00, 5.30, 8.75],
             'educ': [11.0, 12.0, 11.0, 8.0, 12.0, 16.0],
             'exper': [2.0, 22.0, 2.0, 44.0, 7.0, 9.0],
             'gender': ['F', 'F', 'M', 'M', 'M', 'M'],
             'married': [False, True, False, True, True, True]}

data_frame = pd.DataFrame(data_dict)
```

Function that creates a data frame



	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

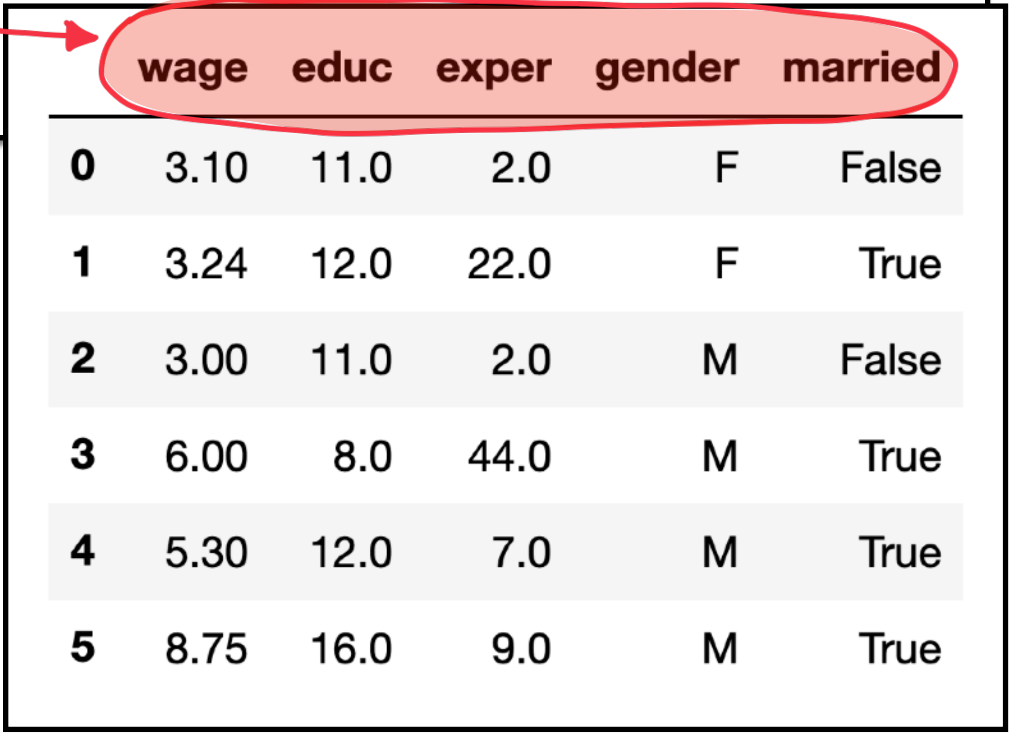
Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Create a `pandas.DataFrame` object

```
data_dict = {'wage': [3.10, 3.24, 3.00, 6.00, 5.30, 8.75],
             'educ': [11.0, 12.0, 11.0, 8.0, 12.0, 16.0],
             'exper': [2.0, 22.0, 2.0, 44.0, 7.0, 9.0],
             'gender': ['F', 'F', 'M', 'M', 'M', 'M'],
             'married': [False, True, False, True, True, True]}

data_frame = pd.DataFrame(data_dict)
```

Dictionary keys become column labels

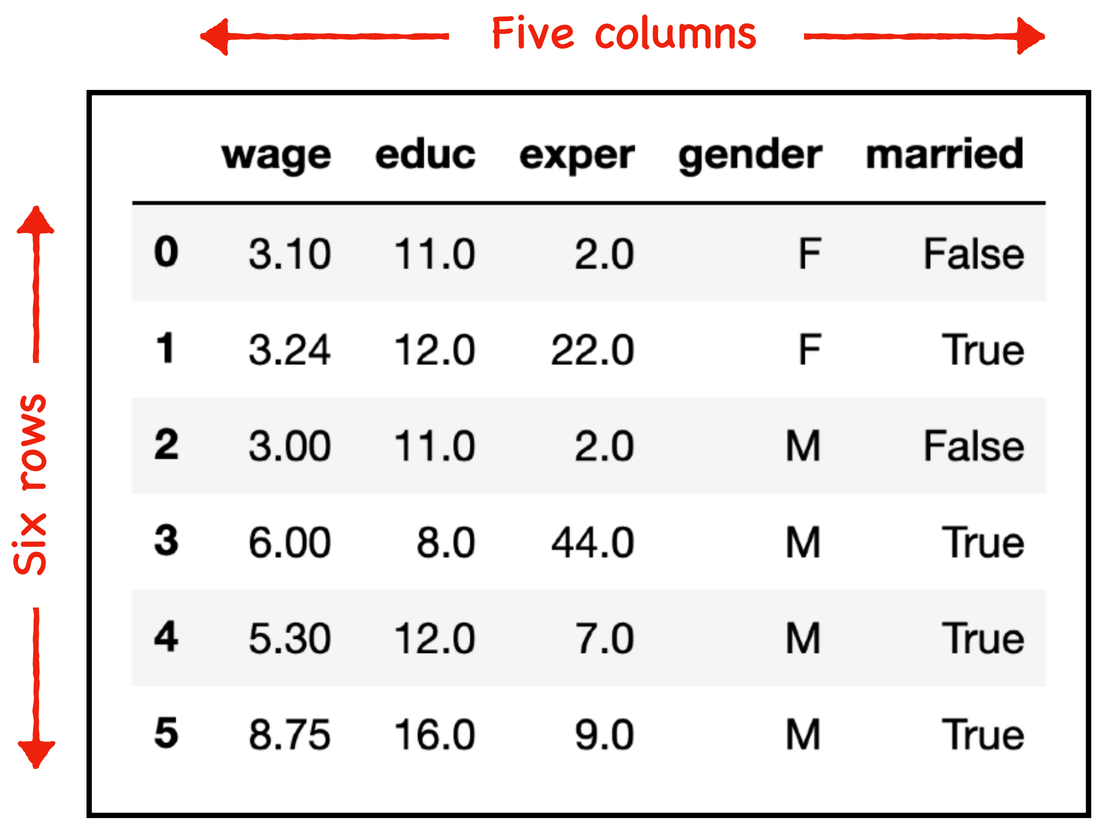


	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Attributes of data frames

Two-dimensional array



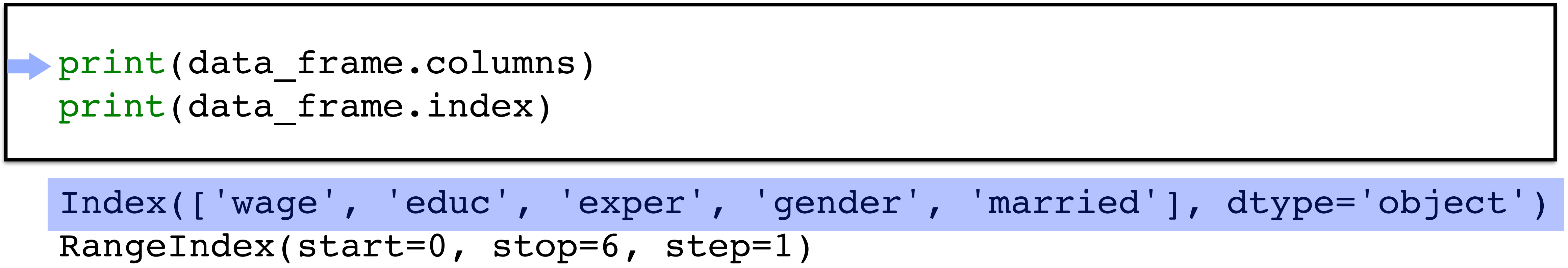
Five columns

Six rows

	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

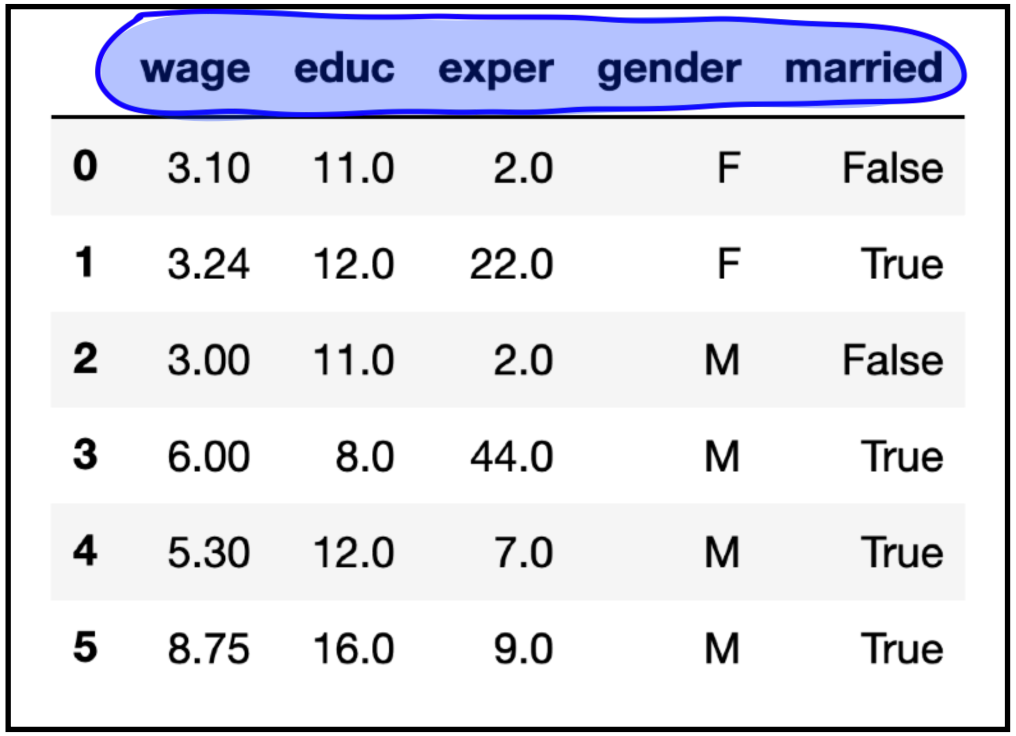
Pandas for Data Analysis

- The `pandas.DataFrame` data structure
    - Attributes of data frames



```
print(data_frame.columns)
print(data_frame.index)
```

```
Index(['wage', 'educ', 'exper', 'gender', 'married'], dtype='object')
RangeIndex(start=0, stop=6, step=1)
```



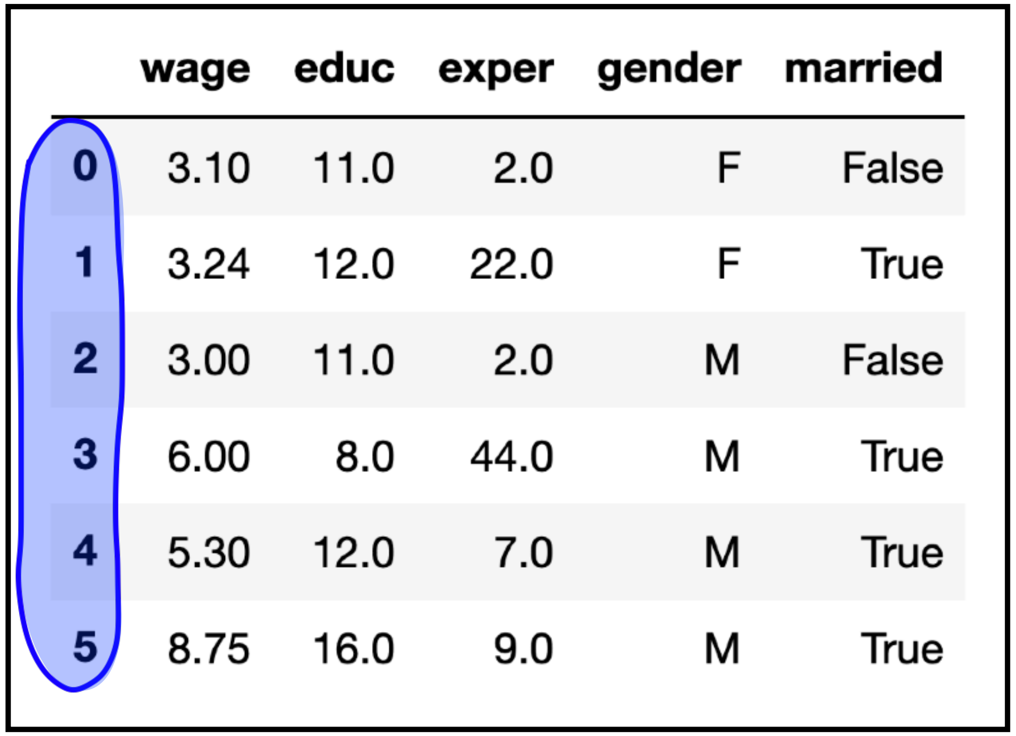
	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Attributes of data frames

```
print(data_frame.columns)
print(data_frame.index)
```

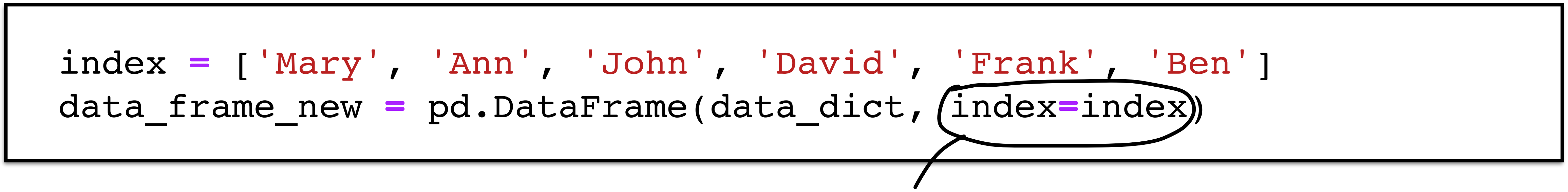
```
Index(['wage', 'educ', 'exper', 'gender', 'married'], dtype='object')
RangeIndex(start=0, stop=6, step=1)
```



	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

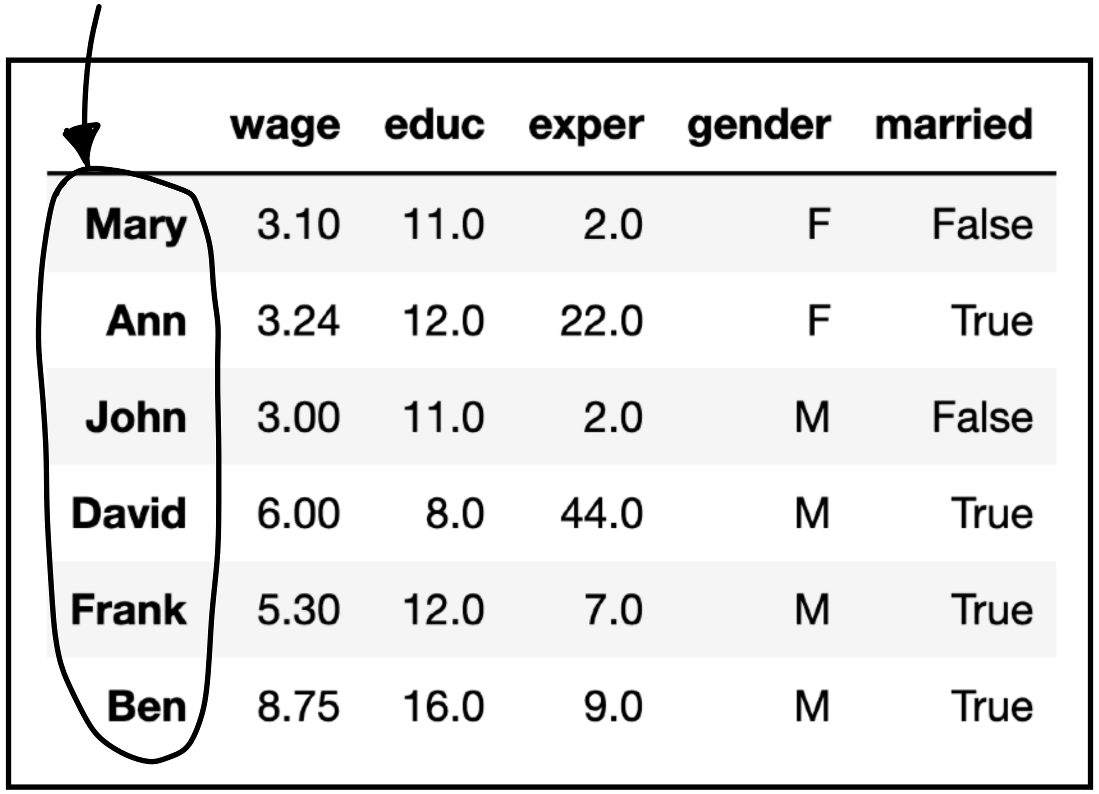
Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Attributes of data frames



```
index = ['Mary', 'Ann', 'John', 'David', 'Frank', 'Ben']
data_frame_new = pd.DataFrame(data_dict, index=index)
```

Specify data frame row indexes via the keyword argument



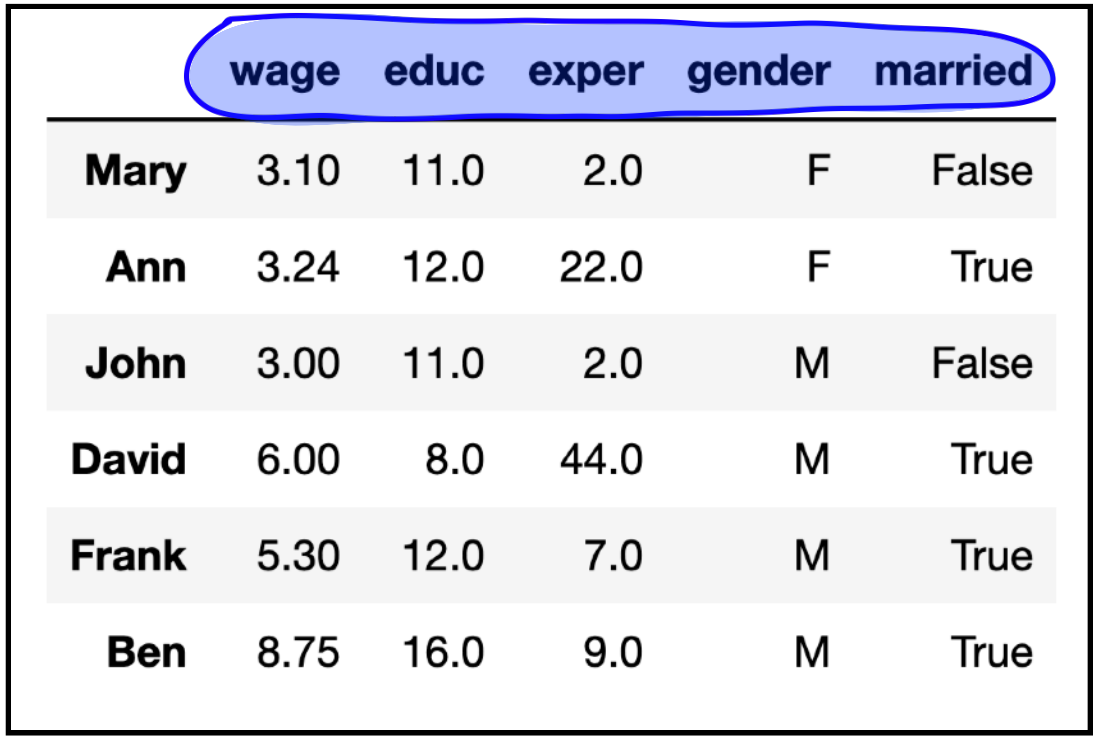
	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Attributes of data frames

```
print(data_frame_new.columns)
print(data_frame_new.index)
```

```
Index(['wage', 'educ', 'exper', 'gender', 'married'], dtype='object')
Index(['Mary', 'Ann', 'John', 'David', 'Frank', 'Ben'], dtype='object')
```



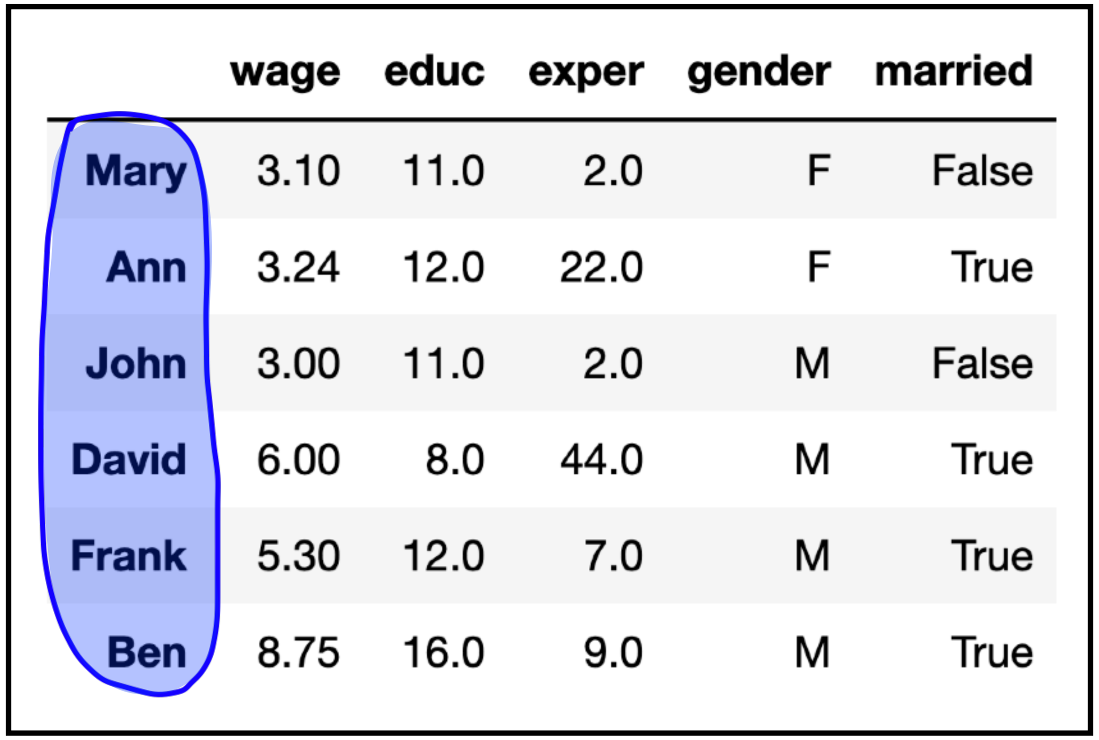
	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Attributes of data frames

```python
print(data_frame_new.columns)
print(data_frame_new.index)
```

```
Index(['wage', 'educ', 'exper', 'gender', 'married'], dtype='object')
Index(['Mary', 'Ann', 'John', 'David', 'Frank', 'Ben'], dtype='object')
```



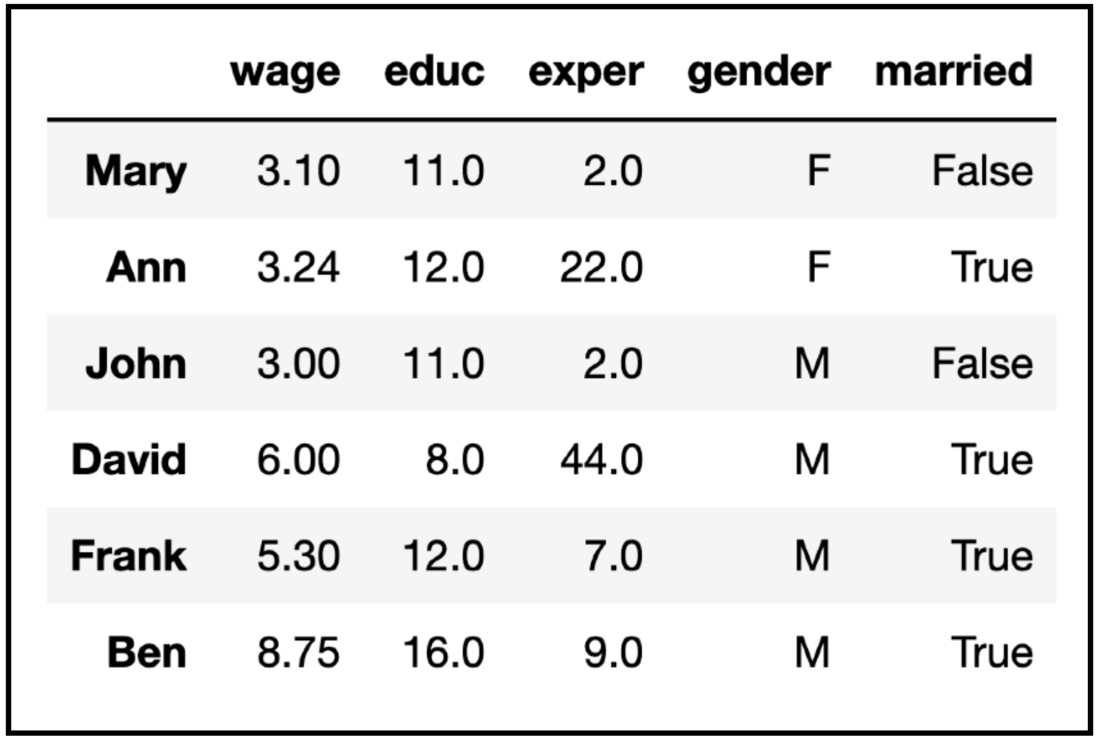
	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Attributes of data frames

```
data_frame.dtypes
```

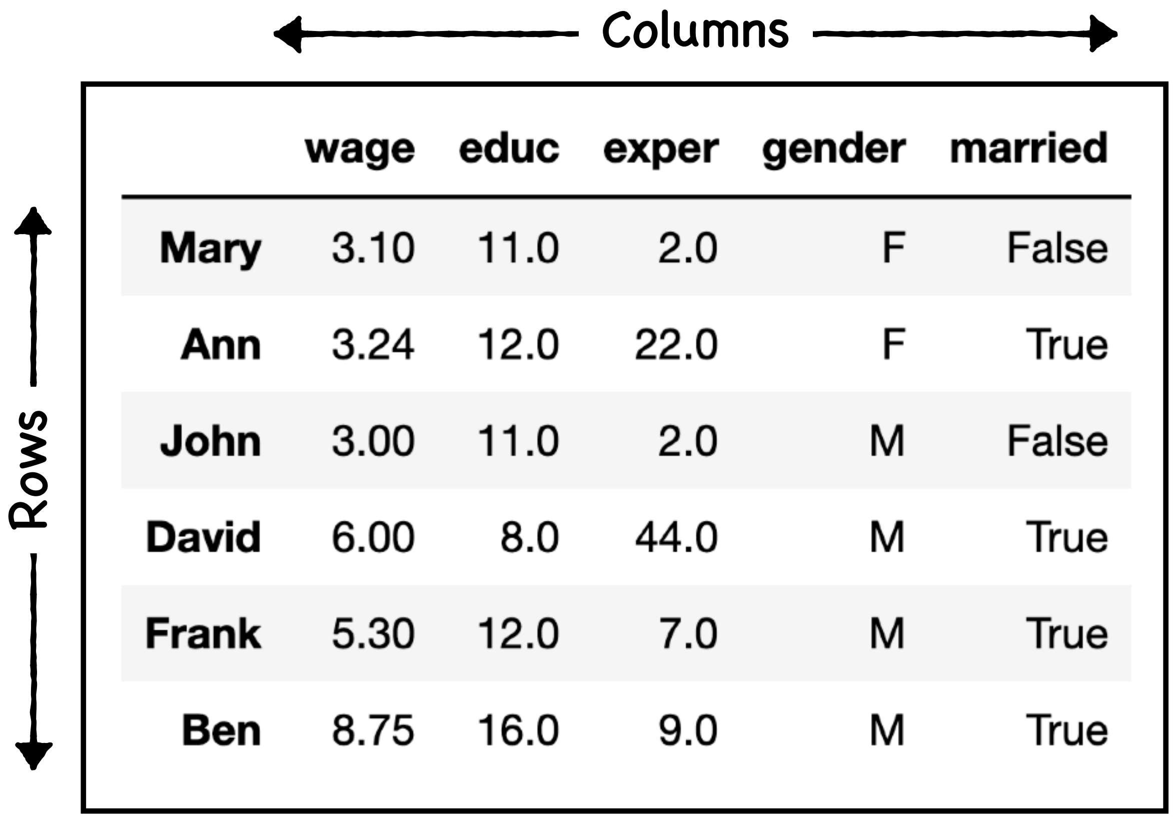
```
wage          float64
educ          float64
exper         float64
gender         object
married          bool
dtype: object
```



	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

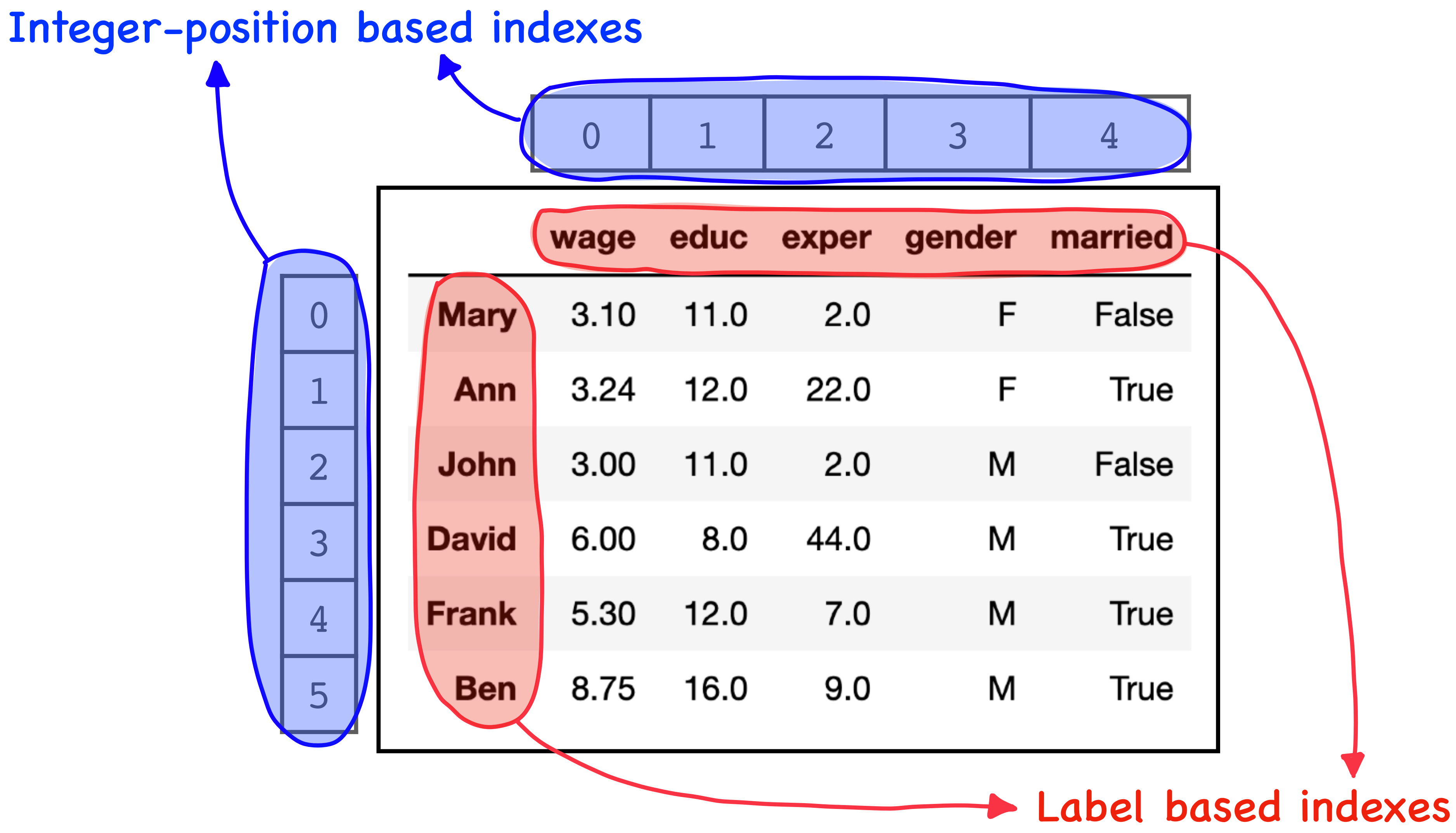
- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers


Columns
Rows

	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

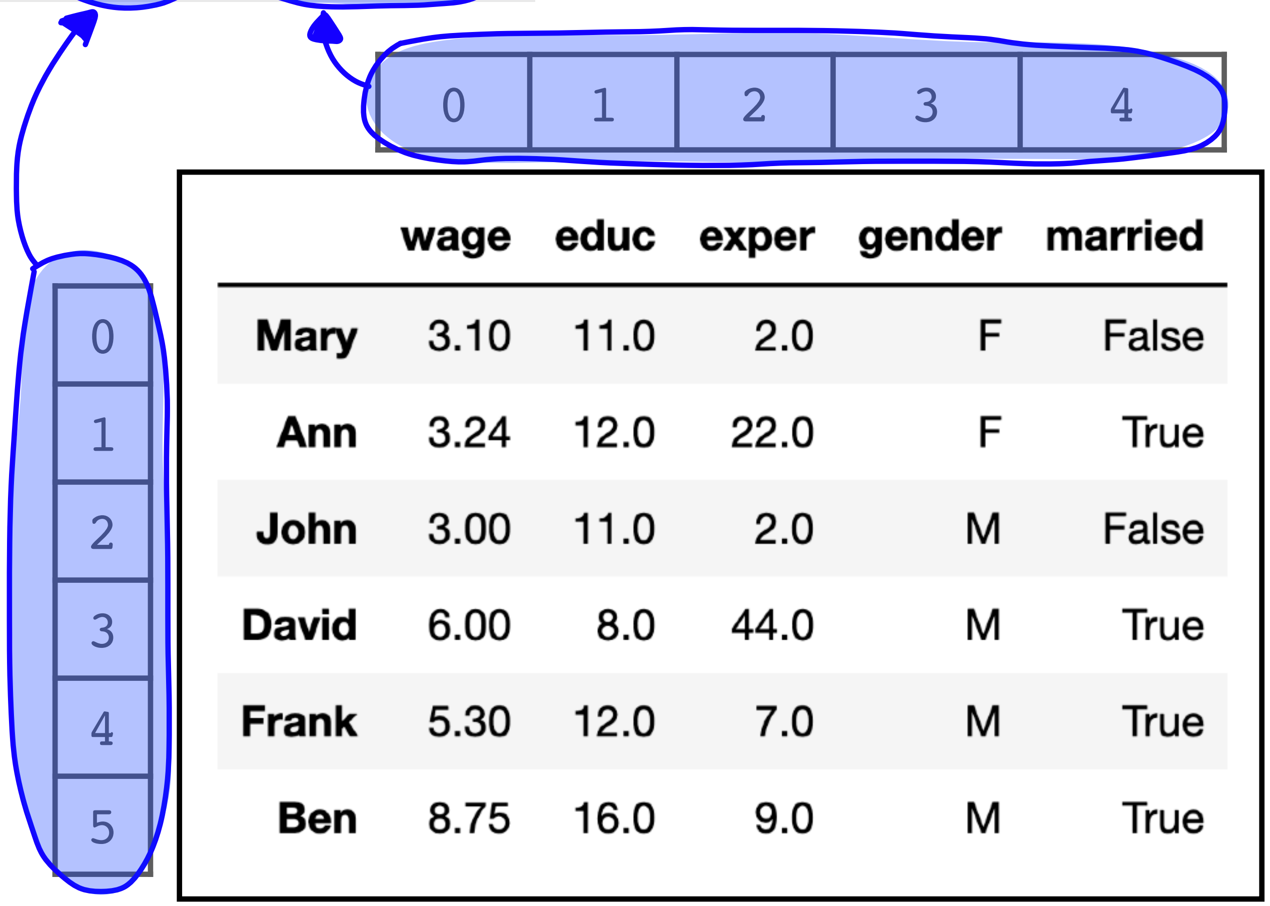


	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

Selection using `iloc[rows, columns]`



	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Column selections
- A single column index
- A slice of column indexes
- A list of column indexes

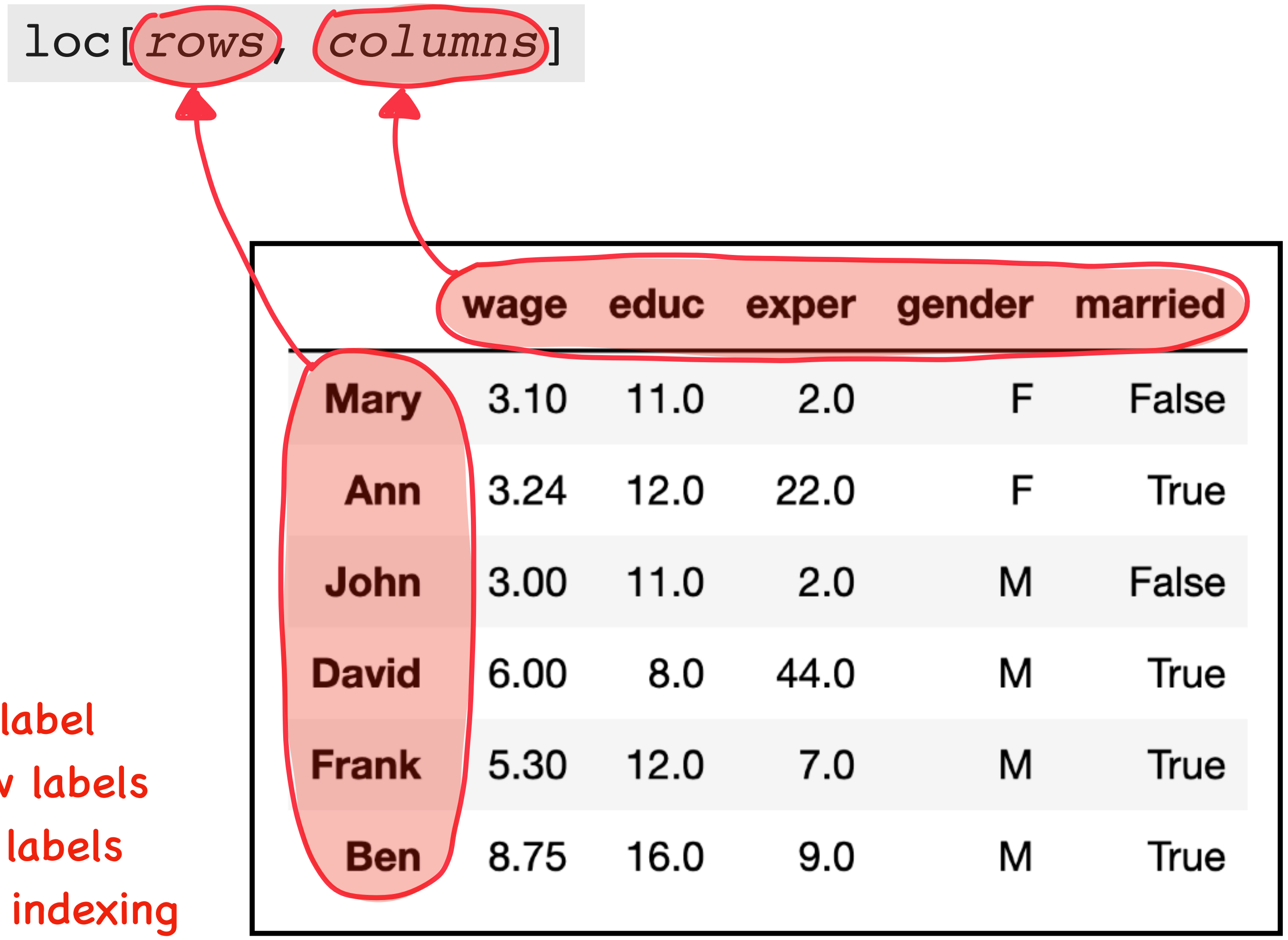
Row selections
- A single row index
- A slice of row indexes
- A list of row indexes

The stop index is excluded from the row/column selection

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

Selection using `loc[rows, columns]`



	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Column selections

- A single column label
- A slice of column labels
- A list of column labels

Row selections

- A single row label
- A slice of row labels
- A list of row labels
- Boolean type indexing

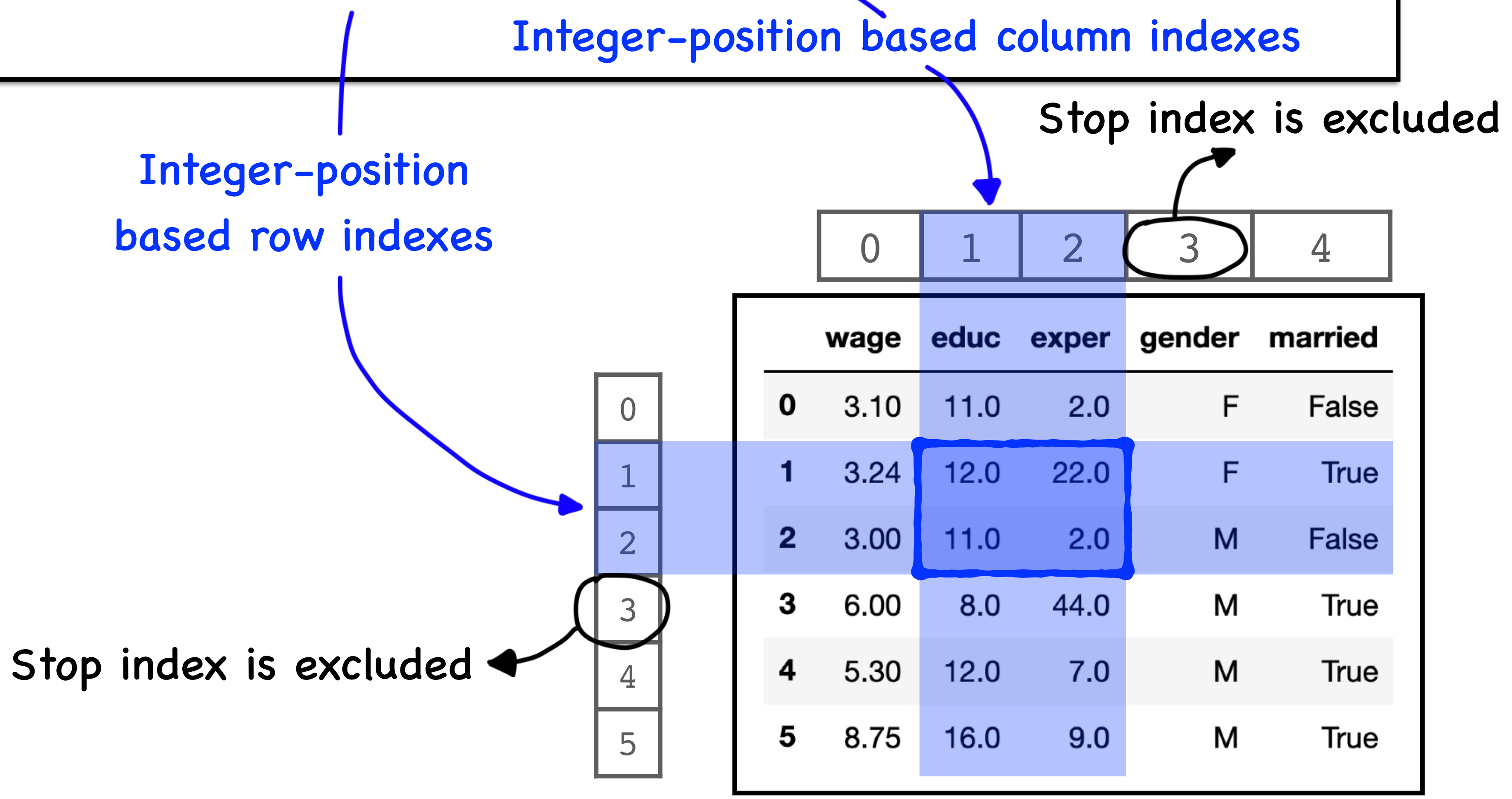
The stop index is included in the row/column selection

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
    - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
data_subset = data_frame.iloc[1:3, 1:3]
print(data_subset)
```

```
   educ  exper
1  12.0   22.0
2  11.0    2.0
```



Integer-position based column indexes

Integer-position based row indexes

Stop index is excluded

Stop index is excluded

	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

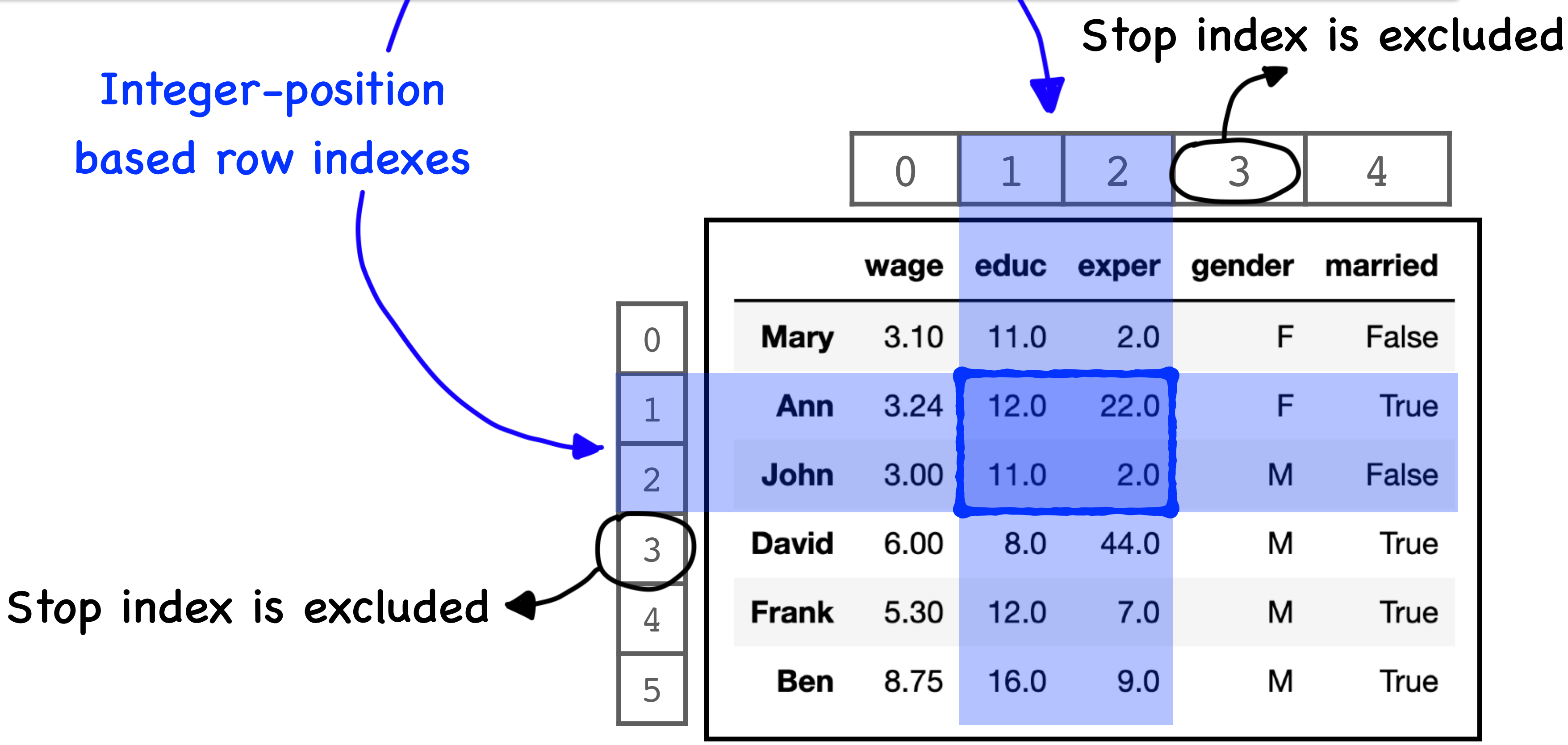
Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
data_subset = data_frame_new.iloc[1:3, 1:3]
print(data_subset)
```

Integer-position based column indexes

```
      educ  exper
Ann   12.0   22.0
John  11.0    2.0
```



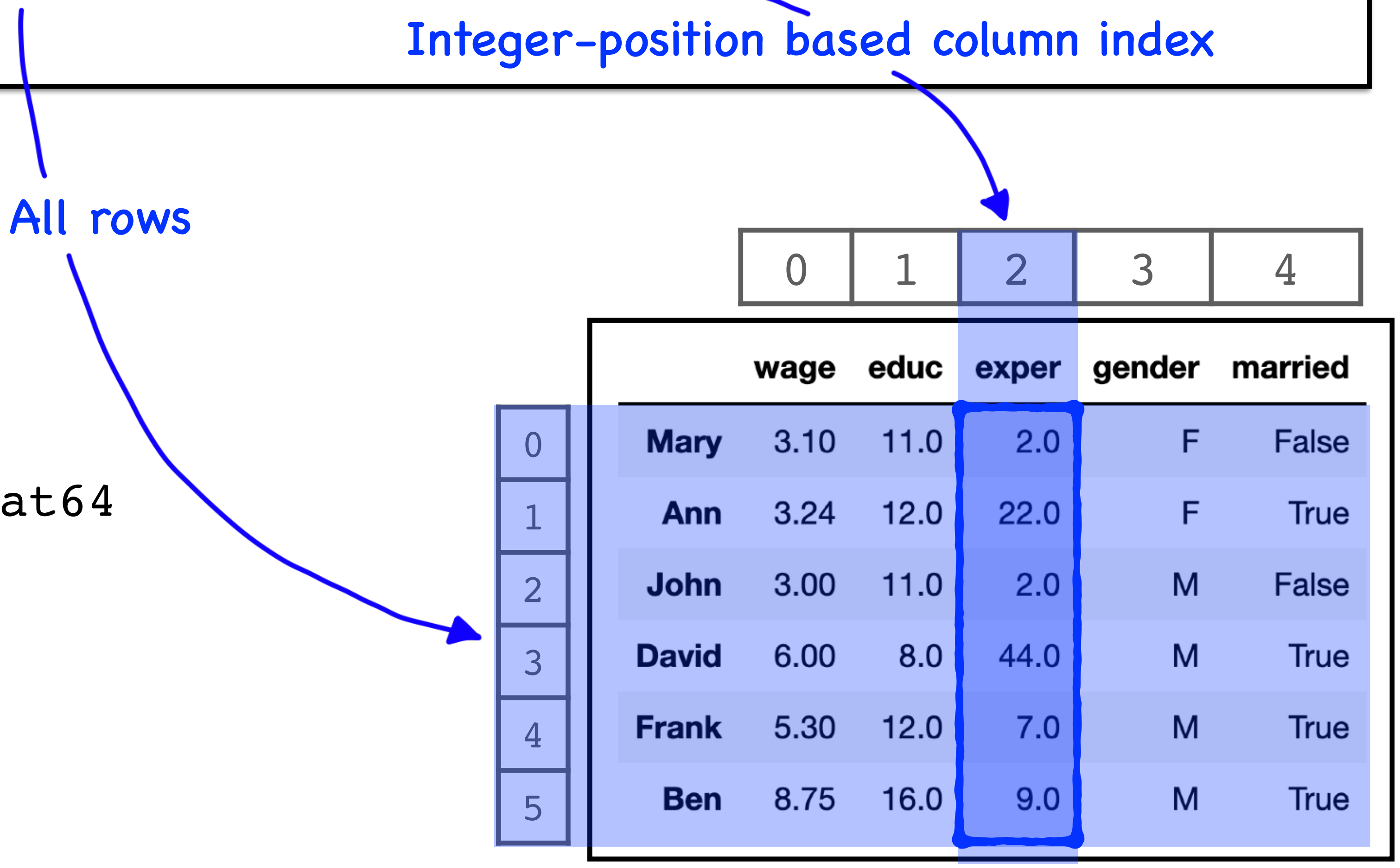
		0	1	2	3	4
		wage	educ	exper	gender	married
0	Mary	3.10	11.0	2.0	F	False
1	Ann	3.24	12.0	22.0	F	True
2	John	3.00	11.0	2.0	M	False
3	David	6.00	8.0	44.0	M	True
4	Frank	5.30	12.0	7.0	M	True
5	Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
cols = data_frame.iloc[:, 2]
print(cols)
```

```
0     2.0
1    22.0
2     1.0
3    44.0
4     7.0
5     9.0
Name: exper, dtype: float64
```



Integer-position based column index

All rows

		0	1	2	3	4
		wage	educ	exper	gender	married
0	Mary	3.10	11.0	2.0	F	False
1	Ann	3.24	12.0	22.0	F	True
2	John	3.00	11.0	2.0	M	False
3	David	6.00	8.0	44.0	M	True
4	Frank	5.30	12.0	7.0	M	True
5	Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

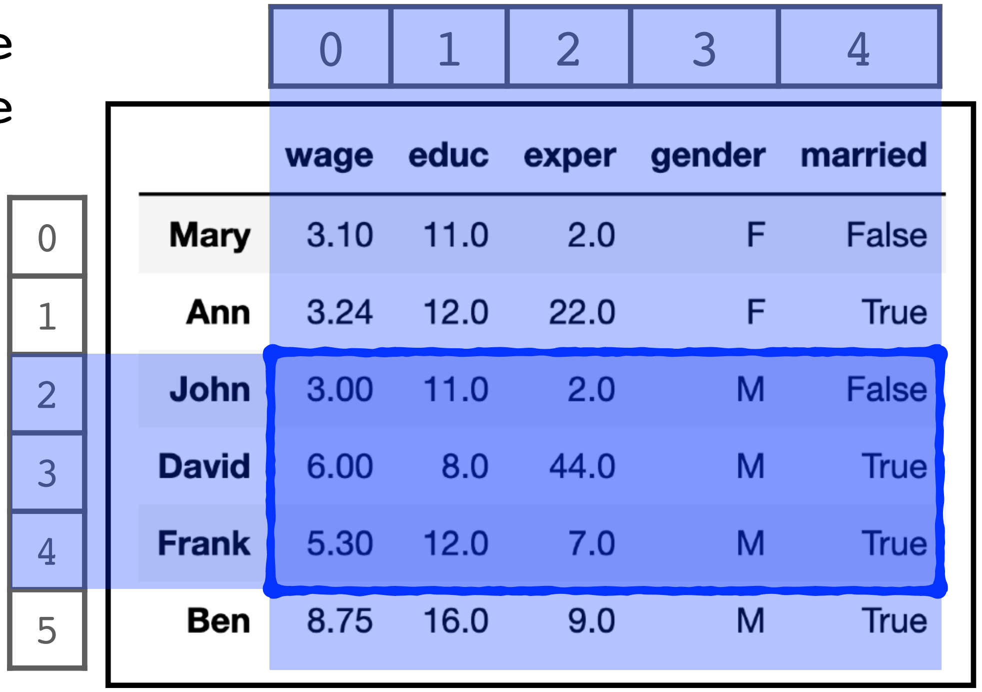
- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
rows = data_frame_new.iloc[2:5, :]
print(rows)
```

```
      wage  educ  exper gender  married
John   3.0  11.0    2.0      M    False
David  6.0   8.0   44.0      M     True
Frank  5.3  12.0    7.0      M     True
```

All columns

Integer-position based row indexes



		0	1	2	3	4
		wage	educ	exper	gender	married
0	Mary	3.10	11.0	2.0	F	False
1	Ann	3.24	12.0	22.0	F	True
2	John	3.00	11.0	2.0	M	False
3	David	6.00	8.0	44.0	M	True
4	Frank	5.30	12.0	7.0	M	True
5	Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

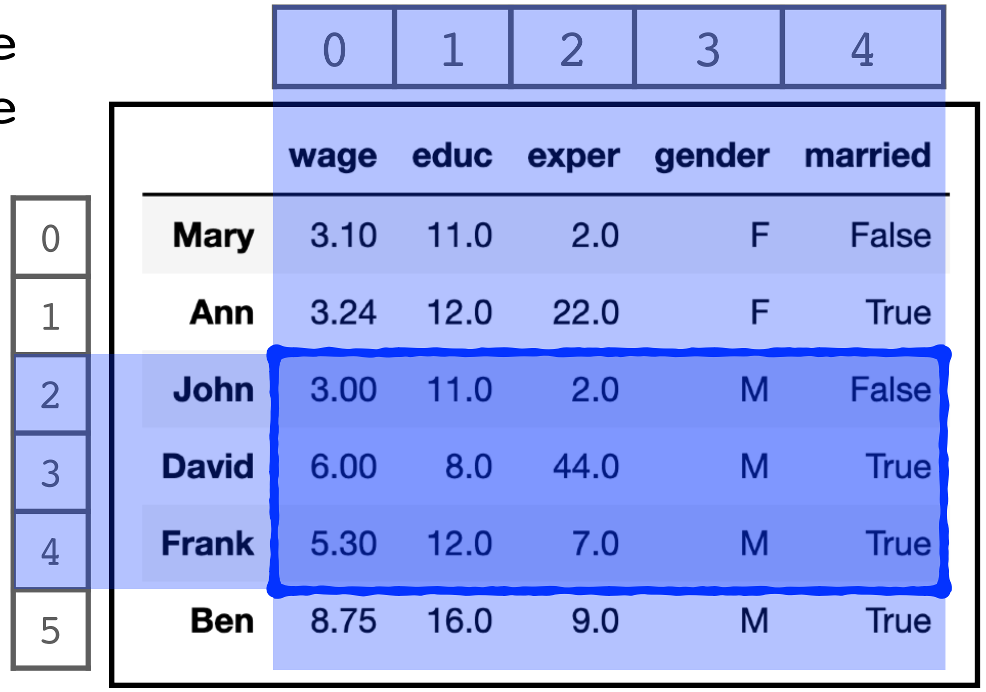
- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
rows = data_frame_new.iloc[2:5]
print(rows)
```

```
      wage  educ  exper gender  married
John   3.0  11.0    2.0      M    False
David  6.0   8.0   44.0      M     True
Frank  5.3  12.0    7.0      M     True
```

Integer-position based row indexes

All columns by default



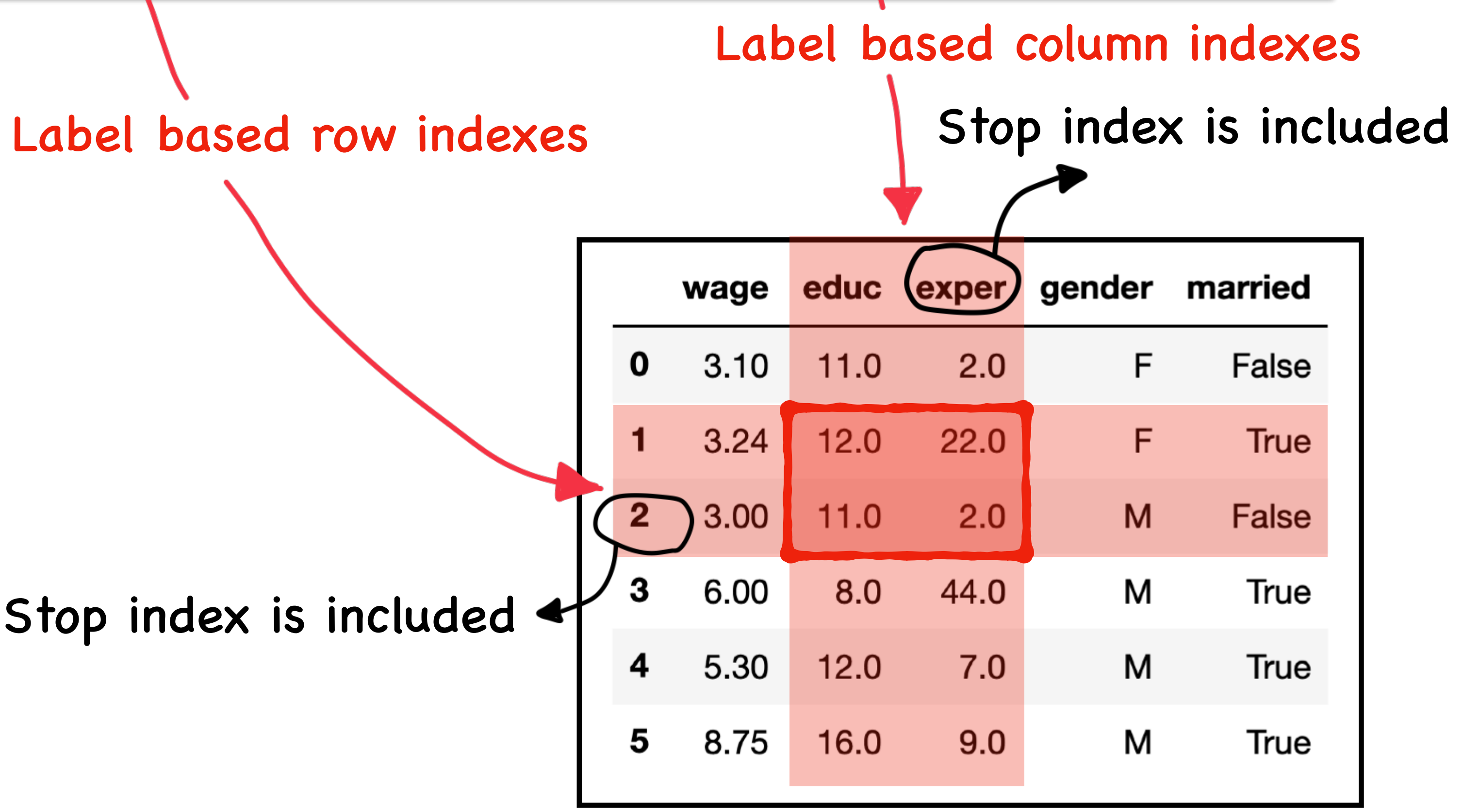
		0	1	2	3	4
		wage	educ	exper	gender	married
0	Mary	3.10	11.0	2.0	F	False
1	Ann	3.24	12.0	22.0	F	True
2	John	3.00	11.0	2.0	M	False
3	David	6.00	8.0	44.0	M	True
4	Frank	5.30	12.0	7.0	M	True
5	Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
data_subset = data_frame.loc[1:2, 'educ':'exper']
print(data_subset)
```

```
    educ  exper
1   12.0   22.0
2   11.0    2.0
```



Label based column indexes

Label based row indexes

Stop index is included

Stop index is included

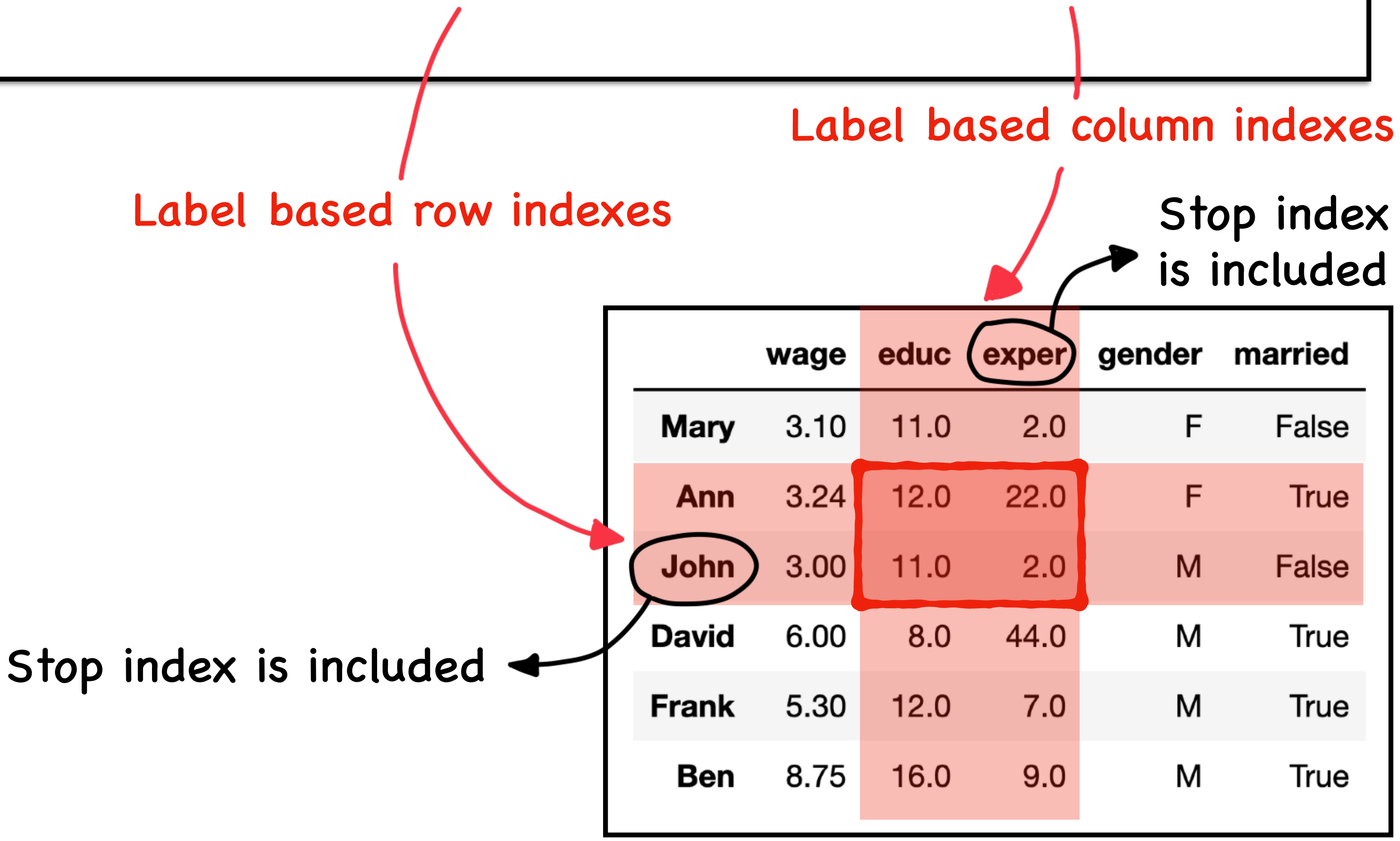
	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
data_new_subset = data_frame_new.loc['Ann':'John', 'educ':'exper']
print(data_new_subset)
```

```
      educ  exper
Ann   12.0   22.0
John  11.0    2.0
```



Label based column indexes

Label based row indexes

Stop index is included

Stop index is included

	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

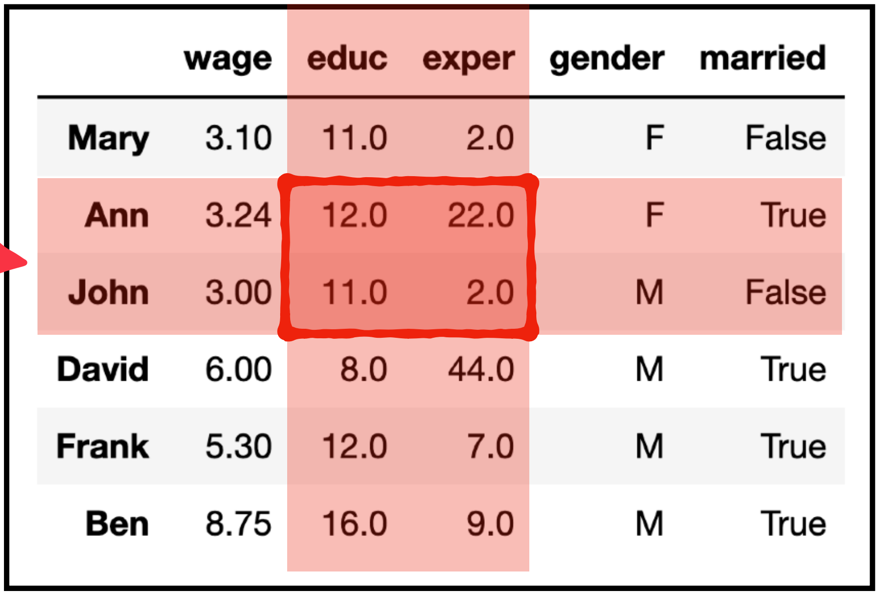
- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
data_new_subset = data_frame_new.loc['Ann':'John', 'educ':'exper']
print(data_new_subset)
```

```
      educ  exper
Ann   12.0   22.0
John  11.0    2.0
```

Label based column indexes

Label based row indexes



	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Notes: In the slicing expressions for label based indexes, the item indexed by **stop** are included in the selection.

Pandas for Data Analysis

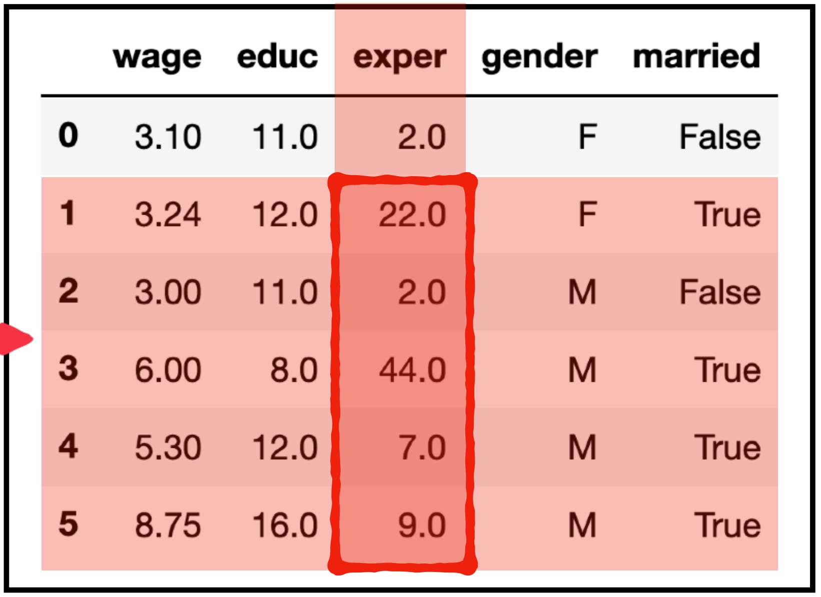
- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```python
cols = data_frame.loc[:, 'exper']
print(cols)
```

```
0     2.0
1    22.0
2     1.0
3    44.0
4     7.0
5     9.0
Name: exper, dtype: float64
```

All rows

Label based column indexes



	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Pandas for Data Analysis

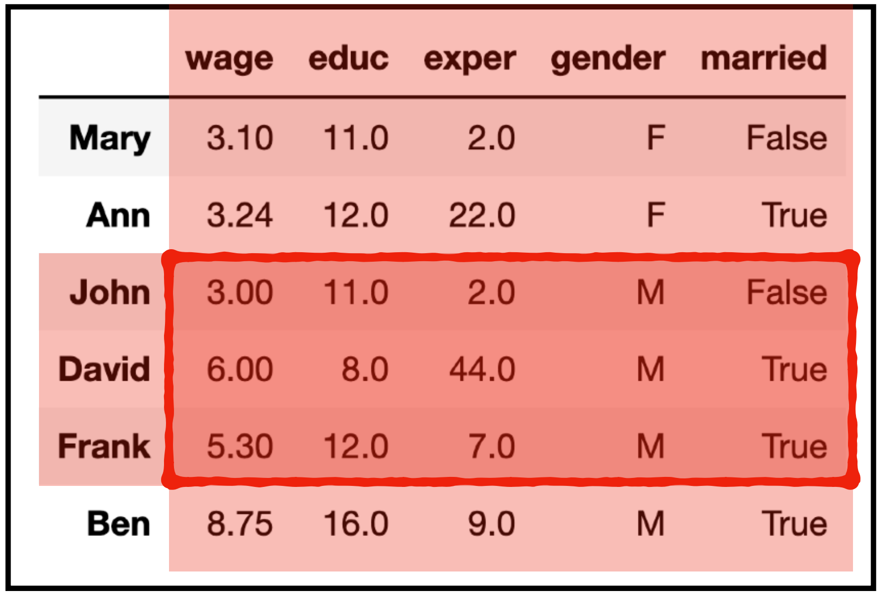
- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
rows = data_frame_new.loc['John':'Frank', :]
print(rows)
```

```
       wage  educ  exper gender  married
John   3.00  11.0    2.0      M    False
David  6.00   8.0   44.0      M     True
Frank  5.30  12.0    7.0      M     True
```

All columns

Label based row indexes



	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

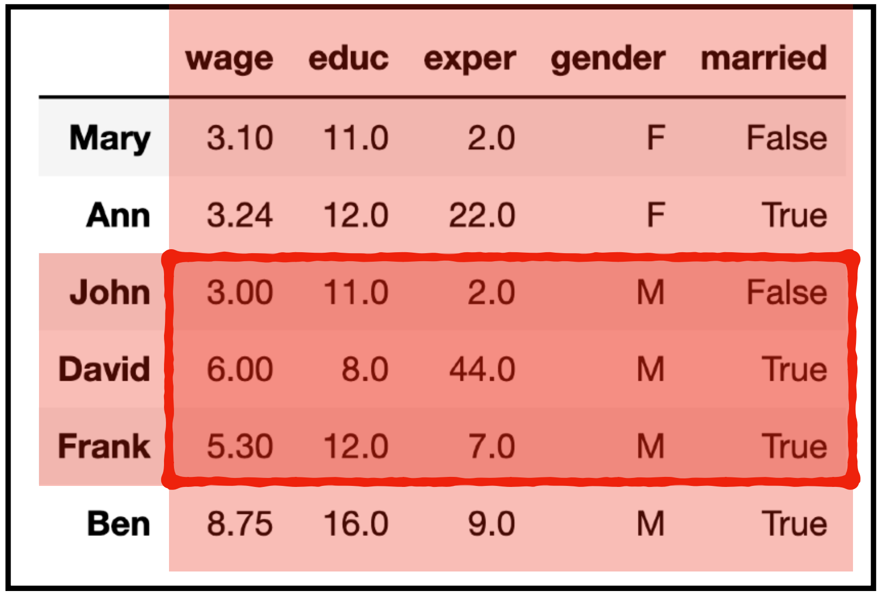
- The `pandas.DataFrame` data structure
  - Indexing and slicing via `iloc[]` and `loc[]` indexers

```
rows = data_frame_new.loc['John':'Frank']
print(rows)
```

```
       wage  educ  exper gender  married
John   3.00  11.0    2.0      M    False
David  6.00   8.0   44.0      M     True
Frank  5.30  12.0    7.0      M     True
```

Label based row indexes

All columns by default



	wage	educ	exper	gender	married
Mary	3.10	11.0	2.0	F	False
Ann	3.24	12.0	22.0	F	True
John	3.00	11.0	2.0	M	False
David	6.00	8.0	44.0	M	True
Frank	5.30	12.0	7.0	M	True
Ben	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Indexing of columns

```
data_frame['educ']
```

```
0     11.0
1     12.0
2     11.0
3      8.0
4     12.0
5     16.0
Name: educ, dtype: float64
```

```
data_frame_new[‘educ’]
```

```
Mary      11.0
Ann       12.0
John      11.0
David      8.0
Frank     12.0
Ben       16.0
Name: educ, dtype: float64
```

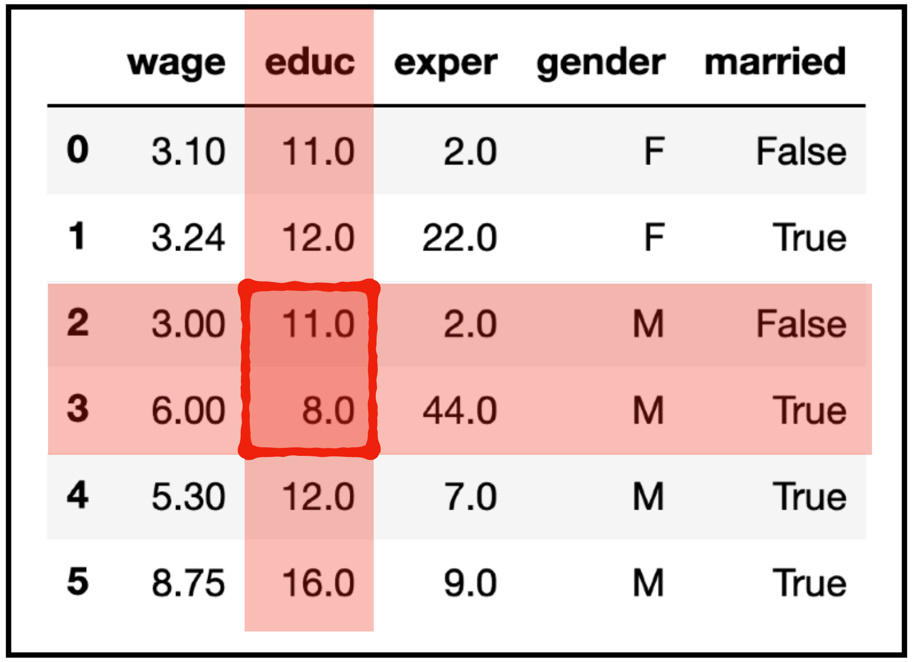
Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Change `pandas.DataFrame` object in-place
    - ✓ Similar syntax as other mutable data types

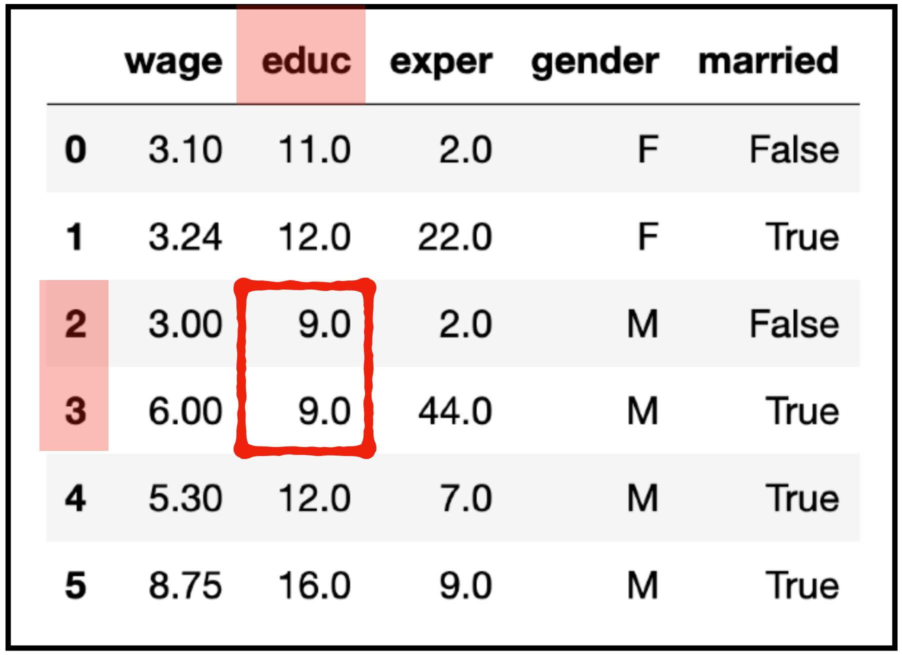
Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Change `pandas.DataFrame` object in-place

```
data_frame.loc[2:3, 'educ'] = 9.0
```



	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

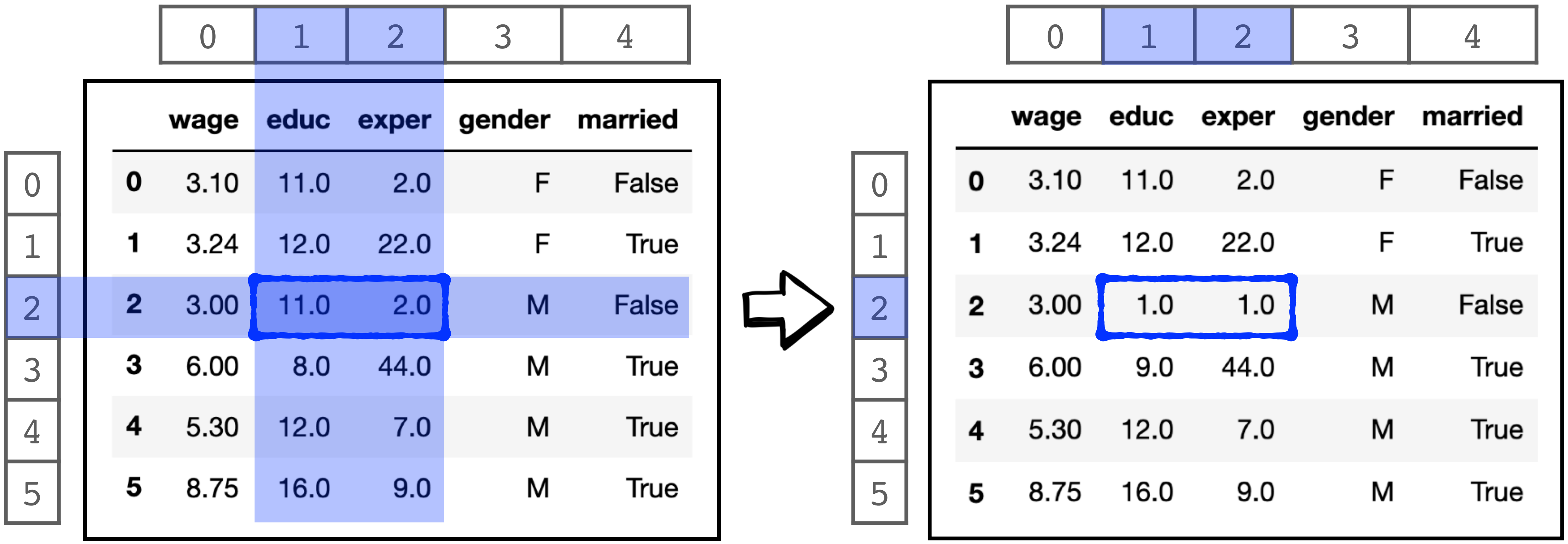


	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	9.0	2.0	M	False
3	6.00	9.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Change `pandas.DataFrame` object in-place

```
data_frame.iloc[2, 1:3] = 1.0
```

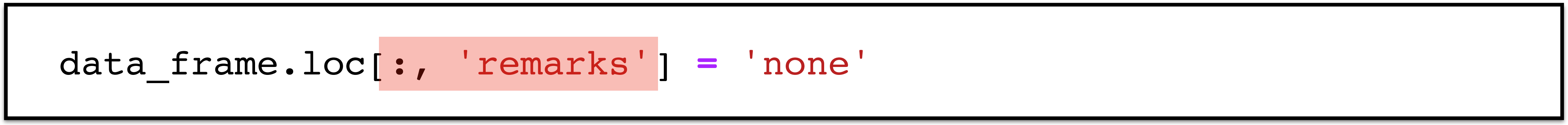


		0	1	2	3	4
		wage	educ	exper	gender	married
0	**0**	3.10	11.0	2.0	F	False
1	**1**	3.24	12.0	22.0	F	True
2	**2**	3.00	11.0	2.0	M	False
3	**3**	6.00	8.0	44.0	M	True
4	**4**	5.30	12.0	7.0	M	True
5	**5**	8.75	16.0	9.0	M	True

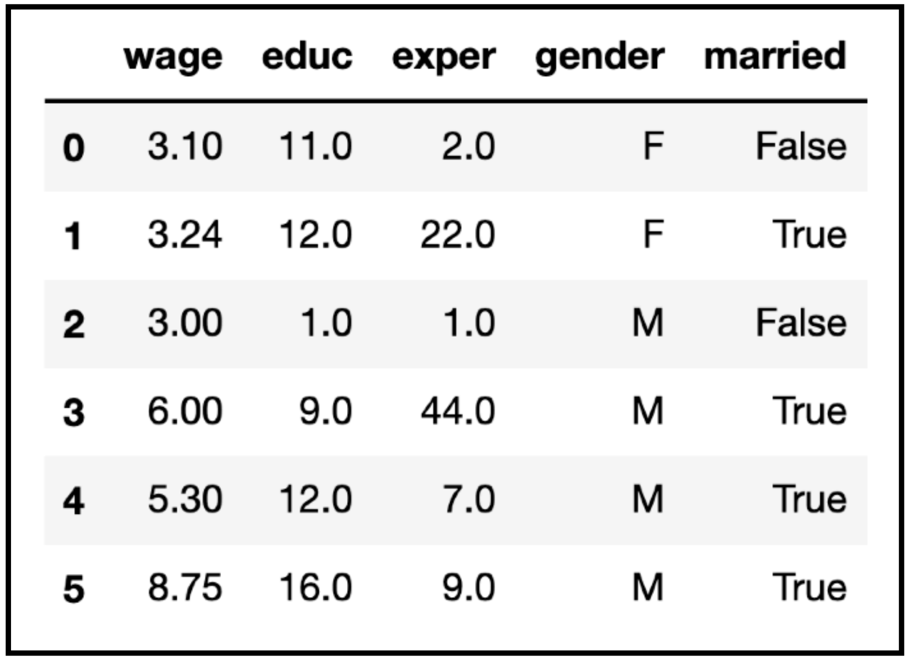
		0	1	2	3	4
		wage	educ	exper	gender	married
0	**0**	3.10	11.0	2.0	F	False
1	**1**	3.24	12.0	22.0	F	True
2	**2**	3.00	1.0	1.0	M	False
3	**3**	6.00	9.0	44.0	M	True
4	**4**	5.30	12.0	7.0	M	True
5	**5**	8.75	16.0	9.0	M	True

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Change `pandas.DataFrame` object in-place



```
data_frame.loc[:, 'remarks'] = 'none'
```

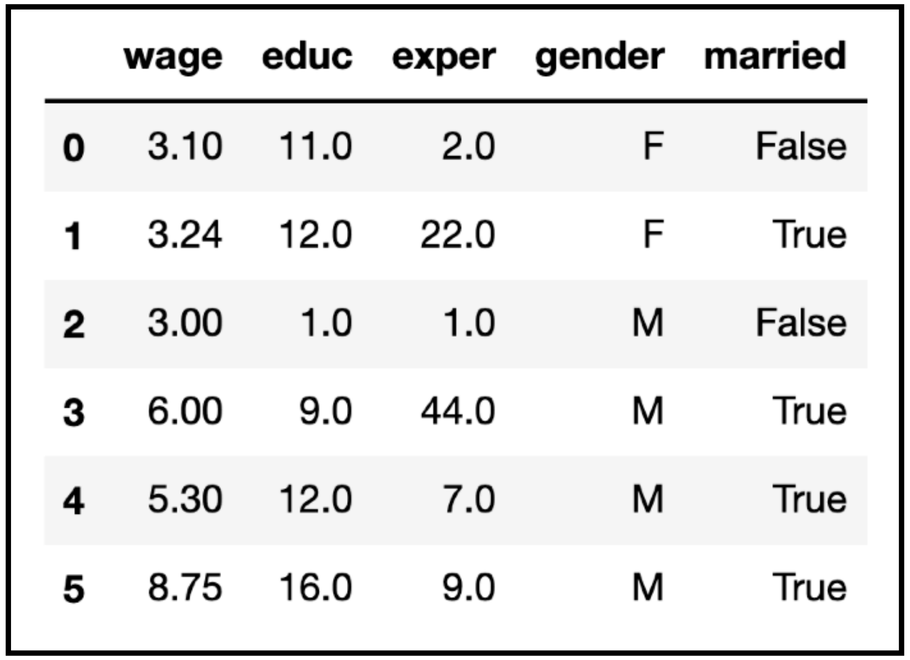


	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	1.0	1.0	M	False
3	6.00	9.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True

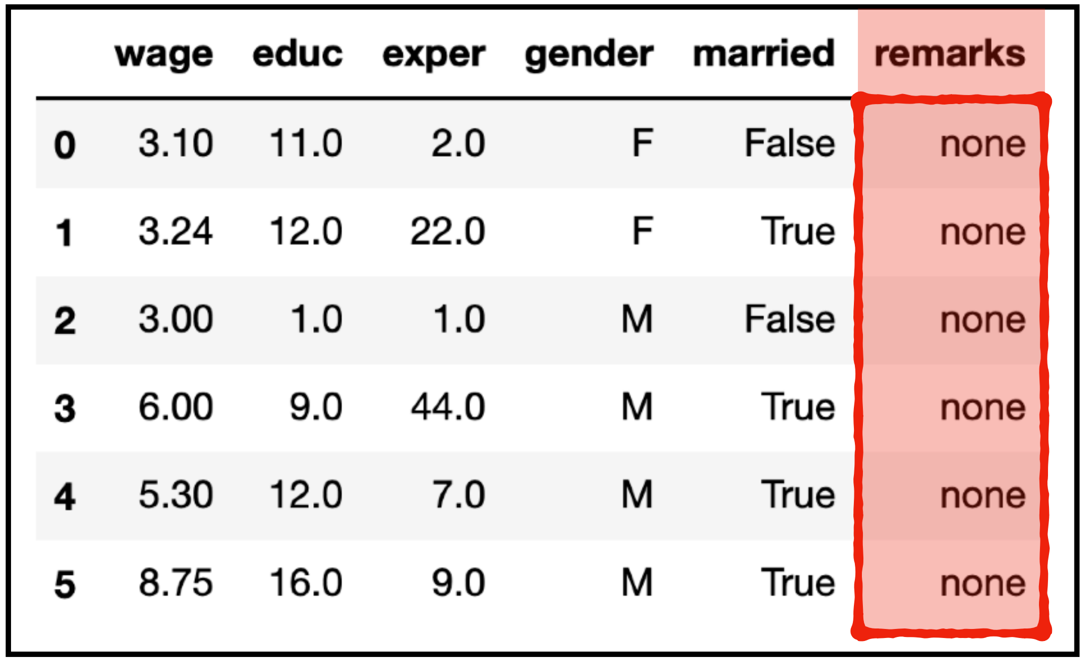
Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Change `pandas.DataFrame` object in-place

```
data_frame.loc[:, 'remarks'] = 'none'
```



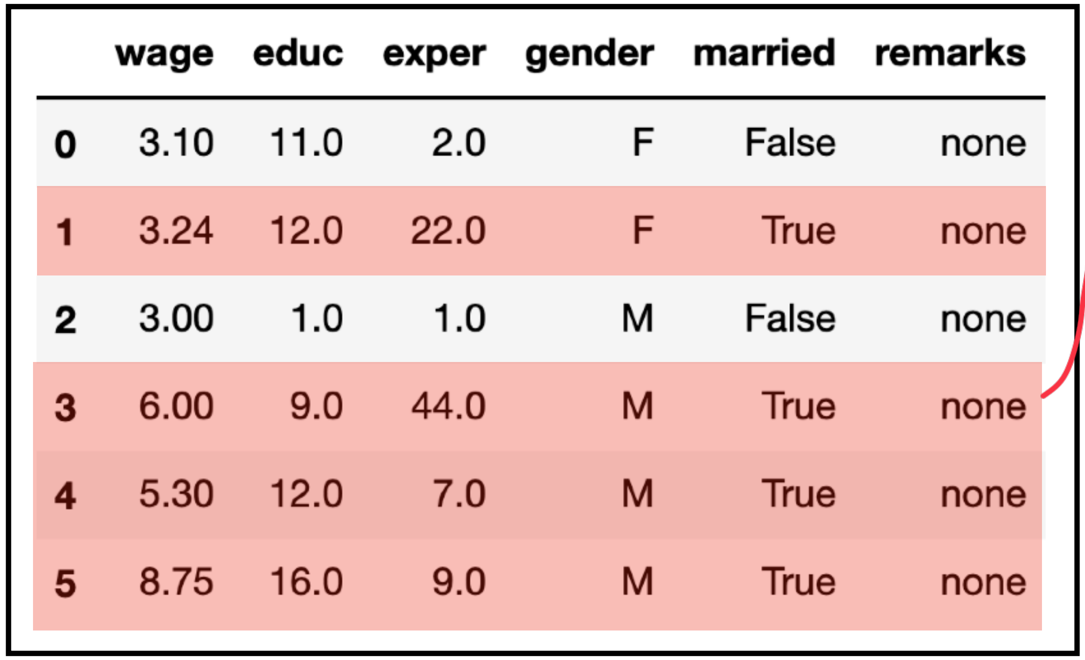
	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	1.0	1.0	M	False
3	6.00	9.0	44.0	M	True
4	5.30	12.0	7.0	M	True
5	8.75	16.0	9.0	M	True



	wage	educ	exper	gender	married	remarks
0	3.10	11.0	2.0	F	False	none
1	3.24	12.0	22.0	F	True	none
2	3.00	1.0	1.0	M	False	none
3	6.00	9.0	44.0	M	True	none
4	5.30	12.0	7.0	M	True	none
5	8.75	16.0	9.0	M	True	none

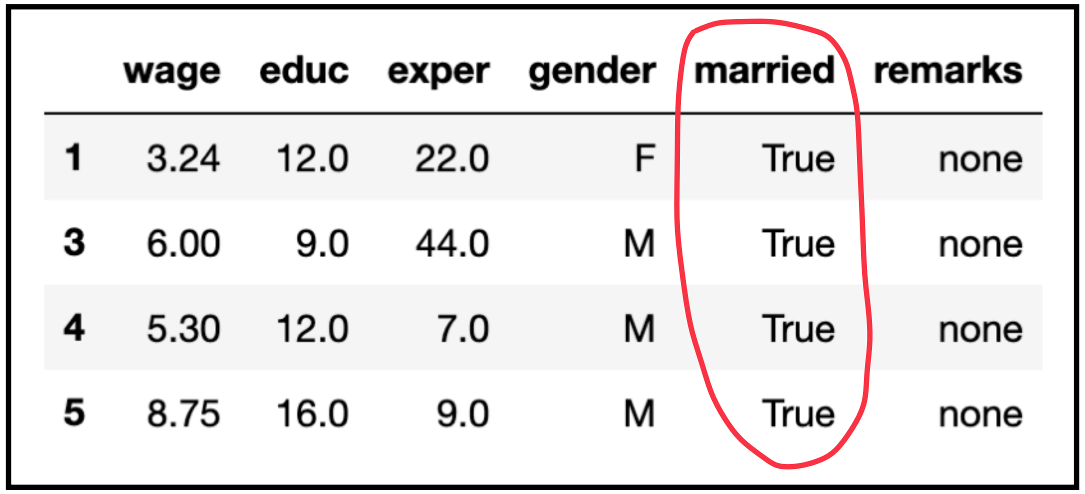
Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing



	wage	educ	exper	gender	married	remarks
0	3.10	11.0	2.0	F	False	none
1	3.24	12.0	22.0	F	True	none
2	3.00	1.0	1.0	M	False	none
3	6.00	9.0	44.0	M	True	none
4	5.30	12.0	7.0	M	True	none
5	8.75	16.0	9.0	M	True	none

Select all married cases



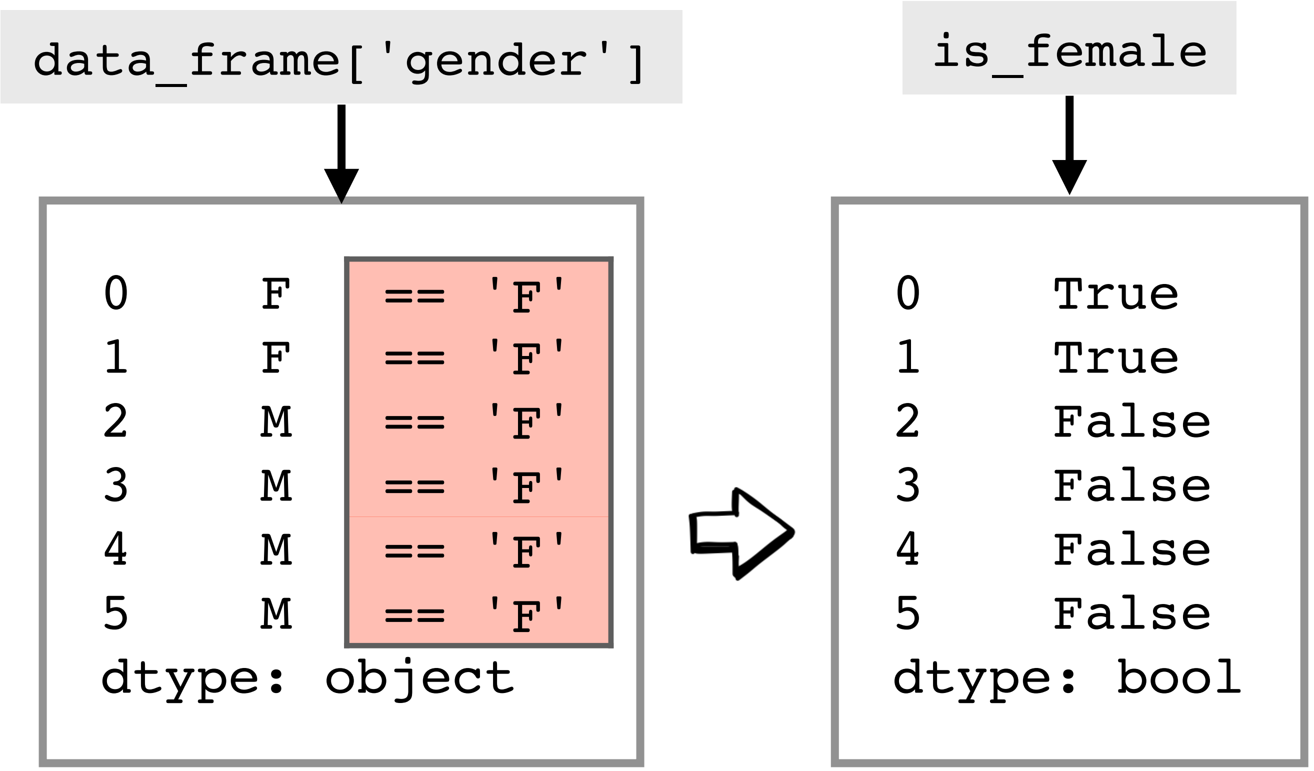
	wage	educ	exper	gender	married	remarks
1	3.24	12.0	22.0	F	True	none
3	6.00	9.0	44.0	M	True	none
4	5.30	12.0	7.0	M	True	none
5	8.75	16.0	9.0	M	True	none

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```
is_female = data_frame['gender'] == 'F'
print(is_female)
```

```
0     True
1     True
2    False
3    False
4    False
5    False
Name: gender, dtype: bool
```



```
0    F   == 'F'
1    F   == 'F'
2    M   == 'F'
3    M   == 'F'
4    M   == 'F'
5    M   == 'F'
dtype: object
```

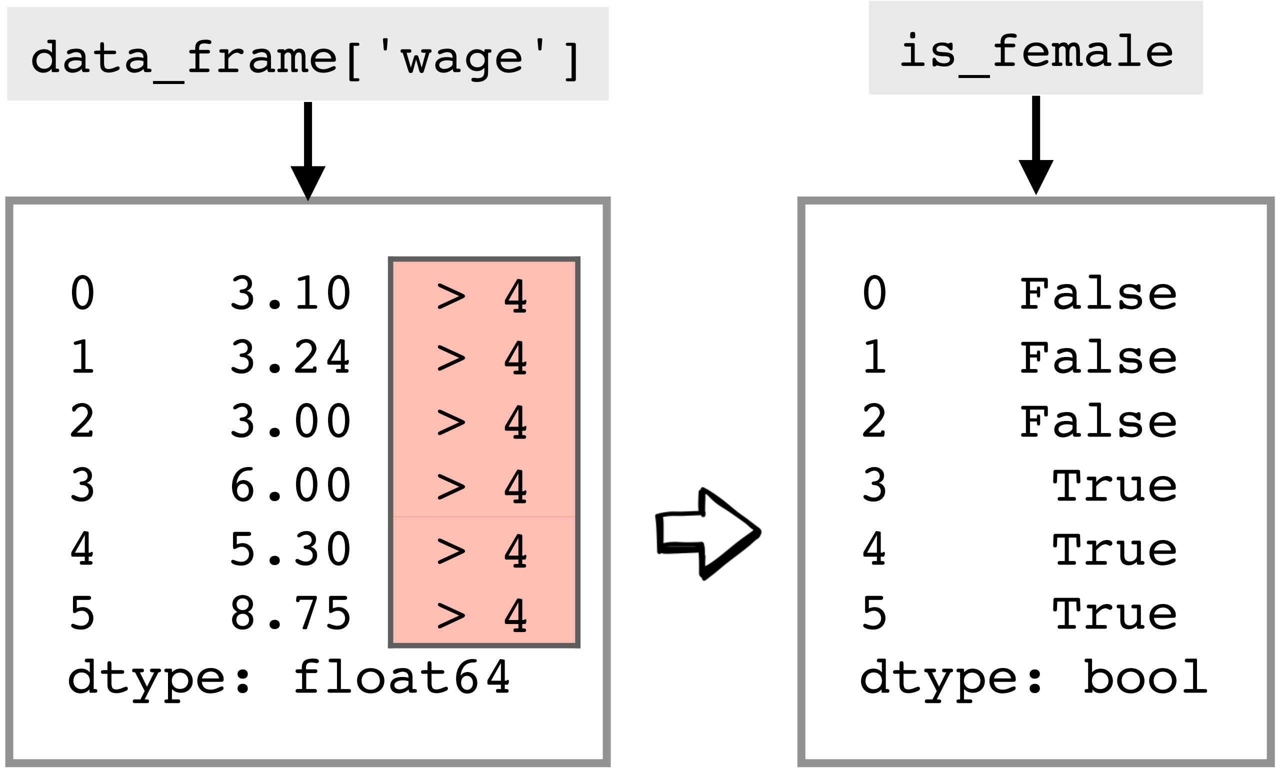
```
0     True
1     True
2    False
3    False
4    False
5    False
dtype: bool
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```
is_high_wage = data_frame['wage'] > 4
print(is_high_wage)
```

```
0     False
1     False
2     False
3      True
4      True
5      True
Name: wage, dtype: bool
```



`data_frame['wage']`

```
0     3.10   > 4
1     3.24   > 4
2     3.00   > 4
3     6.00   > 4
4     5.30   > 4
5     8.75   > 4
dtype: float64
```

`is_female`

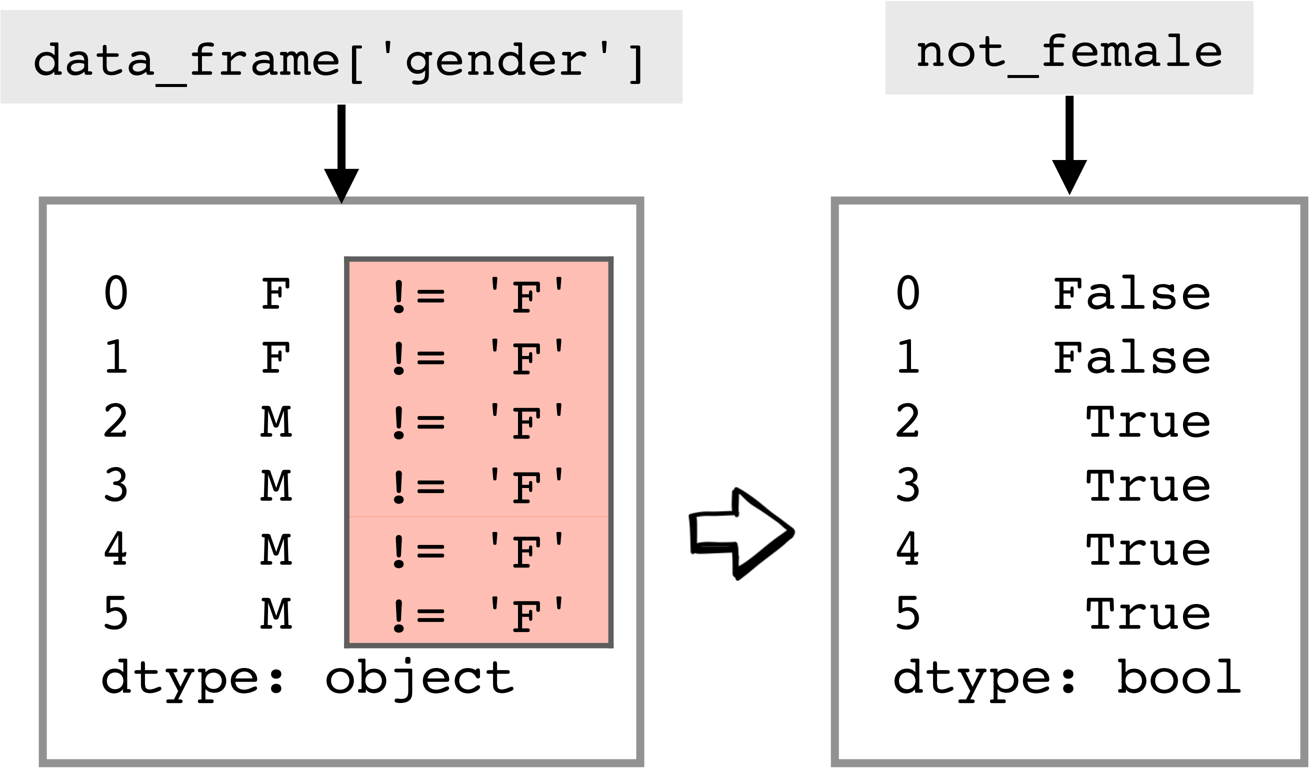
```
0     False
1     False
2     False
3      True
4      True
5      True
dtype: bool
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```
not_female = data_frame['gender'] != 'F'
print(not_female)
```

```
0     False
1     False
2      True
3      True
4      True
5      True
Name: gender, dtype: bool
```



data_frame['gender']

```
0     F   != 'F'
1     F   != 'F'
2     M   != 'F'
3     M   != 'F'
4     M   != 'F'
5     M   != 'F'
dtype: object
```

not_female

```
0     False
1     False
2      True
3      True
4      True
5      True
dtype: bool
```

Pandas for Data Analysis

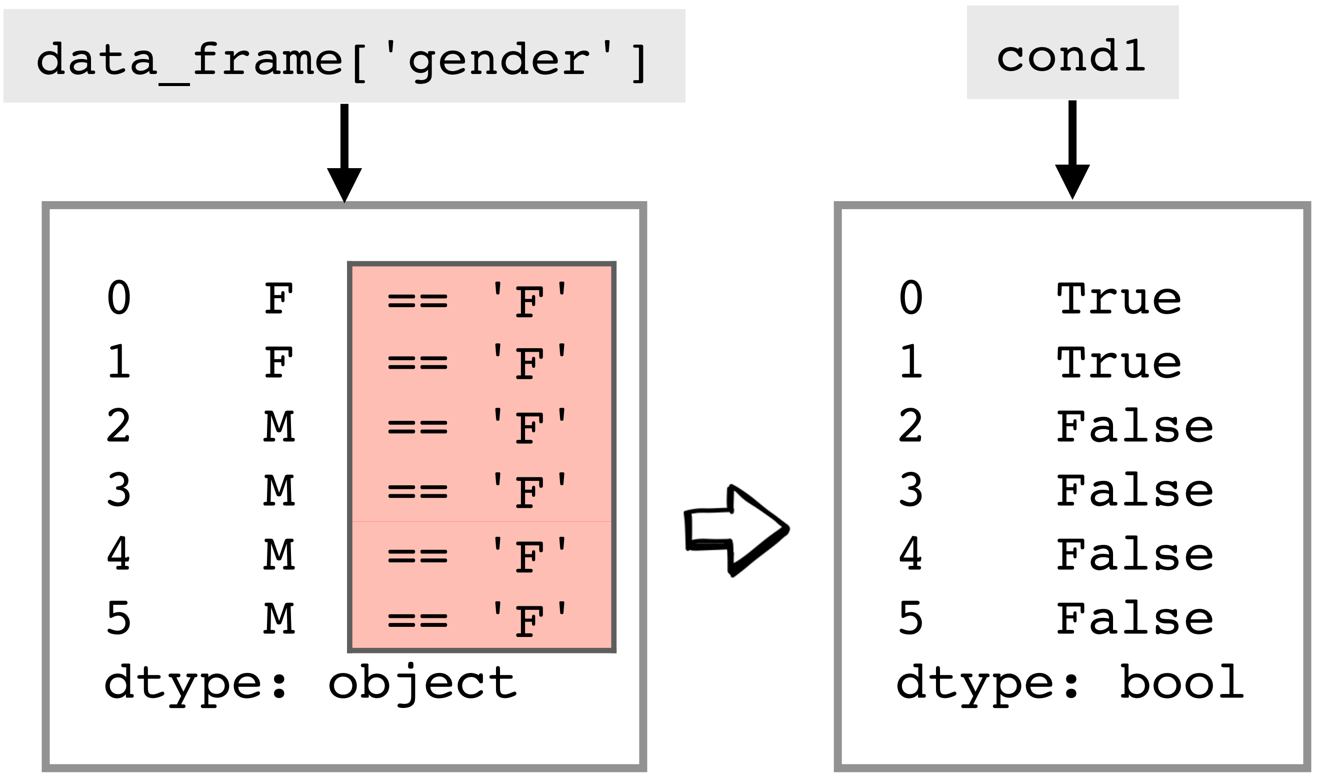
- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing
    - ✓ Bitwise "and" logic: `&`
    - ✓ Bitwise "or" logic: `|`
    - ✓ Bitwise "not" logic: `~`

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
    - Boolean series and boolean indexing

```
cond1 = data_frame['gender'] == 'F'
cond2 = data_frame['married']
is_wife = cond1 & cond2
print(is_wife)
```

```
0     False
1      True
2     False
3     False
4     False
5     False
dtype: bool
```



```
data_frame['gender']
```

```
0     F    == 'F'
1     F    == 'F'
2     M    == 'F'
3     M    == 'F'
4     M    == 'F'
5     M    == 'F'
dtype: object
```

```
cond1
```

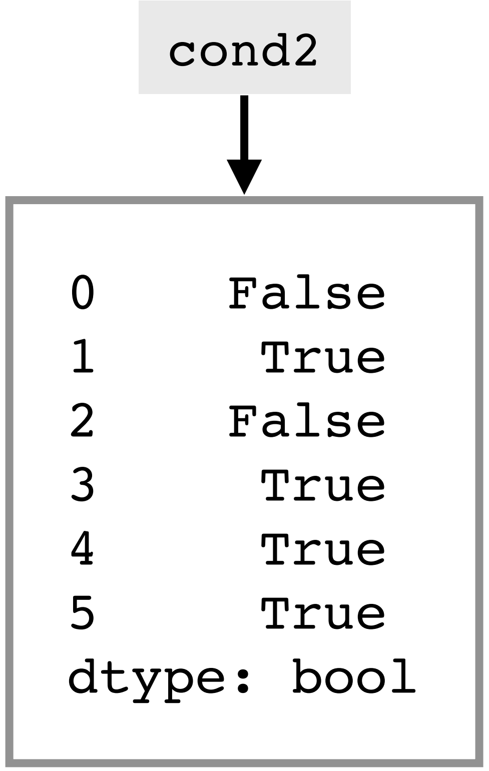
```
0     True
1     True
2     False
3     False
4     False
5     False
dtype: bool
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```
cond1 = data_frame['gender'] == 'F'
cond2 = data_frame['married']
is_wife = cond1 & cond2
print(is_wife)
```

```
0     False
1      True
2     False
3     False
4     False
5     False
dtype: bool
```



cond2

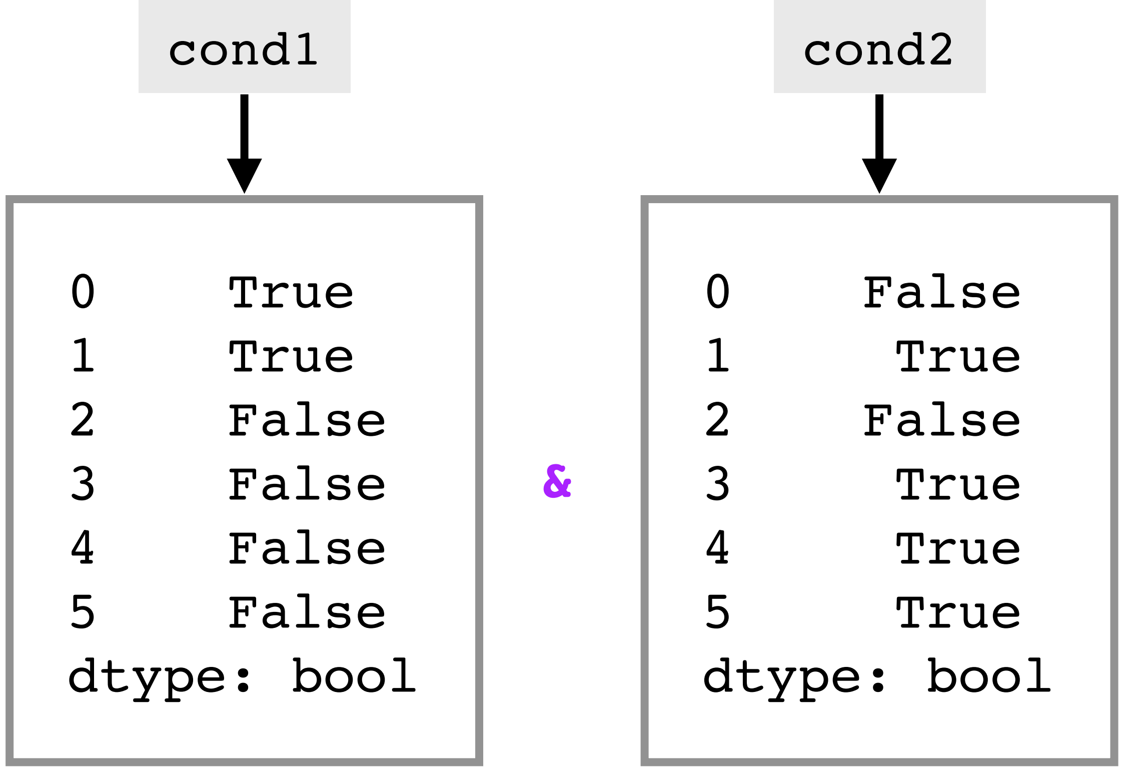
```
0     False
1      True
2     False
3      True
4      True
5      True
dtype: bool
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
    - Boolean series and boolean indexing

```
cond1 = data_frame['gender'] == 'F'
cond2 = data_frame['married']
is_wife = cond1 & cond2
print(is_wife)
```

```
0     False
1      True
2     False
3     False
4     False
5     False
dtype: bool
```



cond1

```
0     True
1     True
2     False
3     False
4     False
5     False
dtype: bool
```

&

cond2

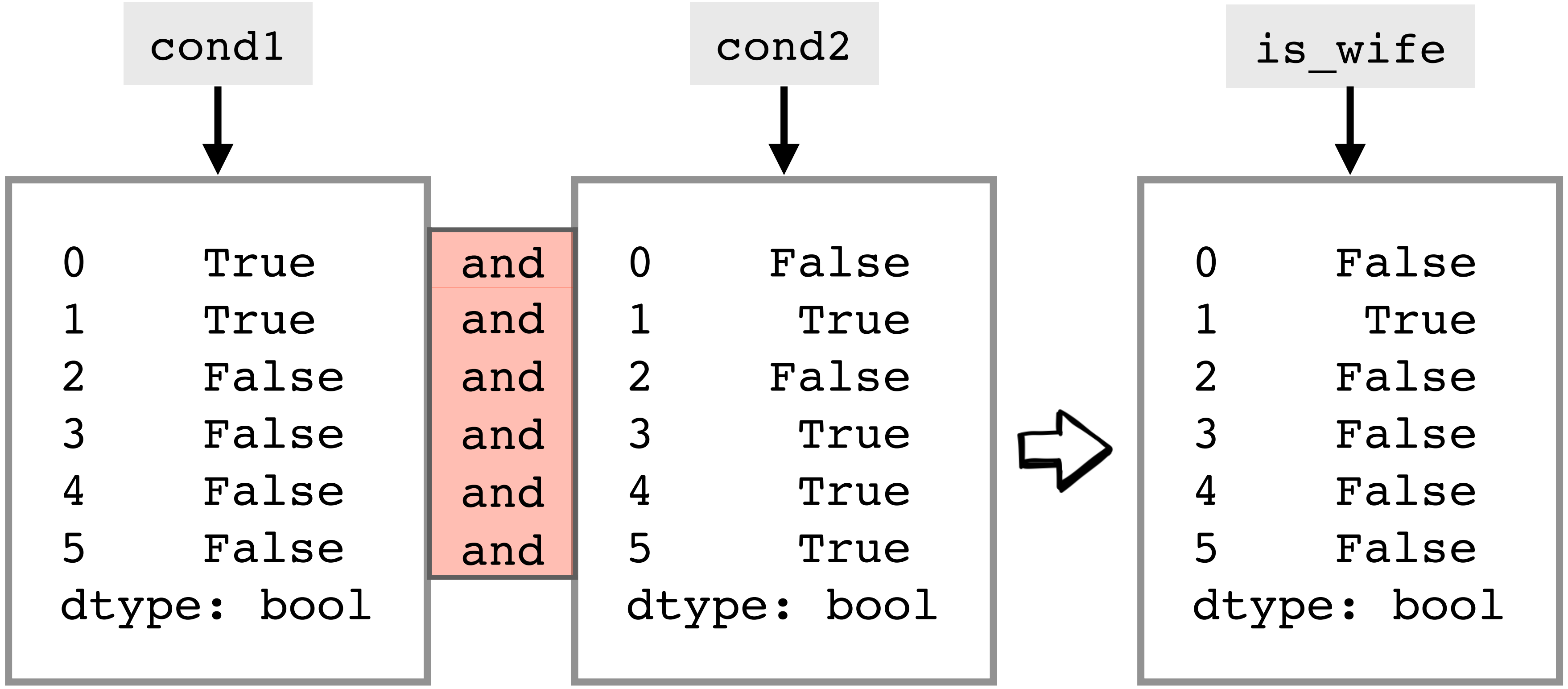
```
0     False
1      True
2     False
3      True
4      True
5      True
dtype: bool
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```
cond1 = data_frame['gender'] == ‘F'
cond2 = data_frame['married']
is_wife = cond1 & cond2
print(is_wife)
```

```
0     False
1      True
2     False
3     False
4     False
5     False
dtype: bool
```



cond1

```
0     True
1     True
2     False
3     False
4     False
5     False
dtype: bool
```

and
and
and
and
and
and

cond2

```
0     False
1      True
2     False
3      True
4      True
5      True
dtype: bool
```

is_wife

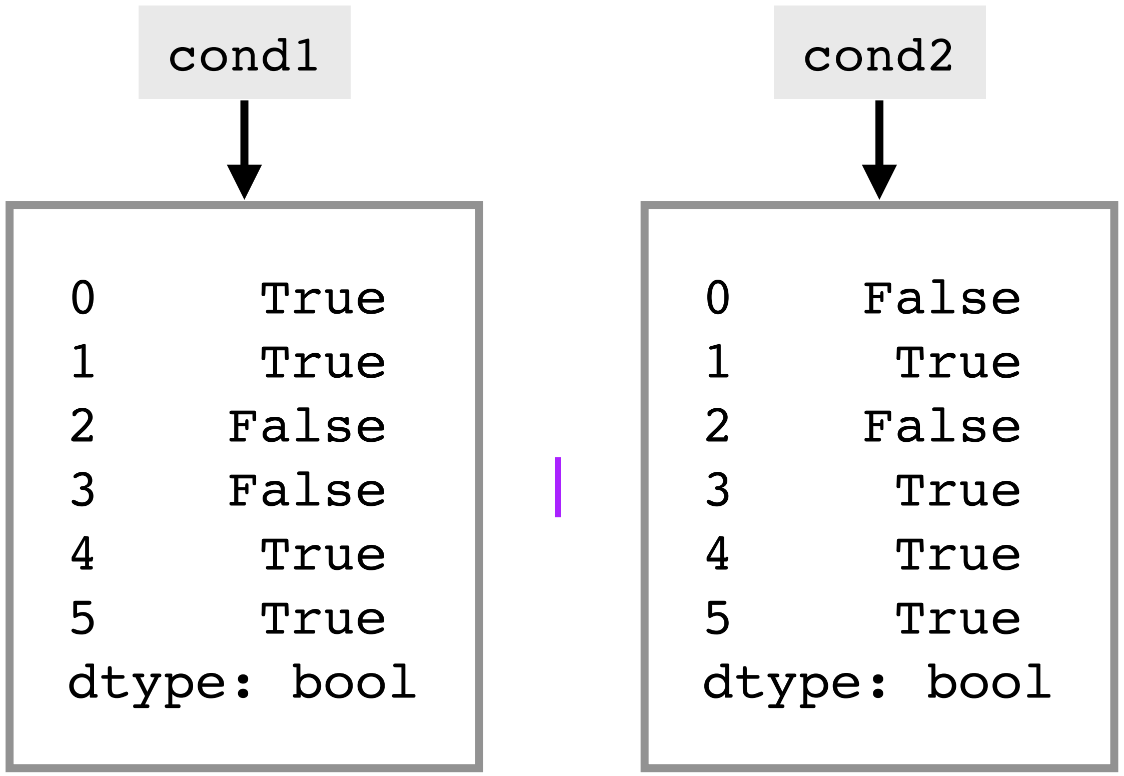
```
0     False
1      True
2     False
3     False
4     False
5     False
dtype: bool
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```python
cond1 = data_frame['educ'] > 9
cond2 = data_frame['exper'] > 3
is_skillful = cond1 | cond2
print(is_skillful)
```

```
0      True
1      True
2     False
3      True
4      True
5      True
dtype: bool
```



cond1

```
0      True
1      True
2     False
3     False
4      True
5      True
dtype: bool
```

|

cond2

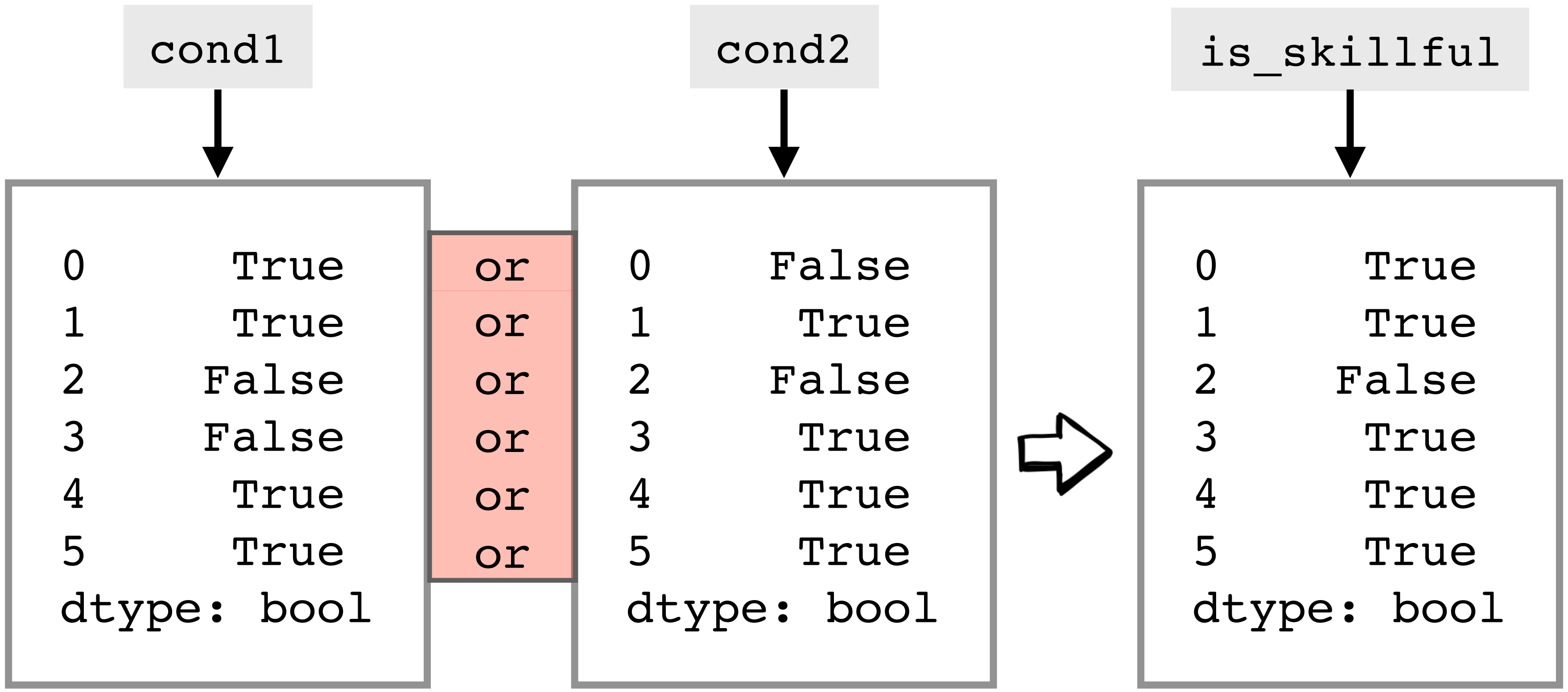
```
0     False
1      True
2     False
3      True
4      True
5      True
dtype: bool
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```
cond1 = data_frame['educ'] > 9
cond2 = data_frame['exper'] > 3
is_skillful = cond1 | cond2
print(is_skillful)
```

```
0      True
1      True
2     False
3      True
4      True
5      True
dtype: bool
```



cond1

```
0      True
1      True
2     False
3     False
4      True
5      True
dtype: bool
```

cond2

```
0     False
1      True
2     False
3      True
4      True
5      True
dtype: bool
```

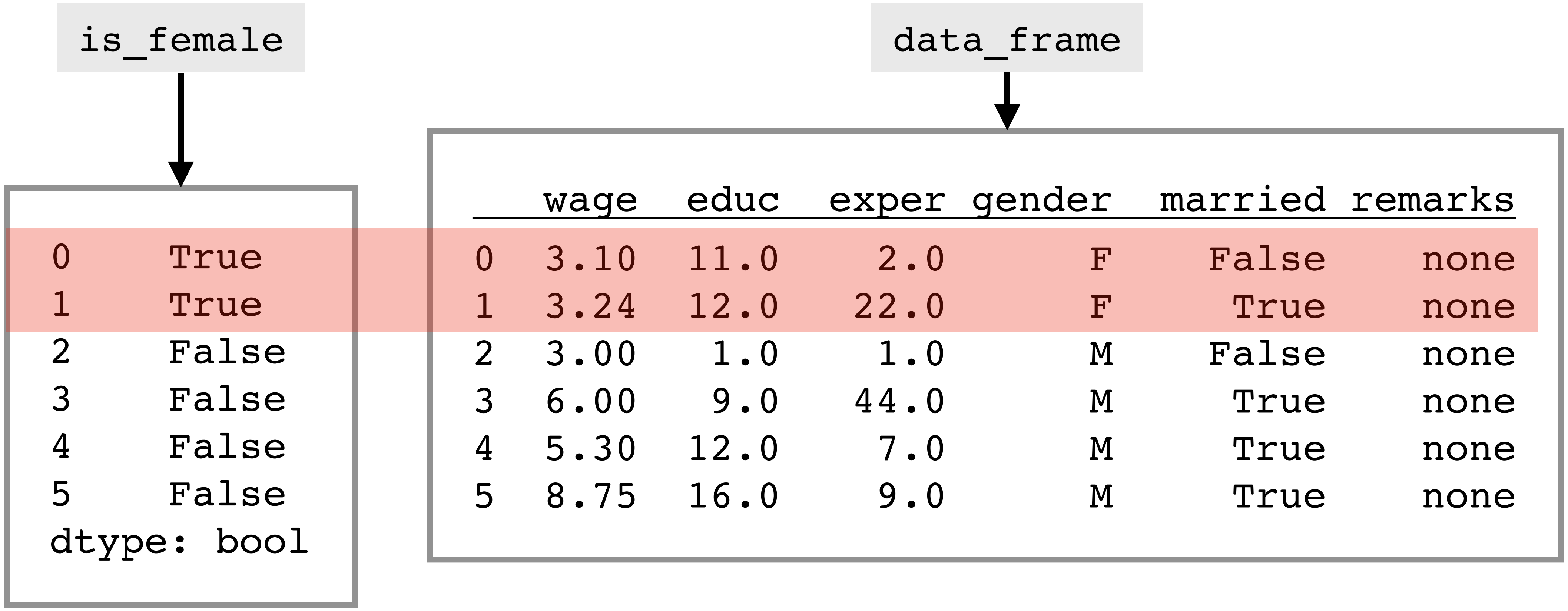
is_skillful

```
0      True
1      True
2     False
3      True
4      True
5      True
dtype: bool
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```
is_female = data_frame['gender'] == 'F'
females = data_frame.loc[is_female]
```



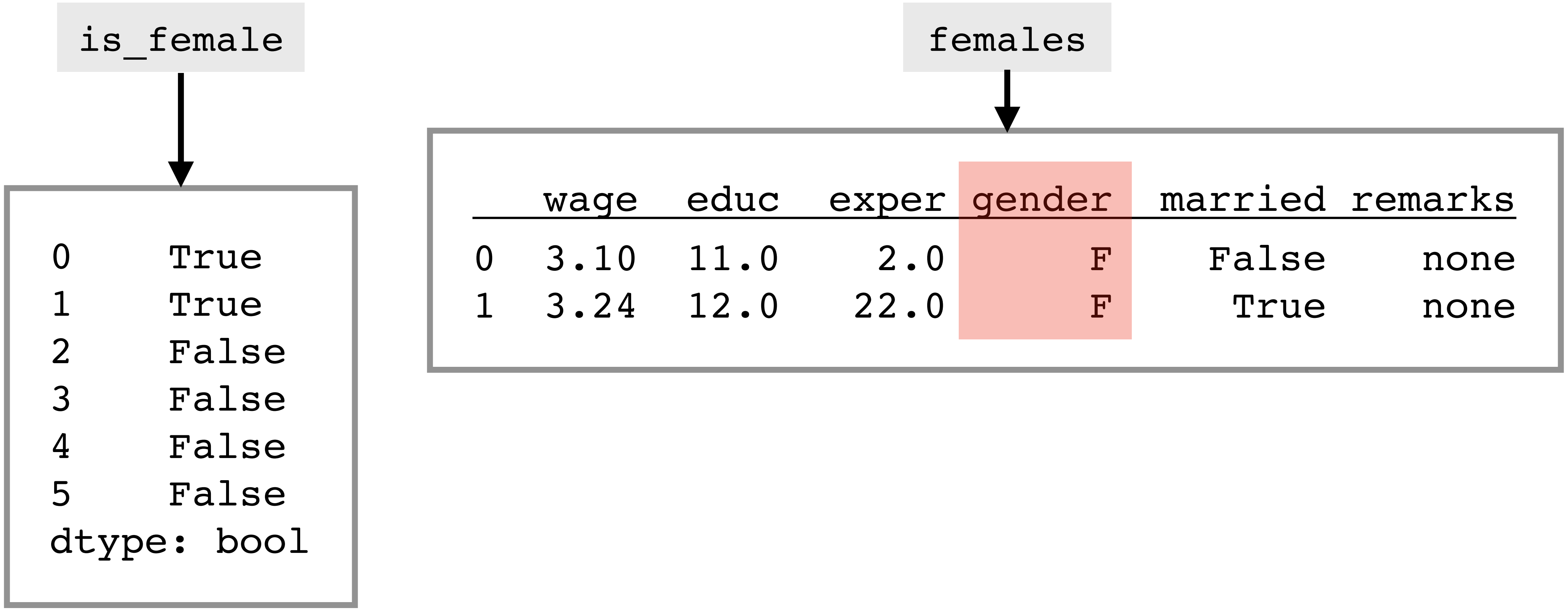
```
0     True
1     True
2    False
3    False
4    False
5    False
dtype: bool
```

```
   wage  educ  exper gender  married remarks
0  3.10  11.0    2.0      F    False    none
1  3.24  12.0   22.0      F     True    none
2  3.00   1.0    1.0      M    False    none
3  6.00   9.0   44.0      M     True    none
4  5.30  12.0    7.0      M     True    none
5  8.75  16.0    9.0      M     True    none
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
    - Boolean series and boolean indexing

```
is_female = data_frame['gender'] == 'F'
females = data_frame.loc[is_female]
```



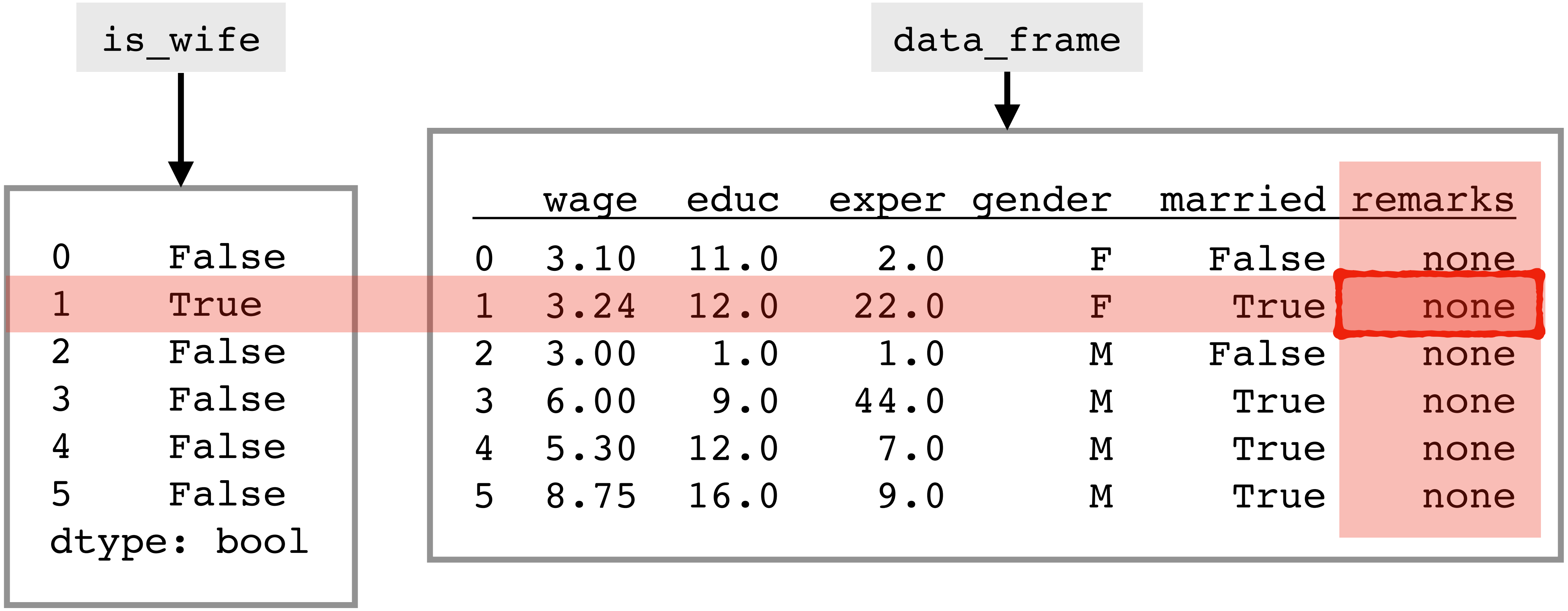
```
0      True
1      True
2      False
3      False
4      False
5      False
dtype: bool
```

```
   wage  educ  exper gender  married remarks
0  3.10  11.0    2.0      F    False    none
1  3.24  12.0   22.0      F     True    none
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```
cond1 = data_frame['gender'] == ‘F'
cond2 = data_frame['married']
is_wife = cond1 & cond2
data_frame.loc[is_wife, 'remarks'] = 'Wife'
```



`is_wife`

```
0      False
1      True
2      False
3      False
4      False
5      False
dtype: bool
```

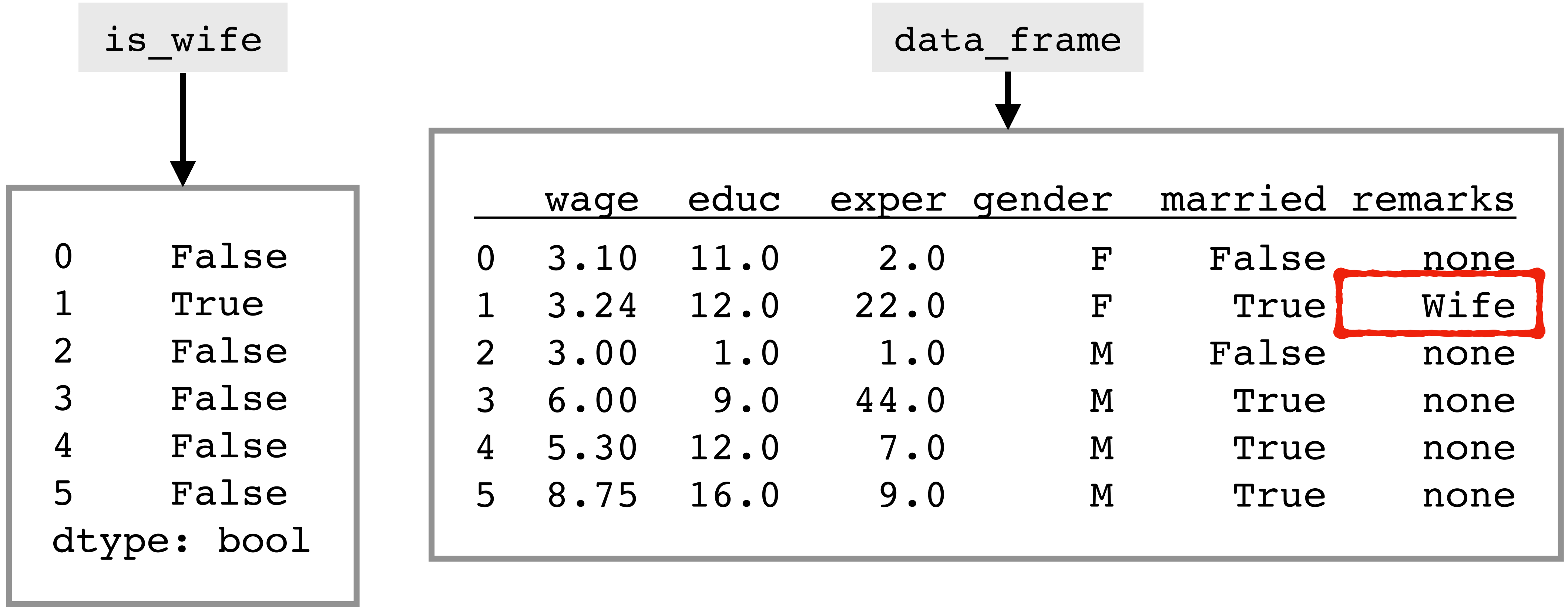
`data_frame`

```
   wage  educ  exper gender  married remarks
0  3.10  11.0    2.0      F    False    none
1  3.24  12.0   22.0      F     True    none
2  3.00   1.0    1.0      M    False    none
3  6.00   9.0   44.0      M     True    none
4  5.30  12.0    7.0      M     True    none
5  8.75  16.0    9.0      M     True    none
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

```
cond1 = data_frame['gender'] == ‘F'
cond2 = data_frame['married']
is_wife = cond1 & cond2
data_frame.loc[is_wife, 'remarks'] = 'Wife'
```



is_wife

```
0       False
1       True
2       False
3       False
4       False
5       False
dtype: bool
```

data_frame

```
    wage   educ   exper  gender  married  remarks
0   3.10   11.0     2.0       F    False     none
1   3.24   12.0    22.0       F     True     Wife
2   3.00    1.0     1.0       M    False     none
3   6.00    9.0    44.0       M     True     none
4   5.30   12.0     7.0       M     True     none
5   8.75   16.0     9.0       M     True     none
```

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

Question 1: Make changes to the data frame above such that:

- Values for married males in the column **`'remarks'`** are changed to the string **`'Husband'`**.
- Values for unmarried males or unmarried females in the column **`'remarks'`** are changed to the string **`'Single'`**.

Pandas for Data Analysis

- The `pandas.DataFrame` data structure
  - Boolean series and boolean indexing

Question 1: Make changes to the data frame above such that:

- Values for married males in the column **`'remarks'`** are changed to the string **`'Husband'`**.
- Values for unmarried males or unmarried females in the column **`'remarks'`** are changed to the string **`'Single'`**.

```
cond1 = data_frame['gender'] == 'F'
cond2 = data_frame['married']
is_husband = ~cond1 & cond2
data_frame.loc[is_husband, 'remarks'] = 'Husband'
is_single = ~cond2
data_frame.loc[is_single, 'remarks'] = 'Single'
```

Pandas for Data Analysis

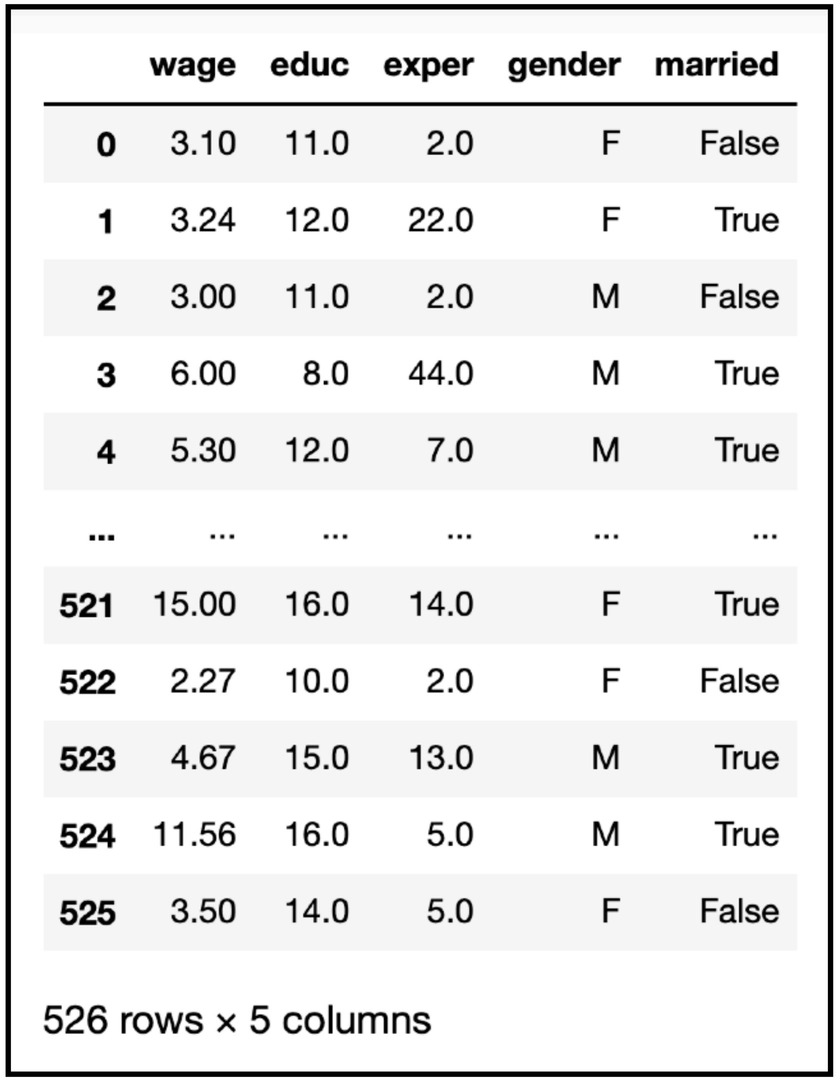
- Read data from files

```python
data = pd.read_csv('wage.csv')
```

```python
print(data.head(6))
```

The first six records

```
   wage  educ  exper gender  married
0  3.10  11.0    2.0      F    False
1  3.24  12.0   22.0      F     True
2  3.00  11.0    2.0      M    False
3  6.00   8.0   44.0      M     True
4  5.30  12.0    7.0      M     True
5  8.75  16.0    9.0      M     True
```

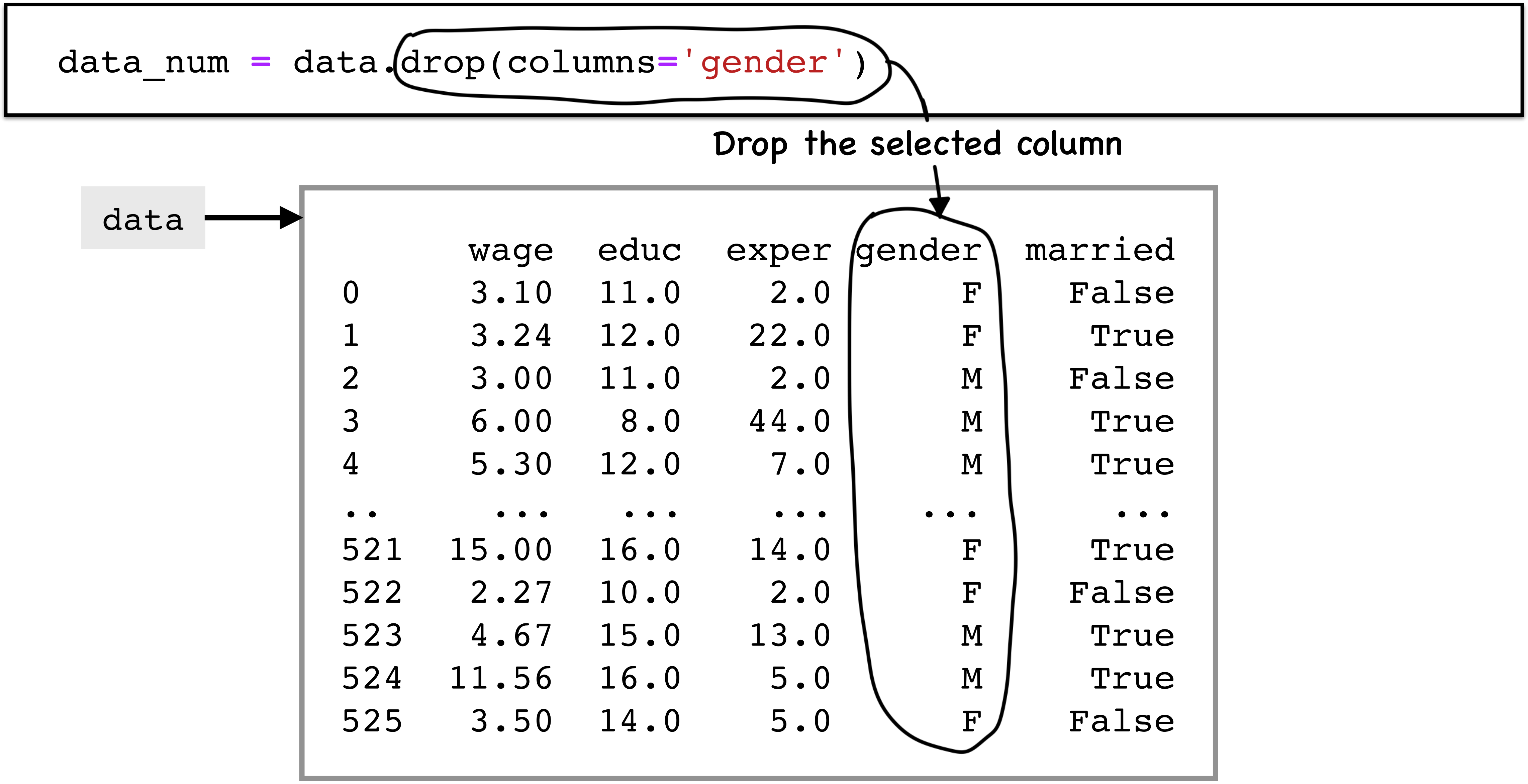


	wage	educ	exper	gender	married
0	3.10	11.0	2.0	F	False
1	3.24	12.0	22.0	F	True
2	3.00	11.0	2.0	M	False
3	6.00	8.0	44.0	M	True
4	5.30	12.0	7.0	M	True
...	...	...	...	...	...
521	15.00	16.0	14.0	F	True
522	2.27	10.0	2.0	F	False
523	4.67	15.0	13.0	M	True
524	11.56	16.0	5.0	M	True
525	3.50	14.0	5.0	F	False

526 rows × 5 columns

Basics of Descriptive Analytics

- Descriptive measures



```
data_num = data.drop(columns='gender')
```

Drop the selected column

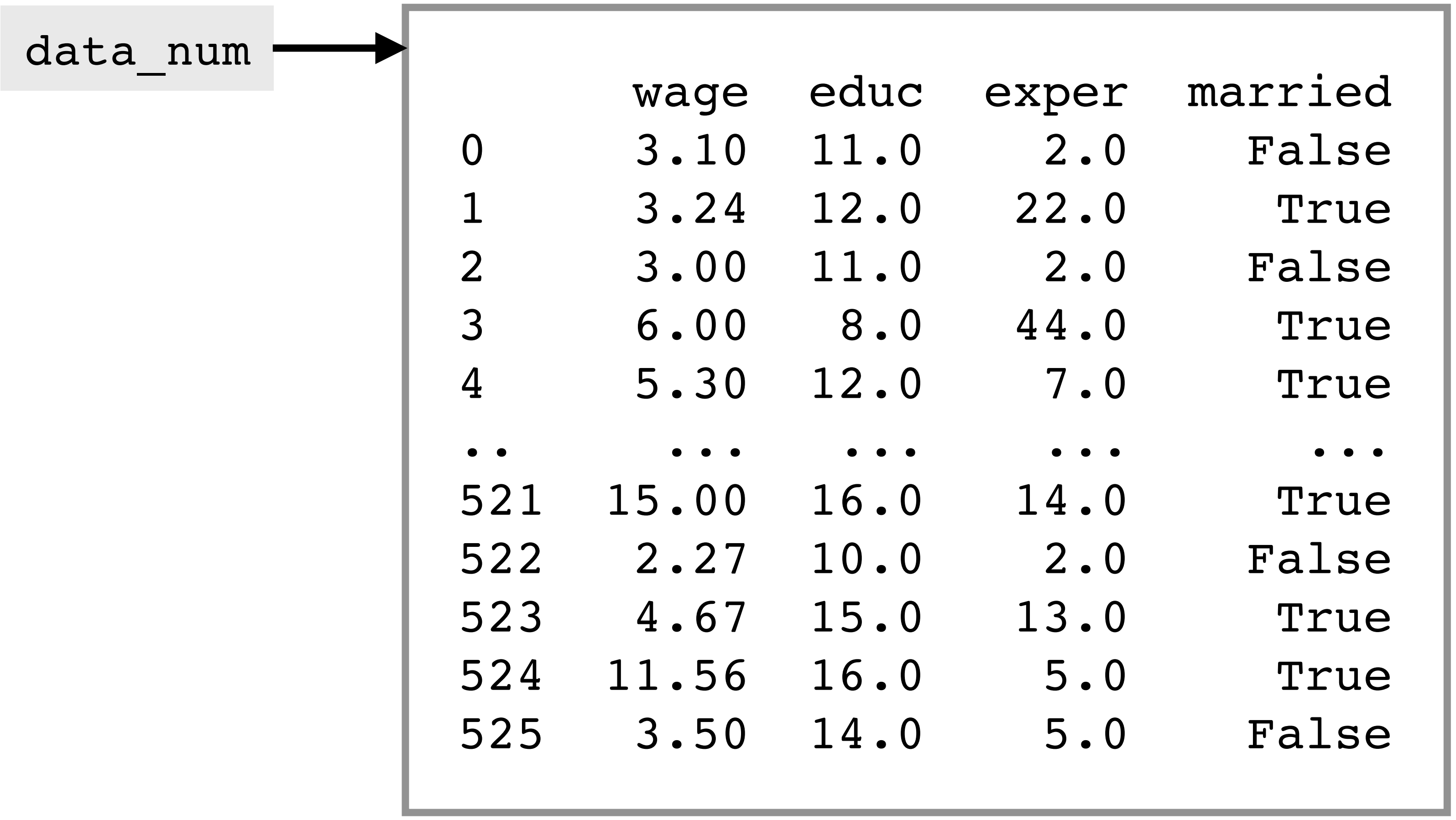
data

```
        wage  educ  exper gender  married
0       3.10  11.0    2.0      F    False
1       3.24  12.0   22.0      F     True
2       3.00  11.0    2.0      M    False
3       6.00   8.0   44.0      M     True
4       5.30  12.0    7.0      M     True
..       ...   ...    ...    ...      ...
521    15.00  16.0   14.0      F     True
522     2.27  10.0    2.0      F    False
523     4.67  15.0   13.0      M     True
524    11.56  16.0    5.0      M     True
525     3.50  14.0    5.0      F    False
```

Basics of Descriptive Analytics

- Descriptive measures

```
data_num = data.drop(columns='gender')
```



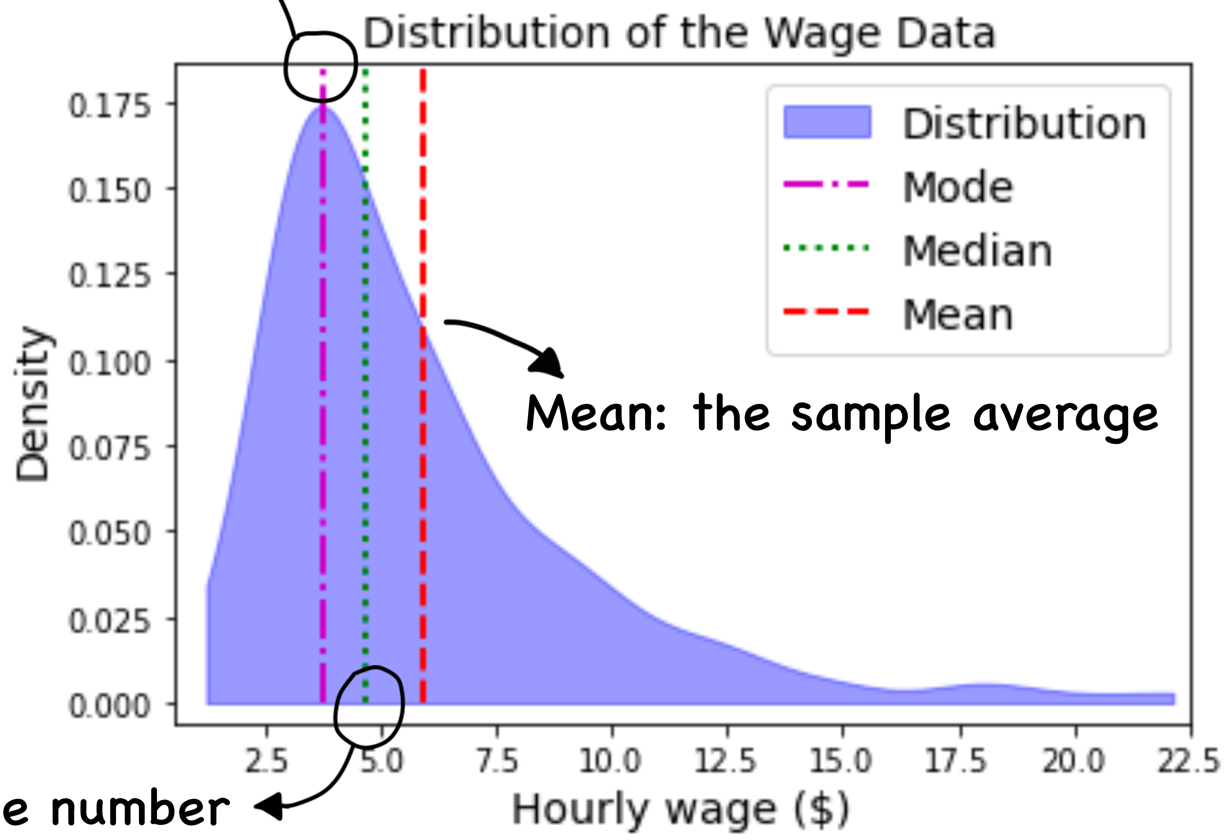
data_num

```
        wage  educ  exper  married
0       3.10  11.0    2.0    False
1       3.24  12.0   22.0     True
2       3.00  11.0    2.0    False
3       6.00   8.0   44.0     True
4       5.30  12.0    7.0     True
..       ...   ...    ...      ...
521    15.00  16.0   14.0     True
522     2.27  10.0    2.0    False
523     4.67  15.0   13.0     True
524    11.56  16.0    5.0     True
525     3.50  14.0    5.0    False
```

Basics of Descriptive Analytics

- Descriptive measures
  - Measures of centers
    - ✓ Mean
    - ✓ Median
    - ✓ Mode

Mode: the most frequent value



Mean: the sample average

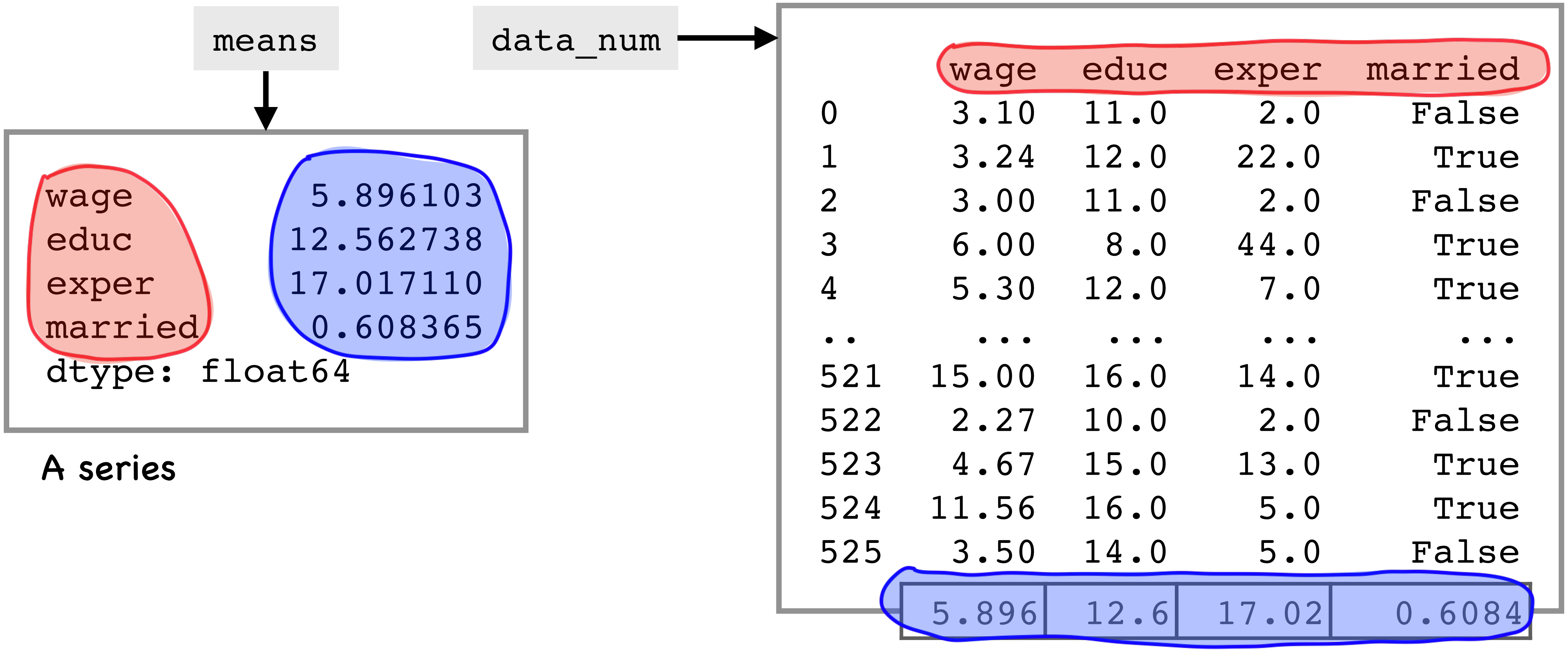
Median: the middle number in the ordered dataset

Basics of Descriptive Analytics

- Descriptive measures
  - Measures of centers

```
means = data_num.mean()
```

Mean values



```
wage        5.896103
educ       12.562738
exper      17.017110
married     0.608365
dtype: float64
```

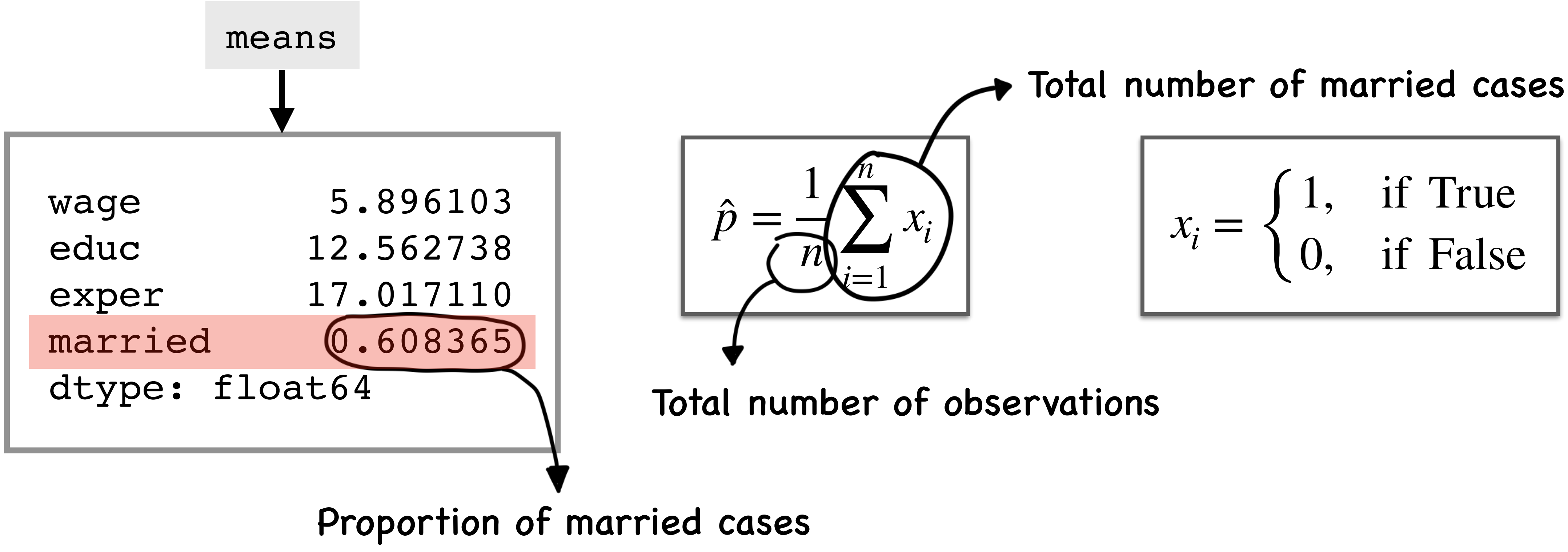
A series

```
      wage  educ  exper  married
0     3.10  11.0    2.0    False
1     3.24  12.0   22.0     True
2     3.00  11.0    2.0    False
3     6.00   8.0   44.0     True
4     5.30  12.0    7.0     True
..     ...   ...    ...      ...
521  15.00  16.0   14.0     True
522   2.27  10.0    2.0    False
523   4.67  15.0   13.0     True
524  11.56  16.0    5.0     True
525   3.50  14.0    5.0    False
     5.896  12.6  17.02   0.6084
```

Basics of Descriptive Analytics

- Descriptive measures
  - Measures of centers

```
means = data_num.mean()
```



means

```
wage         5.896103
educ        12.562738
exper       17.017110
married      0.608365
dtype: float64
```

Proportion of married cases

$$\hat{p} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Total number of married cases

Total number of observations

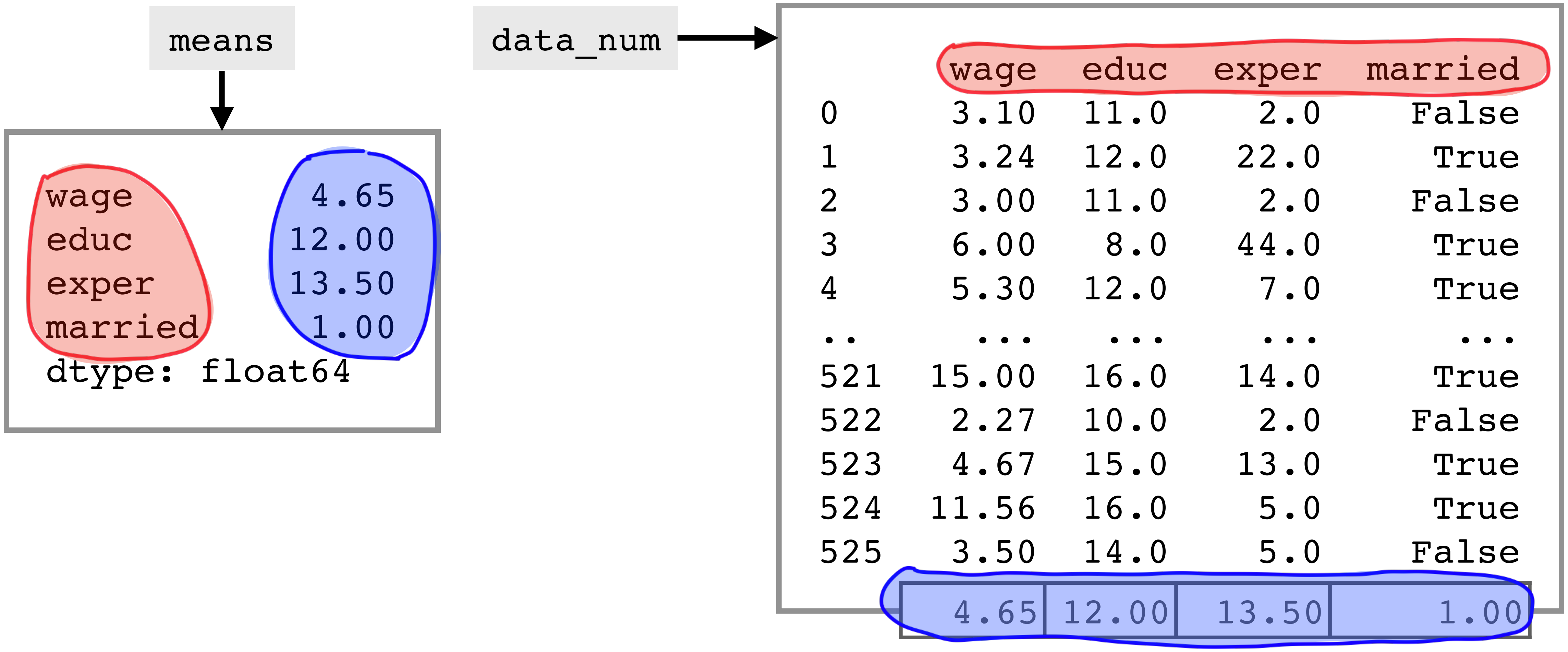
$$x_i = \begin{cases} 1, & \text{if True} \\ 0, & \text{if False} \end{cases}$$

Basics of Descriptive Analytics

- Descriptive measures
  - Measures of centers

```
medians = data_num.median()
```

→ Median values



```
wage        4.65
educ       12.00
exper      13.50
married     1.00
dtype: float64
```

```
      wage  educ  exper  married
0     3.10  11.0    2.0    False
1     3.24  12.0   22.0     True
2     3.00  11.0    2.0    False
3     6.00   8.0   44.0     True
4     5.30  12.0    7.0     True
..     ...   ...    ...      ...
521  15.00  16.0   14.0     True
522   2.27  10.0    2.0    False
523   4.67  15.0   13.0     True
524  11.56  16.0    5.0     True
525   3.50  14.0    5.0    False
      4.65 12.00  13.50     1.00
```

Basics of Descriptive Analytics

- Descriptive measures
  - Measures of variations
    - ✓ Variance
    - ✓ Standard deviation

```
data_num.var()
```

```
wage         13.638884
educ          7.667485
exper       184.203516
married       0.238711
dtype: float64
```

```
data_num.std()
```

```
wage          3.693086
educ          2.769022
exper        13.572160
married       0.488580
dtype: float64
```

Basics of Descriptive Analytics

- Descriptive measures
  - Extreme points
    - ✓ Minimum value
    - ✓ Maximum value

```
data.min()
```

```
wage          0.53
educ           0.0
exper          1.0
gender           F
married      False
dtype: object
```

```
data.max()
```

```
wage         24.98
educ          18.0
exper         51.0
gender           M
married       True
dtype: object
```

Basics of Descriptive Analytics

- Descriptive measures
  - Counts of categorical values

```
data['gender'].value_counts()
```

```
M     274
F     252
Name: gender, dtype: int64
```

Counts (frequencies) of all unique values

```
data['gender'].value_counts(normalize=True)
```

```
M     0.520913
F     0.479087
Name: gender, dtype: float64
```

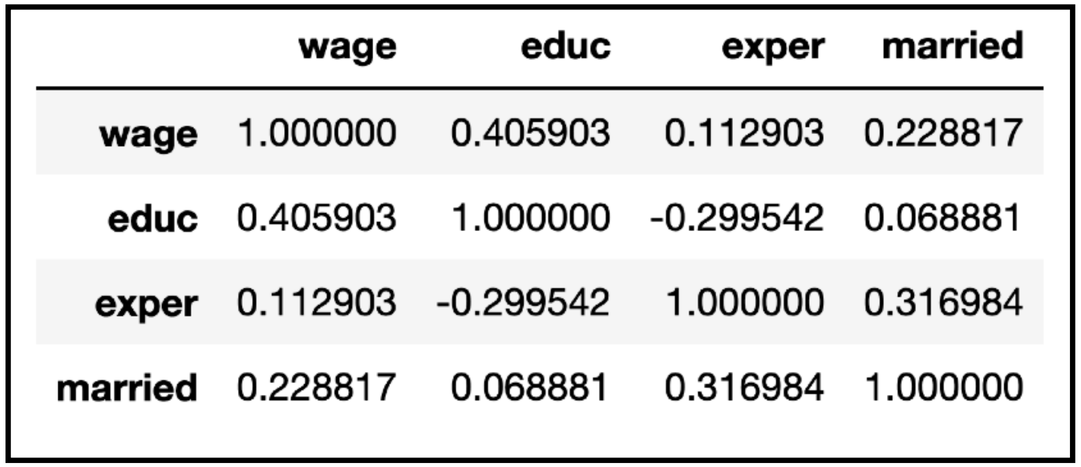
Proportions of all unique values

Basics of Descriptive Analytics

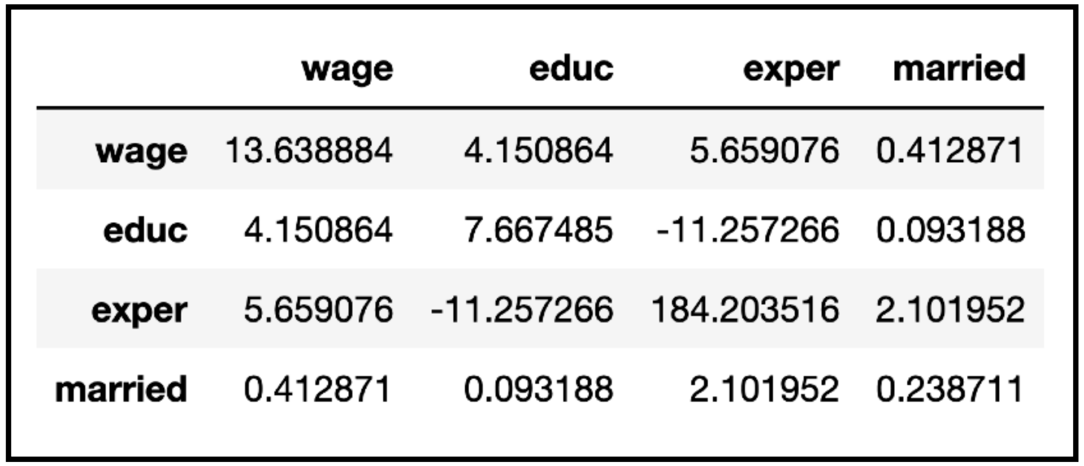
- Descriptive measures
  - The `corr()` and `cov()` methods

```
data.corr()
```

```
data.cov()
```



	wage	educ	exper	married
wage	1.000000	0.405903	0.112903	0.228817
educ	0.405903	1.000000	-0.299542	0.068881
exper	0.112903	-0.299542	1.000000	0.316984
married	0.228817	0.068881	0.316984	1.000000

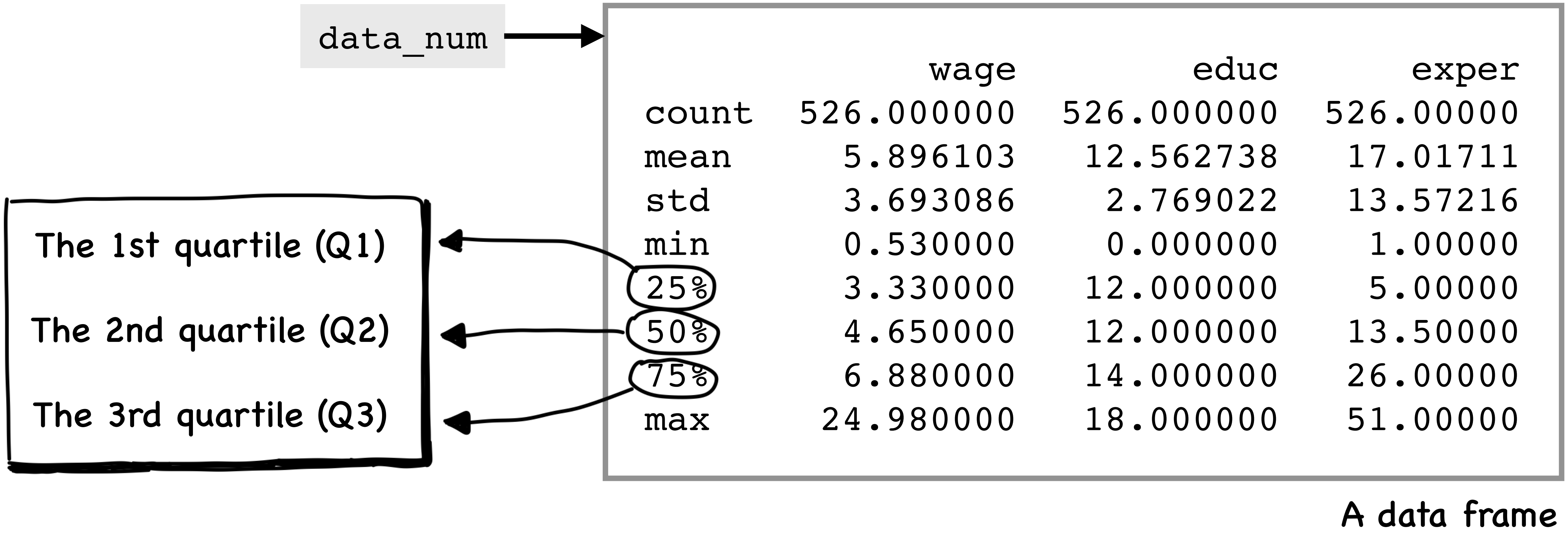


	wage	educ	exper	married
wage	13.638884	4.150864	5.659076	0.412871
educ	4.150864	7.667485	-11.257266	0.093188
exper	5.659076	-11.257266	184.203516	2.101952
married	0.412871	0.093188	2.101952	0.238711

Basics of Descriptive Analytics

- Descriptive measures
  - The `describe()` method

```
wage_summary = data.describe()
```



`data_num`

```
              wage         educ      exper
count   526.000000   526.000000  526.00000
mean      5.896103    12.562738   17.01711
std       3.693086     2.769022   13.57216
min       0.530000     0.000000    1.00000
25%       3.330000    12.000000    5.00000
50%       4.650000    12.000000   13.50000
75%       6.880000    14.000000   26.00000
max      24.980000    18.000000   51.00000
```

The 1st quartile (Q1)

The 2nd quartile (Q2)

The 3rd quartile (Q3)

A data frame

Basics of Descriptive Analytics

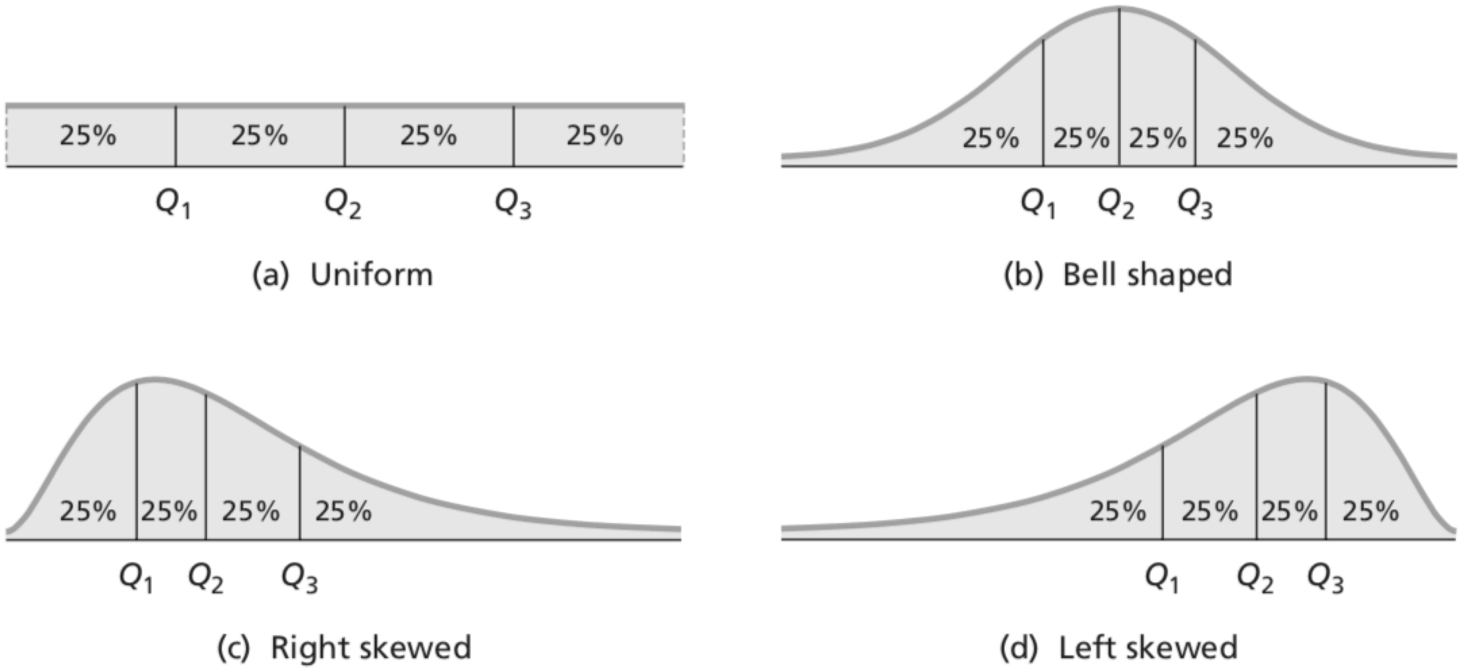
- Descriptive measures
  - The `describe()` method

```
wage_summary = data.describe()
```

The 1st quartile (Q1)

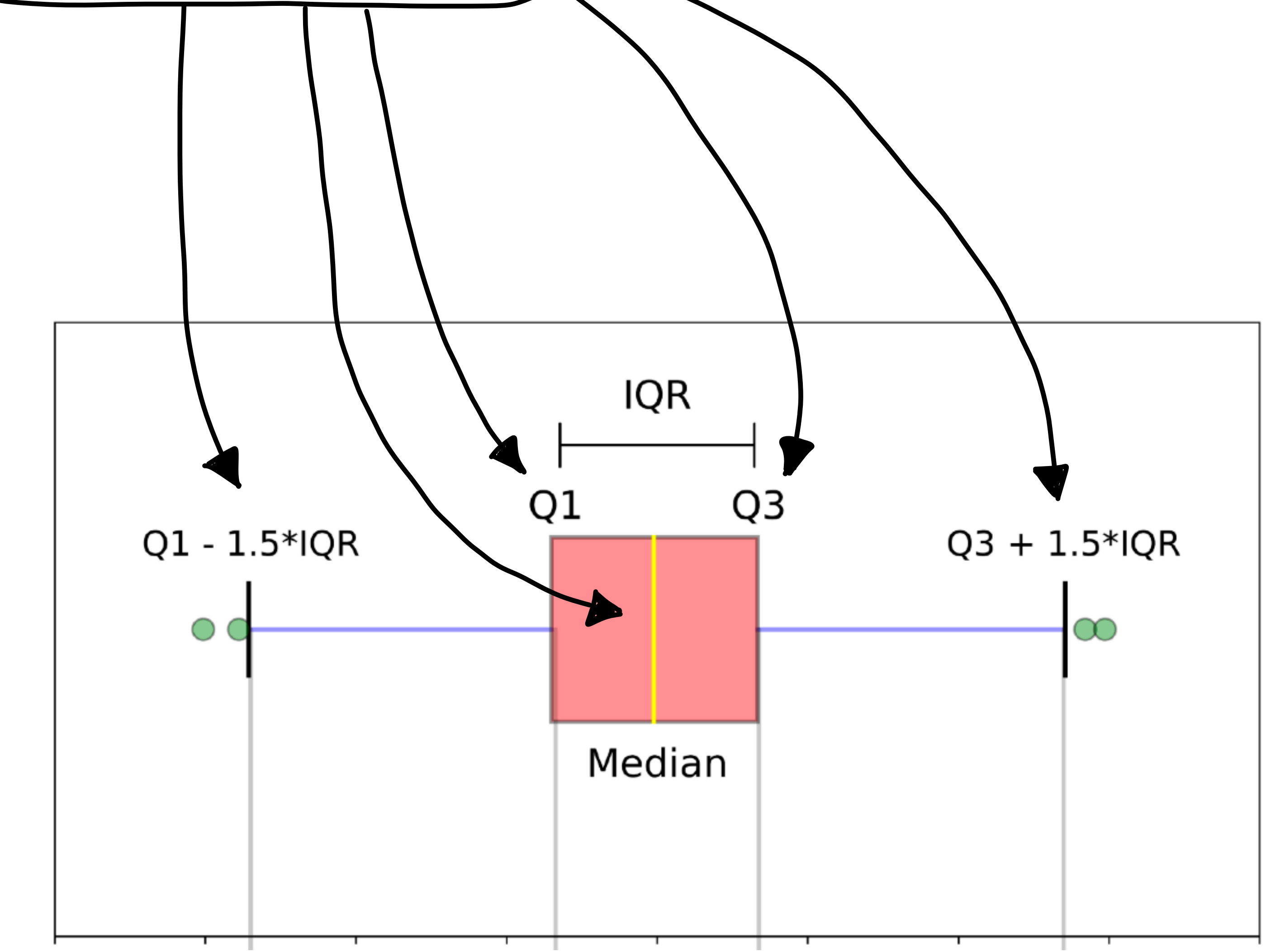
The 2nd quartile (Q2)

The 3rd quartile (Q3)



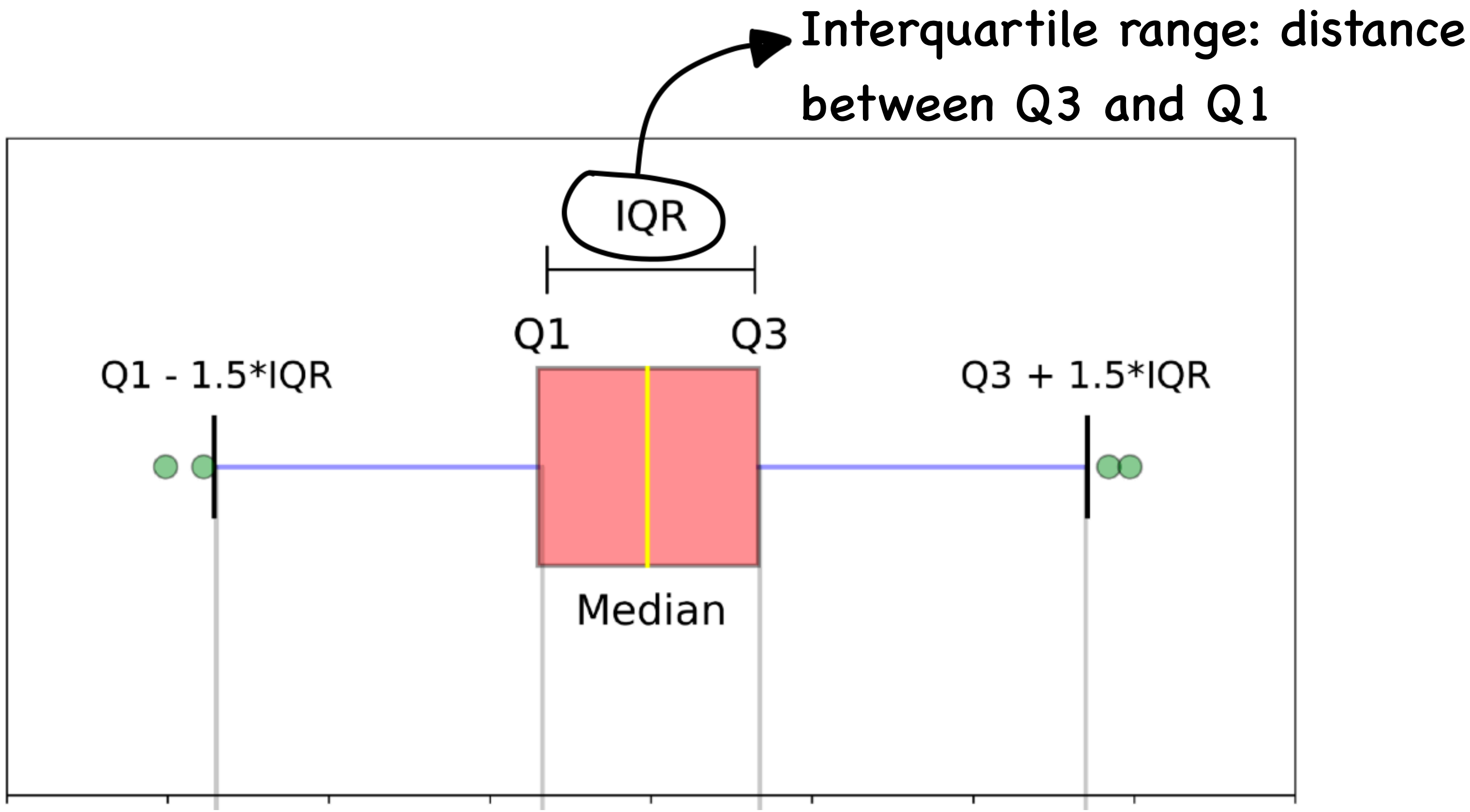
Basics of Descriptive Analytics

- Visualizing data
  - The boxplot: five-value summary



Basics of Descriptive Analytics

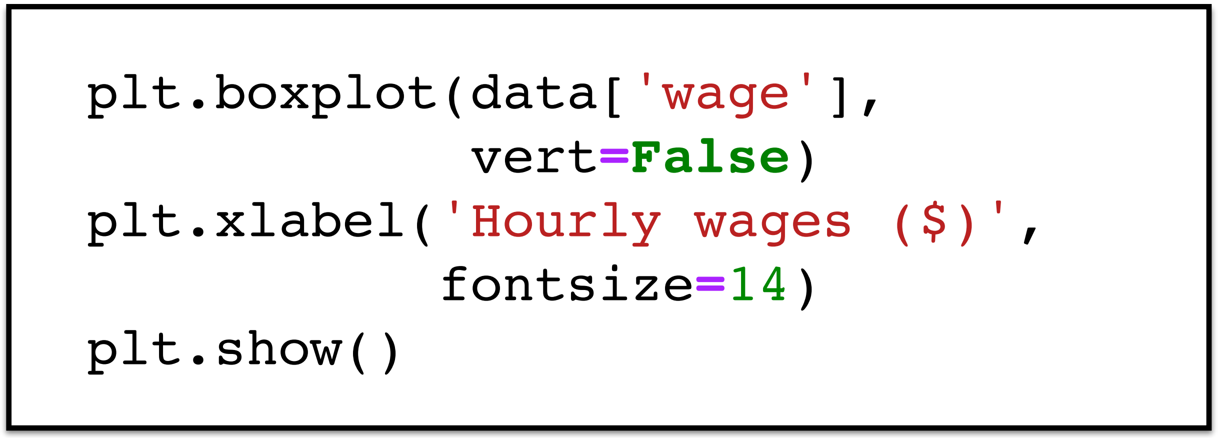
- Visualizing data
  - The boxplot: five-value summary



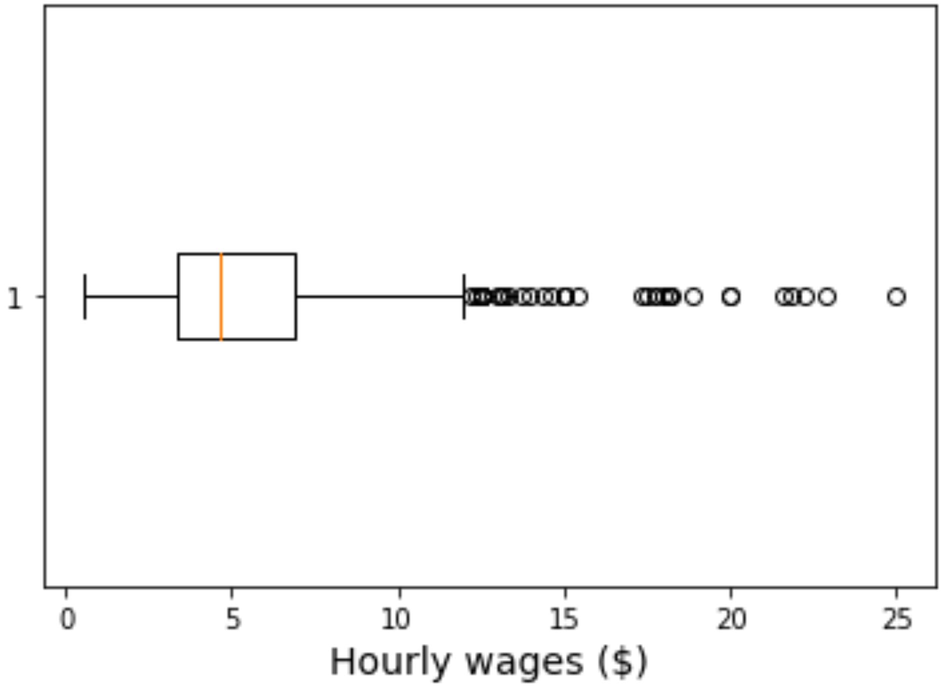
Interquartile range: distance between Q3 and Q1

Basics of Descriptive Analytics

- Visualizing data
  - The boxplot: five-value summary

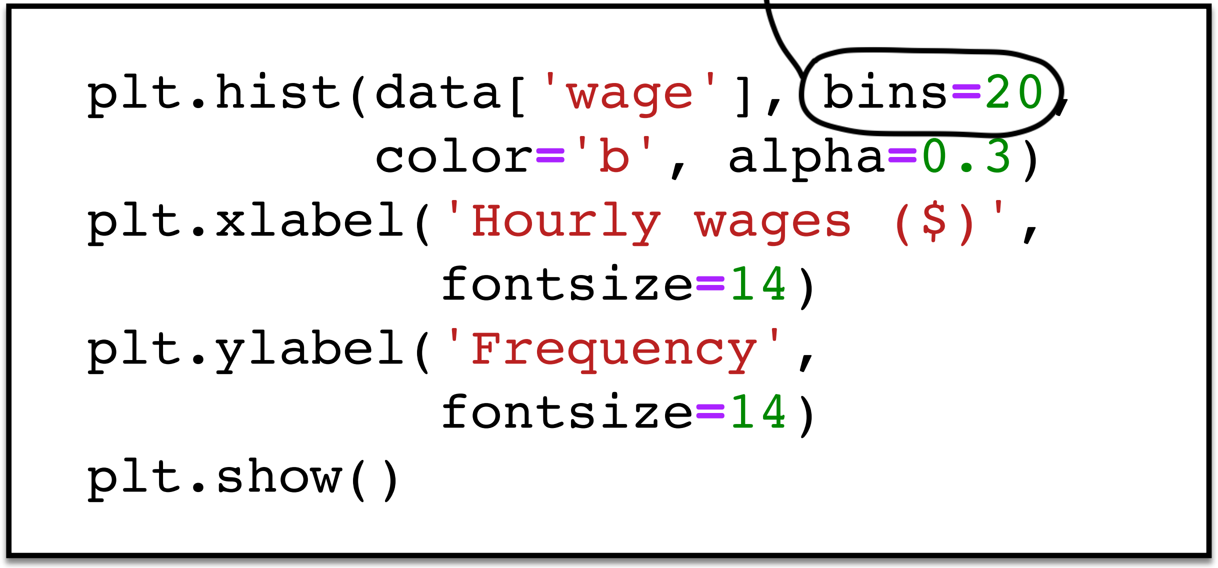


```
plt.boxplot(data['wage'],
            vert=False)
plt.xlabel('Hourly wages ($)',
           fontsize=14)
plt.show()
```



Basics of Descriptive Analytics

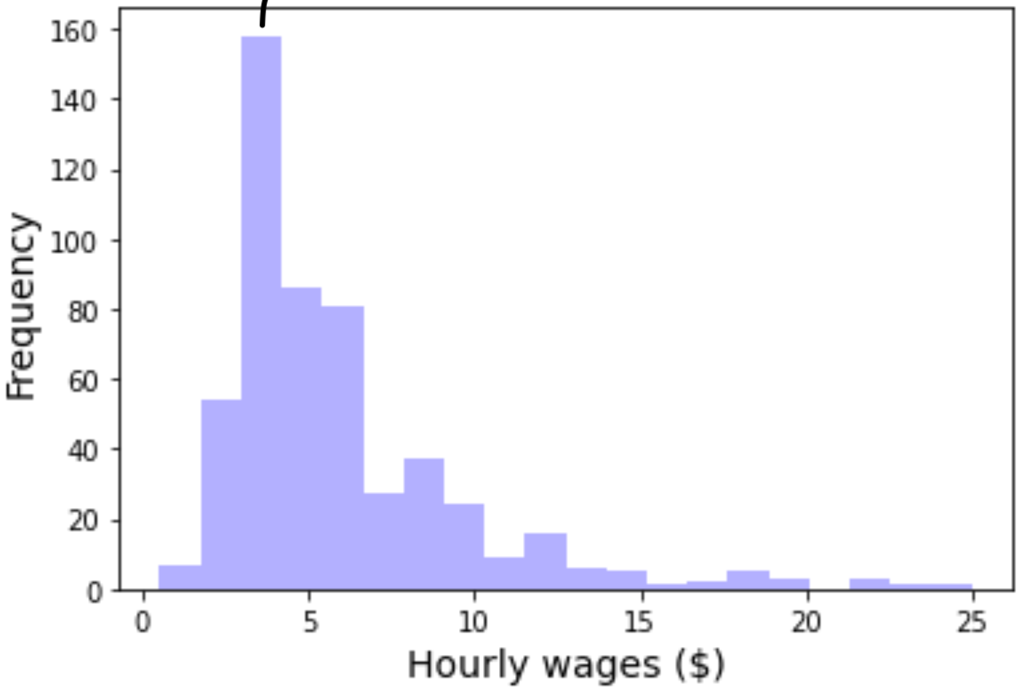
- Visualizing data
  - Histogram



Number of bins

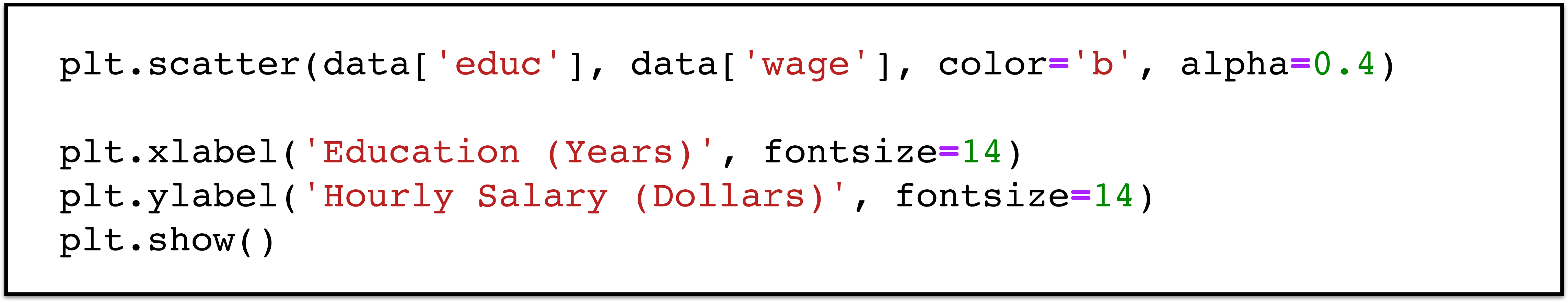
```
plt.hist(data['wage'], bins=20,
         color='b', alpha=0.3)
plt.xlabel('Hourly wages ($)',
           fontsize=14)
plt.ylabel('Frequency',
           fontsize=14)
plt.show()
```

One bin



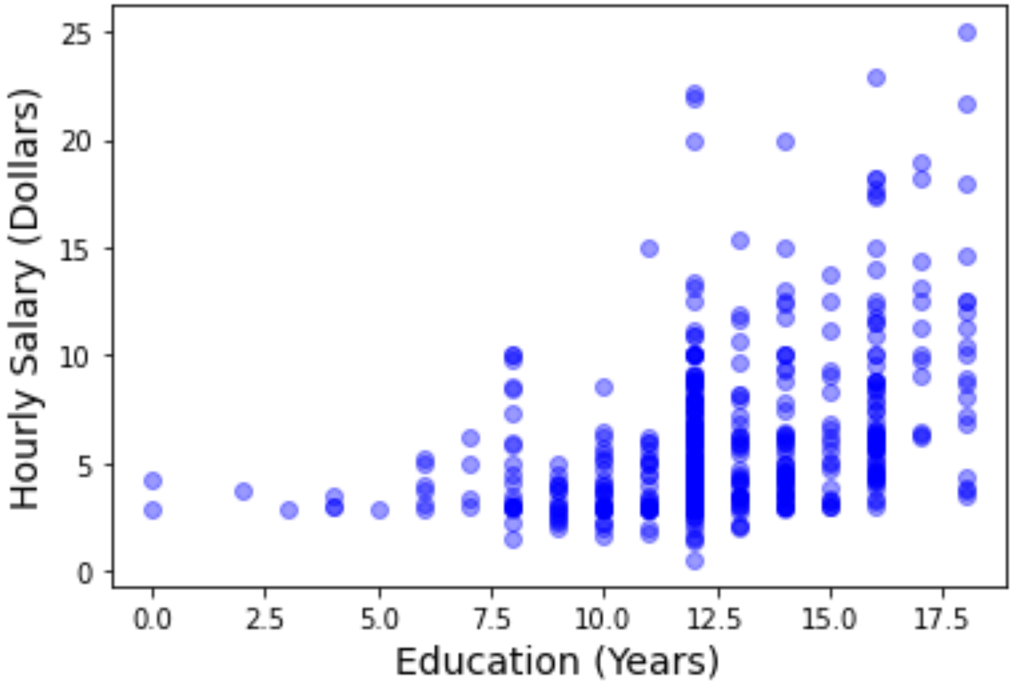
Basics of Descriptive Analytics

- Visualizing data
  - Scatterplots



```
plt.scatter(data['educ'], data['wage'], color='b', alpha=0.4)

plt.xlabel('Education (Years)', fontsize=14)
plt.ylabel('Hourly Salary (Dollars)', fontsize=14)
plt.show()
```



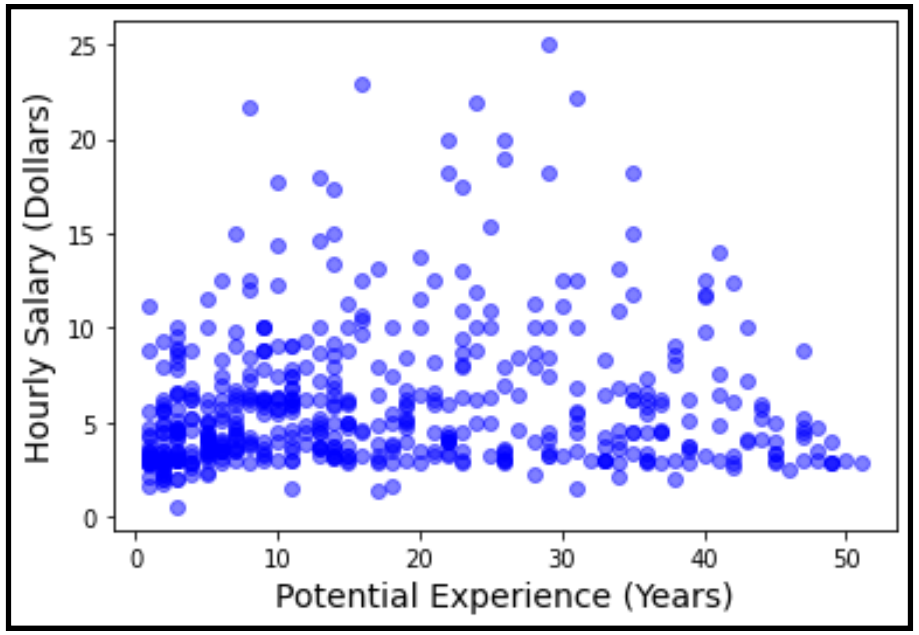
Pandas for Data Analysis

- Visualizing data

Question 2:

- Visualize how workers' potential experiences affect their hourly wages.

```
plt.scatter(data['exper'], data['wage'], alpha=0.5, color='b')
plt.xlabel(‘Potential Experience (Years)', fontsize=14)
plt.ylabel('Hourly Salary (Dollars)', fontsize=14)
plt.show()
```



Pandas for Data Analysis

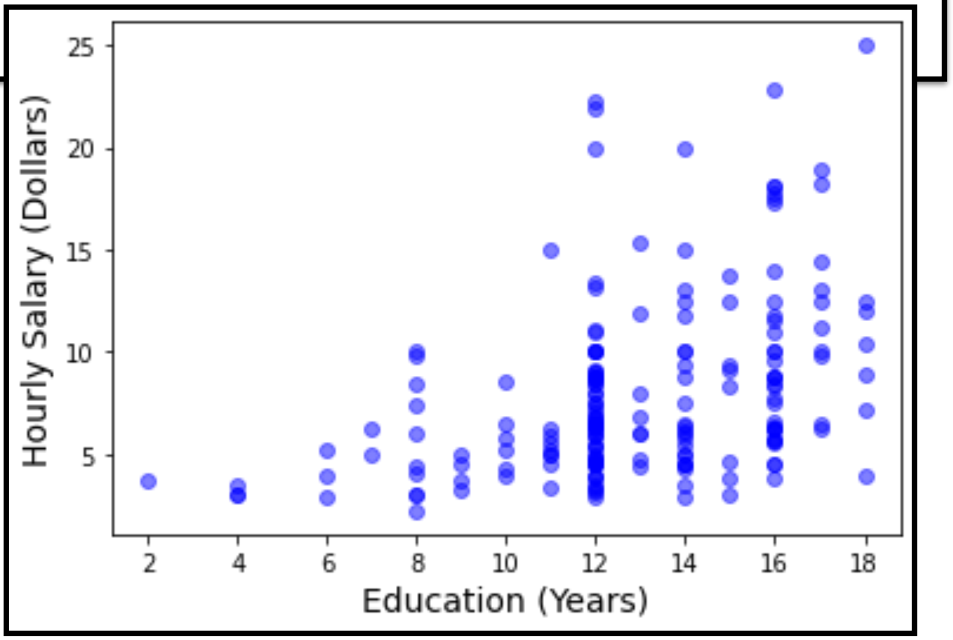
- Visualizing data

Question 2:

- Visualize how married male workers' education years affect their hourly wages.

```
subset = data.loc[data['married'] & (data['gender'] == 'M')]

plt.scatter(subset['educ'], subset['wage'], alpha=0.5, color='b')
plt.xlabel('Education (Years)', fontsize=14)
plt.ylabel('Hourly Salary (Dollars)', fontsize=14)
plt.show()
```



Pandas for Data Analysis

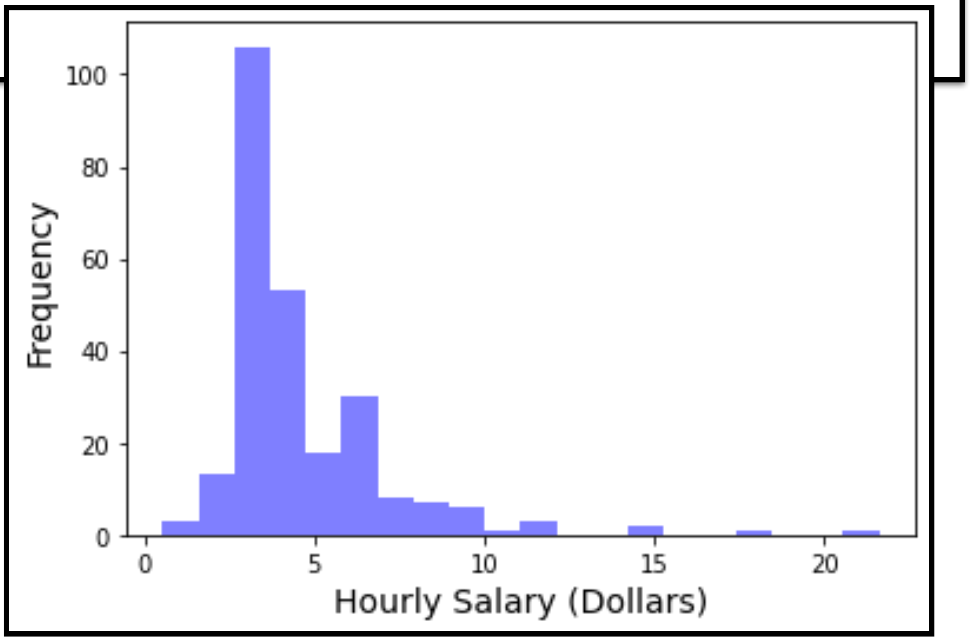
- Visualizing data

Question 2:

- Visualize the distribution of female workers' wages.

```
subset = data.loc[data['gender'] == 'F']

plt.hist(subset['wage'], bins=20, alpha=0.5, color='b')
plt.xlabel('Hourly Salary (Dollars)', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.show()
```


Frequency
Hourly Salary (Dollars)