

Programming Principles in Python (CSCI 503/490)

Data

Dr. David Koop

pandas

- Contains high-level data structures and manipulation tools designed to make data analysis fast and easy in Python
- Built on top of NumPy
- Built with the following requirements:
 - Data structures with labeled axes (aligning data)
 - Support time series data
 - Do arithmetic operations that include metadata (labels)
 - Handle missing data
 - Add merge and relational operations

Series

- A one-dimensional array (with a type) with an **index**
- Index defaults to numbers but can also be text (like a dictionary)
- Allows easier reference to specific items
- `obj = pd.Series([7, 14, -2, 1])`
- Basically two arrays: `obj.values` and `obj.index`
- Can specify the index explicitly and use strings
- `obj2 = pd.Series([4, 7, -5, 3],
 index=['d', 'b', 'a', 'c'])`
- Kind of like fixed-length, ordered dictionary + can create from a dictionary
- `obj3 = pd.Series({'Ohio': 35000, 'Texas': 71000,
 'Oregon': 16000, 'Utah': 5000})`

Data Frame

- A dictionary of Series (labels for each series)
- A spreadsheet with row keys (the index) and column headers
- Has an index shared with each series
- Allows easy reference to any cell
- ```
df = DataFrame({'state': ['Ohio', 'Ohio', 'Ohio', 'Nevada'],
 'year': [2000, 2001, 2002, 2001],
 'pop': [1.5, 1.7, 3.6, 2.4]})
```
- Index is automatically assigned just as with a series but can be passed in as well via index kwarg
- Can reassign column names by passing columns kwarg

# DataFrame Access and Manipulation

---

- `df.values` → 2D NumPy array
- Accessing a column:
  - `df["<column>"]`
  - `df.<column>`
  - Both return Series
  - Dot syntax only works when the column is a valid identifier
- Assigning to a column:
  - `df["<column>"] = <scalar>` # all cells set to same value
  - `df["<column>"] = <array>` # values set in order
  - `df["<column>"] = <series>` # values set according to match  
# between df and series indexes

# Data Frame

```
df = pd.read_csv('penguins_lter.csv')
```

|     | studyName | Sample Number | Species                             | Region | Island    | Stage              | Individual ID | Clutch Completion | Date Egg | Culmen Length (mm) |
|-----|-----------|---------------|-------------------------------------|--------|-----------|--------------------|---------------|-------------------|----------|--------------------|
| 0   | PAL0708   | 1             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A1          | Yes               | 11/11/07 | 39.1               |
| 1   | PAL0708   | 2             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A2          | Yes               | 11/11/07 | 39.5               |
| 2   | PAL0708   | 3             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A1          | Yes               | 11/16/07 | 40.3               |
| 3   | PAL0708   | 4             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A2          | Yes               | 11/16/07 | NaN                |
| 4   | PAL0708   | 5             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N3A1          | Yes               | 11/16/07 | 36.7               |
| ... | ...       | ...           | ...                                 | ...    | ...       | ...                | ...           | ...               | ...      | ...                |
| 339 | PAL0910   | 120           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N38A2         | No                | 12/1/09  | NaN                |
| 340 | PAL0910   | 121           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A1         | Yes               | 11/22/09 | 46.8               |
| 341 | PAL0910   | 122           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A2         | Yes               | 11/22/09 | 50.4               |
| 342 | PAL0910   | 123           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A1         | Yes               | 11/22/09 | 45.2               |
| 343 | PAL0910   | 124           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A2         | Yes               | 11/22/09 | 49.9               |

344 rows x 17 columns





# Data Frame

```
df = pd.read_csv('penguins_lter.csv')
```

Column Names

|     | studyName | Sample Number | Species                             | Region | Island    | Stage              | Individual ID | Clutch Completion | Date Egg | Culmen Length (mm) |
|-----|-----------|---------------|-------------------------------------|--------|-----------|--------------------|---------------|-------------------|----------|--------------------|
| 0   | PAL0708   | 1             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A1          | Yes               | 11/11/07 | 39.1               |
| 1   | PAL0708   | 2             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A2          | Yes               | 11/11/07 | 39.5               |
| 2   | PAL0708   | 3             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A1          | Yes               | 11/16/07 | 40.3               |
| 3   | PAL0708   | 4             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A2          | Yes               | 11/16/07 | NaN                |
| 4   | PAL0708   | 5             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N3A1          | Yes               | 11/16/07 | 36.7               |
| ... | ...       | ...           | ...                                 | ...    | ...       | ...                | ...           | ...               | ...      | ...                |
| 339 | PAL0910   | 120           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N38A2         | No                | 12/1/09  | NaN                |
| 340 | PAL0910   | 121           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A1         | Yes               | 11/22/09 | 46.8               |
| 341 | PAL0910   | 122           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A2         | Yes               | 11/22/09 | 50.4               |
| 342 | PAL0910   | 123           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A1         | Yes               | 11/22/09 | 45.2               |
| 343 | PAL0910   | 124           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A2         | Yes               | 11/22/09 | 49.9               |

344 rows x 17 columns



# Data Frame

```
df = pd.read_csv('penguins_lter.csv')
```

Column Names

| studyName | Sample Number | Species | Region | Island | Stage | Individual ID | Clutch Completion | Date Egg | Culmen Length (mm) |
|-----------|---------------|---------|--------|--------|-------|---------------|-------------------|----------|--------------------|
|-----------|---------------|---------|--------|--------|-------|---------------|-------------------|----------|--------------------|

Index

|     |         |     |                                     |        |           |                    |       |     |          |      |
|-----|---------|-----|-------------------------------------|--------|-----------|--------------------|-------|-----|----------|------|
| 0   | PAL0708 | 1   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A1  | Yes | 11/11/07 | 39.1 |
| 1   | PAL0708 | 2   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A2  | Yes | 11/11/07 | 39.5 |
| 2   | PAL0708 | 3   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A1  | Yes | 11/16/07 | 40.3 |
| 3   | PAL0708 | 4   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A2  | Yes | 11/16/07 | NaN  |
| 4   | PAL0708 | 5   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N3A1  | Yes | 11/16/07 | 36.7 |
| ... | ...     | ... | ...                                 | ...    | ...       | ...                | ...   | ... | ...      | ...  |
| 339 | PAL0910 | 120 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N38A2 | No  | 12/1/09  | NaN  |
| 340 | PAL0910 | 121 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A1 | Yes | 11/22/09 | 46.8 |
| 341 | PAL0910 | 122 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A2 | Yes | 11/22/09 | 50.4 |
| 342 | PAL0910 | 123 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A1 | Yes | 11/22/09 | 45.2 |
| 343 | PAL0910 | 124 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A2 | Yes | 11/22/09 | 49.9 |

344 rows x 17 columns





# Data Frame

```
df = pd.read_csv('penguins_lter.csv')
```

Column Names

| studyName | Sample Number | Species | Region | Island | Stage | Individual ID | Clutch Completion | Date Egg | Culmen Length (mm) |
|-----------|---------------|---------|--------|--------|-------|---------------|-------------------|----------|--------------------|
|-----------|---------------|---------|--------|--------|-------|---------------|-------------------|----------|--------------------|

Index

|     |         |     |                                     |        |           |                    |       |     |          |      |
|-----|---------|-----|-------------------------------------|--------|-----------|--------------------|-------|-----|----------|------|
| 0   | PAL0708 | 1   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A1  | Yes | 11/11/07 | 39.1 |
| 1   | PAL0708 | 2   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A2  | Yes | 11/11/07 | 39.5 |
| 2   | PAL0708 | 3   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A1  | Yes | 11/16/07 | 40.3 |
| 3   | PAL0708 | 4   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A2  | Yes | 11/16/07 | NaN  |
| 4   | PAL0708 | 5   | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N3A1  | Yes | 11/16/07 | 36.7 |
| ... | ...     | ... | ...                                 | ...    | ...       | ...                | ...   | ... | ...      | ...  |
| 339 | PAL0910 | 120 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N38A2 | No  | 12/1/09  | NaN  |
| 340 | PAL0910 | 121 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A1 | Yes | 11/22/09 | 46.8 |
| 341 | PAL0910 | 122 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A2 | Yes | 11/22/09 | 50.4 |
| 342 | PAL0910 | 123 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A1 | Yes | 11/22/09 | 45.2 |
| 343 | PAL0910 | 124 | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A2 | Yes | 11/22/09 | 49.9 |

344 rows x 17 columns

Column: df[ 'Island' ]



# Data Frame

```
df = pd.read_csv('penguins_lter.csv')
```

Column Names

|     | studyName | Sample Number | Species                             | Region | Island    | Stage              | Individual ID | Clutch Completion | Date Egg | Culmen Length (mm) |
|-----|-----------|---------------|-------------------------------------|--------|-----------|--------------------|---------------|-------------------|----------|--------------------|
| 0   | PAL0708   | 1             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A1          | Yes               | 11/11/07 | 39.1               |
| 1   | PAL0708   | 2             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A2          | Yes               | 11/11/07 | 39.5               |
| 2   | PAL0708   | 3             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A1          | Yes               | 11/16/07 | 40.3               |
| 3   | PAL0708   | 4             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A2          | Yes               | 11/16/07 | NaN                |
| 4   | PAL0708   | 5             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N3A1          | Yes               | 11/16/07 | 36.7               |
| ... | ...       | ...           | ...                                 | ...    | ...       | ...                | ...           | ...               | ...      | ...                |
| 339 | PAL0910   | 120           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N38A2         | No                | 12/1/09  | NaN                |
| 340 | PAL0910   | 121           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A1         | Yes               | 11/22/09 | 46.8               |
| 341 | PAL0910   | 122           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A2         | Yes               | 11/22/09 | 50.4               |
| 342 | PAL0910   | 123           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A1         | Yes               | 11/22/09 | 45.2               |
| 343 | PAL0910   | 124           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A2         | Yes               | 11/22/09 | 49.9               |

344 rows x 17 columns

Row: df.loc[2]

Index

Column: df['Island']

# Data Frame

```
df = pd.read_csv('penguins_lter.csv')
```

Column Names

|     | studyName | Sample Number | Species                             | Region | Island    | Stage              | Individual ID | Clutch Completion | Date Egg | Culmen Length (mm) |
|-----|-----------|---------------|-------------------------------------|--------|-----------|--------------------|---------------|-------------------|----------|--------------------|
| 0   | PAL0708   | 1             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A1          | Yes               | 11/11/07 | 39.1               |
| 1   | PAL0708   | 2             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A2          | Yes               | 11/11/07 | 39.5               |
| 2   | PAL0708   | 3             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A1          | Yes               | 11/16/07 | 40.3               |
| 3   | PAL0708   | 4             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A2          | Yes               | 11/16/07 | NaN                |
| 4   | PAL0708   | 5             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N3A1          | Yes               | 11/16/07 | 36.7               |
| ... | ...       | ...           | ...                                 | ...    | ...       | ...                | ...           | ...               | ...      | ...                |
| 339 | PAL0910   | 120           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N38A2         | No                | 12/1/09  | NaN                |
| 340 | PAL0910   | 121           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A1         | Yes               | 11/22/09 | 46.8               |
|     |           |               | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A2         | Yes               | 11/22/09 | 50.4               |
| 342 | PAL0910   | 123           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A1         | Yes               | 11/22/09 | 45.2               |
| 343 | PAL0910   | 124           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A2         | Yes               | 11/22/09 | 49.9               |

Row: df.loc[2]

Index

Cell: df.loc[341, 'Species']

344 rows x 17 columns

Column: df['Island']



# Data Frame

```
df = pd.read_csv('penguins_lter.csv')
```

Column Names

|     | studyName | Sample Number | Species                             | Region | Island    | Stage              | Individual ID | Clutch Completion | Date Egg | Culmen Length (mm) |
|-----|-----------|---------------|-------------------------------------|--------|-----------|--------------------|---------------|-------------------|----------|--------------------|
| 0   | PAL0708   | 1             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A1          | Yes               | 11/11/07 | 39.1               |
| 1   | PAL0708   | 2             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A2          | Yes               | 11/11/07 | 39.5               |
| 2   | PAL0708   | 3             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A1          | Yes               | 11/16/07 | 40.3               |
| 3   | PAL0708   | 4             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A2          | Yes               | 11/16/07 | NaN                |
| 4   | PAL0708   | 5             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N3A1          | Yes               | 11/16/07 |                    |
| ... | ...       | ...           | ...                                 | ...    | ...       | ...                | ...           | ...               | ...      | ...                |
| 339 | PAL0910   | 120           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N38A2         | No                | 12/1/09  | NaN                |
| 340 | PAL0910   | 121           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A1         | Yes               | 11/22/09 | 46.8               |
|     |           |               | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A2         | Yes               | 11/22/09 | 50.4               |
| 342 | PAL0910   | 123           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A1         | Yes               | 11/22/09 | 45.2               |
| 343 | PAL0910   | 124           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A2         | Yes               | 11/22/09 | 49.9               |

Row: df.loc[2]

Index

Missing Data

Cell: df.loc[341, 'Species']

Column: df['Island']

344 rows x 17 columns

# Indexing

---

- Same as with NumPy arrays but can use Series's index labels
- Slicing with labels: NumPy is **exclusive**, Pandas is **inclusive**!
  - `s = Series(np.arange(4))`  
`s[0:2]` # gives two values like numpy
  - `s = Series(np.arange(4), index=['a', 'b', 'c', 'd'])`  
`s['a':'c']` # gives three values, not two!
- Obtaining data subsets
  - `[]`: get columns by label
  - `loc`: get rows/cols by label
  - `iloc`: get rows/cols by position (integer index)
  - For single cells (scalars), also have `at` and `iat`



# Arithmetic

---

- Add, subtract, multiply, and divide are element-wise like numpy
- ...but use labels to align
- ...and missing labels lead to NaN (not a number) values

```
In [28]: obj3
Out[28]:
Ohio 35000
Oregon 16000
Texas 71000
Utah 5000
dtype: int64
```

```
In [29]: obj4
Out[29]:
California NaN
Ohio 35000
Oregon 16000
Texas 71000
dtype: float64
```

```
In [30]: obj3 + obj4
Out[30]:
California NaN
Ohio 70000
Oregon 32000
Texas 142000
Utah NaN
dtype: float64
```

- also have `.add`, `.subtract`, ... that allow `fill_value` argument
- `obj3.add(obj4, fill_value=0)`

# Filtering

---

- Same as with numpy arrays but allows use of column-based criteria
  - `data[data < 5] = 0`
  - `data[data['three'] > 5]`
- `data < 5` → boolean data frame, can be used to select specific elements
- Multiple criteria, use `&`, `|`, and `~`; remember parentheses!
  - `data[(data['three'] > 5) & (data['two'] < 10)]`

# Filtering

```
df[df['Culmen Length (mm)'] > 40]
```

|     | studyName | Sample Number | Species                             | Region | Island    | Stage              | Individual ID | Clutch Completion | Date Egg | Culmen Length (mm) |
|-----|-----------|---------------|-------------------------------------|--------|-----------|--------------------|---------------|-------------------|----------|--------------------|
| 0   | PAL0708   | 1             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A1          | Yes               | 11/11/07 | 39.1               |
| 1   | PAL0708   | 2             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A2          | Yes               | 11/11/07 | 39.5               |
| 2   | PAL0708   | 3             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A1          | Yes               | 11/16/07 | 40.3               |
| 3   | PAL0708   | 4             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A2          | Yes               | 11/16/07 | NaN                |
| 4   | PAL0708   | 5             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N3A1          | Yes               | 11/16/07 | 36.7               |
| ... | ...       | ...           | ...                                 | ...    | ...       | ...                | ...           | ...               | ...      | ...                |
| 339 | PAL0910   | 120           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N38A2         | No                | 12/1/09  | NaN                |
| 340 | PAL0910   | 121           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A1         | Yes               | 11/22/09 | 46.8               |
| 341 | PAL0910   | 122           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A2         | Yes               | 11/22/09 | 50.4               |
| 342 | PAL0910   | 123           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A1         | Yes               | 11/22/09 | 45.2               |
| 343 | PAL0910   | 124           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A2         | Yes               | 11/22/09 | 49.9               |

344 rows x 17 columns

# Filtering

```
df[df['Culmen Length (mm)'] > 40]
```

|     | studyName | Sample Number | Species                             | Region | Island    | Stage              | Individual ID | Clutch Completion | Date Egg | Culmen Length (mm) |
|-----|-----------|---------------|-------------------------------------|--------|-----------|--------------------|---------------|-------------------|----------|--------------------|
| 0   | PAL0708   | 1             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A1          | Yes               | 11/11/07 | 39.1               |
| 1   | PAL0708   | 2             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N1A2          | Yes               | 11/11/07 | 39.5               |
| 2   | PAL0708   | 3             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A1          | Yes               | 11/16/07 | 40.3               |
| 3   | PAL0708   | 4             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N2A2          | Yes               | 11/16/07 | NaN                |
| 4   | PAL0708   | 5             | Adelie Penguin (Pygoscelis adeliae) | Anvers | Torgersen | Adult, 1 Egg Stage | N3A1          | Yes               | 11/16/07 | 36.7               |
| ... | ...       | ...           | ...                                 | ...    | ...       | ...                | ...           | ...               | ...      | ...                |
| 339 | PAL0910   | 120           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N38A2         | No                | 12/1/09  | NaN                |
| 340 | PAL0910   | 121           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A1         | Yes               | 11/22/09 | 46.8               |
| 341 | PAL0910   | 122           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N39A2         | Yes               | 11/22/09 | 50.4               |
| 342 | PAL0910   | 123           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A1         | Yes               | 11/22/09 | 45.2               |
| 343 | PAL0910   | 124           | Gentoo penguin (Pygoscelis papua)   | Anvers | Biscoe    | Adult, 1 Egg Stage | N43A2         | Yes               | 11/22/09 | 49.9               |

344 rows x 17 columns

# Sorting by Value (sort\_values)

---

- `sort_values` method on series
  - `obj.sort_values()`
- Missing values (NaN) are at the end by default (`na_position` controls, can be first)
- `sort_values` on DataFrame:
  - `df.sort_values(<list-of-columns>)`
  - `df.sort_values(by=['a', 'b'])`
  - Can also use `axis=1` to sort by index labels



# Statistics

---

- `sum`: column sums (`axis=1` gives sums over rows)
- missing values are excluded unless the whole slice is `NaN`
- `idxmax`, `idxmin` are like `argmax`, `argmin` (return index)
- `describe`: shortcut for easy stats!

```
In [204]: df.describe()
Out[204]:
```

|       | one      | two       |
|-------|----------|-----------|
| count | 3.000000 | 2.000000  |
| mean  | 3.083333 | -2.900000 |
| std   | 3.493685 | 2.262742  |
| min   | 0.750000 | -4.500000 |
| 25%   | 1.075000 | -3.700000 |
| 50%   | 1.400000 | -2.900000 |
| 75%   | 4.250000 | -2.100000 |
| max   | 7.100000 | -1.300000 |

```
In [205]: obj = Series(['a', 'a', 'b', 'c'] * 4)
```

```
In [206]: obj.describe()
Out[206]:
count 16
unique 3
top a
freq 8
dtype: object
```

# Unique Values and Value Counts

---

- `unique()` returns an array with only the unique values (no index)
  - `s = Series(['c', 'a', 'd', 'a', 'a', 'b', 'b', 'c', 'c'])`  
`s.unique()` # `array(['c', 'a', 'd', 'b'])`
- Also `nunique()` to count number of unique entries
- Data Frames use `drop_duplicates`
- `value_counts` returns a Series with index frequencies:
  - `s.value_counts()` # `Series({'c': 3, 'a': 3, 'b': 2, 'd': 1})`

# Reading & Writing Data in Pandas

| Format | Data Description                     | Reader         | Writer       |
|--------|--------------------------------------|----------------|--------------|
| text   | <a href="#">CSV</a>                  | read_csv       | to_csv       |
| text   | Fixed-Width Text File                | read_fwf       |              |
| text   | <a href="#">JSON</a>                 | read_json      | to_json      |
| text   | <a href="#">HTML</a>                 | read_html      | to_html      |
| text   | Local clipboard                      | read_clipboard | to_clipboard |
|        | <a href="#">MS Excel</a>             | read_excel     | to_excel     |
| binary | <a href="#">OpenDocument</a>         | read_excel     |              |
| binary | <a href="#">HDF5 Format</a>          | read_hdf       | to_hdf       |
| binary | <a href="#">Feather Format</a>       | read_feather   | to_feather   |
| binary | <a href="#">Parquet Format</a>       | read_parquet   | to_parquet   |
| binary | <a href="#">ORC Format</a>           | read_orc       |              |
| binary | <a href="#">Msgpack</a>              | read_msgpack   | to_msgpack   |
| binary | <a href="#">Stata</a>                | read_stata     | to_stata     |
| binary | <a href="#">SAS</a>                  | read_sas       |              |
| binary | <a href="#">SPSS</a>                 | read_spss      |              |
| binary | <a href="#">Python Pickle Format</a> | read_pickle    | to_pickle    |
| SQL    | <a href="#">SQL</a>                  | read_sql       | to_sql       |
| SQL    | <a href="#">Google BigQuery</a>      | read_gbq       | to_gbq       |

[[https://pandas.pydata.org/pandas-docs/stable/user\\_guide/io.html](https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html)]

# Reading CSV data with pandas

---

- Convenient method to read csv files
- Lots of different options to help get data into the desired format
- Basic: `df = pd.read_csv(fname)`
- Parameters:
  - `path`: where to read the data from
  - `sep` (or `delimiter`): the delimiter (`,`, `' '`, `'\t'`, `'\s+'`)
  - `header`: if `None`, no header
  - `index_col`: which column to use as the row index
  - `names`: list of header names (e.g. if the file has no header)
  - `skiprows`: number of list of lines to skip

# Writing CSV data with pandas

---

- Basic: `df.to_csv(<fname>)`
- Change delimiter with `sep` kwarg:
  - `df.to_csv('example.dsv', sep='|')`
- Change missing value representation
  - `df.to_csv('example.dsv', na_rep='NULL')`
- Don't write row or column labels:
  - `df.to_csv('example.csv', index=False, header=False)`
- Series may also be written to csv



# Handling Missing Data

---

| Argument             | Description                                                                                                                                 |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <code>dropna</code>  | Filter axis labels based on whether values for each label have missing data, with varying thresholds for how much missing data to tolerate. |
| <code>fillna</code>  | Fill in missing data with some value or using an interpolation method such as <code>'ffill'</code> or <code>'bfill'</code> .                |
| <code>isnull</code>  | Return like-type object containing boolean values indicating which values are missing / NA.                                                 |
| <code>notnull</code> | Negation of <code>isnull</code> .                                                                                                           |

---

[W. McKinney, Python for Data Analysis]

# Derived Data

---

- Create new columns from existing columns
  - `r["PctFail"] = r['Fail'] / r['Total']`
- Note that operations are computed in a vectorized manner
- Similarities to functional paradigm (map/filter):
  - specify the operation once
  - no loops
  - interpreted as an operation on the entire column

# inplace

---

- Generally, when we modify a data frame, we reassign:
  - `rdf = df.reset_index()`
  - This is usually very **efficient**
  - Allows for method chaining
- There are versions where you can do this "inplace":
  - `df.reset_index(inplace=True)`
  - This means **no reassignment**, but it isn't usually any faster nor better
  - Sometimes still creates a copy
  - Will likely be deprecated

# Documentation

---

- pandas [documentation](#) is pretty good
- Lots of recipes on stackoverflow for particular data manipulations/queries

# Teaching Evaluations

---

- This Thursday (Nov. 18) in class



# Assignment 7

---

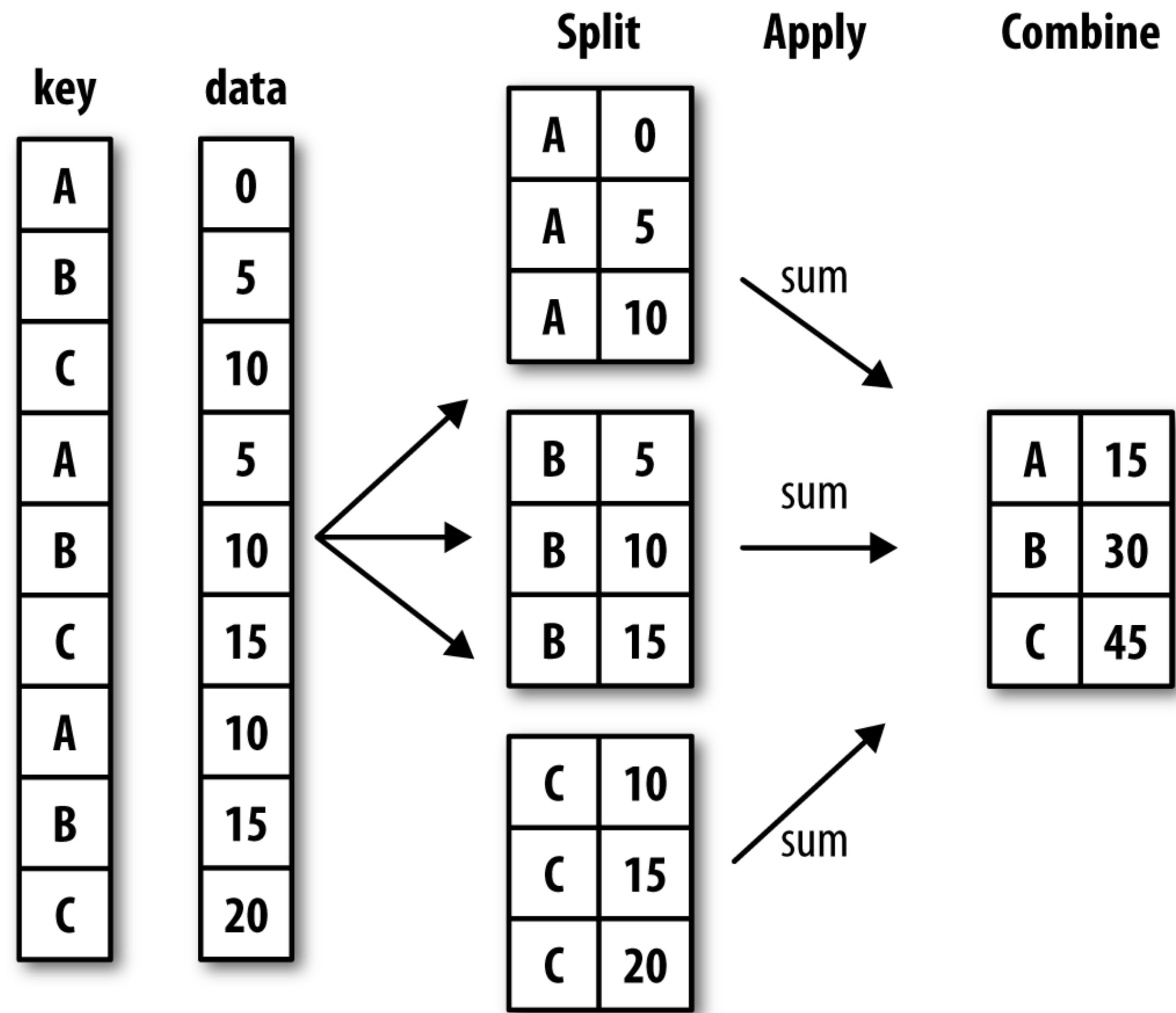
- Downloading and unarchiving files
  - CSCI 503: Use aiohttp, CSCI 490: use requests
- File system manipulation
- Threading
- Basic Data Manipulation
- Due Monday

# Aggregation

---

- Descriptive statistics
  - `df['Culmen Length (mm)'].mean()`
  - `.median()`
  - `.describe()`
  - `.count()`
  - `.min()`, `.max()`
- Also general methods
  - `.sum()`
  - `.product()`

# Split-Apply-Combine



[W. McKinney, Python for Data Analysis]

# Split-Apply-Combine

---

- Similar to Map (split+apply) Reduce (combine) paradigm
- The Pattern:
  1. **Split** the data by some grouping variable
  2. **Apply** some function to each group independently
  3. **Combine** the data into some output dataset
- The apply step is usually one of:
  - Aggregate
  - Transform
  - Filter

[T. Brandt]

# In Pandas

---

- `groupby` method creates a `GroupBy` object
- `groupby` doesn't actually compute anything until there is an `apply/aggregate` step or we wish to examine the groups
- Choose keys (columns) to group by
- `size()` is the count of each group
- Other aggregates also work

# Examples

---

- `df.groupby('Island')`
- `df.groupby('Island').size()`
- `df.groupby('Island')['Culmen Length (mm)'].mean()`



# Split-Apply-Combine

---

- `df.groupby('Island')[['Culmen Length (mm)', 'Culmen Depth (mm)']].mean()`
- `df.groupby('Island').agg({'Culmen Length (mm)': 'mean', 'Culmen Depth (mm)': 'mean'})`
- `df.groupby('Island').agg(cul_length=('Culmen Length (mm)', 'mean'), cul_depth=('Culmen Depth (mm)', 'mean'))`

|           | cul_length | cul_depth |
|-----------|------------|-----------|
| Island    |            |           |
| Biscoe    | 45.257485  | 15.874850 |
| Dream     | 44.167742  | 18.344355 |
| Torgersen | 38.950980  | 18.429412 |

# Different Data Layouts

|              | treatmenta | treatmentb |
|--------------|------------|------------|
| John Smith   | —          | 2          |
| Jane Doe     | 16         | 11         |
| Mary Johnson | 3          | 1          |

Initial Data

| name         | trt | result |
|--------------|-----|--------|
| John Smith   | a   | —      |
| Jane Doe     | a   | 16     |
| Mary Johnson | a   | 3      |
| John Smith   | b   | 2      |
| Jane Doe     | b   | 11     |
| Mary Johnson | b   | 1      |

Tidy Data

|            | John Smith | Jane Doe | Mary Johnson |
|------------|------------|----------|--------------|
| treatmenta | —          | 16       | 3            |
| treatmentb | 2          | 11       | 1            |

Transpose

[H. Wickham, 2014]

# Problem: Variables stored in both rows & columns

Mexico Weather, Global Historical Climatology Network

| id      | year | month | element | d1 | d2   | d3   | d4 | d5   | d6 | d7 | d8 |
|---------|------|-------|---------|----|------|------|----|------|----|----|----|
| MX17004 | 2010 | 1     | tmax    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 1     | tmin    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 2     | tmax    | —  | 27.3 | 24.1 | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 2     | tmin    | —  | 14.4 | 14.4 | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 3     | tmax    | —  | —    | —    | —  | 32.1 | —  | —  | —  |
| MX17004 | 2010 | 3     | tmin    | —  | —    | —    | —  | 14.2 | —  | —  | —  |
| MX17004 | 2010 | 4     | tmax    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 4     | tmin    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 5     | tmax    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 5     | tmin    | —  | —    | —    | —  | —    | —  | —  | —  |

[H. Wickham, 2014]

# Problem: Variables stored in both rows & columns

Mexico Weather, Global Historical Climatology Network

| id      | year | month | element | d1 | d2   | d3   | d4 | d5   | d6 | d7 | d8 |
|---------|------|-------|---------|----|------|------|----|------|----|----|----|
| MX17004 | 2010 | 1     | tmax    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 1     | tmin    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 2     | tmax    | —  | 27.3 | 24.1 | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 2     | tmin    | —  | 14.4 | 14.4 | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 3     | tmax    | —  | —    | —    | —  | 32.1 | —  | —  | —  |
| MX17004 | 2010 | 3     | tmin    | —  | —    | —    | —  | 14.2 | —  | —  | —  |
| MX17004 | 2010 | 4     | tmax    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 4     | tmin    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 5     | tmax    | —  | —    | —    | —  | —    | —  | —  | —  |
| MX17004 | 2010 | 5     | tmin    | —  | —    | —    | —  | —    | —  | —  | —  |

Variable in columns: day; Variable in rows: tmax/tmin

[H. Wickham, 2014]

# Solution: Melting + Pivot

| id      | date       | element | value |
|---------|------------|---------|-------|
| MX17004 | 2010-01-30 | tmax    | 27.8  |
| MX17004 | 2010-01-30 | tmin    | 14.5  |
| MX17004 | 2010-02-02 | tmax    | 27.3  |
| MX17004 | 2010-02-02 | tmin    | 14.4  |
| MX17004 | 2010-02-03 | tmax    | 24.1  |
| MX17004 | 2010-02-03 | tmin    | 14.4  |
| MX17004 | 2010-02-11 | tmax    | 29.7  |
| MX17004 | 2010-02-11 | tmin    | 13.4  |
| MX17004 | 2010-02-23 | tmax    | 29.9  |
| MX17004 | 2010-02-23 | tmin    | 10.7  |

(a) Molten data

| id      | date       | tmax | tmin |
|---------|------------|------|------|
| MX17004 | 2010-01-30 | 27.8 | 14.5 |
| MX17004 | 2010-02-02 | 27.3 | 14.4 |
| MX17004 | 2010-02-03 | 24.1 | 14.4 |
| MX17004 | 2010-02-11 | 29.7 | 13.4 |
| MX17004 | 2010-02-23 | 29.9 | 10.7 |
| MX17004 | 2010-03-05 | 32.1 | 14.2 |
| MX17004 | 2010-03-10 | 34.5 | 16.8 |
| MX17004 | 2010-03-16 | 31.1 | 17.6 |
| MX17004 | 2010-04-27 | 36.3 | 16.7 |
| MX17004 | 2010-05-27 | 33.2 | 18.2 |

(b) Tidy data

[H. Wickham, 2014]



# Melt

- Want to keep each observation separate (tidy), aka pivot\_longer

|   | location | Temperature | Jan-2010 | Feb-2010 | Mar-2010 |
|---|----------|-------------|----------|----------|----------|
| 0 | CityA    | Predict     | 30       | 45       | 24       |
| 1 | CityB    | Actual      | 32       | 43       | 22       |

```
df.melt(id_vars=["location", "Temperature"],
 var_name="Date", value_name="Value")
```

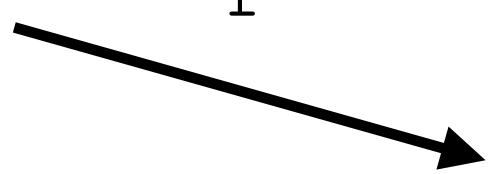
|   | location | Temperature | Date     | Value |
|---|----------|-------------|----------|-------|
| 0 | CityA    | Predict     | Jan-2010 | 30    |
| 1 | CityB    | Actual      | Jan-2010 | 32    |
| 2 | CityA    | Predict     | Feb-2010 | 45    |
| 3 | CityB    | Actual      | Feb-2010 | 43    |
| 4 | CityA    | Predict     | Mar-2010 | 24    |
| 5 | CityB    | Actual      | Mar-2010 | 22    |

[AB Abhi]



# Pivot

- Sometimes, we have data that is given in "long" format and we would like "wide" format (aka pivot\_wider)
- Long format: column names are data values...
- Wide format: more like spreadsheet format
- Example:

|   | date       | item    | value    |                                                                                      | <code>.pivot('date', 'item', 'value')</code> |      |          |       |
|---|------------|---------|----------|--------------------------------------------------------------------------------------|----------------------------------------------|------|----------|-------|
| 0 | 1959-03-31 | realgdp | 2710.349 |  | item                                         | infl | realgdp  | unemp |
| 1 | 1959-03-31 | infl    | 0.000    |                                                                                      | date                                         |      |          |       |
| 2 | 1959-03-31 | unemp   | 5.800    |                                                                                      | 1959-03-31                                   | 0.00 | 2710.349 | 5.8   |
| 3 | 1959-06-30 | realgdp | 2778.801 |                                                                                      | 1959-06-30                                   | 2.34 | 2778.801 | 5.1   |
| 4 | 1959-06-30 | infl    | 2.340    |                                                                                      | 1959-09-30                                   | 2.74 | 2775.488 | 5.3   |
| 5 | 1959-06-30 | unemp   | 5.100    |                                                                                      | 1959-12-31                                   | 0.27 | 2785.204 | 5.6   |
| 6 | 1959-09-30 | realgdp | 2775.488 |                                                                                      | 1960-03-31                                   | 2.31 | 2847.699 | 5.2   |
| 7 | 1959-09-30 | infl    | 2.740    |                                                                                      |                                              |      |          |       |
| 8 | 1959-09-30 | unemp   | 5.300    |                                                                                      |                                              |      |          |       |
| 9 | 1959-12-31 | realgdp | 2785.204 |                                                                                      |                                              |      |          |       |

[W. McKinney, Python for Data Analysis]

# Reshaping Data

---

- Reshape/pivoting are fundamental operations
- Can have a nested index in pandas
- Example: Congressional Districts (Ohio's 1st, 2nd, 3rd, Colorado's 1st, 2nd, 3rd) and associated representative rankings
- Could write this in different ways:

| number   | one | two | three |
|----------|-----|-----|-------|
| state    |     |     |       |
| Ohio     | 0   | 1   | 2     |
| Colorado | 3   | 4   | 5     |

| state  | Ohio | Colorado |
|--------|------|----------|
| number |      |          |
| one    | 0    | 3        |
| two    | 1    | 4        |
| three  | 2    | 5        |

| state    | number |   |
|----------|--------|---|
| Ohio     | one    | 0 |
|          | two    | 1 |
|          | three  | 2 |
| Colorado | one    | 3 |
|          | two    | 4 |
|          | three  | 5 |

# Reshaping Data

- Reshape/pivoting are fundamental operations
- Can have a nested index in pandas
- Example: Congressional Districts (Ohio's 1st, 2nd, 3rd, Colorado's 1st, 2nd, 3rd) and associated representative rankings
- Could write this in different ways:

| number   | one | two | three |
|----------|-----|-----|-------|
| state    |     |     |       |
| Ohio     | 0   | 1   | 2     |
| Colorado | 3   | 4   | 5     |

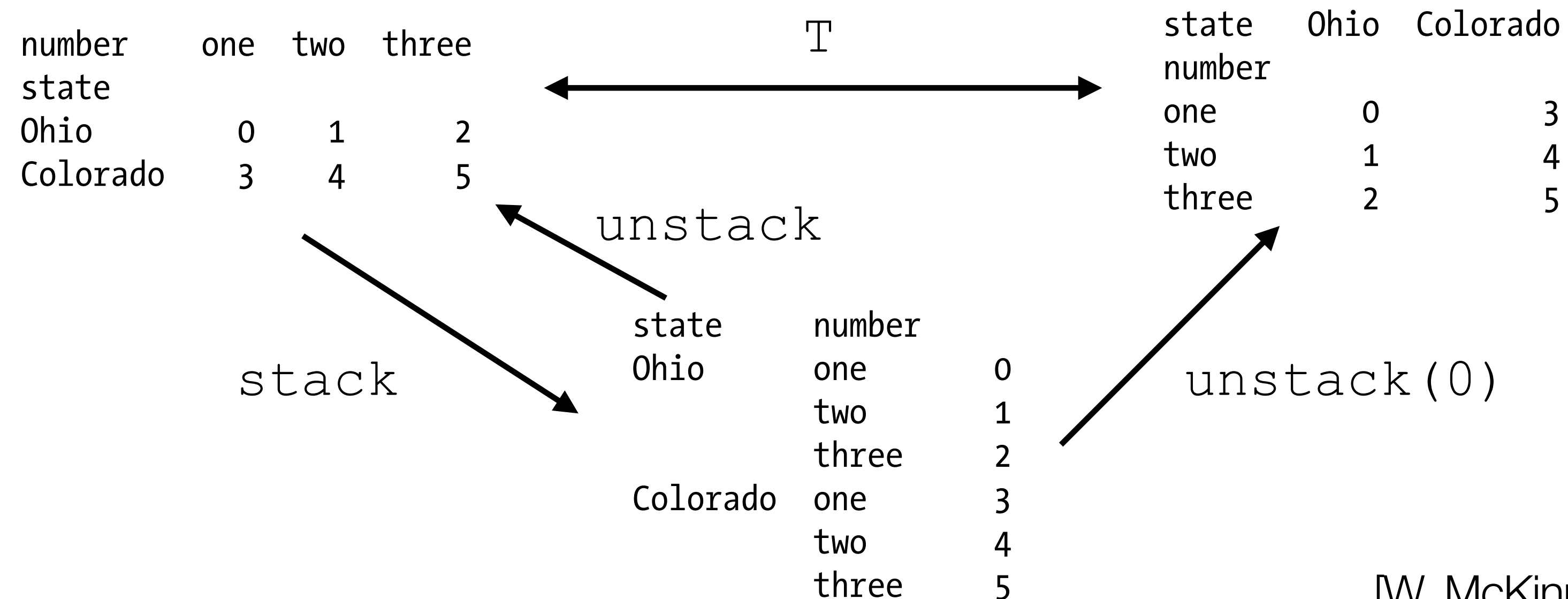
| state  | Ohio | Colorado |
|--------|------|----------|
| number |      |          |
| one    | 0    | 3        |
| two    | 1    | 4        |
| three  | 2    | 5        |

MultilIndex

| state    | number |   |
|----------|--------|---|
| Ohio     | one    | 0 |
|          | two    | 1 |
|          | three  | 2 |
| Colorado | one    | 3 |
|          | two    | 4 |
|          | three  | 5 |

# Stack and Unstack

- `stack`: pivots from the columns into rows (may produce a Series!)
- `unstack`: pivots from rows into columns
- unstacking may add missing data
- stacking filters out missing data (unless `dropna=False`)
- can unstack at a different level by passing it (e.g. 0), defaults to innermost level



[W. McKinney, Python for Data Analysis]

# String Methods

---

- Can do many of the same methods used for single strings on entire columns
- Requires `.str` prefix before calling the method
  - `violations.value.str.strip().str.split(' - Comments:')`
- Also helps when extracting from a list
  - `comments.str[1]`

# String Methods

| Argument                                                             | Description                                                                                                                                                   |
|----------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>count</code>                                                   | Return the number of non-overlapping occurrences of substring in the string.                                                                                  |
| <code>endswith</code>                                                | Returns <code>True</code> if string ends with suffix.                                                                                                         |
| <code>startswith</code>                                              | Returns <code>True</code> if string starts with prefix.                                                                                                       |
| <code>join</code>                                                    | Use string as delimiter for concatenating a sequence of other strings.                                                                                        |
| <code>index</code>                                                   | Return position of first character in substring if found in the string; raises <code>ValueError</code> if not found.                                          |
| <code>find</code>                                                    | Return position of first character of <i>first</i> occurrence of substring in the string; like <code>index</code> , but returns <code>-1</code> if not found. |
| <code>rfind</code>                                                   | Return position of first character of <i>last</i> occurrence of substring in the string; returns <code>-1</code> if not found.                                |
| <code>replace</code>                                                 | Replace occurrences of string with another string.                                                                                                            |
| <code>strip</code> ,<br><code>rstrip</code> ,<br><code>lstrip</code> | Trim whitespace, including newlines; equivalent to <code>x.strip()</code> (and <code>rstrip</code> , <code>lstrip</code> , respectively) for each element.    |
| <code>split</code>                                                   | Break string into list of substrings using passed delimiter.                                                                                                  |
| <code>lower</code>                                                   | Convert alphabet characters to lowercase.                                                                                                                     |
| <code>upper</code>                                                   | Convert alphabet characters to uppercase.                                                                                                                     |
| <code>casefold</code>                                                | Convert characters to lowercase, and convert any region-specific variable character combinations to a common comparable form.                                 |
| <code>ljust</code> ,<br><code>rjust</code>                           | Left justify or right justify, respectively; pad opposite side of string with spaces (or some other fill character) to return a string with a minimum width.  |

[W. McKinney, Python for Data Analysis]



# Support for Datetime

---

- Python has datetime library to support dates and times
- pandas has a Timestamp data type that functions somewhat similarly
- Pandas can convert timestamps
  - `pd.to_datetime`: versatile, can often guess format
- Like string methods, also a `.dt` accessor for datetime methods/properties
- With a timestamp, filtering based on datetimes becomes easier
  - `df[df['Inspection Date'] > '2021']`

# Method chaining in pandas

---

- Tom Augspurger's [post](#)
- [Towards Data Science post](#) by Adiamann Keerthi
- Functions written for chaining, and pipe allows custom functions
- ```
def read(fp):  
    df = (pd.read_csv(fp)  
          .rename(columns=str.lower)  
          .drop('unnamed: 36', axis=1)  
          .pipe(extract_city_name)  
          .pipe(time_to_datetime, ['dep_time', 'arr_time',  
                                   'crs_arr_time', 'crs_dep_time'])  
          .assign(fl_date=lambda x: pd.to_datetime(x['fl_date']),  
                  dest=lambda x: pd.Categorical(x['dest']),  
                  origin=lambda x: pd.Categorical(x['origin']),  
                  tail_num=lambda x: pd.Categorical(x['tail_num']),  
                  unique_carrier=lambda x: pd.Categorical(x['unique_carrier']),  
                  cancellation_code=lambda x: pd.Categorical(x['cancellation_code'])))  
    return df
```

Example: Inspect Intermediate Results

- ```
def csnap(df, fn=lambda x: x.shape, msg=None):
 """ Custom Help function to print things in method chaining.
 Returns back the df to further use in chaining.
 """
 if msg:
 print(msg)
 display(fn(df))
 return df
```
- ```
wine.pipe(csnap) # display data frame  
    .rename(columns={"color_intensity": "ci"})  
    .assign(color_filter=lambda x: np.where(x.hue > 1, 1, 0))  
    .pipe(csnap) # display data frame  
...
```