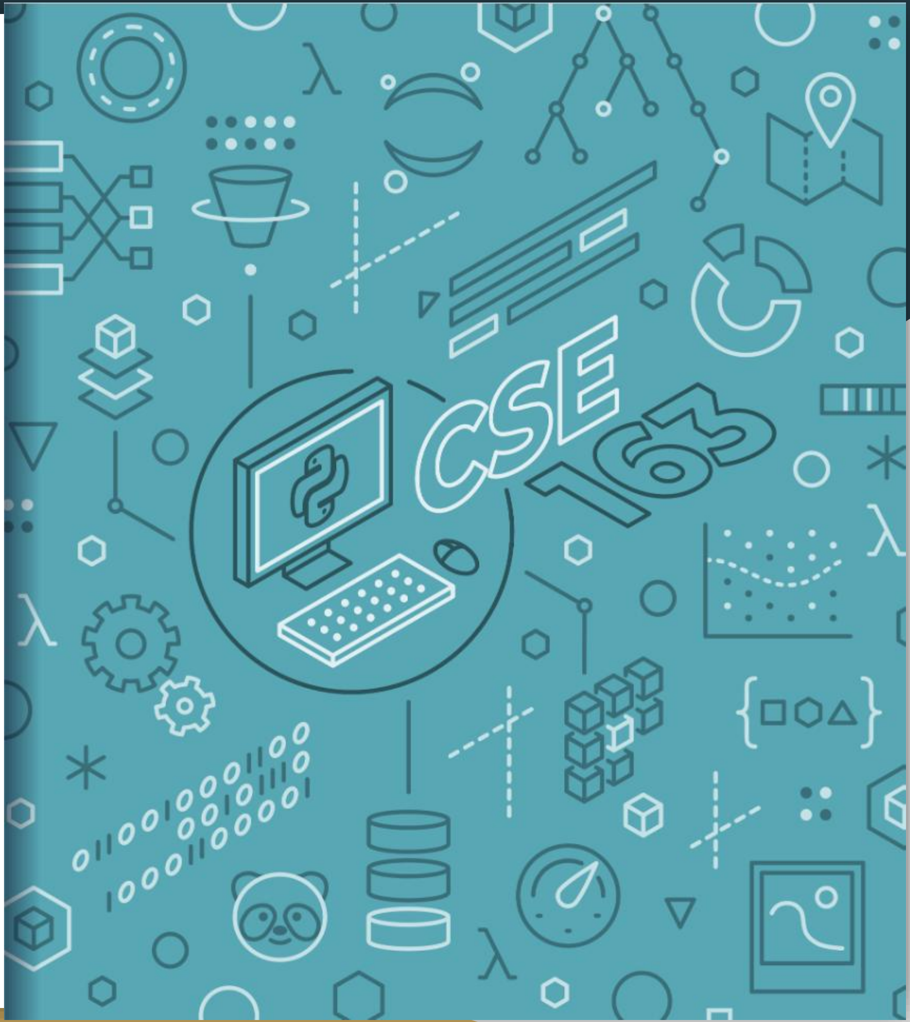
CSE 163
Pandas: DataFrames

Adrian Salguero
Spring 2026

💭 Icebreaker (discuss with neighbors): ‘
Favorite rainy day activity?
Name as many as you’d like! Add to our Slido!

slido.com
#cse163



Announcements

- **Take Home Assessment 2: Networks** due Thursday, April 23rd at 11:59pm!
- **Lesson 9 Canvas Quiz** due tonight at 11:59pm!
- **Checkpoint 2** due tonight at 11:59pm!
- **Resub Cycle 1** is available and due tomorrow at 11:59pm!
  - Make sure you resubmit your code AND fill out the resubmission form!
- **Read your feedback for THA 1** is now available on Gradescope!
  - If you have any questions regarding your grade or feedback (or where to find your feedback) → make a private Ed post, go to TA or instructor office hours

Checking In

- ***Concerned about falling behind?***
  - Reaching out might be scary, uncomfortable, or awkward but we can't help you if we don't know you need help!
  - Office Hours, Ed discussion board
  - In-class and section announcements to track and remind you of due dates!
  - Refer to the website home page regularly!
- ***Lost in translation?***
  - Write out pseudocode before writing your program
  - Write out your code in a programming language you are more comfortable with (e.g. Java) and then work on translating it in Python
  - Refer to things we cover in class/section
    - "What are situations I need to use this?"
    - "Is there a similar problem we did in class or section?"

Lesson Recap

- ***Imports***
    - A way for us to get functionality from modules we did not write ourselves!

- ***Pandas***
    - New data types - DataFrames and Series
    - More tabular-looking than lists of dictionaries
    - Lots of great functional tools for data cleaning, analysis, and visualization!

Importing

- ***Importing*** allows us to use the contents defined in another Python file/module/package
    - Generally there are **three** ways we can import!

```
# Method 1: Import module
import module
module.function()

# Method 2: Import and rename module
import module as m
m.function()

# Method 3: Import specific function from module
from module import function
function()
```

DataFrame

- One of the basic data types from pandas is a ***DataFrame!***

Columns

Index (row)

	id	year	month	day	latitude	longitude	name	magnitude
0	nc72666881	2016	7	27	37.672333	-121.619000	California	1.43
1	us20006i0y	2016	7	27	21.514600	94.572100	Burma	4.90
2	nc72666891	2016	7	27	37.576500	-118.859167	California	0.06

Series

- A ***Series*** is like a 1-dimensional DataFrame (no columns!)
  - Has an index
  - You can think of it like a fancy dictionary/list hybrid

```
df['name']
```

0	California
1	Burma
2	California

```
df['name'][1]  # 'Burma'
```

Series Operations

- You can do ***Series-wide operations*** without loops!

```
df['name'] += " :)"
```

0	California :)
1	Burma :)
2	California :)

```
df['magnitude'] *= 2
```

0	2.86
1	9.80
2	0.12

Series Operations

- What does this operation turn into?

```
df['name'] == “Burma”
```

0	California
1	Burma
2	California

Series Operations

- What does this operation turn into?

```
df['name'] == “Burma”
```

0	California
1	Burma
2	California

False
True
False

Series Operations

- What does this operation turn into?

```
df['name'] == "Burma"
```

	id	year	month	day	latitude	longitude	name	magnitude
0	nc72666881	2016	7	27	37.672333	-121.619000	False	1.43
1	us20006i0y	2016	7	27	21.514600	94.572100	True	4.90
2	nc72666891	2016	7	27	37.576500	-118.859167	False	0.06

Filtering

- We can use a bool Series to select specific rows from the dataset

```
mask = df['magnitude'] > 5
df[mask]
# Same as: data[data['magnitude'] > 5]
```

	id	year	month	day	latitude	longitude	name	magnitude
30	us20006i18	2016	7	27	-24.286000	-67.864700	Chile	5.60
114	us20006i35	2016	7	27	36.492200	140.756800	Japan	5.30
421	us1000683b	2016	7	28	-16.824200	-172.515800	Tonga	5.10

```
df[(df['magnitude'] > 5) & ~(df['day'] == 27)]
```

Can use multiple filters: **&** (and), **|** (or), **~** (not)

Location: Accessing Data in Pandas

- Series

```
series[<indexer>]
```

- DataFrame

```
df[<indexer>]
df.loc[<row indexer>, <column indexer>]
```

- Options for indexers:
  - A single value
  - A list of values or slice
  - A mask
  - : (colon) to select all values

The end of a slice is ***inclusive*** in Pandas, unlike in standard Python!