COMS BC1016
Introduction to Computational Thinking and Data Science

Lecture 5: Tables and Charts

Sept 3, 2025

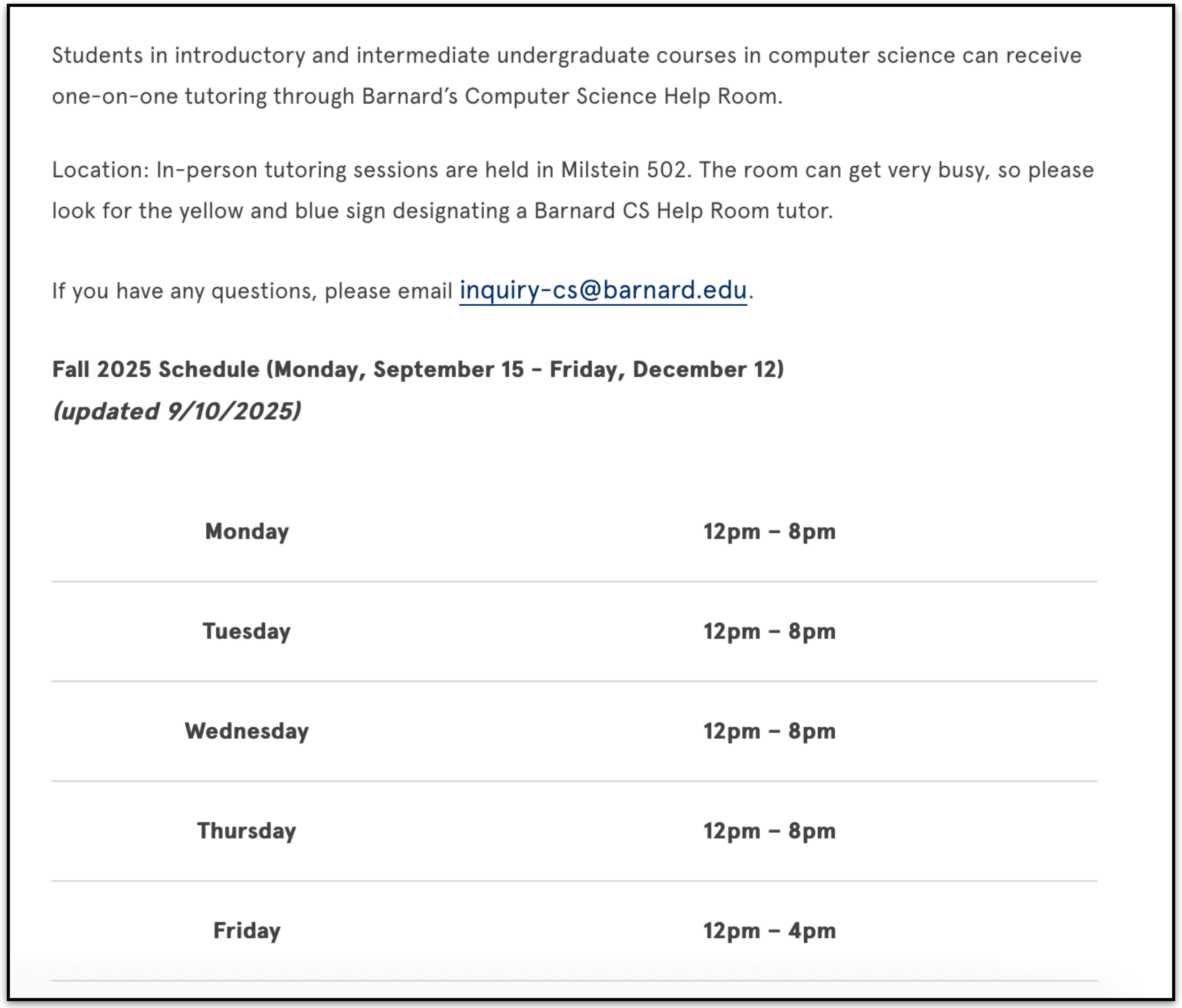
Sept 17, 2025

HW1 Updates

- HW 1 due next week Wednesday
  - Don’t mind the file name, it’s an artifact of the auto grader
  - Most of the homework numberings are going to be slightly off
- Some people were getting a weird error so we reuploaded HW1 with a small fix
  - The .ipynb file is unchanged but the other files were modified to prevent this specific error

CS Help Room

https://cs.barnard.edu/cs-help-room



Students in introductory and intermediate undergraduate courses in computer science can receive one-on-one tutoring through Barnard's Computer Science Help Room.

Location: In-person tutoring sessions are held in Milstein 502. The room can get very busy, so please look for the yellow and blue sign designating a Barnard CS Help Room tutor.

If you have any questions, please email inquiry-cs@barnard.edu.

Fall 2025 Schedule (Monday, September 15 – Friday, December 12)
(updated 9/10/2025)

Monday	**12pm – 8pm**
Tuesday	**12pm – 8pm**
Wednesday	**12pm – 8pm**
Thursday	**12pm – 8pm**
Friday	**12pm – 4pm**

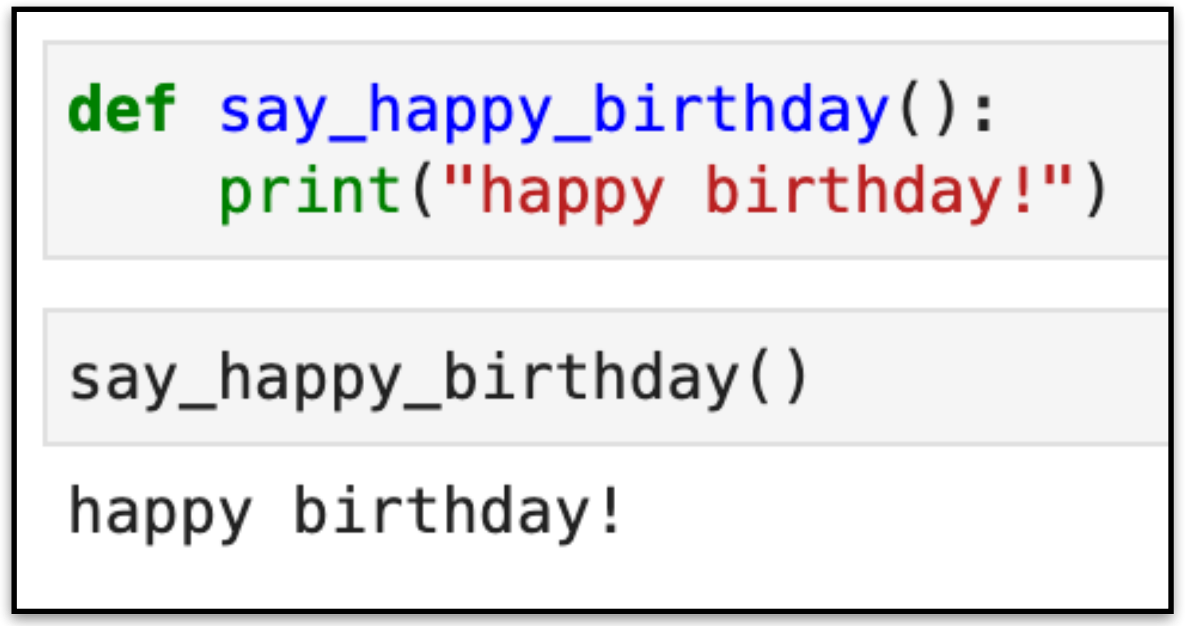
Python Intro

- Last Wednesday
  - Jupyter Notebooks
  - Expressions
  - Data Types
- Monday
  - Tables (and arrays)
- Today
  - Functions
  - Table Review
  - Charts

Functions (and Methods)

Defining functions

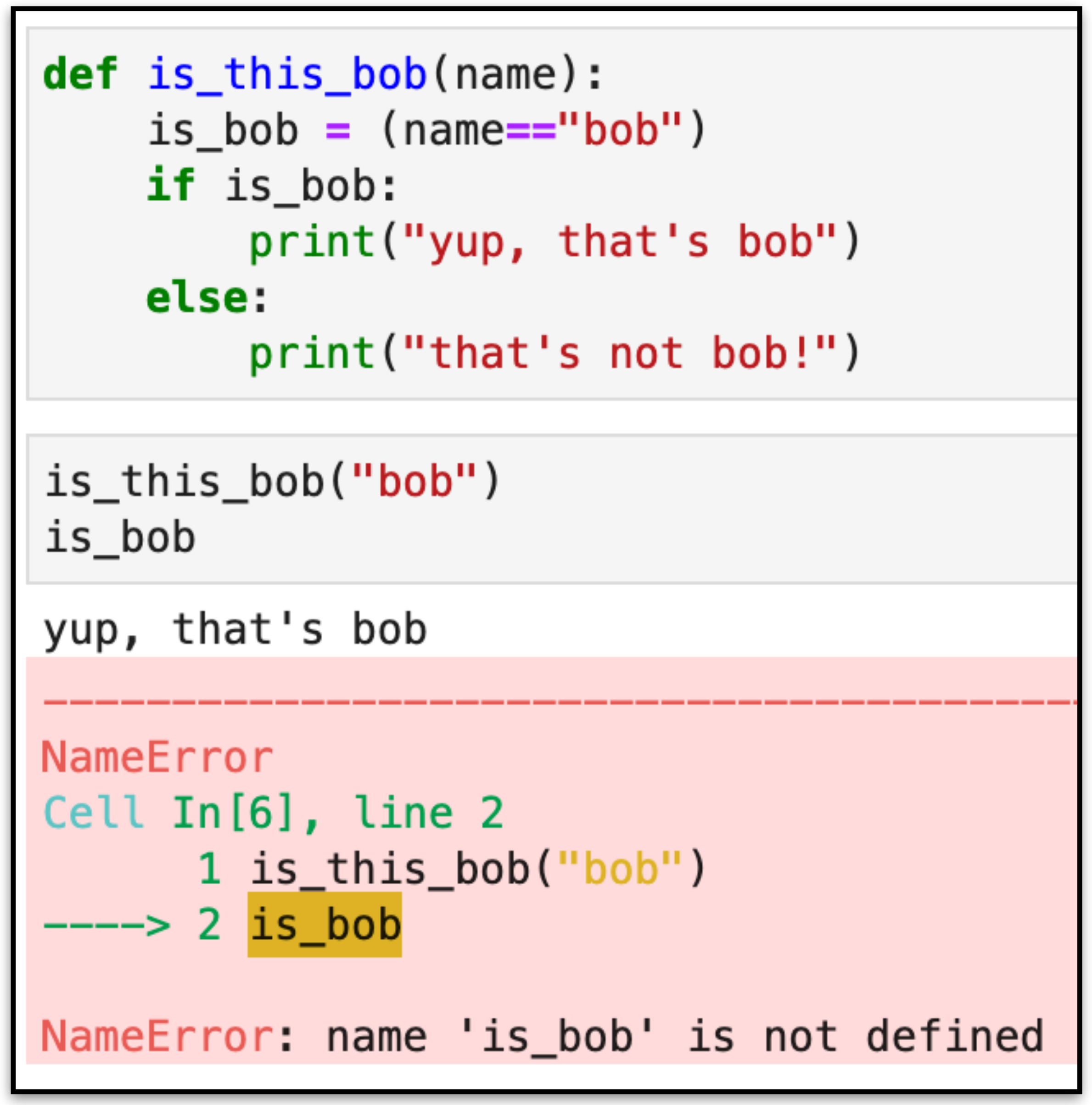
- Use **`def`** to define your own function!
  - The code you want to execute in the function starts on a new line with a single indent
- Variables defined *inside* a function only exist in that function
  - Use **`return`** to have the function output a specific value



```
def say_happy_birthday():
    print("happy birthday!")
```

```
say_happy_birthday()
```

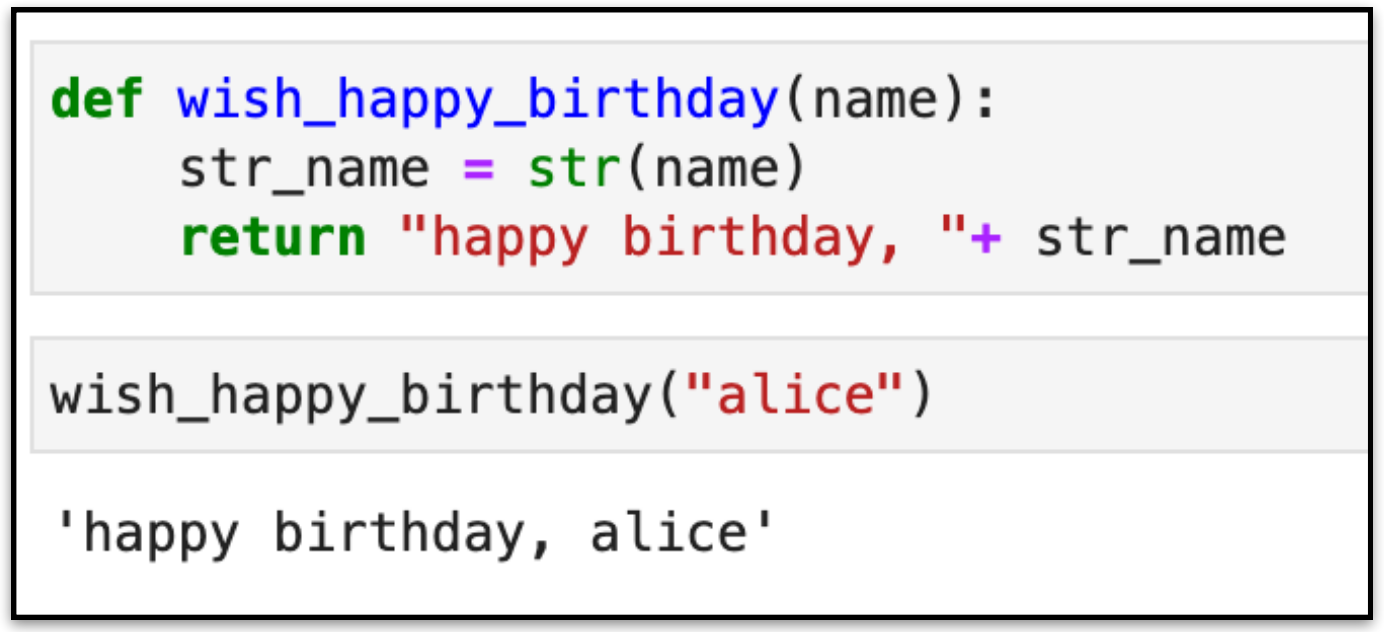
```
happy birthday!
```



```
def is_this_bob(name):
    is_bob = (name=="bob")
    if is_bob:
        print("yup, that's bob")
    else:
        print("that's not bob!")
```

```
is_this_bob("bob")
is_bob
```

```
yup, that's bob
---------------------------------------------------------------------------
NameError
Cell In[6], line 2
      1 is_this_bob("bob")
----> 2 is_bob

NameError: name 'is_bob' is not defined
```



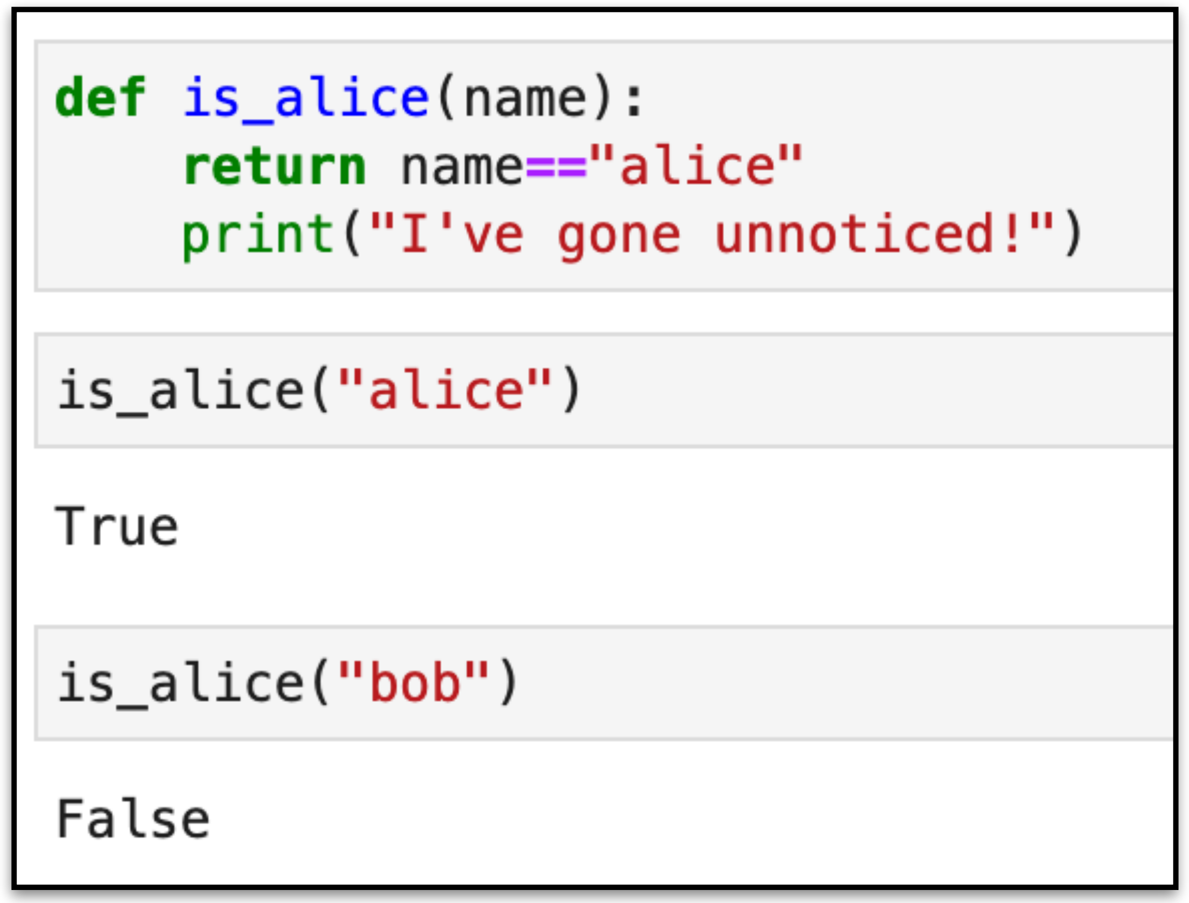
```
def wish_happy_birthday(name):
    str_name = str(name)
    return "happy birthday, "+ str_name
```

```
wish_happy_birthday("alice")
```

```
'happy birthday, alice'
```

Tips for writing functions

- Avoid naming your function something that already exists
- If you find yourself writing the same thing over and over, you probably want to make a function
  - Much easier to edit one place than tracking down everywhere you copied the code!
- `return` will immediately exit a function
  - Typically goes at the end



```
def is_alice(name):
    return name=="alice"
    print("I've gone unnoticed!")
```

```
is_alice("alice")
```

```
True
```

```
is_alice("bob")
```

```
False
```

Terminology: Functions vs Methods

- Functions can be run independently, while methods are associated with an object

Table object

Function	Method

`max(1, 5)`	`skyscrapers = Table.read_table('skyscrapers.csv')`
	
	`skyscrapers.num_rows`

method

Terminology: Functions vs Methods

- It's not just about whether there's a dot!

Array object

Function	Method

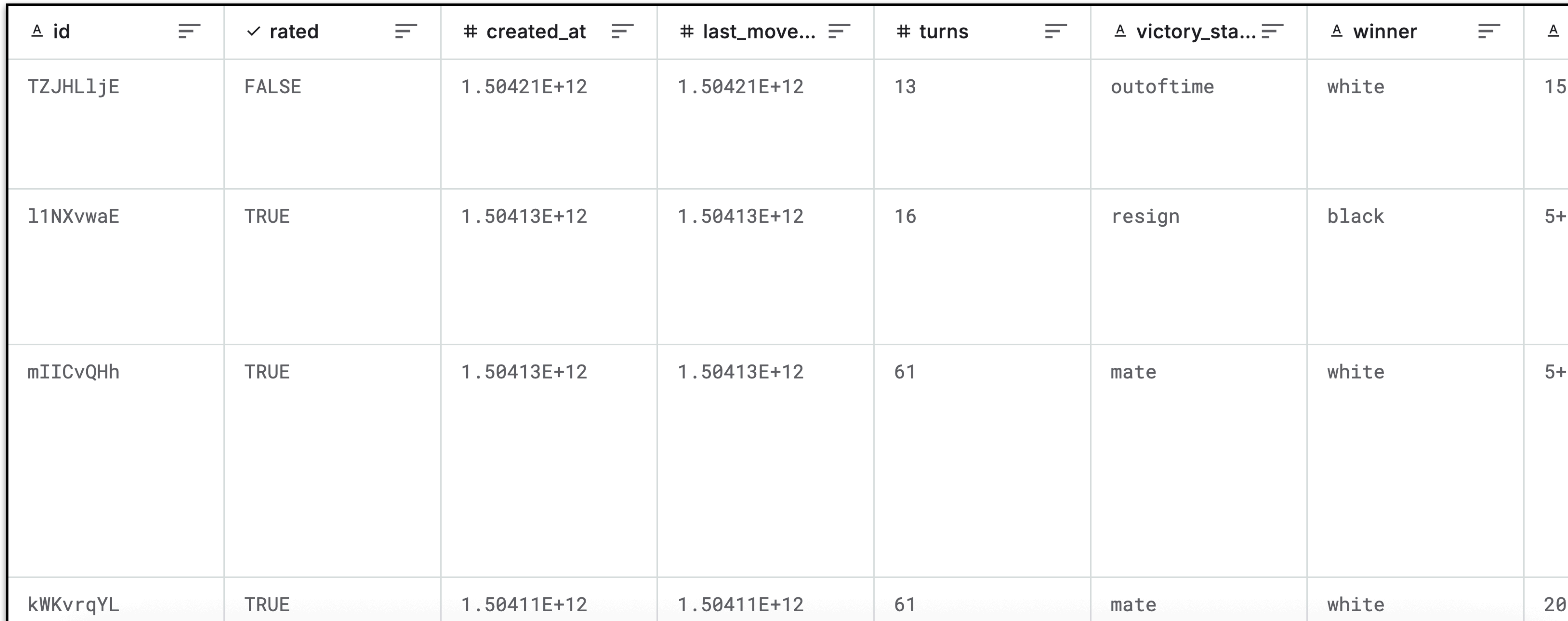
`np.average(make_array(1, 2, 3))`	`my_array = make_array(1, 2, 3)`
	`my_array.item(0)`

NumPy library (not object!)

Tables

Table Review with Chess

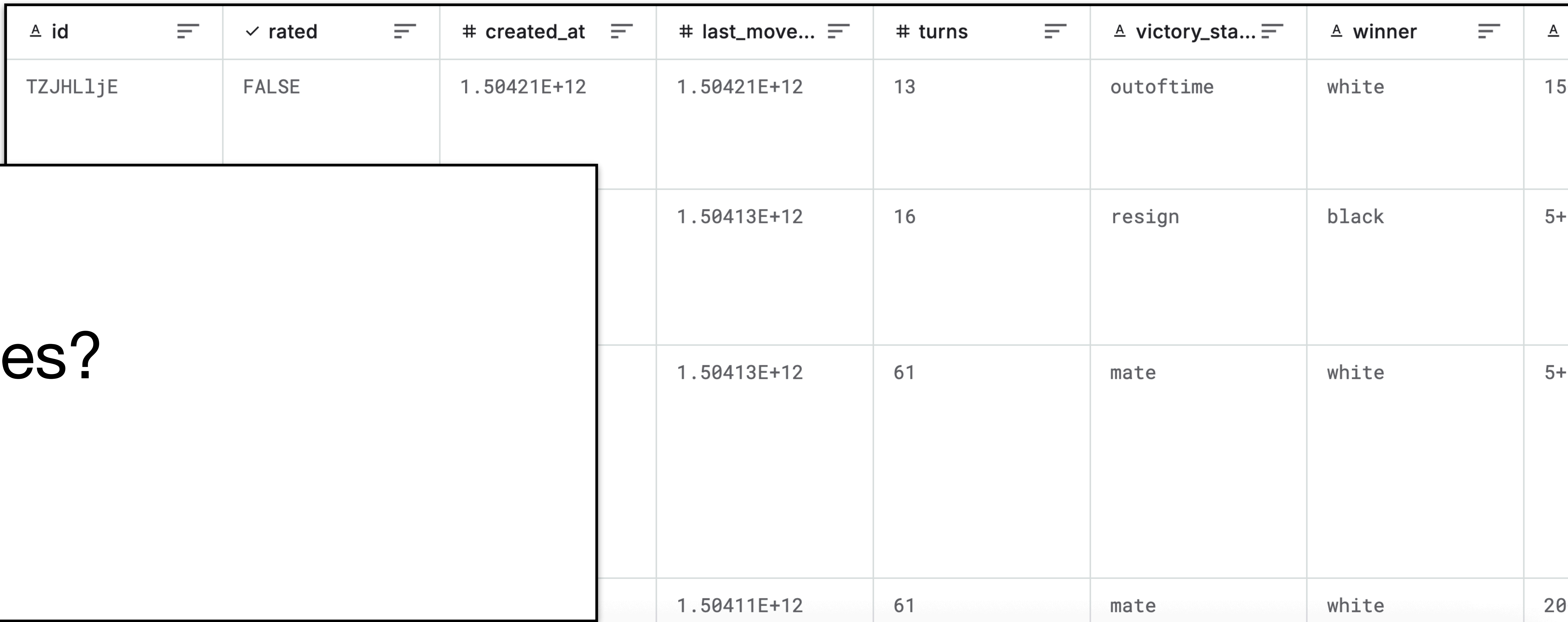
- We're going to look at a data set of some chess games from lichess.com
  - Pieces are black or white
  - Games can end at outoftime, resign, mate, or draw
  - Games are optionally 'rated'



id	rated	created_at	last_move...	turns	victory_sta...	winner	
TZJHLljE	FALSE	1.50421E+12	1.50421E+12	13	outoftime	white	15
l1NXvwaE	TRUE	1.50413E+12	1.50413E+12	16	resign	black	5+
mIICvQHh	TRUE	1.50413E+12	1.50413E+12	61	mate	white	5+
kWKvrqYL	TRUE	1.50411E+12	1.50411E+12	61	mate	white	20

Table Review with Chess

- We’re going to look at a data set of some chess games from lichess.com
  - Pieces are black or white
  - Games can end at outoftime, resign, mate, or draw
  - Games are optionally ‘rated’



id	rated	created_at	last_move...	turns	victory_sta...	winner	
TZJHLljE	FALSE	1.50421E+12	1.50421E+12	13	outoftime	white	15
			1.50413E+12	16	resign	black	5+
			1.50413E+12	61	mate	white	5+
			1.50411E+12	61	mate	white	20

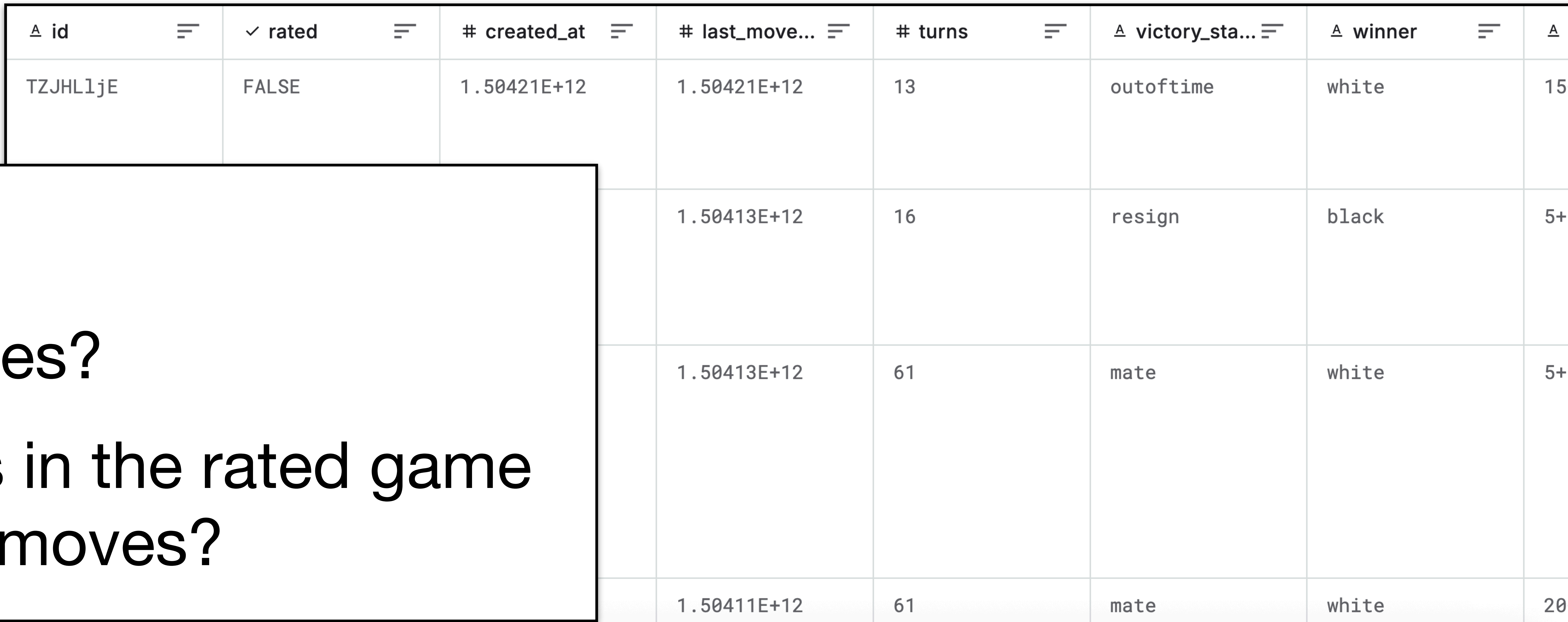
- Questions:
  1. Which color won more games?

Recall: Ways to Create Tables

- Read from a CSV file
  - `Table.read_table(filename)`
- Create a new table from an existing table. Let `tbl` be a table and `c,c1,c2` be column names or indices
  - `tbl.select(c1 ,c2, ...)`
  - `tbl.drop(c1, c2, ...)`
  - `tbl.sort(c[, descending=False])`
  - `tbl.where(c, predicate)` ← Only rows in the table where the value in column c satisfies the predicate
  - `tbl.take(row_indices)`

Table Review with Chess

- We’re going to look at a data set of some chess games from lichess.com
  - Pieces are black or white
  - Games can end at outoftime, resign, mate, or draw
  - Games are optionally ‘rated’



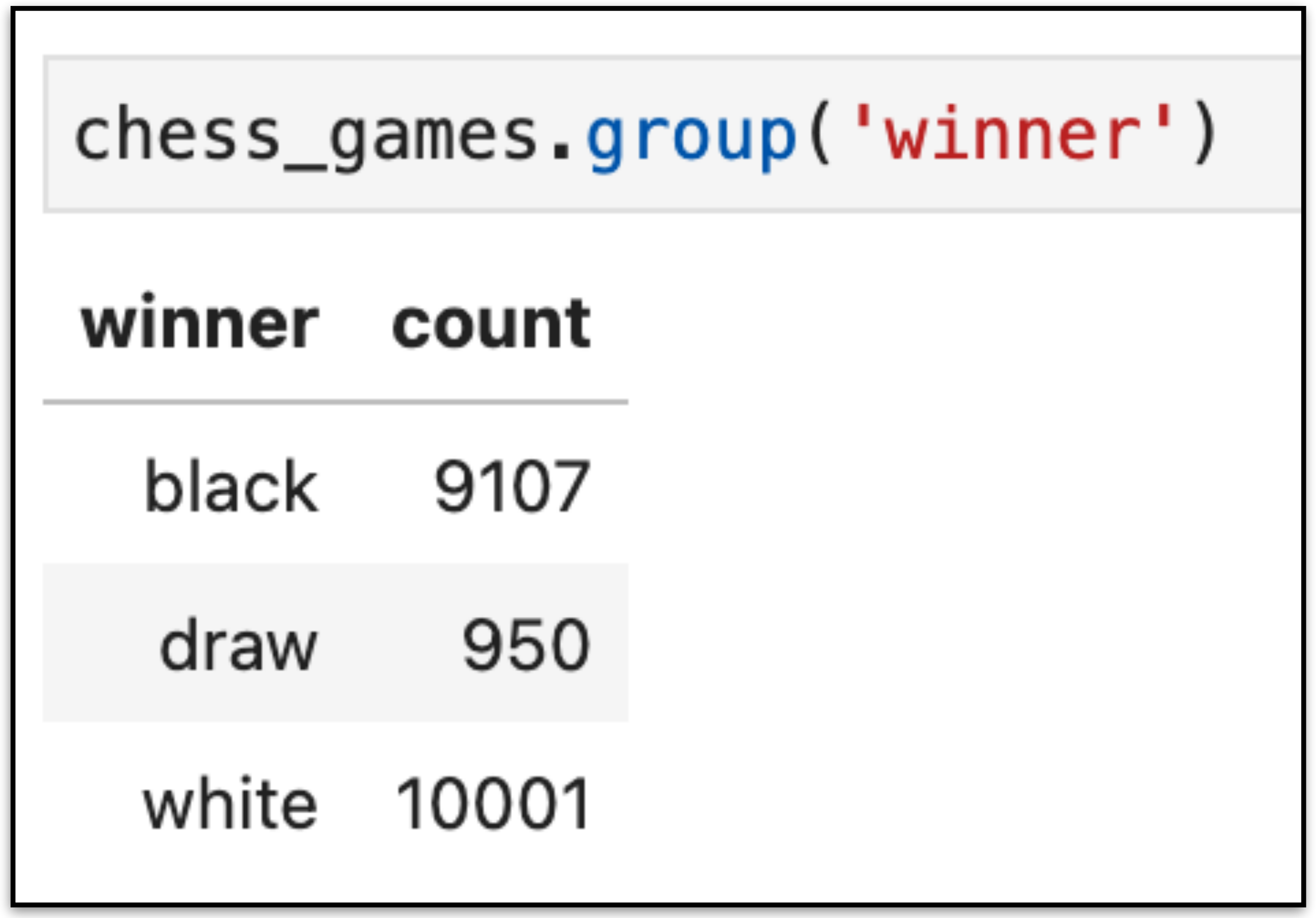
id	rated	created_at	last_move...	turns	victory_sta...	winner	
TZJHLljE	FALSE	1.50421E+12	1.50421E+12	13	outoftime	white	15
			1.50413E+12	16	resign	black	5+
			1.50413E+12	61	mate	white	5+
			1.50411E+12	61	mate	white	20

- Questions:
  1. Which color won more games?
  2. What was the victory status in the rated game with the highest number of moves?

Another Useful Table Method: `group`

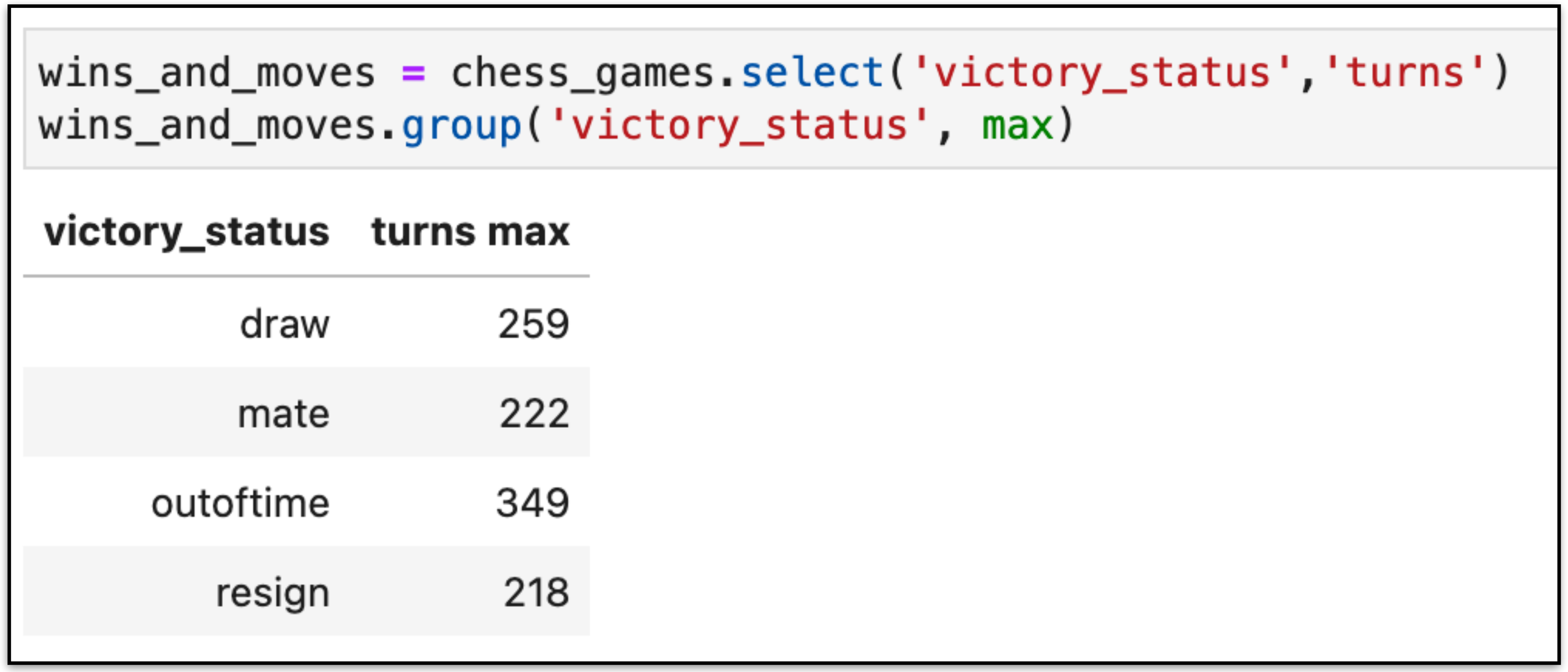
`group` counts the number of rows of each category in a column

- Optionally takes in a function as a second argument and applies to other columns



```
chess_games.group('winner')
```

winner	count
black	9107
draw	950
white	10001



```
wins_and_moves = chess_games.select('victory_status','turns')
wins_and_moves.group('victory_status', max)
```

victory_status	turns max
draw	259
mate	222
outoftime	349
resign	218

Charts

Types of Attributes

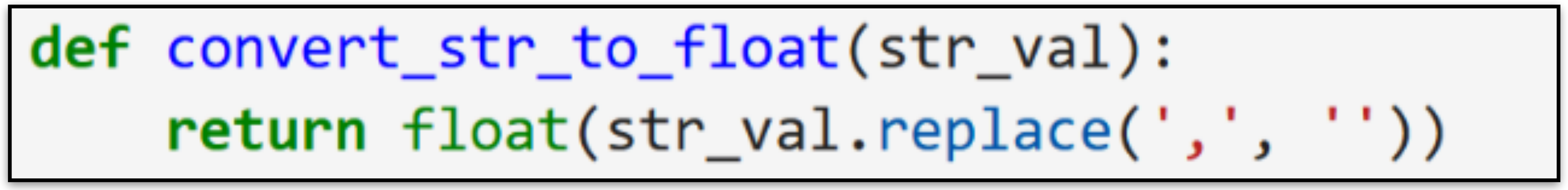
- Attributes are the names of columns in tables
- All values in a column should be the same type and comparable to each other
  - **Numerical -** Values are on a numerical scale (e.g., years)
    - Values are ordered
    - Differences are meaningful
  - **Categorical -** Each value is from a fixed inventory (e.g., material)
    - May not have an ordering
    - Categories are either the same or different

Numerical Caveat

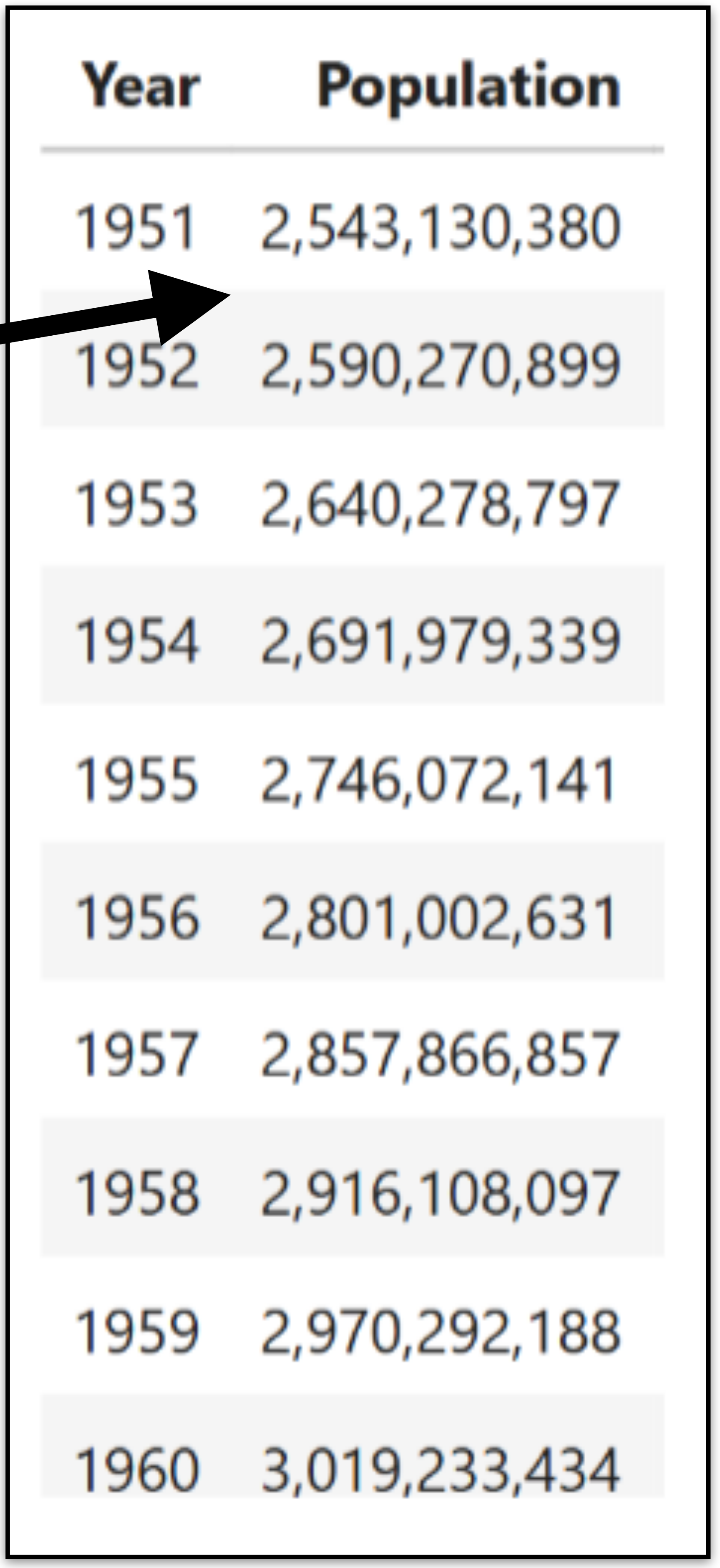
- Values that are numbers are not necessarily numerical
- Example: Sometimes people use numbers instead of strings to represent categories
  - Example: 0, 1 for false, true

Functions (continued)

- Sometimes even the numerical data can be stored as a string
  - Notice the `,`?
- To analyze this, we might need to convert that string to a numerical value
- How can we do that?



```python
def convert_str_to_float(str_val):
    return float(str_val.replace(',', ''))
```



Year	Population
1951	2,543,130,380
1952	2,590,270,899
1953	2,640,278,797
1954	2,691,979,339
1955	2,746,072,141
1956	2,801,002,631
1957	2,857,866,857
1958	2,916,108,097
1959	2,970,292,188
1960	3,019,233,434

Functions (continued)

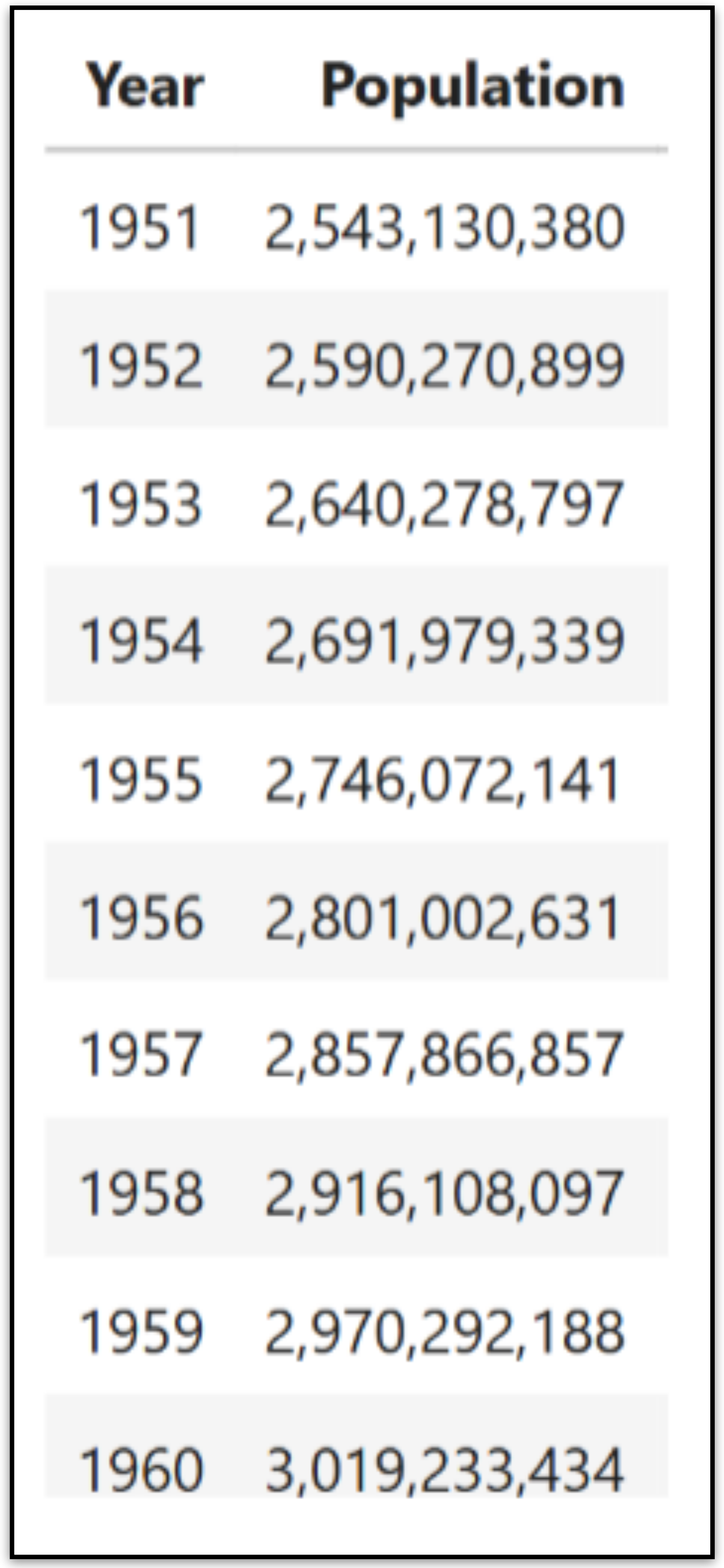
Once we define a function `convert_str_to_float`, two options for converting this:

1. Manually apply the function to each item

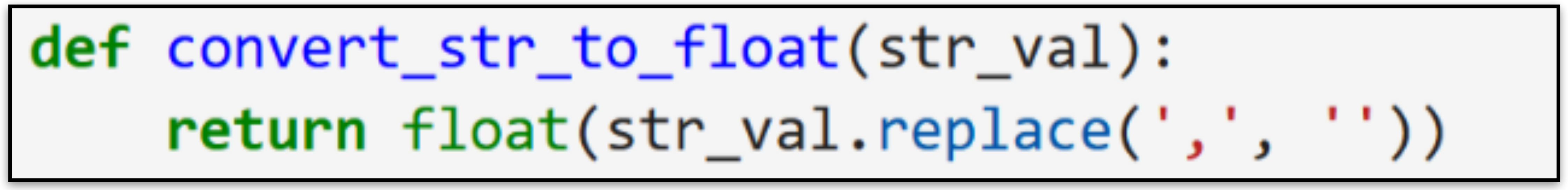
```
item0 =
tbl.column('Population').item(0)

convert_str_to_float(item0)
```

2. Use apply to this function to all values

```
tbl.apply(convert_str_to_float,
'Population')
```



Year	Population
1951	2,543,130,380
1952	2,590,270,899
1953	2,640,278,797
1954	2,691,979,339
1955	2,746,072,141
1956	2,801,002,631
1957	2,857,866,857
1958	2,916,108,097
1959	2,970,292,188
1960	3,019,233,434



```
def convert_str_to_float(str_val):
    return float(str_val.replace(',', ''))
```

Plot Notebook Demo

Line vs Scatter

- Line plots are good for sequential data if
  - x-axis has an order (e.g., time, years, distance)
  - sequential differences in y value are meaningful
  - there’s only one y-value for each x-value
- Use scatter plot for non-sequential quantitative data
  - great for looking for associations

Next Class

- More charts