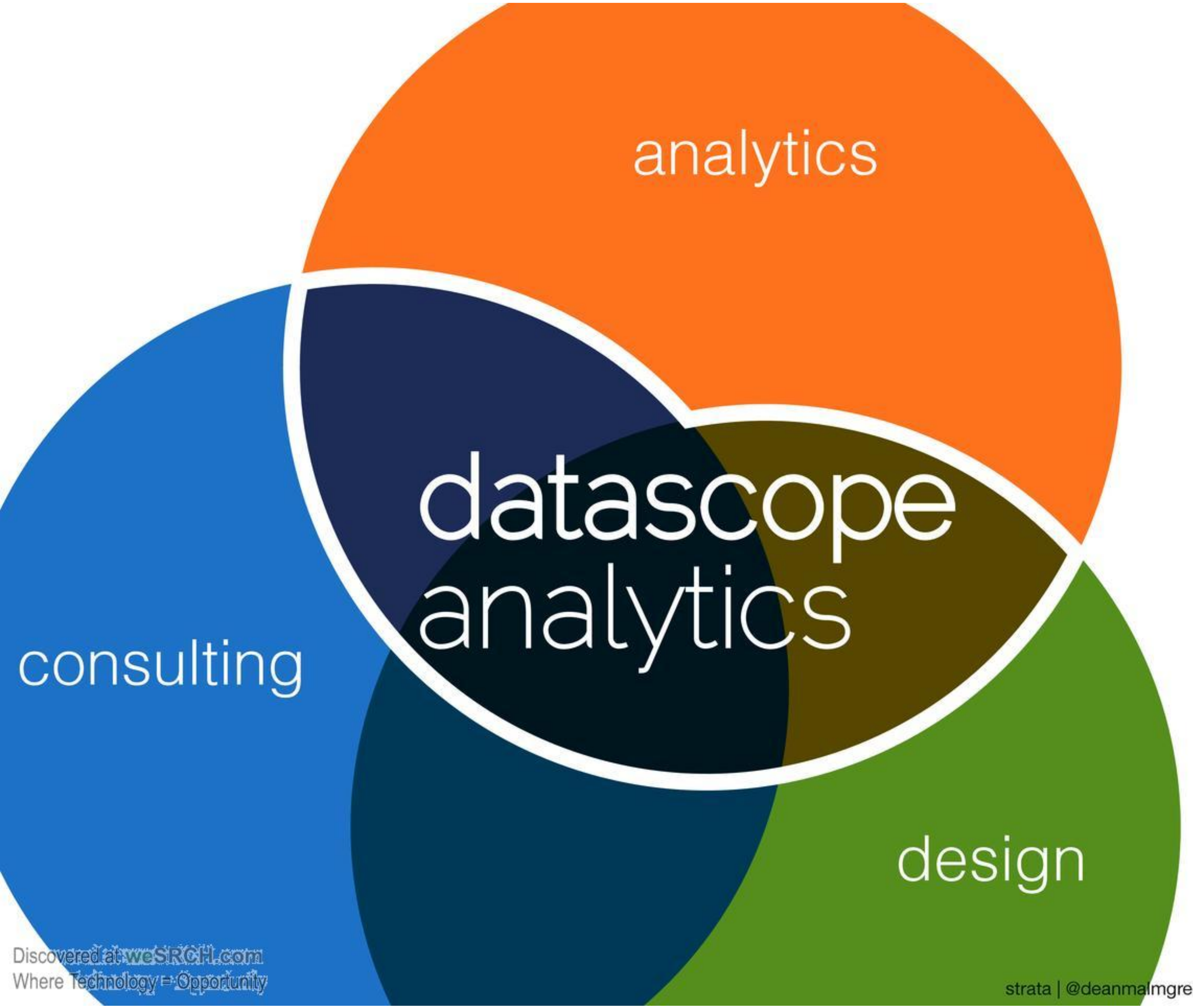
Translating SQL to pandas. And back.

PyData NYC
November 24, 2014

Greg Reda

- @gjreda
- gregreda.com
- Studied economics
- Led data at GrubHub
- Data Scientist at Datascope Analytics



analytics

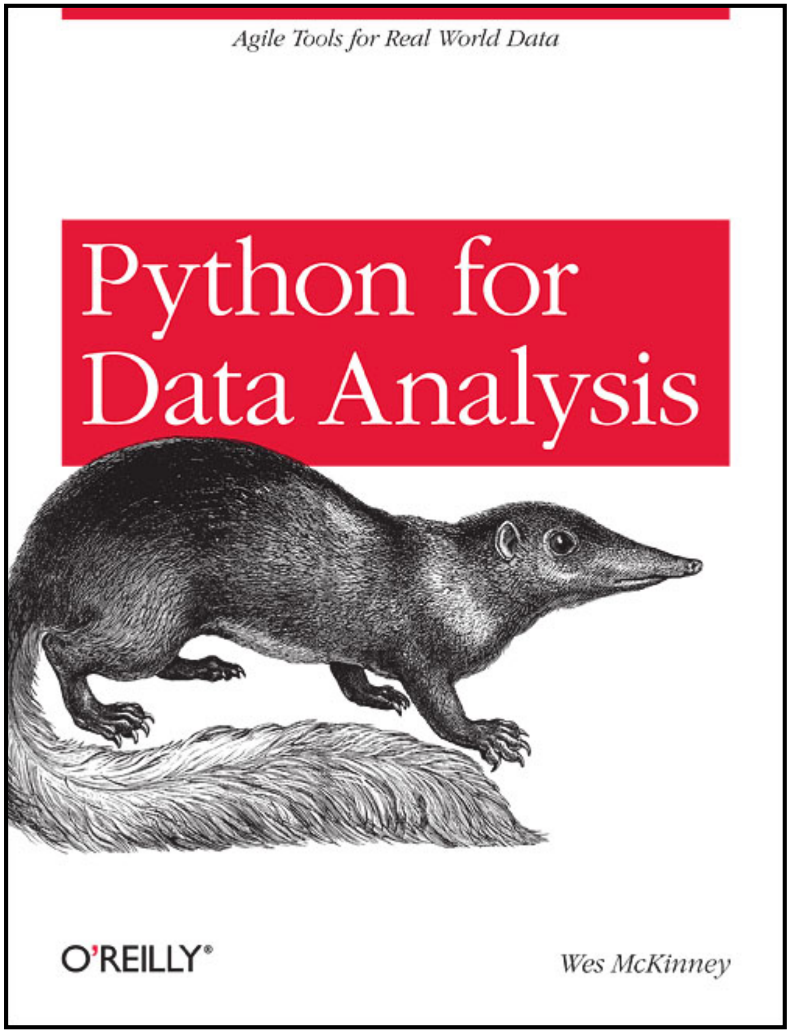
consulting

design

datascope analytics

pandas

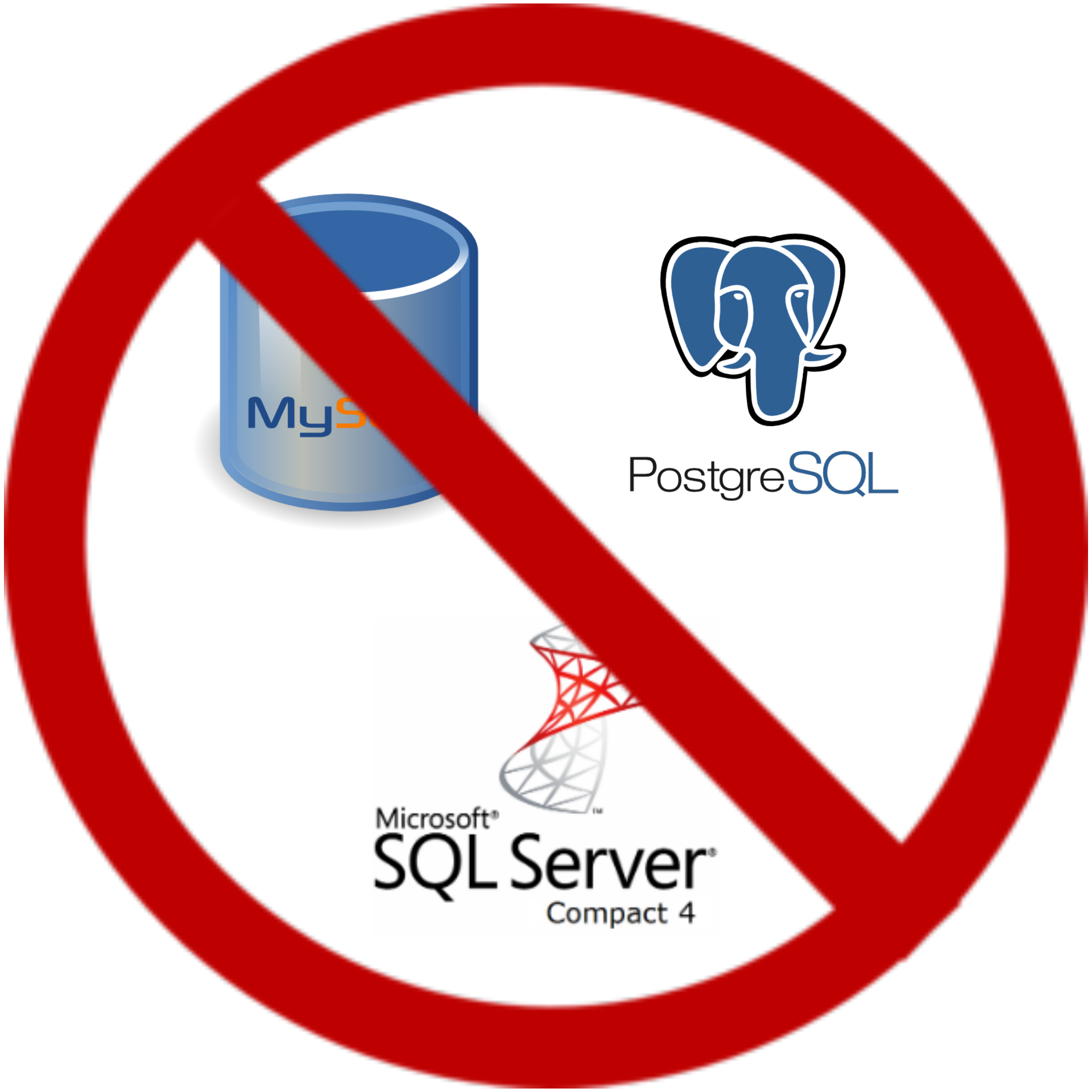
- Started by Wes McKinney in 2008
- Python lacked data *analysis* capabilities
- Built on top of NumPy (that means it’s fast)
- 300+ contributors
- Big and active community led by Jeff Reback



PANDA
ZOO
GIANT PANDAS
ZOO

pandas is PyData glue

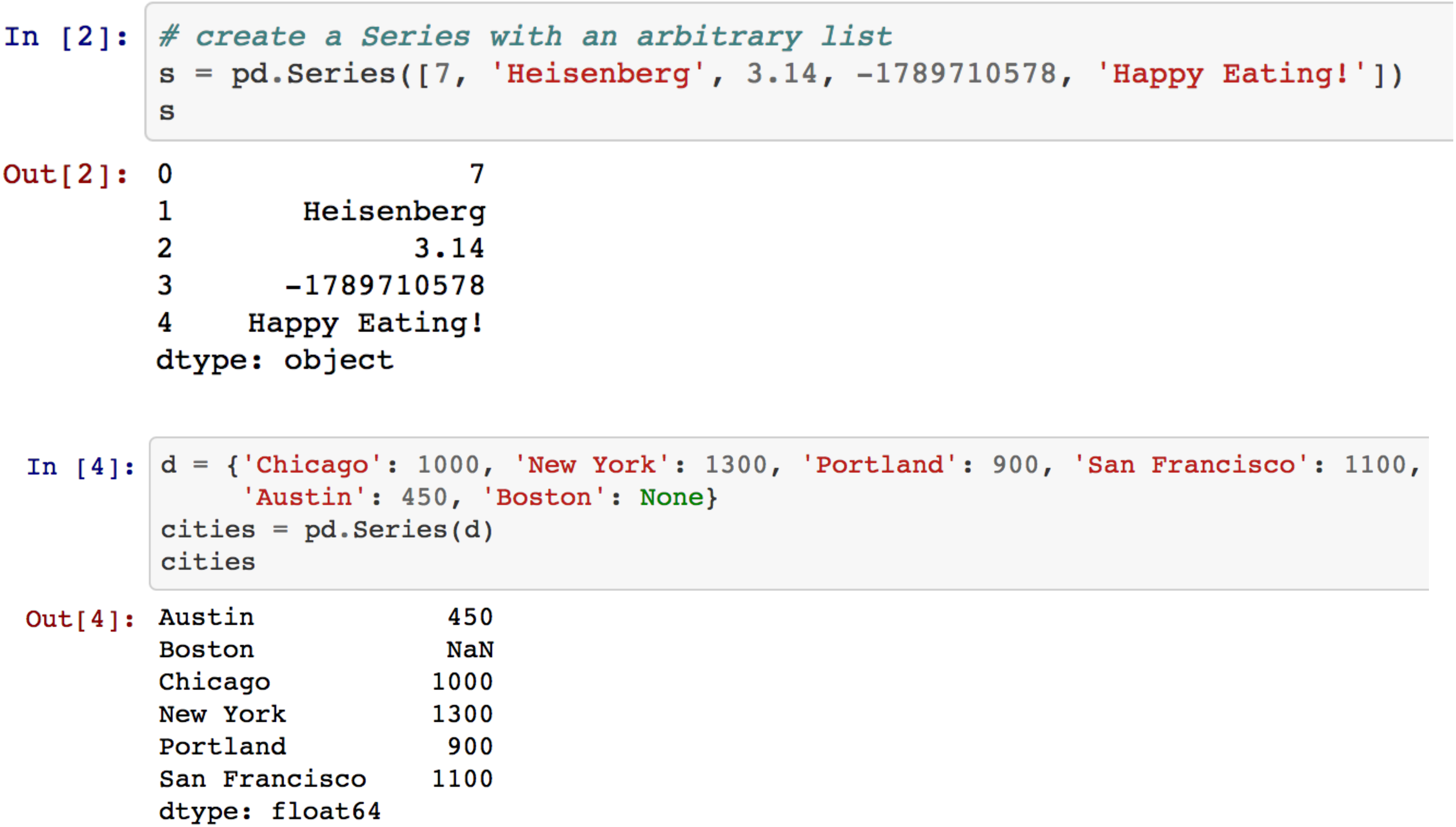
What ***isn't*** pandas?



What we'll cover

- Data structures
    - Series & DataFrames
    - Indexes
- I/O - getting your data in and out of pandas
- Working with DataFrames
- Applied analysis (using IPython Notebook)

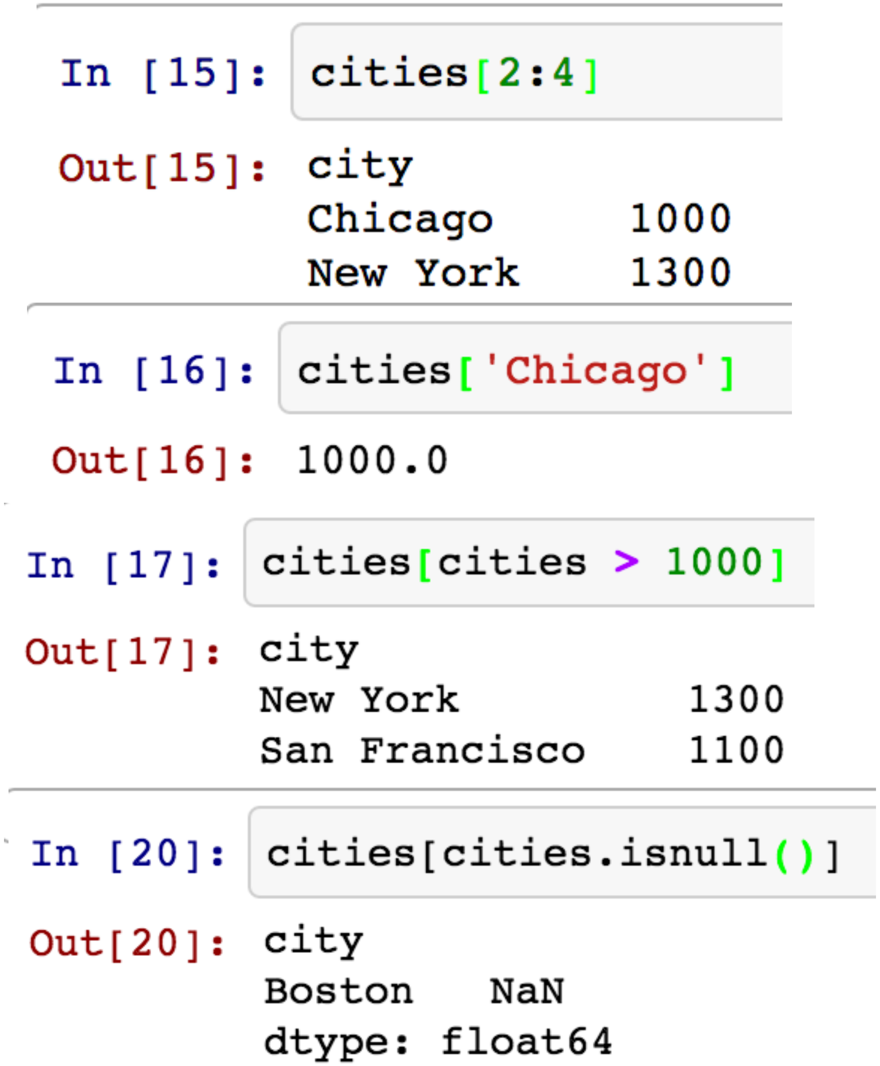
Series



```
In [2]: # create a Series with an arbitrary list
        s = pd.Series([7, 'Heisenberg', 3.14, -1789710578, 'Happy Eating!'])
        s

Out[2]: 0                7
        1       Heisenberg
        2             3.14
        3      -1789710578
        4    Happy Eating!
        dtype: object
```

```
In [4]: d = {'Chicago': 1000, 'New York': 1300, 'Portland': 900, 'San Francisco': 1100,
             'Austin': 450, 'Boston': None}
        cities = pd.Series(d)
        cities

Out[4]: Austin            450
        Boston            NaN
        Chicago          1000
        New York         1300
        Portland          900
        San Francisco    1100
        dtype: float64
```

Series slicing



```
In [15]: cities[2:4]

Out[15]: city
         Chicago      1000
         New York     1300
```

```
In [16]: cities['Chicago']

Out[16]: 1000.0
```

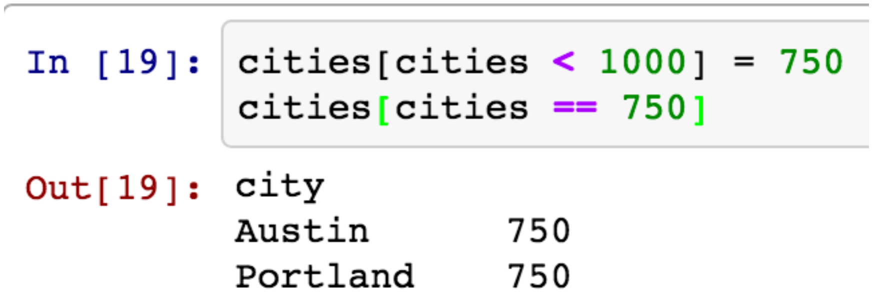
```
In [17]: cities[cities > 1000]

Out[17]: city
         New York          1300
         San Francisco     1100
```

```
In [20]: cities[cities.isnull()]

Out[20]: city
         Boston    NaN
         dtype: float64
```

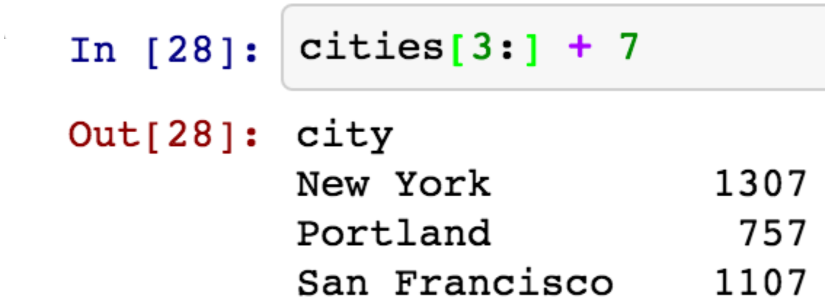
- Standard Python slicing
- Using the index
- Boolean slicing

Series operations

- Changing values

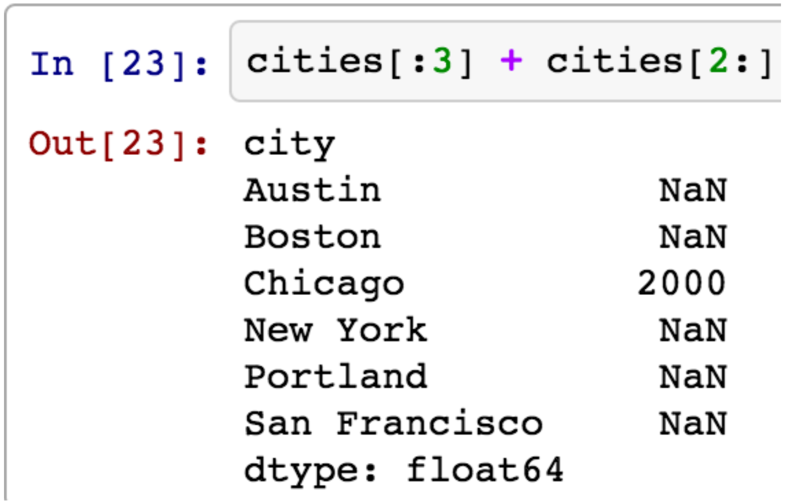


```
In [19]: cities[cities < 1000] = 750
         cities[cities == 750]

Out[19]: city
         Austin      750
         Portland    750
```



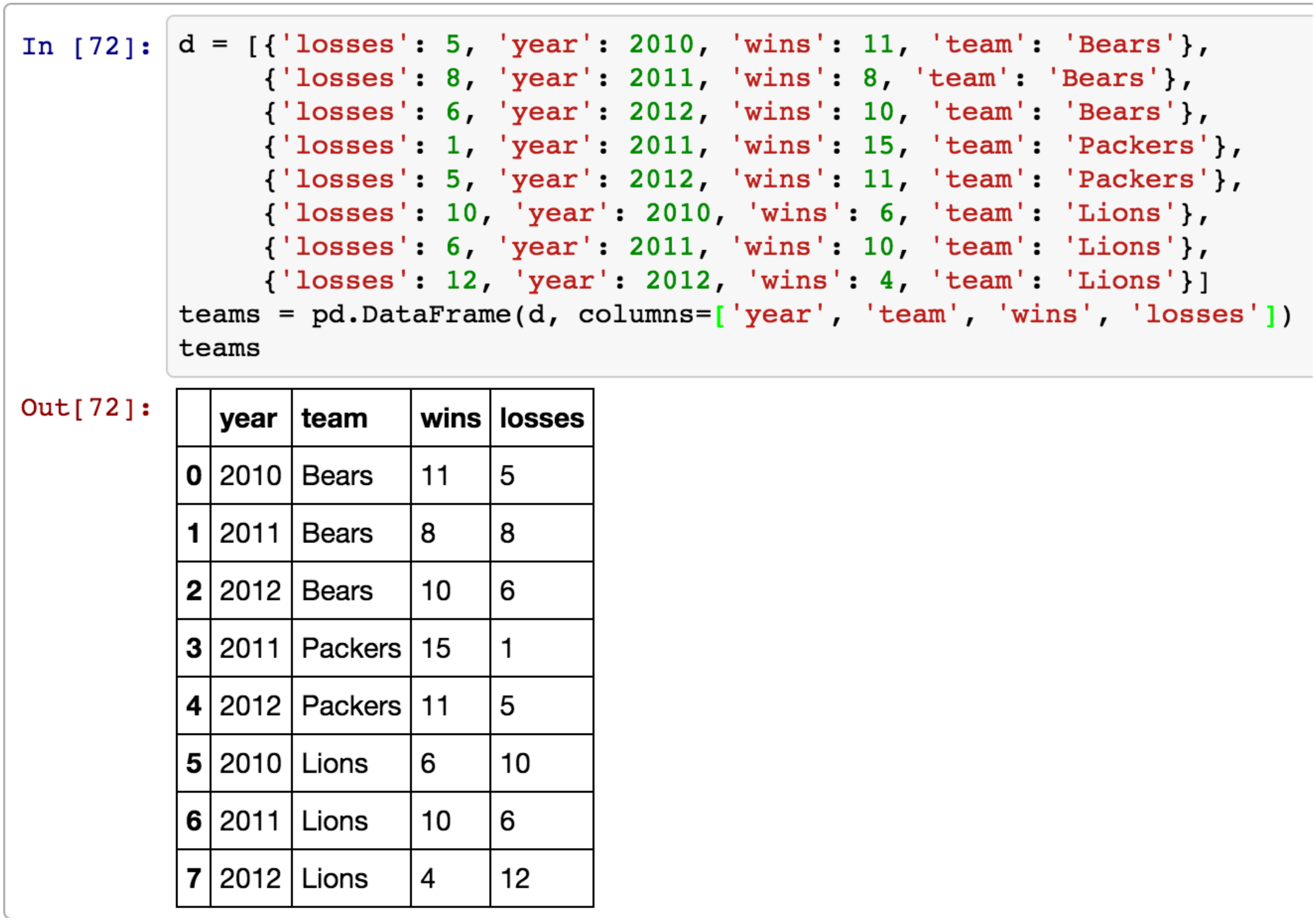
```
In [28]: cities[3:] + 7

Out[28]: city
         New York         1307
         Portland          757
         San Francisco    1107
```

- Math



```
In [23]: cities[:3] + cities[2:]

Out[23]: city
         Austin            NaN
         Boston            NaN
         Chicago          2000
         New York          NaN
         Portland          NaN
         San Francisco     NaN
         dtype: float64
```

DataFrames



In [72]:

```
d = [{'losses': 5, 'year': 2010, 'wins': 11, 'team': 'Bears'},
     {'losses': 8, 'year': 2011, 'wins': 8, 'team': 'Bears'},
     {'losses': 6, 'year': 2012, 'wins': 10, 'team': 'Bears'},
     {'losses': 1, 'year': 2011, 'wins': 15, 'team': 'Packers'},
     {'losses': 5, 'year': 2012, 'wins': 11, 'team': 'Packers'},
     {'losses': 10, 'year': 2010, 'wins': 6, 'team': 'Lions'},
     {'losses': 6, 'year': 2011, 'wins': 10, 'team': 'Lions'},
     {'losses': 12, 'year': 2012, 'wins': 4, 'team': 'Lions'}]
teams = pd.DataFrame(d, columns=['year', 'team', 'wins', 'losses'])
teams
```

Out[72]:

	year	team	wins	losses
0	2010	Bears	11	5
1	2011	Bears	8	8
2	2012	Bears	10	6
3	2011	Packers	15	1
4	2012	Packers	11	5
5	2010	Lions	6	10
6	2011	Lions	10	6
7	2012	Lions	4	12

Indexes

They’re not columns.

Demo Time!

(Live coding is such a bad idea …)