Data Science for Linguists: The *pandas* library

John R. Starr, jrs294@pitt.edu

Pandas Series

A short definition: “one-dimensional array of indexed data.”

What does a series do differently than:

- A list?
  - Index can be a non-integer object
- A one-dimensional NumPy array?
  - Has an explicitly-defined index
- A dictionary?
  - Allows slicing

```
In [11]: population_dict = {'California': 38332521,
                            'Texas': 26448193,
                            'New York': 19651127,
                            'Florida': 19552860,
                            'Illinois': 12882135}
         population = pd.Series(population_dict)
         population
```

```
Out[11]: California    38332521
         Florida       19552860
         Illinois      12882135
         New York      19651127
         Texas         26448193
         dtype: int64
```

Pandas DataFrame

Another short definition: “an analog of a two-dimensional array with both flexible row indices and flexible column names”

Visually-similar to a spreadsheet, with:

- Functionality similar to a Python dictionary
- Mutable column/row names
- Index objects are immutable and cannot be changed with a simple index[num] = ____.

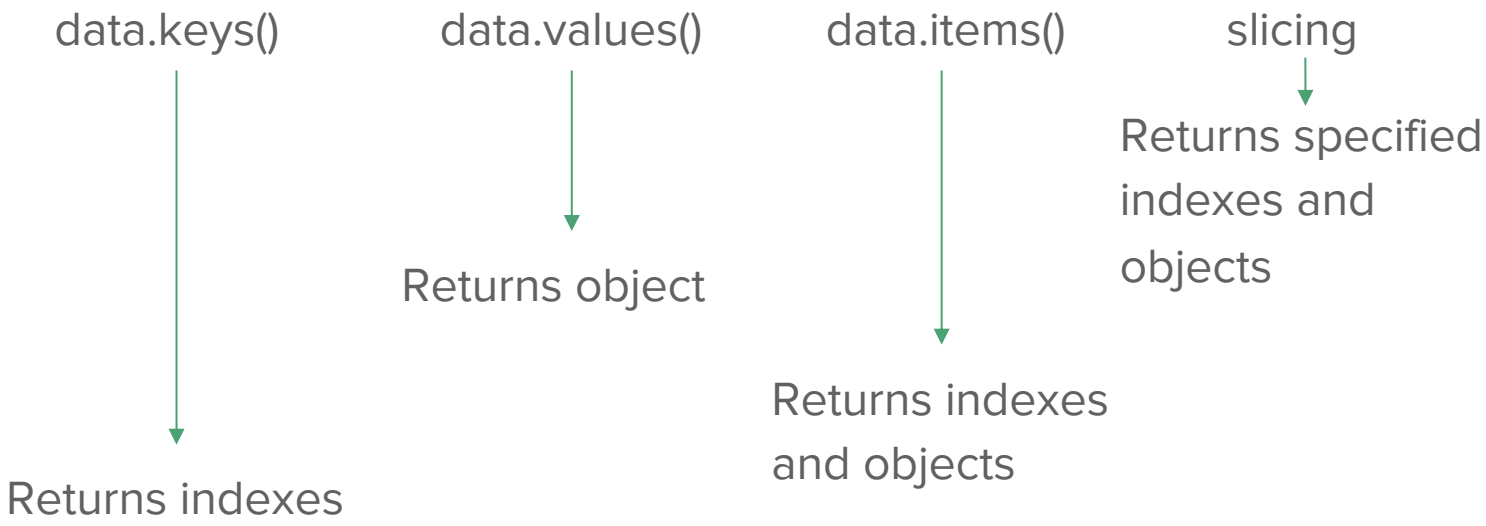
```
In [26]: pd.DataFrame({'population': population,
                       'area': area})
```

Out[26]:

	area	population
California	423967	38332521
Florida	170312	19552860
Illinois	149995	12882135
New York	141297	19651127
Texas	695662	26448193

Data Selection

For Series:



loc: slices Series/DataFrame by row through explicit index

```
In [29]: data.loc[:'Illinois', :'pop']
```

Out[29]:

	area	pop
California	423967	38332521
Florida	170312	19552860
Illinois	149995	12882135

iloc: slices Series/DataFrame by row through inherent integer index

```
In [28]: data.iloc[:3, :2]
```

Out[28]:

	area	pop
California	423967	38332521
Florida	170312	19552860
Illinois	149995	12882135

Data Selection with LOC and ILOC

Some help: looking at your DataFrame (df)

Flash top n-number of items:	df.head(n) (default, n = 5)
Flash bottom n-number of items:	df.tail(n) (default, n = 5)
Flash general information about df:	df.info()
Flash general statistics about df:	df.describe()
Flash value counts of Series:	df.Name.value_counts()

More help: maneuvering through your DataFrame (df)

Examining one column/Series:	df[‘Name’] OR df.Name
Finding one/more rows: df.iloc[index_num] OR slicing	df.loc[‘Index’] OR
Finding match with all values in a Series:	df[df[‘Name’] == value]
Selecting unique items (similar to set):	df[‘Name’].unique()

A little more help: working with columns

Creating a new DataFrame from already existing columns:

df2 = df1[[‘Col1’, ‘Col2’, ‘Col3’]]

Adding a new column from a Series:

df[‘Series_Name’] = Series

Renaming a column:

df.rename[columns = [‘Col1’, ‘Col2’]]