CLEANING DATA IN PYTHON

Putting it all together

Putting it all together

- Use the techniques you've learned on Gapminder data
- Clean and tidy data saved to a file
  - Ready to be loaded for analysis!
- Dataset consists of life expectancy by country and year
- Data will come in multiple parts
  - Load
  - Preliminary quality diagnosis
  - Combine into single dataset

Useful methods

```
In [1]: import pandas as pd

In [2]: df = pd.read_csv('my_data.csv')

In [3]: df.head()

In [4]: df.info()

In [5]: df.columns

In [6]: df.describe()

In [7]: df.column.value_counts()

In [8]: df.column.plot('hist')
```

Data quality

```
In [9]: def cleaning_function(row_data):
   ...:     # data cleaning steps
   ...:     return ...

In [10]: df.apply(cleaning_function, axis=1)

In [11]: assert (df.column_data > 0).all()
```

each row

Combining data

- pd.merge(df1, df2, ...)
- pd.concat([df1, df2, df3, ...])

CLEANING DATA IN PYTHON

Let’s practice!

CLEANING DATA IN PYTHON

Initial impressions of the data

Principles of tidy data

- Rows form observations
- Columns form variables
- Tidying data will make data cleaning easier
- Melting turns columns into rows
- Pivot will take unique values from a column and create new columns

Checking data types

```
In [1]: df.dtypes

In [2]: df['column'] = df['column'].to_numeric()

In [3]: df['column'] = df['column'].astype(str)
```

Additional calculations and saving your data

```
In [4]: df['new_column'] = df['column_1'] + df['column_2']

In [5]: df['new_column'] = df.apply(my_function, axis=1)   row-wise

In [6]: df.to_csv['my_data.csv']
```

CLEANING DATA IN PYTHON

Let’s practice!

CLEANING DATA IN PYTHON

Final thoughts

You've learned how to...

- Load and view data in pandas
- Visually inspect data for errors and potential problems
- Tidy data for analysis and reshape it
- Combine datasets
- Clean data by using regular expressions and functions
- Test your data and be proactive in finding potential errors

CLEANING DATA IN PYTHON

Congratulations!