Concept 1.4: Data Types and Data Wrangling

- Working with different types of data: text files, CSV, JSON objects, HTML and databases.

```
csv_df = pd.read_csv('sample_file.csv')
csv_df.to_csv('sample_file.csv', index=False)

#sometimes dependent on the xlrd library which can be installed by running pip install xlrd in the terminal
excel_df = pd.read_excel('sample_file.xlsx')
excel_df.to_excel('sample_file.xlsx')

#read table from a webpage and save as a dataframe
html_df = pd.read_html('http://www.webpage.com/sampledata.html')
html_df.to_html('sample_file.html')
```

Pandas can connect to databases, get data with queries and save in a dataframe.

```
url='https://github.com/WalePhenomenon/climate_change/blob/master/fuel_ferc1.csv?raw=true'
fuel_data = pd.read_csv(url, error_bad_lines=False)
fuel_data.describe(include='all')

#check for missing values
fuel_data.isnull().sum()

#use groupby to count the sum of each unique value in the fuel unit column
fuel_data.groupby('fuel_unit')['fuel_unit'].count()
fuel_data[['fuel_unit']] = fuel_data[['fuel_unit']].fillna(value='mcf')


#check if missing values have been filled
fuel_data.isnull().sum()

fuel_data.groupby('report_year')['report_year'].count()
#group by the fuel type code year and print the first entries in all the groups formed
fuel_data.groupby('fuel_type_code_pudl').first()
```

Merging in Pandas can be likened to join operations in relational databases like SQL.

```
fuel_df1 = fuel_data.iloc[0:19000].reset_index(drop=True)
fuel_df2 = fuel_data.iloc[19000:].reset_index(drop=True)
```

```
#check that the length of both dataframes sum to the expected length
assert len(fuel_data) == (len(fuel_df1) + len(fuel_df2))

#an inner merge will lose rows that do not match in both dataframes
pd.merge(fuel_df1, fuel_df2, how="inner")

#outer merge returns all rows in both dataframes
pd.merge(fuel_df1, fuel_df2, how="outer")

#removes rows from the right dataframe that do not have a match with the left
#and keeps all rows from the left
pd.merge(fuel_df1, fuel_df2, how="left")
```

Concatenation is performed with the `concat()` function

```
pd.concat([fuel_data, data_to_concat]).reset_index(drop=True)
```

Duplicates are a common occurrence in datasets which alter the results of data analysis.

```
#check for duplicate rows
fuel_data.duplicated().any()
```