Importing Data	Exploring Data
pd.read_csv(filename) # From a CSV file pd.read_table(filename) # From a delimited TSV file pd.read_excel(filename) # From an Excel file pd.read_sql(query, connection_object) # Reads from a SQL table/database pd.read_json(json_string) # Reads from a JSON formatted string, URL or file. pd.read_html(url) # Parses an html URL to a list of dataframes pd.read_clipboard() # Takes the contents of clipboard and passes it to read_table() pd.DataFrame(dict) # From a dict, keys for columns names, values for data as lists	df.shape() # Prints number of rows and columns in dataframe df.head(n) # Prints first n rows of the DataFrame df.tail(n) # Prints last n rows of the DataFrame df.info() # Index, Datatype and Memory information df.describe() # Summary statistics for numerical columns s.value_counts(dropna=False) # Views unique values and counts df.apply(pd.Series.value_counts) # Unique values and counts for all columns df.describe() # Summary statistics for numerical columns df.mean() # Returns the mean of all columns df.corr() # Returns the correlation between columns in a DataFrame
Selecting Data	

df[col] # Returns column with label col as Series

df[[col1, col2]] # Returns Columns as a new DataFrame

s.iloc[0] # Selection by position (selects first element)

s.loc[0] # Selection by index (selects element at index 0)

df.iloc[0,:] # First row df.iloc[0,0] # First element of first column
df[df[col] > 0.5] # Rows where the col column is greater than 0.5

df[(df[col] > 0.5) & (df[col] < 0.7)] # Rows where 0.5 < col < 0.7

df.count() # Returns the number of non-null values in each DataFrame column

df.max() # Returns the highest value in each column

df.min() # Returns the lowest value in each column

df.median() # Returns the median of each column

df.std() # Returns the standard deviation of each column

Filter, Sort, and Group By

df.sort_values(col1) # Sorts values by col1 in ascending order

df.sort_values(col2,ascending=False) # Sorts values by col2 in descending order

df.sort_values([col1,col2], ascending=[True,False]) # Sorts values by col1 ascending col2 descending

df.groupby(col) # Returns a groupby object for values from one column

df.groupby([col1,col2]) # Returns a groupby object values from multiple columns
df.groupby(col1)[col2].mean() # Returns the mean of the values in col2, grouped by the values in col1

df.pivot_table(index=col1, values= col2,col3], aggfunc=mean) # Creates a pivot table

Data Cleaning

df.columns = ['a','b','c'] # Renames columns

pd.isnull() # Checks for null Values, Returns Boolean Array

pd.notnull() # Opposite of s.isnull()

df.dropna() # Drops all rows that contain null values

df.dropna(axis=1) # Drops all columns that contain null values

df.dropna(axis=1,thresh=n) # Drops all rows have have less than n non null values

df.fillna(x) # Replaces all null values with x

s.fillna(s.mean()) # Replaces all null values with the mean

s.astype(float) # Converts the datatype of the series to float

df.groupby(col1).agg(np.mean) # Finds the average across all columns for every unique column 1 df.apply(np.mean) # Applies a function across each column df.apply(np.max, axis=1) # Applies a function across each row	s.replace(1,'one') # Replaces all values equal to 1 with 'one'