Import Data

Function	Description
`pd.DataFrame()`	create dataframe
`pd.read_csv(filename)`	import from csv
`pd.read_excel(filename)`	import from Excel
`pd.read_sql(query,connection_object)`	import from a SQL database
`pd.read_html(url)`	import table from a url (website) or html file
`pd.read_json(json_string)`	import from json format
`pd.read_table(filename)`	import from a csv

Export Data

Function	Description
`df.to_csv(filename)`	export to csv
`df.to_excel(filename)`	export from Excel
`df.to_sql(table_name,connection_object)`	export to a SQL database
`df.to_json(filename)`	export to json file

Data Sorting

Function	Description
`df.sort_index(axis=0, ascending=True/False)`	sort index/row by ascending or descending order
`df.sort_index(axis=1, ascending=True/False)`	sort columns by ascending or descending order
`df.sort_values('col1')`	sort entire df by `col1` values
`df['col1'].sort_values()`	sort only column `col1`

Data Grouping

Function	Description
`df.groupby('col1').sum()`	group df by `col1` values then show sum for all other columns
`df.groupby('col1').agg({'col2':'count','col3':'mean'})`	group df by `col1` values then show count for `col2` and mean for `col3`

Combine Data

Function	Description
`df1.append(df2)`	add df2 at the bottom of df1. deprecated, use concat
`df1.merge(df2)`	combine df1 and df2 using unique key, similar to vlookup
`pd.concat([df1,df2], axis = 0)`	axis=0 -> append, axis=1 -> merge

Data Selection

Select 1 Row

Function	Description
`df.loc[row_name]`	`row_name` is row name
`df.iloc[row_index]`	`row_index` is the row index position

Select Multiple Rows

Function	Description
`df.loc[row_name_start:row_name_end]`	`row_name` is the row name
`df.iloc[row_index_start:row_index_end]`	`row_index` is the row index position

Select 1 Column

Function	Description
`df[col]`	`col` is the column name
`df.col_name`	`col_name` is the column name
`df.loc[:, col_name]`	`col_name` is the column name
`df.iloc[:, col_index]`	`col_index` is the column index

Select Multiple Columns

Function	Description
`df[[col1,col2,col3]]`	`col1` is the column name
`df.loc[:, col_name_start:col_name_end]`	`col_name` is the column name
`df.iloc[:, col_index_start:col_index_end]`	`col_index` is the column index position

Get Cells

Function	Description
`df[col][row]`	`col` and `row` are column and row names
`df.loc[row_name, col_name]`	use `row` and `col` names
`df.iloc[row_index, col_index]`	use `row` and `col` index

Data Cleanup

Function	Description
`df.dropna()`	drop rows which contain missing values
`df.dropna(axis = 1)`	drop columns which contain missing values
`df.fillna(value='a')`	fill NA/NaN values using `a` for entire df
`df[col].fillna(value='a')`	fill NA/NaN values using `a` for column `col`
`df[col].astype(int/str/float)`	cast/convert to certain datatype
`df[col].replace('a','b')`	replace all 'a' values with 'b' for column `col`
`df.rename(columns= {'old_name':'new_name'})`	cast/convert to certain datatype
`df.set_index('col')`	set existing column `col` as the index
`df.reset_index()`	reset index, use 0,1,2,3,etc as the index

Data Exploration

Function	Description
`df.head(n)`	show the first `n` rows of data
`df.tail(n)`	show the last `n` rows of data
`df.info()`	show index dtype, columns and memory usage
`df.describe()`	descriptive statistics such as count, mean, std, etc
`df.isnull().any()`	shows if df contains any null value
`df.shape`	shows the # of rows and columns of df
`df.columns`	shows the df column names
`df.index`	shows the df row/index names