# Pandas

**pandas** это высокоуровневая [Python](https://khashtamov.com/category/python/) библиотека для анализа данных. Почему я её называю высокоуровневой, потому что построена она поверх более низкоуровневой библиотеки NumPy (написана на Си), что является большим плюсом в производительности.

import pandas as pd

# Series

Структура/объект Series представляет из себя объект, похожий на питоновский словарь, т.е. отличительной его чертой является наличие ассоциированных меток, т.н. индексов, вдоль каждого элемента из списка.

my\_series = pd.Series([5, 6, 7, 8, 9, 10]) – создание series из списка (автонумерация)  
my\_series[4] -> 9

my\_series2[my\_series2 > 0] – фильтрация

my\_series2[['a', 'b', 'f']] = 0 – групповое присваивание

my\_series3.name = 'numbers' – название объекта  
my\_series3.index.name = 'letters' – название индексов

# DataFrame

Объект DataFrame лучше всего представлять себе в виде обычной таблицы и это правильно, ведь DataFrame является табличной структурой данных. В любой таблице всегда присутствуют строки и столбцы. Столбцами в объекте DataFrame выступают объекты Series, строки которых являются их непосредственными элементами.

df = pd.DataFrame({  
... 'country': ['Kazakhstan', 'Russia', 'Belarus', 'Ukraine'],  
... 'population': [17.04, 143.5, 9.5, 45.5],  
... 'square': [2724902, 17125191, 207600, 603628]  
... }) – создание DataFrame из списков

Объект DataFrame имеет 2 индекса: по строкам и по столбцам. Если индекс по строкам явно не задан (например, колонка по которой нужно их строить), то pandas задаёт целочисленный индекс RangeIndex от 0 до N-1, где N это количество строк в таблице.

>>> df.columns  
Index([u'country', u'population', u'square'], dtype='object')  
>>> df.index  
RangeIndex(start=0, stop=4, step=1)

df.index = ['KZ', 'RU', 'BY', 'UA'] – замена автоиндексов на коды стран  
df.index.name = 'Country Code'

df['country'] – выводит объект Series с индексами от df, т.е. столбец названий стран

df.loc['KZ'] – для вывода по строке по индексу  
df.iloc[0] – для вывода по строке по порядку

df.loc[['KZ', 'RU'], 'population'] – для вывода нескольких строк и нескольких столбцов можно передать списки (в т.ч. и срезы)

df[df.population > 10][['country', 'square']] – фильтрация с выводом нескольких колонок

df['density'] = df['population'] / df['square'] \* 1000000 – создание новой колонки

del df['density'] – удаление колонки

df = df.rename(columns={'Country Code': 'country\_code'}) – переименовывание колонки

# Чтение и запись

pd.read\_csv(‘filename.csv’, sep=’,’) – чтение из csv  
pd.read\_excel(‘filename.xls’) – чтение из excel  
pd.read\_sql(‘filename.sql’) – чтение из sql  
pd.read\_json(‘filename.json’) – чтение из json  
и др.

df.to\_csv('filename.csv') – запись, аналогично для всех остальных типов

# Анализ

Группировка данных один из самых часто используемых методов при анализе данных. В pandas за группировку отвечает метод .groupby.

titanic\_df = pd.read\_csv('titanic.csv') – данные по пассажирам Титаника

titanic\_df.groupby(['Sex', 'Survived'])['PassengerID'].count() – сколько женщин и мужчин выжило  
titanic\_df.groupby(['PClass', 'Survived'])['PassengerID'].count() – сколько бедняков и богачей выжило

pvt = titanic\_df.pivot\_table(index=['Sex'], columns=['PClass'], values='Name', aggfunc='count') – сводная таблица, где сводным элементом будет количество пассажиров

# Визуализация

Для визуального анализа данных, pandas использует библиотеку matplotlib.

import matplotlib.pyplot as plt - библиотека  
new\_sample\_df = df.loc['2012-Feb':'2017-Feb', ['Close']] – цены закрытия с 2012 по 2017  
new\_sample\_df.plot()  
plt.show()