Основы pandas

In [1]:

```
import pandas as pd
```

Пример забора данных с сайтов

In [ ]:

```
import pymysql

connection
```

In [ ]:

```
pd.read_sql('select * from db.table;', connection)
```

In [2]:

```
pd.read_html('http://www.cbr.ru')[0]
```

Out[2]:

	Курсы валют	с 14.08.2020	с 15.08.2020
0	Доллар США	73,6067₽	73,2157₽
1	Евро	87,0399₽	86,4092₽

In [3]:

```
page_url = 'https://www.finanz.ru/valyuty/usd-rub'

# Импортируем нужную нам страницу в df
# attrs = {'class': 'news_table'} ---> указываем какой именно блок нам нужен
# encoding='utf-8' ---> указываем кодировку страниц для корректного отображения кириллицы
df = pd.read_html(page_url, attrs = {'class': 'news_table'}, encoding='utf-8')
```

In [4]:

```
df[:5]
```

Out[4]:

```
[       Дата  Закрытие  Открытие  Максимум  Минимум
 0  14.08.20    728225    729947    734855   727920
 1  13.08.20    729945    738035    738536   728538
 2  12.08.20    738036    731569    738985   730005
 3  11.08.20    731568    735707    736545   726885
 4  10.08.20    735445    734335    740150   731835
 5  09.08.20    734770    737165    738935   734480
 6  07.08.20    737500    733024    739250   731680
 7  06.08.20    733785    729100    736140   726695
 8  05.08.20    729090    733841    735646   726350,
        Имя  Unnamed: 1     %    курс                 Дата
 0  EUR/RUB         NaN   255   86267  15.08.2020 00:32:00
 1  USD/RUB         NaN -2356  728225  15.08.2020 00:41:00
 2  CHF/RUB         NaN  -672   80115  15.08.2020 06:11:00
 3  JPY/RUB         NaN  -245    6836  15.08.2020 06:11:00
 4  GBP/RUB         NaN    -1  952919  15.08.2020 06:11:00
 5  NOK/RUB         NaN  -724   81908  15.08.2020 06:11:00
 6  AUD/RUB         NaN  -949  522283  15.08.2020 06:11:00
 7  NZD/RUB         NaN  -565  476376  15.08.2020 06:11:00
 8  TRY/RUB         NaN  -809    9882  15.08.2020 06:11:00]
```

Импорт данных из файла

In [ ]:

```
csv - comma
```

In [5]:

```
data = pd.read_csv('power.csv')
data.head()
```

Out[5]:

	country	year	quantity	category
0	Austria	1996	5.0	1
1	Austria	1995	17.0	1
2	Belgium	2014	0.0	1
3	Belgium	2013	0.0	1
4	Belgium	2012	35.0	1

In [6]:

```
data.head(30)
```

Out[6]:

	country	year	quantity	category
0	Austria	1996	5.0	1
1	Austria	1995	17.0	1
2	Belgium	2014	0.0	1
3	Belgium	2013	0.0	1
4	Belgium	2012	35.0	1
5	Belgium	2011	25.0	1
6	Belgium	2010	22.0	1
7	Belgium	2009	45.0	1
8	Czechia	1998	1.0	1
9	Czechia	1995	7.0	1
10	Finland	2010	9.0	1
11	Finland	2009	13.0	1
12	Finland	2008	39.0	1
13	Finland	2007	21.0	1
14	Finland	2006	0.0	1
15	Finland	2005	0.0	1
16	Finland	2004	0.0	1
17	Finland	2003	0.0	1
18	Finland	2002	0.0	1
19	Finland	2001	0.0	1
20	Finland	2000	0.0	1
21	Finland	1999	0.0	1
22	Finland	1998	0.0	1
23	Finland	1997	0.0	1
24	Finland	1996	0.0	1
25	Finland	1995	3.0	1
26	France	2014	119.0	1
27	France	2013	102.0	1
28	France	2012	62.0	1
29	France	2011	55.0	1

In [7]:

```
data.tail(10)
```

Out[7]:

	country	year	quantity	category
1189472	Vanuatu	2010	5.17	71
1189473	Vanuatu	2009	5.63	71
1189474	Vanuatu	2008	5.63	71
1189475	Viet Nam	2014	300.00	71
1189476	Viet Nam	2013	92.00	71
1189477	Viet Nam	2012	92.00	71
1189478	Viet Nam	2011	87.00	71
1189479	Viet Nam	2010	50.00	71
1189480	Viet Nam	2009	10.00	71
1189481	Viet Nam	2008	1.00	71

In [8]:

```
type(data)
```

Out[8]:

```
pandas.core.frame.DataFrame
```

In [ ]:

```
data.
```

In [ ]:

```
# если надо указать свои заголовки и разделитель
# data = pd.read_csv('power.csv', names = ['страна', 'год', 'количество', 'категория'], sep
# data.head()
```

In [9]:

```
?pd.read_csv
```

In [10]:

```
# количество строк в датафрейме

len(data)
```

Out[10]:

```
1189482
```

In [11]:

```
# или так

data.shape
```

Out[11]:

```
(1189482, 4)
```

Простые вычисления для нового столбца

In [12]:

```
data['year_plus_one'] = data['year'] + 1
data.head()
```

Out[12]:

	country	year	quantity	category	year_plus_one
0	Austria	1996	5.0	1	1997
1	Austria	1995	17.0	1	1996
2	Belgium	2014	0.0	1	2015
3	Belgium	2013	0.0	1	2014
4	Belgium	2012	35.0	1	2013

Основные сведения о датафрейме

In [13]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1189482 entries, 0 to 1189481
Data columns (total 5 columns):
country          1189482 non-null object
year             1189482 non-null int64
quantity         1189482 non-null float64
category         1189482 non-null int64
year_plus_one    1189482 non-null int64
dtypes: float64(1), int64(3), object(1)
memory usage: 45.4+ MB
```

In [14]:

```
2**64
```

Out[14]:

```
18446744073709551616
```

In [ ]:

```
None
```

In [15]:

```
# немного статистики

data.describe()
```

Out[15]:

	year	quantity	category	year_plus_one
count	1.189482e+06	1.189482e+06	1.189482e+06	1.189482e+06
mean	2.002852e+03	1.842648e+05	3.624763e+01	2.003852e+03
std	7.167345e+00	1.585663e+07	1.809968e+01	7.167345e+00
min	1.990000e+03	-8.643480e+05	1.000000e+00	1.991000e+03
25%	1.997000e+03	1.400000e+01	2.400000e+01	1.998000e+03
50%	2.003000e+03	1.890000e+02	3.500000e+01	2.004000e+03
75%	2.009000e+03	2.265000e+03	5.100000e+01	2.010000e+03
max	2.014000e+03	6.680329e+09	7.100000e+01	2.015000e+03

Отдельный столбец (тип Series)

In [16]:

```
data['year plus one'].head()
```

Out[16]:

```
0    1996
1    1995
2    2014
3    2013
4    2012
Name: year, dtype: int64
```

In [17]:

```
# или так

data.year.head()
```

Out[17]:

```
0    1996
1    1995
2    2014
3    2013
4    2012
Name: year, dtype: int64
```

In [18]:

```
type(data['year'])
```

Out[18]:

```
pandas.core.series.Series
```

In [19]:

```
data.head()
```

Out[19]:

	country	year	quantity	category	year_plus_one
0	Austria	1996	5.0	1	1997
1	Austria	1995	17.0	1	1996
2	Belgium	2014	0.0	1	2015
3	Belgium	2013	0.0	1	2014
4	Belgium	2012	35.0	1	2013

In [20]:

```
# уникальные значения в столбце

data['category'].unique()
```

Out[20]:

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 13, 12, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
       35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51,
       52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68,
       69, 70, 71])
```

In [22]:

```
set([1, 1, 1, 2])
```

Out[22]:

```
{1, 2}
```

In [23]:

```
len(data['category'].unique())
```

Out[23]:

```
71
```

In [24]:

```
data['category'].nunique()
```

Out[24]:

```
71
```

In [ ]:

```
data['category'].head()
```

In [21]:

```
# распределение количества строк по значениям столбца

data['category'].value_counts().head(10)
```

Out[21]:

```
67    133916
27     97645
24     75132
42     64161
37     62156
39     53198
25     52032
21     50229
51     43466
31     42307
Name: category, dtype: int64
```

In [25]:

```
data['category'].value_counts(normalize=True).head()
```

Out[25]:

```
67    0.112583
27    0.082090
24    0.063164
42    0.053940
37    0.052255
Name: category, dtype: float64
```

Фильтры

In [26]:

```
data = pd.read_csv('power.csv')
data.head()
```

Out[26]:

	country	year	quantity	category
0	Austria	1996	5.0	1
1	Austria	1995	17.0	1
2	Belgium	2014	0.0	1
3	Belgium	2013	0.0	1
4	Belgium	2012	35.0	1

In [27]:

```
# выбрать несколько столбцов

country_stats = data.filter(items = ['country', 'quantity'])
country_stats.head()
```

Out[27]:

	country	quantity
0	Austria	5.0
1	Austria	17.0
2	Belgium	0.0
3	Belgium	0.0
4	Belgium	35.0

In [ ]:

```
data['country']
```

In [28]:

```
# или так

data[['country', 'quantity']].head()
```

Out[28]:

	country	quantity
0	Austria	5.0
1	Austria	17.0
2	Belgium	0.0
3	Belgium	0.0
4	Belgium	35.0

Отфильтруем строки с потреблением выше среднего

In [29]:

```
average_level = data['quantity'].mean()
average_level
```

Out[29]:

```
184264.77005012965
```

In [30]:

```
'quantity > {}'.format(average_level)
```

Out[30]:

```
'quantity > 184264.77005012965'
```

In [31]:

```
f'quantity > {average_level}'
```

Out[31]:

```
'quantity > 184264.77005012965'
```

In [32]:

```
# строки с потреблением больше среднего

average_level = data['quantity'].mean()
country_stats.query('quantity > {}'.format(average_level)).head()
```

Out[32]:

	country	quantity
3228	United States	367987.0
3229	United States	384439.0
3230	United States	370625.0
3231	United States	310909.0
3232	United States	335418.0

In [ ]:

```
if data.quantity > average_level:
```

In [33]:

```
# самый популярный способ

data[ data.quantity > average_level ].head()
```

Out[33]:

	country	year	quantity	category
3228	United States	2014	367987.0	2
3229	United States	2013	384439.0	2
3230	United States	2012	370625.0	2
3231	United States	2011	310909.0	2
3232	United States	2010	335418.0	2

Как определить используемый вариант названия страны?

In [34]:

```
data['country'].unique()
```

Out[34]:

```
array(['Austria', 'Belgium', 'Czechia', 'Finland', 'France', 'Greece',
       'Hungary', 'Italy', 'Korea, Republic of', 'Netherlands', 'Romania',
       'Serbia', 'Slovakia', 'Ukraine', 'United Kingdom', 'United States',
       'Brunei Darussalam', 'Bulgaria', 'Canada', 'Chile', 'Croatia',
       'Iran (Islamic Rep. of)', 'Jordan', 'Lithuania', 'Mexico', 'Oman',
       'Other Asia', 'Poland', 'Portugal', 'Spain', 'Sweden',
       'Switzerland', 'T.F.Yug.Rep. Macedonia', 'Turkey', 'Uzbekistan',
       'Argentina', 'Colombia', 'Germany', 'Norway', 'Australia',
       'China, Macao SAR', 'Japan', 'Denmark', 'Ireland', 'Philippines',
       'Bangladesh', 'Eritrea', 'Ethiopia', 'Lesotho', 'Nepal', 'Niger',
       'Pakistan', 'Serbia and Montenegro', 'Tunisia', 'Botswana',
       'Georgia', 'Republic of Moldova', 'Peru', 'South Africa',
       'Iceland', 'Latvia', 'Luxembourg', 'Cuba', 'Kyrgyzstan',
       'Singapore', 'Thailand', 'United Arab Emirates', 'Uruguay',
       'Armenia', 'Fiji', "Korea, Dem.Ppl's.Rep.", 'Russian Federation',
       'Swaziland', 'Zimbabwe', 'Afghanistan', 'Antigua and Barbuda',
       'Azerbaijan', 'Bahamas', 'Belize', 'Bolivia (Plur. State of)',
       'Brazil', 'Burkina Faso', 'Cameroon', 'Cayman Islands',
       'Central African Rep.', 'China', 'Comoros', 'Congo', 'Costa Rica',
       "Côte d'Ivoire", 'Dem. Rep. of the Congo', 'Dominican Republic',
       'Ecuador', 'Egypt', 'El Salvador', 'Estonia',
       'Ethiopia, incl. Eritrea', 'Faeroe Islands', 'French Guiana',
       'Germany, Fed. R. (former)', 'Greenland', 'Guatemala', 'Guinea',
       'Guyana', 'Honduras', 'Indonesia', 'Israel', 'Jamaica', 'Jersey',
       'Kazakhstan', 'Kenya', 'Kuwait', "Lao People's Dem. Rep.",
       'Liberia', 'Libya', 'Malawi', 'Malaysia', 'Maldives', 'Mauritania',
       'Mayotte', 'Morocco', 'Myanmar', 'Namibia', 'New Caledonia',
       'New Zealand', 'Nicaragua', 'Niue', 'Panama', 'Papua New Guinea',
       'Paraguay', 'Sao Tome and Principe', 'Senegal', 'Seychelles',
       'Slovenia', 'Solomon Islands', 'Sri Lanka', 'Sudan (former)',
       'Suriname', 'Togo', 'Tonga', 'Trinidad and Tobago',
       'United Rep. of Tanzania', 'Venezuela (Bolivar. Rep.)',
       'Yugoslavia, SFR (former)', 'Zambia', 'Algeria', 'Curaçao',
       'Gabon', 'Malta', 'Neth. Antilles (former)', 'Sierra Leone',
       'USSR (former)', 'Barbados', 'Benin', 'Chad', 'Djibouti',
       'French Polynesia', 'Ghana', 'Grenada', 'Guadeloupe', 'Haiti',
       'Iraq', 'Kiribati', 'Madagascar', 'Mali', 'Martinique',
       'Pacific Islands (former)', 'Palau', 'St. Helena and Depend.',
       'St. Vincent-Grenadines', 'Syrian Arab Republic', 'Uganda',
       'Yemen', 'Yemen, Dem. (former)', 'Burundi', 'India', 'Mauritius',
       'Mozambique', 'Nigeria', 'Réunion', 'Rwanda', 'Samoa', 'Somalia',
       'St. Kitts-Nevis', 'Viet Nam', 'Belarus', 'Cyprus', 'Cabo Verde',
       'Albania', 'Bahrain', 'Bosnia and Herzegovina',
       'China, Hong Kong SAR', 'Czechoslovakia (former)', 'Angola',
       'Bermuda', 'Bhutan', 'Equatorial Guinea', 'Lebanon',
       'Saudi Arabia', 'St. Lucia', 'St. Pierre-Miquelon',
       'State of Palestine', 'Tajikistan', 'German Dem. R. (former)',
       'Anguilla', 'Mongolia', 'Montenegro', 'Cambodia', 'South Sudan',
       'Sudan', 'Andorra', 'Aruba', 'Bonaire, St Eustatius, Saba',
       'British Virgin Islands', 'Dominica', 'Falkland Is. (Malvinas)',
       'Gambia', 'Gibraltar', 'Qatar', 'Sint Maarten (Dutch part)',
       'Turkmenistan', 'Turks and Caicos Islands', 'Vanuatu',
       'Guinea-Bissau', 'Micronesia (Fed. States of)', 'Timor-Leste',
       'Yemen Arab Rep. (former)', 'American Samoa', 'Liechtenstein',
       'Puerto Rico', 'Cook Islands', 'Guam', 'Guernsey', 'Isle of Man',
```

```
       'Marshall Islands', 'Montserrat', 'Nauru',
       'Northern Mariana Islands', 'Tuvalu', 'United States Virgin Is.',
       'Wallis and Futuna Is.',
       'Commonwealth of Independent States (CIS)', 'Antarctic Fisheries'],
      dtype=object)
```

In [ ]:

```
if 'rus' in 'Russia Federation':
```

In [35]:

```
# найдем как называется Россия и Беларусь в этом датафрейме
# фильтр на подстроку - смотрим все страны, содержащие в названии 'us'

data[ data['country'].str.contains('us', case=False) ]['country'].unique()
```

Out[35]:

```
array(['Austria', 'Brunei Darussalam', 'Australia', 'Russian Federation',
       'USSR (former)', 'Mauritius', 'Belarus', 'Cyprus',
       'Bonaire, St Eustatius, Saba'], dtype=object)
```

In [36]:

```
# фильтр на несколько условий сразу
# | - условие ИЛИ
# & AND
# () | (() | () & ())
filtered_countries = data[ (data['country']=='Russian Federation') | (data['country']=='Bel

filtered_countries.head()
```

Out[36]:

	country	year	quantity	category
6940	Russian Federation	2014	12714.0	3
6941	Russian Federation	2013	11285.0	3
6942	Russian Federation	2012	11302.0	3
6943	Russian Federation	2011	7611.0	3
6944	Russian Federation	2010	9263.0	3

In [37]:

```
filtered_countries['country'].unique()
```

Out[37]:

```
array(['Russian Federation', 'Belarus'], dtype=object)
```

In [38]:

```
# фильтры на номер строки

data.loc[1000:1005]
```

Out[38]:

	country	year	quantity	category
1000	Austria	1998	-14.0	1
1001	Austria	1997	5.0	1
1002	Austria	1996	-1.0	1
1003	Austria	1995	-10.0	1
1004	Austria	1994	-8.0	1
1005	Austria	1993	-11.0	1

Сортировка

In [39]:

```
# Сортировка по столбцу

data.sort_values(by='quantity').head()
```

Out[39]:

	country	year	quantity	category
832375	United States	2000	-864348.0	42
832373	United States	2002	-562414.0	42
832362	United States	2013	-551490.0	42
832380	United States	1995	-493380.0	42
832322	Ukraine	2010	-477263.0	42

In [40]:

```
# сортировка по убыванию

data.sort_values('quantity', ascending=False).head()
```

Out[40]:

	country	year	quantity	category
492581	United States	2014	6.680329e+09	31
492267	United States	2014	6.680329e+09	31
492047	China	2014	5.462672e+09	31
492345	China	2014	5.462672e+09	31
122392	USSR (former)	1990	3.257000e+09	12

In [41]:

```
# сортировка по нескольким столбцам

data.sort_values(by=['year', 'country', 'quantity'], ascending=[False, True, False]).head(5
```

Out[41]:

	country	year	quantity	category
835127	Afghanistan	2014	1.935442e+06	42
491541	Afghanistan	2014	6.600000e+04	31
492004	Afghanistan	2014	6.600000e+04	31
492292	Afghanistan	2014	6.600000e+04	31
791301	Afghanistan	2014	5.537080e+03	42
797879	Afghanistan	2014	5.537080e+03	42
811493	Afghanistan	2014	5.537080e+03	42
813883	Afghanistan	2014	5.537080e+03	42
820564	Afghanistan	2014	5.537080e+03	42
828390	Afghanistan	2014	5.537080e+03	42
832393	Afghanistan	2014	5.537080e+03	42
1097271	Afghanistan	2014	4.486800e+03	67
1091772	Afghanistan	2014	3.767300e+03	67
1108326	Afghanistan	2014	3.710800e+03	67
1061021	Afghanistan	2014	2.967300e+03	67
1045322	Afghanistan	2014	2.204900e+03	67
364480	Afghanistan	2014	1.758409e+03	25
369505	Afghanistan	2014	1.758396e+03	25
518535	Afghanistan	2014	1.517400e+03	31
521820	Afghanistan	2014	1.517400e+03	31
496034	Afghanistan	2014	1.388700e+03	31
502008	Afghanistan	2014	1.388700e+03	31
509843	Afghanistan	2014	1.388700e+03	31
512335	Afghanistan	2014	1.388700e+03	31
1102803	Afghanistan	2014	1.049300e+03	67
334666	Afghanistan	2014	1.002966e+03	25
341782	Afghanistan	2014	1.002966e+03	25
353105	Afghanistan	2014	1.002966e+03	25
357841	Afghanistan	2014	1.002966e+03	25
563371	Afghanistan	2014	8.953000e+02	33
407563	Afghanistan	2014	8.821000e+02	27
419783	Afghanistan	2014	8.821000e+02	27
432306	Afghanistan	2014	8.821000e+02	27
437547	Afghanistan	2014	8.821000e+02	27

	country	year	quantity	category
442787	Afghanistan	2014	8.821000e+02	27
462537	Afghanistan	2014	8.821000e+02	27
1053124	Afghanistan	2014	8.000000e+02	67
1066512	Afghanistan	2014	8.000000e+02	67
1116171	Afghanistan	2014	7.760000e+02	67
374415	Afghanistan	2014	7.554300e+02	25
378276	Afghanistan	2014	7.554300e+02	25
1137168	Afghanistan	2014	7.423000e+02	67
713410	Afghanistan	2014	7.130000e+02	39
719887	Afghanistan	2014	7.130000e+02	39
727865	Afghanistan	2014	7.130000e+02	39
733131	Afghanistan	2014	7.130000e+02	39
738397	Afghanistan	2014	7.130000e+02	39
753073	Afghanistan	2014	7.130000e+02	39
1161790	Afghanistan	2014	5.933000e+02	67
1111003	Afghanistan	2014	5.000000e+02	67

Параметр inplace

In [47]:

```
data.head()
```

Out[47]:

	country	year	quantity	category
12121	Zimbabwe	1990	4.0	4
14259	Zimbabwe	1990	4.0	4
17051	Zimbabwe	1990	4.0	4
19201	Zimbabwe	1990	4.0	4
21685	Zimbabwe	1990	4.0	4

In [46]:

```
data = data.sort_values(['country', 'year'], ascending=[False, True])
```

In [ ]:

```
def calculation(data, inplace=True):
    data.sort_values(['country', 'year'], ascending=[False, True], inplace=inplace)
```

In [48]:

```
data.sort_values(['country', 'year'], ascending=[False, True], inplace=True)
```

Out[48]:

	country	year	quantity	category
12121	Zimbabwe	1990	4.0	4
14259	Zimbabwe	1990	4.0	4
17051	Zimbabwe	1990	4.0	4
19201	Zimbabwe	1990	4.0	4
21685	Zimbabwe	1990	4.0	4
...	...	...	...	...
1161790	Afghanistan	2014	593.3	67
1179197	Afghanistan	2014	3.0	70
1180815	Afghanistan	2014	3.0	70
1182030	Afghanistan	2014	3.0	70
1186132	Afghanistan	2014	3.0	70

1189482 rows × 4 columns

In [ ]:

```
data = data.sort_values(by=['country', 'year', 'quantity'], ascending=[True, True, False])

# чтобы сократить это выражение используем inplace:
data.sort_values(by=['country', 'year', 'quantity'], ascending=[True, True, False], inplace
```