Департамент образования и науки города Москвы
Государственное автономное образовательное учреждение высшего образования города Москвы
«Московский городской педагогический университет»
Институт цифрового образования
Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Инструменты для хранения и обработки больших данных

Лабораторная работа 5.1

Развертывание и настройка кластера Hadoop

Выполнил(а): Ванярина Ю.А., группа: АДЭУ-211

Преподаватель: Босенко Т.М.

Москва

2024

Цель: ознакомление с процессом установки и настройки распределенных систем, таких как Apache(Arenadata) Hadoop. Изучить основные операции и функциональные возможности системы, что позволит понять принципы работы с данными и распределенными вычислениями. Необходимое ПО:

- Ubuntu 24.04 LTS (22.04, 20.04) или новее.
- Java 8 ил Java11 или новее.
- Apache Spark 3.4.3.
- Python 3.12+.
- pip (менеджер пакетов Python).

Вариант 16.

Задачи: Установка Apache Hadoop.

Данные: Исторические данные по акциям Системы (AFKS) с сайта Московской биржи (moex.com)
Операции: Фильтрация данных за 2019 год, расчет средней цены закрытия, группировка по месяцам.

Для начала надо войти пользователем hadoop (рисунок 1).

```
devops@devopsvm:~$ sudo su hadoop
[sudo] password for devops:
```

Рисунок 1. Вход пользователем hadoop.

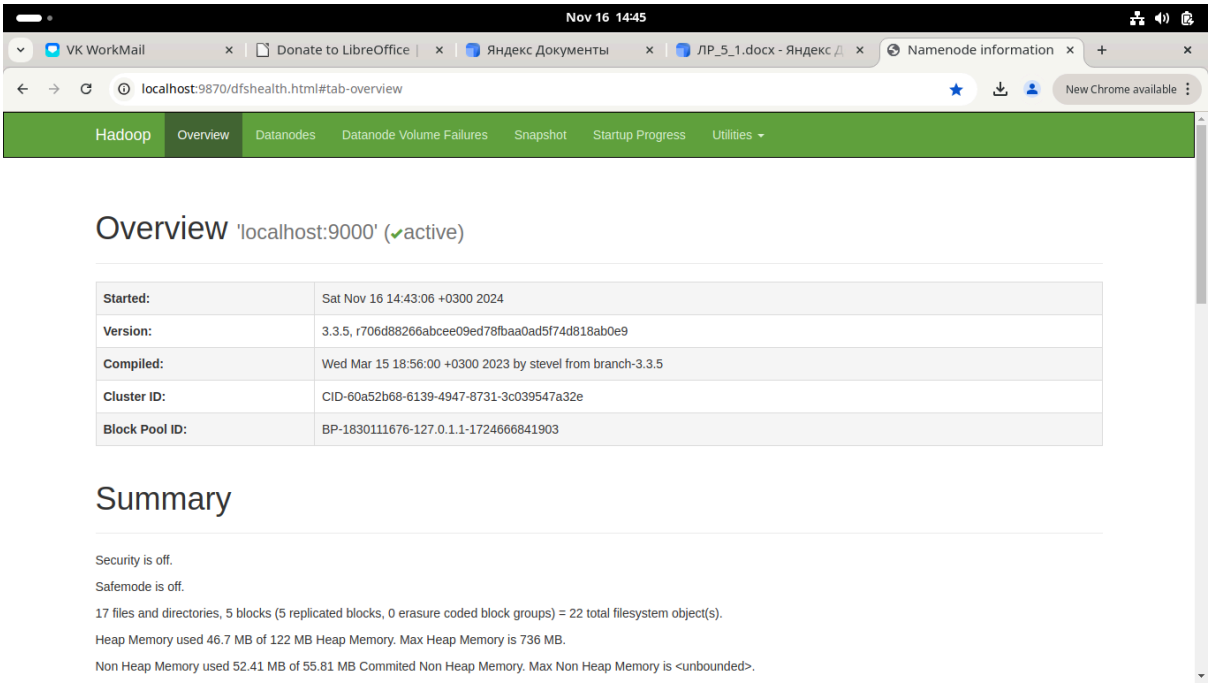
Затем на рисунках 2-3 происходит запуск Hadoop.

```
hadoop@devopsvm:~$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [devopsvm]
2024-10-25 11:20:11,515 WARN util.NativeCodeLoader: Unable to load native-hadoop
 library for your platform... using builtin-java classes where applicable
hadoop@devopsvm:~$
```

Рисунок 2. Запуск Hadoop

```
hadoop@devopsvm:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hadoop@devopsvm:~$
```

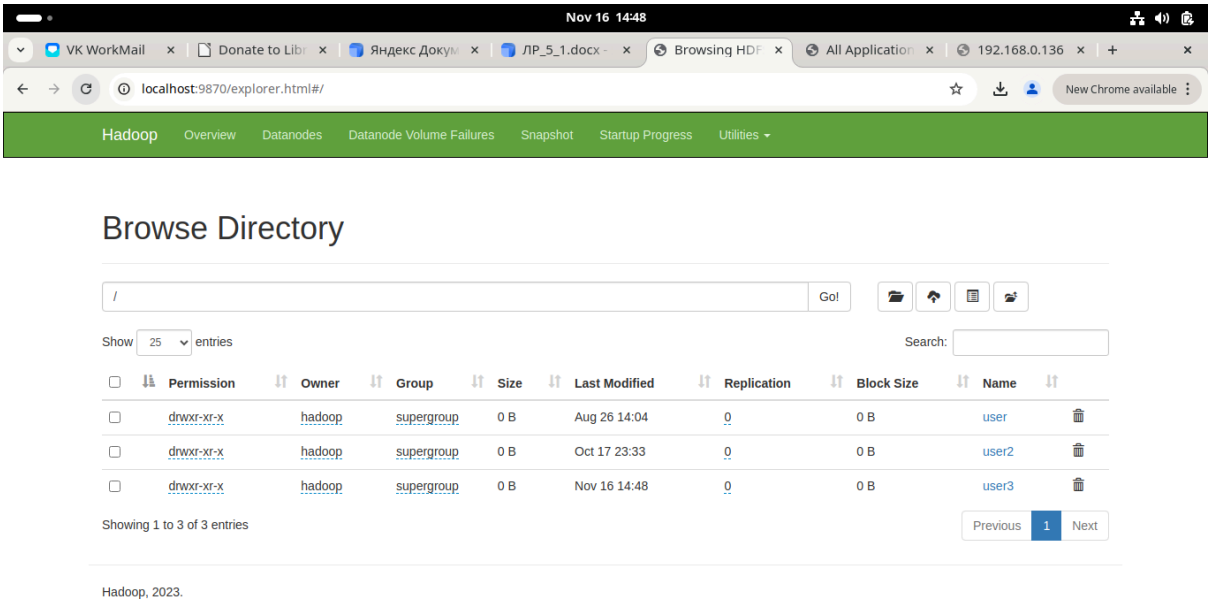
Рисунок 3. Запуск Hadoop



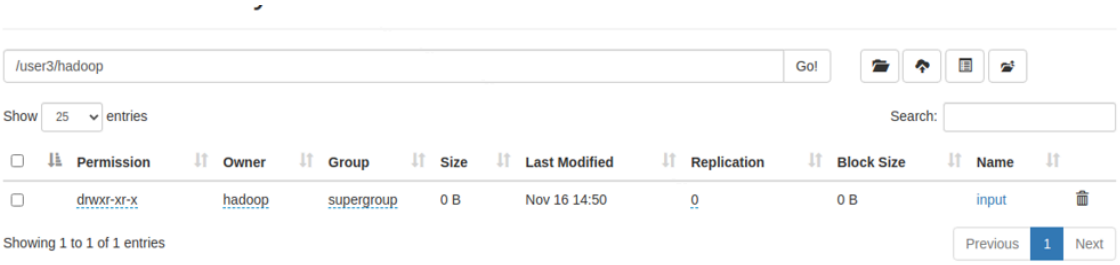
Успешная загрузка hadoop

Директории а хадуп после создания user3 с помощью hdfs dfs -mkdir /user3

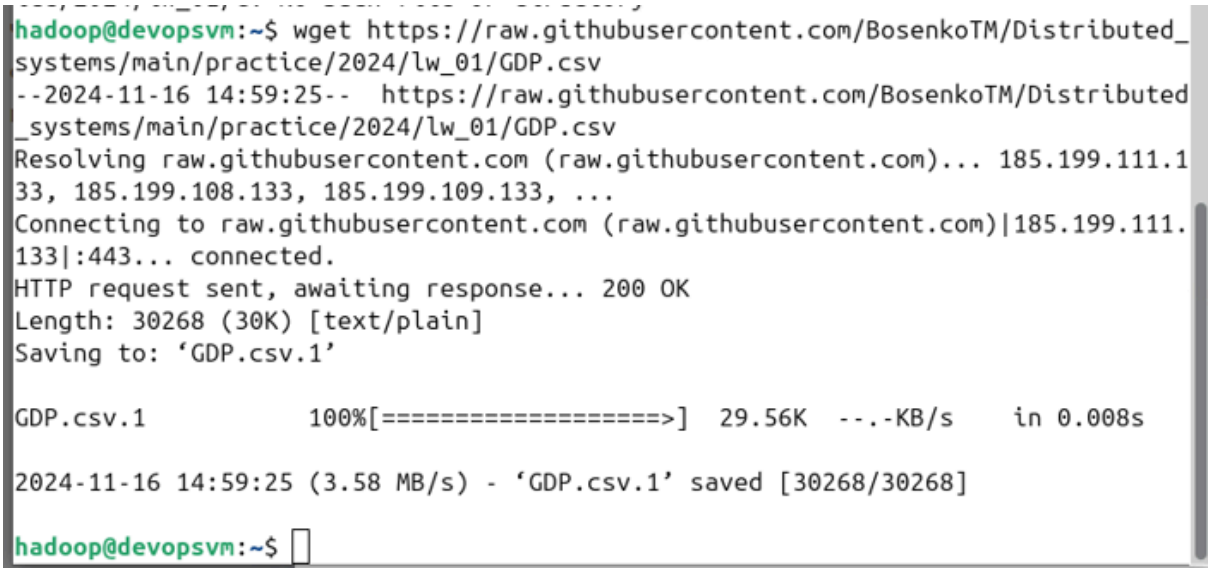
```
hadoop@devopsvm:~$ hdfs dfs -mkdir /user3
2024-11-16 14:48:39,008 WARN util.NativeCodeLoader: Unable to load native-hadoop
 library for your platform... using builtin-java classes where applicable
hadoop@devopsvm:~$
```



Далее создали путь /user3/hadoop/input



Загрузили файл по ссылке

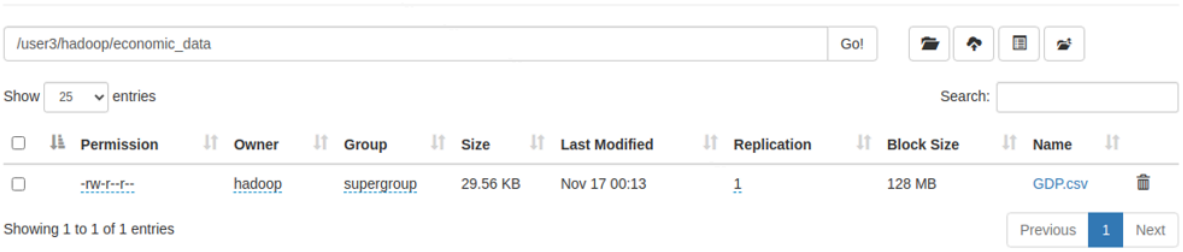


Добавляем файл в нужный нам каталог с помощью hdfs dfs -put

```
hadoop@devopsvm:~$ hdfs dfs -mkdir /user/hadoop/economic_data
2024-11-16 15:00:14,252 WARN util.NativeCodeLoader: Unable to load native-hadoop
 library for your platform... using builtin-java classes where applicable
mkdir: `/user/hadoop/economic_data': File exists
hadoop@devopsvm:~$ hdfs dfs -put GDP.csv /user/hadoop/economic_data/
2024-11-16 15:00:35,026 WARN util.NativeCodeLoader: Unable to load native-hadoop
 library for your platform... using builtin-java classes where applicable
put: `/user/hadoop/economic_data/GDP.csv': File exists
hadoop@devopsvm:~$
```

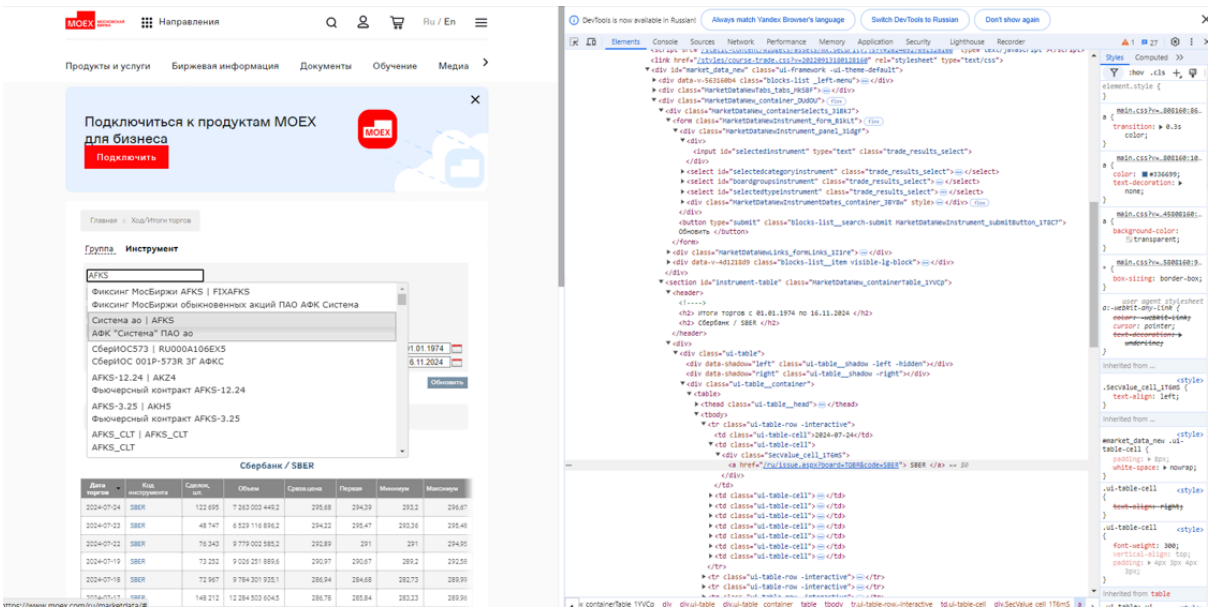
```
hadoop@devopsvm:~$ hdfs dfs -chmod 777 /user/hadoop/economic_data
2024-11-16 15:02:11,934 WARN util.NativeCodeLoader: Unable to load native-hadoop
 library for your platform... using builtin-java classes where applicable
```

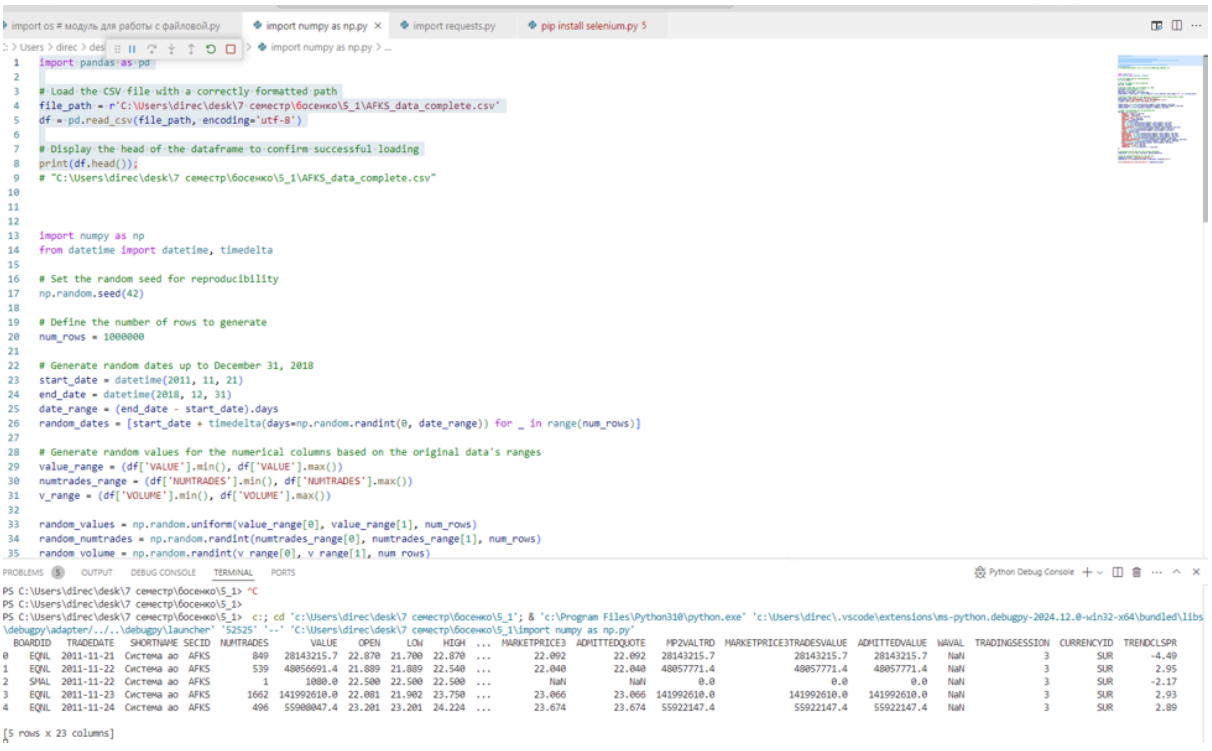
Файл успешно загружен



Индивидуальное задание

По заданию были скачаны данные по акциям AFKS, но данных было всего 2989 строк,поэтому были сгенерированы синтетические данные до 1000000 строк





Так как файл превышал 25 мб,которые могут быть загружены в github , был использован яндекс диск для скачивания файла на виртуальной машине. В этой части работы возникли сложности с загрузкой файла в нужный каталог. Был выбран путь переноса файла из каталога devops (где находился скачанный файл) в каталог hadoop(где находился успешно загруженный файл примера) с помощью команды

sudo mv

```
hadoop@devopsvm:~$ sudo mv /home/devops/Vanyarina_afks.csv /home/hadoop
hadoop@devopsvm:~$ ls
 a_1.csv     GDP.csv.1            output                        Vanyarina_afks.csv
 afks.csv    hadoop-3.3.5.tar.gz  snap                         'view?usp=sharing%2FAKFS_data_extended.csv'
 GDP.csv     hdfs                 spark-3.4.3-bin-hadoop3.tgz
```

далее была использована команда put для переноса файла в нужный каталог

```
hadoop@devopsvm:~$ hdfs dfs -put Vanyarina_afks.csv /user3/hadoop/Vanyarina/
2024-11-21 02:18:14,087 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform..
sing builtin-java classes where applicable
hadoop@devopsvm:~$ hdfs dfs -chmod 777 /user/hadoop/Vanyarina/
2024-11-21 02:19:29,544 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform..
sing builtin-java classes where applicable
hadoop@devopsvm:~$ hdfs dfs -chmod 777 /user/hadoop/Vanyarina/Vanyarina_afks.csv
2024-11-21 02:19:41,910 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform..
sing builtin-java classes where applicable
```

Далее работа проходила в jupyter notebook

было установлено подключение и далее прочитан файл Vanyarina_afks.csv

[ ]:
```
Файл содержит данные о торговой активности
- boardid: идентификатор биржевой доски;
- tradedate: дата торгов;
- secid код ценной бумаги;
- numtrades: количество сделок;
- value: стоимость сделок;
- OPEN, low, high, CLOSE: открытие, минимум, максимум и закрытие цены;
- tradingsession: торговая сессия;
- currencyid: валюта.

Vanyarina
```

[5]:
```
# Чтение данных из HDFS
file_path = "hdfs://localhost:9000/user3/hadoop/Vanyarina/Vanyarina_afks.csv"
df = spark.read.csv(file_path, header=True, inferSchema=True)

# Просмотр первых строк данных
df.show(5)
#Vanyarina
```

```
+-------+----------+-----+---------+---------------+---------------+---------------+---------------+---------------+--------------+----------+
|boardid| tradedate|secid|numtrades|          value|           OPEN|            low|           high|          CLOSE|tradingsession|currencyid|
+-------+----------+-----+---------+---------------+---------------+---------------+---------------+---------------+--------------+----------+
|   EQNL|2016-12-24| AFKS|    81325|  2.192880896E9|  9.962095616E9|1.2048973824E10|  5.867784192E9|1.1757453312E10|             3|       SUR|
|   EQNL|2014-08-25| AFKS|    13567|1.0338044928E10|1.2289158144E10| 1.311008256E10|1.2980217856E10|  7.836886016E9|             3|       SUR|
|   EQNL|2018-07-14| AFKS|    31640|1.2584542208E10| 1.137401856E10|1.3092831232E10|  1.287624704E9|  9.700277248E9|             3|       SUR|
|   EQNL|2013-01-03| AFKS|     3743|  8.577597952E9|   9.73241664E8|  4.239877632E9|  5.714208768E9|1.1269677056E10|             3|       SUR|
|   EQNL|2017-01-18| AFKS|   149951|   1.12538816E9|  5.861766656E9|   1.49738944E9|  7.867397632E9|   1.64064896E9|             3|       SUR|
+-------+----------+-----+---------+---------------+---------------+---------------+---------------+---------------+--------------+----------+
only showing top 5 rows
```

```
[5]: pandas_df = df.toPandas()
     pandas_df.head()
     #Vanyarina
```

```
24/11/22 01:22:04 WARN NettyRpcEnv: Ignored failure: java.util.concurrent.TimeoutException: Cannot receive any reply from 192.168.1.69:43585 in 10000 millis
```

[5]:

	boardid	tradedate	secid	numtrades	value	OPEN	low	high	CLOSE	tradingsession	currencyid
0	EQNL	2016-12-24	AFKS	81325	2.192881e+09	9.962096e+09	1.204897e+10	5.867784e+09	1.175745e+10	3	SUR
1	EQNL	2014-08-25	AFKS	13567	1.033804e+10	1.228916e+10	1.311008e+10	1.298022e+10	7.836886e+09	3	SUR
2	EQNL	2018-07-14	AFKS	31640	1.258454e+10	1.137402e+10	1.309283e+10	1.287625e+09	9.700277e+09	3	SUR
3	EQNL	2013-01-03	AFKS	3743	8.577598e+09	9.732417e+08	4.239878e+09	5.714209e+09	1.126968e+10	3	SUR
4	EQNL	2017-01-18	AFKS	149951	1.125388e+09	5.861767e+09	1.497389e+09	7.867398e+09	1.640649e+09	3	SUR

проведен статистический анализ с помощью describe и проверены нулевые значения

```
[6]: summary_statistics = pandas_df.describe()
     summary_statistics
```

[6]:

	numtrades	value	OPEN	low	high	CLOSE	tradingsession
count	1.006360e+06	1.006360e+06	1.005832e+06	1.005832e+06	1.005832e+06	1.005832e+06	1006360.0
mean	9.233557e+04	6.528773e+09	6.534148e+09	6.537380e+09	6.530233e+09	6.533761e+09	3.0
std	5.394805e+04	3.815805e+09	3.814111e+09	3.816266e+09	3.813887e+09	3.817120e+09	0.0
min	0.000000e+00	0.000000e+00	5.400000e+00	5.020000e+00	5.400000e+00	5.400000e+00	3.0
25%	4.551575e+04	3.217932e+09	3.226179e+09	3.234862e+09	3.226262e+09	3.227875e+09	3.0
50%	9.227700e+04	6.528546e+09	6.534653e+09	6.539789e+09	6.532886e+09	6.531892e+09	3.0
75%	1.390490e+05	9.832759e+09	9.834727e+09	9.841038e+09	9.830817e+09	9.844526e+09	3.0
max	1.859000e+05	1.314302e+10	1.314301e+10	1.314301e+10	1.314300e+10	1.314301e+10	3.0

```
[12]: print(pandas_df.dtypes)
```

```
boardid            object
tradedate          object
secid              object
numtrades           int32
value             float64
OPEN              float64
low               float64
high              float64
CLOSE             float64
tradingsession      int32
currencyid         object
dtype: object
```

```
[23]: print(pandas_df.isna().sum())
```

```
boardid           0
tradedate         0
secid             0
numtrades         0
value             0
OPEN              0
low               0
high              0
CLOSE             0
tradingsession    0
currencyid        0
dtype: int64
```

Тип данных в tradedate изменен с объекта на дату и далее выполнено первое задание по фильтрации датасета с данными за 2019 год

```
[26]: pandas_df['tradedate'] = pd.to_datetime(pandas_df['tradedate'])
```

```
[27]: data_2019 = pandas_df[pandas_df['tradedate'].dt.year == 2019]
data_2019.head()
```

[27]:

	boardid	tradedate	secid	numtrades	value	OPEN	low	high	CLOSE	tradingsession	currencyid
373043	EQDP	2019-01-03	AFKS	0	0.0	6.534653e+09	6.539789e+09	6.532886e+09	6.531892e+09	3	SUR
373044	SMAL	2019-01-03	AFKS	2	120.4	8.099000e+00	7.990000e+00	8.099000e+00	7.990000e+00	3	SUR
373045	TQBR	2019-01-03	AFKS	1838	32593754.0	8.030000e+00	7.961000e+00	8.049000e+00	8.043000e+00	3	SUR
373046	EQDP	2019-01-04	AFKS	0	0.0	6.534653e+09	6.539789e+09	6.532886e+09	6.531892e+09	3	SUR
373047	TQBR	2019-01-04	AFKS	2193	47043900.0	8.043000e+00	8.022000e+00	8.240000e+00	8.240000e+00	3	SUR

Определено среднее значение цены закрытия акций

```
[28]: average_close_2019 = data_2019['CLOSE'].mean()
      average_close_2019
```

```
[28]: 506252526.9033347
```

Далее с помощью dt.to_period мы изменили тип данных даты на период,чтобы рассчитать среднюю цену закрытия по месяцам

```
[29]: monthly_avg_close = (
          data_2019.groupby(data_2019['tradedate'].dt.to_period('M'))['CLOSE']
          .mean()
          .reset_index()
          .rename(columns={'tradedate': 'Month', 'CLOSE': 'Avg_Close'})
      )

      print("Средняя цена закрытия за 2019 год:", average_close_2019)
      print("Средняя цена закрытия по месяцам:")
      print(monthly_avg_close)
```

```
Средняя цена закрытия за 2019 год: 506252526.9033347
Средняя цена закрытия по месяцам:
      Month     Avg_Close
0   2019-01  2.252377e+09
1   2019-02  2.252377e+09
2   2019-03  1.674844e+08
3   2019-04  9.181071e+00
4   2019-05  8.923564e+00
5   2019-06  9.456684e+00
6   2019-07  1.162480e+01
7   2019-08  1.150775e+01
8   2019-09  1.254505e+01
9   2019-10  1.317944e+01
10  2019-11  1.542564e+01
11  2019-12  1.538695e+01
```

Чтобы построить визуализацию по этим данным нужно воспользоваться dt.to_timestap чтобы изменить тип на подходящий

```
memory usage: 324.0 bytes
```

[37]:
```
month_avg['Month'] = month_avg['Month'].dt.to_timestamp('s').dt.strftime('%Y-%m')
month_avg
```

[37]:

	Month	Avg_Close
0	2019-01	2.252377e+09
1	2019-02	2.252377e+09
2	2019-03	1.674844e+08
3	2019-04	9.181071e+00
4	2019-05	8.923564e+00
5	2019-06	9.456684e+00
6	2019-07	1.162480e+01
7	2019-08	1.150775e+01
8	2019-09	1.254505e+01
9	2019-10	1.317944e+01
10	2019-11	1.542564e+01
11	2019-12	1.538695e+01

Также углубимся в визуализацию и скачаем seaborn чтобы отобразить распределение цены с помощью sns.histplot

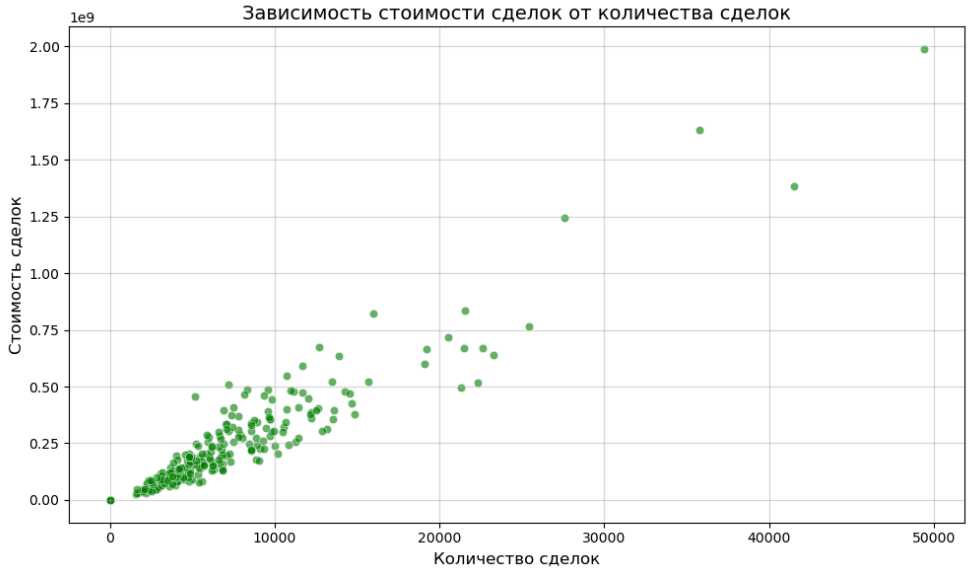
[41]:
```
!pip install seaborn
```

```
Collecting seaborn
  Downloading seaborn-0.13.2-py3-none-any.whl.metadata (5.4 kB)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from seaborn) (1.26.4)
Requirement already satisfied: pandas>=1.2 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from seaborn) (2.2.2)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from seaborn) (3.8.4)
Requirement already satisfied: contourpy>=1.0.1 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (24.0)
Requirement already satisfied: pillow>=8 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (2.9.0)
Requirement already satisfied: pytz>=2020.1 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: six>=1.5 in /home/devops/.config/jupyterlab-desktop/jlab_server/lib/python3.12/site-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.16.0)
Downloading seaborn-0.13.2-py3-none-any.whl (294 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 294.9/294.9 kB 1.6 MB/s eta 0:00:00a 0:00:01
Installing collected packages: seaborn
Successfully installed seaborn-0.13.2
```

[42]:
```
import seaborn as sns
plt.figure(figsize=(10, 6))
sns.histplot(data_2019['CLOSE'], kde=True, bins=30, color='blue', alpha=0.7)
plt.title('Распределение цены закрытия', fontsize=14)
plt.xlabel('Цена закрытия', fontsize=12)
plt.ylabel('Частота', fontsize=12)
plt.grid(alpha=0.5)
plt.tight_layout()
plt.show()
```

[38]:
```
plt.figure(figsize=(12, 6))
plt.plot(month_avg['Month'], month_avg['Avg_Close'], marker='o', label='Средняя цена закрытия')
plt.title('Динамика средней цены закрытия по месяцам (2019)', fontsize=14)
plt.xlabel('Месяц', fontsize=12)
plt.ylabel('Средняя цена закрытия', fontsize=12)
plt.grid(alpha=0.5)
plt.legend()
plt.tight_layout()
plt.show()
```

[43]:
```
plt.figure(figsize=(10, 6))
sns.scatterplot(data=data_2019, x='numtrades', y='value', alpha=0.6, color='green')
plt.title('Зависимость стоимости сделок от количества сделок', fontsize=14)
plt.xlabel('Количество сделок', fontsize=12)
plt.ylabel('Стоимость сделок', fontsize=12)
plt.grid(alpha=0.5)
plt.tight_layout()
plt.show()
```



Выводы по работе:

Hadoop позволяет обрабатывать большие объёмы данных и обеспечивает гибкость в работе с ними. Проведен анализ акций afks,где можно наблюдать динамику цен акций по месяцам. Снижение стоимости

акций началось в феврале-марте и больше не поднималось.Есть Зависимость стоимости сделок от их количества