

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:

Распределенные системы

Лабораторная работа 1

«Установка и настройка распределенной системы. Простейшие операции и знакомство с функциональностью системы»

Выполнил: Башашкин А.М., группа: АДЭУ-221

Преподаватель: Босенко Т.М.

Москва

2024

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

Вариант 5.

Постановка задачи: Настройка кластерного режима для Apache(Arenadata) Hadoop на 2 узлах и проверка работоспособности. Данные: Исторические данные по акциям Роснефти (ROSN)

Операции: Фильтрация данных за последние 3 года, расчет медианной цены закрытия, группировка по месяцам.

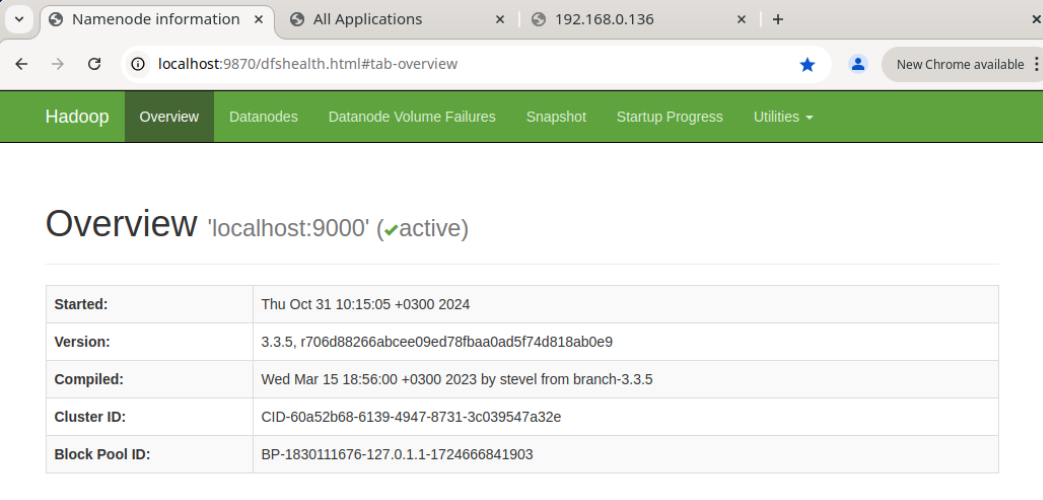
Ход работы:

1. Подключение к Hadoop и загрузка данных

Был запущен Hadoop

```
hadoop@devopsvm:/$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [devopsvm]
2024-10-31 10:15:14,952 WARN util.NativeCodeLoader: Unable to load native-hadoop library for optimization: java.lang.ClassNotFoundException: org.apache.hadoop.util.NativeCodeLoader
hadoop@devopsvm:/$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
```

Проверка web-интерфейса



The screenshot shows a web browser window with the Hadoop web interface. The address bar shows 'localhost:9870/dfshealth.html#tab-overview'. The interface has a green header with tabs: Hadoop, Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. The 'Overview' tab is selected, showing 'localhost:9000' (active). Below this is a table with the following information:

| | |
|----------------|--|
| Started: | Thu Oct 31 10:15:05 +0300 2024 |
| Version: | 3.3.5, r706d88266abcee09ed78fbaa0ad5f74d818ab0e9 |
| Compiled: | Wed Mar 15 18:56:00 +0300 2023 by stevel from branch-3.3.5 |
| Cluster ID: | CID-60a52b68-6139-4947-8731-3c039547a32e |
| Block Pool ID: | BP-1830111676-127.0.1.1-1724666841903 |

2. Исследование и очистка данных.

С официального сайта Роснефть (rosneft.ru) были выгружены исторические данные по акциям с начала 2018 года в формате .xlsx

Так выглядят исходные данные.

| | A | | B | | C | D | E | F |
|----|------------|---------------|---|--------|--------------------|--------|--------|---|
| 1 | Market | | | | MICEX, Main Market | | | |
| 2 | Stock | | | | Common Share | | | |
| 3 | Measure | | | | Russian Rubles | | | |
| 4 | Period | | | | | | | |
| 5 | From: | | | | 03.01.2018 | | | |
| 6 | To: | | | | 29.10.2024 | | | |
| 7 | Date | Volume, items | | Open | High | Low | Close | |
| 8 | 03.01.2018 | 2232020 | | 292.25 | 297.95 | 292.15 | 297.95 | |
| 9 | 04.01.2018 | 5762200 | | 298.95 | 308.25 | 297.6 | 307 | |
| 10 | 05.01.2018 | 4237350 | | 307 | 313.65 | 304.65 | 311.95 | |
| 11 | 09.01.2018 | 4577460 | | 312.6 | 316.25 | 311.15 | 315.1 | |
| 12 | 10.01.2018 | 5564030 | | 316.1 | 319.85 | 312 | 317.75 | |
| 13 | 11.01.2018 | 4586170 | | 317.75 | 325 | 316.05 | 325 | |
| 14 | 12.01.2018 | 5490740 | | 325.5 | 326.95 | 318.55 | 324.35 | |
| 15 | 15.01.2018 | 3518870 | | 325 | 326.7 | 318.75 | 319.95 | |
| 16 | 16.01.2018 | 4620330 | | 320 | 321.45 | 314 | 317.5 | |
| 17 | 17.01.2018 | 6126070 | | 316 | 326.5 | 312.95 | 326.5 | |
| 18 | 18.01.2018 | 10934640 | | 326.9 | 333.35 | 325.05 | 333.1 | |
| 19 | 19.01.2018 | 4224540 | | 332.8 | 332.95 | 326.65 | 328 | |
| 20 | 22.01.2018 | 4538960 | | 326 | 332.95 | 324.4 | 332.95 | |

Были очищены лишние строки:

| | | | | | | |
|----|------------|---------------|--------|--------|--------|--------|
| 1 | Date | Volume, items | Open | High | Low | Close |
| 2 | 03.01.2018 | 2232020 | 292.25 | 297.95 | 292.15 | 297.95 |
| 3 | 04.01.2018 | 5762200 | 298.95 | 308.25 | 297.6 | 307 |
| 4 | 05.01.2018 | 4237350 | 307 | 313.65 | 304.65 | 311.95 |
| 5 | 09.01.2018 | 4577460 | 312.6 | 316.25 | 311.15 | 315.1 |
| 6 | 10.01.2018 | 5564030 | 316.1 | 319.85 | 312 | 317.75 |
| 7 | 11.01.2018 | 4586170 | 317.75 | 325 | 316.05 | 325 |
| 8 | 12.01.2018 | 5490740 | 325.5 | 326.95 | 318.55 | 324.35 |
| 9 | 15.01.2018 | 3518870 | 325 | 326.7 | 318.75 | 319.95 |
| 10 | 16.01.2018 | 4620330 | 320 | 321.45 | 314 | 317.5 |
| 11 | 17.01.2018 | 6126070 | 316 | 326.5 | 312.95 | 326.5 |

Затем данные были сохранены в исходном формате и загружены в датафрейм:

```
[1]: import pandas as pd

[5]: df = pd.read_excel(r"C:\Users\alesh\Downloads\rosn.xlsx")
df

[5]:
```

| | Date | Volume, items | Open | High | Low | Close |
|------|------------|---------------|--------|--------|--------|--------|
| 0 | 2018-01-03 | 2232020 | 292.25 | 297.95 | 292.15 | 297.95 |
| 1 | 2018-01-04 | 5762200 | 298.95 | 308.25 | 297.60 | 307.00 |
| 2 | 2018-01-05 | 4237350 | 307.00 | 313.65 | 304.65 | 311.95 |
| 3 | 2018-01-09 | 4577460 | 312.60 | 316.25 | 311.15 | 315.10 |
| 4 | 2018-01-10 | 5564030 | 316.10 | 319.85 | 312.00 | 317.75 |
| ... | ... | ... | ... | ... | ... | ... |
| 1696 | 2024-10-23 | 4361513 | 478.50 | 480.35 | 467.20 | 469.00 |
| 1697 | 2024-10-24 | 4013241 | 469.55 | 472.10 | 464.10 | 469.75 |
| 1698 | 2024-10-25 | 6547227 | 469.60 | 470.65 | 455.50 | 456.10 |
| 1699 | 2024-10-28 | 8597298 | 451.05 | 455.40 | 433.30 | 434.75 |
| 1700 | 2024-10-29 | 7598465 | 434.75 | 446.55 | 434.50 | 442.35 |

1701 rows × 6 columns

Описание данных.

Ниже представлены типы данных:

```
df2.dtypes

Date                object
Volume, items       int64
Open                float64
High                float64
Low                 float64
Close               float64
dtype: object
```

Date – дата открытия

Volume, Items – объем в штуках

Open – цена открытия акций в начале торгового дня

High - максимальная цена акций за день.

Low - минимальная цена акций за день.

Close - цена закрытия акций в конце торгового дня.

Сохраним датафрейм в формате .csv. Убедимся в корректности данных, повторно выгрузив данные

```
[9]: df.to_csv(r'C:\Users\alesh\Downloads\rosh.csv', index=False)

[10]: df2 = pd.read_csv(r'C:\Users\alesh\Downloads\rosh.csv')
df2

[10]:
```

| | Date | Volume, items | Open | High | Low | Close |
|------|------------|---------------|--------|--------|--------|--------|
| 0 | 2018-01-03 | 2232020 | 292.25 | 297.95 | 292.15 | 297.95 |
| 1 | 2018-01-04 | 5762200 | 298.95 | 308.25 | 297.60 | 307.00 |
| 2 | 2018-01-05 | 4237350 | 307.00 | 313.65 | 304.65 | 311.95 |
| 3 | 2018-01-09 | 4577460 | 312.60 | 316.25 | 311.15 | 315.10 |
| 4 | 2018-01-10 | 5564030 | 316.10 | 319.85 | 312.00 | 317.75 |
| ... | ... | ... | ... | ... | ... | ... |
| 1696 | 2024-10-23 | 4361513 | 478.50 | 480.35 | 467.20 | 469.00 |
| 1697 | 2024-10-24 | 4013241 | 469.55 | 472.10 | 464.10 | 469.75 |
| 1698 | 2024-10-25 | 6547227 | 469.60 | 470.65 | 455.50 | 456.10 |
| 1699 | 2024-10-28 | 8597298 | 451.05 | 455.40 | 433.30 | 434.75 |
| 1700 | 2024-10-29 | 7598465 | 434.75 | 446.55 | 434.50 | 442.35 |

```
1701 rows x 6 columns

[ ]:
```

3. Обработка данных в Spark

Данные были выгружены на GitHub, затем в HDFS

```
hadoop@devopsvm:~$ wget https://raw.githubusercontent.com/Bashashkin/DS/refs/heads/main/rosh.csv
--2024-10-31 10:45:45-- https://raw.githubusercontent.com/Bashashkin/DS/refs/heads/main/rosh.csv
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 77620 (76K) [text/plain]
Saving to: 'rosh.csv'

rosh.csv          100%[=====] 75.80K  ---KB/s   in 0.1s
2024-10-31 10:45:45 (746 KB/s) - 'rosh.csv' saved [77620/77620]
```

Далее был запущен Spark и были выгружены данные

```
scala> val data = spark.read.option("header", "true").csv("file:///home/hadoop/rosh.csv")
data: org.apache.spark.sql.DataFrame = [Date: string, Volume, items: string ... 4 more fields]

scala> data.printSchema()
root
 |-- Date: string (nullable = true)
 |-- Volume, items: string (nullable = true)
 |-- Open: string (nullable = true)
 |-- High: string (nullable = true)
 |-- Low: string (nullable = true)
 |-- Close: string (nullable = true)
```

Была объявлена переменная `three_years_ago` для отсчета 3 лет с настоящего времени – 31 октября 2024 года.

```
scala> val three_years_ago = date_sub(current_date(), 365 * 3)
three_years_ago: org.apache.spark.sql.Column = date_sub(current_date(), 1095)
```

Была произведена фильтрация данных, результат сохраняется в `filtered_data`:

```
scala> val filtered_data = data.filter(col("Date") >= three_years_ago)
filtered_data: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Date: string,
s: string ... 4 more fields]
```

Сохранение результата в нужную директорию

```
scala> filtered_data.write.option("header", "true").csv("home/hadoop/my_output/filtered_data.csv")
```

Для удобства переназовем файл с данными на `filtered_data.csv`, т.к. по умолчанию Hadoop сохраняет результаты в файле `part-00000-*.csv`

```
hadoop@devopsvm:~$ ls
GDP.csv  hadoop-3.3.5.tar.gz  hdfs  my_output  output  rosn.csv  snap  spark-3.4.3-bin-hadoop3.tg
hadoop@devopsvm:~$ cd my_output
hadoop@devopsvm:~/my_output$ ls
filtered_data.csv
hadoop@devopsvm:~/my_output$ cd filtered_data.csv
hadoop@devopsvm:~/my_output/filtered_data.csv$ ls
part-00000-c9b0a8d1-fe6a-4771-8faf-12f0cd240e54-c000.csv  _SUCCESS
hadoop@devopsvm:~/my_output/filtered_data.csv$ mv part-00000-c9b0a8d1-fe6a-4771-8faf-12f0cd240e54-c000.csv filtered_data.csv
hadoop@devopsvm:~/my_output/filtered_data.csv$ ls
filtered_data.csv  _SUCCESS
hadoop@devopsvm:~/my_output/filtered_data.csv$ █
```

Загрузка данных в HDFS, проверка в терминале

```
hadoop@devopsvm:/$ hdfs dfs -put /home/hadoop/my_output/filtered_data.csv /user3/hadoop/input/
2024-10-31 11:10:58,305 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hadoop@devopsvm:/$ hdfs dfs -ls /user3/hadoop/input/
2024-10-31 11:11:51,841 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
drwxr-xr-x   - hadoop supergroup          0 2024-10-31 11:11 /user3/hadoop/input/filtered_data.csv
-rw-r--r--   1 hadoop supergroup        27 2024-10-18 14:29 /user3/hadoop/input/part-00000-32e037td-eaa0-47e8-b522-dc24d302abe2-c000.csv
```

Проверка в web-интерфейсе

Browse Directory

Show 25 entries

Search:

| <input type="checkbox"/> | Permission | Owner | Group | Size | Last Modified | Replication | Block Size | Name | |
|--------------------------|------------|--------|------------|----------|---------------|-------------|------------|-----------------------------------|--|
| <input type="checkbox"/> | -rw-r--r-- | hadoop | supergroup | 0 B | Oct 31 11:11 | 1 | 128 MB | _SUCCESS | |
| <input type="checkbox"/> | -rw-r--r-- | hadoop | supergroup | 31.95 KB | Oct 31 11:11 | 1 | 128 MB | filtered_data.csv | |

Showing 1 to 2 of 2 entries

Также убедимся, что фильтрация прошла успешно, просмотрев первые записи в данных

File information - filtered_data.csv ×

[Download](#) [Head the file \(first 32K\)](#) [Tail the file \(last 32K\)](#)

Block information --

Block 0

Block ID: 1073741834

Block Pool ID: BP-1830111676-127.0.1.1-1724666841903

Generation Stamp: 1010

Size: 32721

Availability:

- devopsvm

File contents

Date,"Volume, items",Open,High,Low,Close

2021-11-01,3588960,637.3,643.25,633.0,641.9

2021-11-02,5482020,643.0,646.3,628.55,630.2

2021-11-03,5037076,627.1,627.95,615.45,619.75

2021-11-05,4556911,612.95,626.75,605.2,626.25

2021-11-08,5018395,633.05,636.5,628.35,631.7

2021-11-09,4362147,632.5,637.95,629.3,636.4

2021-11-10,5309324,638.0,642.5,627.0,629.8

Действительно, записи начинаются с 1 ноября 2021 года, а это ровно 1095 дней с 31 октября 2024 года.

4. Обработка данных в PySpark

Была создана Spark-сессия, также были загружены данные из HDFS.

```
[28]: from pyspark.sql import SparkSession

# Создание SparkSession
spark = SparkSession.builder \
    .appName("Rosneft Data Analysis") \
    .config("spark.hadoop.fs.defaultFS", "hdfs://http://localhost:9870") \
    .config("spark.ui.port", "4050") \
    .getOrCreate()

# Установка количества разделов для shuffle операций
spark.conf.set("spark.sql.shuffle.partitions", "50")

24/10/31 11:47:17 WARN SparkSession: Using an existing Spark session; only runtime SQL configurations will take effect.

[38]: # Чтение данных из HDFS
file_path = "hdfs://localhost:9000/user3/hadoop/input/filtered_data/filtered_data.csv"
df = spark.read.csv(file_path, header=True)

# Просмотр первых строк данных
df.show(5)

+-----+-----+-----+-----+-----+-----+
|   Date|Volume, items| Open| High| Low| Close|
+-----+-----+-----+-----+-----+
|2021-11-01|   3588960| 637.3|643.25| 633.0| 641.9|
|2021-11-02|   5482020| 643.0| 646.3|628.55| 630.2|
|2021-11-03|   5037076| 627.1|627.95|615.45|619.75|
|2021-11-05|   4556911|612.95|626.75| 605.2|626.25|
|2021-11-08|   5018395|633.05| 636.5|628.35| 631.7|
+-----+-----+-----+-----+-----+
only showing top 5 rows
```

Для удобства переименуем столбец «Volume, items» на «Volume»

```
[51]: pandas_df.rename(columns={'Volume, items': 'Volume'}, inplace=True)
pandas_df.head()
```

```
[51]:
```

| | Date | Volume | Open | High | Low | Close |
|---|------------|---------|--------|--------|--------|--------|
| 0 | 2021-11-01 | 3588960 | 637.3 | 643.25 | 633.0 | 641.9 |
| 1 | 2021-11-02 | 5482020 | 643.0 | 646.3 | 628.55 | 630.2 |
| 2 | 2021-11-03 | 5037076 | 627.1 | 627.95 | 615.45 | 619.75 |
| 3 | 2021-11-05 | 4556911 | 612.95 | 626.75 | 605.2 | 626.25 |
| 4 | 2021-11-08 | 5018395 | 633.05 | 636.5 | 628.35 | 631.7 |

Расчет медианной цены закрытия за последние 3 года. Чтобы это сделать, следует преобразовать столбец «Close» в тип данных float64.

```
[56]: pandas_df.dtypes
```

```
[56]: Date      object
      Volume   object
      Open     object
      High     object
      Low      object
      Close    object
      dtype: object
```

```
[59]: pandas_df['Close'] = pd.to_numeric(pandas_df['Close'])
```

```
[60]: pandas_df.dtypes
```

```
[60]: Date      object
      Volume   object
      Open     object
      High     object
      Low      object
      Close    float64
      dtype: object
```

```
[61]: # расчет медианной цены закрытия
```

```
median_close = pandas_df['Close'].median()
median_close
```

```
[61]: 490.15
```


Медианная цена закрытия за последние 3 года – 490 рублей, 15 копеек.

Группировка по месяцам. Чтобы сделать группировку по месяцам следует преобразовать столбец «Date» в тип данных datetime64.

Далее был добавлен столбец «year_month», который показывает год и месяц

```
[68]: pandas_df['Date'] = pd.to_datetime(pandas_df['Date'])

[69]: pandas_df.dtypes

[69]: Date          datetime64[ns]
      Volume          object
      Open           object
      High           object
      Low            object
      Close          float64
      year_month      period[M]
      dtype: object

[63]: pandas_df['year_month'] = pandas_df['Date'].dt.to_period('M')

[65]: pandas_df.head()

[65]:
```

| | Date | Volume | Open | High | Low | Close | year_month |
|---|------------|---------|--------|--------|--------|--------|------------|
| 0 | 2021-11-01 | 3588960 | 637.3 | 643.25 | 633.0 | 641.90 | 2021-11 |
| 1 | 2021-11-02 | 5482020 | 643.0 | 646.3 | 628.55 | 630.20 | 2021-11 |
| 2 | 2021-11-03 | 5037076 | 627.1 | 627.95 | 615.45 | 619.75 | 2021-11 |
| 3 | 2021-11-05 | 4556911 | 612.95 | 626.75 | 605.2 | 626.25 | 2021-11 |
| 4 | 2021-11-08 | 5018395 | 633.05 | 636.5 | 628.35 | 631.70 | 2021-11 |

Расчет медианной цены закрытия по месяцам.

Следующий код вернет данные типа Series, так что названия для второго столбца нет. Чтобы дать название второму столбцу, Series был преобразован в DataFrame.

```
[66]: month_median = pandas_df.groupby('year_month')['Close'].median()
      month_median

[66]: year_month
      2021-11    619.500
      2021-12    575.000
      2022-01    576.950
      2022-02    569.000
      2022-03    367.150
      2022-04    399.000
      2022-05    388.350
      2022-06    375.950
```

```
[72]: month_median_df = month_median.reset_index()
month_median_df.columns = ['year_month', 'median_close']

month_median_df.head()
```

```
[72]:
```

| | year_month | median_close |
|---|------------|--------------|
| 0 | 2021-11 | 619.50 |
| 1 | 2021-12 | 575.00 |
| 2 | 2022-01 | 576.95 |
| 3 | 2022-02 | 569.00 |
| 4 | 2022-03 | 367.15 |

5. Визуализация данных

Был построен линейный график, изображающий отношение медианной цены закрытия по месяцам последних 3 лет.

```
[90]: plt.figure(figsize=(15, 5))
plt.plot(month_median_df['year_month'], month_median_df['median_close'])
plt.title('Медианные цены закрытия по месяцам')
plt.xlabel('Месяц')
plt.ylabel('Медианная цена закрытия')
plt.xticks(rotation=45)
plt.grid()
plt.show()
```



6. Сохранение и экспорт результатов

Был перезаписан файл с исходными данными rosn.csv.

```
[97]: # Преобразование Pandas DataFrame обратно в PySpark DataFrame
month_median_spark = spark.createDataFrame(month_median_df)
```

```
[101]: # Путь в HDFS для сохранения
file_path_hdfs = "hdfs://localhost:9000/user3/hadoop/economic_data/rosn.csv"

# Сохранение DataFrame в формате CSV в HDFS
month_median_spark.write.csv(file_path_hdfs, header=True, mode='overwrite')
```

Можно заметить, что создались два файла .csv с идентичными названиями. Hadoop разделил получившийся результат на две равные части.

Browse Directory

Show 25 entries

Search:

| <input type="checkbox"/> | Permission | Owner | Group | Size | Last Modified | Replication | Block Size | Name | <input type="checkbox"/> |
|--------------------------|----------------------------|------------------------|----------------------------|-------|---------------|-------------------|------------|---|--------------------------|
| <input type="checkbox"/> | -rw-r--r-- | devops | supergroup | 0 B | Oct 31 12:54 | 3 | 128 MB | _SUCCESS | <input type="checkbox"/> |
| <input type="checkbox"/> | -rw-r--r-- | devops | supergroup | 304 B | Oct 31 12:54 | 3 | 128 MB | part-00000-32f177e0-ebf3-41d8-849f-894eab0814bbc000.csv | <input type="checkbox"/> |
| <input type="checkbox"/> | -rw-r--r-- | devops | supergroup | 299 B | Oct 31 12:54 | 3 | 128 MB | part-00001-32f177e0-ebf3-41d8-849f-894eab0814bbc000.csv | <input type="checkbox"/> |

Showing 1 to 3 of 3 entries

Previous

1

Next

Hadoop, 2023.

В терминале посмотрим первую часть данных

```
hadoop@devopsvm:~$ hdfs dfs -cat /user3/hadoop/economic_data/rosn.csv/part-00000-32f177e0-ebf3-41d8-849f-894eab0814bbc000.csv
2024-10-31 13:25:11,138 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
year_month,median_close
2021-11,619.5
2021-12,575.0
2022-01,576.95
2022-02,569.0
2022-03,367.15
2022-04,399.0
2022-05,388.35
2022-06,375.95
2022-07,344.42499999999995
2022-08,344.8
2022-09,363.175
2022-10,305.45
2022-11,336.9
2022-12,336.775
2023-01,344.15
2023-02,349.625
2023-03,365.625
2023-04,393.65
```

Затем – вторую часть. Можно заметить, что данные начинаются с мая 2023 года, то есть ровно полтора года назад. Выгрузка в HDFS была проведена успешно.

```

hadoop@devopsvm:~$ hdfs dfs -cat /user3/hadoop/economic_data/rosn.csv/part-00001-32f177e0-ebf3-41d8-849f-894eab08
14bb-c000.csv
2024-10-31 13:25:59,999 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... usi
ng builtin-java classes where applicable
year_month,median_close
2023-05,413.65
2023-06,475.85
2023-07,485.95
2023-08,539.75
2023-09,555.15
2023-10,570.8499999999999
2023-11,583.3
2023-12,574.0
2024-01,578.4
2024-02,586.25
2024-03,581.45
2024-04,579.15
2024-05,583.7
2024-06,562.2
2024-07,526.7
2024-08,491.725
2024-09,507.4
2024-10,495.85

```

7. Завершение процессов

```

hadoop@devopsvm:~$ stop-all.sh
WARNING: Stopping all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: Use CTRL-C to abort.
Stopping namenodes on [localhost]
Stopping datanodes
Stopping secondary namenodes [devopsvm]
2024-10-31 13:55:09,148 WARN util.NativeCodeLoader: Unable to load n
ng builtin-java classes where applicable
Stopping nodemanagers
Stopping resourcemanager
hadoop@devopsvm:~$ jps
7655 SparkSubmit
13804 Jps
hadoop@devopsvm:~$ kill -9 7655
hadoop@devopsvm:~$ jps
13867 Jps

```

Вывод: В ходе лабораторной работы была развернута распределенная файловая система Hadoop, куда были выгружены исторические данные по акциям Роснефти. Для фильтрации данных был использован функционал Spark на языке Scala, а для выполнения таких операций, как расчет медианной цены закрытия и группировка по месяцам, был использован PySpark в комбинации с Pandas на языке Python. Получившийся результат был визуализирован на графике и выгружен в Hadoop.