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

Ли Александр Андреевич БД-241м

## **Практическая работа 1. Введение в большие данные и их хранение. Инструменты обработки больших данных (Hadoop)**

Направление подготовки/специальность

38.04.05 - Бизнес-информатика

Бизнес-аналитика и большие данные

(очная форма обучения)

Вариант 12

Москва

2024

Цель

Изучить основные операции и функциональные возможности системы, что позволит понять принципы работы с данными и распределенными вычислениями.

# Основная часть

# Запускаем hadoop

# Start-dfs.sh

**Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание**

# Рис.1 Запускаем файловую систему

**Start-yarn.sh**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Рис.2 Запускаем yarn

Проверяем запущенные службы командой

**jps**

**Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание**

Рис.3 Проверка запущенных служб

Проверяем доступность запущенных систем

Переходим по ссылке для проверки запущен ли dfs по ссылке

**Localhost:9870/dfshealth.html#tab-overview**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание**

Рис.4 проверка доступности систем

Проверяем запущен ли yarn по ссылке

**Localhost:8088/cluster**

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

**Рис.5 Проверка работы yarn**

Создаем пользователя и каталог командой

**Hdfs dfs -mkdir -p /lee/Hadoop/input**

**Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание**

**Рис.6 Создание пользователя**

**Изображение выглядит как текст, снимок экрана, число, программное обеспечение

Автоматически созданное описание**

**Рис.7 Проверка созданного пользователя**

**Скачиваем данные**

**wget https://raw.githubusercontent.com/BosenkoTM/Distributed\_systems/main/practice/2024/l w\_01/GDP.csv**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описаниеРис.8 Скачиваем данные**

Создаем каталог для данных

**Hdfs dfs -mkdir -p /lee/Hadoop/input/economic\_data**

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание**

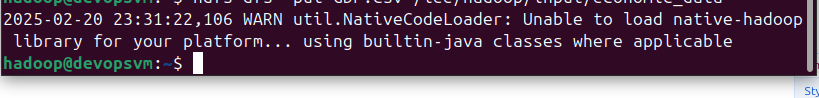
**Рис.9 Создание каталога economic\_data**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание**

**Рис.10 Проверка созданного каталога economic\_data**

**Переносим данные в каталог**

**Hdfs dfs -put GDP.csv /lee/Hadoop/input/economic\_data** ****

**Рис.11 Перенос данных в каталог**

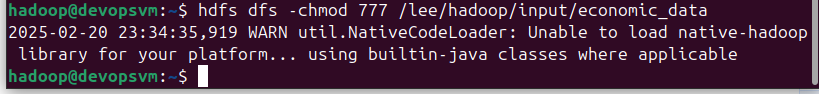
**Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание**

**Рис.12 Проверка перенесенных данных**

Задаем права доступа

**hdfs dfs -chmod 777 /lee/hadoop/input/economic\_data**



**Рис.13** Устанавливаем права доступа

**Обрабатываем данные при помощи Spark**

**Spark-shell**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

**Рис.14** Запуск spark

Загружаем данных из hdfs

**val data = spark.read.option("header", "true").csv("file:///home/hadoop/GDP.csv")**

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

**Рис.15** Загрузка данных

Проверка полученной схемы данных

**data.printSchema()**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

**Рис.12** Выводим схему

Вычисление среднего значения GDP

**val result = data.selectExpr("avg(GDP) as avg\_GDP")**

**Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание**

**Рис.13** Вычисление среднего значения GDP

Сохраняем результата в CSV файл

**result.write.option("header", "true").csv("/home/hadoop/output/avg\_GDP.csv")**

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

**Рис.14** сохранение результатов

Переходим в директорию с результатами

**cd /home/hadoop/output/avg\_GDP.csv**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**Рис.15 Проверяем полученный файл**

Загружаем полученный файл в HDFS

**hdfs dfs -put /home/hadoop/output/avg\_GDP.csv /user01/hadoop/input/**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание**

**Рис.16 Загрузка файла в HDFS**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание**

**Рис.17 Проверка загруженного файла в hdfs**

Проверка загрузки

**hdfs dfs -ls /user01/hadoop/input/**

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

**Рис.18 Проверка загрузки**

Завершение работы с Hadoop

**stop-yarn.sh**

**stop-dfs.sh**

**stop-all.sh**

Проверка остановки всех процессов: **jps**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**Рис.18 завершение**

**Задание для самостоятельной работы**

Подключиться к HDFS и убедиться, что файл доступен по пути hdfs://localhost:9000/lee/hadoop/economic\_data/GDP.csv

Запускаем jupyterlab и загружаем файлы из hdfs

**File\_path= “hdfs://localhost:9000/lee/Hadoop/input/economic\_data/GDP.csv”**

**df = spark.read.csv(file\_path, header=True, inferShema =True )**

**df.show(5)**

Изображение выглядит как текст, число, Шрифт, линия

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

**Рис.19 Загружаем данные из hdfs и выводим 5 строк полученных данных**

Проверяем структуру данных и типы столбцов.

Изображение выглядит как текст, Шрифт, линия, число

Автоматически созданное описание

**Рис.20 Проверка структуры данных**

Просматриваем типы данных столбцов

Print(pandas\_df.dtypes)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

**Рис.21 Тип данных столбцов**

Проверяем данные на наличие пропущенных.

Изображение выглядит как текст, программное обеспечение, Значок на компьютере, число

Автоматически созданное описание

**Рис.22 Проверка полученных значений**

Вычисляем среднее, медиану, минимум и максимум для экономических параметров.

**Print(“mean:”,pandas\_df[‘GDP’].mean())**

**Print(“median:”,pandas\_df[‘GDP’].median())**

**Print(“min:”,pandas\_df[‘GDP’].min())**

**Print(“max:”,pandas\_df[‘GDP’].max())**

Изображение выглядит как текст, снимок экрана, Шрифт, дисплей

Автоматически созданное описание

**Рис.23 Вычисляем параметры**

Выявляем тенденцию

**Trend = pandas\_df.groupby(‘Year’)[‘GDP’].mean()**

**Print (trend)**

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

**Рис.24 Выявление тенденций**

Строим временные ряды, чтобы понять, как изменялась их экономика с течением времени для Италии

**Italy\_data = pandas\_df[pandas\_df[‘Country’] == ‘Italy’]**

**Italy\_summary\_statistics = Italy\_data.describe()**

**plt.figure(figsize=(10,6))**

**plt.plot(Italy\_data[‘Year’],Italy\_data[‘GDP’],marker = ‘o’, linestyle =’-’, color = ‘b’, label = ‘GDP’)**

**plt.title(‘GDP of Italy over years’)**

**plt.xlabel(‘year’)**

**plt.ylabel(‘GDP’)**

**plt.legend()**

**plt.show()**

**Italy\_summary\_statistics**

Изображение выглядит как текст, снимок экрана, линия, График

Автоматически созданное описание

**Рис.25 Построение временного ряда**

Строим диаграммы для сравнения экономических показателей.

**italy\_data = pandas\_df[pandas\_df['Country'] == 'Italy']**

**labels = ['Industry', 'Business', 'Mining', 'Manufacturing', 'Electricity\_supply', 'Water\_supply','Retail trade', 'Transportation', 'Accommodation', 'Information','Financial', 'Professional\_scientific', 'Administrative', 'Education','Human\_health', 'Arts', 'Other']**

**data\_mean\_italy = italy\_data[labels].mean().values**

**plt.figure(figsize=(10, 6))**

**plt.bar(labels,data\_mean\_italy,color = 'blue')**

**plt.xticks(rotation=90)**

**plt.legend()**

**plt.tight\_layout()**

**plt.show()**

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Автоматически созданное описание

Рис.26 Диаграмма для сравнения экономических показателей

# Фильтруем данные для Италии

italy\_data = pandas\_df[pandas\_df['Country'] == 'Italy']

labels = ['Industry', 'Business', 'Mining', 'Manufacturing', 'Electricity\_supply', 'Water\_supply',

'Construction', 'Retail trade', 'Transportation', 'Accommodation', 'Information',

'Financial', 'Real estate', 'Professional\_scientific', 'Administrative', 'Education',

'Human\_health', 'Arts', 'Other']

data\_mean\_italy = italy\_data[labels].mean().values

plt.figure(figsize=(10, 6))

plt.bar(labels, data\_mean\_italy, color='blue')

plt.xticks(rotation=90)

plt.tight\_layout()

plt.show()

Изображение выглядит как текст, снимок экрана, График, линия

Автоматически созданное описание

Рис.26 Диаграмма для сравнения экономических показателей для Италии

Сохраняем результат анализа и визуализации в формате CSV.

**italy\_data\_spark = spark.createDataFrame(Italy\_data)**

**file\_path\_hdfs = "hdfs://localhost:9000/lee/hadoop/input/economic\_data/Italy\_data.csv"**

**italy\_data\_spark.write.csv(file\_path\_hdfs, header=True, mode='overwrite')**

Сохраняем обработанные данные обратно в HDFS

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рис.27 Сохраняем обработанные данные в hdfs

Изображение выглядит как текст, Шрифт, число, снимок экрана

Автоматически созданное описание

Рис.28 Сохраняем обработанные данные в hdfs

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рис.29 Сохраняем обработанные данные в hdfs

**Индивидуальное задание**

**Вариант 12**

Исторические данные по акциям Сургутнефтегаз

[**https://ru.investing.com/equities/surgutneftegas\_rts-historical-data**](https://ru.investing.com/equities/surgutneftegas_rts-historical-data)

Так как ссылка не работала, нашёл в интернете данные по акциям Сургутнефтегаз в период с 03.09.2020 по 03.04.2025

Для начала создаем каталог в hdfs

**Hdfs dfs -mkdir -p /lee/hadoop/input/surgutneftgaz**

**Изображение выглядит как текст, Шрифт, число, линия

Автоматически созданное описание**

**Рис.30 Cоздание каталога surgutneftgaz в hdfs**

Переносим файл из пользователя devops в пользователя Hadoop

**sudo mv /home/devops/Desktop/dataset\_surgutneftgaz.csv /home/hadoop**

Переносим данные в каталог

**Hdfs dfs -put dataset\_surgutneftgaz.csv /lee/Hadoop/input/economic\_data**

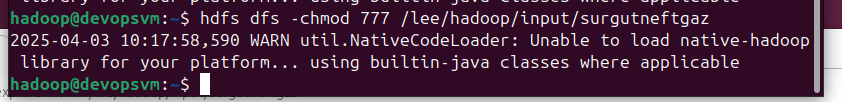
**Изображение выглядит как текст, Шрифт, линия, число

Автоматически созданное описание**

**Рис.31 Переносим данные об акциях в hdfs**

Устанавливаем права доступа

**hdfs dfs -chmod 777 /lee/hadoop/input/surgutneftgaz**



**Рис.32** Устанавливаем права доступа

Загружаем данные из hdfs

Import pandas as pd

Import matplotlib.pyplot as plt

From pyspart.sql import SparkSession

Spark = SparkSession.builder\

.appName(“Economic Data Analysis”)\

.config(“spark.hadoop.fs.defaultFS”,”hdfs://localhost:9000”)\

.get0Create()

Spark.conf.set(“spark.sql.shuffle.partition”, “50”)

File\_path = “hdfs://localhost:9000/lee/Hadoop/input/surgutneftgaz/dataset.csv”

Df= spark.read.csv(file\_path, header = True, inferSchema =True)

Df.show(5)

**Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание**

**Рис.33 Загрузка данных в hdfs**

Просматриваем полученные данные

**Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание**

**Рис.34** Просмотр полученные данные

Фильтрация данных за 2020 год, преобразуем date из числового формата в дату

**pandas\_df["data"] = pd.to\_datetime(pandas\_df["data"], dayfirst=True)**

**df\_2020 = pandas\_df[pandas\_df["data"].dt.year == 2020]**

**print(df\_2020.head())**

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рис.35 Преобразование date в формат даты

Рассчитываем стандартное отклонение цены открытия

Df\_2020[“open”] = df\_2020[“open”].str.replace(“,”,””).astype(float)

Std\_open =df\_2020[“open”].std()

Print(f”standart deviation:{std\_open:.2f}”)

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рис.36 Стандартное отклонение цены открытия

Тренд-анализ

**df\_2020["Days"] = (df\_2020["data"] - df\_2020["data"].min()).dt.days**

**slope, intercept, r\_value, p\_value, std\_err = linregress(df\_2020["Days"], df\_2020["open"])**

**df\_2020["Trend"] = intercept + slope \* df\_2020["Days"]**

**plt.figure(figsize=(10, 5))**

**plt.plot(df\_2020["data"], df\_2020["open"], label="Цена открытия", linestyle="dotted", color="blue")**

**plt.plot(df\_2020["data"], df\_2020["Trend"], label="Тренд (Линейная регрессия)", color="red")**

**plt.xlabel("data")**

**plt.ylabel("price open")**

**plt.title("Тренд цены открытия за 2020 год")**

**plt.legend()**

**plt.show()**

**Изображение выглядит как текст, диаграмма, График, линия

Автоматически созданное описание**

**Рис.37 Тренд анализ**

Расчет волатильности и анализ динамики цен открытия

**Pandas\_df[“open”] = pandas\_df[“open”].str.replace(“,”,”.”).astype(float)**

**Volatility =padndas\_df[“open”].std()**

**Print(volatility)**

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание**

**Рис.38 Расчет волатильности**

Анализ динамики цен открытия

**pandas\_df["price change"]= pandas\_df["open"].pct\_change() \*100**

**plt.figure(figsize=(12, 8))**

**plt.plot(pandas\_df["data"],pandas\_df["price change"],label= "price change open",color = "green")**

**plt.xlabel('data')**

**plt.ylabel('price change %')**

**plt.legend()**

**plt.show()**

**Изображение выглядит как снимок экрана

Автоматически созданное описание**

**Риc.39 Анализ динамики цен открытия**

# Заключение

В ходе проделанной лабораторной работы, были изучены основные операции и функциональные возможности системы, что позволило понять принципы работы с данными и распределенными вычислениями, также было выполнено задание по вариантам.