



Министерство науки и высшего образования Российской Федерации
Калужский филиал федерального государственного автономного
образовательного учреждения высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК Информатика и управление

КАФЕДРА ИУК4 Программное обеспечение ЭВМ, информационные технологии

ДОМАШНЯЯ РАБОТА

«ОСНОВЫ SPARK. УСТАНОВКА SPARK. ОСНОВНЫЕ КОМАНДЫ ДЛЯ РАБОТЫ С RDD»

по дисциплине: «Технологии обработки больших данных»

Выполнил: студент группы ИУК4-72Б

(Подпись)

Губин Е.В.

(И.О. Фамилия)

Проверил:

(Подпись)

Голубева С.Е.

(И.О. Фамилия)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:

- Оценка:

Калуга, 2025

Цель: выполнения домашней работы является формирование практических навыков работы с платформой Apache Spark для обработки больших данных.

Задачи:

1. Изучить основы Apache Spark.
2. Научиться устанавливать и конфигурировать Spark.
3. Уметь работать с RDD.
4. Получить навыки написания программ для обработки больших данных.

Вариант №8

Формулировка задания:

Подсчитать средний рейтинг фильма. Входной файл рейтингов имеет формат:
userId, movieId, rating, timestamp

Выполнить операцию объединения с файлом, содержащим названия фильмов. Данный файл имеет формат:

movieId, title, genres

Если для фильма указано несколько жанров, оставить только первый. Сгруппировать записи по жанру и подсчитать средний рейтинг жанра. Результат сохранить в файл:
genre, av_rating
Исходные файлы:

[rating.csv](#)

[movies.csv](#)

Ход выполнения работы:

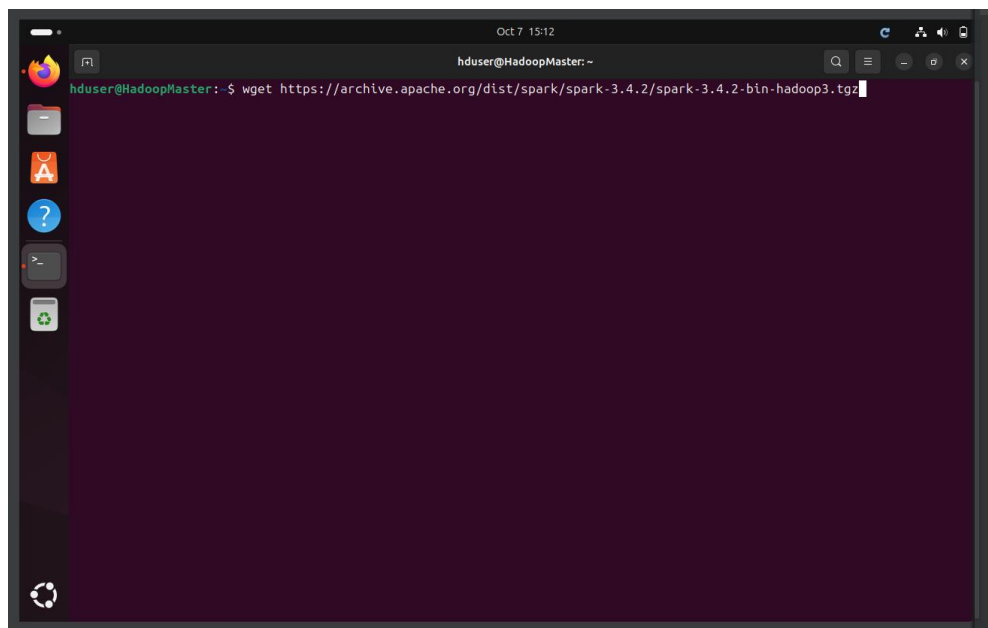
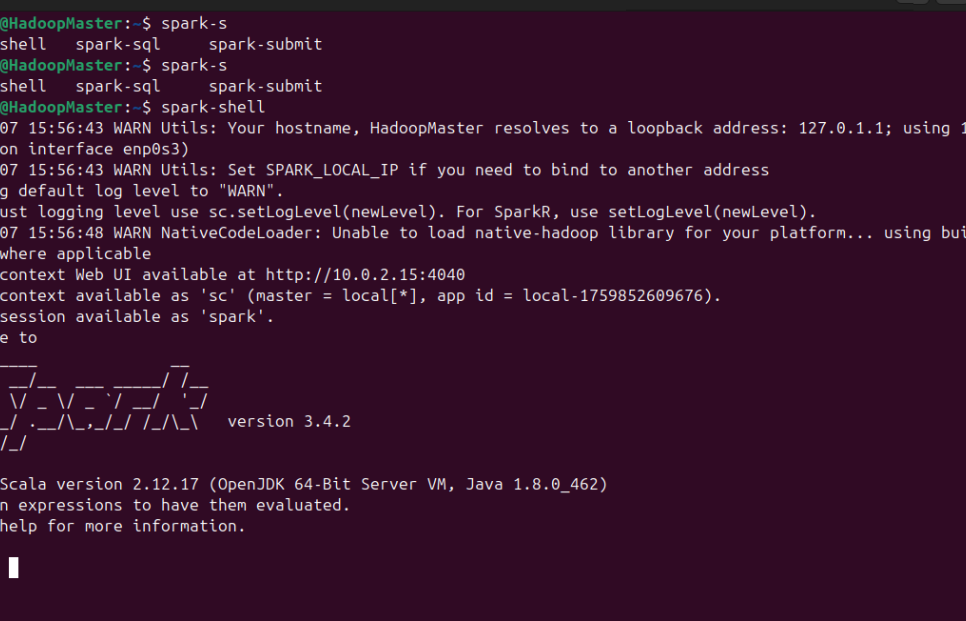


Рисунок 1 Установка Spark

```
# ### SPARK_SETTINGS_START ###  
export SPARK_HOME=/home/hduser/spark  
export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin  
export PYSARK_PYTHON=/usr/bin/python3  
# ### SPARK_SETTINGS_END
```

Рисунок 2 Переменные окружения Spark



The screenshot shows a terminal window titled "hduser@HadoopMaster: ~" with a dark background. The terminal displays the following sequence of commands and output:

```
hduser@HadoopMaster:~$ spark-s
spark-shell spark-sql spark-submit
hduser@HadoopMaster:~$ spark-s
spark-shell spark-sql spark-submit
hduser@HadoopMaster:~$ spark-shell
25/10/07 15:56:43 WARN Utils: Your hostname, HadoopMaster resolves to a loopback address: 127.0.1.1; using 10.0.2.15 instead (on interface enp0s3)
25/10/07 15:56:43 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
25/10/07 15:56:48 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Spark context Web UI available at http://10.0.2.15:4040
Spark context available as 'sc' (master = local[*], app id = local-1759852609676).
Spark session available as 'spark'.
Welcome to

  ____      _
 / ___|    / \
| |  | |  / _ \
| |  | | / ___ \
| |  | |/_/   \_\
| |  | |
| |  | |
|_|  |_|      version 3.4.2

Using Scala version 2.12.17 (OpenJDK 64-Bit Server VM, Java 1.8.0_462)
Type in expressions to have them evaluated.
Type :help for more information.

scala> |
```

The terminal window includes standard Linux desktop icons on the left (Firefox, Files, Applications, Help, Terminal, Recycle Bin) and a top status bar showing the date and time as "Oct 7 15:56".

Рисунок 3 Проверка работоспособности Spark

A	B	C
movielfield	title	genres
101	The Matrix	Action Sci-Fi
102	Toy Story	Animation Adventure Comedy
103	Titanic	Romance Drama
104	The Godfather	Crime Drama

Рисунок 4 movie.csv

	A	D	C	D
	userid	movielid	rating	timestamp
1	1	101	4	9,65E+08
2	1	102	3	9,65E+08
3	2	101	5	9,65E+08
4	3	103	2	9,65E+08
5	3	104	4	9,65E+08
6	4	102	3	9,65E+08
7	5	104	5	9,65E+08

Рисунок 5 rating.csv

```
hduser@HadoopMaster:~$ cd /home/hduser/spark/
bin/      data/      jars/      licenses/  R/          yarn/
conf/     examples/ kubernetes/ python/    sbin/
hduser@HadoopMaster:~$ cd /home/hduser/spark/
bin/      data/      jars/      licenses/  R/          yarn/
conf/     examples/ kubernetes/ python/    sbin/
hduser@HadoopMaster:~$ cd /home/hduser
hduser@HadoopMaster:~$ mkdir -p ./spark_data
hduser@HadoopMaster:~$ ls ~/Downloads/
movies.csv  rating.csv
hduser@HadoopMaster:~$ mv ~/Downloads/movies.csv ./spark_data
hduser@HadoopMaster:~$ mv ~/Downloads/rating.csv ./spark_data
hduser@HadoopMaster:~$ cd spark_data/
hduser@HadoopMaster:~/spark_data$ ls
movies.csv  rating.csv
hduser@HadoopMaster:~/spark_data$
```

Рисунок 6 Инициализация рабочего пространства для Spark

```
hduser@HadoopMaster:~/spark_data$ mv ~/Downloads/main.py .
hduser@HadoopMaster:~/spark_data$ ls
main.py  movies.csv  rating.csv
hduser@HadoopMaster:~/spark_data$ spark-submit ./main.py
25/10/07 16:23:25 WARN Utils: Your hostname, HadoopMaster res
```

Рисунок 7 Запуск скрипта Python

```
25/10/07 17:42:26 INFO CodeGen
+-----+-----+
| genre|avg(rating)|
+-----+-----+
| Crime|         4.5|
| Romance|       2.0|
| Animation|     3.0|
| Action|         4.5|
+-----+-----+
```

Рисунок 8 Результат работы программы

```
genre;av_rating
Action;4.5
Animation;3.0
Crime;4.5
Romance;2.0
```

Рисунок 9 Сохранённые результаты

Листинг программы:

```
#!/usr/bin/env python3
from pyspark.sql import SparkSession
from pyspark.sql import functions as F
```

```

spark = SparkSession.builder.appName("GenreAverageRating").getOrCreate()

ratings_path = "file:///home/hduser/spark_data/rating.csv"
movies_path  = "file:///home/hduser/spark_data/movies.csv"
output_path  = "file:///home/hduser/spark_data/output"

ratings = spark.read.option("header", True).option("inferSchema",
True).option("sep", ";").csv(ratings_path)
movies  = spark.read.option("header", True).option("inferSchema",
True).option("sep", ";").csv(movies_path)

ratings = ratings.select("movieId", "rating")
movies  = movies.select("movieId", "genres")

movies = movies.withColumn("genre", F.split(F.col("genres"),
"\|").getItem(0)).select("movieId", "genre")

joined = ratings.join(movies, on="movieId", how="inner")

genre_avg =
joined.groupBy("genre").agg(F.avg("rating").alias("av_rating_raw"))

genre_avg.show()
genre_avg = genre_avg.withColumn(
    "av_rating",
    (F.floor(F.col("av_rating_raw") * 10 + F.lit(0.5)) / F.lit(10))
)

result = genre_avg.select("genre", "av_rating").orderBy("genre")

result.show(truncate=False)

result.write.mode("overwrite").option("header", True).option("sep",
";").csv(output_path)

spark.stop()

```

Вывод: в ходе выполнения работы были получены практические навыки по работе со Spark.