|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  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»**

**ДИСЦИПЛИНА: «Технологии обработки больших данных»**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил: студент гр. ИУК4-72Б | |  |  | ( | Сафронов Н.С. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |
| Проверил: | |  |  | ( | Голубева С.Е. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |

|  |  |
| --- | --- |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | |
|  | - Балльная оценка:  - Оценка: |

Калуга, 2023

**Цель работы:** формирование практических навыков работы с платформой Apache Spark для обработки больших данных.

**Постановка задачи**

Написать скрипт для платформы Apache Spark для решения задачи, указанной в варианте.

**Вариант 4**

Подсчитать средний рейтинг фильма. Входный файл рейтингов имеет формат:

userId, movieId, rating, timestamp

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

movieId, title, genres

Результат должен быть сохранен в файле в формате:

movieId, title, av\_rating

Объединение выполнять согласно подходу Job-side join. В результате должны быть представлены 20 фильмов с самым высоким средним рейтингом.

Исходные файлы:

rating.csv

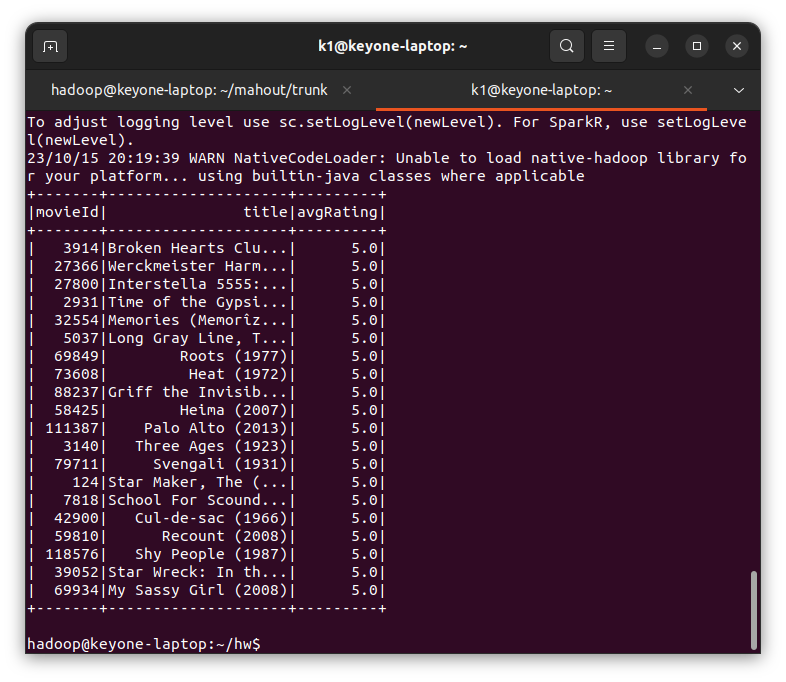
movies.csv

**Результат выполнения работы**

**Листинг программы**

from pyspark.sql import SparkSession, functions  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 spark = SparkSession.builder.appName('keyone-app').getOrCreate()  
  
 movies\_path = 'data/movies.csv'  
 ratings\_path = 'data/ratings.csv'  
 result\_path = 'data/result'  
  
 movies = spark.read.csv(movies\_path, header=True)  
 ratings = spark.read.csv(ratings\_path, header=True)  
  
 average\_ratings = ratings.groupBy('movieId').agg(  
 functions.mean('rating').alias('avgRating')  
 ).select('movieId', 'avgRating')  
  
 joined\_table = average\_ratings.join(movies, 'movieId').select(  
 'movieId', 'title', 'avgRating'  
 )  
 sorted\_table = joined\_table.orderBy('avgRating', ascending=False)  
  
 result\_table = sorted\_table.limit(20)  
 result\_table.show()  
 result\_table.write.mode('overwrite').csv(result\_path)

**Результаты выполнения скрипта**



**Рисунок 1 –** Результат выполнения скрипта

**Вывод**: в ходе выполнения домашней работы были сформированы практические навыки работы с платформой Apache Spark для обработки больших данных.