Project phase 1
Data Analysis with Spark

Due date:

14 March 2018 23:59

The project of the course TDT4305 consists of two phases. This document describes the first one which focuses on learning how to perform data analysis on a large dataset. You will work with the Apache Spark framework[1] and you can choose freely which of the Spark-compatible languages you prefer: Python, Scala or Java. The spark operations should be enough and there is no need to use any other libraries. You can work in groups of max. two people, and both need to deliver on Blackboard.

1 Exploratory Analysis of Music Label Dataset

In this task you will work with a music dataset downloaded from Kaggel. The aim of this phase is to learn the differences and the best practices in usage of various Spark functions.

1.1 Data

A link to the dataset is available on Blackboard, under the “Prosjekt/Project” tab. It is a zip file that you need to unzip it to use it, and it contains the following:

albums.csv

This file contains information about music albums. It is represented in CSV format in the following columns:

1. **id** – the id of the music album
2. **artist_id** – the artist id of the album
3. **album_title**
4. **genre**
5. **year_of_pub** – the year the album was published

[1]http://spark.apache.org/

6. **num_of_tracks**
7. **num_of_sales**
8. **rolling_stone_critic**
9. **mtv_critic**
10. **music_maniac_critic**

artists.csv

This file contains information about music artists. It is representes in CSV format in the following columns:

1. **id** – the artist id. the same id with the column 2 in albums.csv
2. **real_name**
3. **art_name**
4. **role**
5. **year_of_birth**
6. **country**
7. **city**
8. **email**
9. **zip_code**

1.2 Important notes before you start

- Try to use appropriate Spark functions with respect to the tasks you are trying to accomplish. Use of Spark and distributed execution concepts in your design will be taken into consideration, so use Spark as much as possible in your computations.
- Try not to use external libraries. The tasks in the project can be implemented with core Python/Scala/Java.
- Use only data/columns needed for each task to make your solution efficient.
- To output results into a file, use **saveAsTextFile** function, and format your output according to each task. The expected output format is given for each task.
- To avoid multiple files when exporting results into a file, decrease number of data partitions using **coalesce(1)** or **repartition(1)** functions. In your report, write down if you used any, which one and why.
- For the string fields use case-insensitive comparison.
- Have in mind that some fields might be empty.

1.3 RDD API Tasks

On your local machine install Spark, download above mentioned datasets, load the dataset into an **RDD**, and using suitable Spark functions (such as map, reduce, reduceByKey, sort, etc.) perform these tasks:

1. How many distinct genres are there? Write a code (named "task_1") that prints the result and include the result in your report.

2. What is the year of birth of the oldest artist? Write a code (named "task_2") that prints the result and include the result in your report.

3. Find the total number of artists coming from each country and sort them in descending order of artists counts. For countries with equal number of artists, sorting must be in alphabetical order. Write a code (named "task_3") that writes the results in a TSV file in the form of <country_name>tab<artists_count> and name it "result_3".

4. Find the total number of albums each artist has and sort them in descending order of album counts. For artists with equal number of albums, sorting must be in numerical artist_id order. Write a code (named "task_4") that writes the results in a TSV file in the form of <artist_id>tab<albums_count> and name it "result_4".

5. Find the total number of sales per genre and sort them in descending order of album sales. For genre with equal number of total sales, sorting must be in alphabetical order. Write a code (named "task_5") that writes the results in a TSV file in the form of <genre>tab<total_num_sales> and name it "result_5".

6. Each album has 3 critics. Find the top 10 albums with the best average critic according to the following equation:

$$average\_critic = \frac{rolling\_stone\_critic + mtv\_critic + music\_maniac\_critic}{3}$$

   Write a code (named "task_6") that writes the results in a TSV file in the form of <album_id>tab<average_critic> and name it "result_6".

7. For the 10 albums of task 6, find the countries of their artists. Write a code (named "task_7") that writes the results in a TSV file in the form of <album_id>tab<average_critic>tab<artist_country> and name it "result_7".

8. Find the artists have an album with the highest (5.0) MTV critic, sorted alphabetically. Write a code (named "task_8") that writes the results in a TSV file in the form of <artist_name> and name it "result_8".

9. Find the average MTV critic of all albums for each artist from Norway (country = 'Norway') according to the following equation:

$$average\_mtv\_critic = \frac{\sum_{i=1}^{|\text{artist's albums}|} mtv\_critic_i}{|\text{artist's albums}|}$$

   Sort the elements according to the average MTV critis (from high to low). For artists with equal average MTV critics, sorting must be in alphabetical order. Write a code (named "task_9") that writes the results in a TSV file in the form of <artist_name>tab<country>tab<average_mtv_critic> and name it "result_9".

1.4 Dataset API Tasks

Working on Spark, load the datasets into a **Dataframe**, name it, rename the columns with the names of columns described before in section 1.1, and using suitable Dataframe functions perform these tasks:

10. Calculate the following using Spark SQL on your Dataframe:
    (a) Number of distinct artists
    (b) Number of distinct albums
    (c) Number of distinct genres
    (d) Number of distinct countries
    (e) Minimum year_of_pub
    (f) Maximum year_of_pub
    (g) Minimum year_of_birth
    (h) Maximum year_of_birth

Write a code (named "task_10") that calculates the results and prints them with the show() function in the console. Don't forget to include them in your report also.

1.5 Delivery

You should deliver to Blackboard a ZIP file containing:

- a short report (PDF)
- source codes of your script/program with the described names (.py/.scala/ .java[2])
- output files (.tsv) with your results as described for each task

Do not include the datasets in the ZIP file.

In your report, briefly (3-5 sentences) describe which Spark functions you used in your solution and why. Also, write if you used coalesce(1) or repartition(1) and why. If you used Java for your solution, write in the report the compile instructions and the spark submit parameters.

As mentioned before, you can work in groups of max. two people, providing both names in your report and both should deliver the project on Blackboard.

There will be slots for presentation (and questioning) about both parts of the project in weeks 14 and 15.

[2]For java coding, include also a *RUNNABLE* jar with the compile and the spark submit parameters