AY 2020/2021

Politecnico di Milano

Middleware Technologies Analysis of COVID-19 Data

Federico Armellini Luca Pirovano Nicolò Sonnino

Professor
Alessandro Margara

Version 1.1
August 24, 2021

Contents

1 Introduction and assignment

1.1 Description of the project

In this project, you have to implement a program that analyzes open datasets to study the evolution of the COVID-19 situation worldwide. The program starts from the dataset of new reported cases for each country daily and computes the following queries:

1. Seven days moving average of new reported cases, for each county and for each day
2. Percentage increase (with respect to the day before) of the seven days moving average, for each country and for each day
3. Top 10 countries with the highest percentage increase of the seven days moving average, for each day

You can either use real open datasets 1 or synthetic data generated with the simulator developed for Project 4. A performance analysis of the proposed solution is appreciated (but not mandatory). In particular, we are interested in studies that evaluate (1) how the execution time changes when increasing the size of the dataset and/or number of countries; (2) how the execution time decreases when adding more processing cores/hosts.

1.2 Assumptions and guidelines

- When using a real dataset, for countries that provide weekly reports, you can assume that the weekly increment is evenly spread across the day of the week.

2 Solution Overview

2.1 Architecture chosen

Apache Spark

2.2 Assumptions and Definitions

- The dataset is in CSV format. Each line contains day, rank, number of infected people, number of sane people, number of newly infected people (then the day before), the number of new sane people (then the day before).

2.3 General solution

The project is based on SparkSQL library, which permits doing SQL query and distribute them among the machines in the Apache Spark cluster.

We used the data provided from ECDC (European Centre For Disease Control and Prevention) which has a general report of Covid-19 in 2020.

2.4 Queries

2.4.1 Query 1

The first query goal is to calculate a moving average of a window of 7 days.

To do this, we adopted the following approach:

Listing 1: Query 1

```
final WindowSpec ws1 = Window
.partitionBy("countriesAndTerritories")
.orderBy("date", "geoId")
.rowsBetween(-6,0);

final Column col1 = functions.avg("cases").over(ws1);

final Dataset<Row> df1 = df
.withColumn("date", functions
        .to_timestamp(df.col("dateRep"), "dd/MM/yyyy"))
.withColumn("movingAverage", functions.round(col1, 2));
```

The Window declaration defines a custom window which is:

- partitioned by countries name;
- ordered by their geoId and date of observation
- 7 days large, which means that it considers a week from the day considered.

2.4.2 Query 2

The second query goal is to calculate the percentage increase of the moving average calculated at point 1 with respect to the day before.

Listing 2: Query 2

```
final WindowSpec ws2 = Window
.partitionBy("countriesAndTerritories")
.orderBy("date", "geoId");

final Dataset<Row> df2 = df1
.withColumn("prevValue", functions
        .round(functions.lag("movingAverage", 1).over(ws2), 2));

final Column col3 = df2
.col("movingAverage").minus(df2.col("prevValue"));

final Column col2 = col3
.divide(df2.col("prevValue")).multiply(100).cast("float");

Dataset<Row> df3 = df2
.withColumn("perc_increase", functions
        .when(functions.isnull(col3),0)
        .otherwise(functions.round(col2, 2))).drop("prevValue");

df3 = df3.filter(df3.col("perc_increase").isNotNull());
```

Again we have a window which is the same of the one declared for query 1.

Then, we create a new column *prevValue* which represent the moving average value of the day before.

This value is then used to create a column *perc_increase* which is calculated in that way.

$$perc\_increase = \frac{maCurrent - maDayBefore}{maDayBefore} * 100$$

2.4.3 Query 3

Listing 3: Query 3

```
final WindowSpec ws3 = Window
.partitionBy("date")
.orderBy(functions.desc("perc_increase"));
```

```
final Dataset<Row> df4 = df3
.withColumn("rankingPercIncrease", rank().over(ws3));

final Dataset<Row> df5 = df4
.where("rankingPercIncrease<=" + maxRankCountries)
.orderBy("date", "rankingPercIncrease");
```

Again we have a window which is the same of the one declared for query 1.

Then, we calculate the rank of nations relying on their percentage increase.