PARALLEL DISTRIBUTED PROCESSING (HADOOP)

Assignment 2

Abdullah Ghayumi
634072@student.inholland.nl

Assignment 2

Details

Name: Abdullah Ghayumi

Student number: 634072

GitHub URL: https://github.com/Aras53/Hadoop

Table of Contents

Problem

- Make an alphabetic list from all locations from the orders.csv.
- Group by “location” with target “Holland”
- Count how many times Holland was the target from that location
- Code is executed from Pig View
- Upload a document on Moodle with the following information:
- Name and student number
- URL from GitHub with your source code
- Explain in the document what steps have to be taken to execute the code
- Explain always in your own words every step included source code.
- Make a screenshot from the result and include it in your document

Solution

Uploading orders.csv

The first step I took was to upload the file to the Files View, but I quickly found out that the file was too big and wouldn't fully upload. To mitigate this, I tried uploading the file via MobaXTerm.

I uploaded the file first to the /home/maria_dev location that I could access. Then I used the sudo root command to be able to access and execute command on the hdfs repo. I copied the orders file to the /users/maria_dev folder with the following command:

hdfs dfs -put /home/maria_dev/orders.csv /user/maria_dev/

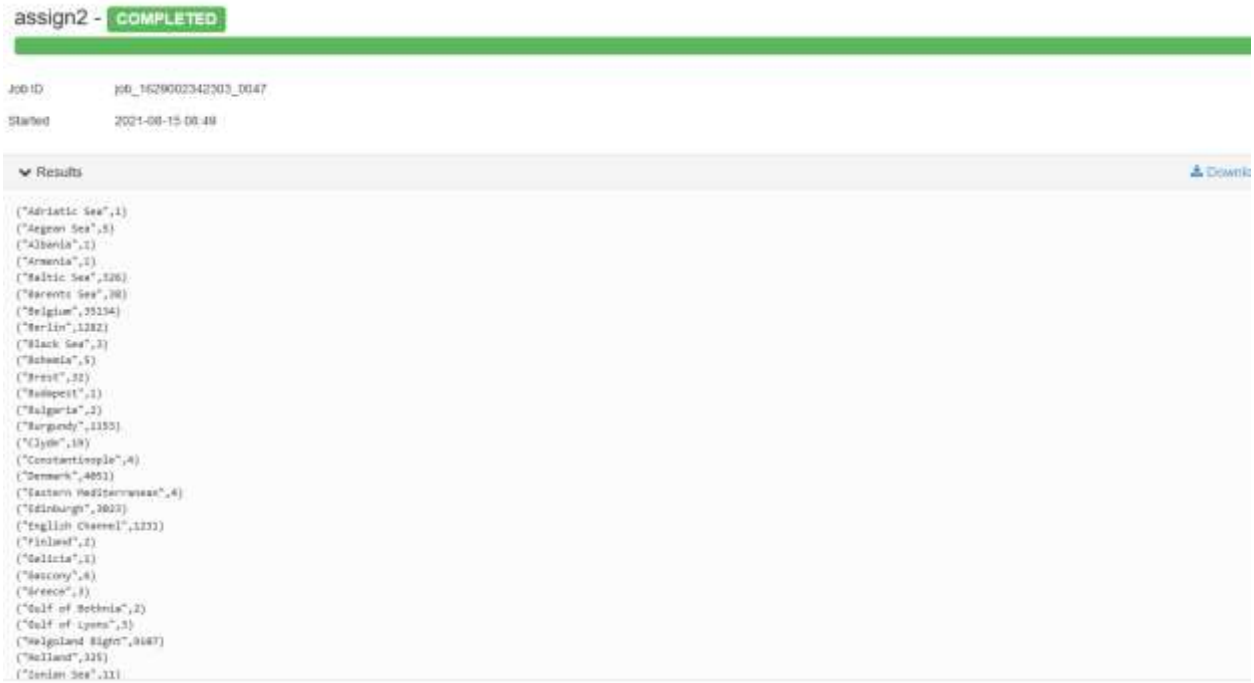
Script

```
ordersCSV = LOAD '/user/maria_dev/orders.csv' USING PigStorage(',')AS
    (game_id:chararray,
    unit_id:chararray,
    unit_order:chararray,
    location:chararray,
    target:chararray,
    target_dest:chararray,
    success:chararray,
    reason:chararray,
    turn_num:chararray);

filter_data = FILTER ordersCSV BY (target == '"Holland"');
grp_by_target = GROUP filter_data BY location;
count_holland = FOREACH grp_by_target GENERATE group as location, COUNT($1);
order_result = ORDER count_holland BY location;

DUMP order_result;
```

First, I filter the data, so that the only data that is left is the rows with the target "Holland". After having filtered the data, I grouped the data by the location. This would result in a dataset with the following structure: {location: {rows}, location2: {rows}, etc.}

After having grouped the data, I counted the times Holland was the target from the location. To close it out I ordered the data by the location to get a list on alphabetical order.

Result



Results

```
("Ionian Sea",11)
("Kiel",44658)
("Liverpool",90)
("Livonia",2)
("London",3519)
("Marseilles",2)
("Mid-Atlantic Ocean",30)
("Moscow",2)
("Munich",2297)
("Naples",1)
("North Africa",1)
("North Sea",16250)
("Norway",740)
("Norwegian Sea",233)
("Paris",10)
("Picardy",501)
("Portugal",5)
("Prussia",8)
("Rome",3)
("Ruhr",22142)
("Rumania",2)
("Serbia",4)
("Silesia",4)
("Skagerrack",164)
("Smyrna",1)
("Spain (South Coast)",1)
("Spain",5)
("St. Petersburg (North Coast)",10)
("St. Petersburg",24)
```

```
("St. Petersburg",24)
("Sweden",73)
("Syria",1)
("Tunis",2)
("Tyrolia",4)
("Tyrrhenian Sea",11)
("Venice",2)
("Vienna",2)
("Wales",37)
("Warsaw",1)
("Western Mediterranean",13)
("Yorkshire",2882)
```