**Analysis of traffic in the city of Valencia and its relationship with air pollution II. MapReduce, Hadoop y AWS**

**1. Make the following queries**

* 1. **(1 point).** Write a mapper function to read, directly from the MongoDB database, for each traffic document, the station ID and the traffic value for each date (one line for each hour of the month for each of the sensors). Write a reducer function to calculate, for each sensor, the average traffic value for the month; write to the output file first the value and then the sensor ID sensor so that it can be sorted using the Linux pipe function and sort command. The result should be the same as query 2.2 of the MongoDB part of the project.

*Put your software code for the Mapper function here*

*Put your software code for the Reducer function here*

*Copy and paste the terminal window with the command and results here*

* 1. **(1 point).** Use the -appendToFile function to upload all the original traffic files to HDFS as a single file. Use the MRJob class to write a code that performs the map and reduce operation required to obtain the same result as in the previous query. The solution must be written in HDFS. From HDFS show the first 20 lines of the solution.

*Copy-paste the command from HDFS from the terminal window to upload the files*

*Put the MapReduce function code here*

*Copy-paste the terminal window with the code execution command*

*Copy and paste the terminal window with the solution*

* 1. **(1 point).** Write a Python program to generate a file from the MongoDB database containing the traffic ID, date, and value for each traffic sensor reading. Upload this file to HDFS (THIS WILL BE THE FILE REFERENCED IN THE REST OF THE QUERIES). Use the MRJob class to write the code to perform the mapping and reduction operation on the previous file saved in HDFS necessary to obtain, for each traffic sensor, the maximum and minimum temperature values, including the exact date (including day, and hour) when each of these values ​​occurred. The output should be via the terminal and sorted by sensor, and the maximum and minimum results should be displayed at the same time. As an example, to check the solution, for sensor A10, the lowest value, 0, occurs on May 13 at 3:03 (in this case it is the first zero found in the entire series) and the highest occurs on May 3 at 2:00 p.m., with a value of 3210.

*Put the function code here to upload the file to HDFS*

*Put the MapReduce function code here*

*Copy-paste the terminal window with the code execution command*

*Copy and paste the terminal window with the solution*

* 1. **(1 point).** Use the MRJob class to write the code that performs the map-reduce operation on the traffic file recorded in HDFS, necessary to obtain, for each traffic sensor and day of the month, the mean traffic value. Write the output to a directory in HDFS, move it to local and display the values ​​for the days of the month for station “A111”. The result should be the same as that of MongoDB query 2.3.

*Put the MapReduce function code here*

*Copy-paste the terminal window with the code execution command*

*Copy-paste the command from HDFS from the terminal window to pass the output file to local*

*Copy and paste the text editor window with the solution*

* 1. **(1 point).** Add a step to the code of the previous query to know, for each traffic sensor, which day has had the highest average traffic value. The output must be by the terminal ordered by sensor. As an example, to check the solution, for sensor A102, the day is 17 and the value is 318.26.

*Put the MapReduce function code here*

*Copy-paste the terminal window with the code execution command*

*Copy and paste the terminal window with the solution*

* 1. **(1 point).** Write a Python code to read directly from the MongoDB database, for each weather station document, the station ID, the date and the NO2 value for each date (one line for each hour of the month for each of the stations). In the same code upload this file to HDFS. From HDFS read the first lines of the generated file.

*Put the MapReduce function code here*

*Copy and paste the terminal window with the file reading from HDFS*

* 1. **(1 point).** Using the MRJob class, write the code similar to that in query 1.5 to find out, for each station, which day had the highest average NO2 value, sorting the output by weather station. As an example, to check the solution, for the Boulevard station, the day is 5 and the NO2 value is 21.875.

*Put the MapReduce function code here*

*Copy-paste the terminal window with the code execution command*

*Copy and paste the terminal window with the solution*

* 1. **(1 point).** Use the MRJob class to write the code that reads the traffic file stored in HDFS and generates, for each day of the week, the average traffic value considering all the traffic sensors. Obtain the result in the terminal by sorting it by day of the week. As an example, to check the solution, on Monday (day 0) the result is 812.769.

*Put the MapReduce function code here*

*Copy-paste the terminal window with the code execution command*

*Copy and paste the terminal window with the solution*

* 1. **(1 point).** Get the same result as the previous query, but for the weather stations file recorded in HDFS. As an example, to check the solution, on Monday (day 0) the result is 13.78.

*Put the MapReduce function code here*

*Copy-paste the terminal window with the code execution command*

*Copy and paste the terminal window with the solution*

* 1. **(1 punto).** Realizar la consulta anterior subiendo el fichero de estaciones a S3 usando la librería de Python *BOTO3*, generando un clúster en EMR dentro de Amazon AWS. Guardar en resultado en S3, descargarlo y mostrar el resultado. Cuando acabes recuerda revisar que el clúster generado se ha borrado y elimina los ficheros de S3 para que no sigas incurriendo en gastos.

*Put your software code here to upload to S3 using BOTO3*

*Make a copy-paste of the terminal window with the code execution order and the end of the execution which should show that it has been carried out correctly.*

*Copy and paste the text editor window with the solution*

**Conclusions (Evaluation of the responsibility and decision-making skills)**

Conclusions must be presented regarding the data worked with (precision, granularity - spatial and temporal resolution -, confusion, value, validity for the objectives pursued, etc.), the platform or framework used for its storage and/or analysis both locally and in the cloud (ease of use, generation of results, etc.), the results obtained (observed correlations, consequences, predictions, groupings, etc.) and the bibliography consulted by the student.

**Bibliography (Evaluation of the responsibility and decision-making competence)**

The corresponding bibliography must be presented, which must contain all the searches carried out by the student, both of a theoretical nature in books and articles and of a technical nature (searching the Internet for solutions for the correct installation of the tools, for the generation of code - e.g. in blogs, the Stack Overflow platform, free information from different courses, etc.). All these references must be adequately referenced in the text of the project so that there is no bibliography without its corresponding reference in the text or in the developed code. Finally, in the conclusions section there must be a part on the bibliography used, where the student must consider the importance, reliability and ease of use or implementation of the different sources used.