CS-562 Programming Assignment 2

Deadline: March 29th 2024

Create a folder including all the .scala files you used and the (.pdf or .docx) report for the answers. Send an email to hy562@csd.uoc.gr (**not the mailing list!**) with subject:

Assign2_StudentIdNumber.

Exercise 0 – Setting up the Environment

Dataset – Wikimedia Projects

The Wikimedia Foundation supports hundreds of thousands of people around the world in creating the largest free knowledge projects in history. The work of volunteers helps millions of people around the globe discover information, contribute knowledge, and share it with others no matter their bandwidth.

In this assignment you are going to explore the page views of Wikimedia projects. Download the page view statistics generated between 0-1am on Jan 1, 2016 from here.

Each line, delimited by a white space, contains the statistics for one Wikimedia page. The schema looks as follows:

Field	**Meaning**
Project code	The project identifier for each page.
Page title	A string containing the title of the page.
Page hits	Number of requests on the specific hour.
Page size	Size of the page.

Spark Framework – Initialize

Launch the spark shell and then create an RDD named pagecounts from the input file (the file must be copied on the same directory as the spark-shell).

A) Libraries need to be imported

- *import org.apache.spark;*
- *import org.apache.spark.rdd.RDD;*
- *import org.apache.spark.storage.StorageLevel._;*
- *import org.apache.spark.sql._;*

- *import org.apache.spark.sql.SparkSession;*

B) Create a new Spark Session

- *val spark = SparkSession.builder().getOrCreate()*

C) Load Dataset

- *val pagecounts = sc.textFile("directory_to/pagecounts-20160101-000000_parsed.out")*

Exercise 1 – Explore the Web Logs with Spark (40%)

First convert the **pagecounts** from RDD[String] into RDD[Log] using the following guideline:

1. create a case class called Log using the four field names of the dataset.
2. create a function that takes a string, split it by white space and converts it into a log object.
3. create a function that takes an RDD[String] and returns an RDD[Log] using your convert function via the built in map function.

In the rest sections of this exercise you have to make use of the RDD[Log] that you have created. For each of the questions below implement a Scala function that takes as input an RDD[Log] and prints requested values. You must also include all of those results in your report.

Question 1 (3 points)

Retrieve the first k records and beautify.

Use the take() operation of an RDD to get the first k records, with k = 15.The take() operation returns an array and Scala simply prints the array with each element separated by a comma. This is not easy to read. Make the output prettier by traversing the array to print each record on its own line.

Question 2 (3 points)

Determine the number of records the dataset has in total.

Question 3 (4 points)

Compute the min, max, and average page size.

Hint: Use Map and Reduce/ReducebyKey functions provided by the RDD api. See the following links

- https://spark.apache.org/examples.html
- https://spark.apache.org/docs/2.0.1/api/java/org/apache/spark/rdd/RDD.html

Question 4 (4 points)

Determine the record with the largest page size. If multiple records have the same size, list all of them.

Question 5 (6 points)

Determine the record with the largest page size again. But now, pick the most popular.

Question 6 (4 points)

Determine the record with the largest page title. If multiple titles have the same length, list all of them.

Question 7 (6 points)

Use the results of question 3, and create a new RDD with the records that have greater page size from the average.

Hint: use the function **filter** provided by the RDD api. See the following link

- https://spark.apache.org/docs/2.1.1/programming-guide.html

Question 8 (3 points)

Compute the total number of pageviews for each project (as the schema shows, the first field of each record contains the project code).

Question 9 (5 points)

Report the 10 most popular pageviews of all projects, sorted by the total number of hits.

Question 10 (5 points)

Determine the number of page titles that start with the article "The". How many of those page titles are not part of the English project (Pages that are part of the English project have "en" as first field)?

Question 11 (5 points)

Determine the percentage of pages that have only received a single page view in this one hour of log data.

Question 12 (6 points)

Determine the number of unique terms appearing in the page titles. Note that in page titles, terms are delimited by "_" instead of a white space. You can use any number of normalization steps (e.g. lowercasing, removal of non-alphanumeric characters).

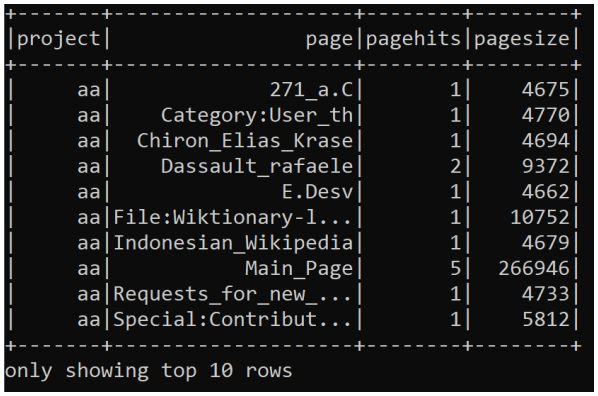
Question 13 (6 points)

Determine the most frequently occurring page title term in this dataset.

Exercise 2 – Explore the Web Logs with Spark SQL (40%)

First convert the **pagecounts** from RDD[String] into DataFrame using the toDF function with appropriate arguments similarly to the following example here.

Hint: You may need to transform RDD[String] into RDD[Log] and then DataFrame. Your resulted DataFrame (DF) should look similar to the following figure:



```
+-------+--------------------+--------+--------+
|project|                page|pagehits|pagesize|
+-------+--------------------+--------+--------+
|     aa|             271_a.C|       1|    4675|
|     aa|    Category:User_th|       1|    4770|
|     aa|  Chiron_Elias_Krase|       1|    4694|
|     aa|    Dassault_rafaele|       2|    9372|
|     aa|              E.Desv|       1|    4662|
|     aa|File:Wiktionary-l...|       1|   10752|
|     aa|Indonesian_Wikipedia|       1|    4679|
|     aa|           Main_Page|       5|  266946|
|     aa|Requests_for_new_...|       1|    4733|
|     aa|Special:Contribut...|       1|    5812|
+-------+--------------------+--------+--------+
only showing top 10 rows
```

Figure 1 – The first n = 10 rows of the DataFrame using the show(n) built in function of the DataFrame api.

Next you must use your DF to answer again to the questions 3, 5, 7, 12, 13 of Ex.1, but this time by **running SQL queries programmatically**, as shown in the following tutorial here. You must also include all of those results (with the table format to be visible) in your report.

Hint: From the DF api you have to use the following functions,

- *createOrReplaceTempView*
- *sql*
- *show()*

Exercise 3 – Spark Pseudo-distributed Execution (20%)

On this exercise you are going to experiment with the pseudo-distributed execution on Spark. You have to change the default configurations in order to emulate a local cluster environment with more than one nodes. Follow the examples of the given link to setup the cluster settings: here.

Configure the number of slaves and memory/cores that you think is more suitable for your PC, and create two simple topologies:

- Small cluster: with small number of workers, number of cores and slaves. (**20 points**)
- Big cluster: with the maximum computation power that you can give. (**20 points**)

For each setup mentioned above, run again questions 3, 5, 6, 7, 12, 13 of Ex.1, and compare the **execution time** on the two emulated clusters. You are asked not only to report the execution times for the two topologies, but **compare and explain** the two setup configurations and answer the following questions:

1. How master and slaves interact with the Spark environment?
2. Does the number of slaves affect the execution time?
3. Suppose you have input files that are greater than 10 GB and you are given a machine (like a google cloud) with 16 virtual CPUs and 60 GB of memory. You have two options:
   a) 1 master and 1 slave with the maximum memory/computation power.
   b) 1 master and **n** slaves with the maximum computation power and memory divided across slaves.

   What is the range of **n** on this setup? What would you choose?

How would you compare the options above with your emulated topologies? Is it better or worse?

Have Fun!