CS585/DS503 Project 1

Total Points: 155

Release Date: **09/23/2022**

Due Date: 10/03/2022 (11:59PM)

Teams: Project to be done in teams of two.

Short Description

In this project, you will write map-reduce jobs in Java language, and run them on a Hadoop system.

Detailed Description

You are asked to perform three activities in this project, (1) Create datasets, (2) upload the datasets into Hadoop HDFS, (3) Query the data by writing map-reduce Java code

1-Creating Datasets [20 Points]

Write a java program that creates two datasets (two files), ***Customers*** and ***Transactions***. Each line in the Customers file represents one customer, and each line in the Transactions file represents one transaction. The attributes within each line are comma separated.

The ***Customers*** dataset should have the following attributes for each customer:

ID: unique sequential number (integer) from 1 to 50,000 (that is the file will have 50,000 line)

Name: random sequence of characters of length between 10 and 20 ***(do not include commas)***

Age: random number (integer) between 10 to 70

Gender: string that is either “male” or “female”

CountryCode: random number (integer) between 1 and 10

Salary: random number (float) between 100 and 10000

The ***Transactions*** dataset should have the following attributes for each transaction:

TransID: unique sequential number (integer) from 1 to 5,000,000 (the file has 5M transactions) CustID: References one of the customer IDs, i.e., from 1 to 50,000 (on Avg. a customer has 100 trans.)

TransTotal: random number (float) between 10 and 1000

TransNumItems: random number (integer) between 1 and 10

TransDesc: random text of characters of length between 20 and 50 ***(do not include commas)***

Note: The column names will NOT be stored in the file. Only the comma separated values. Form the order of the columns; you will know each column represents what.

2-Uploading Data into Hadoop [5 Points]

Use hadoop file system commands (e.g., put) to upload the files you created to Hadoop cluster. To learn about the file system commands check this link:

https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html

Note: It is good to check your files and see how the files are divided into blocks and each block is replicated.

3-Writing MapReduce Jobs [130 Points]

You will write Java programs to query the data in Hadoop. Before writing your code you should perfectly understand the “WordCount” example (it is like the “Hello World...” example in Java). You can find its code online

Notes:

- You should decide whether each query is a map-only job or a map-reduce job, and write your code based on that. A given query may require more than a single map-reduce job to be done.
- You can always check the query output file from the HDFS website and see its content.
- You can test your code on a small file first to make sure it is working correctly before running it on the large datasets.

Hint: It is important two know how Hadoop reads and writes integers, floats, and text fields. Check IntWritable, FloatWritable, and Text classes to know which one to use and when.

3.1) Query 1 [20 Points]

Write a job(s) that reports the customers whose Age between 20 and 50 (inclusive).

3.2) Query 2 [20 Points]

Write a job(s) that reports for every customer, the number of transactions that customer did and the total sum of these transactions. The output file should have one line for each customer containing:

CustomerID, CustomerName, NumTransactions, TotalSum

You are required to use a Combiner in this query.

3.3) Query 3 [30 Points]
Write a job(s) that joins the Customers and Transactions datasets (based on the customer ID) and reports for each customer the following info:

CustomerID, Name, Salary, NumOfTransactions, TotalSum, MinItems

Where *NumOfTransactions* is the total number of transactions done by the customer, *TotalSum* is the sum of field "TransTotal" for that customer, and *MinItems* is the minimum number of items in transactions done by the customer.

3.4) Query 4 [30 Points]

Write a job(s) that reports for every country code, the number of customers having this code as well as the min and max of *TransTotal* fields for the transactions done by those customers. The output file should have one line for each country code containing:

CountryCode, NumberOfCustomers, MinTransTotal, MaxTransTotal

Hint: *To get the full mark of Query 4, you need to do it in a single map-reduce job. If you did it using two map-reduce jobs, you will lose 8 Points.*

3.5) Query 5 [30 Points]

Assume we want to design an analytics task on the data as follows:

1) The Age attribute is divided into six groups, which are [10, 20), [20, 30), [30, 40), [40, 50), [50, 60), and [60, 70]. The bracket "[" means the lower bound of a range is included, whereas ")" means the upper bound of a range is excluded.

2) Within each of the above age ranges, further division is performed based on the "Gender", i.e., each of the 6 age groups is further divided into two groups.

3) For each group, we need to report the following info:
Age Range, Gender, MinTransTotal, MaxTransTotal, AvgTransTotal

What to Submit

You will submit a single zip file containing the Java programs for ***Creating Data Files, Java code for the MapReduce Queries*** plus a document (.pdf) containing any comments you would like to provide regarding your code.

How to Submit

Use the Canvas system to submit your files.