Big Data Management Project 3

Spring 2017

Team 18 Caitlin Kuhlman, Wenlei Cao

Code Organization:

* Project3
  + Problem1
  + Problem2
    - DataGenerator.java
    - Problem2.scala

**Problem 1. Spark SQL (transaction data processing)**

**To run:**

The file Problem1.scala contains the SparkSQL workflow for problem 1. It first creates a RDD from the local transactions.csv file. The path is hardcoded in the file, so you need to change the path to your test file location.

To run:

./spark-shell

:load <PATH TO Problem1.scala>

Example:

If your file located at home/tmp/transactions.csv

You need to change

val T = spark.sparkContext.textFile("file:///home/mqp/Documents/transactions.csv")

to

val T = spark.sparkContext.textFile("file:/// home/tmp/transactions.csv ")

Then you can run the rest of code

**Explanation:**

First, create a RDD from transaction.csv.

Next, in order to be able to use Spark SQL query, we add schema (column name) to the RDD and transform it into a dataframe. Function, createOrReplaceTempView, is used to create a view for Spark sql query.

Last, we apply SQL query per question’s requirement for filter, aggregation, join.

**Problem 2. Grid cell of high Relative Density index**

**Step 1: Generate dataset**

DataGenerate.java generates the point data. There is one record for each point with the format: “x,y”.

It writes to a file called “points.csv”

**Step 2: Report the TOP 50 grid cells w.r.t Relative-Density Index**

The file Problem2.scala contains the workflow for the next 2 steps. The path to the data file is hard-coded in the file, so you need to change it to the path to the point file. You can execute the code by running:

./spark-shell

:load <PATH TO Project2.scala

The first output is the top 50 Relative density index (RDI) points ordered by RDI value descending.

Format: (cell\_id, RDI)

**Step 3: Report the TOP 50 grid cells neighbor’s RDI**

The second output is the top 50 grid cells again. This time each cell has a list of their neighboring cells and each neighbor cell’s RDI.

Format: (cell\_id, RDI, neighbor1\_id | neighbor1\_RDI, neighbor2\_id | neighbor2\_RDI, … neighborN\_id | neighborN\_RDI)