CS6350
Big data Management Analytics and Management
Spring 2016
Homework 2
Submission Deadline: 10th March, 2016

In this homework, you will be using Apache spark/scala to analyze social network data.
Note: Q1 is modified to output all recommendations in any order.

Q1

Write a Scala/Spark program that implements a simple "People You Might Know" social network friendship recommendation algorithm. The key idea is that if two people have a lot of mutual friends, then the system should recommend that they connect with each other.
Input:
Input files
1. soc-LiveJournal1Adj.txt located in /socNetData/networkdata in hdfs on cs6360 cluster

The input contains the adjacency list and has multiple lines in the following format:
<User><TAB><Friends>
2. userdata.txt located in /socNetData/userdata in hdfs on cs6360 cluster
The userdata.txt contains dummy data which consist of
column1 : userid
column2 : firstname
column3 : lastname
column4 : address
column5: city
column6 :state
column7 : zipcode
column8 :country
column9 :username
column10 : date of birth.

Here, <User> is a unique integer ID corresponding to a unique user and <Friends> is a comma-separated list of unique IDs corresponding to the friends of the user with the unique ID <User>. Note that the friendships are mutual (i.e., edges are undirected): if A is friend with B then B is also friend with A. The data provided is consistent with that rule as there is an explicit entry for each side of each edge.

Algorithm: *Let us use a simple algorithm such that, for each user U, the algorithm* ***recommends all users who are not already friends with U****. Note that you are to use only the second level friendship to find mutual friends.*

Output: The output should contain one line per user in the following format:
<User><TAB><Recommendations>

where <User> is a unique ID corresponding to a user and <Recommendations> is a comma-separated list of unique IDs corresponding to the algorithm's recommendation of people that <User> might know.
If a user has no friends, you can provide an empty list of recommendations. If there are multiple users with the same number of mutual friends, ties are broken by ordering them in a numerically ascending order of their user IDs.

What to submit
(i) Submit the source code via the elearning website.
(ii) Output the recommendations for the users with following user IDs:
924, 8941, 8942, 9019, 9020, 9021, 9022, 9990, 9992, 9993.

Q2.
Using spark/scala Please answer this question by using dataset from Q1.
Given any two Users as input, output the list of the user id of their mutual friends.

Output format:
UserA, UserB list userid of their mutual Friends.

Q3.
Using spark/scala, Given any two Users as input, output the list of the names and the zipcode of their mutual friends.
Note: use the userdata.txt to get the extra user information.
Output format:
UserA id, UserB id, list of [names:zipcodes] of their mutual Friends.

Sample Output:
```
1234     4312       [John:75075, Jane : 75252, Ted:45045]
```

Q4. Using spark/scala
Step 1: Calculate the average age of the direct friends of each user.
Step 2: Sort the users by the calculated average age from step 1 in descending order.
Step 3. Output the top 20 users from step 2 with their address and the calculated average age.

Sample output.

User A, 1000 Anderson blvd, Dallas, TX, average age of direct friends.