CS6350
Big data Management Analytics and Management
Spring 2022
Homework 1
Submission Deadline: February 27th, 2022

In this homework, you will be using hadoop/mapreduce to analyze social network data.

Q1: Write a MapReduce program in Hadoop that implements a simple **"Mutual/Common friend list of two friends"**. The key idea is that if two people are friend then they have a lot of mutual/common friends. This program will find the common/mutual friend list for them.

For example:
Alice's friends are Bob, Sam, Sara, Nancy
Bob's friends are Alice, Sam, Clara, Nancy
Sara's friends are Alice, Sam, Clara, Nancy

As Alice and Bob are friend and so, their mutual friend list is [Sam, Nancy]
As Sara and Bob are not friend and so, their mutual friend list is empty. (In this case you may exclude them from your output).

Input:
1. mutual.txt
The input contains the adjacency list and has multiple lines in the following format:
<User><TAB><Friends>
Hence, each line represents a particular user's friend list separated by comma.

2. userdata.txt
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. So, when you make the pair, always consider (A, B) or (B, A) for user A and B but not both.

Output:
The output should contain one line per pair in the following format:
<User_A>, <User_B><TAB><Mutual/Common Friend List>

where <User_A> & <User_B> are unique IDs corresponding to a user A and B (A and B are friend). < Mutual/Common Friend List > is a comma-separated list of unique IDs corresponding to mutual friend list of User A and B.

Specifically generate/print the Mutual/Common Friend list for the following user pairs: (0,1), (20, 28193), (1, 29826), (6222, 19272), (28041, 28056)

Q2: Please **use in-memory join at the Mapper** to answer the following question.
Given any two Users (they are friend) as input, output a list containing the dates of birth of all their mutual friends and the number of those friends who were born after 1995. Note that the userdata.txt will be used to get the extra user information and cached/replicated at each mapper.

Output format:
<User_A>, <User_B><TAB>< List of DOBs [$date_1$, $date_2$, ... $date_n$], Number of friends born after 1995>

Sample output:
1234 4312 [10/12/1994, 01/03/1996, 11/11/1995], 1

Q3: Please use **in-memory join at the Reducer** to answer the following question.
For each user print User ID and minimum age of direct friends of this user.

Output format:
<User><TAB><Minimum age of direct friends>

Sample output:
1234 60

Q4: Write a program that will construct inverted index in the following ways.

The **map** function parses each line in an input file, userdata.txt, and emits a sequence of <word, line number> pairs. The **reduce** function accepts all pairs for a given word, sorts the corresponding line numbers, and emits a <word, list(line numbers)> pair. The set of all the output pairs forms a simple inverted index.

Q5: Please **use a Combiner** to answer the following question.
Find all those words(s) whose number of occurrences is the maximum among all the words in the inverted index. Note that you need to use the same output from Q4.

Output Format:
<Word><TAB><Number of occurrences>

What to submit
(i) Submit the source code via the eLearning website. (ii) Submit the output file for each question.