**CSEE5590/CS490: Big Data Programming**

**LAB ASSIGNMENT #1**

**Lab Submission before October/6th/2019 (M) 11:59PM.**

Submit a lab report including screenshots and source code & data to GitHub. Post your GitHub URL through Lab 1 form at <https://docs.google.com/forms/d/e/1FAIpQLSdyiUtTiKlvBnXkebNTzIXcKQbJQ1Lifw_mSDaU6MiQ15J5KA/viewform?usp=sf_link>

1. **Hadoop MapReduce Algorithm**

Implement MapReduce algorithm for finding Facebook common friends problem and run the MapReduce job on Apache Hadoop. Show your implementation through map-reduce diagram as shown in Lesson Plan 2:

<https://umkc.box.com/s/jhpgd8yeerlkurqwjtgp5ej5izpt47lk>

Write a report including your algorithm and result screenshots.

**Finding Facebook common friends**: Facebook has a list of friends (note that friends are a bi-directional thing on Facebook. If I'm your friend, you're mine). They also have lots of disk space and they serve hundreds of millions of requests every day. They've decided to pre-compute calculations when they can to reduce the processing time of requests. One common processing request is the "You and Joe have 230 friends in common" feature. When you visit someone's profile, you see a list of friends that you have in common. We're going to use MapReduce so that we can calculate everyone's common friends once a day and store those results. Later on it's just a quick lookup. We've got lots of disk, it's cheap.

**Example (What is the Key/Value Pair?)**

Assume the friends are stored as Person -> [List of Friends], our friends list is then:

A -> B C D

B -> A C D E

C -> A B D E

D -> A B C E

E -> B C D

The result after reduction is:

(A B) -> (C D)

(A C) -> (B D)

(A D) -> (B C)

(B C) -> (A D E)

(B D) -> (A C E)

(B E) -> (C D)

(C D) -> (A B E)

(C E) -> (B D)

(D E) -> (B C)

When D visits B's profile, we can quickly look up (B D) and see that they have three friends in common, (A C E).

1. **Use Case:**

Implement MapReduce algorithm to perform analysis on Youtube dataset.

Using dataset perform some Analysis and draw out some insights like what are the top 10 rated videos on YouTube, who uploaded the most number of videos.

Dataset: <https://umkc.box.com/s/69u5uxkf8v703cqco7tik453izhfg2gu>

**Column 1:** Video id of 11 characters.  
**Column 2:** uploader of the video  
**Column 3:** Interval between the day of establishment of Youtube and the date of uploading of the video.  
**Column 4:** Category of the video.  
**Column 5:** Length of the video.  
**Column 6:** Number of views for the video.  
**Column 7:** Rating on the video.  
**Column 8:** Number of ratings given for the video  
**Column 9:** Number of comments done on the videos.  
**Column 10:** Related video ids with the uploaded video.

**Problem Statement 1:**

Find out what are the top 5 categories with maximum number of videos uploaded.

**Problem Statement 2:**

Find the top 10 rated videos on youtube.

Write a report including your algorithm and result screenshots.

1. **Consider one of the following use cases,**
2. Zomato Restaurants Data

<https://www.kaggle.com/shrutimehta/zomato-restaurants-data>

1. Super Heros Dataset

<https://www.kaggle.com/claudiodavi/superhero-set/data>

1. Google Job Skills Dataset

<https://www.kaggle.com/niyamatalmass/google-job-skills/data>

1. Seinfeld Chronicles Dataset

<https://www.kaggle.com/thec03u5/seinfeld-chronicles/data>

***HIVE USECASE***

1. Create a Hive Table including Complex Data Types
2. Use built-in functions in your queries
3. Perform 10 intuitive questions in Dataset (e.g.: pattern recognition, topic discussion, most important terms, etc.). Use your innovation to think out of box.

***SOLR USECASE***

* 1. Create a Solr Collection including our own Field Types
  2. Perform 10 intuitive questions in Dataset (e.g.: pattern recognition, topic discussion, most important terms, etc.). Use your innovation to think out of box. Implement at least 5 nested queries among the 10.
  3. Record the time execution for the queries.

Write a report including your algorithm and result screenshots.

**Submission Guidelines (for all students):**

1. Submit your source code and documentation to GitHub and represent the work through wiki page properly (submit your screenshots as well. The screenshot should have both the code and the output)
2. Comment your code appropriately.
3. Submit a brief demo video 2-3 min showing your assignment with a voice over explaining your work through the Submission Link.
4. Submit source\_code and report to Canvas. Your similarity score should be less than 15%
5. Use the following Google link to submit your assignment

(LabSubmissionLink#): <https://docs.google.com/forms/d/e/1FAIpQLSdyiUtTiKlvBnXkebNTzIXcKQbJQ1Lifw_mSDaU6MiQ15J5KA/viewform?usp=sf_link>

***Cheating, plagiarism, disruptive behavior and other forms of unacceptable conduct are subject to strong sanctions in accordance with university policy. See detailed description of university policy at the following URL:*** [*https://catalog.umkc.edu/special-notices/academic-honesty/*](https://catalog.umkc.edu/special-notices/academic-honesty/)