Homework 2 – Programming Assignment (MapReduce, Pig, Hive) (200 points)

Submission Deadline: Tuesday, March 24, 2020 (@4:00pm)

HW2 Solution Drive Link:
https://drive.google.com/open?id=1ekrW0jnkGs46uS5Um3ENRqF9f2ChcR9s

Problem 1 (Use Pig and MapReduce) (80 points)

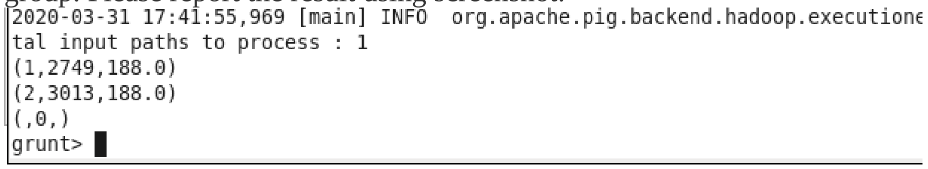
The file **p1.csv** is a sample patient dataset with the following patient characteristics: pid, gender, race, height, weight, asthma, and hypertension. Here "pid" is the unique identifier for a patient. The units for height and weight are centimeters and kilograms, respectively. The integer codings for gender, race, asthma, and hypertension are given as follows:

Column	**Codings**
gender	1: Male, 2: Female
race	1: White, 2: Black, 3: Other
asthma	0: No, 1: Yes, 8: Don't Know/Not sure/Not applicable
hypertension	0: No, 1: Yes, 8: Don't Know/Not sure/Not applicable

Requirements:
(R1.1) Launch Pig in shell mode. Use LOAD operator to load the dataset **p1.csv** in HDFS into Pig using the following schema (pid: integer, gender: integer, race: integer, height: double, weight: double, asthma: integer, hypertension: integer).
>>>> Pthird = load 'p1.csv' using PigStorage(',') as (pid: int, gender: int, race: int, height: double, weight: double, asthma: int, hypertension: int);

(R1.2) Use GROUP operator to group patient by gender.
>>>> Groupss = group Pthird by gender;

(R1.3) Based on the grouped result obtained in (R1.2), use FOREACH operator to return the number of patients within each gender group and the maximum height of patients in the gender group. Please report the result using screenshot.


```
2020-03-31 17:41:55,969 [main] INFO  org.apache.pig.backend.hadoop.executione
tal input paths to process : 1
(1,2749,188.0)
(2,3013,188.0)
(,0,)
grunt> 
```

>>>> Outputss = foreach Groupss GENERATE FLATTEN(group), COUNT(Pthird.height), MAX(Pthird.height);
Submission details for (R1.1-R1.3): Please insert your pig statements below each of the requirements.

(**R1.4**) Write a MapReduce algorithm to directly compute the number of patients within each gender group and the maximum height of patients in the gender group.

```
  public static class Map extends Mapper<LongWritable, Text, Text,
DoubleWritable> {
 // private final static IntWritable one = new IntWritable(1);
  private long numRecords = 0;
  private static final Pattern WORD_BOUNDARY = Pattern.compile(",");

  public void map(LongWritable offset, Text lineText, Context context)
      throws IOException, InterruptedException {

    String[] line = WORD_BOUNDARY.split(lineText.toString());
String s2 = line[1];
String s4 = line[3];

      if (!(s2.equals("gender"))){
          if (s2 != null && Integer.parseInt(s2) == 1){
                       if (s4.equals("NULL") || s4.equals(null)){
                       context.write(new Text(s2),new DoubleWritable(0.0));
   } else {
                       double height = Double.parseDouble(s4);
                       context.write(new Text(s2),new DoubleWritable(height));
                 }
}
          if (s2 != null && Integer.parseInt(s2) == 2){
                       if (s4.equals("NULL") || s4.equals(null)){
                       context.write(new Text(s2),new DoubleWritable(0.0));
   } else {
                       double height = Double.parseDouble(s4);
                       context.write(new Text(s2),new DoubleWritable(height));
                 }
}
        }
  }
}
public static class Reduce extends Reducer<Text,DoubleWritable, IntWritable,
DoubleWritable> {
    @Override
    public void reduce(Text gender, Iterable<DoubleWritable> Vals, Context
context)
        throws IOException, InterruptedException {
      int totalGender = 0;
      double maxVal = 0.0;

      for (DoubleWritable val : Vals) {
         totalGender = totalGender + 1;
        if (val.get() > maxVal){
         maxVal = val.get();
         }
      }
      context.write(new IntWritable(totalGender), new DoubleWritable
(maxVal));
    }
  }
}
```

Submission details for (R1.4): Please insert your MapReduce algorithm below R1.4 and submit the source code in a separate file.

Problem 2 (Use Pig) (60 points)

The file **p2.txt** contains a fraction of the Google books n-grams dataset stored in a TAB(\t) separated format. An ‘n-gram’ is a phrase with *n* words. Each line in the file has the following format:

n-gram TAB year TAB occurrences TAB books NEWLINE

An example for 3-grams would be:

Each bond issued	*1984*	*5*	*3*
Each bond issued	*1993*	*8*	*5*

This tells us that in 1984, the words "Each bond issued" occurred 5 times in 3 distinct books. In 1993, “Each bond issued” occurred 8 times in 5 distinct books.

Requirements:

(R2.1)

For each unique n-gram, compute its average number of appearances per book (round the average number to three decimal places). For the above example, the result will be the following:

Each bond issued *(5+8)/(3+5)=1.625*

```
p2 = Load 'p2.txt' USING PigStorage('\t') AS (ngram:chararray, year:int,
occurrences:double, book:double);

P2_Grouped = GROUP p2 by ngram;

P2_Averaged = FOREACH P2_Grouped { sum_occur = SUM(p2.occurrences);
sum_books = SUM(p2.book); division = (double)(sum_occur/sum_books);
GENERATE group, division as sumoccurbooks;}

P2_Rounded = FOREACH P2_Averaged { Rounded =
(double)ROUND(sumoccurbooks*1000.0)/1000.0; GENERATE group, Rounded as
Final;}
STORE P2_Rounded INTO 'HW2/HW2_Q2_1.txt' using PigStorage('\t');
```

(R2.2)

Output the 15 n-grams with the highest average number of appearances per book along with their corresponding average sorted in descending order. If multiple n-grams share the same average, sort them in alphabetical order.

```
p2 = Load 'p2.txt' USING PigStorage('\t') AS (ngram:chararray, year:int,
occurrences:double, books:double);

P2_Grouped= GROUP p2 by ngram;

P2_Averaged= FOREACH P2_Grouped{ sum_occur = SUM(p2.occurrences);
sum_books = SUM(p2.books); division = (double)(sum_occur/sum_books);
GENERATE group, division as sum_occurbooks;}
```

```
P2_Rounded = FOREACH P2_Averaged{ Rounded =
(double)ROUND(sum_occurbooks*1000.0)/1000.0; GENERATE group, Rounded as
Final;}

Order_Alphabetically = ORDER P2_Rounded BY group DESC;

Order_Maximum = ORDER Order_Alphabetically BY Final DESC;

P2_Limit = LIMIT Order_Maximum 15;
STORE P2_Limit INTO 'HW2/HW2_Q2_2.txt' using PigStorage('\t');
```

Submission details: Please insert your pig statements below each of the requirements and submit the source pig scripts that are directly executable (e.g., R2.1.pig, R2.2.pig). Please provide a link to download the results.

HW2 Solution Drive Link:

https://drive.google.com/open?id=1ekrW0jnkGs46uS5Um3ENRqF9f2ChcR9s

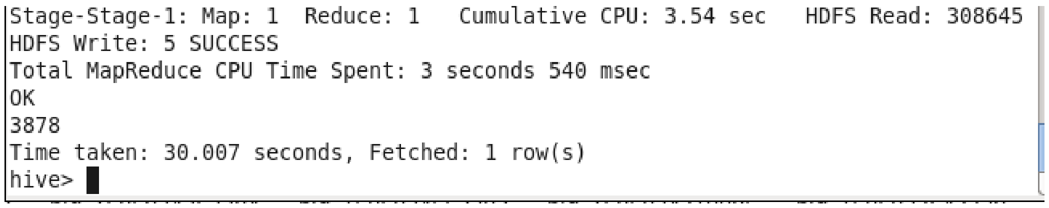
Problem 3 (Use Hive) (60 points)
The file **p3.csv** is a sample patient dataset with the following patient characteristics: pid, gender, race, height, weight, asthma, hypertension, and year. Here "pid" is the unique identifier for a patient, and "year" is the year of the patient visit. The units for height and weight are centimeters and kilograms, respectively. The integer codings for gender, race, asthma, and hypertension are given as follows:

Column	**Codings**
gender	1: Male, 2: Female
race	1: White, 2: Black, 3: Other, 7: Unknown
asthma	0: No, 1: Yes, 8: Unknown
hypertension	0: No, 1: Yes, 2: Unknown

Requirements:
(**R3.1**) Create a Hive database named "hiveDB". Create a table named "patients" in this database with proper data schema specified. Load the patient dataset **p3.csv** to the "patients" table.

create table patients (
> pid int,
> gender int,
> race int,
> height double,
> weight double,
> asthma int,
> hypertension int,
> year int)
> row format delimited fields terminated by ',';
LOAD DATA LOCAL INPATH 'p3.csv' INTO TABLE patients;

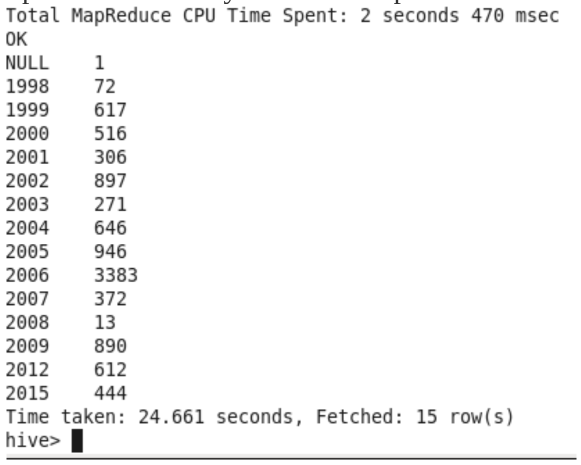
(**R3.2**) Query the number of female patients with hypertension but without asthma. Please report the returned number. Please report the result using screenshot.



```
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.54 sec   HDFS Read: 308645
HDFS Write: 5 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 540 msec
OK
3878
Time taken: 30.007 seconds, Fetched: 1 row(s)
hive>
```

>>>> select * from patients where hypertension=1 AND asthma=0;

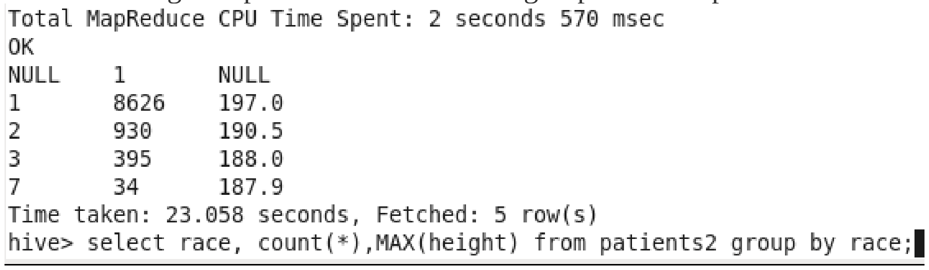
(**R3.3**) Get the number of patients for each year. Please report the result using screenshot.



```
Total MapReduce CPU Time Spent: 2 seconds 470 msec
OK
NULL    1
1998    72
1999    617
2000    516
2001    306
2002    897
2003    271
2004    646
2005    946
2006    3383
2007    372
2008    13
2009    890
2012    612
2015    444
Time taken: 24.661 seconds, Fetched: 15 row(s)
hive>
```

>>>> select year, count(*) from patients group by year;

(**R3.4**) Group patients by race, and get the number of patients within each race group and the maximum height of patients within the race group. Please report the result using screenshot.



```
Total MapReduce CPU Time Spent: 2 seconds 570 msec
OK
NULL    1       NULL
1       8626    197.0
2       930     190.5
3       395     188.0
7       34      187.9
Time taken: 23.058 seconds, Fetched: 5 row(s)
hive> select race, count(*),MAX(height) from patients2 group by race;
```

>>>> select race, count(*),MAX(height) from patients2 group by race;

(**R3.5**) For each patient, output the number of patients who have the same gender and height with the patient. Please report the result in a separate result file (e.g., R3.5.txt).

>>>> insert overwrite local directory 'Output.txt'
>>>> row format delimited
>>>> fields terminated by '\t'
>>>> select gender, height, count(*) from patients group by gender, height;

Submission details: Please insert your Hive statements below each of the requirements.

NOTE: Please e-mail a compressed file containing the required contents to jzhang@cs.uky.edu.

HW2 Solution Drive Link:

https://drive.google.com/open?id=1ekrW0jnkGs46uS5Um3ENRqF9f2ChcR9s