Hadoop Lab Exercises, v6.0

Course: CSci5751-001, Fall 2019

Instructor: Donald Sawyer

Group: Sparklers

Students: Aviral Bhatnagar, Euna Mehnaz Khan, Shreyasi Pal

Table of Contents

Introduction **1**

Exercises Overview 1

Deliverables 2

Exercise 1: Include Carrier Names and Airport Names (40 points) **3**

Exercise 2: Update the Grep MapReduce (25 points) **9**

Exercise 3: Loading HBase using Spark (5 bonus points) **12**

Appendix **13**

5.1 Intermediate Data 13

5.2 Python codes for Hive UDF 13

5.3 Python Script for validating the results of Exercise 2.2 13

5.4 Output of Exercise 2.1 13

Introduction

These exercises will be used to evaluate your learnings from the Hadoop lab. They are not necessarily trivial and will require you to pay attention to your data and think through the problems. As always, ask questions and get help using Slack.

1.1 Exercises Overview

The table below describes each exercise, data required, and the skills needed to complete them.

Exercise	Points	Data	Skills Required/Learned
1: Hive Denorm	40	• Flight 12 months data • Carrier Codes • Airport Codes *(can also be found in Google drive)*	• Hive modeling • Hive/Spark ETL • Denormalization • Data analysis
2: Grep MapReduce	25	• Names.zip	• Designing MR • Python • Code/data verification

3: Spark/HBase	5 bonus	● Flight 12 months data	● Spark, HBase ● Integrating Spark and HBase

1.2 Deliverables

There will be two things you need to turn into Canvas for credit:

1. PDF of this document with the answers and code embedded.
2. MP4 video recording showing your spark/Hive ETL code running and queries being executed with their results. You can use Screen Cast-o-Matic to do the screen recording for free for up to a 15 minute video.

Exercise 1: Include Carrier Names and Airport Names (40 points)

The carriers are only listed by their carrier code and the airports are only listed by their airport code. Download the lookup tables in csv format from the BTS site to do this exercise.

Data	Link
Carrier Code csv	https://www.transtats.bts.gov/Download_Lookup.asp?Lookup=L_CARRIER_HISTORY
Airport Code csv	https://www.transtats.bts.gov/Download_Lookup.asp?Lookup=L_AIRPORT

As you work through the exercise, note down the commands, DDL, and code used to perform the various operations.

For this exercise, you need to do the following:

1. Upload the lookup data to HDFS.
2. Create external or managed Hive tables for the carrier and airport lookup data.
3. Create a managed Hive table using a format other than text (ORC, Parquet, etc.) to store the flight data along with the carrier names (descriptions) and airport names (descriptions). Call this table flight_data_denorm. The table should have only TWO new columns to store the carrier name and airport names (*hint: use a map, array, or struct column to store both the arrival and departure airport name in a single column)*.
4. Populate flight_data_denorm with the flight data using Spark or Hive.
    a. Spark is already available on the Hadoop cluster.
    b. You will want to use the JOIN command in Spark/Hive to join your different relations.
    c. You can select the lookup data from the csv files or the Hive tables.

Now that you have tables available, you need to answer the questions below.

Question	Answer	Query
Which origin airport [name] has the highest average departure delay? (use flight_data_orc2 with a lookup table)	Wendover, UT: Wendover Airport	`select z.description from (select origin, avg(dep_delay) as delay from flight_data_orc2 GROUP BY origin ORDER BY delay DESC LIMIT 1) as y join( select code, description from airport_lookup) as z on z.code=y.origin;`
Which origin airport [name] has the highest average departure delay? (use flight_data_denorm table)	Wendover, UT: Wendover Airport	`select z.AIRPORT_NAMES[0] from ( select origin, avg(dep_delay) as delay from flight_data_denorm GROUP BY ORIGIN ORDER BY delay DESC LIMIT 1)as y join (select AIRPORT_NAMES, origin from flight_data_denorm) as z on z.origin = y.origin ;`
Which carrier [name] had the highest arrival delays on 3/14/2016? (use flight_data_date with a lookup table)	SkyWest Airlines Inc. (2003 -)	`select z.description from (select UNIQUE_CARRIER, max(arr_delay) as delay from flight_data_date where fl_date="2016-03-14" GROUP BY UNIQUE_CARRIER ORDER BY delay DESC LIMIT 1) as y join( select code,`

		description from carrier_lookup) as z on z.code=y.UNIQUE_CARRIER;
Which carrier [name] had the highest arrival delays 3/14/2016? (use flight_data_denorm)	SkyWest Airlines Inc. (2003 -)	select z.carrier_names from (select unique_carrier, max(arr_delay) as delay from flight_data_denorm where fl_date="2016-03-14" GROUP BY unique_carrier ORDER BY delay DESC LIMIT 1)as y join (select distinct unique_carrier, carrier_names from flight_data_denorm) as z on z.unique_carrier = y.unique_carrier ;
What is the total departure delay for flights that took off from an airport with Beaumont in the name? (use flight_data_denorm)	9200.0	select sum(dep_delay) from flight_data_denorm where airport_names[0] LIKE "%Beaumont%" ;
How many flights landed in an airport with "TX" in the name operated by a carrier with "Virgin" in the name? (use flight_data_denorm)	5727	select count(*) from flight_data_denorm where airport_names[1] like "%TX%" and carrier_names like "%Virgin%";
How many distinct carrier descriptions are there? (use flight_data_denorm)	12	select count(distinct carrier_names) from flight_data_denorm ;
What are the top 3 airport names that received the most flights that departed after December 20, 2016, and how many flights were there for each? (use flight_data_denorm))	Los Angeles, CA: Los Angeles International 6479 Atlanta, GA: Hartsfield-Jackson Atlanta International 10439 Denver, CO: Denver International 6680	select distinct z.airport_names[1], y.c from (select dest, count(*) as c from flight_data_denorm where unix_timestamp(fl_date,'yyyy-MM-dd') > unix_timestamp('2016-12-20','yyyy-MM-dd') GROUP BY dest ORDER BY c DESC limit 3) as y join (select dest, airport_names from flight_data_denorm) as z on z.dest = y.dest;
Which airport code in Montana (MT) had the most flights scheduled to arrive, and how many flights were scheduled? (use flight_data_denorm)	BZN 4251	select dest, count(*) as c from flight_data_denorm where airport_names[1] like '%, MT:%' GROUP BY dest ORDER BY c DESC limit 1;
How many rows are in flight_data_denorm?	5617658	select count(*) from flight_data_denorm;

Results:

Paste all your code for setting up and ingesting the data below. Be clear on order and which system (HDFS, Hive, Spark, etc.) they are run in.

1. Upload the lookup data to HDFS:

First, we transfer the codes data to the edge node. For that we create a directory named 'codes_data' on the edge node.

```
edge> mkdir codes_data
```

Then we transfer the L_AIRPORT.csv_ and L_CARRIER_HISTORY.csv_ files to the edge node from our local machine.

```
local machine>scp -P 2222 L_AIRPORT.csv_ root@127.0.0.1:/root/codes_data
local machine>scp -P 2222  L_CARRIER_HISTORY.csv_ root@127.0.0.1:/root/codes_data
```

Next, we copy the files from the edge node to the hdfs. For that, we create two folders called 'airport_codes_data' and 'carrier_codes_data' on the hdfs.

```
edge> hdfs dfs -mkdir /user/root/airport_codes_data
edge> hdfs dfs -mkdir /user/root/carrier_codes_data
```

Then we copy the file L_AIRPORT.csv_ to airport_codes_data folder and L_CARRIER_HISTORY.csv_ to carrier_codes_data folder on hdfs.

```
edge>cd codes_data
edge>hdfs dfs -put L_AIRPORT.csv_ /user/root/airport_codes_data
edge>hdfs dfs -put L_CARRIER_HISTORY.csv_ /user/root/carrier_codes_data
```

We also verify that the data was copied properly using appropriate **`ls`** commands.

2. Create external Hive tables for the carrier and airport lookup data:

We create a table called 'airport_lookup' for the airport lookup data in Hive:

```
CREATE EXTERNAL TABLE flights. airport_lookup (
 CODE STRING,
 DESCRIPTION STRING
)ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES
(
"separatorChar" = ",",
"quoteChar"="\""
)
LOCATION '/user/root/airport_codes_data/';
```

Next, we create another table named 'carrier_lookup' for the carrier lookup data in Hive:

```
CREATE EXTERNAL TABLE flights. carrier_lookup (
 CODE STRING,
 DESCRIPTION STRING
) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES
```

```
(
"separatorChar" = ",",
"quoteChar"="\""
)
LOCATION '/user/root/carrier_codes_data/';
```

We also verify if all the data were ingested properly:

```
hive>select count(*) from airport_lookup;  (6555)
hive>select count(*) from carrier_lookup; (1932)
```

We noticed that the carrier_lookup table has multiple entries for the same code (but with different descriptions).

```
hive>select count(distinct code) from carrier_lookup; (1730)
```

To handle the situation, we need to join the flight data with carrier codes in a proper way. If a flight has a carrier code, and that code has multiple descriptions in the lookup file, then we need to look at the descriptions of the carrier code and see which year in the descriptions matches the flight year.

For that, we extract the start and end years from each description and add two new columns in the carrier lookup csv file. A new test.csv file is created which is the same as L_CARRIER_HISTORY.csv with two additional columns: start_year and end_year. Code for doing that:

```
#!/usr/bin/env python
import sys
import string
import csv
with open('L_CARRIER_HISTORY.csv') as csv_file, open('test.csv', mode = 'w') as
write_file:
    csv_reader = csv.reader(csv_file, delimiter=',')
    csv_writer = csv.writer(write_file, delimiter = ',')
    line_count = 0
    for row in csv_reader:
       if line_count == 0:
           csv_writer.writerow(['Code', 'Description', 'Start_year', 'End_year'])
           line_count += 1
       else:
            description = string.strip(row[1], "\n ")
            print 'This is des' + description
            year_string = description [ description.rfind("(")+1 :
description.rfind(")") ].strip()
            print line_count
            print year_string
            start_year, end_year = string.split(year_string, "-")
            start_year = start_year.strip()
            end_year = end_year.strip()
            if end_year == '':
               end_year = '2019'

            csv_writer.writerow([row[0], row[1],start_year, end_year])
```

```
            print 'row1' +row[0]+'  row2   '+row[1]
            line_count += 1
```

Next, we transfer this test.csv file to edge node and then hdfs (/user/root/new_carrier_data/) . Then we create a table named new_carrier_lookup from this csv:

```
create external table flights.new_carrier_lookup(
   > code string,
   > description string,
   > start_year int,
   > end_year int
   > )
   > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
   > WITH SERDEPROPERTIES
   > (
   > "separatorChar" = ",",
   > "quoteChar"="\""
   > )
   > LOCATION '/user/root/new_carrier_data/'
   > ;
```

This table has the start_year and end_year columns as string in spite of declaring as int. So we create a view carrier_view with appropriate type conversion:

```
hive> create view carrier_view as
   > select code, description, cast(start_year as int), cast(end_year as int)
from new_carrier_lookup;
```

We use this view to populate the flight_data_denorm table. While joining, we match the carrier code and choose the code whose start and end year covers the flight year.

We also attempted to write a Hive UDF in Python to create the start_year and end_year columns in the Hive ecosystem itself. We encountered an error that said there were too few values to unpack, but we could not reproduce the error in our standalone code. The python code and corresponding HiveQL query are in the Appendix 5.1.

3. Create a managed Hive table flight_data_denorm:

```
CREATE TABLE flights.flight_data_denorm(
    YEAR INT,
    MONTH INT,
    DAY_OF_MONTH INT,
    FL_DATE DATE,
    UNIQUE_CARRIER STRING,
    AIRLINE_ID INT,
    CARRIER STRING,
    TAIL_NUM STRING,
    FL_NUM INT,
```

```
    ORIGIN_AIRPORT_ID INT ,
    ORIGIN_AIRPORT_SEQ_ID INT,
    ORIGIN STRING,
    DEST_AIRPORT_ID INT,
    DEST_AIRPORT_SEQ_ID INT,
    DEST STRING,
    DEP_DELAY FLOAT,
    ARR_DELAY FLOAT,
    CANCELLED INT,
    DIVERTED INT,
    DISTANCE FLOAT,
    AIRPORT_NAMES ARRAY<STRING>,
    CARRIER_NAMES STRING
    )     ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS ORC ;
```

4. Populate flight_data_denorm with the flight data using Hive:

We populate the table flight_data_denorm using flight_data_orc, airport_lookup and carrier_view along with the condition that only those codes will be chosen whose start and end year covers the flight year.

```
hive> INSERT OVERWRITE TABLE flight_data_denorm SELECT f.year, f.month,
f.day_of_month, f.fl_date, f.unique_carrier, f.airline_id, f.carrier, f.tail_num,
f.fl_num, f.origin_airport_id, f.origin_airport_seq_id, f.origin,
f.dest_airport_id, f.dest_airport_seq_id, f.dest, f.dep_delay, f.arr_delay,
f.cancelled, f.diverted, f.distance, ARRAY(air1.description, air2.description),
car.description FROM flight_data_orc f LEFT OUTER JOIN airport_lookup air1 ON
(f.origin = air1.code) LEFT OUTER JOIN airport_lookup air2 ON (f.dest =
air2.code) LEFT OUTER JOIN carrier_view car ON (f.unique_carrier = car.code)
WHERE f.year BETWEEN car.start_year and car.end_year;
```

Exercise 2: Update the Grep MapReduce (25 points)

Make some updates to the python mapper and reducer from the Hadoop Streaming section of the lab. The current mapper emits a file name if the string you're searching for is in the line of text at all. A helpful example for this can be found here.

Task:

Update the program so it only compares to the name field using an exact name match.

Results:

Paste your updated mapper and reducer below.

Mapper:

```
#!/bin/bash
import os
import sys
SEARCH_STRING = os.environ["SEARCH_STRING"]

for line in sys.stdin:
      words = line.split(",")
      if SEARCH_STRING==words[0]:
            print os.environ["map_input_file"]
```

Reducer:

```
#!/bin/bash
import sys
current_file = None
filename = None

for line in sys.stdin:
  filename = line.strip()
  if current_file != filename:
      print '%s' % (filename)
      current_file = filename
```

Paste an example test case/output proving that your MR program produced more accurate results from the original.

Testcase: string: "Jo"

The outputs are pretty big, so we have attached it to Appendix 5.3. Also, the output of the original code (from the lab) is also included there.

Note: The codes were run on the names folder from the google drive shared by professor.

Describe how your test case proves it gives more accurate results.

The file yob1880.txt does not have the name "Jo". And our code does not give this file as output.

But 1880.txt contains a name “Joella”, which contains the substring “Jo”, that’s why the original code from the lab includes this file as output. So, this test case proves the validity of our method/ code.

Task:

Update the python code so that it only returns the year with the maximum occurrences of the name you’re searching for. **You are not allowed to limit the number of reducers to 1.**

Result:

Paste your mapper and reducer below.

Mapper:

```
#!/bin/bash
import os
import sys

SEARCH_STRING = os.environ["SEARCH_STRING"]
prev_filepath = None
prev_occur = 0

for line in sys.stdin:
      words = line.split(",")
      if SEARCH_STRING==words[0]:
            filepath = os.environ["map_input_file"]
            filename = filepath.split("/")[-1]
            year = filename[3:7]
            occurence = words[2].strip()
            occurence = int(occurence)
            if prev_filepath == filepath: #multiple occurence in the same file
                  occurence += prev_occur

            prev_filepath = filepath
            prev_occur = occurence

            element = str(year)+'_'+str(occurence)
            print "1\t"+ element
```

Essentially, we are enforcing all keys in the outputs of all the mappers to be 1 so that all of them go to one reducer. So even if we specify 16 reducers while running the code, only one of them will get all <key, value> pairs from all the mappers. We also identified that key and value are assumed to be separated by ‘\t’ by default.

Reducer:

```
#!/bin/bash
import sys
max_occurence = None
max_year = 0
for line in sys.stdin:
      line = line.strip()
```

```
      key, element = line.split('\t')
      year, occurence = element.split('_')

       try:
            occurence = int(occurence)
       except:
            continue
       if max_occurence==None:
            max_occurence = occurence
            max_year = year
       if max_occurence < occurence:
            max_occurence = occurence
            max_year = year
max_occurence = str(max_occurence)
print  str(max_year) + "\t" + max_occurence
```

Paste an example test case/output showing that your MR program produced the proper results.

Test case 1: "Zyking". From the lab portion, we know that this name exists in files yob2014.txt and yob2015.txt. By manual checking we see that their occurrences are 6 and 5 respectively. So 2014 and 6 should be the answer for this name. Our code also generates this output.

Output: 2014 6

Test case 2: "Hermine".

Output: 1916 51

Test case 3: "Jo"

Output: 1954 8080

We verify our answers of test case 2 and 3 by writing a python script which reads all the files and finds the file/year with the maximum occurences of the name. The python script is attached in Appendix 5.2 . Our mapreduce output matches the output of the python script.

Note: We noticed that the same name can exist multiple times in the same file. For example, the name "Jo" exists in file yob1954.txt twice (as Jo,M,25 and Jo,F,8055). So, the occurrence for "Jo" in the year 1954 should be 8055+25=8080, which is the maximum among all years for this name. Our code handles this situation and produces the correct output.

EXERCISE 3: LOADING HBASE USING SPARK (5 BONUS POINTS)

Using Spark, you will load an HBase table called delays_spark. It is preferred you do this on your cluster, but if you prefer to try this with Docker and Spark standalone, feel free to do so.

The model of the table is as follows:

Model Attribute	Value	Note
Namespace	flights_spark	
Table Name	delays_spark	
Column Families	dep, arr	dep contains columns that have all the data for flight departure delays, whereas arr contains columns that have all the data for flight arrival delays.
Row Key	[from airport]_[to airport]	Example: MSP_DFW
Column Family dep Columns	Column Name: [flight date]_[tail_num] Value: the total departure delay for that date and tail number	Example: 2016-01-01_N3CTAA => -5.00
Column Family arr Columns	Column Name: [flight date]_[tail_num] Value: the total arrival delay for that date and tail number	Example: 2016-01-01_N3CTAA => -30.00

Task:

Load the HBase table with the data from the 12 months of flight data using Spark.

Results:

Paste your Spark code below.

STANDALONE APPROACH

The three environments needed for this task (Hadoop, HBase and Spark) couldn't work together at a given time. We encountered several bugs and have been trying to debug them all since the morning of December 8th. We made significant headway but still couldn't get the data to load. As a result, we could not run our queries on HBase. However, we would like to explain what we did.

Using standalone Spark and HBase.
Step 1:

We downloaded the relevant Hadoop, HBase and Spark versions from the website and appropriately edited code in the relevant XML files. (core-site.xml, hdfs-site.xml etc.).

Step 2:

Downloaded the relevant Java JDK and JRE files and defined the JAVA_HOME path.

Step 3:

Ran the HBase shell and created the table.

Step 4:

Created the yaml files according to the structure of the csv database.

Step 5 (This is where we encountered problem)

The HBase ran out of runtime while creating the table. There was some problem with the Zookeeper quorum. As a result, we could not fulfil the spark job that would load the data into HBase.

HADOOP CLUSTER APPROACH

In the hadoop cluster approach, the following steps were intended to be completed:

Step 1: Create a SparkDataframe with the data from all the csv

```
>>> df = spark.read.format('csv').option('header',
'true').load("flight_data/*.csv")
```

Step 2: Drop the unnecessary columns from df

```
>>> df2 =
df.select(['FL_DATE','TAIL_NUM','ORIGIN_AIRPORT_ID','DEST_AIRPORT_ID','DEP_DELA
Y',
'ARR_DELAY'])
```

Step 3. Concatenate the columns to meet the specification of the columns family of the HBase table

Step 4. Ingest the data in the HBase table

We attempt multiple times to use the Spark-HBase connector, but encounter the following error everytime:

```
You probably access the destination server through a proxy server that is not
well configured.
```

CODE: (CODE WE INTENDED TO EXECUTE IN SPARKSQL AFTER LOADING DATA)

Write an HBase command that will show the arrival and departure delays for flights on 6/12/2016 from MSP->PDX. Share the command and the results. *If you want to use one command for departures and one for arrivals, that is fine.*

1. **Select avg(arr_delay) from delays_spark where origin = 'MSP' and dest ='PDX' and fl_date = '2016-06-12';**

Select avg(dep_delay) from delays_spark where origin = 'MSP' and dest ='PDX' and fl_date = '2016-06-12';

Task:

Using Spark, query the HBase data for some metrics.

Result:

For each of the following metrics, share the Spark code and results:

1. Average arrival delays on 8/11/2016

```
Select avg(arr_delay) from delays_spark where fl_date =‘ 2016-08-11’;
```

2. Total number of flights with arrivals that were early or on time on 1/2/2016

```
Select count(*) from delays_spark where arr_delay < 0 or arr_delay =0;
```

3. The flight on 11/23/2016 that had the greatest delay, where it left from, and where it was going.

```
Select origin, dest from delays_spark where fl_date = 2016-11-23’ order by
arr_delay desc limit 1;
```

5. Appendix

5.1 Codes for handling duplicate keys in carrier lookup

We attempted to handle duplicate keys in Carrier lookup data using Python UDF in Hive but failed. We encountered an error that said there were too few values to unpack, but we could not reproduce the error in our standalone code.

```
#!/usr/bin/env python

#Python code which gives the start and end year

import sys
import string
import csv

while True:

    line = sys.stdin.readline()
    if not line:
        break


    line = string.strip(line, "\n ")
    year_string = line[line.rfind("(")+1:line.rfind(")")].strip()
    print(year_string)
    start_year, end_year = string.split(year_string, "-")
    start_year = start_year.strip()
    end_year = end_year.strip()
    if end_year == '':
      end_year = '2019'
    print "\t".join([start_year, end_year])
```

HiveQL for using the python UDF

```
SELECT TRANSFORM (description)
```

```
USING 'python hiveudf.py' AS
(start_year string, end_year string)
FROM carrier_lookup;
```

5.2 Python Script for validating the results of Exercise 2.2

```
# Test case for 'Jo'
import os
from csv import reader
import operator
source = '/content/'
dictio ={}
name = 'Jo'
for (root, dirs, files1) in os.walk('/content/', topdown= True):
  for filename in files1:
    if filename.endswith(".txt"):
      if '_' in filename:
        continue
      else:
        opened_file=open(filename)
        read_file=reader(opened_file)
        apps_data = list(read_file)
        for row in apps_data:
          name_col=row[0]
          if name_col == name:
            if filename in dictio:
              dictio[filename]+=int(row[2])
            else:
              dictio[filename] = int(row[2])
result = max(dictio.items(), key=operator.itemgetter(1))[0]
print(dictio)
```

5.3 Output of Exercise 2.1

Output of our code:

hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1890.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1906.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1917.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1928.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1939.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2000.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2011.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1891.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1907.txt

hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1918.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1929.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1990.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2001.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2012.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1881.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1892.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1908.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1919.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1980.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1991.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2002.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2013.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1893.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1909.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1970.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1981.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1992.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2003.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2014.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1883.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1894.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1960.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1971.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1982.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1993.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2004.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2015.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1884.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1895.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1950.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1961.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1972.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1983.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1994.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2005.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1885.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1896.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1940.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1951.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1962.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1973.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1984.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1995.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2006.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1886.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1897.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1930.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1941.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1952.txt

hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1963.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1974.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1985.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1996.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2007.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1887.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1898.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1920.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1931.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1942.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1953.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1964.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1975.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1986.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1997.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2008.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1888.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1899.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1910.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1921.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1932.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1943.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1954.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1965.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1976.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1987.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1998.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2009.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1889.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1900.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1911.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1922.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1933.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1944.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1955.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1966.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1977.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1988.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1999.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1901.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1912.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1923.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1934.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1945.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1956.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1967.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1978.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1989.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1902.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1913.txt

hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1924.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1935.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1946.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1957.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1968.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1979.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1903.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1914.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1925.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1936.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1947.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1958.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1969.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1904.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1915.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1926.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1937.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1948.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1959.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1905.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1916.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1927.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1938.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1949.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2010.txt

Output of the original code from the lab:

hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1890.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1906.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1917.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1928.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1939.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2000.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2011.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1880.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1891.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1907.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1918.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1929.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1990.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2001.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2012.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1881.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1892.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1908.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1919.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1980.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1991.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2002.txt

hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2013.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1882.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1893.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1909.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1970.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1981.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1992.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2003.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2014.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1883.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1894.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1960.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1971.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1982.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1993.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2004.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2015.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1884.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1895.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1950.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1961.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1972.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1983.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1994.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2005.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1885.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1896.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1940.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1951.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1962.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1973.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1984.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1995.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2006.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1886.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1897.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1930.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1941.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1952.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1963.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1974.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1985.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1996.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2007.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1887.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1898.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1920.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1931.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1942.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1953.txt

hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1964.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1975.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1986.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1997.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2008.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1888.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1899.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1910.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1921.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1932.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1943.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1954.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1965.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1976.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1987.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1998.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2009.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1889.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1900.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1911.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1922.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1933.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1944.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1955.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1966.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1977.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1988.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1999.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1901.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1912.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1923.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1934.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1945.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1956.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1967.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1978.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1989.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1902.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1913.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1924.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1935.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1946.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1957.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1968.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1979.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1903.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1914.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1925.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1936.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1947.txt

hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1958.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1969.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1904.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1915.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1926.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1937.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1948.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1959.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1905.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1916.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1927.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1938.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob1949.txt
hdfs://sandbox-hdp.hortonworks.com:8020/user/root/names/yob2010.txt