Hadoop Lab Exercises, v6.0

Course: CSci5751-001, Fall 2019

Instructor: Donald Sawyer

Group/Student: Gavin Celichowski, Jamsrandorj Dagvadorj (gavin_dagva)

Table of Contents

1 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	25	● 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.

   Video of DDL/ETL/MapReduce - Beware of audio feedback starting around ~10:40

2 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  157.0`	`WITH joined_data as (`
		`select origin, dep_delay, description from flight_data_orc2 f`
		`join airport a on f.origin = a.code`
		`)`
		`select description, avg(dep_delay) as avg_delay`
		`from joined_data`
		`group by description`
		`order by avg_delay desc`
		`limit 1;`

Which origin airport [name] has the highest average departure delay? (use flight_data_denorm table)	`Wendover, UT: Wendover Airport  157`	`select origin_dest_airport_names[0], avg(dep_delay) as avg_delay from flight_data_denorm group by origin_dest_airport_names[0]`
		`order by avg_delay desc`
		`limit 1;`

Which carrier [name] had the highest arrival delays 3/14/2016? (use flight_data_date with a lookup table)	SkyWest Airlines Inc. (2003 -) 17881	WITH joined_data as (select carrier, arr_delay, description, fl_date from flight_data_orc2 f join carrier c on f.carrier = c.code) select description, sum(arr_delay) as sum_delay from joined_data where fl_date = '2016-03-14' group by description order by sum_delay desc limit 1;
Which carrier [name] had the highest arrival delays 3/14/2016? (use flight_data_denorm)	SkyWest Airlines Inc. (2003 -) 17881	select carrier_name, sum(arr_delay) as sum_delay from flight_data_denorm where fl_date = '2016-03-14' group by carrier_name order by sum_delay desc limit 1;
What is the total departure delay for flights that took off from an airport with Beaumont in the name? (use flight_data_denorm)	Beaumont/Port Arthur, TX: Jack Brooks Regional 27600	select origin_dest_airport_names[0], sum(dep_delay) as sum_delay from flight_data_denorm where origin_dest_airport_names[0] like '%Beaumont%' group by origin_dest_airport_names[0];
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 origin_dest_airport_names[1] like '%TX%' and carrier_name like '%Virgin%';
How many distinct carrier descriptions are there? (use flight_data_denorm)	15	with distinct_carriers as (select distinct carrier_name from flight_data_denorm) select count(*) from distinct_carriers;

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))	`Atlanta, GA: Hartsfield-Jackson Atlanta International    14200`
	`Chicago, IL: Chicago O'Hare International        8148`
	`Dallas/Fort Worth, TX: Dallas/Fort Worth International   7284`
		`select origin_dest_airport_names[1] , count(fl_num) as num_flights from flight_data_denorm where fl_date>'2016-12-20' and cancelled != 1 group by origin_dest_airport_names[1] order by num_flights desc limit 3;`

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(fl_num) as num_flights from flight_data_denorm where origin_dest_airport_names[1] like '% MT%' group by dest order by num_flights desc limit 1;`
How many rows are in flight_data_denorm?	`6668759`	`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.

```
ALL IS RUN ON HIVE AND IN THE ORDER GIVEN BELOW, VERIFIABLE ON THE VIDEO

carrier DDL

create external table if not exists flights.carrier (

        code string,

        description string)

        ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'

        WITH SERDEPROPERTIES

        (

            "separatorChar" = ",",

            "quoteChar"="\""

        )

        location '/user/root/carrier/';
```

airport DDL

```
create external table if not exists flights.airport (
	code string,
	description string)
	ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
	WITH SERDEPROPERTIES
	(
	    "separatorChar" = ",",
	    "quoteChar"="\""
	)
	location '/user/root/airport/';
```

flight_data_denorm DDL

```
create table if not exists 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 decimal,
	ARR_DELAY decimal,
	CANCELLED decimal,
	DIVERTED decimal,
	DISTANCE decimal,
	CARRIER_NAME STRING,
	ORIGIN_DEST_AIRPORT_NAMES array<STRING>)
STORED AS ORC;
```

Denormalization ETL Query

```
INSERT INTO flights.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, c.description, array(o.description, d.description)

from flight_data_orc2 f

join carrier c on f.carrier = c.code

join airport o on f.origin = o.code

join airport d on f.dest = d.code;
```

3 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.

We used the same reducer

Updated Mapper:

```
#!/bin/bash

import os

import sys

SEARCH_STRING = os.environ["SEARCH_STRING"]

for line in sys.stdin:

    line = line.strip()

    words = line.split(',')

    if SEARCH_STRING == words[0]:

        print os.environ["map_input_file"]
```

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

Search Phrase: "And"

Updated:

```
hdfs dfs -cat /user/root/grep_out/part-* | wc -l

0
```

Original:

```
hdfs dfs -cat /user/root/grep_out/part-* | wc -l

136
```

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

Our test case proves that the updated MR gives more accurate results because And is an uncommon name, but it is a part of names like Andrew, Andre, Andrea, etc. which are very common. The name “And” showed up in 0 files for exact match mapper, but with the original mapper it showed up in 136 files, proving that it gives much more accurate results. To verify that the algorithm works. Additional results- Michael: 136/136 files in original and updated.

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"]

for line in sys.stdin:

    line = line.strip()

    words = line.split(',')

    if SEARCH_STRING == words[0]:

      print '%s\t%s,%s' % (words[0], os.environ["map_input_file"], words[2])
```

REDUCER:

```
#!/bin/bash
import sys
curword = None
curcount = 0
mx = 0
ans = ""
myMap = {}
for line in sys.stdin:
	line = line.strip()
	words= line.split('\t')
	if len(words) < 2:
		continue
	filename, cnt = words[1].split(',')
	year = filename[-8:][0:4]
	if year not in myMap:
		myMap[year] = 0
	currCount = myMap[year]
	myMap[year] = currCount + int(cnt)
	if currCount + int(cnt) > mx:
		mx = currCount + int(cnt)
		ans = year
print ans
```

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

EXAMPLE:

```
hadoop jar  /usr/hdp/2.5.0.0-1245/hadoop-mapreduce/hadoop-streaming.jar -D
mapred.reduce.tasks=16 -input /user/root/names/*.txt -mapper "python
grep_search_count.py" -file "grep_search_count.py" -reducer "python
grep_search_reduce_count.py" -file "grep_search_reduce_count.py" -output
/user/root/grep_out -cmdenv SEARCH_STRING="Andrew"
```

```
hdfs dfs -cat /user/root/grep_out/*
```

```
1987
```

This result is confirmed using a local test. The code for the test is below (see Exercise 2 Notes)

Additionally, the MapReduce program correctly identified 2014 for Zyking, and 1922 for Stclair. These results were confirmed by both manual inspection and the local test, and the MapReduce results are below, with the full output for Zyking in the Appendix:

-Zyking:

```
hadoop jar  /usr/hdp/2.5.0.0-1245/hadoop-mapreduce/hadoop-streaming.jar -D
mapred.reduce.tasks=16 -input /user/root/names/*.txt -mapper "python
grep_search_count.py" -file "grep_search_count.py" -reducer "python
grep_search_reduce_count.py" -file "grep_search_reduce_count.py" -output
/user/root/grep_out -cmdenv SEARCH_STRING="Zyking"
```

```
hdfs dfs -cat /user/root/grep_out/*
```

```
2014
```

-Stclair

```
hadoop jar  /usr/hdp/2.5.0.0-1245/hadoop-mapreduce/hadoop-streaming.jar -D
mapred.reduce.tasks=16 -input /user/root/names/*.txt -mapper "python
grep_search_count.py" -file "grep_search_count.py" -reducer "python
grep_search_reduce_count.py" -file "grep_search_reduce_count.py" -output
/user/root/grep_out -cmdenv SEARCH_STRING="Stclair"
```

```
hdfs dfs -cat /user/root/grep_out/*
```

```
1922
```

4 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.

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.*

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

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

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

5 Appendix:

EXERCISE 2 NOTES:

Local Test Code for MapReduce Verification (Runs on Local Machine):

```
import os
files = os.listdir("./")
SEARCH_STRING = 'Andrew'
cnt = {}
mx = 0
year = ""
for fname in files:
      if '.txt' not in fname:
            continue
      for line in open(fname, 'r'):
            line = line.strip()
            words = line.split(',')
            if (len(words) < 3):
                  continue
            if SEARCH_STRING == words[0]:
                  if fname not in cnt:
                        cnt[fname] = 0
                  tmp = cnt[fname]
                  cnt[fname] = tmp + int(words[2])
                  if tmp + int(words[2]) > mx:
                        mx = tmp + int(words[2])
                        year = fname
print(year)
print(mx)
```

EXERCISE 2 FULL RESULTS:

-Zyking:

```
root@sandbox-hdp names]# hadoop jar
/usr/hdp/2.5.0.0-1245/hadoop-mapreduce/hadoop-streaming.jar -D mapred.reduce.tasks=16
-input /us

er/root/names/*.txt -mapper "python grep_search_count.py" -file "grep_search_count.py"
-reducer "python grep_search_reduce_count.py" -

file "grep_search_reduce_count.py" -output /user/root/grep_out -cmdenv
SEARCH_STRING="Zyking"

19/12/10 19:27:38 WARN streaming.StreamJob: -file option is deprecated, please use
generic option -files instead.

packageJobJar: [grep_search_count.py, grep_search_reduce_count.py]
[/usr/hdp/2.6.4.0-91/hadoop-mapreduce/hadoop-streaming-2.7.3.2.6.4.

0-91.jar] /tmp/streamjob4818033407082433874.jar tmpDir=null

19/12/10 19:27:41 INFO client.RMProxy: Connecting to ResourceManager at
sandbox-hdp.hortonworks.com/172.17.0.2:8032

19/12/10 19:27:42 INFO client.AHSProxy: Connecting to Application History server at
sandbox-hdp.hortonworks.com/172.17.0.2:10200

19/12/10 19:27:42 INFO client.RMProxy: Connecting to ResourceManager at
sandbox-hdp.hortonworks.com/172.17.0.2:8032

19/12/10 19:27:42 INFO client.AHSProxy: Connecting to Application History server at
sandbox-hdp.hortonworks.com/172.17.0.2:10200

19/12/10 19:27:44 INFO mapred.FileInputFormat: Total input paths to process : 136

19/12/10 19:27:45 INFO mapreduce.JobSubmitter: number of splits:136

19/12/10 19:27:46 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1576005393438_0001

19/12/10 19:27:48 INFO impl.YarnClientImpl: Submitted application
application_1576005393438_0001

19/12/10 19:27:48 INFO mapreduce.Job: The url to track the job:
http://sandbox-hdp.hortonworks.com:8088/proxy/application_157600539343

8_0001/

19/12/10 19:27:48 INFO mapreduce.Job: Running job: job_1576005393438_0001

19/12/10 19:28:21 INFO mapreduce.Job: Job job_1576005393438_0001 running in uber mode :
false

19/12/10 19:28:21 INFO mapreduce.Job:  map 0% reduce 0%

19/12/10 19:29:43 INFO mapreduce.Job:  map 1% reduce 0%

19/12/10 19:29:45 INFO mapreduce.Job:  map 2% reduce 0%
```

```
19/12/10 19:29:46 INFO mapreduce.Job:  map 3% reduce 0%
19/12/10 19:29:49 INFO mapreduce.Job:  map 4% reduce 0%
19/12/10 19:29:50 INFO mapreduce.Job:  map 5% reduce 0%
19/12/10 19:29:51 INFO mapreduce.Job:  map 6% reduce 0%
19/12/10 19:29:52 INFO mapreduce.Job:  map 8% reduce 0%
19/12/10 19:31:03 INFO mapreduce.Job:  map 10% reduce 0%
19/12/10 19:31:13 INFO mapreduce.Job:  map 11% reduce 0%
19/12/10 19:31:14 INFO mapreduce.Job:  map 12% reduce 0%
19/12/10 19:31:16 INFO mapreduce.Job:  map 13% reduce 0%
19/12/10 19:31:17 INFO mapreduce.Job:  map 15% reduce 0%
19/12/10 19:31:22 INFO mapreduce.Job:  map 16% reduce 0%
19/12/10 19:32:25 INFO mapreduce.Job:  map 17% reduce 0%
19/12/10 19:32:28 INFO mapreduce.Job:  map 18% reduce 0%
19/12/10 19:32:41 INFO mapreduce.Job:  map 20% reduce 0%
19/12/10 19:32:43 INFO mapreduce.Job:  map 22% reduce 0%
19/12/10 19:32:46 INFO mapreduce.Job:  map 24% reduce 0%
19/12/10 19:33:38 INFO mapreduce.Job:  map 25% reduce 0%
19/12/10 19:33:39 INFO mapreduce.Job:  map 25% reduce 1%
19/12/10 19:33:59 INFO mapreduce.Job:  map 26% reduce 1%
19/12/10 19:34:01 INFO mapreduce.Job:  map 28% reduce 1%
19/12/10 19:34:02 INFO mapreduce.Job:  map 29% reduce 1%
19/12/10 19:34:04 INFO mapreduce.Job:  map 30% reduce 1%
19/12/10 19:34:41 INFO mapreduce.Job:  map 31% reduce 1%
19/12/10 19:34:48 INFO mapreduce.Job:  map 32% reduce 1%
19/12/10 19:35:12 INFO mapreduce.Job:  map 33% reduce 1%
19/12/10 19:35:13 INFO mapreduce.Job:  map 34% reduce 2%
19/12/10 19:35:14 INFO mapreduce.Job:  map 35% reduce 2%
19/12/10 19:35:19 INFO mapreduce.Job:  map 36% reduce 2%
19/12/10 19:35:48 INFO mapreduce.Job:  map 37% reduce 2%
19/12/10 19:35:51 INFO mapreduce.Job:  map 38% reduce 2%
19/12/10 19:36:11 INFO mapreduce.Job:  map 39% reduce 2%
19/12/10 19:36:15 INFO mapreduce.Job:  map 40% reduce 2%
```

```
19/12/10 19:36:22 INFO mapreduce.Job:  map 41% reduce 3%

19/12/10 19:36:23 INFO mapreduce.Job:  map 42% reduce 3%

19/12/10 19:36:50 INFO mapreduce.Job:  map 43% reduce 3%

19/12/10 19:36:51 INFO mapreduce.Job:  map 43% reduce 4%

19/12/10 19:37:13 INFO mapreduce.Job:  map 44% reduce 4%

19/12/10 19:37:14 INFO mapreduce.Job:  map 45% reduce 4%

19/12/10 19:37:18 INFO mapreduce.Job:  map 46% reduce 4%

19/12/10 19:37:23 INFO mapreduce.Job:  map 47% reduce 4%

19/12/10 19:37:49 INFO mapreduce.Job:  map 48% reduce 4%

19/12/10 19:37:59 INFO mapreduce.Job:  map 49% reduce 4%

19/12/10 19:38:12 INFO mapreduce.Job:  map 50% reduce 5%

19/12/10 19:38:14 INFO mapreduce.Job:  map 51% reduce 5%

19/12/10 19:38:39 INFO mapreduce.Job:  map 52% reduce 5%

19/12/10 19:38:45 INFO mapreduce.Job:  map 53% reduce 5%

19/12/10 19:38:48 INFO mapreduce.Job:  map 53% reduce 6%

19/12/10 19:38:57 INFO mapreduce.Job:  map 54% reduce 6%

19/12/10 19:39:03 INFO mapreduce.Job:  map 55% reduce 6%

19/12/10 19:39:06 INFO mapreduce.Job:  map 56% reduce 6%

19/12/10 19:39:27 INFO mapreduce.Job:  map 57% reduce 6%

19/12/10 19:39:46 INFO mapreduce.Job:  map 58% reduce 6%

19/12/10 19:39:49 INFO mapreduce.Job:  map 59% reduce 6%

19/12/10 19:39:50 INFO mapreduce.Job:  map 60% reduce 6%

19/12/10 19:40:12 INFO mapreduce.Job:  map 61% reduce 6%

19/12/10 19:40:24 INFO mapreduce.Job:  map 62% reduce 6%

19/12/10 19:40:33 INFO mapreduce.Job:  map 63% reduce 6%

19/12/10 19:40:37 INFO mapreduce.Job:  map 63% reduce 7%

19/12/10 19:40:38 INFO mapreduce.Job:  map 64% reduce 7%

19/12/10 19:40:42 INFO mapreduce.Job:  map 65% reduce 7%

19/12/10 19:41:23 INFO mapreduce.Job:  map 67% reduce 7%

19/12/10 19:41:24 INFO mapreduce.Job:  map 68% reduce 7%

19/12/10 19:41:30 INFO mapreduce.Job:  map 69% reduce 7%

19/12/10 19:41:47 INFO mapreduce.Job:  map 70% reduce 7%
```

```
19/12/10 19:42:04 INFO mapreduce.Job:  map 71% reduce 7%

19/12/10 19:42:15 INFO mapreduce.Job:  map 72% reduce 7%

19/12/10 19:42:17 INFO mapreduce.Job:  map 73% reduce 7%

19/12/10 19:42:18 INFO mapreduce.Job:  map 73% reduce 8%

19/12/10 19:42:19 INFO mapreduce.Job:  map 74% reduce 8%

19/12/10 19:42:57 INFO mapreduce.Job:  map 75% reduce 8%

19/12/10 19:43:04 INFO mapreduce.Job:  map 76% reduce 8%

19/12/10 19:43:09 INFO mapreduce.Job:  map 77% reduce 8%

19/12/10 19:43:11 INFO mapreduce.Job:  map 78% reduce 8%

19/12/10 19:43:26 INFO mapreduce.Job:  map 79% reduce 8%

19/12/10 19:43:52 INFO mapreduce.Job:  map 80% reduce 8%

19/12/10 19:43:54 INFO mapreduce.Job:  map 81% reduce 8%

19/12/10 19:44:02 INFO mapreduce.Job:  map 82% reduce 8%

19/12/10 19:44:04 INFO mapreduce.Job:  map 82% reduce 9%

19/12/10 19:44:16 INFO mapreduce.Job:  map 83% reduce 9%

19/12/10 19:44:32 INFO mapreduce.Job:  map 84% reduce 9%

19/12/10 19:44:40 INFO mapreduce.Job:  map 85% reduce 9%

19/12/10 19:44:50 INFO mapreduce.Job:  map 86% reduce 9%

19/12/10 19:44:51 INFO mapreduce.Job:  map 87% reduce 9%

19/12/10 19:44:57 INFO mapreduce.Job:  map 88% reduce 9%

19/12/10 19:45:30 INFO mapreduce.Job:  map 89% reduce 9%

19/12/10 19:45:33 INFO mapreduce.Job:  map 90% reduce 9%

19/12/10 19:45:41 INFO mapreduce.Job:  map 91% reduce 9%

19/12/10 19:45:43 INFO mapreduce.Job:  map 92% reduce 9%

19/12/10 19:45:44 INFO mapreduce.Job:  map 92% reduce 10%

19/12/10 19:46:06 INFO mapreduce.Job:  map 93% reduce 10%

19/12/10 19:46:21 INFO mapreduce.Job:  map 94% reduce 10%

19/12/10 19:46:25 INFO mapreduce.Job:  map 95% reduce 10%

19/12/10 19:46:28 INFO mapreduce.Job:  map 96% reduce 10%

19/12/10 19:46:52 INFO mapreduce.Job:  map 97% reduce 10%

19/12/10 19:47:07 INFO mapreduce.Job:  map 98% reduce 10%

19/12/10 19:47:08 INFO mapreduce.Job:  map 99% reduce 10%
```

```
19/12/10 19:47:15 INFO mapreduce.Job:  map 100% reduce 10%
19/12/10 19:47:17 INFO mapreduce.Job:  map 100% reduce 12%
19/12/10 19:47:18 INFO mapreduce.Job:  map 100% reduce 19%
19/12/10 19:47:19 INFO mapreduce.Job:  map 100% reduce 21%
19/12/10 19:47:21 INFO mapreduce.Job:  map 100% reduce 27%
19/12/10 19:47:22 INFO mapreduce.Job:  map 100% reduce 31%
19/12/10 19:47:23 INFO mapreduce.Job:  map 100% reduce 35%
19/12/10 19:47:24 INFO mapreduce.Job:  map 100% reduce 38%
19/12/10 19:47:50 INFO mapreduce.Job:  map 100% reduce 42%
19/12/10 19:47:56 INFO mapreduce.Job:  map 100% reduce 44%
19/12/10 19:48:14 INFO mapreduce.Job:  map 100% reduce 52%
19/12/10 19:48:19 INFO mapreduce.Job:  map 100% reduce 54%
19/12/10 19:48:23 INFO mapreduce.Job:  map 100% reduce 56%
19/12/10 19:48:28 INFO mapreduce.Job:  map 100% reduce 60%
19/12/10 19:48:29 INFO mapreduce.Job:  map 100% reduce 65%
19/12/10 19:48:31 INFO mapreduce.Job:  map 100% reduce 69%
19/12/10 19:48:34 INFO mapreduce.Job:  map 100% reduce 73%
19/12/10 19:48:36 INFO mapreduce.Job:  map 100% reduce 85%
19/12/10 19:48:38 INFO mapreduce.Job:  map 100% reduce 100%
19/12/10 19:48:41 INFO mapreduce.Job: Job job_1576005393438_0001 completed successfully
19/12/10 19:48:41 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=254
                FILE: Number of bytes written=23810652
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=24000488
                HDFS: Number of bytes written=6
                HDFS: Number of read operations=456
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=32
```

```
Job Counters

        Launched map tasks=136

        Launched reduce tasks=16

        Data-local map tasks=136

        Total time spent by all maps in occupied slots (ms)=8153624

        Total time spent by all reduces in occupied slots (ms)=4691705

        Total time spent by all map tasks (ms)=8153624

        Total time spent by all reduce tasks (ms)=4691705

        Total vcore-milliseconds taken by all map tasks=8153624

        Total vcore-milliseconds taken by all reduce tasks=4691705

        Total megabyte-milliseconds taken by all map tasks=2038406000

        Total megabyte-milliseconds taken by all reduce tasks=1172926250

Map-Reduce Framework

        Map input records=1858689

        Map output records=2

        Map output bytes=154

        Map output materialized bytes=13214

        Input split bytes=16184

        Combine input records=0

        Combine output records=0

        Reduce input groups=1

        Reduce shuffle bytes=13214

        Reduce input records=2

        Reduce output records=1

        Spilled Records=4

        Shuffled Maps =2176

        Failed Shuffles=0

        Merged Map outputs=2176

        GC time elapsed (ms)=304399

        CPU time spent (ms)=262300

        Physical memory (bytes) snapshot=28879081472

        Virtual memory (bytes) snapshot=322243117056
```

```
                Total committed heap usage (bytes)=16950755328
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=23984304
        File Output Format Counters
                Bytes Written=6
19/12/10 19:48:41 INFO streaming.StreamJob: Output directory: /user/root/grep_out
[root@sandbox-hdp names]# hdfs dfs -cat /user/root/grep_out/*
2014
```