Document search using Map Reduce and HIVE

Chaitra Hosmani
Date:Dec 06, 2018

Contents

1. Problem Statement

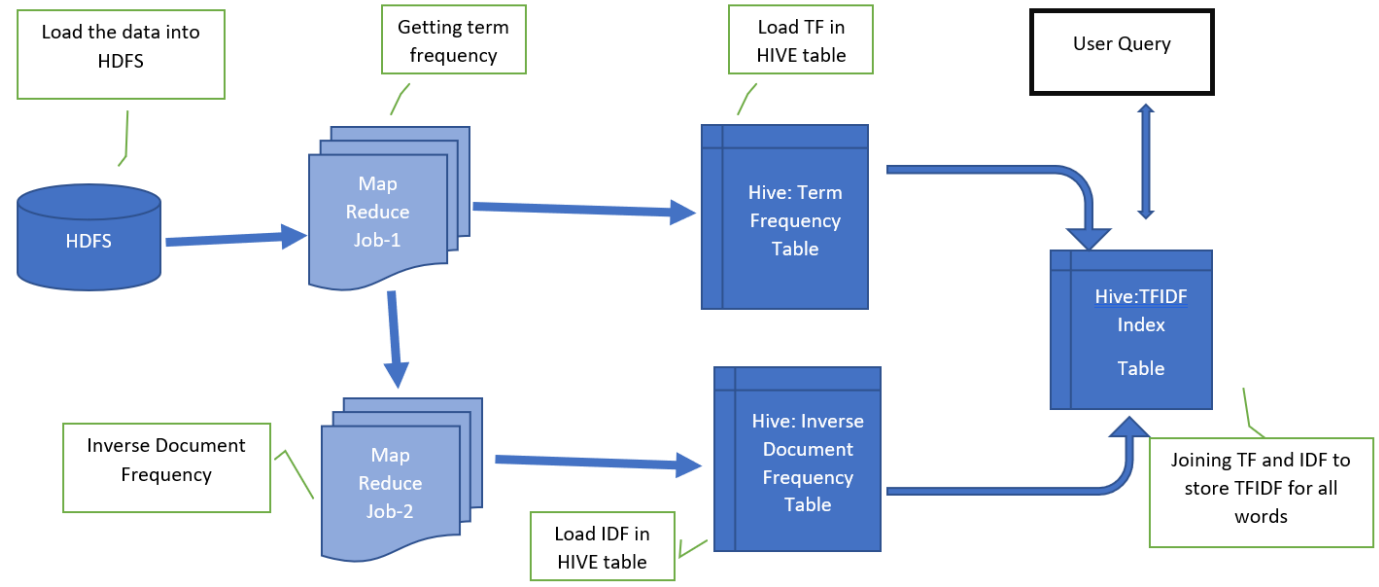
Find relevant documents based on user's keyword as an input

2. Summary

- The solution involves uses Map Reduce and Hive for its implementation
- Map Reduce code is easily comprehensible and HIVE QL is similar to SQL like language
- Map Reduce is written in python which is easy to code as I have exposure writing in python
- HIVE makes it easy to query using different data manipulation methods
- I am choosing tennis data set to do the document search which contain 100 text files

3. Architecture

- Once the data is loaded in HDFS, it makes use of map reduce jobs to create term frequency and inverse document frequency for each word.
- The output of map reduce jobs will be stored in hive table
- The join of two tables in hive will create the final index table from which search will happen
- One can query directly into hive table to search for relevant document
- The details methods used for Map Reduce job is given in section 7:Map Reduce Code Snippets
- The details of execution of Map Reduce and using HIVE is given in section 6: Execution Snippets



4. Query Results

This section shows the some of the query performed and the results obtained.

Query: Show top 1 document containing "Roger"

Command: select * from tfidf_index where word =LOWER('Roger') order by tfidf desc limit 1;

This is implemented to handle case insensitive searches

```
hive> select * from tfidf_index where word =LOWER('Roger') order by tfidf desc limit 1;
Query ID = root_20181001210803_13d75d76-1581-4a0a-8d13-19ca2fd12a0b
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1537877116120_0233)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 3.56 s
--------------------------------------------------------------------------------
OK
roger   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/005.txt        4.41
Time taken: 4.141 seconds, Fetched: 1 row(s)
```

Query: Show top 3 document containing "Roger"

select * from tfidf_index where word =LOWER('Roger') order by tfidf desc limit 3;

```
hive> select * from tfidf_index where word =LOWER('Roger') order by tfidf desc limit 3;
Query ID = root_20181001210918_050a4cb6-22dc-49f7-96de-dfd1873507c1
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1537877116120_0233)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 3.73 s
--------------------------------------------------------------------------------
OK
roger   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/005.txt        4.41
roger   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/013.txt        4.41
roger   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/083.txt        2.94
Time taken: 4.348 seconds, Fetched: 3 row(s)
```

Query: Show top 3 document containing “Roger”

select * from tfidf_index where word =LOWER('Roger') order by tfidf desc limit 5;

```
hive> select * from tfidf_index where word =LOWER('Roger') order by tfidf desc limit 5;
Query ID = root_20181001211044_1be6f9ae-4c35-48bd-8298-f66b92966b23
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1537877116120_0233)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 2.84 s
--------------------------------------------------------------------------------
OK
roger   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/005.txt        4.41
roger   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/013.txt        4.41
roger   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/083.txt        2.94
roger   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/095.txt        2.94
roger   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/056.txt        2.94
Time taken: 3.453 seconds, Fetched: 5 row(s)
```

Query: Show top 1 document containing “wildcard entry”

select * from tfidf_index where word =LOWER('wildcard') or word=LOWER('ENTRY') order by tfidf desc limit 1;

```
hive> select * from tfidf_index where word =LOWER('wildcard') or word=LOWER('ENTRY') order by tfidf desc limit 1;
Query ID = root_20181001223839_41e08f37-83ce-431c-b7ff-9b9c35b56300
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1537877116120_0234)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 4.56 s
--------------------------------------------------------------------------------
OK
wildcard        hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/006.txt        7.82
Time taken: 5.182 seconds, Fetched: 1 row(s)
```

Query: Show top 3 document containing “wildcard entry”

select * from tfidf_index where word =LOWER('wildcard') or word=LOWER('ENTRY') order by tfidf desc limit 3;

```
hive> select * from tfidf_index where word =LOWER('wildcard') or word=LOWER('ENTRY') order by tfidf desc limit 3;
Query ID = root_20181001223940_18e3bdcd-3fe4-4d40-ba54-c0465db4ce23
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1537877116120_0234)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 3.74 s
--------------------------------------------------------------------------------
OK
wildcard        hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/006.txt        7.82
entry   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/059.txt        6.0
entry   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/058.txt        6.0
Time taken: 4.338 seconds, Fetched: 3 row(s)
```

Query: Show top 5 document containing “wildcard entry”

select * from tfidf_index where word =LOWER('wildcard') or word=LOWER('ENTRY') order by tfidf desc limit 5;

```
hive> select * from tfidf_index where word =LOWER('wildcard') or word=LOWER('ENTRY') order by tfidf desc limit 5;
Query ID = root_20181001224016_f669ea20-401d-496a-b9d5-239d6e76d222
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1537877116120_0234)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 3.50 s
--------------------------------------------------------------------------------
OK
wildcard        hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/006.txt        7.82
entry   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/058.txt        6.0
entry   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/059.txt        6.0
wildcard        hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/043.txt        3.91
entry   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/029.txt        3.0
Time taken: 4.085 seconds, Fetched: 5 row(s)
hive>
```

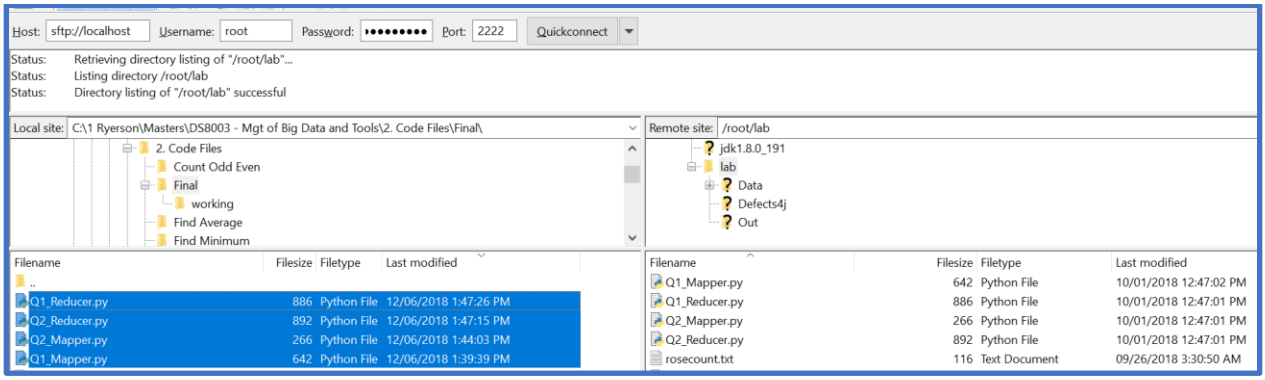
5. Real time Implementation

We can provide an user interface for end users to access hive query directly. With this, user can search for relevant documents on demand

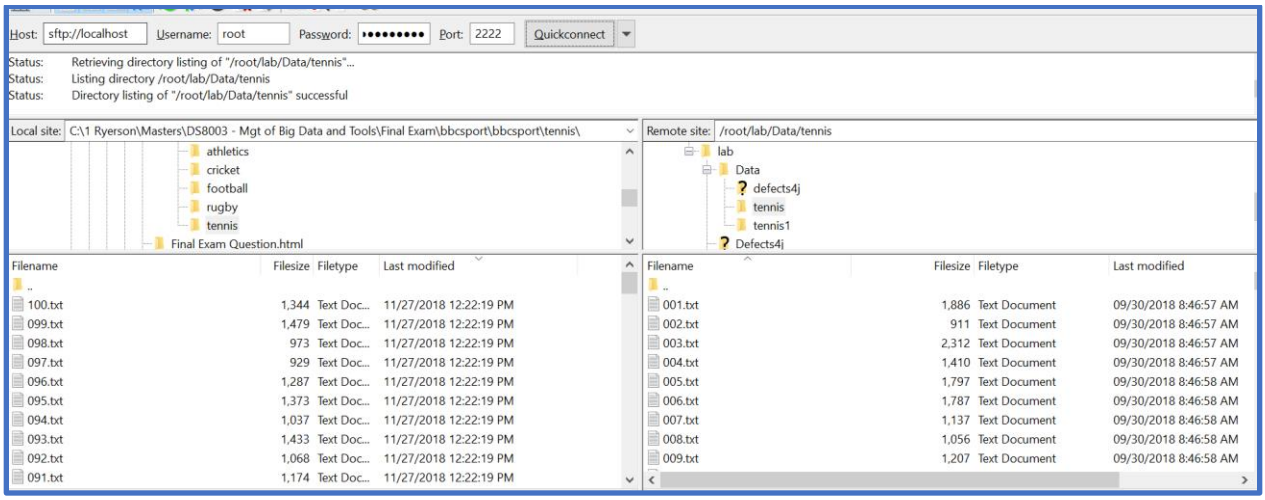
6. Execution Snippets

STEP1: Copying of files

a. Copy the map-reduce files in Unix system



b. Copy data files(tennis folder) from local to Unix system



c. Copy data files(tennis folder) from Unix system to HDFS.

```
[root@sandbox-hdp ~]# hdfs dfs -put /root/lab/Data/tennis /user/root/
```

d. Showing files loaded in HDFS

```
[root@sandbox-hdp ~]# hdfs dfs -ls /user/root/tennis/
Found 100 items
-rw-r--r--   1 root hdfs       1886 2018-10-01 16:49 /user/root/tennis/001.txt
-rw-r--r--   1 root hdfs        911 2018-10-01 16:49 /user/root/tennis/002.txt
-rw-r--r--   1 root hdfs       2312 2018-10-01 16:49 /user/root/tennis/003.txt
-rw-r--r--   1 root hdfs       1410 2018-10-01 16:49 /user/root/tennis/004.txt
-rw-r--r--   1 root hdfs       1797 2018-10-01 16:49 /user/root/tennis/005.txt
-rw-r--r--   1 root hdfs       1787 2018-10-01 16:49 /user/root/tennis/006.txt
-rw-r--r--   1 root hdfs       1137 2018-10-01 16:49 /user/root/tennis/007.txt
-rw-r--r--   1 root hdfs       1056 2018-10-01 16:49 /user/root/tennis/008.txt
-rw-r--r--   1 root hdfs       1207 2018-10-01 16:49 /user/root/tennis/009.txt
-rw-r--r--   1 root hdfs       1603 2018-10-01 16:49 /user/root/tennis/010.txt
-rw-r--r--   1 root hdfs       1591 2018-10-01 16:49 /user/root/tennis/011.txt
```

STEP 2: Executing map reduce jobs

a. **Run the first map-reduce job** which will read all the input files (tennis folder) and produce term frequency for each word per document

Execution code:

```
hadoop jar /usr/hdp/2.6.5.0-292/hadoop-mapreduce/hadoop-streaming-2.7.3.2.6.5.0-292.jar -
file  /root/lab/Q1_Mapper.py -mapper Q1_Mapper.py -file /root/lab/Q1_Red
ucer.py -reducer Q1_Reducer.py -input /user/root/tennis/*.txt -output
/user/root/tennis_fin_01
```

```
[root@sandbox-hdp ~]# hadoop jar /usr/hdp/2.6.5.0-292/hadoop-mapreduce/hadoop-streaming-2.7.3.2.6.5.0-292.jar -file  /root/lab/Q1_Mapper.py -mapper Q1_Mapper.py -file /root/la
ucer.py -reducer Q1_Reducer.py -input /user/root/tennis/*.txt -output /user/root/tennis_fin_01
18/10/01 19:01:37 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [/root/lab/Q1_Mapper.py, /root/lab/Q1_Reducer.py] [/usr/hdp/2.6.5.0-292/hadoop-mapreduce/hadoop-streaming-2.7.3.2.6.5.0-292.jar] /tmp/streamjob5522916062252790
mpDir=null
18/10/01 19:01:38 INFO client.RMProxy: Connecting to ResourceManager at sandbox-hdp.hortonworks.com/172.18.0.2:8032
18/10/01 19:01:38 INFO client.AHSProxy: Connecting to Application History server at sandbox-hdp.hortonworks.com/172.18.0.2:10200
18/10/01 19:01:39 INFO client.RMProxy: Connecting to ResourceManager at sandbox-hdp.hortonworks.com/172.18.0.2:8032
18/10/01 19:01:39 INFO client.AHSProxy: Connecting to Application History server at sandbox-hdp.hortonworks.com/172.18.0.2:10200
18/10/01 19:01:39 INFO mapred.FileInputFormat: Total input paths to process : 100
18/10/01 19:01:39 INFO mapreduce.JobSubmitter: number of splits:100
18/10/01 19:01:40 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1537877116120_0227
18/10/01 19:01:40 INFO impl.YarnClientImpl: Submitted application application_1537877116120_0227
18/10/01 19:01:40 INFO mapreduce.Job: The url to track the job: http://sandbox-hdp.hortonworks.com:8088/proxy/application_1537877116120_0227/
18/10/01 19:01:40 INFO mapreduce.Job: Running job: job_1537877116120_0227
18/10/01 19:01:46 INFO mapreduce.Job: Job job_1537877116120_0227 running in uber mode : false
18/10/01 19:01:46 INFO mapreduce.Job:  map 0% reduce 0%
18/10/01 19:02:00 INFO mapreduce.Job:  map 7% reduce 0%
18/10/01 19:02:01 INFO mapreduce.Job:  map 11% reduce 0%
18/10/01 19:02:13 INFO mapreduce.Job:  map 14% reduce 0%
18/10/01 19:02:14 INFO mapreduce.Job:  map 17% reduce 0%
18/10/01 19:02:15 INFO mapreduce.Job:  map 21% reduce 0%
18/10/01 19:02:21 INFO mapreduce.Job:  map 21% reduce 7%
18/10/01 19:02:27 INFO mapreduce.Job:  map 25% reduce 7%
18/10/01 19:02:28 INFO mapreduce.Job:  map 30% reduce 7%
```

Successful execution of map-reduce 1

```
18/10/01 19:04:05 INFO mapreduce.Job: Job job_1537877116120_0227 completed successfully
```

Check few lines of output from Map Reduce 1

```
[root@sandbox-hdp ~]# hdfs dfs -cat /user/root/tennis_fin_01/part-00000 | tail -10
zheng   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/010.txt        1
zheng   hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/016.txt        1
zib     hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/087.txt        1
zone    hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/011.txt        1
zone    hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/029.txt        1
zuluaga hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/056.txt        1
zvonareva       hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/054.txt        3
zvonareva       hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/056.txt        1
£115,000        hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/036.txt        1
£28,000 hdfs://sandbox-hdp.hortonworks.com:8020/user/root/tennis/031.txt        1
```

b. **Run the second map reduce job**. The input is the result of previous map-reduce job. This will count the documents containing each word and calculate it's inverse document frequency

Execution code

```
hadoop jar /usr/hdp/2.6.5.0-292/hadoop-mapreduce/hadoop-streaming-2.7.3.2.6.5.0-292.jar -
file  /root/lab/Q2_Mapper.py -mapper Q2_Mapper.py -file /root/lab/Q2_Red
ucer.py -reducer Q2_Reducer.py -input /user/root/tennis_fin_01/part-00000 -output
/user/root/tennis_fin_07
```

```
[root@sandbox-hdp ~]# hadoop jar /usr/hdp/2.6.5.0-292/hadoop-mapreduce/hadoop-streaming-2.7.3.2.6.5.0-292.jar -file  /root/lab/Q2_Mapper.py -mapper Q2_Mapper.py -file /root/la
ucer.py -reducer Q2_Reducer.py -input /user/root/tennis_fin_01/part-00000 -output /user/root/tennis_fin_07
18/10/01 19:26:14 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [/root/lab/Q2_Mapper.py, /root/lab/Q2_Reducer.py] [/usr/hdp/2.6.5.0-292/hadoop-mapreduce/hadoop-streaming-2.7.3.2.6.5.0-292.jar] /tmp/streamjob7987575574429555
mpDir=null
18/10/01 19:26:15 INFO client.RMProxy: Connecting to ResourceManager at sandbox-hdp.hortonworks.com/172.18.0.2:8032
18/10/01 19:26:15 INFO client.AHSProxy: Connecting to Application History server at sandbox-hdp.hortonworks.com/172.18.0.2:10200
18/10/01 19:26:16 INFO client.RMProxy: Connecting to ResourceManager at sandbox-hdp.hortonworks.com/172.18.0.2:8032
18/10/01 19:26:16 INFO client.AHSProxy: Connecting to Application History server at sandbox-hdp.hortonworks.com/172.18.0.2:10200
18/10/01 19:26:16 INFO mapred.FileInputFormat: Total input paths to process : 1
18/10/01 19:26:16 INFO mapreduce.JobSubmitter: number of splits:2
18/10/01 19:26:17 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1537877116120_0231
18/10/01 19:26:17 INFO impl.YarnClientImpl: Submitted application application_1537877116120_0231
18/10/01 19:26:17 INFO mapreduce.Job: The url to track the job: http://sandbox-hdp.hortonworks.com:8088/proxy/application_1537877116120_0231/
18/10/01 19:26:17 INFO mapreduce.Job: Running job: job_1537877116120_0231
18/10/01 19:26:23 INFO mapreduce.Job: Job job_1537877116120_0231 running in uber mode : false
18/10/01 19:26:23 INFO mapreduce.Job:  map 0% reduce 0%
18/10/01 19:26:28 INFO mapreduce.Job:  map 100% reduce 0%
```

Check few lines from the output of MapReduce 2

```
[root@sandbox-hdp ~]# hdfs dfs -cat /user/root/tennis_fin_07/part-00000 | tail -10
zabaleta        4.61
zealand 4.61
zero    4.61
zheng   3.91
zib     4.61
zone    3.91
zuluaga 4.61
zvonareva       3.91
£115,000        4.61
£28,000 4.60517018599
```

STEP 3: Save the output of map reduce jobs for loading HIVE tables

a. copy output files from both MapReduce jobs and assign a meaningful name

Execution Code

```
hdfs dfs -mkdir /user/root/tennis_mapred_output

hdfs dfs -cp /user/root/tennis_fin_01/part-00000
/user/root/tennis_mapred_output/term_frequency

hdfs dfs -cp /user/root/tennis_fin_07/part-00000
/user/root/tennis_mapred_output/inv_doc_frequency
```

```
[root@sandbox-hdp ~]# hdfs dfs -mkdir /user/root/tennis_mapred_output
[root@sandbox-hdp ~]# hdfs dfs -cp /user/root/tennis_fin_01/part-00000 /user/root/tennis_mapred_output/term_frequency
[root@sandbox-hdp ~]# hdfs dfs -cp /user/root/tennis_fin_07/part-00000 /user/root/tennis_mapred_output/inv_doc_frequency
[root@sandbox-hdp ~]#
```

STEP 4: Using HIVE Tables to load TFIDF

a. Create new database called db

```
hive> create database db;
OK
Time taken: 2.268 seconds
hive> use db;
OK
```

b. Create and load table for storing term_frequency calculated from map reduce 1

Execution Code

create table term_frequency(word string, file_name string, word_count bigint) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

load data inpath '/user/root/tennis_mapred_output/term_frequency' overwrite into table db.term_frequency

```
hive> create table term_frequency(word string, file_name string, word_count bigint) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';
OK
Time taken: 1.888 seconds
hive> describe term_frequency;
OK
word                    string
file_name               string
word_count              bigint
Time taken: 0.526 seconds, Fetched: 3 row(s)
hive> load data inpath '/user/root/tennis_mapred_output/term_frequency' overwrite into table db.term_frequency;
Loading data to table db.term_frequency
chgrp: changing ownership of 'hdfs://sandbox-hdp.hortonworks.com:8020/apps/hive/warehouse/db.db/term_frequency/term_frequency': User null does not belong to hadoop
Table db.term_frequency stats: [numFiles=1, numRows=0, totalSize=1193615, rawDataSize=0]
OK
Time taken: 1.32 seconds
```

c. Create and load table for storing idf which was produced in map-reduce 2

Execution Code

create table idoc_frequency(word string, idf float) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

load data inpath '/user/root/tennis_mapred_output/inv_doc_frequency' overwrite into table db.idoc_frequency

```
hive> create table idoc_frequency(word string, idf float) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
    > ;
OK
Time taken: 0.664 seconds
hive> describe idoc_frequency;
OK
word                    string
idf                     float
Time taken: 0.471 seconds, Fetched: 2 row(s)
hive> load data inpath '/user/root/tennis_mapred_output/inv_doc_frequency' overwrite into table db.idoc_frequency;
Loading data to table db.idoc_frequency
chgrp: changing ownership of 'hdfs://sandbox-hdp.hortonworks.com:8020/apps/hive/warehouse/db.db/idoc_frequency/inv_doc_frequency': User null does not belong to hadoop
Table db.idoc_frequency stats: [numFiles=1, numRows=0, totalSize=42926, rawDataSize=0]
OK
Time taken: 1.14 seconds
```

d. Create new table for storing final tfidf index for all words

Execution Code

create table tfidf_index(word string,file_name string, tfidf float) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

```
hive> create table tfidf_index(word string,file_name string, tfidf float) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';
OK
```

e. **Load the result of join of two tables into TFIDF_index table**. This command joins the "term_frequency" table and "idoc_frequency" table using word as the commom key. The result will be in the format of

Table: TFIDF_INDEX		
Word	Document_name	TFIDF

Execution Code

insert overwrite table db.tfidf_index select TF_table.word, TF_table.file_name, (TF_table.word_count * IDF_table.idf) from term_frequency TF_table, idoc_frequency as IDF_table where TF_table.word = IDF_table.word

```
hive> insert overwrite table db.tfidf_index select TF_table.word, TF_table.file_name, (TF_table.word_count * IDF_table.idf) from term_frequency TF_table, idoc_frequency as IDF_
word = IDF_table.word
    > ;
Query ID = root_20181001210134_13b20192-fd1e-4f81-bb0e-43a67c848803
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1537877116120_0233)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 2 ..........   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 5.00 s
--------------------------------------------------------------------------------
Loading data to table db.tfidf_index
Table db.tfidf_index stats: [numFiles=1, numRows=16301, totalSize=1243107, rawDataSize=1226806]
OK
Time taken: 7.679 seconds
```

7. Map Reduce Code Snippet

Q1_Mapper.py

Code Explanation

This program is reading all the lines from the list of input files and producing term frequency document.

Line 13: Gets the name of file using python environment variable

Line 15: Removes all the special characters for cleaner dataset

Line 17: Converts all words into lower case which helps to produce case insensitive search

Line 20: Joins word+document_name as key. This will help in sorting and counting in reducer based on per word, per document

```
1  #!/usr/bin/env python
2
3  import sys
4  import os
5  import re
6
7  concat_two = None
8  for line in sys.stdin:
9      line = line.strip()
10     words = line.split()
11     for word in words:
12         # get the name of the file using below command
13         file_name = os.getenv('mapreduce_map_input_file')
14         # remove all the leading and trailing special characters
15         word = word.strip('%$.,;\'\"&|_\(\)')
16         # covert all into lower characters; useful in case insensitive search
17         word = word.lower()
18         # create word and file_name as keyword, this helps in sorting
19         if(len(word) > 0):
20             concat_two = str(word) +"|"+str(file_name)
21             print '%s\t%s' % (concat_two,1)
```

Q1_Reducer.py

Code Explanation

Reducer is counting the occurrence of each word under each document

Line 31,32: Once the count is done for a word+document_name, I am splitting into word and document_name

The output will be in the format of

Word	Document_name	Term Frequency

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

from operator import itemgetter
import sys
import os

current_word = None
current_count = 0
word = None

# input comes from STDIN
for line in sys.stdin:
    line = line.strip()

    word,count = line.split('\t', 1)

    # Referenced from Session3-Lab-MapReduce
    try:
        count = int(count)
    except ValueError:
        # count was not a number, so silently
        # ignore/discard this line
        continue


    if current_word == word:
        current_count += count
    else:
        if current_word:
            # split the keyword into word and file_name
            a,b = current_word.split("|")
            print '%s\t%s\t%s' % (a,b, current_count)
        current_count = count
        current_word = word

if current_word == word:
    # split the keyword into word and file_name
    a,b = current_word.split("|")
    print '%s\t%s\t%s' % (a,b, current_count)
```

Q2_Mapper.py

Code Explanation

This mapper takes the output of previous map-reduce job and outputs only distinct word from each document. This will help to count occurrence of each word across all documents.

```
1  #!/usr/bin/env python
2
3  import sys
4  import os
5
6  for line in sys.stdin:
7      line = line.strip()
8      words, file_name, count = line.split()
9      # output the word which is present across documents
10     # this word will be counted in reducer
11     print '%s\t%s' % (words, 1)
```

Q2_Reducer.py

The reducer performs two operations

- Calculating occurrences of each word in all documents
- Calculating Inverse Document Frequency

Line 31,32: Measures to handle log 0

The output will be in the format of

Word	Inverse Document Frequency

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

from operator import itemgetter
import sys
import math

current_word = None
current_count = 0
word = None
# Input the number of files in tennis folder
file_count=100
idf=0.00
# input comes from STDIN
for line in sys.stdin:
    line = line.strip()

    word, count = line.split('\t', 1)
    # Referenced from Session3-Lab-MapReduce
    try:
        count = float(count)
    except ValueError:
        # count was not a number, so silently
        # ignore/discard this line
        continue

    if current_word == word:
        current_count += count
    else:
        if current_word:
            # calculate the IDF
            idf = file_count/current_count
            if(idf != 0):
                idf = math.log(file_count/current_count)
            print '%s\t%.2f' % (current_word, idf)
        current_count = count
        current_word = word

if current_word == word:
    idf = file_count/current_count
    if(idf != 0):
        idf = math.log(file_count/current_count)
    print '%s\t%s' % (current_word, idf)
```