ASSIGNMENT_01: DATA ANALYSIS

1. Student details:

A. **Name:** Shreyash Shashikant Kumbhar
B. **Student ID:** 21263426
C. **Email ID:** shreyash.kumbhar2@mail.dcu.ie

2. Link for the Git repository:

https://github.com/SSHREYASH10/CLOUD_TECHNOLOGIES.git

3.

A. About the dataset:

"Stack Exchange is a network of question and answer websites on diverse topics in many different fields, each site covering a specific topic, where questions, answers, and users are subject to a reputation award process. The sites are modelled after Stack Overflow, a forum for computer programming questions that was the original site in this network."
(Information Source: https://en.wikipedia.org/wiki/Stack_Exchange)

B. Steps taken to acquire the dataset:

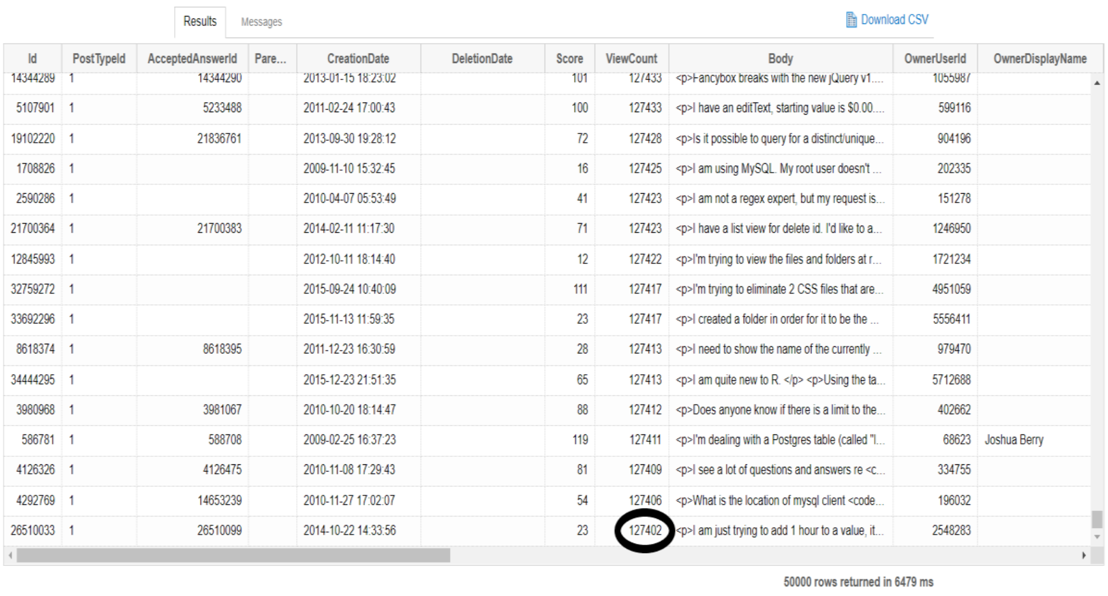
I am required to acquire the top 200,000 posts by ViewCount but the problem is that I can only download a maximum of 50,000 records at a time. This means that I would need to run at least 4 to 5 queries in total to obtain the 200,000 posts. The first thing to figure out would be the range of the values in the "ViewCount" field that constitutes the top 200,000 posts. After a series of attempts to find the lower bound value in "ViewCount" that accommodates the 200,000 records, I discover that the values in "ViewCount" greater than 41322 give me 200,011 data records and so it is safe to say that I can get at least 200,000 records by offering 41322 as my lower bound in "ViewCount". Therefore, on running the following query, I got the 200,000 posts' record.

select count(*) from posts where posts.ViewCount > 41322

Next, given that I can only obtain 50,000 records at a time, I can break down the whole range of "posts.ViewCount > 41322" into 4 parts, each of which has 50,000 records. In order to sort this out simple, I will arrange them in the descending order.

For top 25%
select top 50000 * from posts where posts.ViewCount > 41322 order by posts.ViewCount DESC

From the last few records fetched from the query, I can obtain the upper bound in "ViewCount" to use for the next query. For example, in the following screenshot, we can see that 127402 is the ViewCount value which can be utilized as the upper bound for the next query. (I have applied the same logic for the next queries as well)



Id	PostTypeId	AcceptedAnswerId	Pare...	CreationDate	DeletionDate	Score	ViewCount	Body	OwnerUserId	OwnerDisplayName
14344289	1	14344290		2013-01-15 18:23:02		101	127433	<p>Fancybox breaks with the new jQuery v1....	1055987	
5107901	1	5233488		2011-02-24 17:00:43		100	127433	<p>I have an editText, starting value is $0.00....	599116	
19102220	1	21836761		2013-09-30 19:28:12		72	127428	<p>Is it possible to query for a distinct/unique...	904196	
1708826	1			2009-11-10 15:32:45		16	127425	<p>I am using MySQL. My root user doesn't ...	202335	
2590286	1			2010-04-07 05:53:49		41	127423	<p>I am not a regex expert, but my request is...	151278	
21700364	1	21700383		2014-02-11 11:17:30		71	127423	<p>I have a list view for delete id. I'd like to a...	1246950	
12845993	1			2012-10-11 18:14:40		12	127422	<p>I'm trying to view the files and folders at r...	1721234	
32759272	1			2015-09-24 10:40:09		111	127417	<p>I'm trying to eliminate 2 CSS files that are...	4951059	
33692296	1			2015-11-13 11:59:35		23	127417	<p>I created a folder in order for it to be the ...	5556411	
8618374	1	8618395		2011-12-23 16:30:59		28	127413	<p>I need to show the name of the currently ...	979470	
34444295	1			2015-12-23 21:51:35		65	127413	<p>I am quite new to R. </p> <p>Using the ta...	5712688	
3980968	1	3981067		2010-10-20 18:14:47		88	127412	<p>Does anyone know if there is a limit to the...	402662	
586781	1	588708		2009-02-25 16:37:23		119	127411	<p>I'm dealing with a Postgres table (called "l...	68623	Joshua Berry
4126326	1	4126475		2010-11-08 17:29:43		81	127409	<p>I see a lot of questions and answers re <c...	334755	
4292769	1	14653239		2010-11-27 17:02:07		54	127406	<p>What is the location of mysql client <code...	196032	
26510033	1	26510099		2014-10-22 14:33:56		23	127402	<p>I am just trying to add 1 hour to a value, it...	2548283	

50000 rows returned in 6479 ms

For next 25%
select top 50000 * from posts where posts.ViewCount <= 127402 order by posts.ViewCount DESC

- This CSV file consists of **1 duplicate record** ------------ (1) with respect to the previous CSV file.

For next 25%
select top 50000 * from posts where posts.ViewCount <= 74596 order by posts.ViewCount DESC

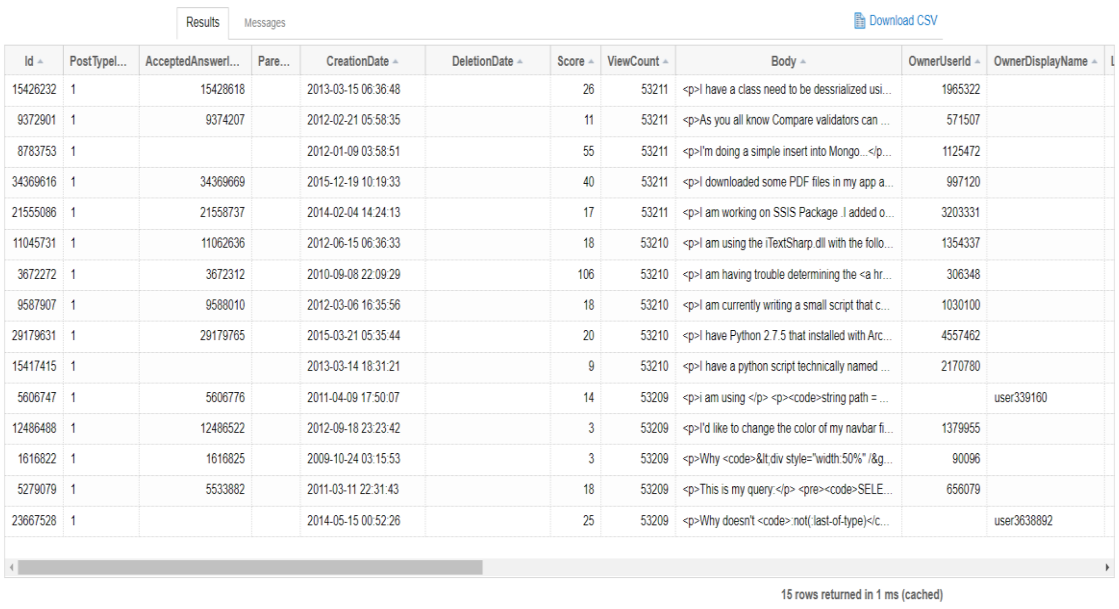
- This CSV file consists of **1 duplicate record** ------------ (2) with respect to the previous CSV file.
- It also consists of **1 empty record** which will be removed during the data cleaning process.

For next 25%
select top 50000 * from posts where posts.ViewCount <= 53211 order by posts.ViewCount DESC

- This CSV file consists of **4 duplicate record** ------------ (3) with respect to the previous CSV file.

Now, from (1), (2) and (3), we have **6 duplicate records** in total which will be removed during the data cleaning process and so we will **fall short of 6 records to complete the 200,000 records count**. Therefore, I am applying the following additional query which will give me 15 records (as shown in the following screenshot) of which **5 duplicate records** with respect to the previous CSV file and **the additional records after the 200,000 records count is completed** will be removed during the data cleaning process.

select top 50000 * from posts where posts.ViewCount >= 453209 and posts.ViewCount <= 41325 order by posts.ViewCount DESC



Results | Messages | Download CSV

Id	PostTypeI...	AcceptedAnswerI...	Pare...	CreationDate	DeletionDate	Score	ViewCount	Body	OwnerUserId	OwnerDisplayName	L
15426232	1	15428618		2013-03-15 06:36:48		26	53211	<p>I have a class need to be dessrialized usi...	1965322		
9372901	1	9374207		2012-02-21 05:58:35		11	53211	<p>As you all know Compare validators can ...	571507		
8783753	1			2012-01-09 03:58:51		55	53211	<p>I'm doing a simple insert into Mongo...</p...	1125472		
34369616	1	34369669		2015-12-19 10:19:33		40	53211	<p>I downloaded some PDF files in my app a...	997120		
21555086	1	21558737		2014-02-04 14:24:13		17	53211	<p>I am working on SSIS Package .I added o...	3203331		
11045731	1	11062636		2012-06-15 06:36:33		18	53210	<p>I am using the iTextSharp.dll with the follo...	1354337		
3672272	1	3672312		2010-09-08 22:09:29		106	53210	<p>I am having trouble determining the <a hr...	306348		
9587907	1	9588010		2012-03-06 16:35:56		18	53210	<p>I am currently writing a small script that c...	1030100		
29179631	1	29179765		2015-03-21 05:35:44		20	53210	<p>I have Python 2.7.5 that installed with Arc...	4557462		
15417415	1			2013-03-14 18:31:21		9	53210	<p>I have a python script technically named ...	2170780		
5606747	1	5606776		2011-04-09 17:50:07		14	53209	<p>i am using </p> <p><code>string path = ...		user339160	
12486488	1	12486522		2012-09-18 23:23:42		3	53209	<p>I'd like to change the color of my navbar fi...	1379955		
1616822	1	1616825		2009-10-24 03:15:53		3	53209	<p>Why <code>&lt;div style="width:50%" /&g...	90096		
5279079	1	5533882		2011-03-11 22:31:43		18	53209	<p>This is my query:</p> <pre><code>SELE...	656079		
23667528	1			2014-05-15 00:52:26		25	53209	<p>Why doesn't <code>:not(:last-of-type)</c...		user3638892	

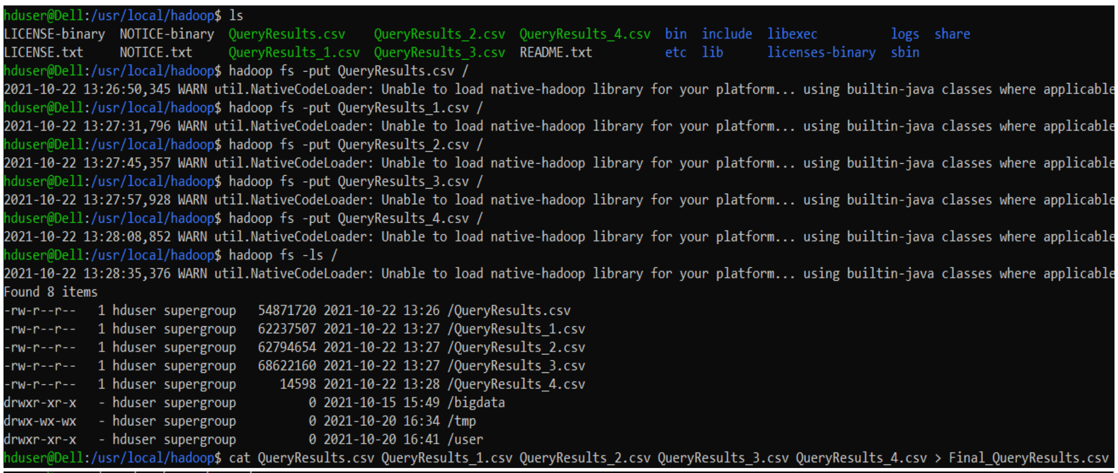
15 rows returned in 1 ms (cached)

4. Steps taken to:

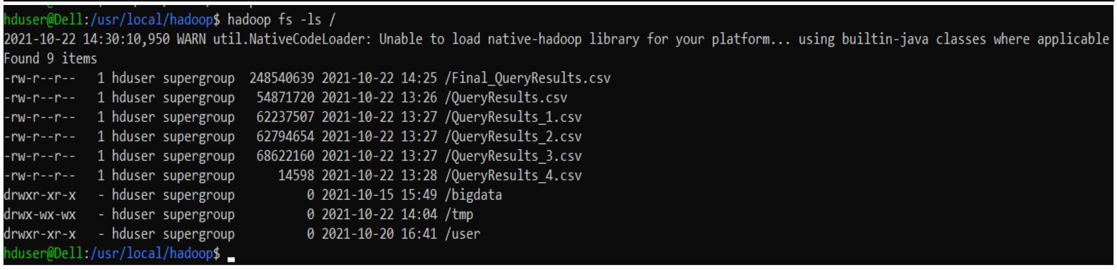
A. Load and merge the data in HDFS:

Firstly, upload the 5 downloaded CSV files in the local home directory and check whether the same are present in it using the ls command. Then, move the files on Hadoop using the put command and check for their presence. Lastly, merge all the 5 CSV files into a single CSV file using the cat command and again check whether that single CSV file is present in it.

```
hduser@Dell:/usr/local/hadoop$ ls

hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults.csv /

hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults_1.csv /

hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults_2.csv /

hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults_3.csv /

hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults_4.csv /

hduser@Dell:/usr/local/hadoop$ hadoop fs -ls /

hduser@Dell:/usr/local/hadoop$ cat QueryResults.csv QueryResults_1.csv QueryResults_2.csv QueryResults_3.csv QueryResults_4.csv > Final_QueryResults.csv

hduser@Dell:/usr/local/hadoop$ hadoop fs -ls /
```



```
hduser@Dell:/usr/local/hadoop$ ls
LICENSE-binary  NOTICE-binary  QueryResults.csv    QueryResults_2.csv  QueryResults_4.csv  bin  include  libexec          logs  share
LICENSE.txt     NOTICE.txt     QueryResults_1.csv  QueryResults_3.csv  README.txt          etc  lib      licenses-binary  sbin
hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults.csv /
2021-10-22 13:26:50,345 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults_1.csv /
2021-10-22 13:27:31,796 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults_2.csv /
2021-10-22 13:27:45,357 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults_3.csv /
2021-10-22 13:27:57,928 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@Dell:/usr/local/hadoop$ hadoop fs -put QueryResults_4.csv /
2021-10-22 13:28:08,852 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@Dell:/usr/local/hadoop$ hadoop fs -ls /
2021-10-22 13:28:35,376 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 8 items
-rw-r--r--   1 hduser supergroup   54871720 2021-10-22 13:26 /QueryResults.csv
-rw-r--r--   1 hduser supergroup   62237507 2021-10-22 13:27 /QueryResults_1.csv
-rw-r--r--   1 hduser supergroup   62794654 2021-10-22 13:27 /QueryResults_2.csv
-rw-r--r--   1 hduser supergroup   68622160 2021-10-22 13:27 /QueryResults_3.csv
-rw-r--r--   1 hduser supergroup      14598 2021-10-22 13:28 /QueryResults_4.csv
drwxr-xr-x   - hduser supergroup          0 2021-10-15 15:49 /bigdata
drwx-wx-wx   - hduser supergroup          0 2021-10-20 16:34 /tmp
drwxr-xr-x   - hduser supergroup          0 2021-10-20 16:41 /user
hduser@Dell:/usr/local/hadoop$ cat QueryResults.csv QueryResults_1.csv QueryResults_2.csv QueryResults_3.csv QueryResults_4.csv > Final_QueryResults.csv
```



```
hduser@Dell:/usr/local/hadoop$ hadoop fs -ls /
2021-10-22 14:30:10,950 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 9 items
-rw-r--r--   1 hduser supergroup  248540639 2021-10-22 14:25 /Final_QueryResults.csv
-rw-r--r--   1 hduser supergroup   54871720 2021-10-22 13:26 /QueryResults.csv
-rw-r--r--   1 hduser supergroup   62237507 2021-10-22 13:27 /QueryResults_1.csv
-rw-r--r--   1 hduser supergroup   62794654 2021-10-22 13:27 /QueryResults_2.csv
-rw-r--r--   1 hduser supergroup   68622160 2021-10-22 13:27 /QueryResults_3.csv
-rw-r--r--   1 hduser supergroup      14598 2021-10-22 13:28 /QueryResults_4.csv
drwxr-xr-x   - hduser supergroup          0 2021-10-15 15:49 /bigdata
drwx-wx-wx   - hduser supergroup          0 2021-10-22 14:04 /tmp
drwxr-xr-x   - hduser supergroup          0 2021-10-20 16:41 /user
hduser@Dell:/usr/local/hadoop$ _
```

B. Clean the data in PIG:

Firstly, load the data from HDFS to PIG, specifying each data type.

```
grunt>stackdata=LOAD'hdfs://localhost:9870/usr/local/hadoopFinal_QueryResults.csv' USING PigStorage (',') AS (id:int, posttypeid:int, acceptedanswerid:int, parentid:int, creationdate:chararray, deletiondate:chararray, score:int, viewcount:int, body:chararray, owneruserid:int, ownerdisplayname:chararray, lasteditoruserId:int, lasteditordisplayname:chararray, lasteditdate:chararray, lastactivitydate:chararray, title:chararray, tags:chararray, answercount:int, commentcount:int, favoritecount:int, closeddate:chararray, communityowneddate:chararray );
```

```
grunt> DESCRIBE stackdata;
```

Now, we do not need every field present in the data. Therefore, we generate a new table with the required fields only and also remove the empty records present in the data.

```
grunt> pickCols = FOREACH stackdata GENERATE id, score, viewcount, owneruserid, title, tags, (REPLACE(body,'[\r\n]+',' ')) AS body;
```

```
grunt> DESCRIBE pickCols;
```

As mentioned earlier, we need to remove the duplicate records present in the data. So, removing the same using the DISTINCT function.

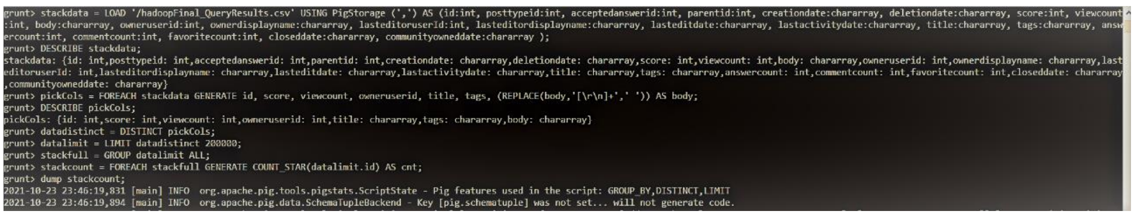
```
grunt> datadistinct = DISTINCT pickCols;
```

Also, the additional records after the 200,000 records count is completed need to be removed. Therefore, removing the same using the LIMIT function and thereafter reverifying the count of the remaining records using the COUNT_STAR function which return the value 200,000!

```
grunt> datalimit = LIMIT datadistinct 200000;

grunt> stackfull = GROUP datalimit ALL;

grunt> stackcount = FOREACH stackfull GENERATE COUNT_STAR(datalimit.id) AS cnt;

grunt> dump stackcount;
```

Lastly, creating a new file folder to save this cleaned up data.

```
grunt> STORE datalimit INTO 'Pig_QueryResults' USING PigStorage(',');
```

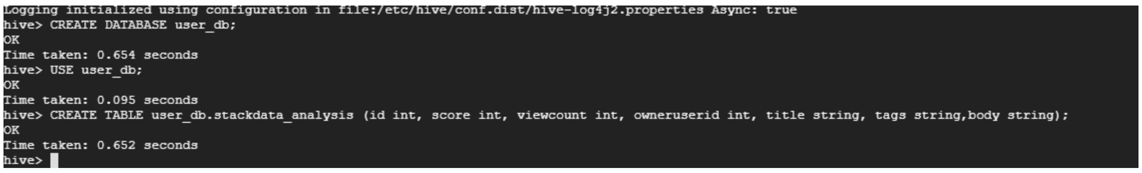


```
grunt> stackdata = LOAD '/hadoopFinal_QueryResults.csv' USING PigStorage (',') AS (id:int, posttypeid:int, acceptedanswerid:int, parentid:int, creationdate:chararray, deletiondate:chararray, score:int, viewcount
:int, body:chararray, owneruserid:int, ownerdisplayname:chararray, lasteditoruserId:int, lasteditordisplayname:chararray, lasteditdate:chararray, lastactivitydate:chararray, title:chararray, tags:chararray, answ
ercount:int, commentcount:int, favoritecount:int, closeddate:chararray, communityowneddate:chararray );
grunt> DESCRIBE stackdata;
stackdata: {id: int,posttypeid: int,acceptedanswerid: int,parentid: int,creationdate: chararray,deletiondate: chararray,score: int,viewcount: int,body: chararray,owneruserid: int,ownerdisplayname: chararray,last
editoruserId: int,lasteditordisplayname: chararray,lasteditdate: chararray,lastactivitydate: chararray,title: chararray,tags: chararray,answercount: int,commentcount: int,favoritecount: int,closeddate: chararray
,communityowneddate: chararray}
grunt> pickCols = FOREACH stackdata GENERATE id, score, viewcount, owneruserid, title, tags, (REPLACE(body,'[\r\n]+',' ')) AS body;
grunt> DESCRIBE pickCols;
pickCols: {id: int,score: int,viewcount: int,owneruserid: int,title: chararray,tags: chararray,body: chararray}
grunt> datadistinct = DISTINCT pickCols;
grunt> datalimit = LIMIT datadistinct 200000;
grunt> stackfull = GROUP datalimit ALL;
grunt> stackcount = FOREACH stackfull GENERATE COUNT_STAR(datalimit.id) AS cnt;
grunt> dump stackcount;
2021-10-23 23:46:19,831 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP_BY,DISTINCT,LIMIT
2021-10-23 23:46:19,894 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not generate code.
```

C. Query the data using HIVE:

In order to load the cleaned data from PIG to HIVE, we create a database and a table in HIVE.

```
hive> CREATE DATABASE user_db;

hive> USE user_db;

hive> CREATE TABLE user_db.stackdata_analysis (id int, score int, viewcount int, owneruserid int, ownerusername string, title string, tags string, body string);
```



```
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: true
hive> CREATE DATABASE user_db;
OK
Time taken: 0.654 seconds
hive> USE user_db;
OK
Time taken: 0.095 seconds
hive> CREATE TABLE user_db.stackdata_analysis (id int, score int, viewcount int, owneruserid int, title string, tags string,body string);
OK
Time taken: 0.652 seconds
hive>
```

Now, we can transfer the cleaned data from PIG into the above created table in HIVE.

```
hive> STORE Pig_QueryResults INTO 'user_db.stackdata_analysis' USING org.apache.hive.hcatalog.pig.HCatStorer();
```

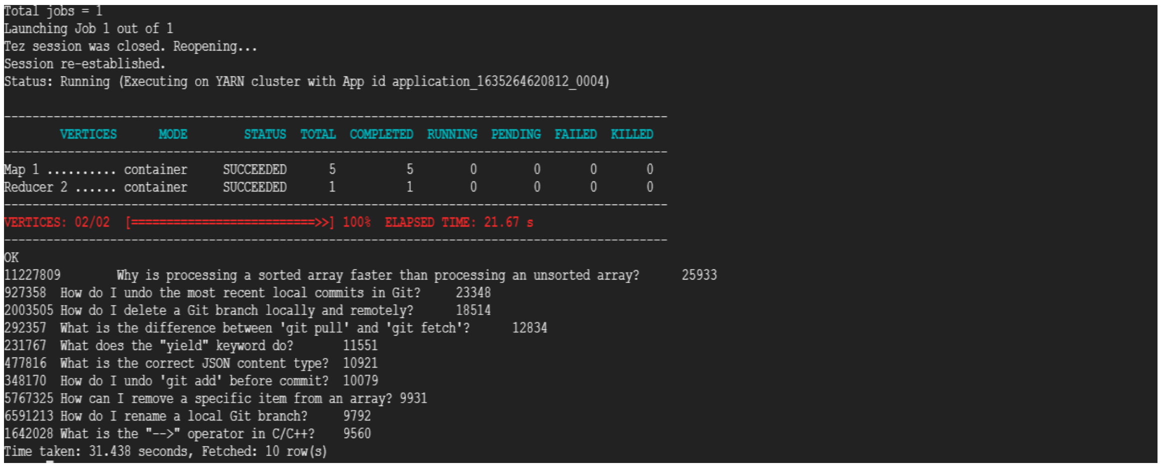
In order to verify that the data has been transferred properly, we check the count of the 'id' field which returns the value 200,000!

```
hive> SELECT COUNT(id) FROM user_db.stackdata_analysis;
```

Queries for:

- The top 10 posts by score

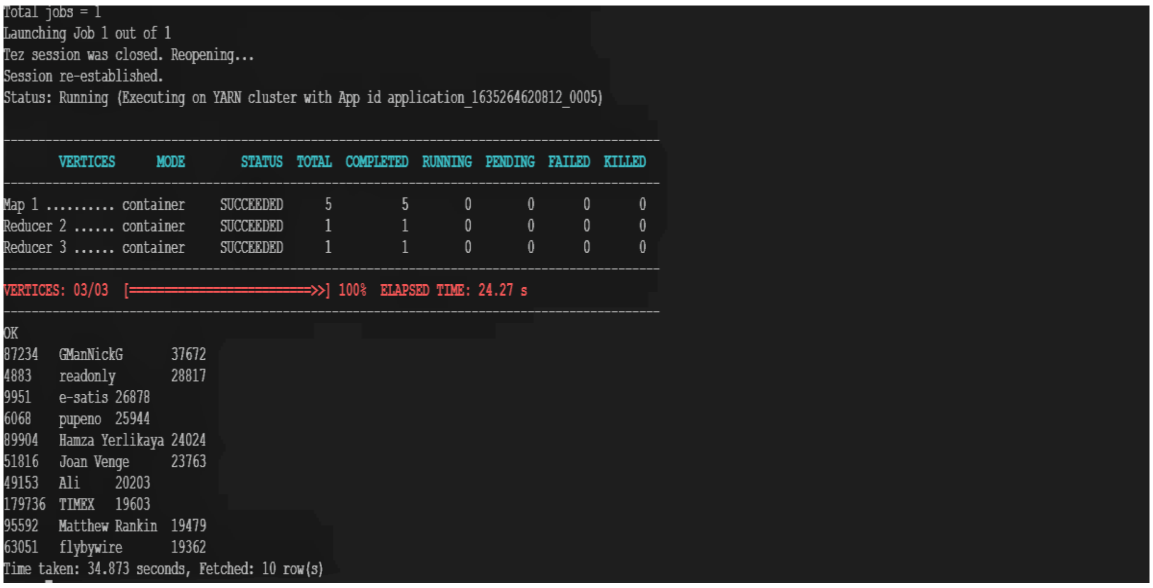
  ```
  hive> SELECT id, title, score FROM user_db.stackdata_analysis ORDER BY score
  DESC LIMIT 10;
  ```



```
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1635264620812_0004)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      5          5        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 21.67 s
----------------------------------------------------------------------------------------------
OK
11227809        Why is processing a sorted array faster than processing an unsorted array?      25933
927358  How do I undo the most recent local commits in Git?     23348
2003505 How do I delete a Git branch locally and remotely?      18514
292357  What is the difference between 'git pull' and 'git fetch'?      12834
231767  What does the "yield" keyword do?       11551
477816  What is the correct JSON content type?  10921
348170  How do I undo 'git add' before commit?  10079
5767325 How can I remove a specific item from an array? 9931
6591213 How do I rename a local Git branch?     9792
1642028 What is the "-->" operator in C/C++?    9560
Time taken: 31.438 seconds, Fetched: 10 row(s)
```

- The top 10 users by post score

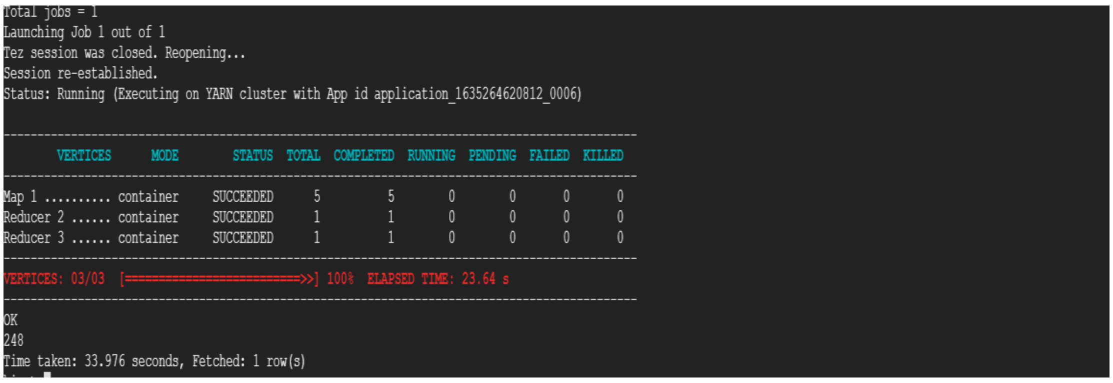
  ```
  hive> SELECT owneruserid AS USERID, ownerusername, SUM(score) AS SCORE
  FROM user_db.stackdata_analysis GROUP BY owneruserid having owneruserid
  is not null SORT BY score DESC LIMIT 10;
  ```



```
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1635264620812_0005)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      5          5        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 24.27 s
----------------------------------------------------------------------------------------------
OK
87234   GManNickG       37672
4883    readonly        28817
9951    e-satis 26878
6068    pupeno  25944
89904   Hamza Yerlikaya 24024
51816   Joan Venge      23763
49153   Ali     20203
179736  TIMEX   19603
95592   Matthew Rankin  19479
63051   flybywire       19362
Time taken: 34.873 seconds, Fetched: 10 row(s)
```

- The number of distinct users, who used the word “cloud” in one of their posts

  hive> SELECT COUNT(DISTINCT owneruserid) FROM user_db.stackdata_analysis WHERE UPPER(body) LIKE '%CLOUD%' OR UPPER(title) LIKE '%CLOUD%' OR LOWER(body) LIKE '%cloud%' OR UPPER(tags) LIKE '%CLOUD%' OR LOWER(title) LIKE '%cloud%' OR LOWER(tags) LIKE '%CLOUD%';



```
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1635264620812_0006)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      5          5        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 23.64 s
----------------------------------------------------------------------------------------------
OK
248
Time taken: 33.976 seconds, Fetched: 1 row(s)
```

D. Calculate TF-IDF with HIVE:

Creating a table with the columns required to calculate TF-IDF and putting it into the local.

hive> create table user_db.grouped_users_posts as select owneruserid as a, body as b, SUM(score) as c from user_db.stackdata_analysis group by owneruserid,body;

```
hive> INSERT OVERWRITE LOCAL DIRECTORY '/home/hduser/tfidfdata'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
select owneruserid, body from user_db.stackdata_analysis where owneruserid in
(select id from user_db.grouped_users_posts);
```

The MapReduce program needs the input file to be separated by space rather than comma. Therefore, replacing the comma with spaces and then moving the file into a folder in the local from HDFS.

```
> cd tfidfdata
> sed 's/,/ /g' 000000_0 > inputfile

> hadoop fs -mkdir /mappred
> hadoop fs -put inputfile /mappred
```

Now, the Implementation of TF-IDF in Hadoop using Python will be in three phases using three mappers and three reducers. Using the following commands to run TF-IDF on the cluster.

```
> hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -file /home/hduser/MAPPER_01.py /home/hduser/REDUCER_01.py -mapper "python MAPPER_01.py" -reducer "python REDUCER_01.py" -input hdfs://cluster-3299-m/mapinput/inputfile -output hdfs://cluster-3299-m/mappred1
```

```
> hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -file /home/hduser/MAPPER_02.py /home/hduser/REDUCER_02.py -mapper "python MAPPER_02.py" -reducer "python REDUCER_02.py" -input hdfs://cluster-3299-m/mappred1/part-00000 hdfs://cluster-3299-m/mappred1/part-00001 hdfs://cluster-3299-m/mappred1/part-00002 hdfs://cluster-3299-m/mappred1/part-00003 hdfs://cluster-3299-m/mappred1/part-00004 -output hdfs://cluster-3299-m/mappred2
```

```
> hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -file /home/hduser/MAPPER_03.py /home/hduser/REDUCER_03.py -mapper "python MAPPER_03.py" -reducer "python REDUCER_03.py" -input hdfs://cluster-3299-m/mappred2/part-00000 hdfs://cluster-3299-m/mappred2/part-00001 hdfs://cluster-3299-m/mappred2/part-00002 hdfs://cluster-3299-m/mappred2/part-00003 hdfs://cluster-3299-m/mappred2/part-00004 -output hdfs://cluster-3299-m/mappredf
```

Using the hadoop fs -getmerge command to merge the output files into a single CSV file (tfidfout.csv). Now, replacing the spaces and saving the file into another CSV file(tfidfout1.csv).

```
> sed -e 's/\s/,/g' tfidfout.csv > tfidfout1.csv
```

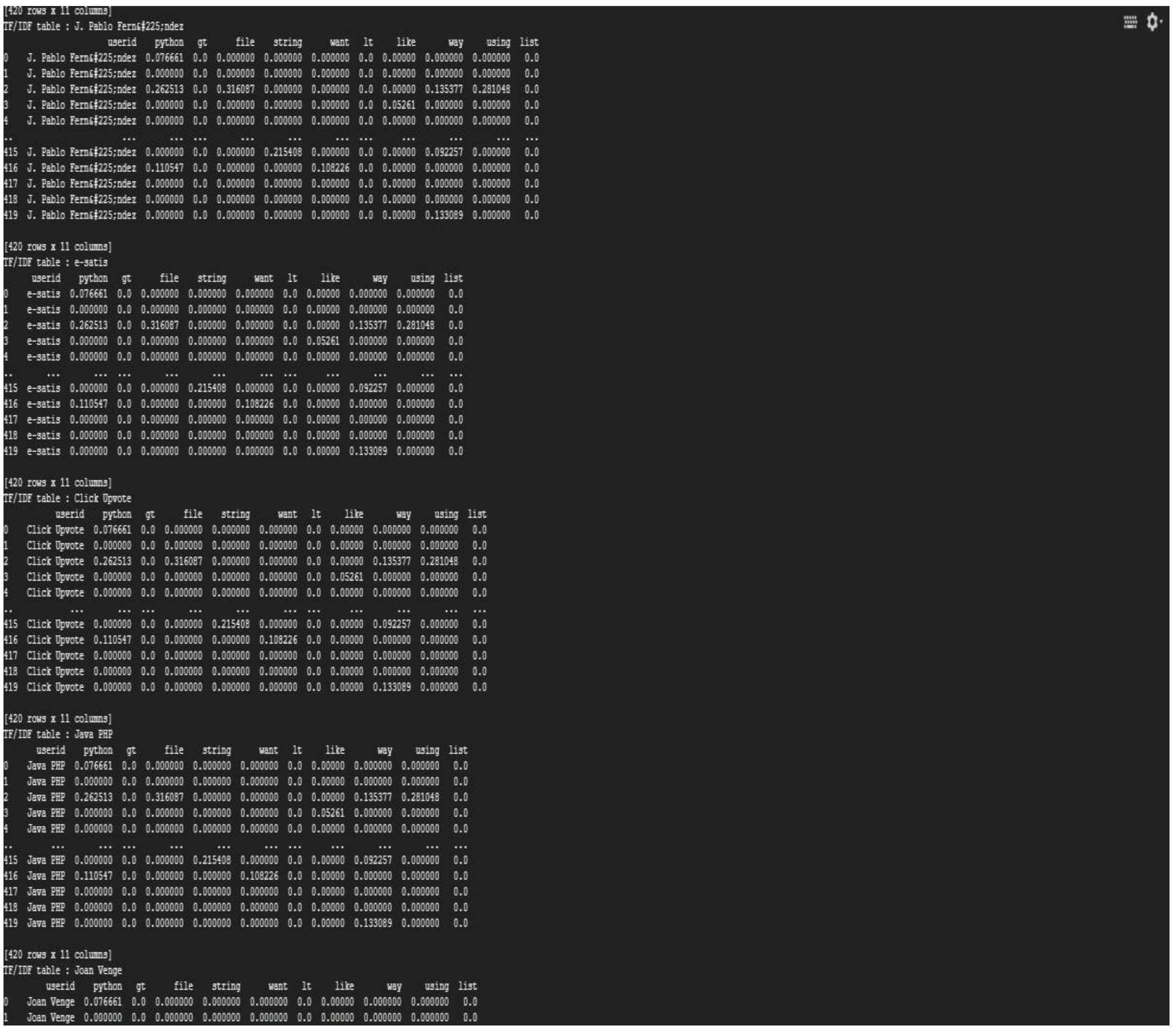
Thereafter, creating an external table in HIVE and loading the CSV file into the table.

```
hive> create external table if not exists TFIDF_data2 (Term String, Id int, tfidf float)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

```
hive> load data local inpath 'tfidfout1.csv' overwrite into table TFIDF_data2;
```

Finally, running the following query to calculate the per-user TF-IDF of the top 10 terms for each of the top 10 users.

```
hive> SELECT *
FROM (
SELECT ROW_NUMBER()
OVER(PARTITION BY Id
ORDER BY tfidf DESC) AS TfidfRank, *
FROM TfIDF_data2) n
WHERE TfidfRank IN (1,2,3,4,5,6,7,8,9,10);
```



```
[420 rows x 11 columns]
TF/IDF table : J. Pablo Fern&#225;ndez
                      userid    python   gt      file    string      want   lt     like       way     using  list
0    J. Pablo Fern&#225;ndez  0.076661  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
1    J. Pablo Fern&#225;ndez  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
2    J. Pablo Fern&#225;ndez  0.262513  0.0  0.316087  0.000000  0.000000  0.0  0.00000  0.135377  0.281048   0.0
3    J. Pablo Fern&#225;ndez  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.05261  0.000000  0.000000   0.0
4    J. Pablo Fern&#225;ndez  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
..                       ...       ...  ...       ...       ...       ...  ...      ...       ...       ...   ...
415  J. Pablo Fern&#225;ndez  0.000000  0.0  0.000000  0.215408  0.000000  0.0  0.00000  0.092257  0.000000   0.0
416  J. Pablo Fern&#225;ndez  0.110547  0.0  0.000000  0.000000  0.108226  0.0  0.00000  0.000000  0.000000   0.0
417  J. Pablo Fern&#225;ndez  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
418  J. Pablo Fern&#225;ndez  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
419  J. Pablo Fern&#225;ndez  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.133089  0.000000   0.0

[420 rows x 11 columns]
TF/IDF table : e-satis
       userid    python   gt      file    string      want   lt     like       way     using  list
0    e-satis  0.076661  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
1    e-satis  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
2    e-satis  0.262513  0.0  0.316087  0.000000  0.000000  0.0  0.00000  0.135377  0.281048   0.0
3    e-satis  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.05261  0.000000  0.000000   0.0
4    e-satis  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
..       ...       ...  ...       ...       ...       ...  ...      ...       ...       ...   ...
415  e-satis  0.000000  0.0  0.000000  0.215408  0.000000  0.0  0.00000  0.092257  0.000000   0.0
416  e-satis  0.110547  0.0  0.000000  0.000000  0.108226  0.0  0.00000  0.000000  0.000000   0.0
417  e-satis  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
418  e-satis  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
419  e-satis  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.133089  0.000000   0.0

[420 rows x 11 columns]
TF/IDF table : Click Upvote
            userid    python   gt      file    string      want   lt     like       way     using  list
0    Click Upvote  0.076661  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
1    Click Upvote  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
2    Click Upvote  0.262513  0.0  0.316087  0.000000  0.000000  0.0  0.00000  0.135377  0.281048   0.0
3    Click Upvote  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.05261  0.000000  0.000000   0.0
4    Click Upvote  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
..            ...       ...  ...       ...       ...       ...  ...      ...       ...       ...   ...
415  Click Upvote  0.000000  0.0  0.000000  0.215408  0.000000  0.0  0.00000  0.092257  0.000000   0.0
416  Click Upvote  0.110547  0.0  0.000000  0.000000  0.108226  0.0  0.00000  0.000000  0.000000   0.0
417  Click Upvote  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
418  Click Upvote  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
419  Click Upvote  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.133089  0.000000   0.0

[420 rows x 11 columns]
TF/IDF table : Java PHP
        userid    python   gt      file    string      want   lt     like       way     using  list
0    Java PHP  0.076661  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
1    Java PHP  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
2    Java PHP  0.262513  0.0  0.316087  0.000000  0.000000  0.0  0.00000  0.135377  0.281048   0.0
3    Java PHP  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.05261  0.000000  0.000000   0.0
4    Java PHP  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
..        ...       ...  ...       ...       ...       ...  ...      ...       ...       ...   ...
415  Java PHP  0.000000  0.0  0.000000  0.215408  0.000000  0.0  0.00000  0.092257  0.000000   0.0
416  Java PHP  0.110547  0.0  0.000000  0.000000  0.108226  0.0  0.00000  0.000000  0.000000   0.0
417  Java PHP  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
418  Java PHP  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
419  Java PHP  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.133089  0.000000   0.0

[420 rows x 11 columns]
TF/IDF table : Joan Venge
          userid    python   gt      file    string      want   lt     like       way     using  list
0    Joan Venge  0.076661  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
1    Joan Venge  0.000000  0.0  0.000000  0.000000  0.000000  0.0  0.00000  0.000000  0.000000   0.0
```

❖ To conclude, I have used all the 3 technologies for the following tasks:

1. HDFS: Load and merge the data
2. PIG: Clean the data
3. HIVE: Query the data and calculate TF-IDF

My concept behind using all the 3 technologies for various tasks was simply to get use to all the 3 platforms irrespective of the advantageous of one platform over the others and thereby get good hands-on!

5. REFERENCES:

1. https://data.stackexchange.com/stackoverflow/query/new
2. https://www.guru99.com/file-permissions.html
3. https://pig.apache.org/docs/r0.17.0/api/org/apache/pig/piggybank/storage/CSVExcelStorage.html
4. https://www.geeksforgeeks.org/tf-idf-model-for-page-ranking/
5. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html
6. https://www.projectpro.io/article/mapreduce-vs-pig-vs-hive/163