**CA675 Cloud Technologies**

**Assignment – 1**

|  |  |
| --- | --- |
| **Name** | Haresh Jairam Bhanushali |
| **Student ID** | 20210139 |
| **Email** | [haresh.bhanushali2@mail.dcu.ie](mailto:%68ar%65%73%68%2eb%68an%75s%68a%6c%69%32@%6d%61il%2ed%63u%2e%69%65) |
| **Git Repository** | https://github.com/hareshbhanushali1998/Cloud\_assignment\_1 |

**Task 1 - Acquire the top 200,000 posts by ViewCount**

The first task of the assignment was to acquire data from the StackExchange website. The total data to be acquired was 200,000 based on ViewCount. I have used the Data Explorer feature of StackExchange to run the query and acquire the data. I first wrote the query for getting the count of the rows in data based on limits set for ViewCount.

1. select count(\*) from posts where posts.ViewCount > 1000000

The above query was used to get the count of rows based on ViewCount greater than 1000000. Similarly, it was used with different counts.

1. Now to acquire 200,000 data I executed a query 4 times with different ViewCounts condition extracting 50,000 data each time.
2. select top 50000 \* from posts where posts.ViewCount > 127400 ORDER BY posts.ViewCount desc

The above query gives 50,000 results for ViewCount greater than 127400 in the descending order which can help us to work on data from higher ViewCount to lower while merging.

1. select top 50000 \* from posts where posts.ViewCount < 127400 ORDER BY posts.ViewCount desc

The above query gives another 50,000 data for ViewCount less than 127400. As we have already acquired data above that now we want the top 50,000 data starting from 127400.

1. select top 50000 \* from posts where posts.ViewCount < 74595 ORDER BY posts.ViewCount desc

Again, in this query I gave the condition less than 74595 as we already acquired data till 74595 from the previous query.

1. select top 50000 \* from posts where posts.ViewCount < 53210 ORDER BY posts.ViewCount desc

In the final query the condition was less than 53210 and I got the top 50,000 data from that ViewCount.

So, in total I acquired 4 csv files each containing 50,000 data.

**Using Google Cloud Platform**

To carry out my assignment I have used Google Cloud Platform (GCP). I made a cluster with ubuntu as image version. I have executed all my tasks in GCP.

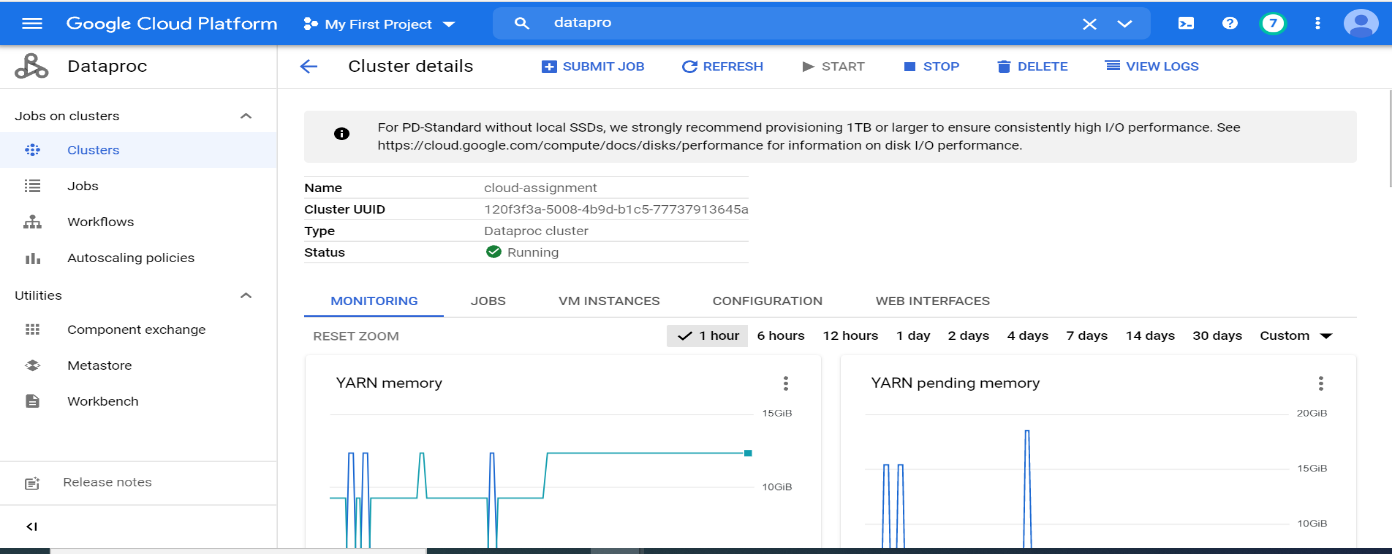


Figure 1: Cluster in Google Cloud Platform

**Task 2 – Loading and cleaning data**

1. I have used Google Cloud Platform to perform my tasks. I created a cluster named cloud-assignment and wrote my code in it.
2. The first step after starting shell was that I uploaded all my csv files downloaded while extracting data.
3. I read and studied my data first. It contained html tags and other punctuations which needs to be removed for further processing.
4. I tried cleaning my data using sed and awk. I have uploaded screenshot of the same below.

GitHub Link: https://github.com/hareshbhanushali1998/Cloud\_assignment\_1/blob/main/Data\_cleaning\_pig.txt

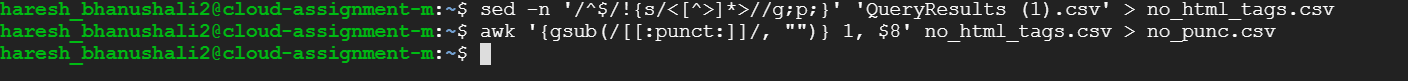


Figure 2: Using sed and awk for removing html tags and punctuations

1. It didn’t give any error but the data was not cleaned as per need. So, I also used Python to clean the data. Removing null values and columns is performed in Pig. I have attached the screenshots of Python code as well.

Github link: https://github.com/hareshbhanushali1998/Cloud\_assignment\_1/blob/main/Data\_Cleaning\_Python.ipynb



Figure 3: Removing html tags using Python

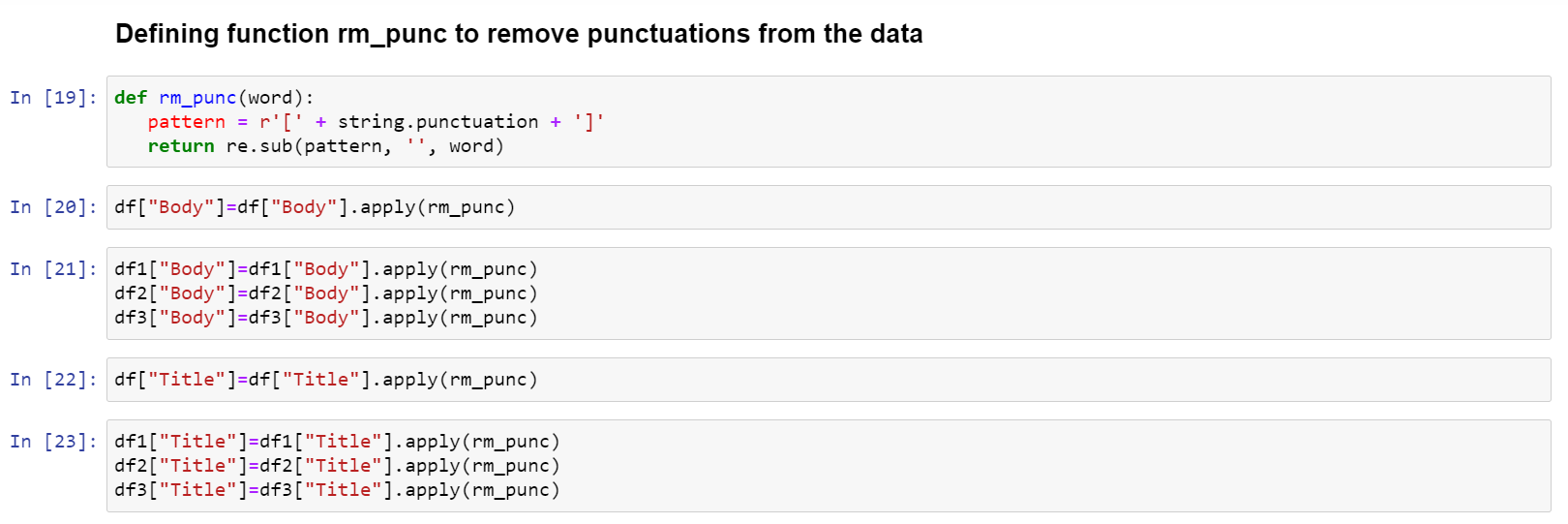


Figure 4: Removing punctuations using Python

1. For cleaning data in Python I have defined function rm\_tags for removing html tags and rm\_punc for removing punctuations.
2. For both the functions I have downloaded necessary libraries such as beautifulsoup for working on html tags and regular expression module (re) for punctuations.
3. After cleaning the data, I downloaded the clean data as csv file and text file as well.
4. I uploaded all the clean data csv file in the shell.
5. After uploading the csv file, I loaded the pig. I have uploaded the screenshot.

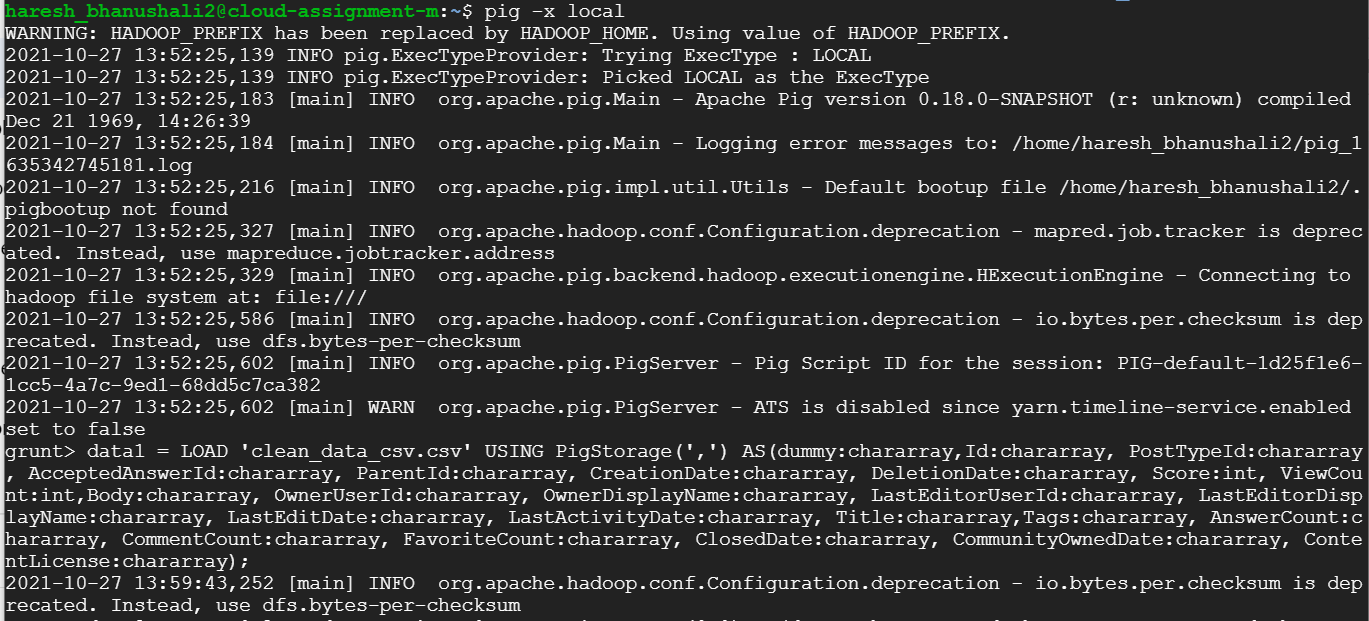


Figure 5: Loading pig

1. I loaded all the csv file in pig using LOAD command.

GitHub link: https://github.com/hareshbhanushali1998/Cloud\_assignment\_1/blob/main/Pig\_Query.txt

data1 = LOAD 'clean\_data\_csv.csv' USING PigStorage(',') AS(dummy:chararray,Id:chararray, PostTypeId:chararray, AcceptedAnswerId:chararray, ParentId:chararray, CreationDate:chararray, DeletionDate:chararray, Score:int, ViewCount:int,Body:chararray, OwnerUserId:chararray, OwnerDisplayName:chararray, LastEditorUserId:chararray, LastEditorDisplayName:chararray, LastEditDate:chararray, LastActivityDate:chararray, Title:chararray,Tags:chararray, AnswerCount:chararray, CommentCount:chararray, FavoriteCount:chararray, ClosedDate:chararray, CommunityOwnedDate:chararray, ContentLicense:chararray);

data2 = LOAD 'clean\_data\_csv1.csv' USING PigStorage(',') AS(dummy:chararray,Id:chararray, PostTypeId:chararray, AcceptedAnswerId:chararray, ParentId:chararray, CreationDate:chararray, DeletionDate:chararray, Score:int, ViewCount:int,Body:chararray, OwnerUserId:chararray, OwnerDisplayName:chararray, LastEditorUserId:chararray, LastEditorDisplayName:chararray, LastEditDate:chararray, LastActivityDate:chararray, Title:chararray,Tags:chararray, AnswerCount:chararray, CommentCount:chararray, FavoriteCount:chararray, ClosedDate:chararray, CommunityOwnedDate:chararray, ContentLicense:chararray);

data3 = LOAD 'clean\_data\_csv2.csv' USING PigStorage(',') AS(dummy:chararray,Id:chararray, PostTypeId:chararray, AcceptedAnswerId:chararray, ParentId:chararray, CreationDate:chararray, DeletionDate:chararray, Score:int, ViewCount:int,Body:chararray, OwnerUserId:chararray, OwnerDisplayName:chararray, LastEditorUserId:chararray, LastEditorDisplayName:chararray, LastEditDate:chararray, LastActivityDate:chararray, Title:chararray,Tags:chararray, AnswerCount:chararray, CommentCount:chararray, FavoriteCount:chararray, ClosedDate:chararray, CommunityOwnedDate:chararray, ContentLicense:chararray);

data4 = LOAD 'clean\_data\_csv3.csv' USING PigStorage(',') AS(dummy:chararray,Id:chararray, PostTypeId:chararray, AcceptedAnswerId:chararray, ParentId:chararray, CreationDate:chararray, DeletionDate:chararray, Score:int, ViewCount:int,Body:chararray, OwnerUserId:chararray, OwnerDisplayName:chararray, LastEditorUserId:chararray, LastEditorDisplayName:chararray, LastEditDate:chararray, LastActivityDate:chararray, Title:chararray,Tags:chararray, AnswerCount:chararray, CommentCount:chararray, FavoriteCount:chararray, ClosedDate:chararray, CommunityOwnedDate:chararray, ContentLicense:chararray);

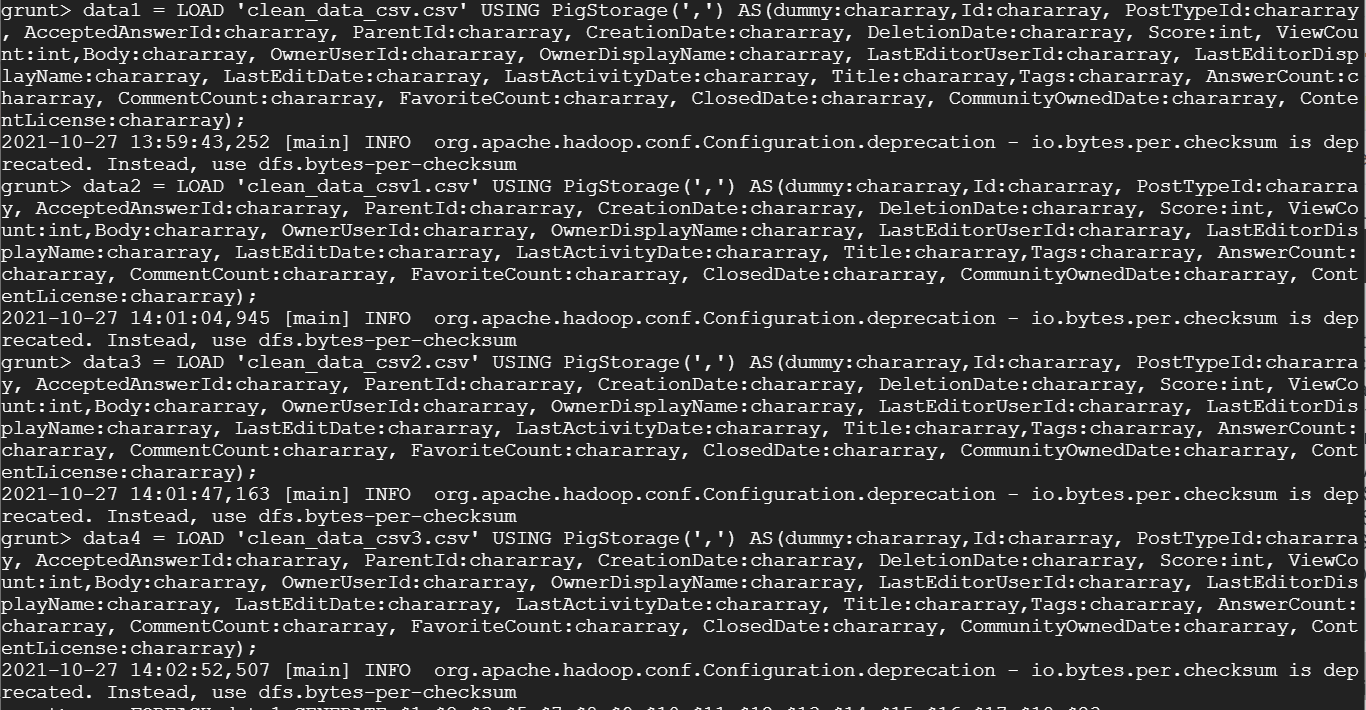


Figure 6: Loading data in Pig

1. After loading data using load command, I filtered out the columns and removed some columns such as ParentId, DeletionDate, CommentCount, FavoriteCount, ClosedDate and CommunityOwnedDate.

a = FOREACH data1 GENERATE $1,$2,$3,$5,$7,$8,$9,$10,$11,$12,$13,$14,$15,$16,$17,$18,$23;

b = FOREACH data2 GENERATE $1,$2,$3,$5,$7,$8,$9,$10,$11,$12,$13,$14,$15,$16,$17,$18,$23;

c = FOREACH data3 GENERATE $1,$2,$3,$5,$7,$8,$9,$10,$11,$12,$13,$14,$15,$16,$17,$18,$23;

d = FOREACH data4 GENERATE $1,$2,$3,$5,$7,$8,$9,$10,$11,$12,$13,$14,$15,$16,$17,$18,$23;

1. Now, I have merged all the files together in new variable using UNION.

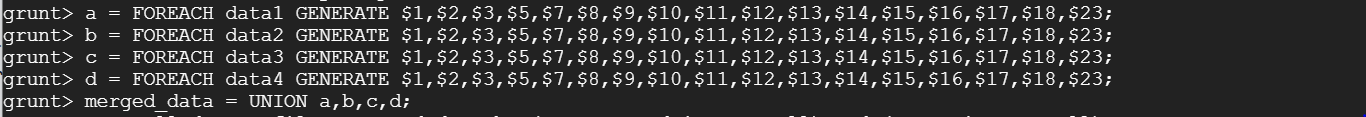


Figure 7: Merging data

1. The data was merged in the variable named merged\_data.

Merged the data using below command:

merged\_data = UNION a,b,c,d;

1. There were few null values present in the data. The columns to be used are OwnerId and Score which contained null values.
2. So, I used FILTER command to remove the rows containing null values.

The command for removing null value is:

non\_null\_data = filter merged\_data by (OwnerUserId is not null) and (Score is not null);

Github link: https://github.com/hareshbhanushali1998/Cloud\_assignment\_1/blob/main/Data\_cleaning\_pig.txt

1. I used below Python command to know the sum of null values in all columns.

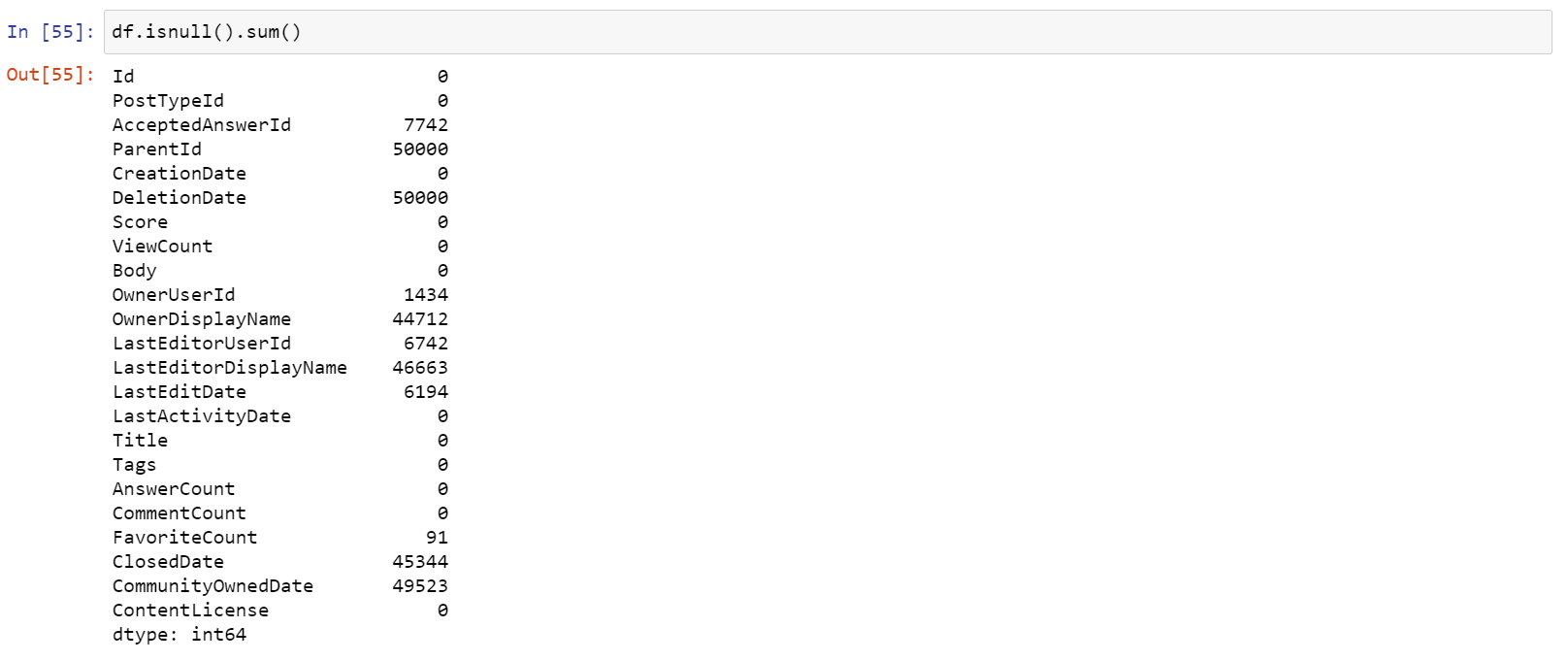


Figure 8: Python command to check null values

1. Now data is ready to be stored in csv file. I used STORE command to store the data in clean\_stack\_data.csv

STORE non\_null\_data INTO 'clean\_stack\_data.csv' USING PigStorage(',');

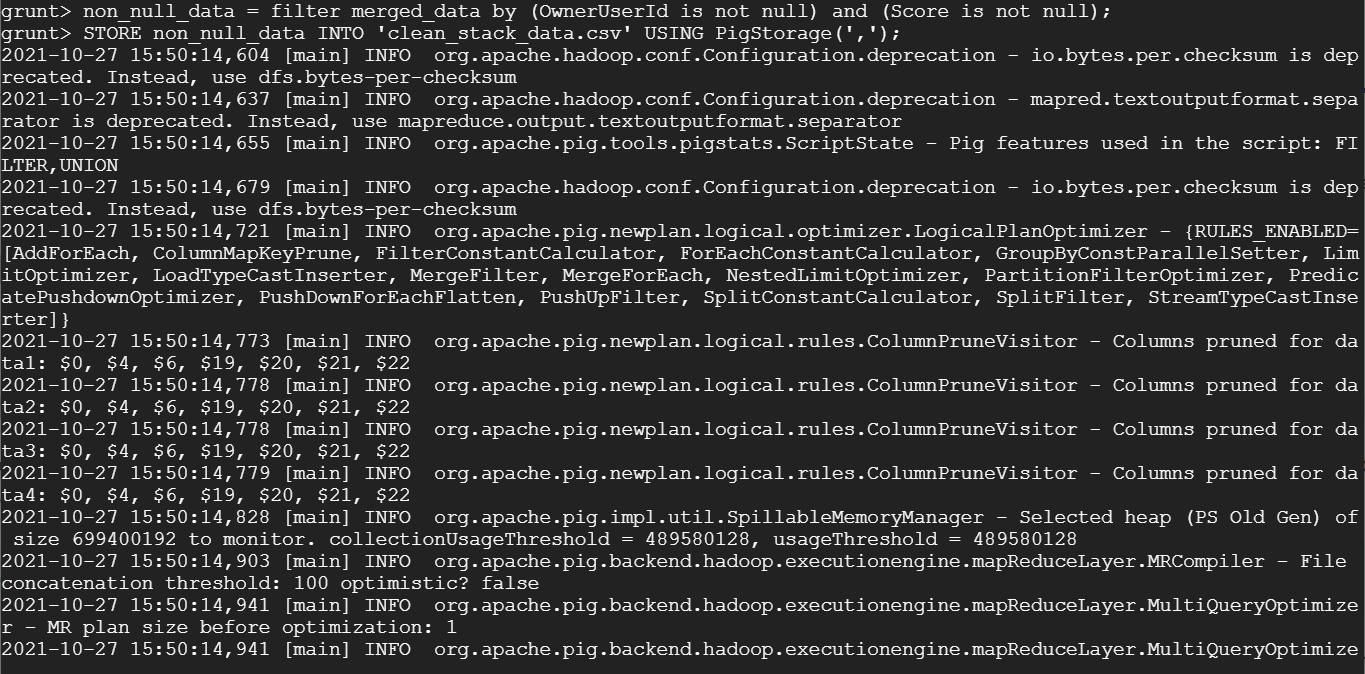


Figure 9: Removing null values and storing data in csv file

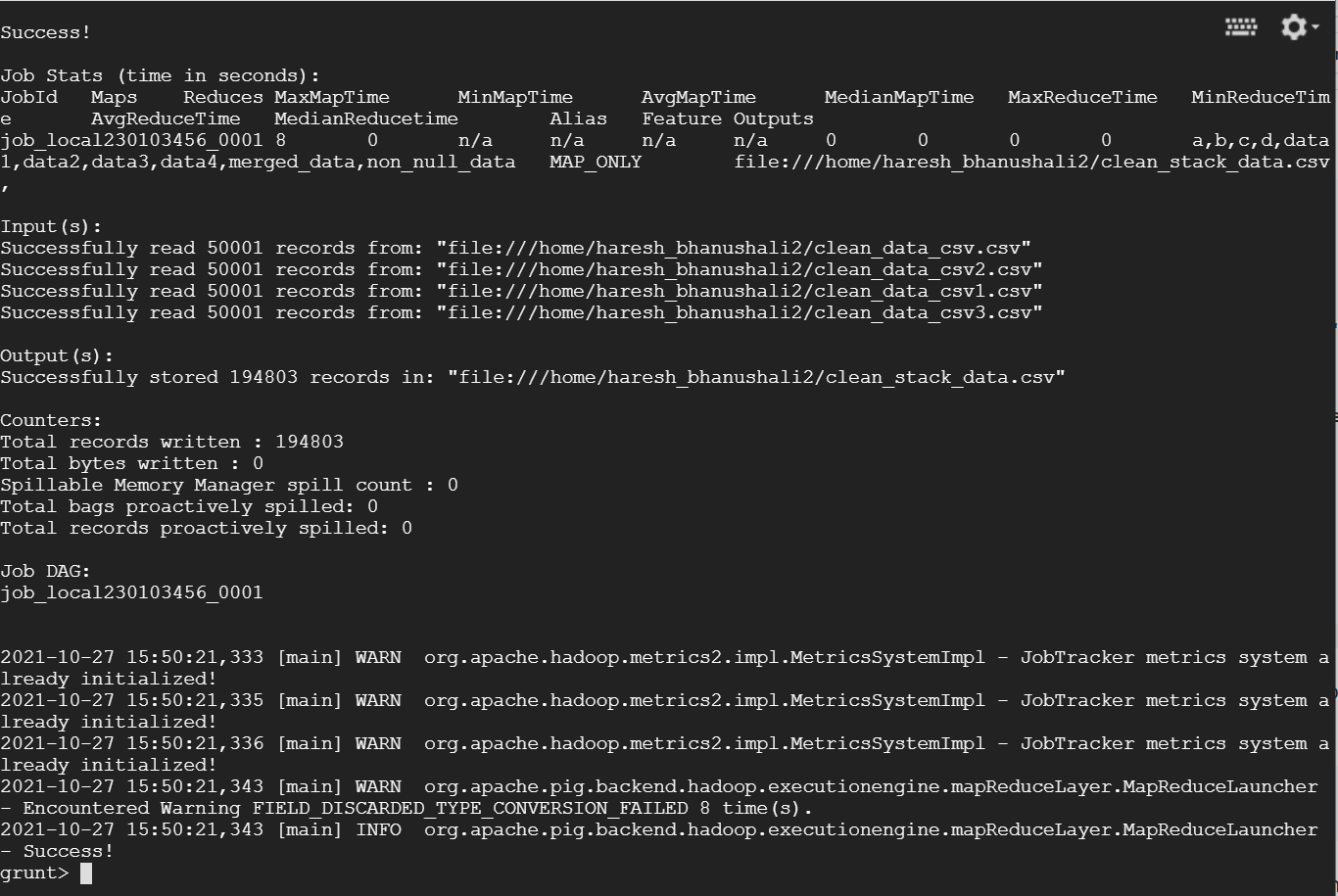


Figure 10: Data stored successfully

1. After storing the data, we need to copy the data in HDFS. The data was stored in local which we will now copy to HDFS.

To copy data from local to HDFS I used below command:

hdfs dfs -copyFromLocal /home/haresh\_bhanushali2/clean\_stack\_data.csv /

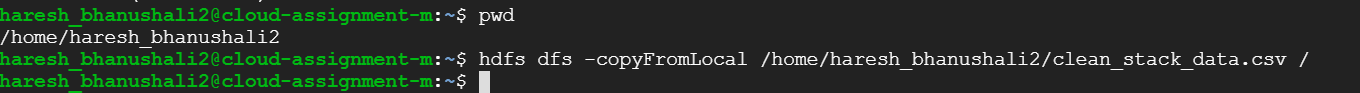
****

Figure 11: Copy from local to HDFS

1. Now after copying data from local to HDFS. I started hive to run my other queries.

I have used Hive to work on my other queries as it uses query language similar to SQL. As we know Hive can work very well for structured data and our data for the case is structured so it can be used efficiently. MapReduce needs some complex codes to be written in order to achieve the results. But working with Hive can ease out the situation of writing complex codes by writing simple HQL queries that are very similar to SQL.

1. Starting the hive, I first created the table named haresh\_stack\_data. I have used the columns that were earlier used in Pig.

GitHub Link: https://github.com/hareshbhanushali1998/Cloud\_assignment\_1/blob/main/Hive\_Query.txt

CREATE TABLE haresh\_stack\_data(Id INT ,PostTypeId INT,AcceptedAnswerId INT,CreationDate STRING,Score INT,ViewCount INT,Body VARCHAR(10000),OwnerUserId INT,OwnerDisplayName STRING,LastEditorUserId INT,LastEditorDisplayName STRING,LastEditDate STRING,LastActivityDate STRING,Title STRING,Tags STRING,AnswerCount INT,ContentLicense STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

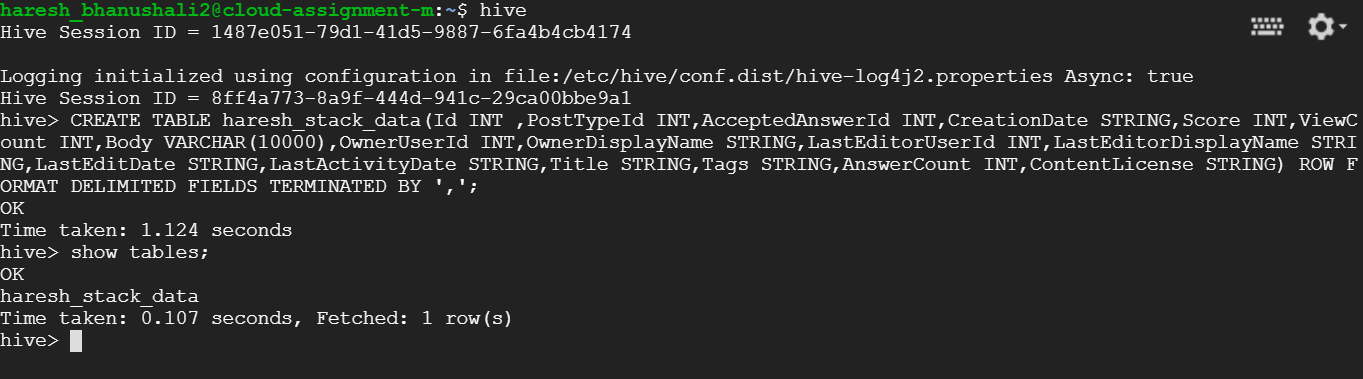


Figure 12: Creating table in Hive

1. After creating the table, we need to load the data from HDFS to the table. To load the data in the table I have used load command. The complete load command is shown below.

load data inpath 'hdfs://cloud-assignment-m/clean\_stack\_data.csv' into table haresh\_stack\_data;

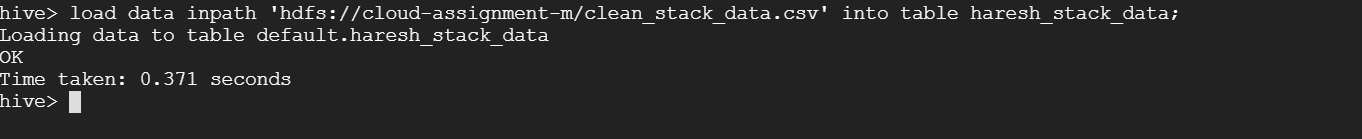


Figure 13: Loading data from HDFS to Hive

Sources:

1. https://www.unix.com/linux/45584-how-remove-only-html-tags-inside-file.html

2. https://unix.stackexchange.com/questions/255380/how-to-remove-symbols-from-a-column-using-awk/255832

3. <https://stackoverflow.com/questions/26411054/filter-out-null-value-via-pig>

4. http://www.hadooplessons.info/2014/12/loading-data-into-hive-table.html

**Task 3 – Query data using Hive**

1. The top 10 posts by score.

In this question we need to find our top 10 posts based on their score.

select id, title, score from haresh\_stack\_data order by score desc limit 10;

So, I selected Id, Title and Score from the table and ordered them in descending order of score to get highest score post at top. I set the limit to 10 to get top 10 posts only.

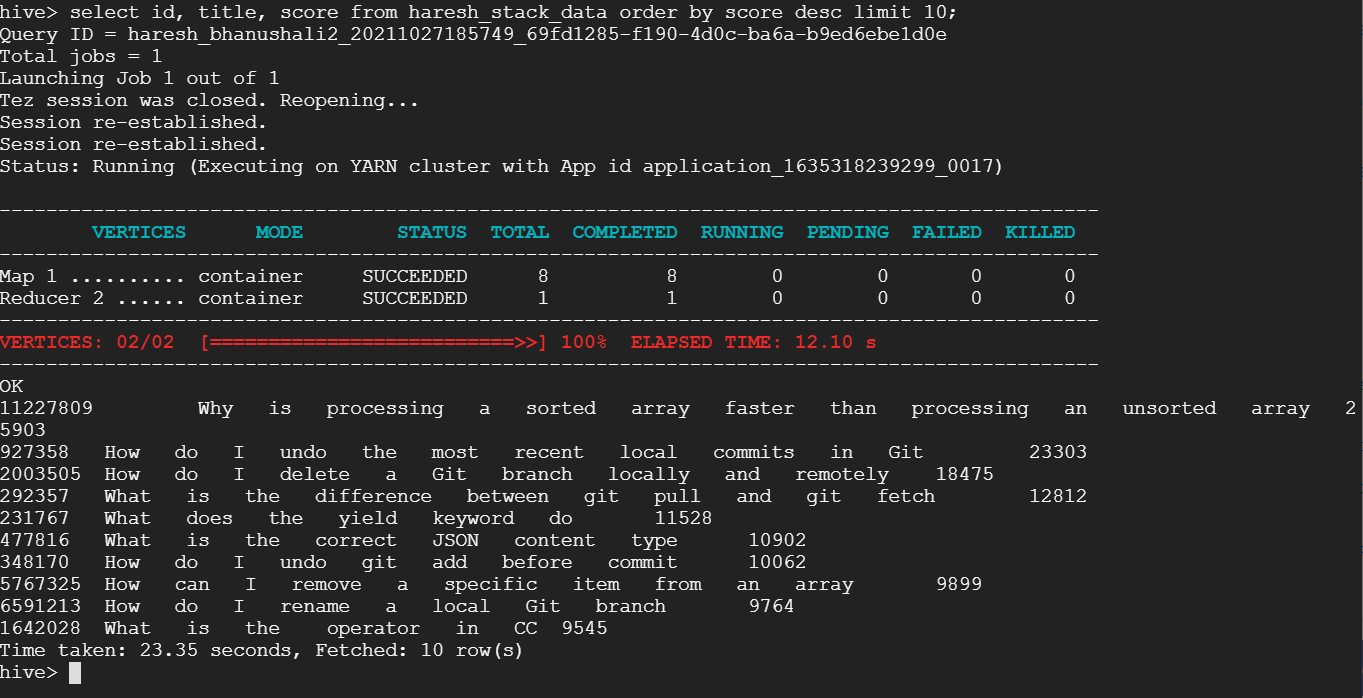


Figure 14: Top 10 posts by score

1. The top 10 users by postscore

Now, here we need to find out top 10 users by post score.

select OwnerUserId, sum(Score) as OverallScore from haresh\_stack\_data group by OwnerUserId order by OverallScore desc limit 10;

For this query I selected OwnerId, calculated sum of score and stored as OverallScore which will be our post score and grouped them by OwnerId. This will give total postscore for that particular OwnerId. Again here I have ordered the query based on OverallScore as we need top users based on post score and limited it to 10 for top 10 users.

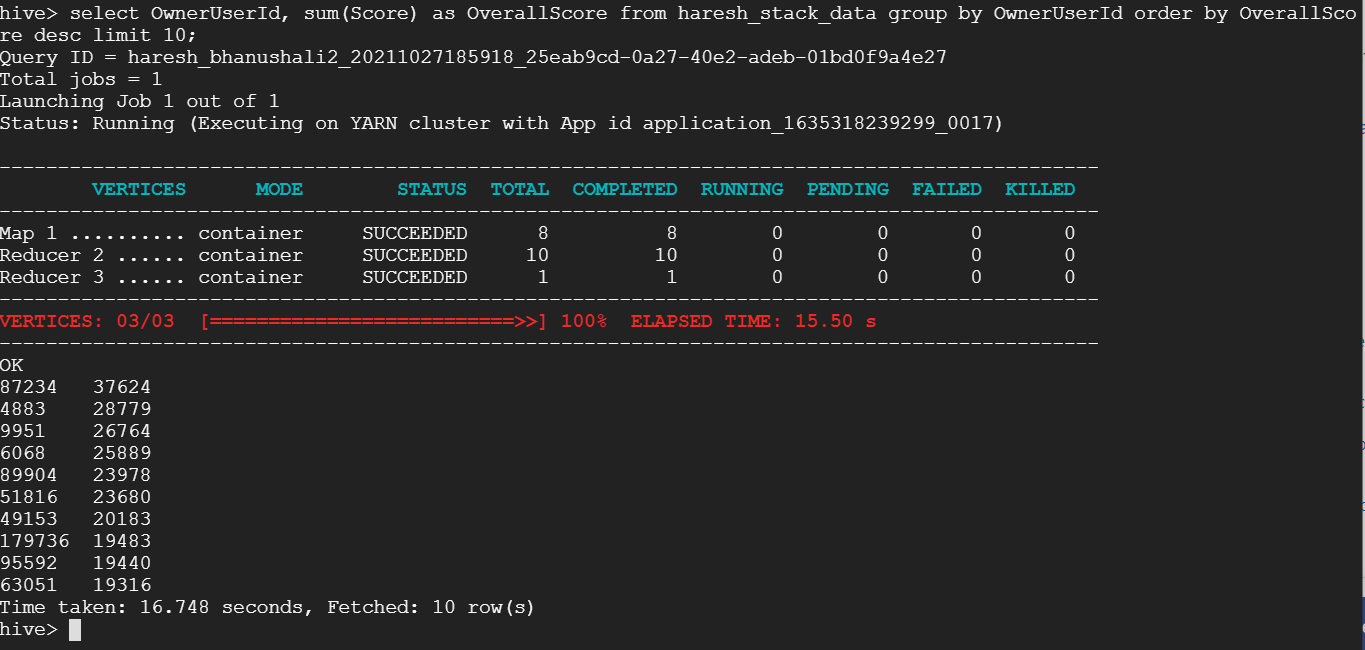


Figure 15: Top 10 users based on postscore

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

Here we need to find out the number of users who have used the word cloud in one of their posts.

select count (distinct OwnerUserId) from haresh\_stack\_data where (lower(body) like '%cloud%' or lower(title) like '%cloud%' or lower(tags) like '%cloud%');

So, for this query I selected distinct OwnerId and had a count as we need to count number of distinct users who have used the word cloud in one of their posts. After selection I set the where condition that Body or Title or Tags wherever user has used word cloud count them. But here I have mentioned lower before the column names as I want to get users who have used cloud in lower case.

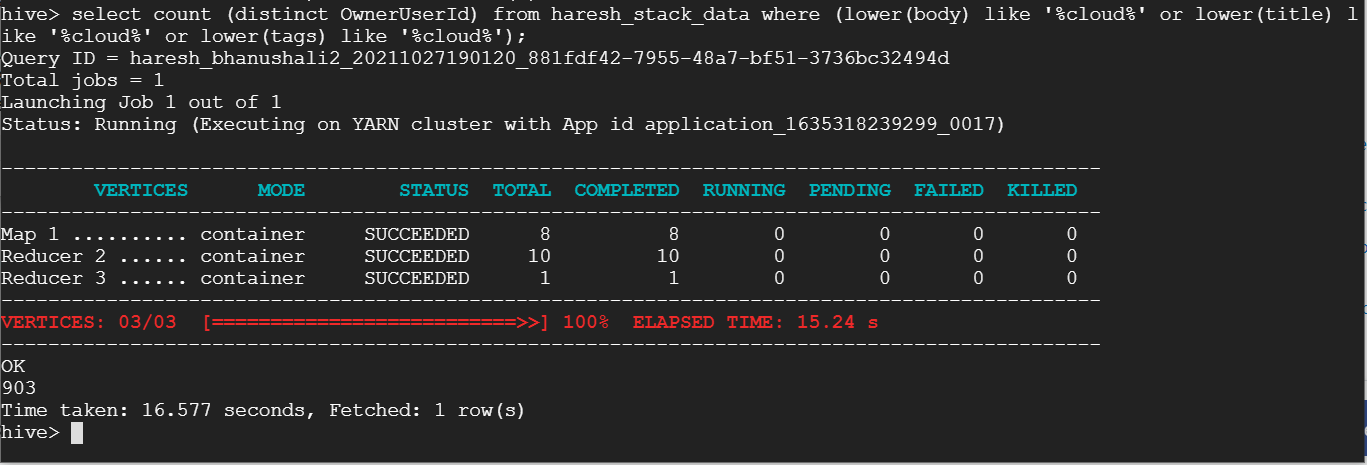


Figure 16: Number of distinct users who used word cloud in one of their posts

select count (distinct OwnerUserId) from haresh\_stack\_data where (body like '%cloud%' or tag like '%cloud%' or title like '%cloud%');

I also tried the same query removing the lower function and got following result.

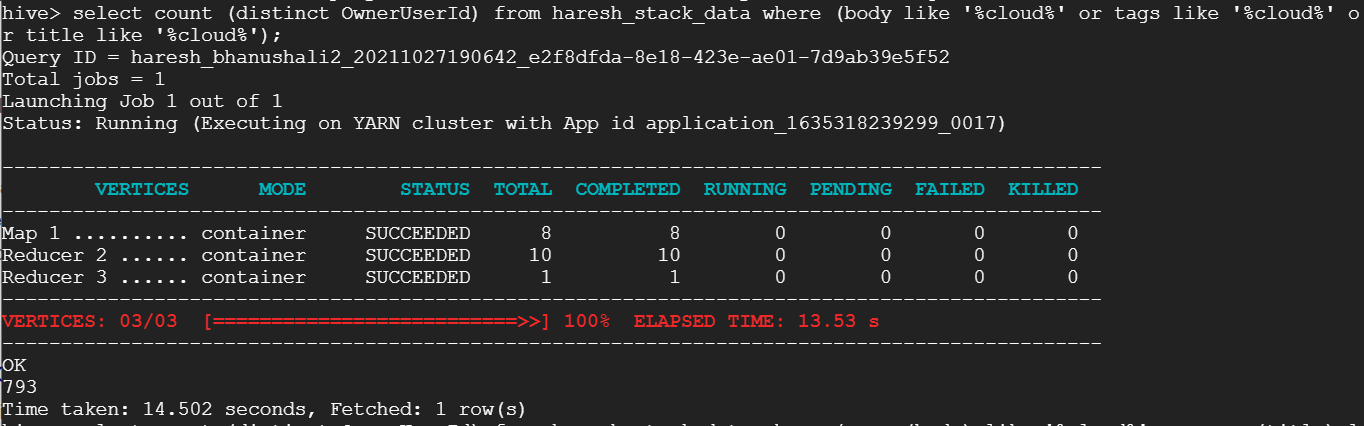


Figure 17: Without lower function

Sources:

1. <https://stackoverflow.com/questions/31450622/search-for-a-particular-text-in-a-string-hive/31455066>

**Task 4 – Using Hive to calculate the per-user TF- IDF of the top 10 terms for each of the top 10 users**

To calculate the per user TF-IDF of 10 terms of each of the top users I have downloaded some dependencies to use some functions in Hive. I am not thorough with the implementation of TF-IDF but studied and tried to implement using following sources.

I executed following commands in order to achieve the results.

1. create temporary macro max2(x INT, y INT) if(x>y,x,y);
2. create temporary macro tfidf(tf FLOAT, df\_t INT, n\_docs INT) tf \* (log(10, CAST(n\_docs as FLOAT)/max2(1,df\_t)) + 1.0);
3. create table highest\_users as select Owneruserid, sum(Score) as OverallScore from haresh\_stack\_data group by OwnerUserId order by OverallScore desc limit 10;

Creating table highest\_users based on sum of score storing as OverallScore and ordering it in descending order of OverallScore and limiting it to 10 for getting top 10 users. (Task 3 Query 2)

1. create table new\_users as select d.OwnerUserID,title from haresh\_stack\_data d join highest\_users t on d.OwnerUserID = t.OwnerUserID;
2. create or replace view view\_user as select OwnerUserId, eachword from new\_users LATERAL VIEW explode(tokenize(Title, True)) t as eachword where not is\_stopword(eachword);

Tokenizing each word so that it can be given a count each time the word is encountered. Also, where condition is eachword is not a stopword that is a, the, in, etc which needs not to be counted.

1. create or replace view user\_tf as select OwnerUserid, eachword, freq from (select OwnerUserId, tf(eachword) as word2freq from view\_user group by ownerUserId) t LATERAL VIEW explode(word2freq) t2 as eachword, freq;

Calculating the frequency of word.

1. create or replace view term\_freq as select \* from (select ownerUserId, eachword, freq, rank() over (partition by OwnerUserId order by freq desc) as rank from user\_tf) t where rank < 10;
2. select Owneruserid, eachword, freq from term\_freq;

Selecting OwnerUserId , each word of that user and its frequency.

Sources:

1. <https://stackoverflow.com/questions/36250868/tokenize-function-in-hive>
2. <https://hivemall.incubator.apache.org/userguide/ft_engineering/tfidf.html>
3. <https://github-wiki-see.page/m/daijyc/hivemall/wiki/TFIDF-calculation>

