***Assignment1***

***DSCI 5350***

***Lakshmi Sunanda Nagareddygari***

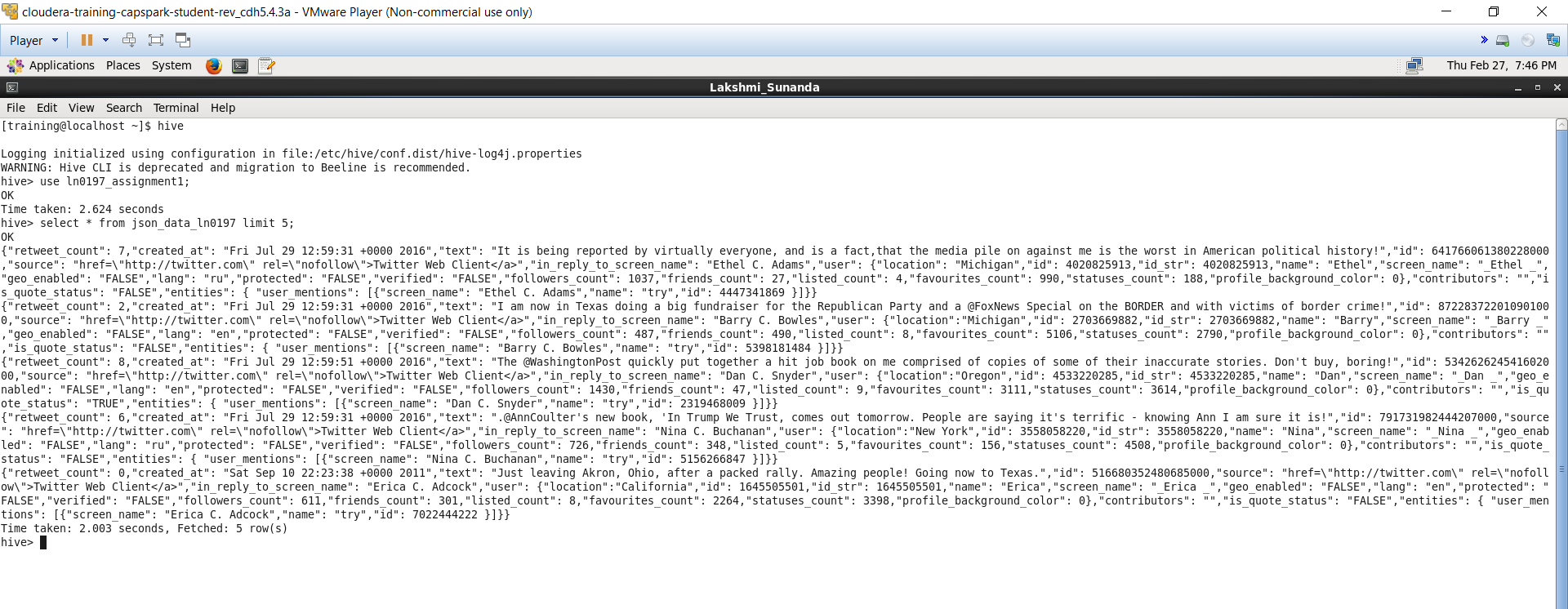
1. **Correctly process and store the files in Hive. All tables created for the solution must have your student\_id as a prefix to table name. For example, if I were to store the dictionary table, I would name it dictionary\_ks0776**

**SOLUTION:**

create table json\_data\_ln0197 (dump string);

load data local inpath '/home/training/Desktop/Assignment1/Twitter.json' into table json\_data\_ln0197;

Below screenshot depicts the twitter data loaded to the table “json\_data\_ln0197”.



create table json\_project\_ln0197(

retweet\_count int,

create\_time\_stamp string,

tweets string,

id bigint,

source string,

location string,

user\_id bigint,

name string,

screen\_name string,

geo\_enabled string,

lang string,

protected string,

verified string,

followers\_count bigint,

friends\_count int,

listed\_count int,

favourites\_count int,

statuses\_count int,

profile\_background\_color int,

contributors string,

is\_quote\_status string,

screen\_name\_re string,

name\_re string,

id\_re string);

insert into table json\_project\_ln0197 select

get\_json\_object(dump,'$.retweet\_count') as retweet\_count,

get\_json\_object(dump,'$.created\_at') as created\_time\_stamp,

get\_json\_object(dump,'$.text') as tweets,

get\_json\_object(dump,'$.id') as id,

get\_json\_object(dump,'$.source') as source,

get\_json\_object(dump,'$.user.location') as location,

get\_json\_object(dump,'$.user.id') as user\_id,

get\_json\_object(dump,'$.user.name') as name,

get\_json\_object(dump,'$.user.screen\_name') as screen\_name,

get\_json\_object(dump,'$.user.geo\_enabled') as geo\_enabled,

get\_json\_object(dump,'$.user.lang') as lang,

get\_json\_object(dump,'$.user.protected') as protected,

get\_json\_object(dump,'$.user.verified') as verified,

get\_json\_object(dump,'$.user.followers\_count') as followers\_count,

get\_json\_object(dump,'$.user.friends\_count') as friends\_count,

get\_json\_object(dump,'$.user.listed\_count') as listed\_count,

get\_json\_object(dump,'$.user.favourites\_count') as favourites\_count,

get\_json\_object(dump,'$.user.statuses\_count') as statuses\_count,

get\_json\_object(dump,'$.user.profile\_background\_color') as profile\_background\_color,

get\_json\_object(dump,'$.contributors') as contributors,

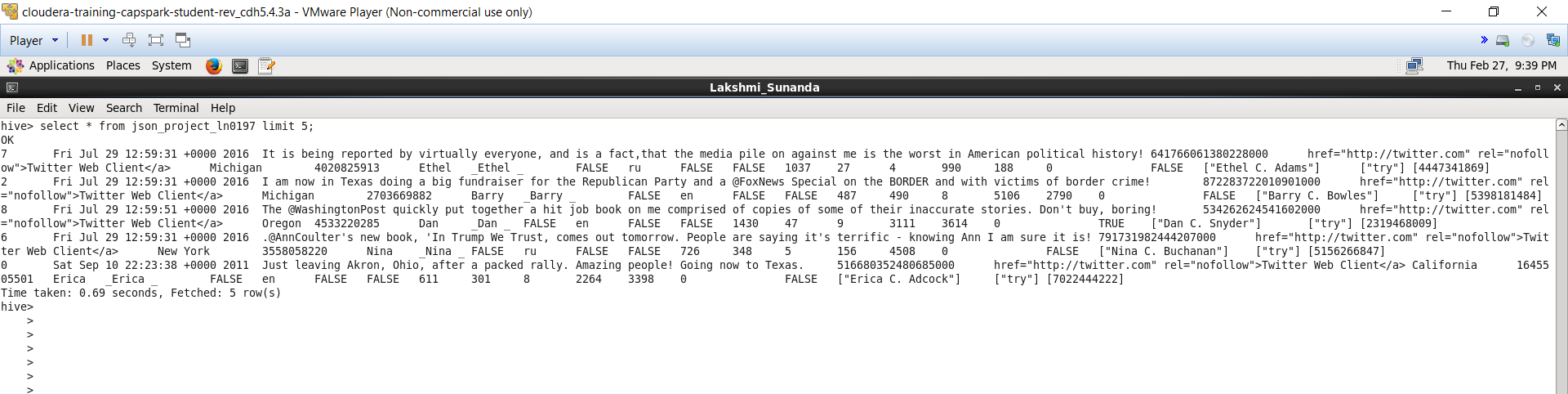
get\_json\_object(dump,'$.is\_quote\_status') as is\_quote\_status,

get\_json\_object(dump,'$.entities.user\_mentions.screen\_name') as screen\_name\_re,

get\_json\_object(dump,'$.entities.user\_mentions.name') as name\_re,

get\_json\_object(dump,'$.entities.user\_mentions.id') as en\_id from json\_data\_ln0197;

Below Screenshot depicts data load to the table I have created with proper columns and data types respectively.



create table hash\_dictionary\_ln0197(word string,rating int)

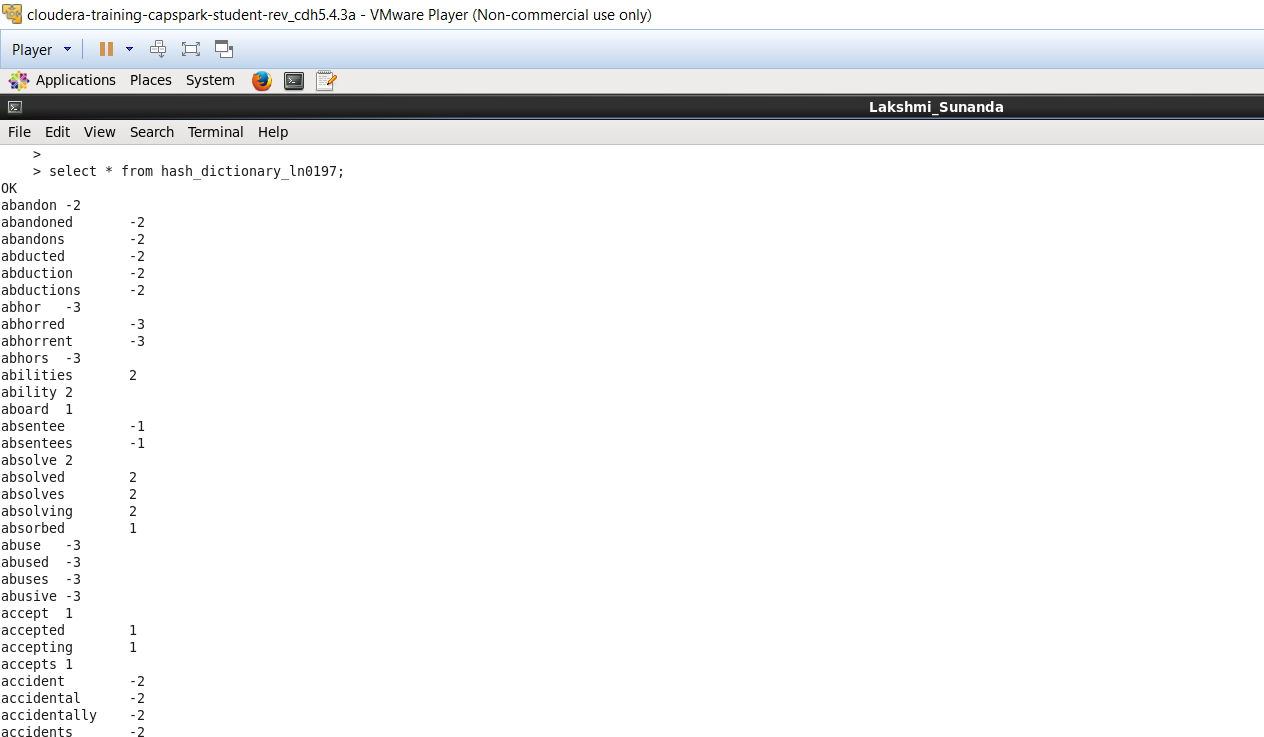
row format delimited

fields terminated by ','

lines terminated by '\n';

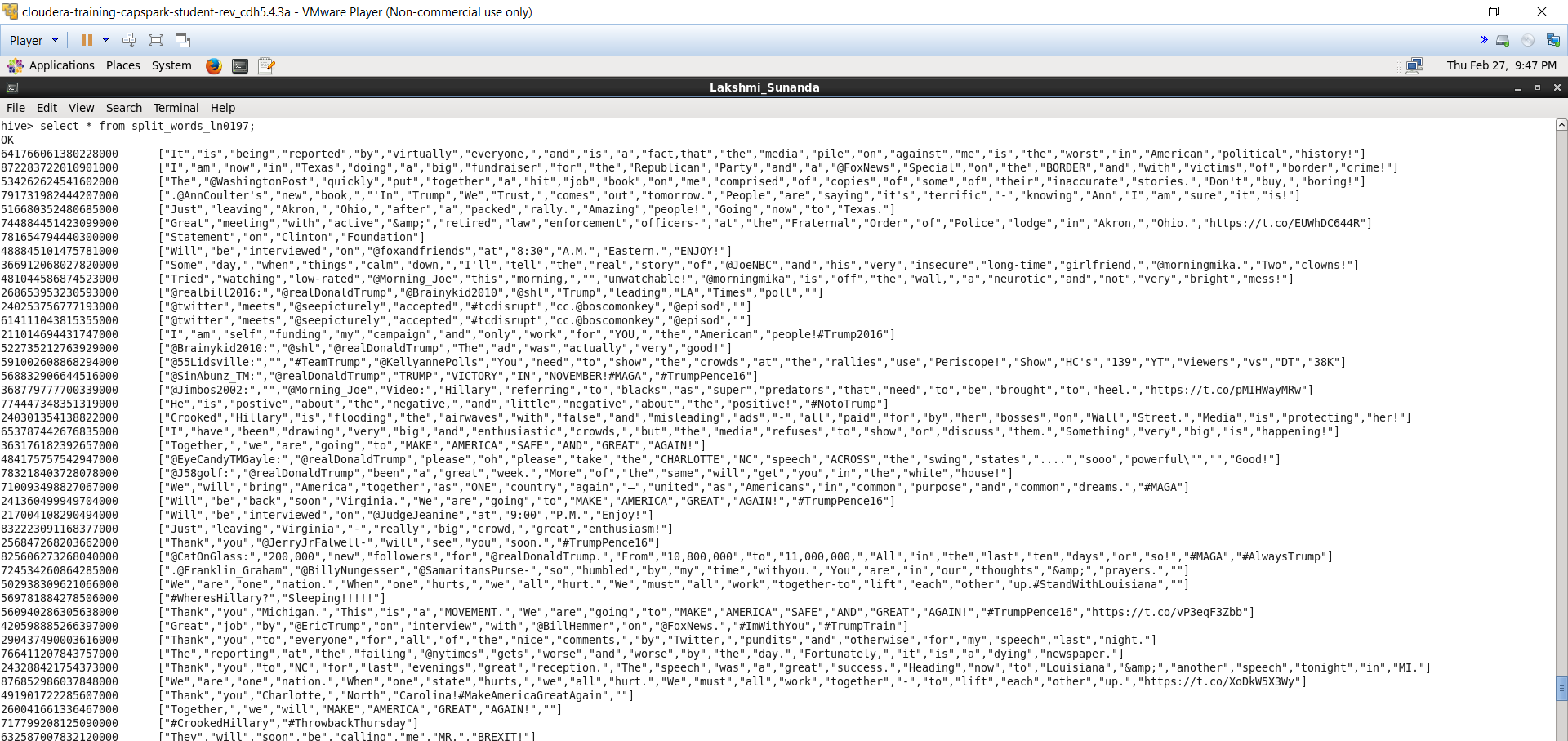
load data local inpath 'home/training/Desktop/Dictionary.txt' into table hash\_dictionary\_ln0197;

Below screenshot depicts all the dictionary data loaded to the table “hash\_dictionary\_ln0197”

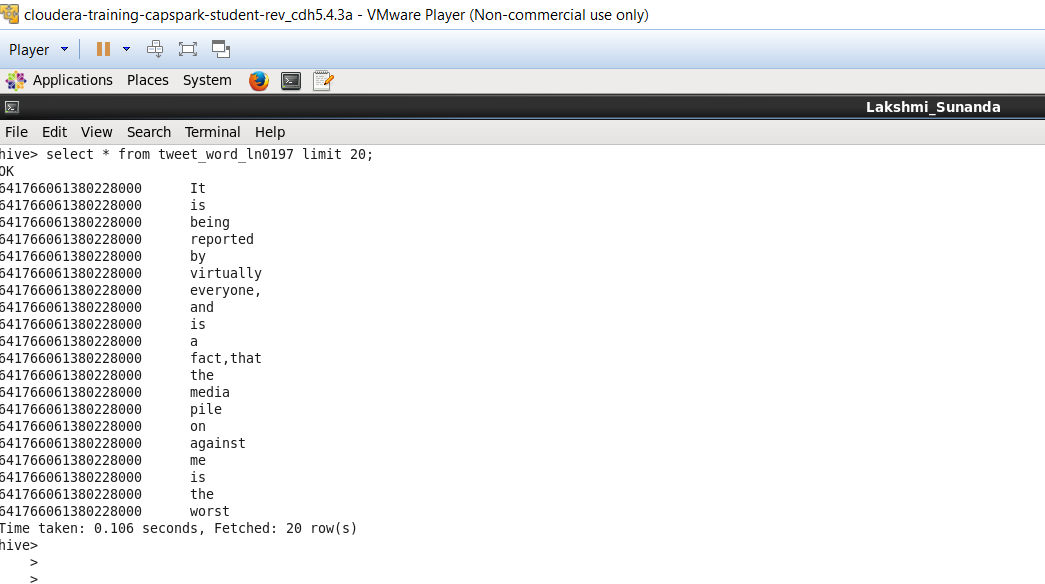


create table split\_words\_ln0197 as select id as id,split(tweets,' ') as words from json\_project\_ln0197;

Below screenshot depicts with the data of the tweet words which are split.



create table tweet\_word\_ln0197 as select id as id,word from split\_words\_ln0197\_ln0197 LATERAL VIEW explode(words) w as word;



**2. Using the files provided, answer the following:**

**a. What were the hashtags used in the file, and how many times each hashtag was used? (10 points)**

SOLUTION:

create table hash\_extract\_ln0197 as select id,regexp\_extract(word,'^#.+',0) as hash from tweet\_word\_ln0197 where word=regexp\_extract(word,'^#.+',0) and word<>'';

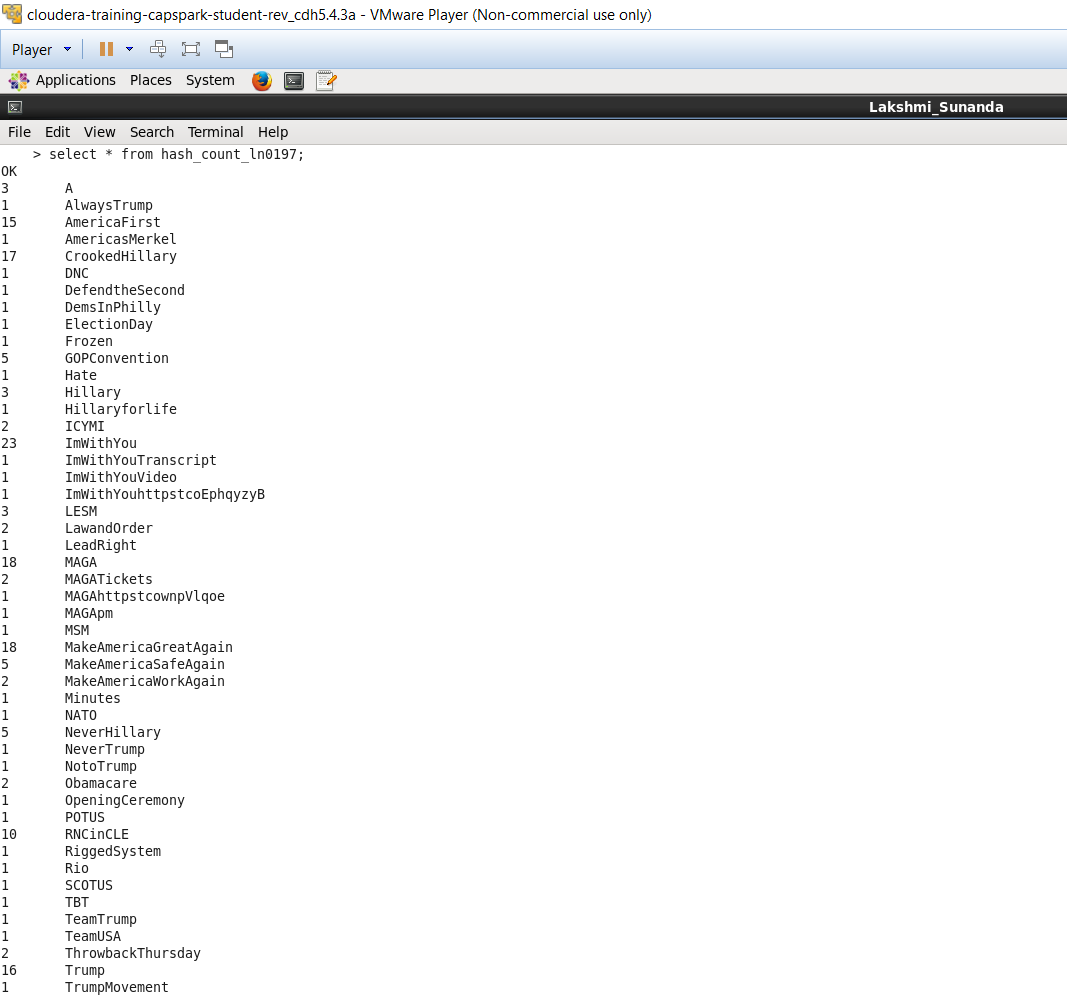
table hash\_cleaning1\_ln0197 as select id as id,split(hash,'#') as hash\_clean from hash\_extract\_ln0197;

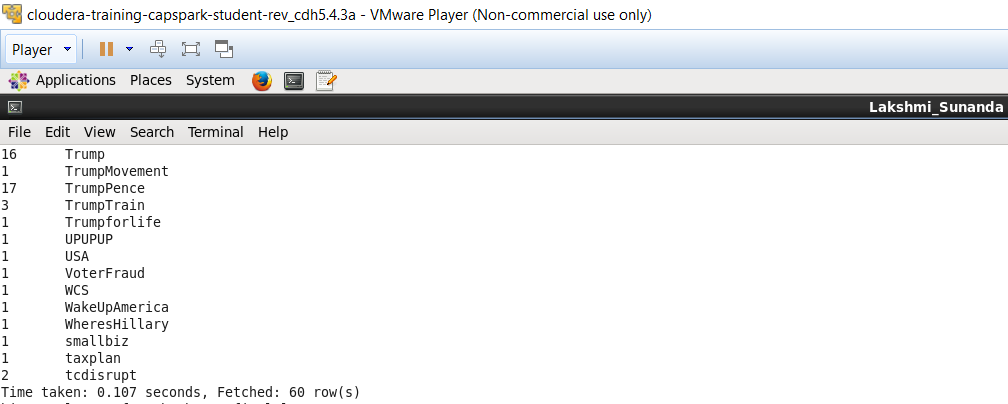
create table hash\_cleaning2\_ln0197 as select id ,hash from hash\_cleaning1\_ln0197 LATERAL VIEW explode(hash\_clean) h as hash where hash<>'';

create table hash\_cleaning3\_ln0197 as select id ,regexp\_replace(hash,'[,:\\!-:."?]|https//t.co/wnpVlq6oe4|https//t.co/Eph6qy7zyB','') as final\_clean from hash\_cleaning2\_ln0197 where id<>106841024367780000;

create table hash\_count\_ln0197 as select count(final\_clean) as hash\_count, final\_clean from hash\_cleaning3\_ln0197 group by final\_clean;

Below screenshot depicts with the data of the word and the count the word has been used to tweet.





**b. Identify the most trending hashtag by the day. How many times the most trending hashtag was tweeted? (10 points) [Note: day should be in the format ‘yyyy-mm-dd’]**

**SOLUTION:**

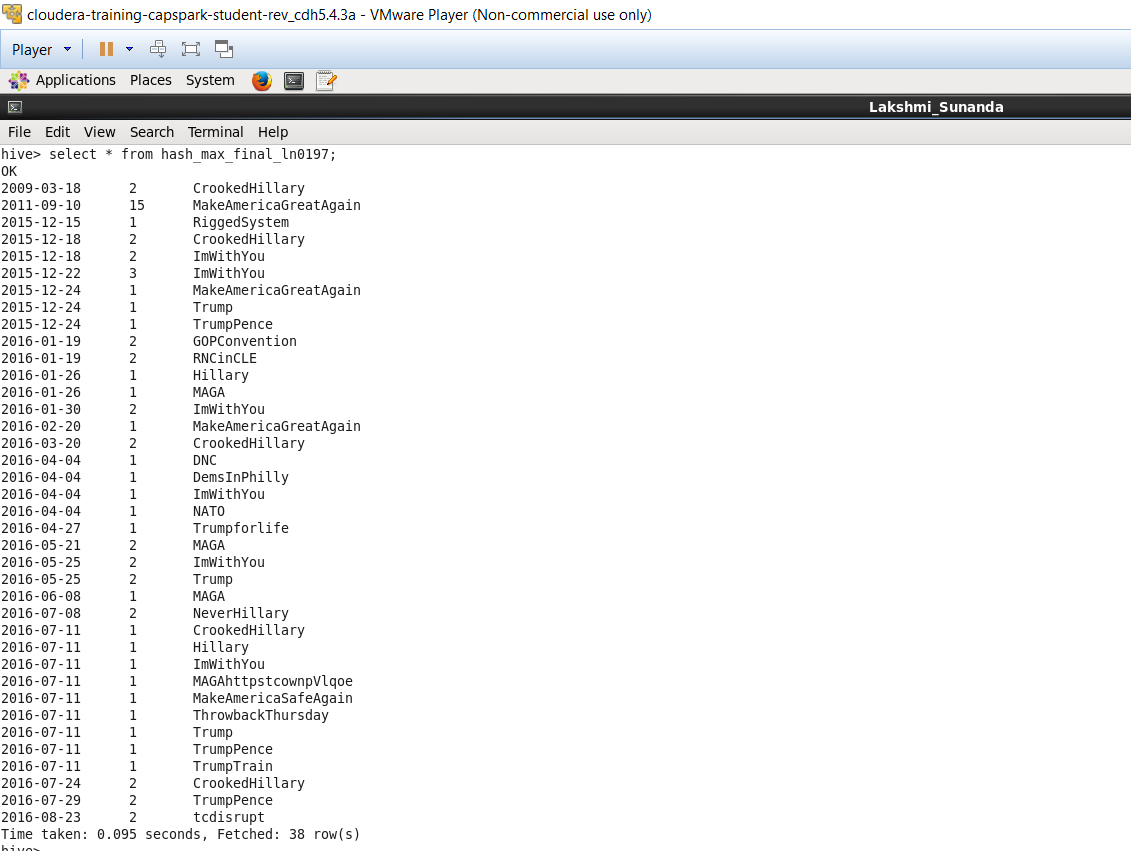
create table hash2\_ln0197 as select hc.id,hc.final\_clean,from\_unixtime(unix\_timestamp(regexp\_replace(js.create\_time\_stamp,'^\\w{3}\\s|\\s\\d+:\\d+:\\d+\\s|\\+\\d{4}|\\s',''),'MMMddyyyy'),'yyyy-MM-dd') as date from hash\_cleaning3\_ln0197 hc inner join json\_project\_ln0197 js on hc.id=js.id;

create table count1\_ln0197 as select date,final\_clean,count(final\_clean) as count from hash2\_ln0197 group by date,final\_clean;

create table max\_count\_ln0197 as select date,max(count) as max\_count from count1\_ln0197 group by date;

create table hash\_max\_final\_ln0197 as select hm.date,hm.max\_count,hc.final\_clean from max\_count\_ln0197 hm left outer join count1\_ln0197 hc on hm.date=hc.date where hm.max\_count=hc.count;

Below screenshot depicts with the data of the count the most trending hashtag was tweeted.



**c. Determine the score for each tweet that was posted? Identify whether the tweet had a positive or negative sentiment? Use the dictionary.txt file for determining the score. Note: Include the date (’yyyy-mm-dd’), tweet\_id, user\_name, and the score in the resulting query.**

**SOLUTION:**

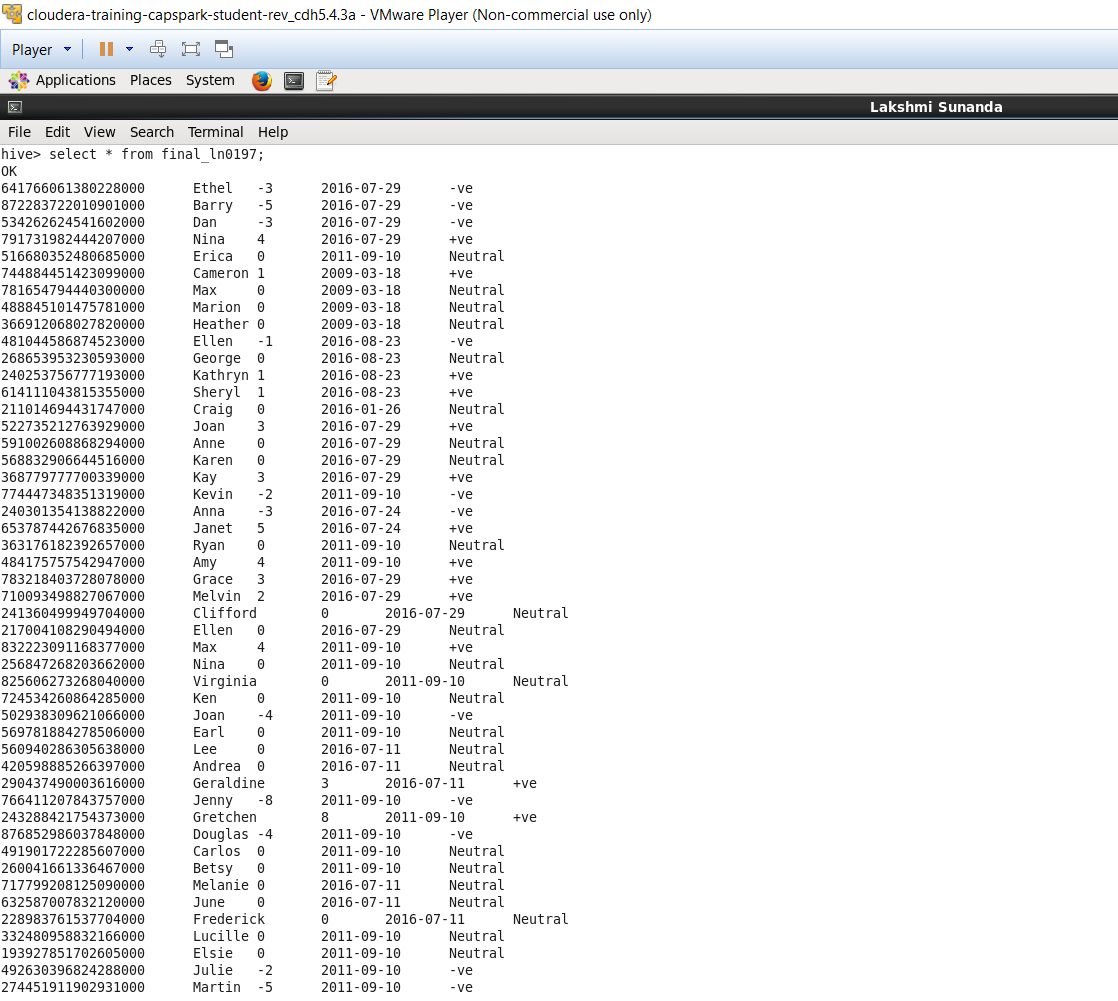
create table att1\_ln0197 as select tw.id,from\_unixtime(unix\_timestamp(regexp\_replace(js.create\_time\_stamp,'^\\w{3}\\s|\\s\\d+:\\d+:\\d+\\s|\\+\\d{4}|\\s',''),'MMMddyyyy'),'yyyy-MM-dd') as date,regexp\_replace(tw.word,'[@#.,\\!-?:"]','') as word from tweet\_word\_ln0197 tw inner join json\_project\_ln0197 js on tw.id=js.id;

create table att11\_ln0197 as select at.id,at.date,at.word,COALESCE(hd.rating, 0 ) as rating from att1\_ln0197 at left outer join hash\_dictionary\_ln0197 hd on at.word=hd.word;

create table att3\_ln0197 as select id,sum(rating) as final\_rating from att11\_ln0197 group by id;

create table final\_ln0197 as select a.id,js.user\_id,a.final\_rating,from\_unixtime(unix\_timestamp(regexp\_replace(js.create\_time\_stamp,'^\\w{3}\\s|\\s\\d+:\\d+:\\d+\\s|\\+\\d{4}|\\s',''),'MMMddyyyy'),'yyyy-MM-dd') as create\_time\_stamp, regexp\_replace(regexp\_replace(regexp\_replace(final\_rating,'^0','neutral') ,'^-\\d','negative'),'^\\d','positive') as sentiment from att3\_ln0197 a inner join json\_project\_ln0197 js on a.id=js.id;

Below screenshot depicts the data whether the tweet had a positive or negative sentiment.



1. **Propose a better solution for the sentiment analysis as compared to 1(c). Cite the source. (5 points) Note: You just need to provide the solution, you are not required to solve the problem using the solution.**

**SOLUTION:**

We can solve sentimental analysis using two approaches Machine Learning and Dictionary Approach. We used Dictionary Approach.

The best ways are through Natural Processing Algorithms. Natural language processing also known as text analytics or data mining, uses software to analyze words and extract their actual meanings. It takes emotion and context into consideration and adheres to a set of rules developed as a result of identified patterns and themes in language.

This analysis can also be possible by implementing Machine Learning Methodologies so as it can learn itself on premise of what words were utilized, how often they were utilized and attempts to comprehend the situation in which they were utilized.

Below is the link which tells a way to use Natural Processing Algorithms for Sentiment Analysis.

<https://casmodeling.springeropen.com/articles/10.1186/s40294-016-0016-9>