**BUAN 6346 Bigdata Analytics**

**Assignment 2**

**Submitted by**

**Geethika Veluri**

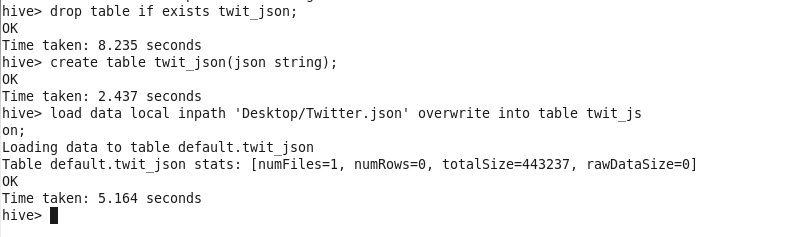
**Q1. a.What are the hashtags used and how many times each are used?**

**Step 1: Importing into Hive from Desktop**

***drop table if exists twit\_json;***

***create table twit\_json(json string);***

***load data local inpath 'Desktop/Twitter.json' overwrite into table twit\_json;***



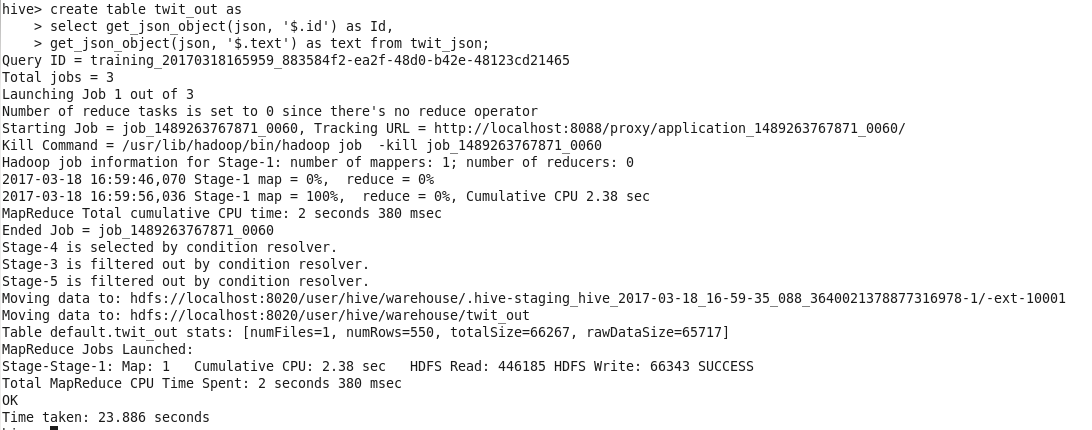
**Step 2: Reading the Json data into columns**

***drop table if exists twit\_out;***

***create table twit\_out as***

***select get\_json\_object(json, '$.id') as Id,***

***get\_json\_object(json, '$.text') as text from twit\_json;***



**Step 3: Filter out the hashtags and count then number**

***drop table if exists hashtag\_count;***

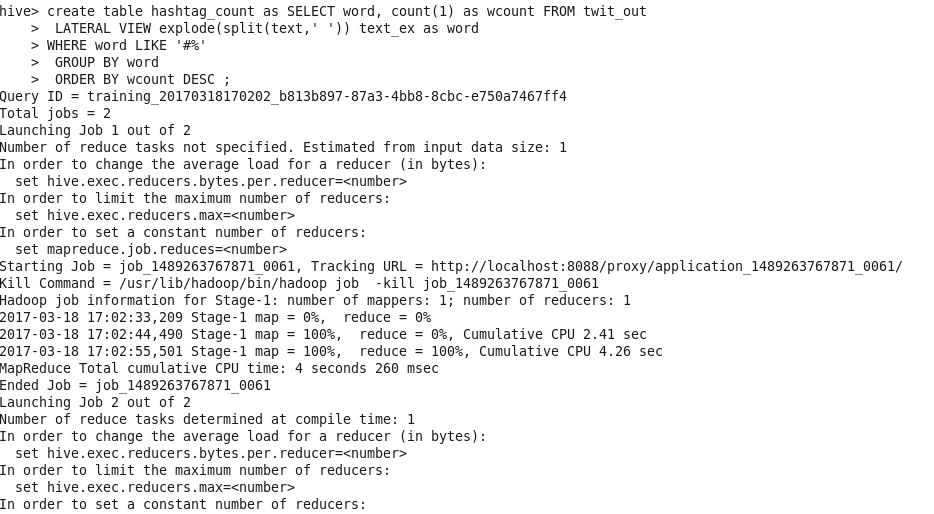
***create table hashtag\_count as SELECT word, count(1) as wcount FROM twit\_out***

***LATERAL VIEW explode(split(text,' ')) text\_ex as word***

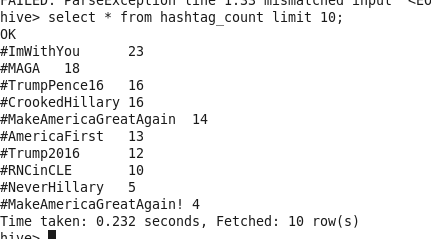
***WHERE word LIKE '#%'***

***GROUP BY word***

***ORDER BY wcount DESC ;***



**Output: This output shows Top 10 Hashtags out of total 70 Hashtags**



**Q1. b. What is the most trending hashtag in a day and how many times is it tweeted? [Note: day should be in the format ‘yyyy-mm-dd’]**

**Step1: Reading JSON data into columns**

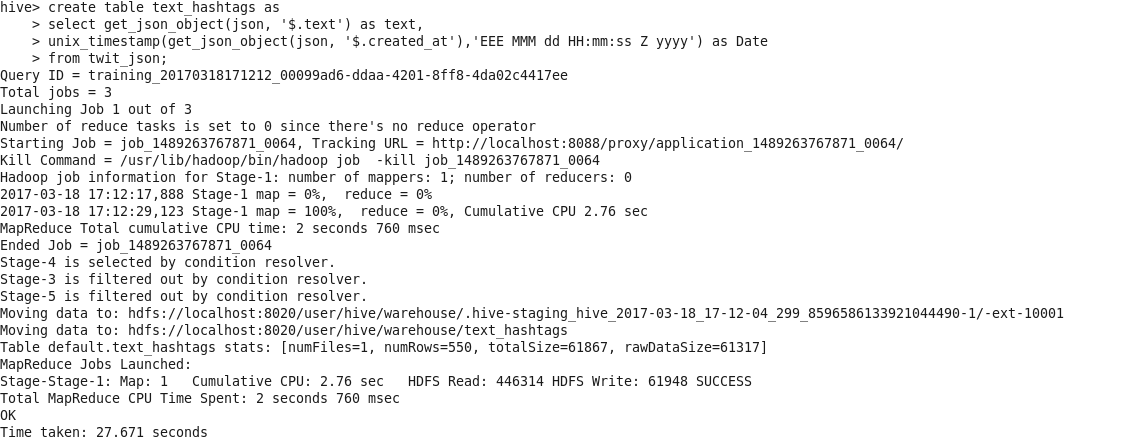
***drop table if exists text\_hashtags;***

***create table text\_hashtags as***

***select get\_json\_object(json, '$.text') as text,***

***unix\_timestamp(get\_json\_object(json, '$.created\_at'),'EEE MMM dd HH:mm:ss Z yyyy') as Date***

***from twit\_json;***



**Step 2: Converting date format and extracting the words from text**

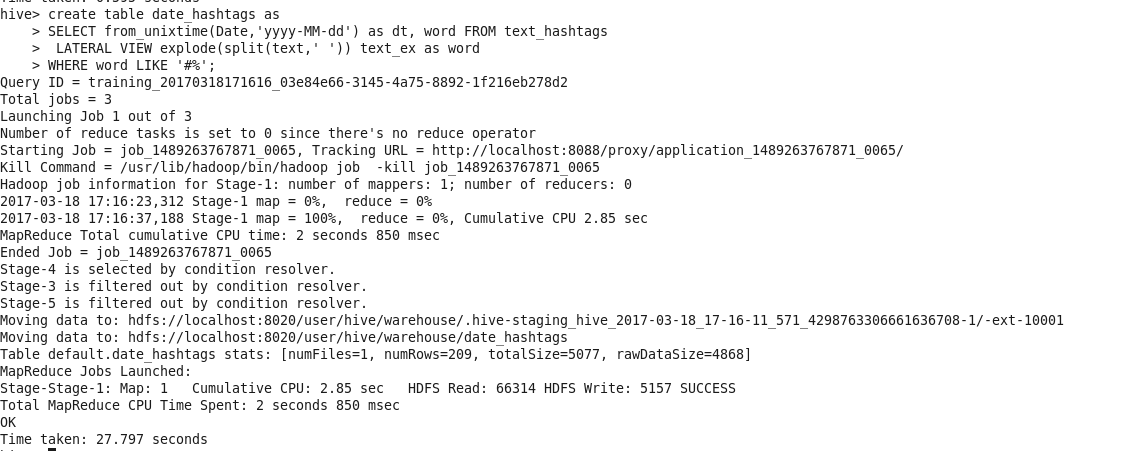
***drop table if exists date\_hashtags;***

***create table date\_hashtags as***

***SELECT from\_unixtime(Date,'yyyy-MM-dd') as dt, word FROM text\_hashtags***

***LATERAL VIEW explode(split(text,' ')) text\_ex as word***

***WHERE word LIKE '#%';***



**Step 3: Counting the number of Hashtags each day**

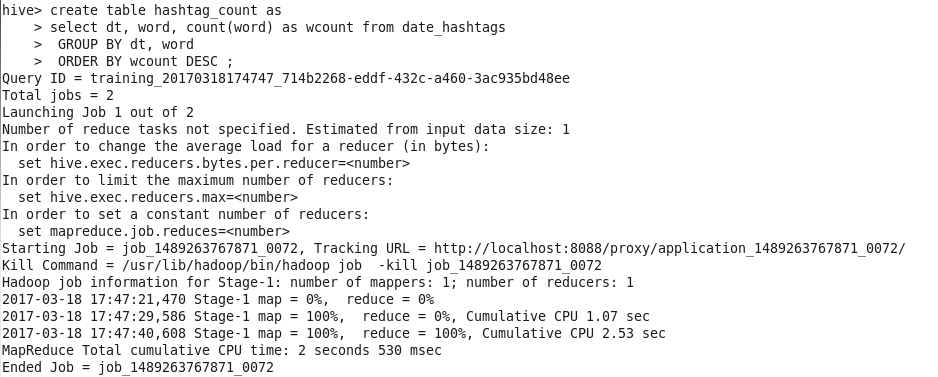
***drop table if exists hashtag\_count;***

***create table hashtag\_count as***

***select dt, word, count(word) as wcount from date\_hashtags***

***GROUP BY dt, word***

***ORDER BY wcount DESC ;***



**Step 4: Extracting the Hashtag used the most in a day**

***drop table if exists highest\_hashtag\_count;***

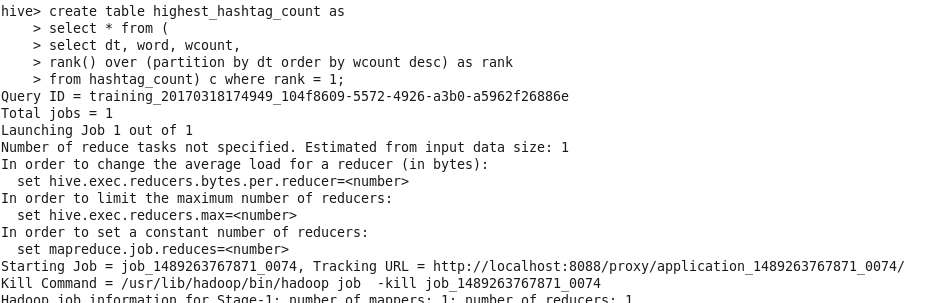
***create table highest\_hashtag\_count as***

***select \* from (***

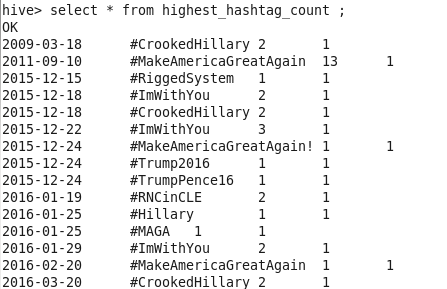
***select dt, word, wcount,***

***rank() over (partition by dt order by wcount desc) as rank***

***from hashtag\_count) c where rank = 1;***



**Output: 15 rows are shown in screenshot below out of 37 rows. The third column represents the frequency and the fourth column represents rank.**



**Q1. c. Which state users are most active and how many tweets are posted by State?**

**Step 1: Reading the JSON data into columns**

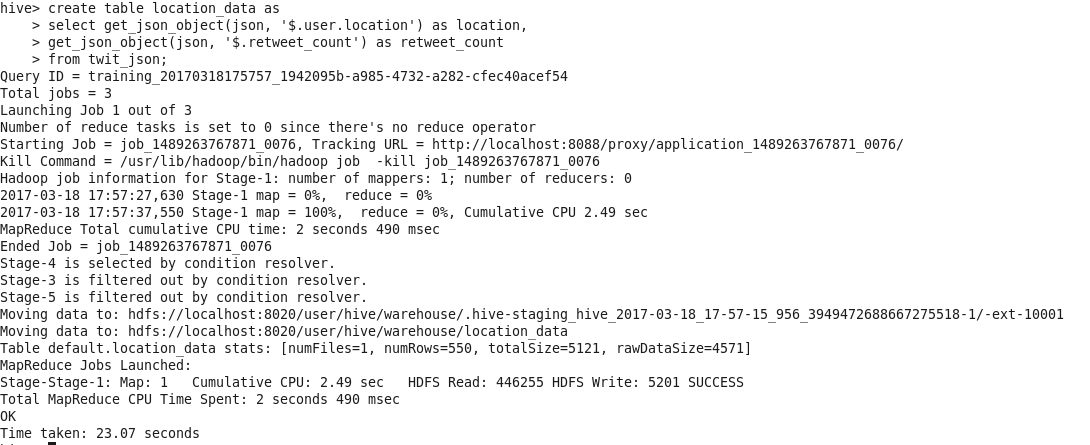
***drop table if exists location\_data;***

***create table location\_data as***

***select get\_json\_object(json, '$.user.location') as location,***

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

***from twit\_json;***



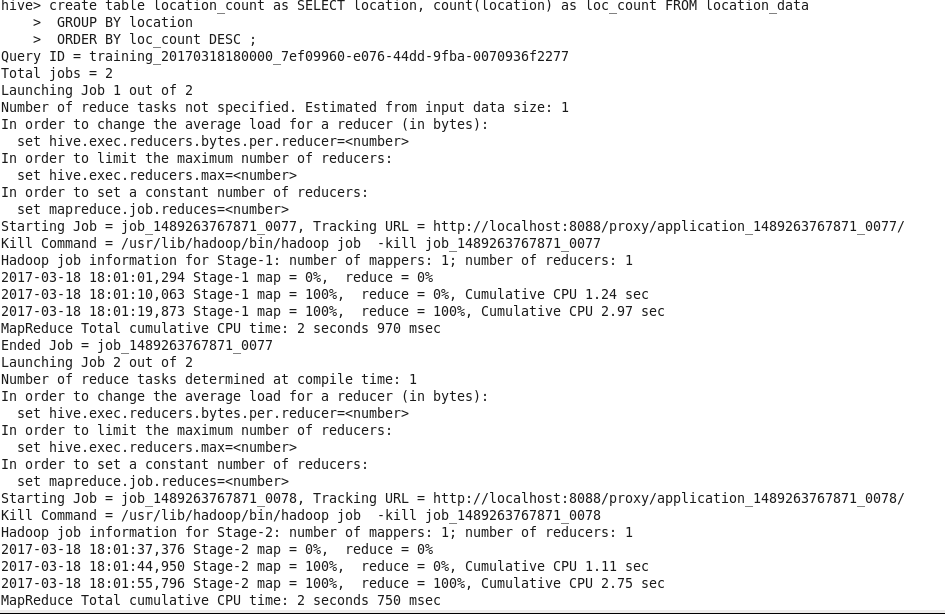
**Step 2: Counting the number of users by location**

***drop table if exists location\_count;***

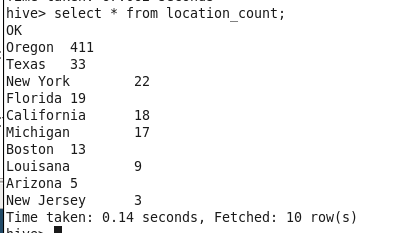
***create table location\_count as SELECT location, count(location) as loc\_count FROM location\_data***

***GROUP BY location***

***ORDER BY loc\_count DESC ;***



**Output: California has the most active users**



**Q1. d. Based on the user’s followers count, who are the top ten users who have tweeted?**

**Step 1: Reading the JSON data into columns**

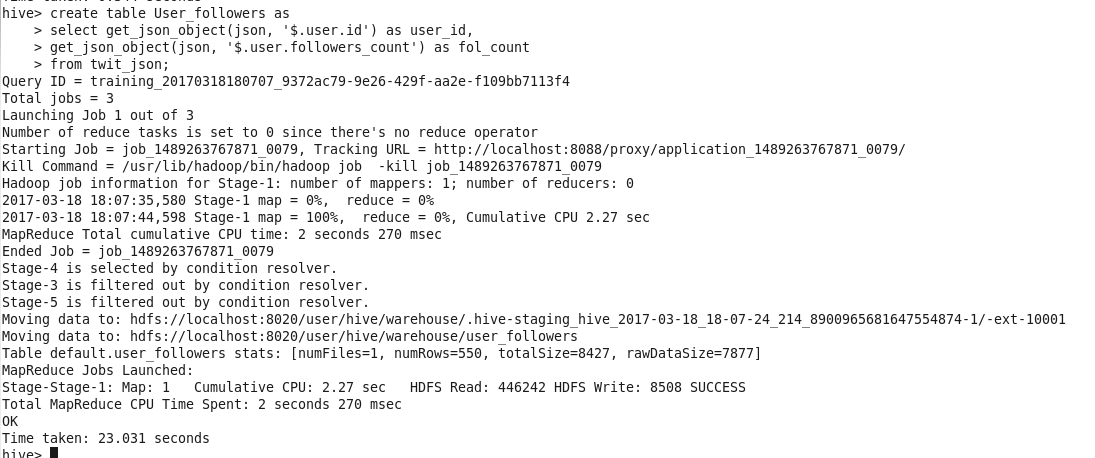
***drop table if exists User\_followers;***

***create table User\_followers as***

***select get\_json\_object(json, '$.user.id') as user\_id,***

***get\_json\_object(json, '$.user.followers\_count') as fol\_count***

***from twit\_json;***



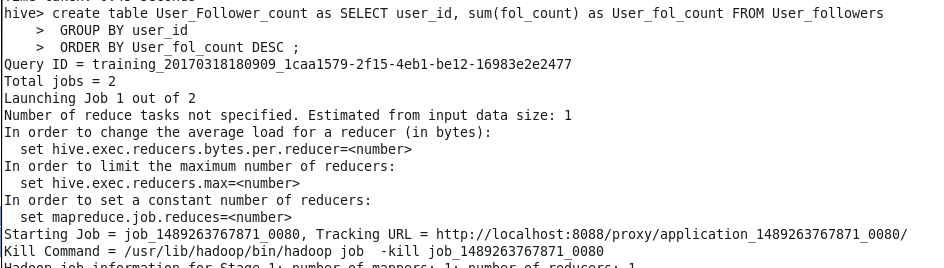
**Step 2: Sorting the users by user followers count**

***drop table if exists User\_Follower\_count;***

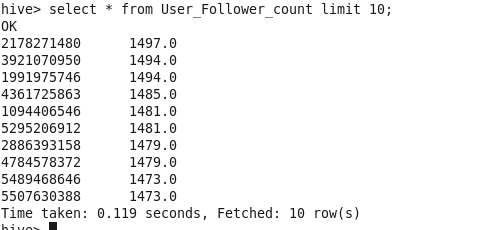
***create table User\_Follower\_count as SELECT user\_id, sum(fol\_count) as User\_fol\_count FROM User\_followers***

***GROUP BY user\_id***

***ORDER BY User\_fol\_count DESC ;***



**Output: Top 10 users based on followers count**



**Q1. e. What is the score for each tweet that was posted? Does the tweet have a positive or negative sentiment?**

**Step 1: Reading JSON data into columns**

***drop table if exists User\_data;***

***create table User\_data as***

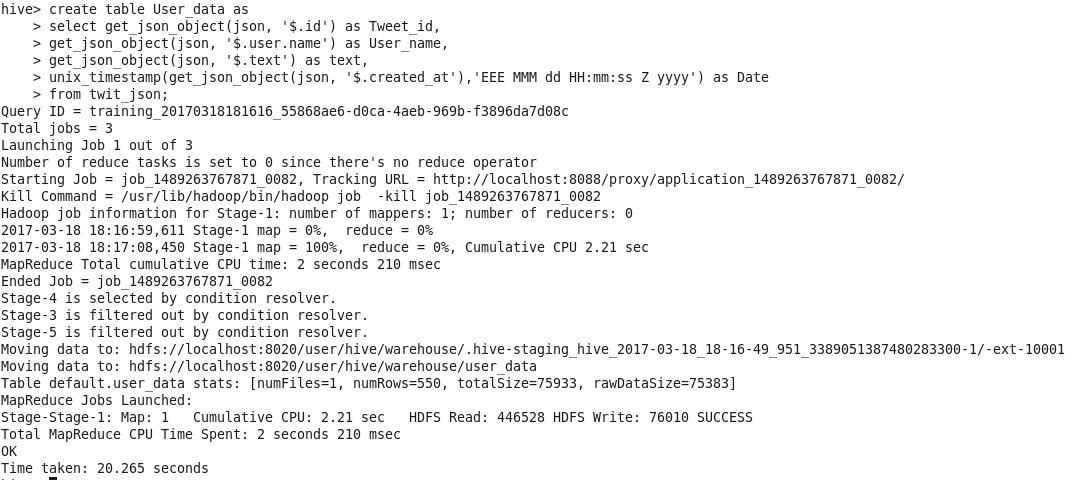
***select get\_json\_object(json, '$.id') as Tweet\_id,***

***get\_json\_object(json, '$.user.name') as User\_name,***

***get\_json\_object(json, '$.text') as text,***

***unix\_timestamp(get\_json\_object(json, '$.created\_at'),'EEE MMM dd HH:mm:ss Z yyyy') as Date***

***from twit\_json;***



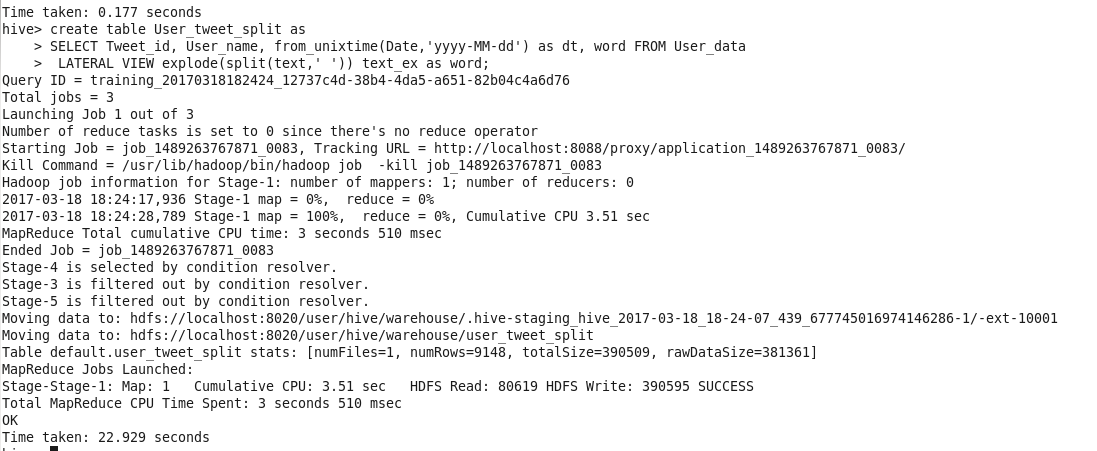
**Step 2: Split the text into words for each tweet**

***drop table if exists User\_tweet\_split;***

***create table User\_tweet\_split as***

***SELECT Tweet\_id, User\_name, from\_unixtime(Date,'yyyy-MM-dd') as dt, word FROM User\_data***

***LATERAL VIEW explode(split(text,' ')) text\_ex as word;***

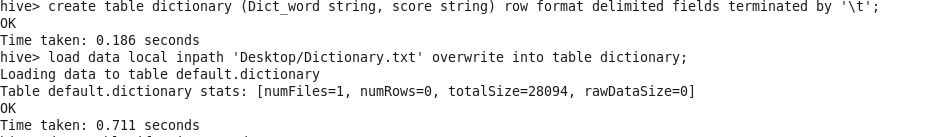


**Step 3: Reading the dictionary data into hive**

***drop table if exists dictionary;***

***create table dictionary (Dict\_word string, score string) row format delimited fields terminated by '\t';***

***load data local inpath 'Desktop/Dictionary.txt' overwrite into table dictionary;***



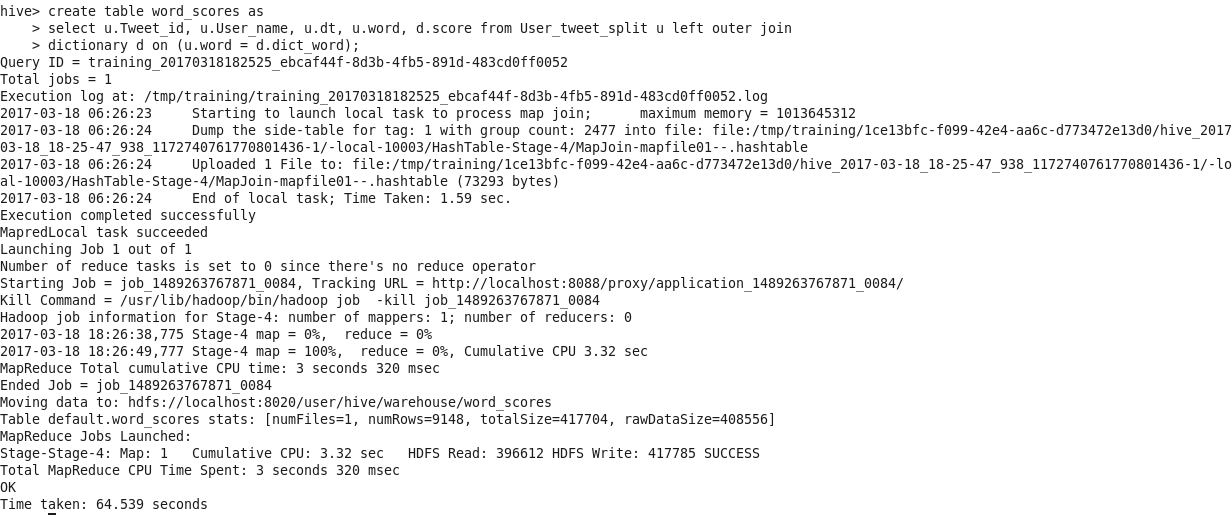
**Step 4: Mapping the scores from dictionary to twitter data**

***drop table if exists word\_scores;***

***create table word\_scores as***

***select u.Tweet\_id, u.User\_name, u.dt, u.word, d.score from User\_tweet\_split u left outer join***

***dictionary d on (u.word = d.dict\_word);***



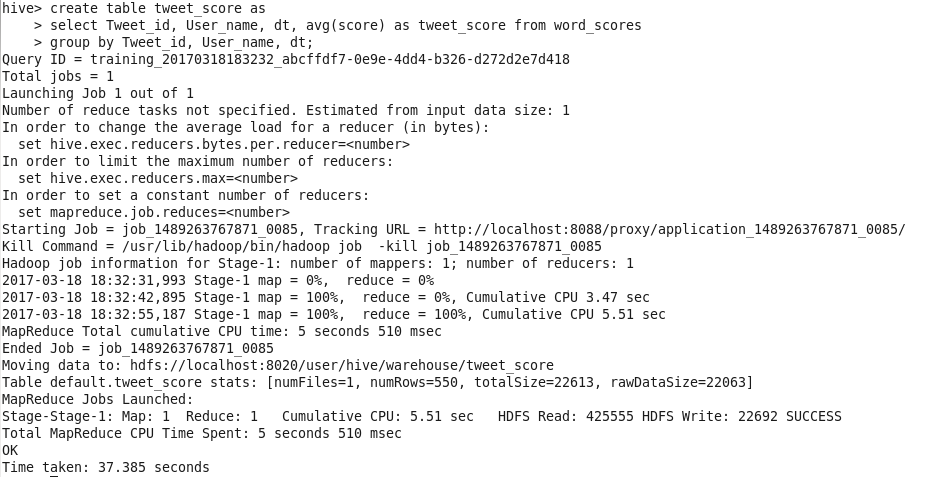
**Step 5: Average the score of words by tweet id**

***drop table if exists tweet\_score;***

***create table tweet\_score as***

***select Tweet\_id, User\_name, dt, avg(score) as tweet\_score from word\_scores***

***group by Tweet\_id, User\_name, dt;***



**Step 6:**

***drop table if exists tweet\_sentiment;***

***create table tweet\_sentiment as***

***select Tweet\_id, User\_name, dt,***

***case***

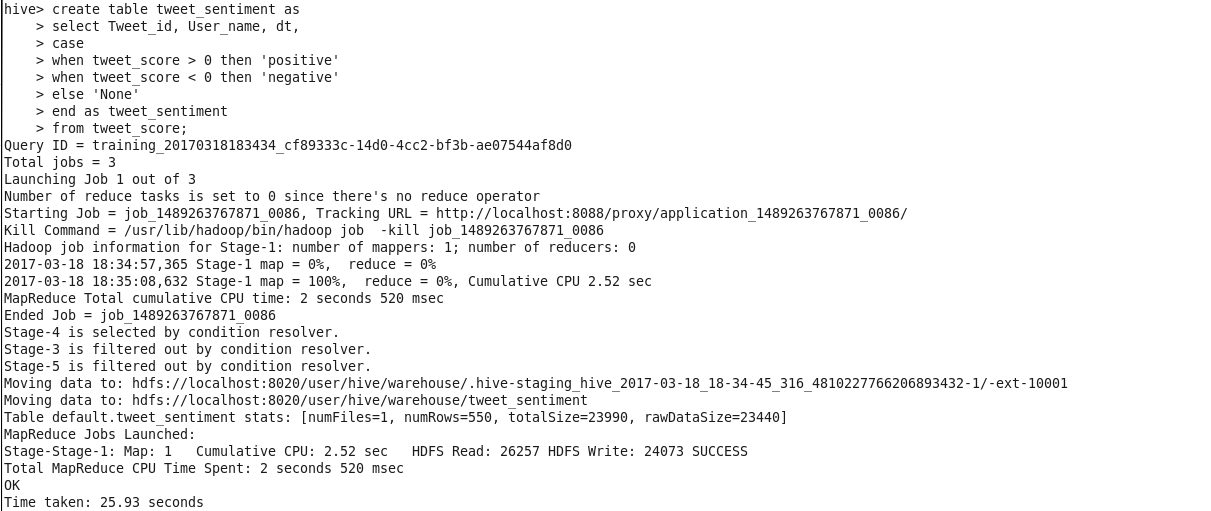
***when tweet\_score > 0 then 'positive'***

***when tweet\_score < 0 then 'negative'***

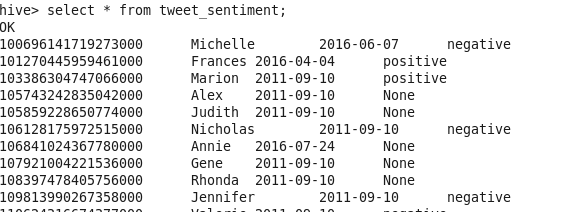
***else 'None'***

***end as tweet\_sentiment***

***from tweet\_score;***



**Output: Out of 550 tweets, the sentiment is shown for 10 tweets in the screen shot below.**



**Q2. Did you find any problem in the previous question? If so briefly explain what exactly the problem is and what solution would you propose.**

Problem: The score given in the data dictionary is very subjective. For example, the word ‘condemn’ or ‘condemnation’ have negativity associated with it. But in the dictionary ‘condemn’ is given positive score and ‘condemnation’ is given negative score. The sentiment of the tweet is not accurately captured because of this reason.

Solution: The score should be assigned based on the connotative meaning of the word. Also, the list of words in the dictionary used should be made as exhaustive as possible to capture the sentiment of the tweet accurately.