**Assignment-1**

**Twitter Data Sentiment Analysis**

**Basic Initial Steps:**

1. Starting hive :

$DEV1/scripts/training\_setup\_dev1.sh

hive

1. Loading Dictionary.txt file into hive:

CREATE TABLE dictionary\_lk0177(word string, rating int)

row format delimited

fields terminated by '\t';

LOAD DATA LOCAL INPATH 'Desktop/Dictionary.txt' INTO TABLE dictionary\_lk0177;

1. Loading JSON file into hive:

First whole file is loaded into twitter\_json table as string and then parsed using **get\_json\_object**

CREATE TABLE twitter\_json ( json string );

LOAD DATA LOCAL INPATH 'Desktop/Twitter.json' INTO TABLE twitter\_json;

CREATE TABLE twitter\_lk0177 as

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

unix\_timestamp(get\_json\_object(twitter\_json.json, "$.created\_at"),

"EEE MMM d HH:mm:ss Z yyyy") as created\_at,

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

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

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

get\_json\_object(twitter\_json.json, '$.in\_reply\_to\_screen\_name') as in\_reply\_to\_screen\_name,

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

get\_json\_object(twitter\_json.json, '$.user.id') as u\_id,

get\_json\_object(twitter\_json.json, '$.user.id\_str') as id\_str,

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

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

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

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

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

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

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

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

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

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

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

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

profile\_background\_color

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

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

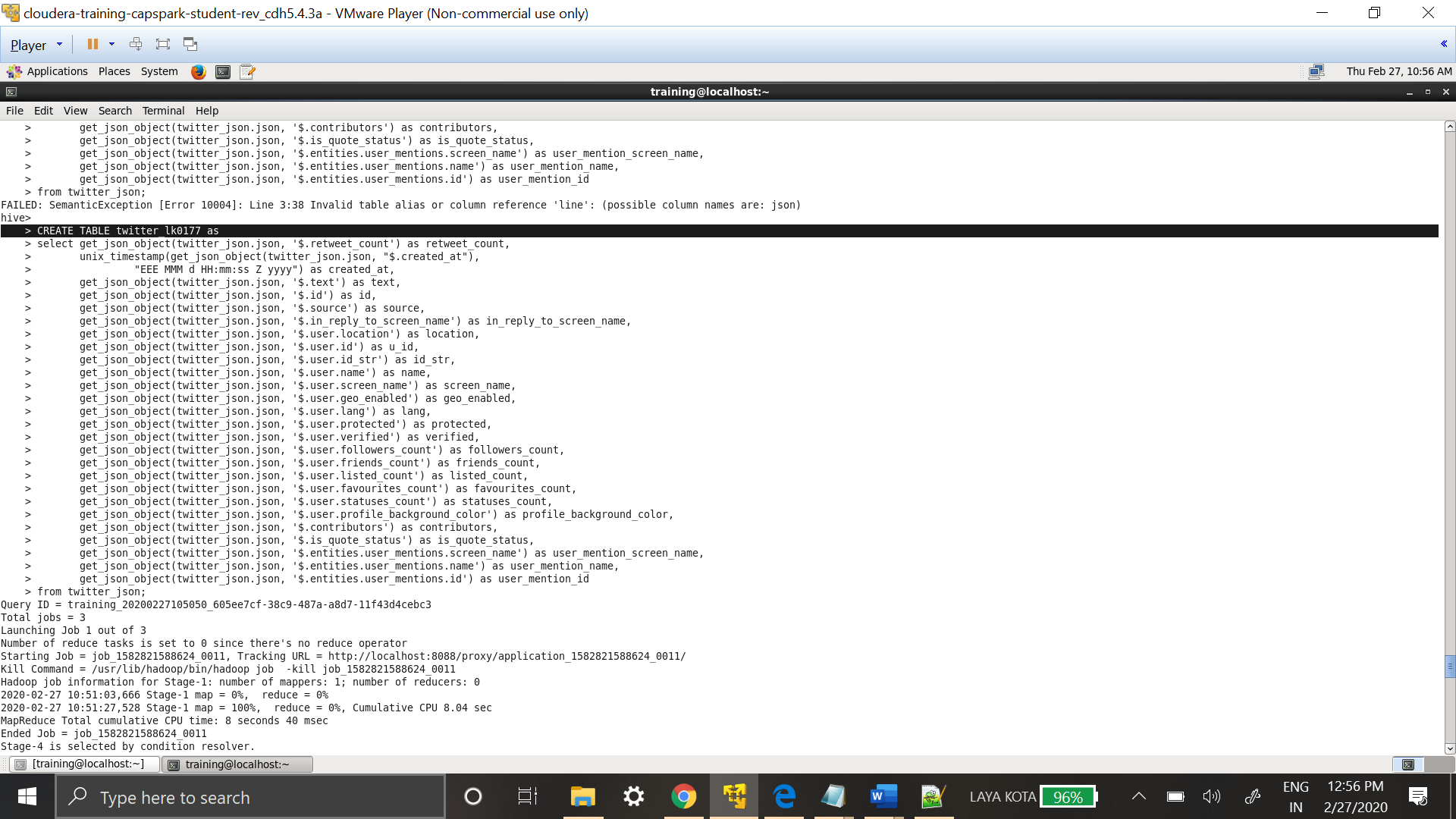
get\_json\_object(twitter\_json.json,'$.entities.user\_mentions.screen\_name')as

user\_mention\_screen\_name,

get\_json\_object(twitter\_json.json, '$.entities.user\_mentions.name') as user\_mention\_name,

get\_json\_object(twitter\_json.json, '$.entities.user\_mentions.id') as user\_mention\_id

from twitter\_json;



In **twitter\_lk0177** table, data is stored as string in order to convert **created\_at** column into required date column **unix\_timestamp**  is used which will store column values as BIGINT, this step will help in further analysis.

**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)**

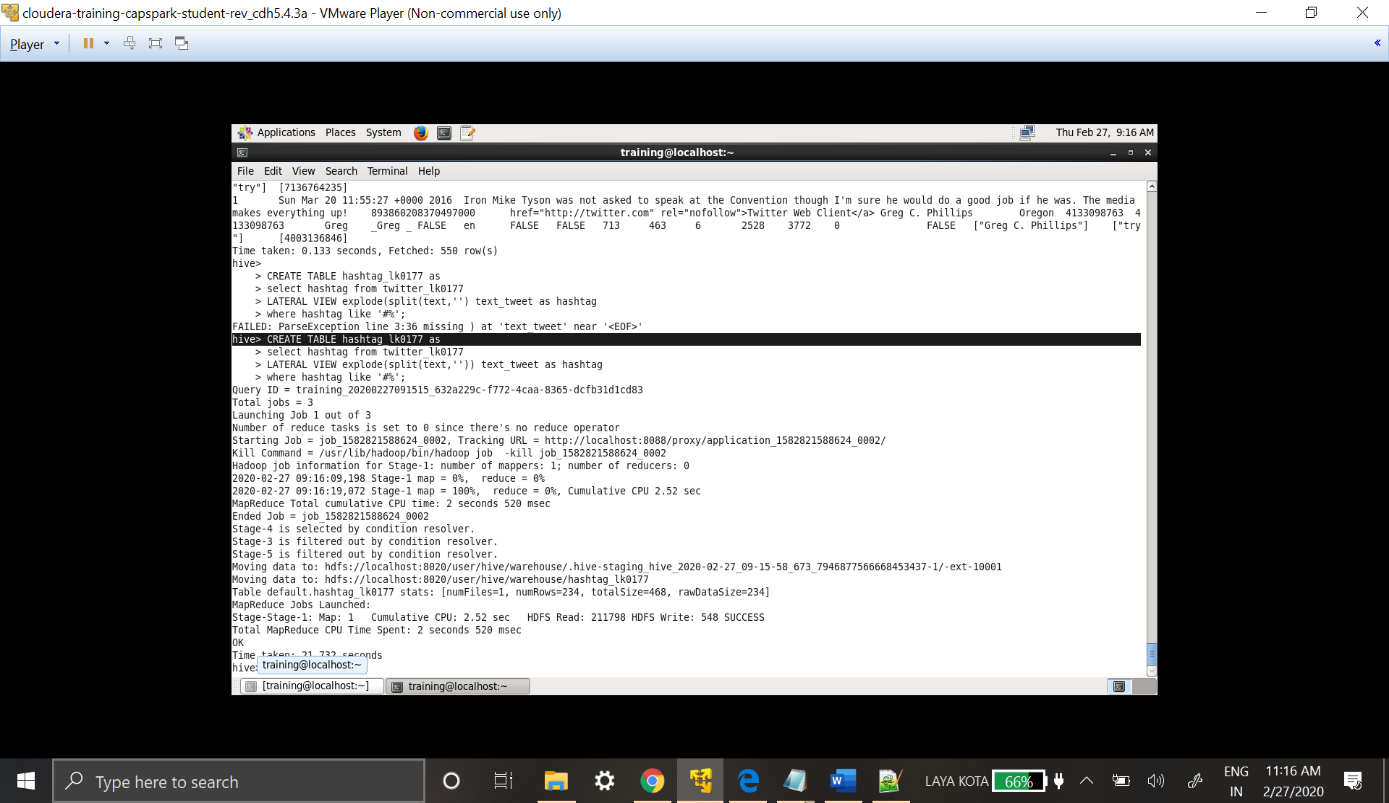
Hashtags are in between the text in **text** column, **split** function is used to extract hashtags and **hashtag\_lk0177** table is created to store the extracted hashtags.

CREATE TABLE hashtag\_lk0177 as

select hashtag from twitter\_lk0177

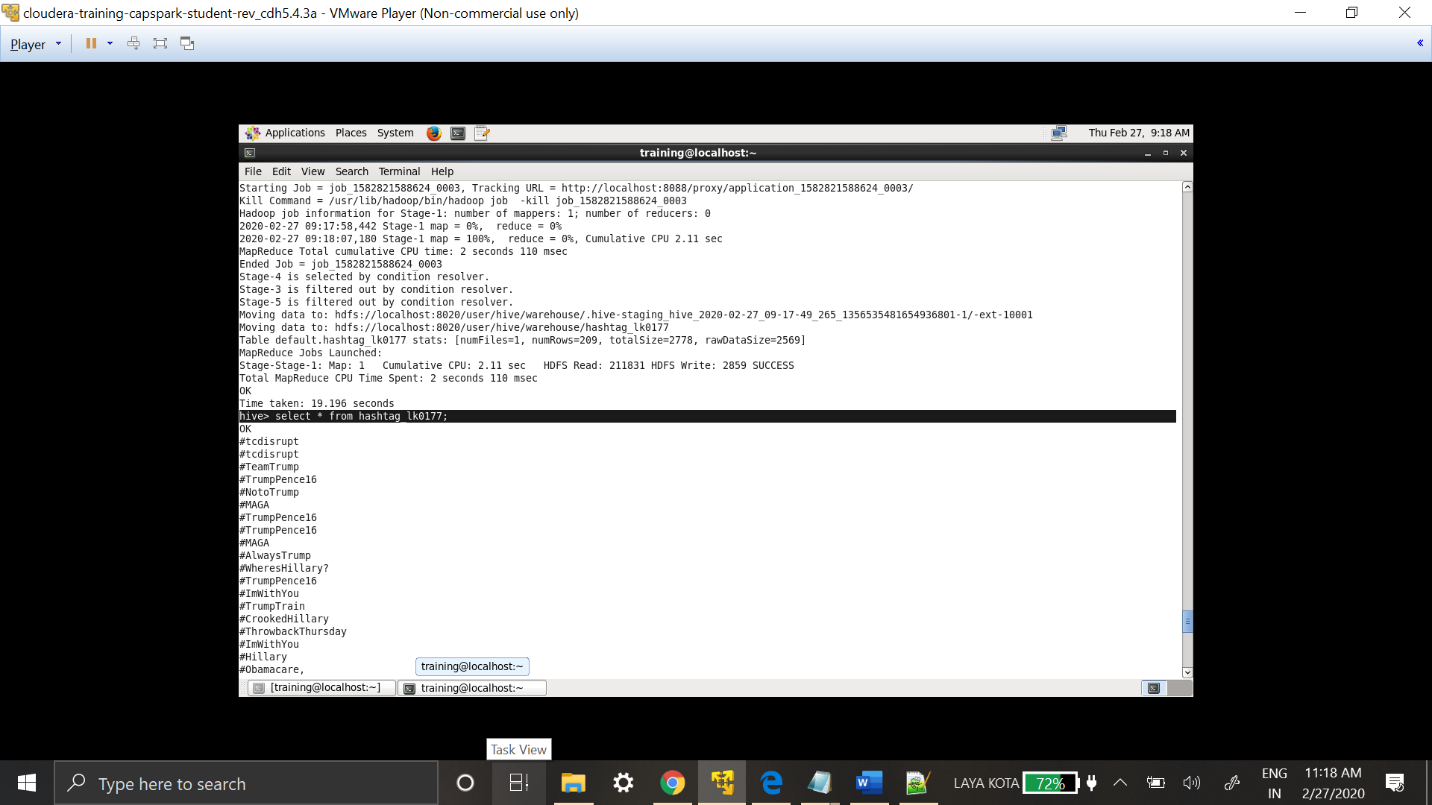
LATERAL VIEW explode(split(text,' ')) text\_tweet as hashtag

where hashtag like '#%';



Select \* from hashtag\_lk0177;

Query is used to validate the table.



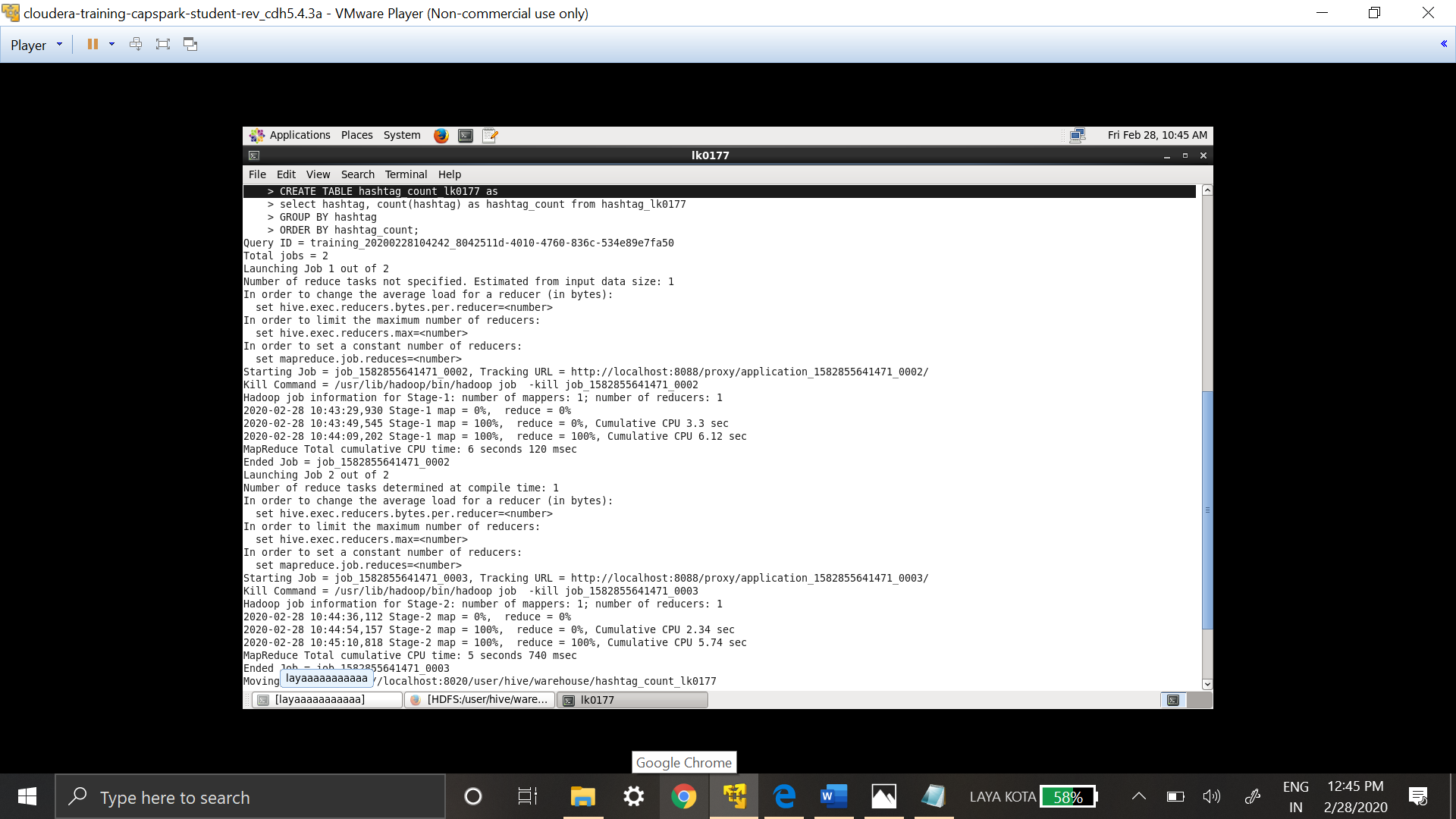
The second part of the question is count of hashtags, in order to obtain that **hashtag\_count\_lk0177** table is created to store hashtags and their count.

CREATE TABLE hashtag\_count\_lk0177 as

select hashtag, count(hashtag) as hashtag\_count from hashtag\_lk0177

GROUP BY hashtag

ORDER BY hashtag\_count;



Select \* from hashtag\_count\_lk0177 ORDER BY hashtag\_count DESC;

Query is used to validate table creation. Hashtag\_count\_lk0177 table has all the hashtags and its counts.

During the table creating ORDER BY hashtag\_count DESC; can be used to print the results in descending order instead of using it during select statement.

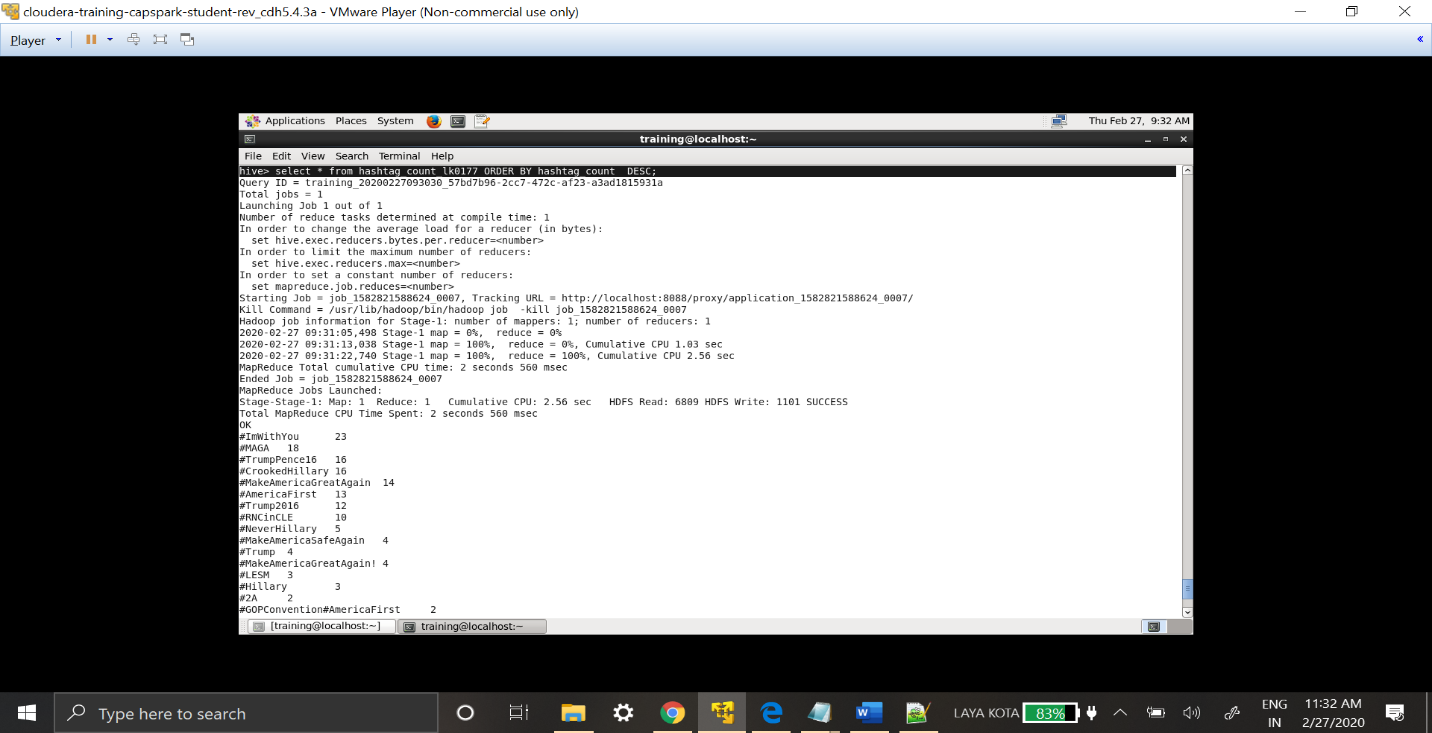
Most commonly used hashtags are:

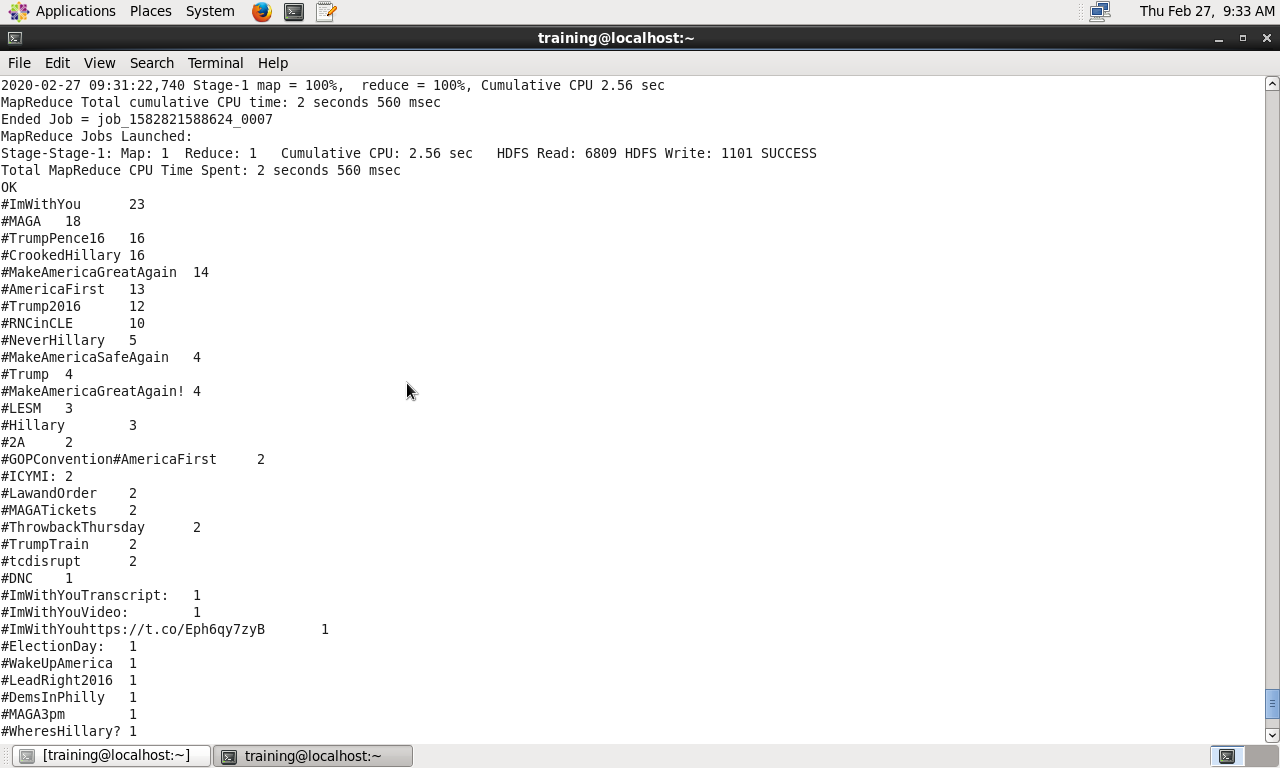
#ImWithYou – 23

#MAGA - 18

#TrumpPence – 16

#CrookedHillary – 16





**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’]**

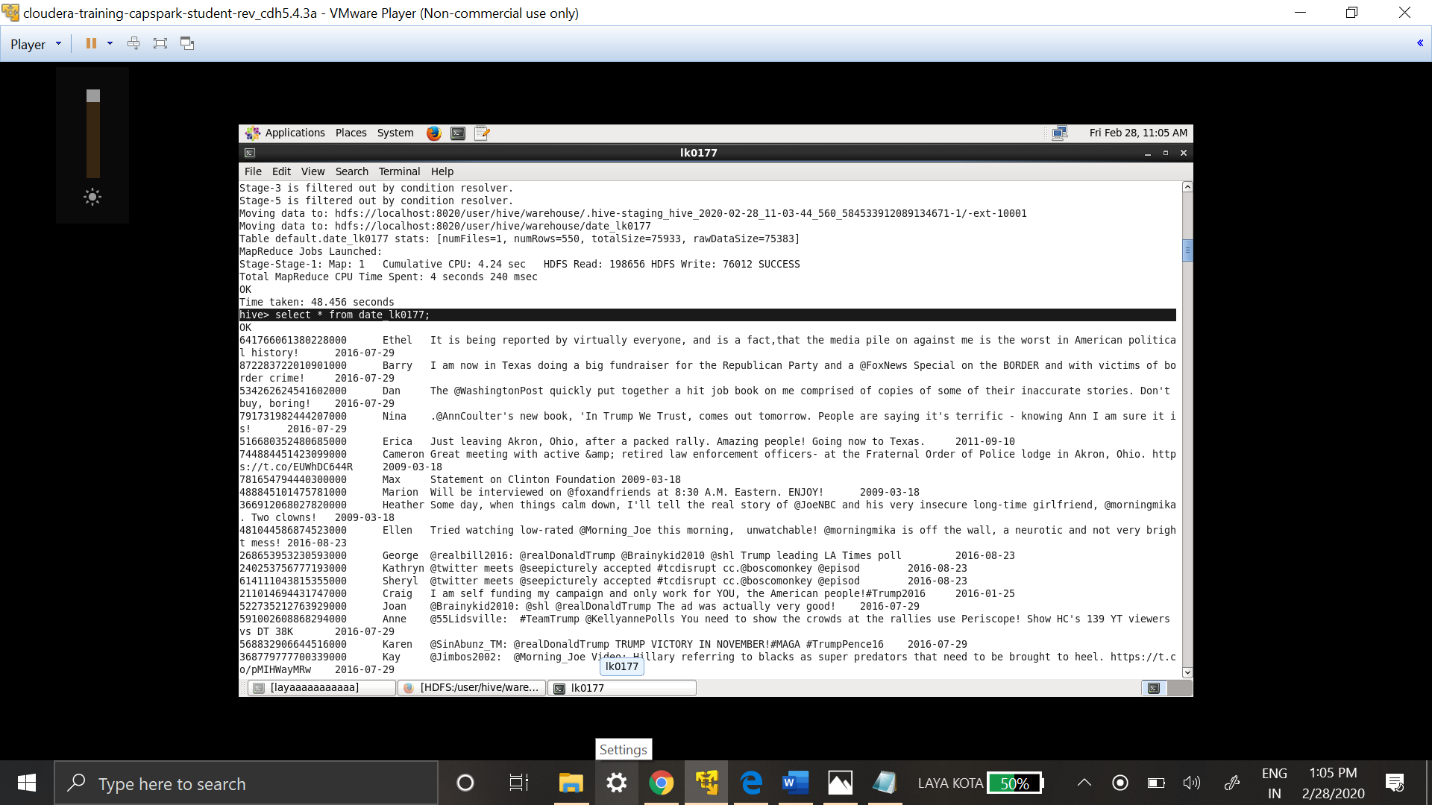
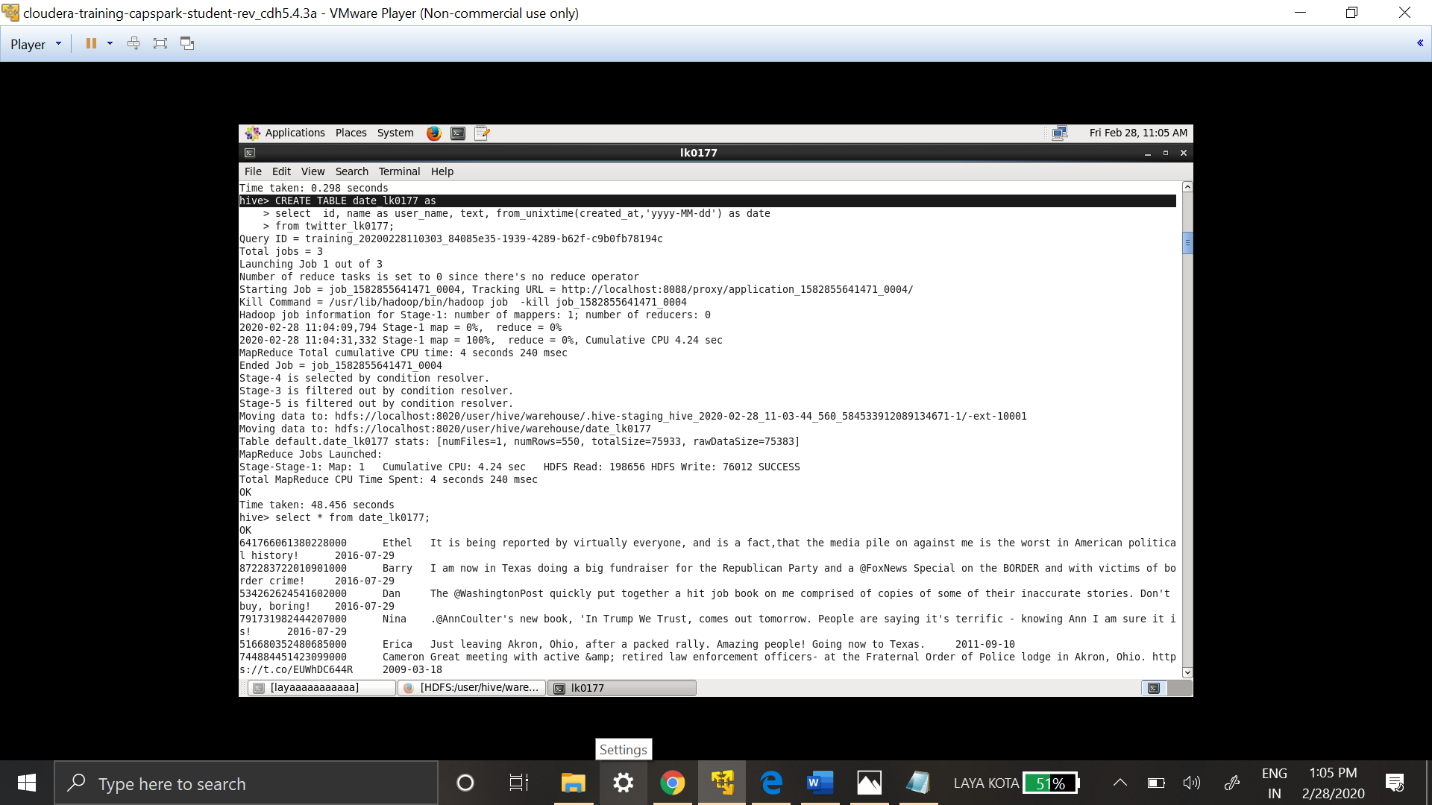
The approach to achieve solution to above question is to first convert the date into mentioned

‘yyyy-MM-dd’ format by using **from\_unixtime** function and store id, name, text and data in one table. Next using **split** function to extract hashtags from text column.

CREATE TABLE date\_lk0177 as

select id, name as user\_name, text, from\_unixtime(created\_at,'yyyy-MM-dd') as date

from twitter\_lk0177;



Select \* from date\_lk0177; is used to validate table creation.

**Creating a table with hashtags and date:**

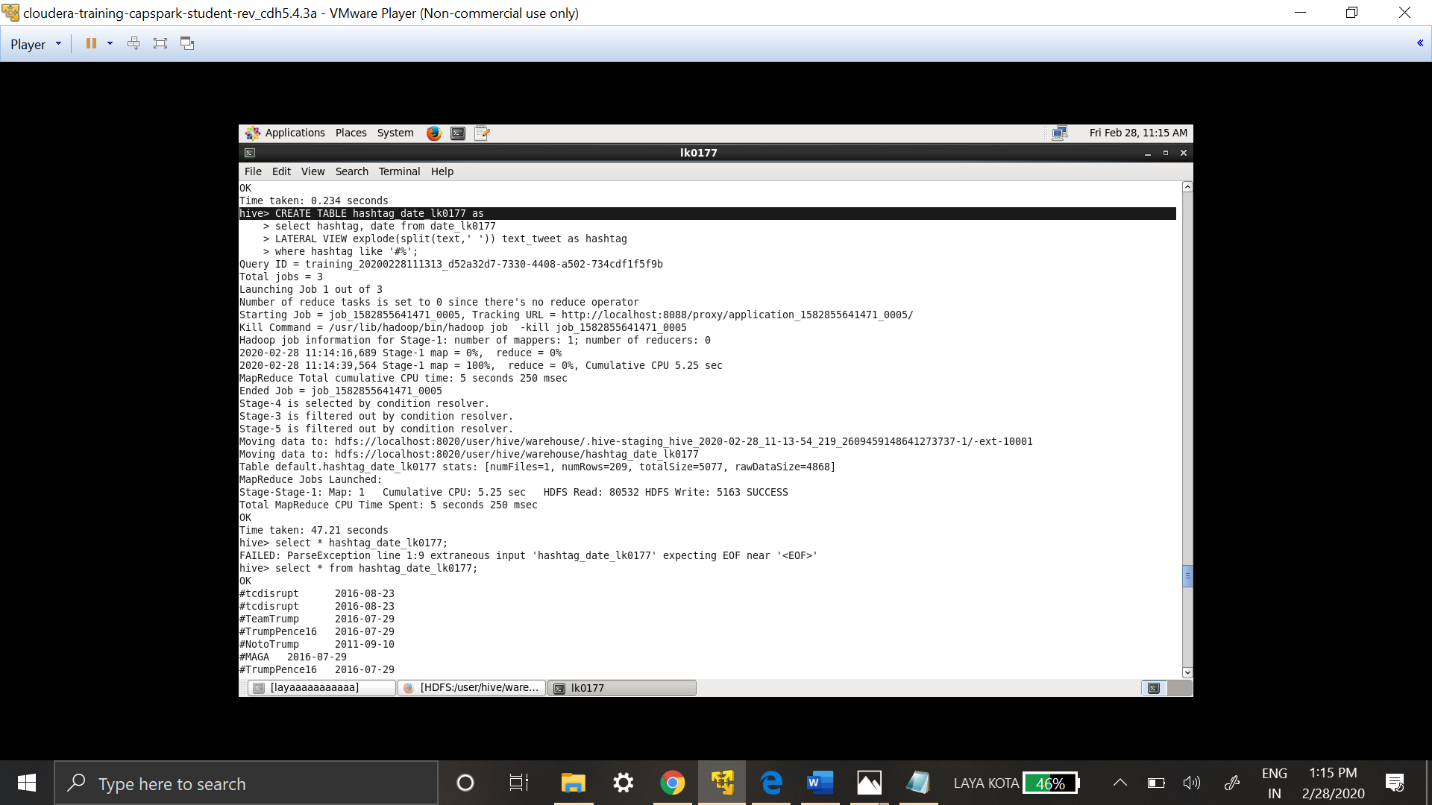
Here two columns are used date column from date\_lk0177 table and text is split using split functions to get only hashtags from text.

CREATE TABLE hashtag\_date\_lk0177 as

select hashtag, date from date\_lk0177

LATERAL VIEW explode(split(text,' ')) text\_tweet as hashtag

where hashtag like '#%';



Select \* from hashtag\_date\_lk0177;

This query is used to validate the table creation



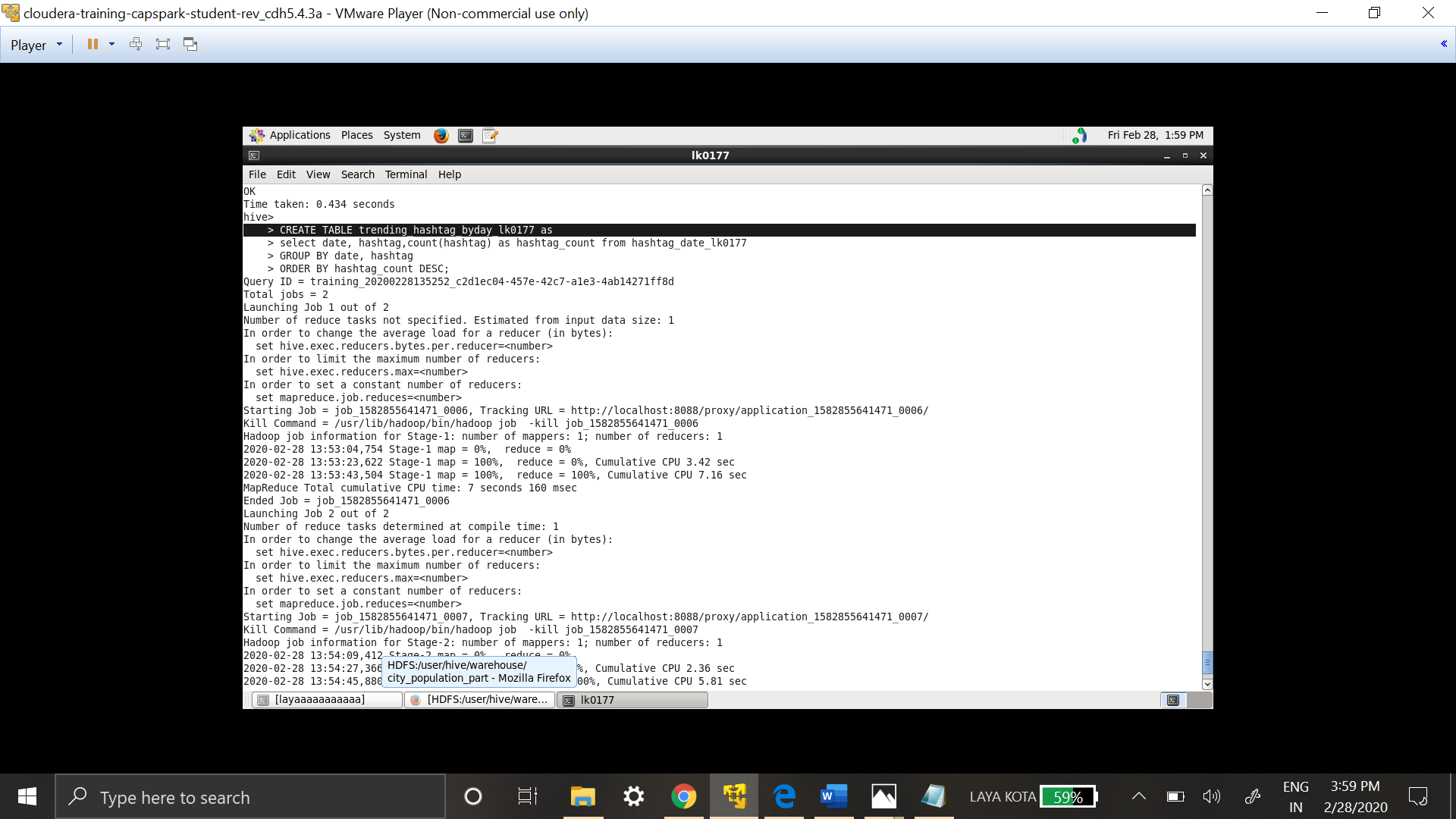
**Creation of table which has trending hashtags by date :**

CREATE TABLE trending\_hashtag\_byday\_lk0177 as

select date, hashtag,count(hashtag) as hashtag\_count from hashtag\_date\_lk0177

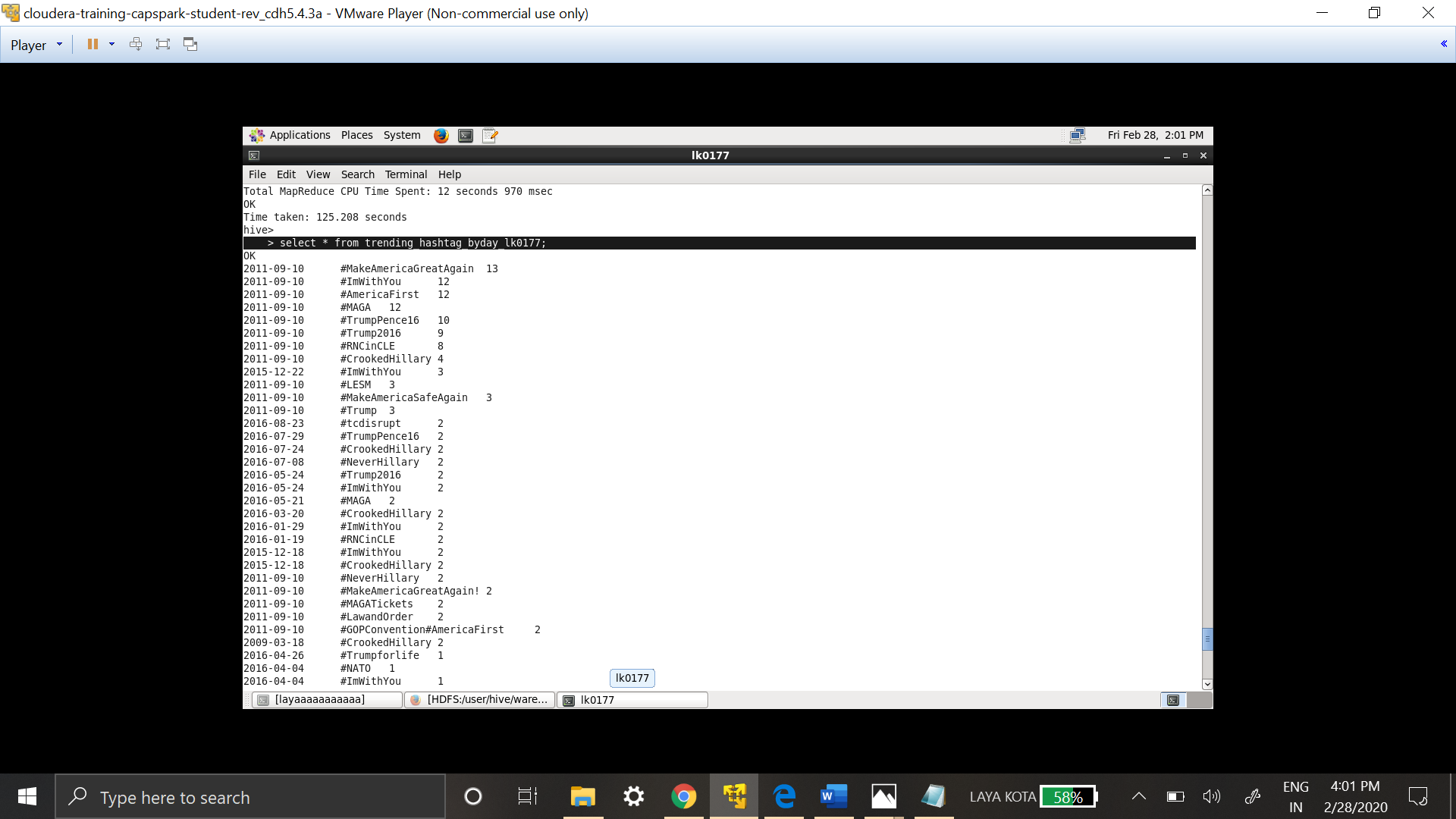
GROUP BY date, hashtag

ORDER BY hashtag\_count DESC;



In order to get the most trending hashtag by date :

Select \* from trending\_hashtag\_byday\_lk0177;



Most trending hashtags are # MakeAmericaGreatAgain, #ImWithYou, #MAGA, #AmericaFirst.

**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. (20 points)**

Approach to achieve above sentiment analysis, its required:

1. To have a table with all the required columns of twitter\_lk0177.
2. To create join between twitter table and dictionary, matching all the words in dictionary with text in tweets.
3. Find the rating of all the matched words and provide sentiment analysis results.

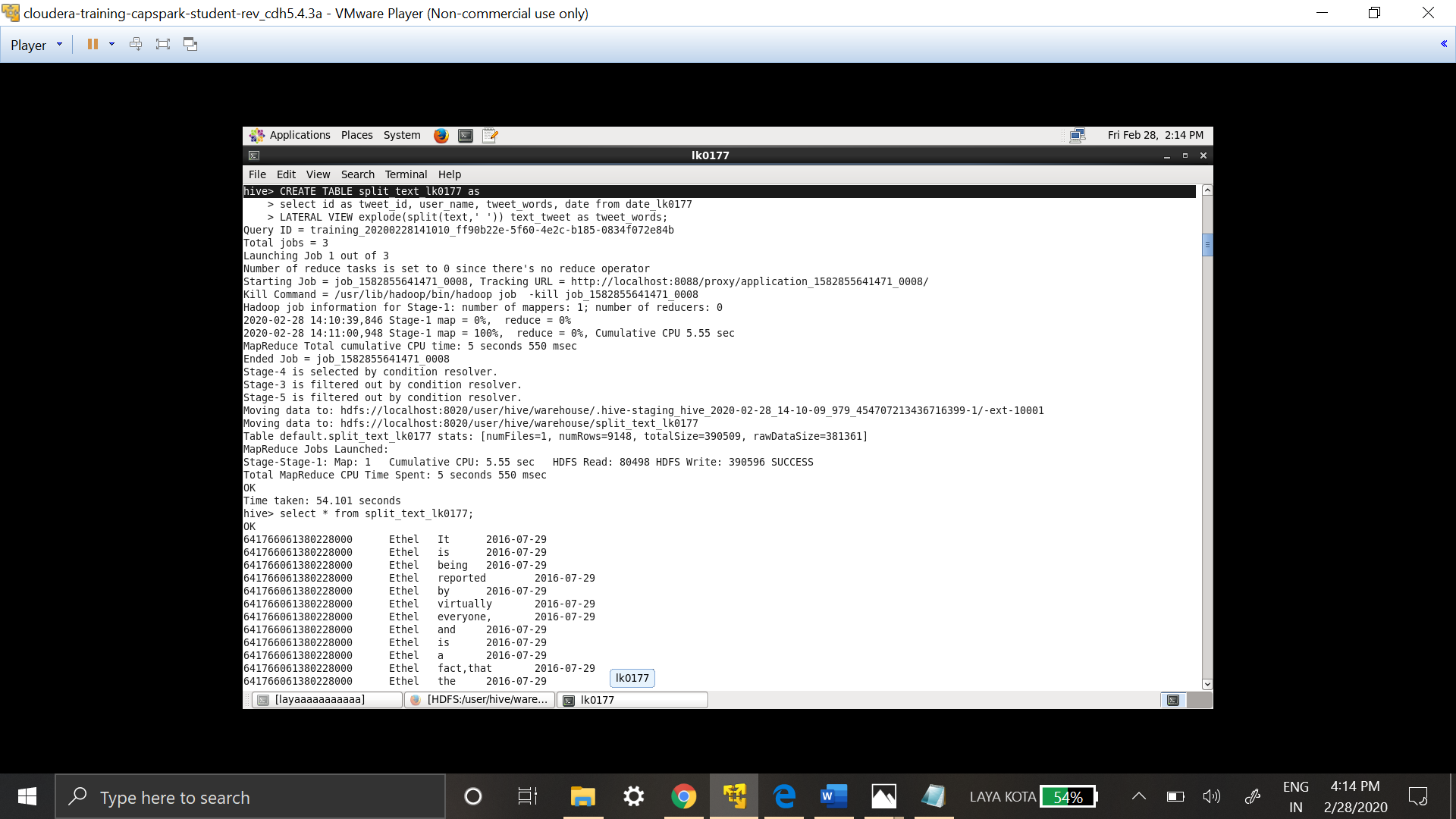
**Creation of table with all the required columns from twitter data:**

Date table created earlier is used here. It has all the required columns and now text data is split into words in order to compare them with dictionary words.

CREATE TABLE split\_text\_lk0177 as

select id as tweet\_id, user\_name, tweet\_words, date from date\_lk0177

LATERAL VIEW explode(split(text,' ')) text\_tweet as tweet\_words;

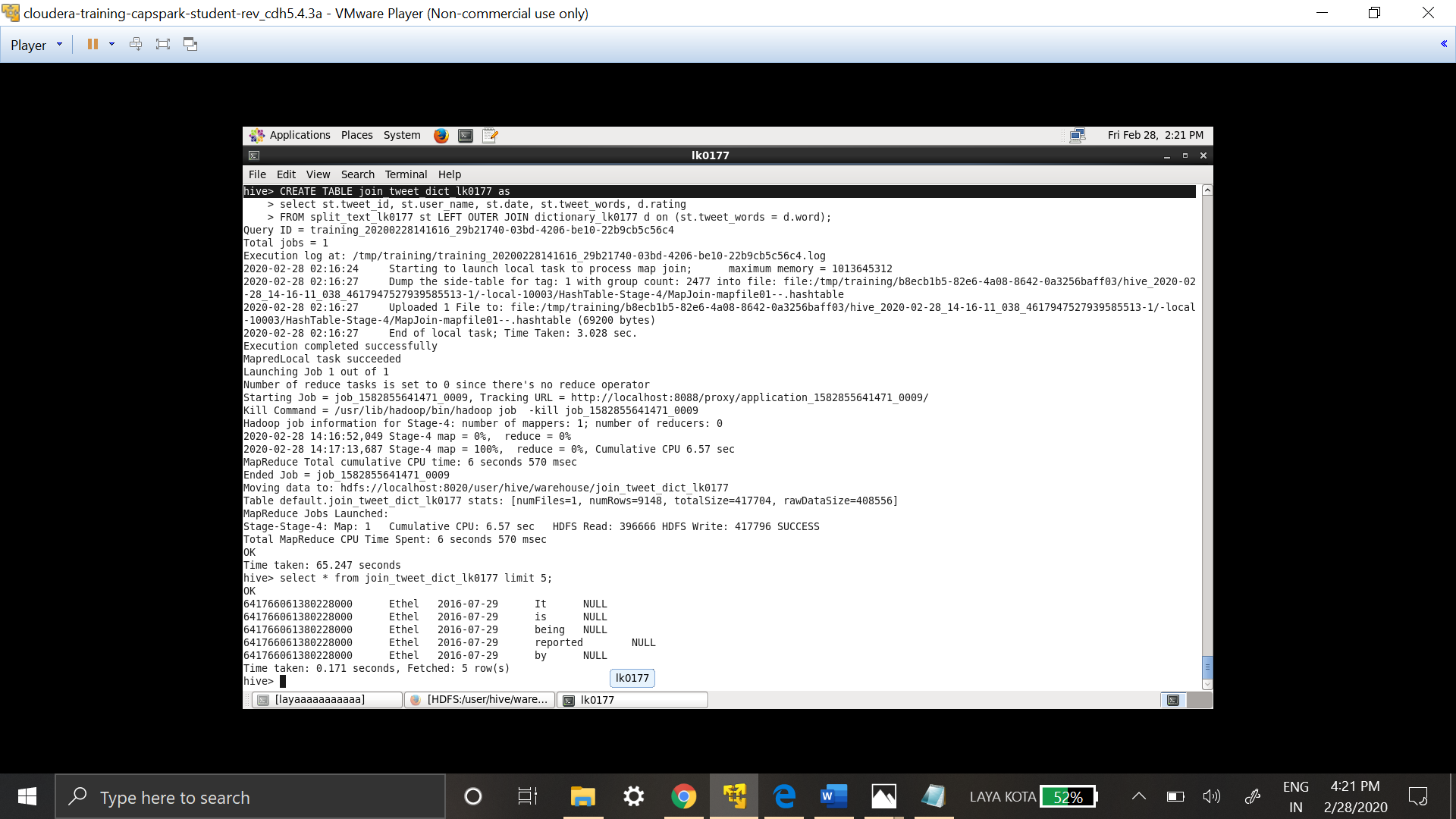


**Join between split\_text table and dictionary table:**

CREATE TABLE join\_tweet\_dict\_lk0177 as

select st.tweet\_id, st.user\_name, st.date, st.tweet\_words, d.rating

FROM split\_text\_lk0177 st LEFT OUTER JOIN dictionary\_lk0177 d on (st.tweet\_words = d.word);



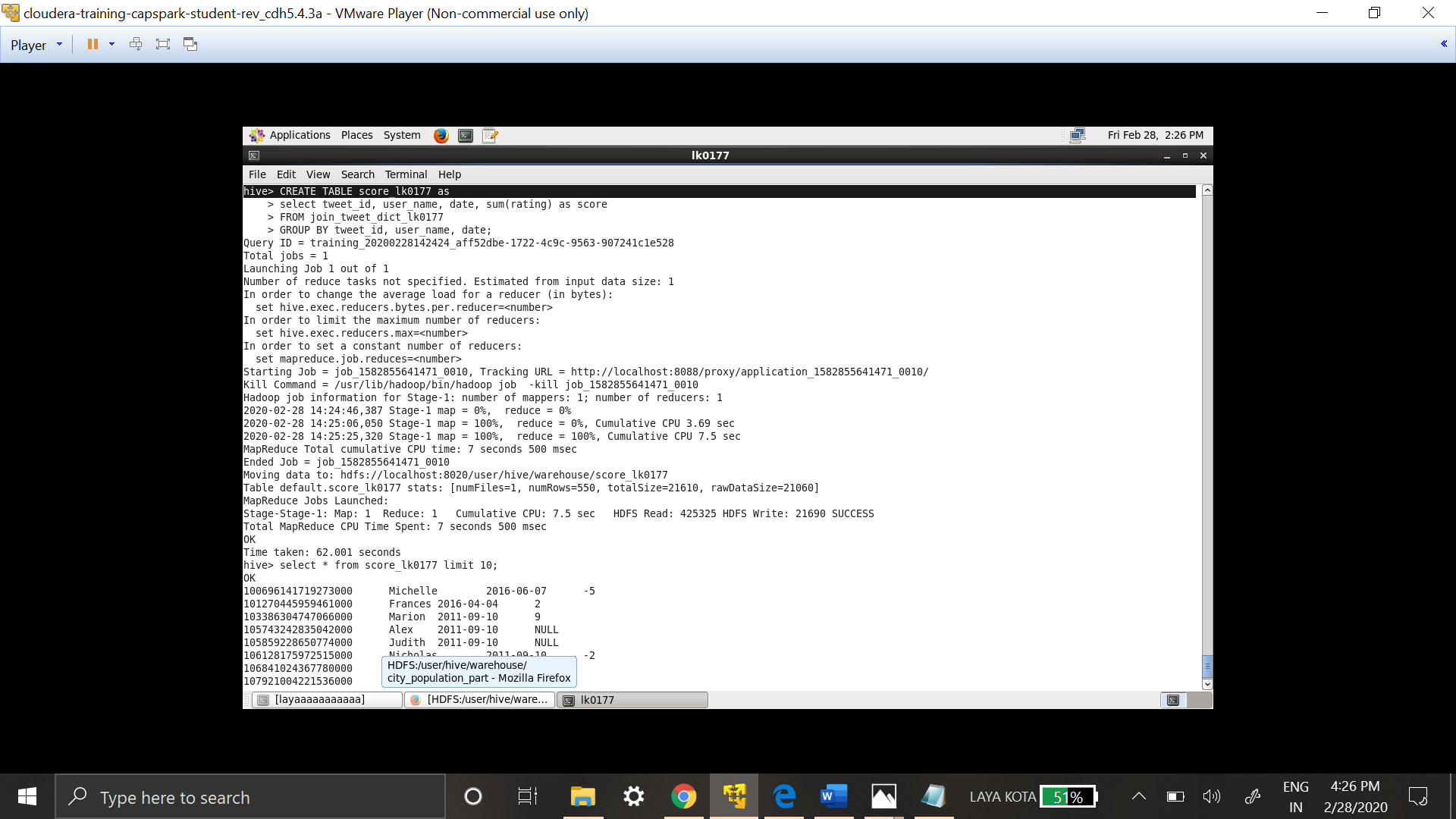
**Creation of table that has score – over all rating and providing sentiment analysis results:**

CREATE TABLE score\_lk0177 as

select tweet\_id, user\_name, date, sum(rating) as score

FROM join\_tweet\_dict\_lk0177

GROUP BY tweet\_id, user\_name, date;



Score table has all the sum of ratings of tweet data.

CREATE table sentiment\_analysis\_lk0177 as

SELECT tweet\_id, user\_name, date, score,

CASE WHEN score > 0 THEN 'Positive'

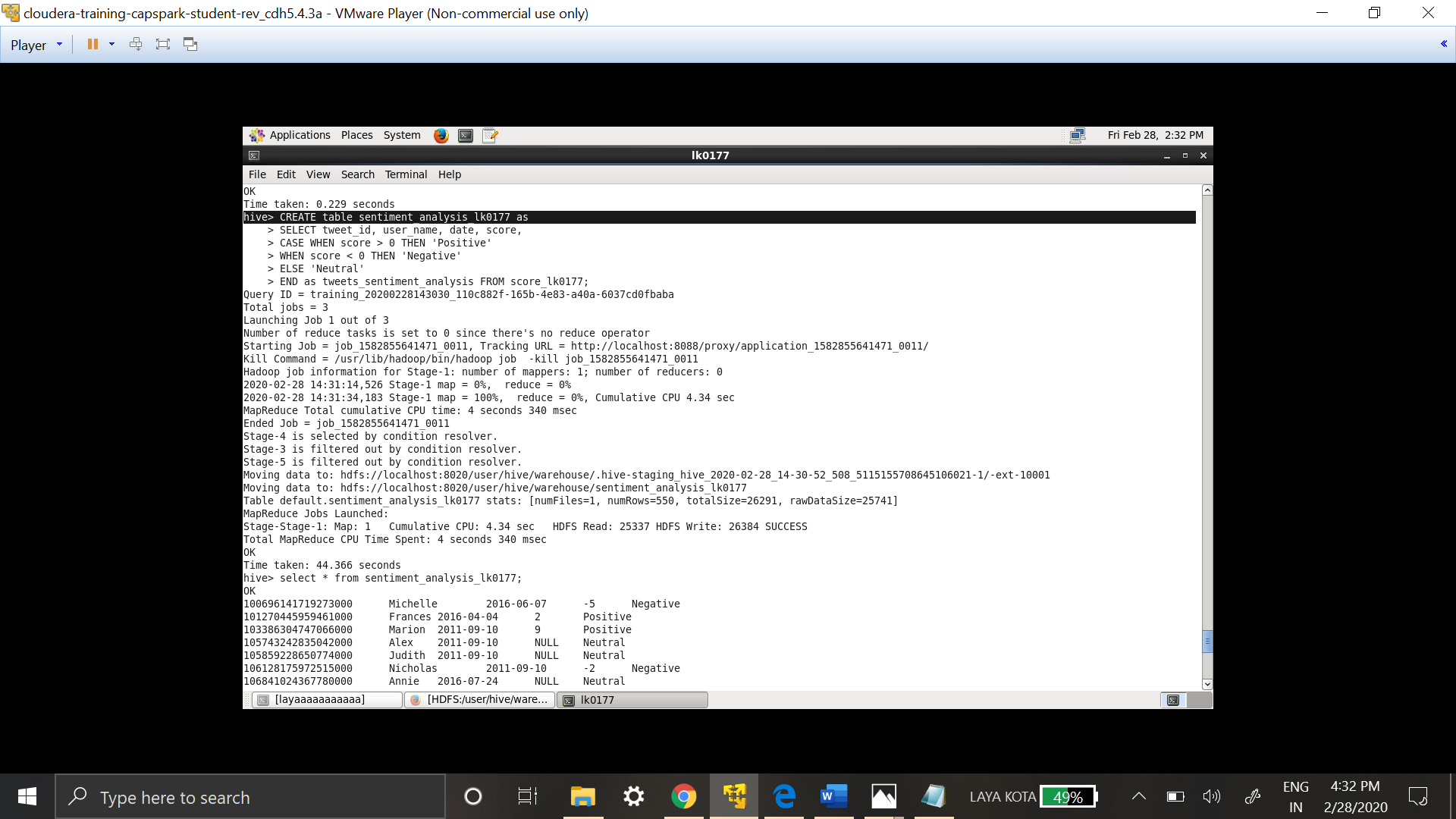
WHEN score < 0 THEN 'Negative'

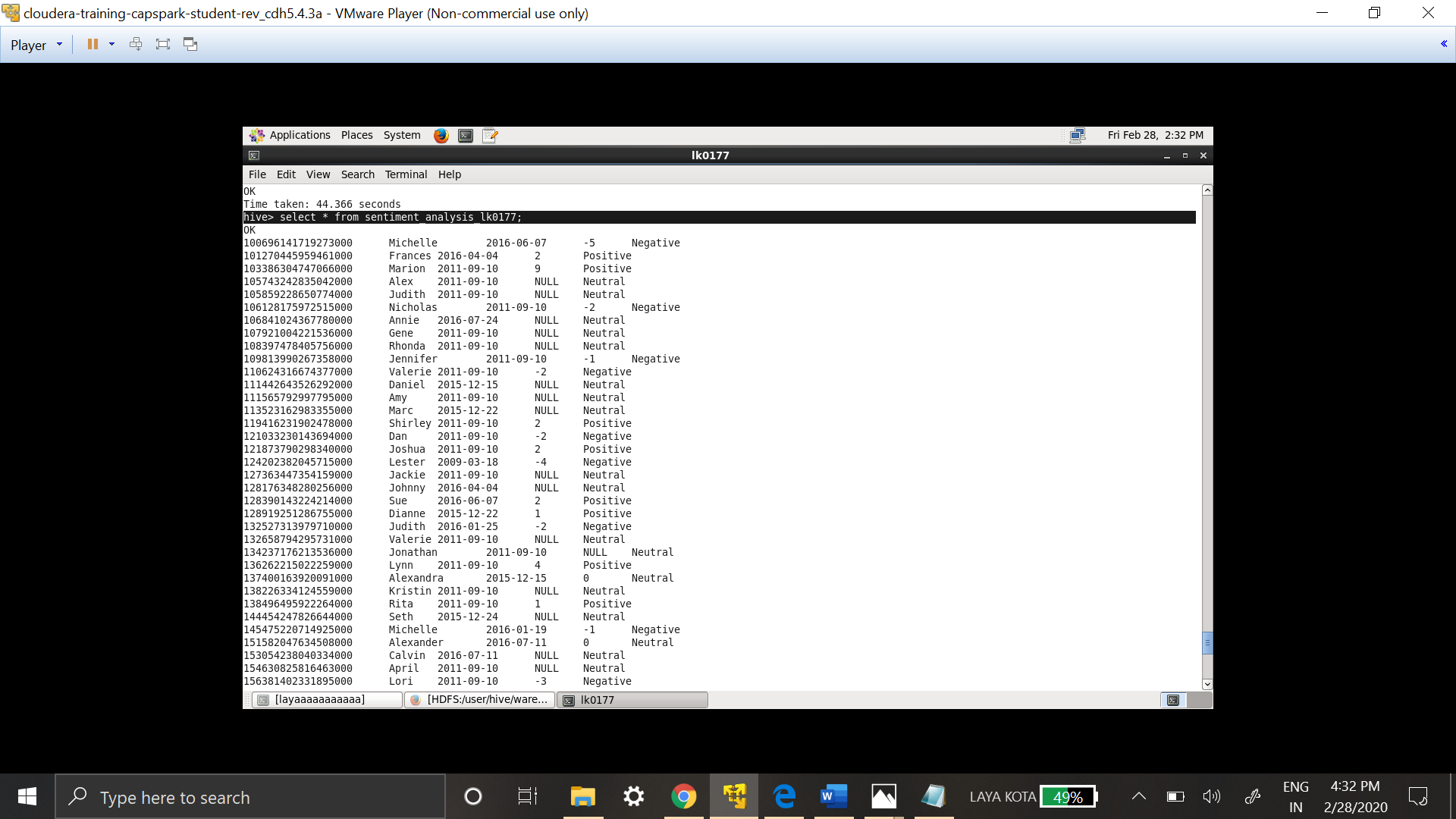
ELSE 'Neutral'

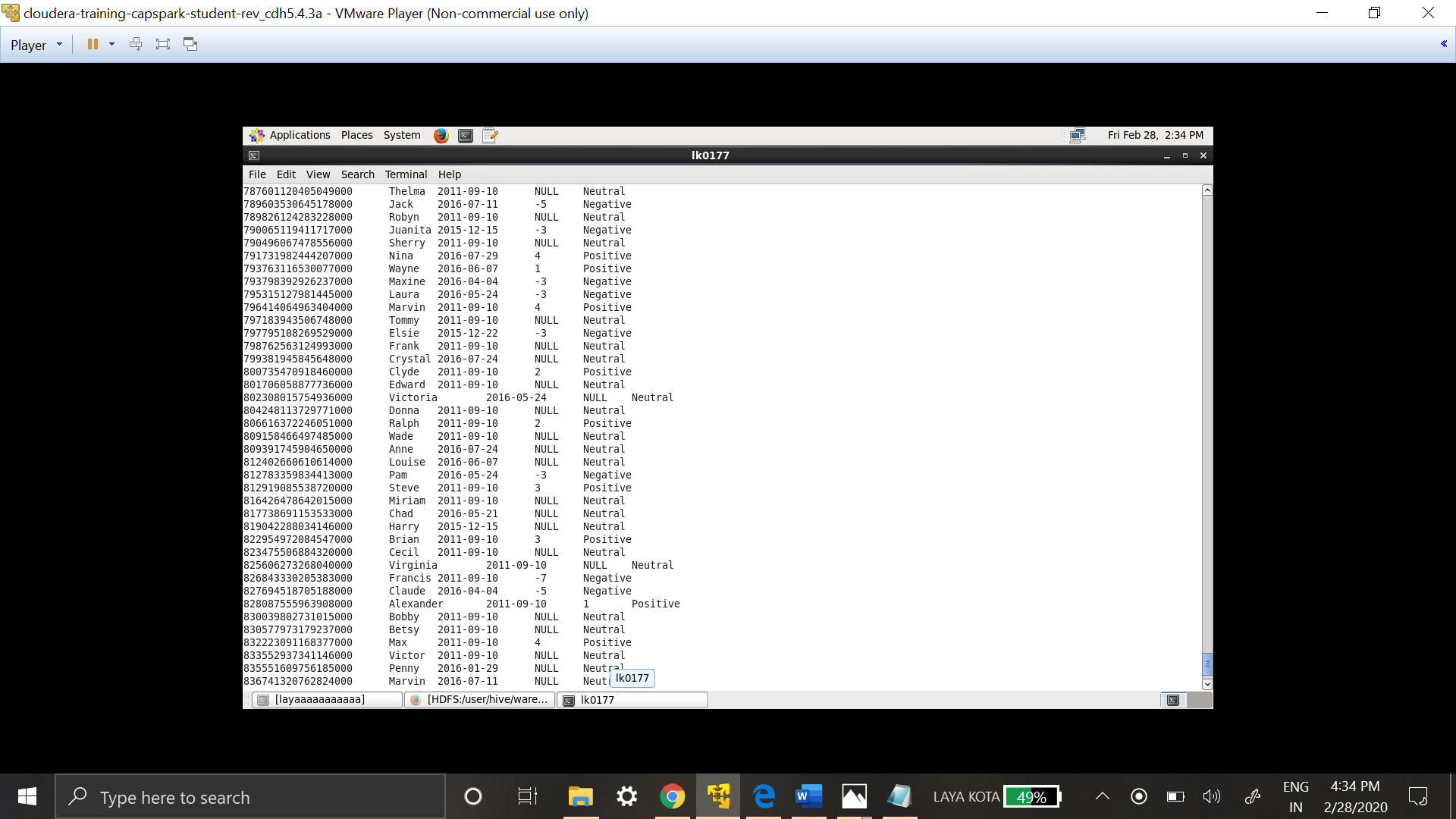
END as tweets\_sentiment\_analysis FROM score\_lk0177;

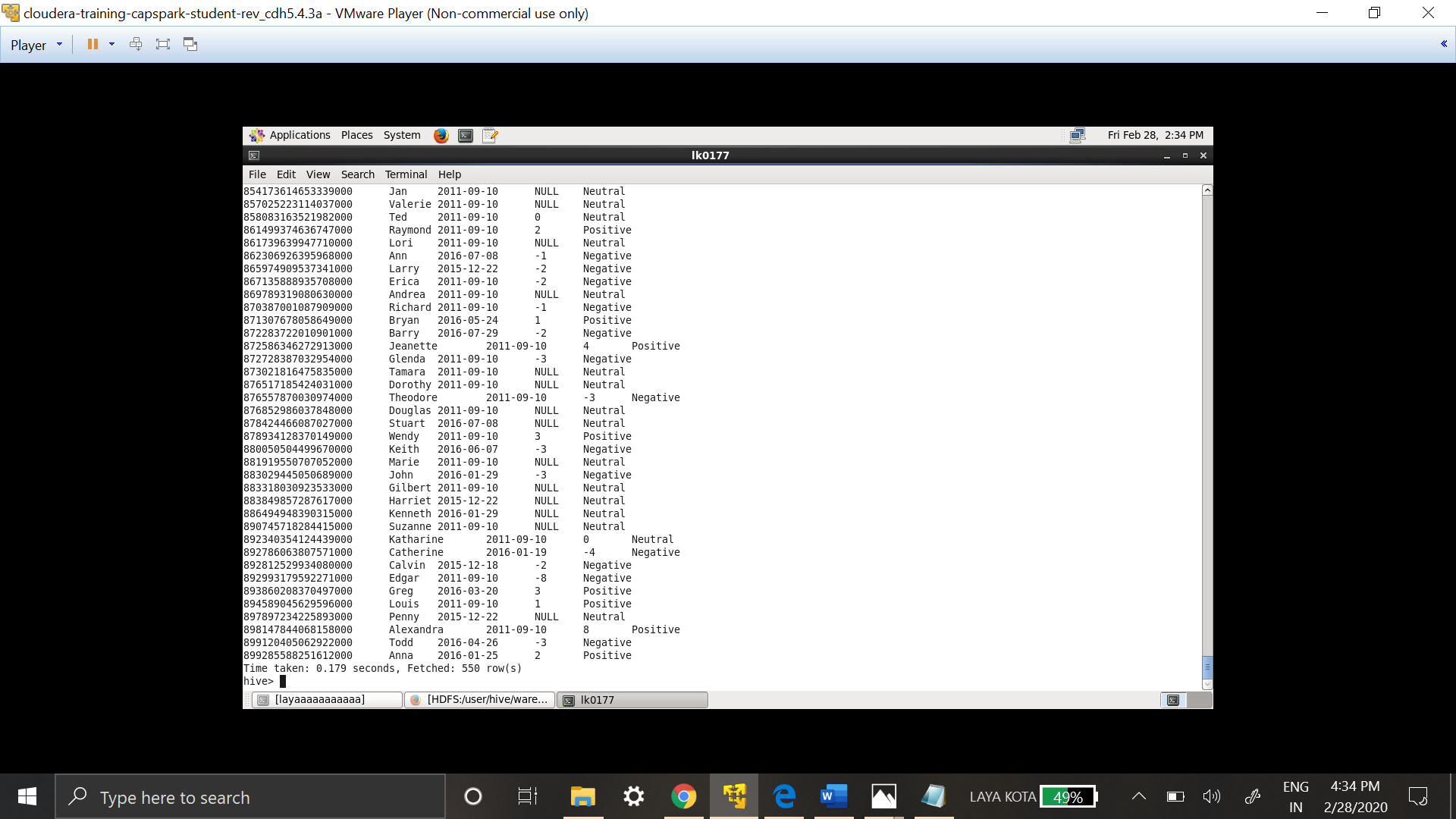
Sentiment analysis table is created which results as positive if the sum of rating is more than 0, negative if sum is less than 0 and neutral if sum is 0. The following query is used to validate table.

Select \* from sentiment\_analysis\_lk0177;









**3. 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.**

Many advance techniques are used these for sentiment analysis like Machine Learning with python using the Natural Language Toolkit (NLTK).

<https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/>

Many tools that analysis data as it comes on daily bases like monkey learn.

<https://monkeylearn.com/blog/sentiment-analysis-of-twitter/>