

**Twitter Sentiment Analysis in Hive**

**(**[**Big Data Analytics**](https://dicecamp.com/lms/lecture-projects?set_class=28-334-21)**)**

**Submitted by**

**Murtaza Jamali**

**Project Advisor**

Moeed Tariq

Muhammad Fayyaz

**December 30, 2023**

**Dice Analytic**

**ACKNOWLEDGEMENT**

First and foremost, I would like to express our gratitude to Almighty Allah for granting me the strength and conviction to advance our project to its current level of accomplishment. Secondly, this dissertation/monograph would not have materialized without the guidance of our instructor, Moeed Tariq. He has been remarkably supportive, and his feedback has significantly enhanced our work. His direct and active involvement in supervising our project is exemplary. His continuous support, endless provision, and consistent inspiration have become our driving force throughout this project.

I also extend our thanks to him for directing us to other individuals involved in the creation of this project. I am deeply appreciative of all those from whom I have gained substantial knowledge:

Our co-instructor, Muhammad Fayyaz, deserves special gratitude for illuminating the right path, providing encouragement during stressful times, having faith in me when confidence was lacking, and dedicating precious time to explain ideas and various aspects of the project. He has offered valuable suggestions, engaged in

**Introduction**

**Introduction:**

The field of Twitter Sentiment Analysis has gained significant prominence in recent years, offering a powerful means to comprehend public opinion and reactions on a global scale. This project delves into the abstraction of Twitter Sentiment Analysis, employing Apache Hive for data processing and analysis, and Power BI for visualization.

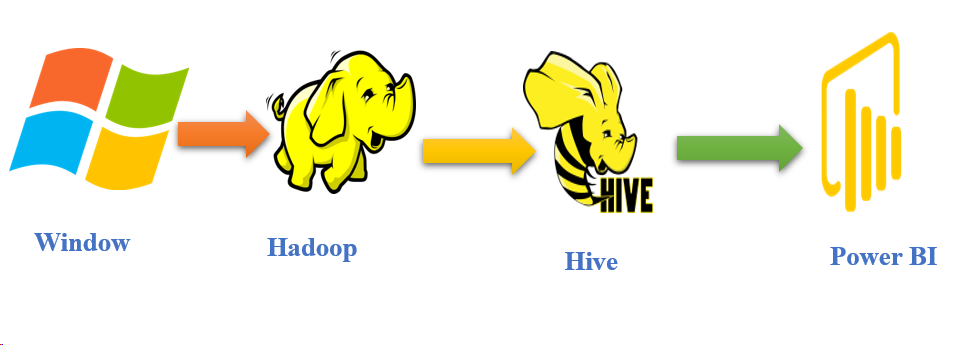
**Objective:**

The primary goal of this project is to develop an efficient and scalable solution for Twitter Sentiment Analysis. By utilizing Hive, a data warehousing and SQL-like query language system for Hadoop, in conjunction with Power BI, a robust visualization tool, we aim to provide a comprehensive understanding of sentiments expressed in tweets.

**Conclusion:**

In summary, this project seeks to offer a robust and scalable solution for Twitter Sentiment Analysis, leveraging the combined capabilities of Apache Hive and Power BI to provide a nuanced understanding of sentiments expressed on the dynamic platform of Twitter.

**Work Flow**

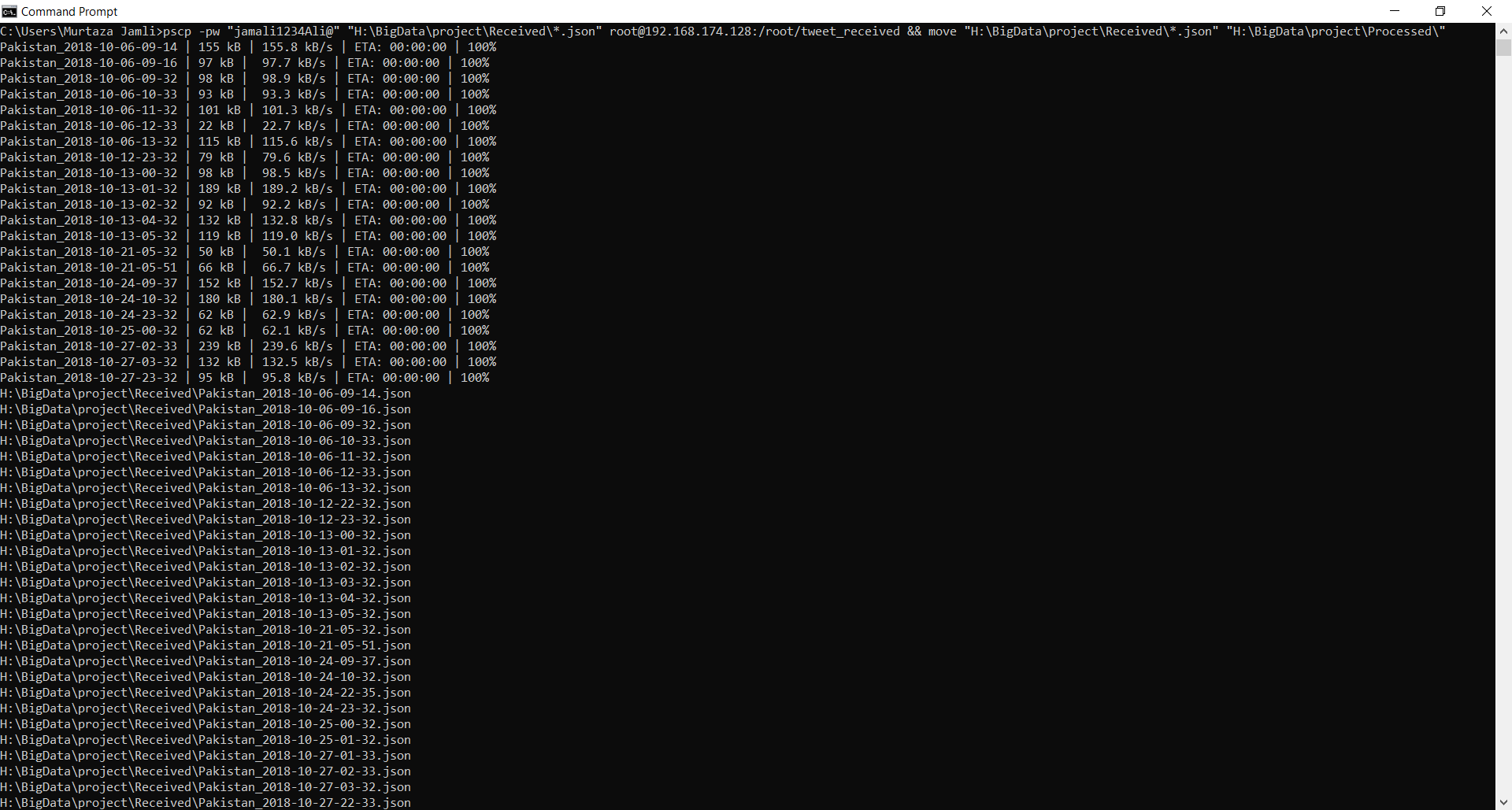
The workflow ensures a systematic approach to Twitter Sentiment Analysis, integrating the capabilities of Hive for data processing and Power BI for intuitive visualization. It enables users to gain valuable insights from the dynamic landscape of sentiments expressed on Twitter.

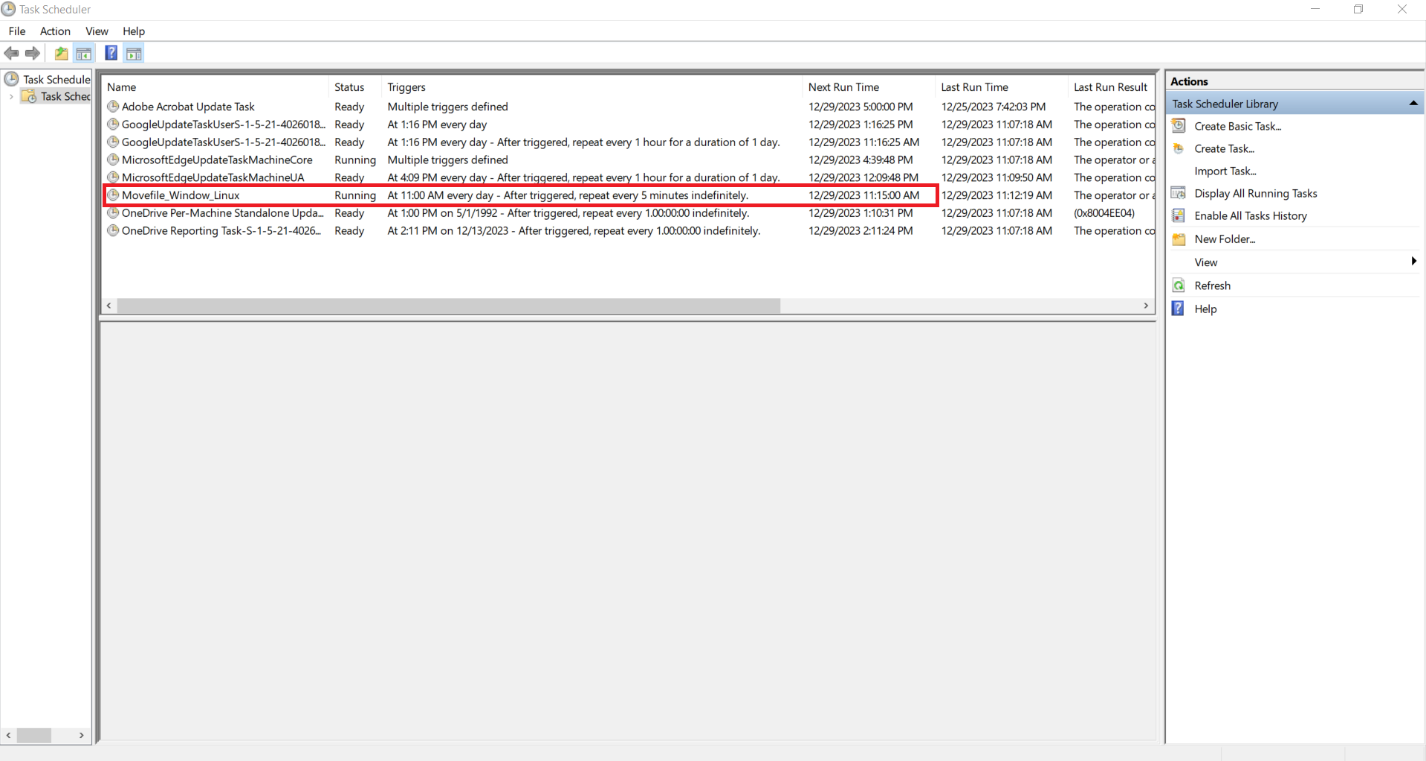
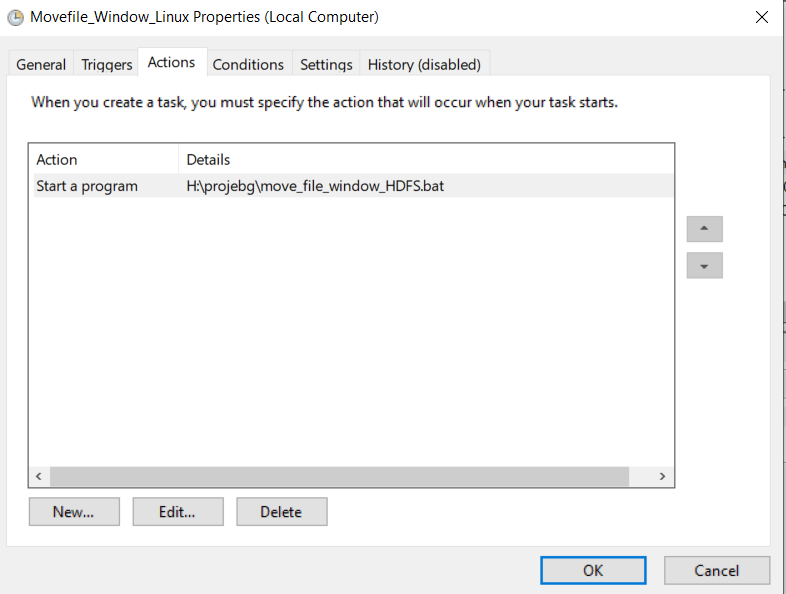
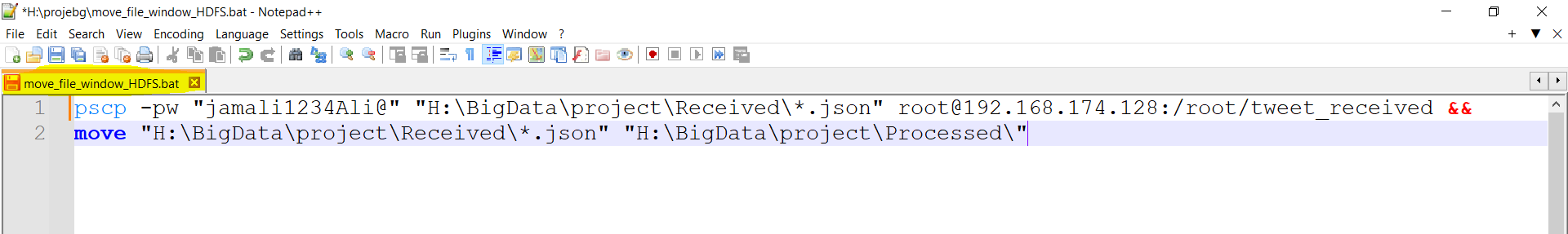
**Copy files window to NDFS and Move Files Window Processed folder for backup purpose**

This code snippet appears to be a command using the `pscp` utility for secure file transfer. It copies all JSON files from the local directory "H:\BigData\project\Received\" to a remote server with the IP address 192.168.174.128. The destination path on the server is "/root/tsweet\_received". Additionally, after the files are successfully transferred, the code uses the ‘move’ command to relocate the same set of JSON files from the local "Received" directory to the "Processed" directory under the same project path. The password for authentication is provided as "jamali1234Ali@" with the `-pw` option in the ‘pscp’ command.

**Code:**

pscp -pw "jamali1234Ali@" "H:\BigData\project\Received\\*. json" root@192.168.174.128:/root/tweet\_received && move "H:\BigData\project\Received\\*.json" "H:\BigData\project\Processed\"



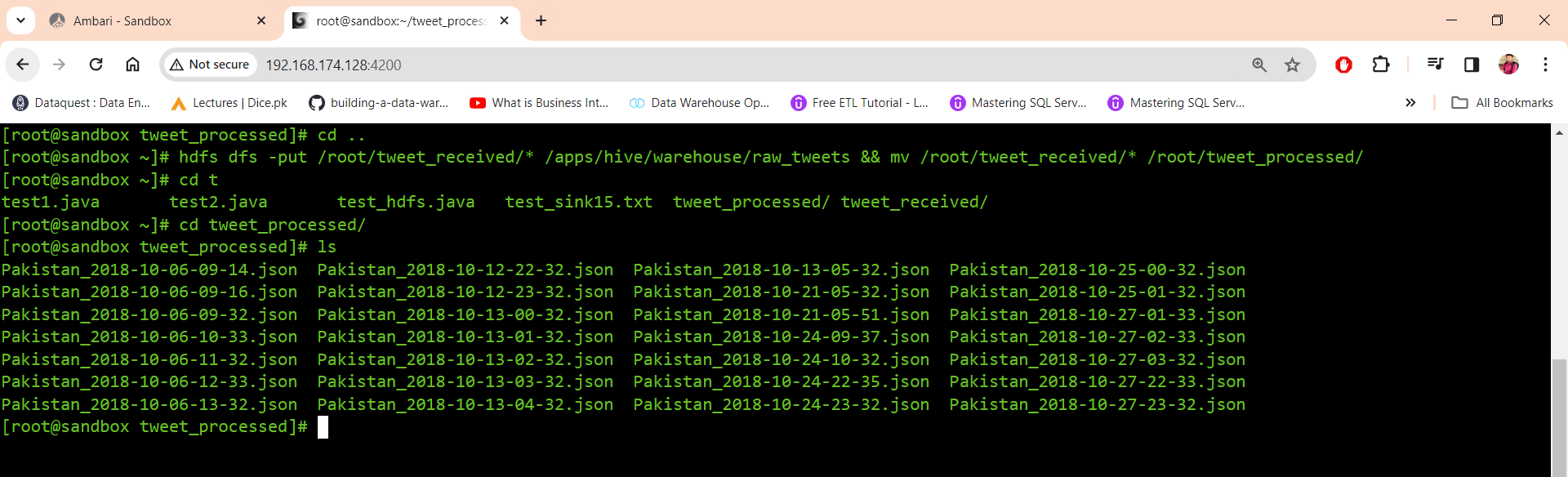
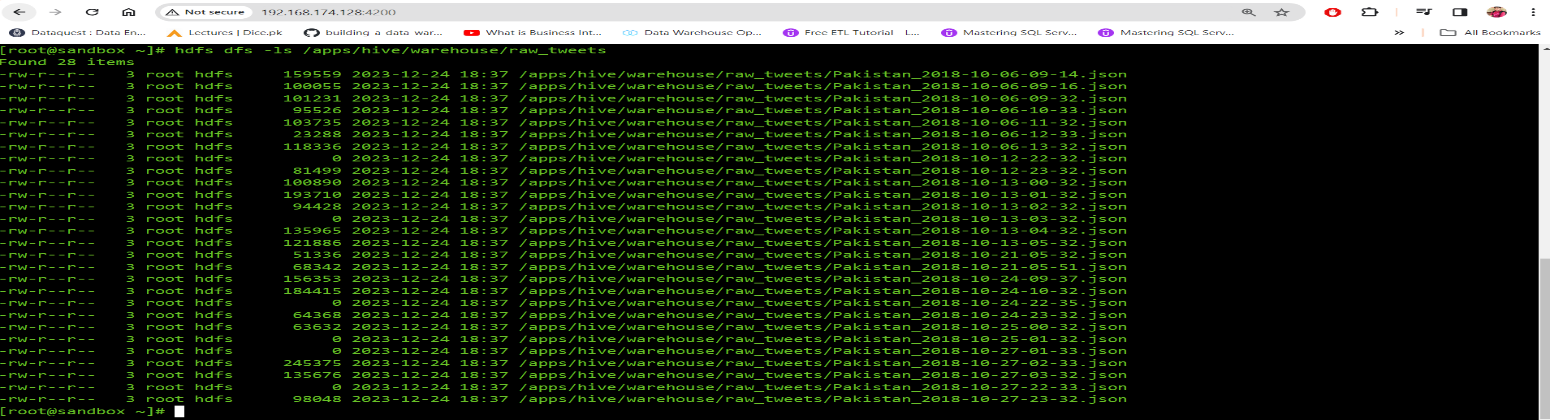
Automation window to NDFS

**Copy files NDFS to HDFS and Move Files NDFS tweet\_processed folder for backup purpose**

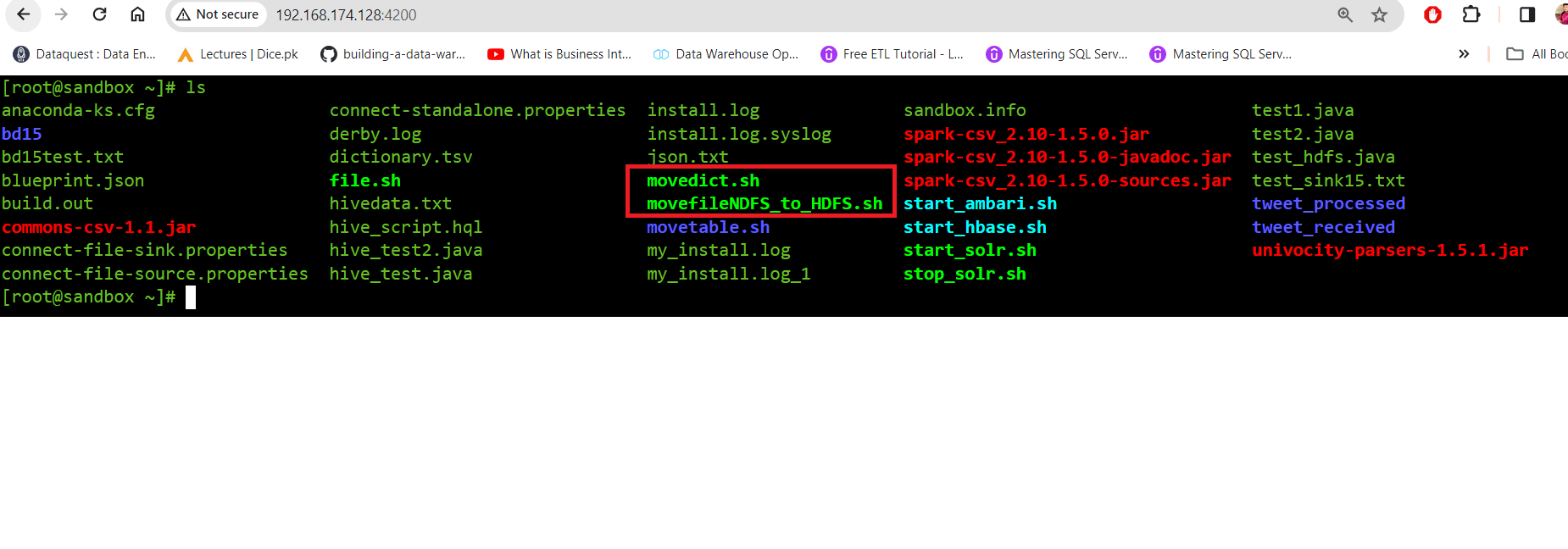
This sequence suggests a workflow where raw tweet data is initially uploaded to an HDFS location, likely for processing or storage, and then the local copies are moved to a different directory for further processing or archiving.

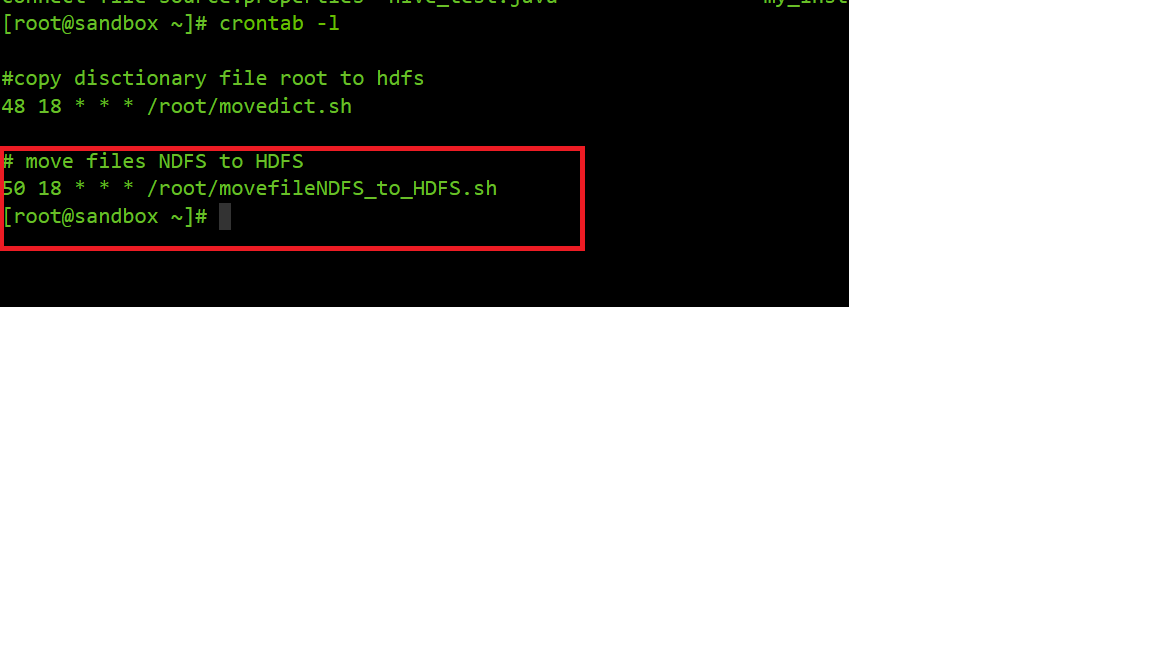
**Code:**

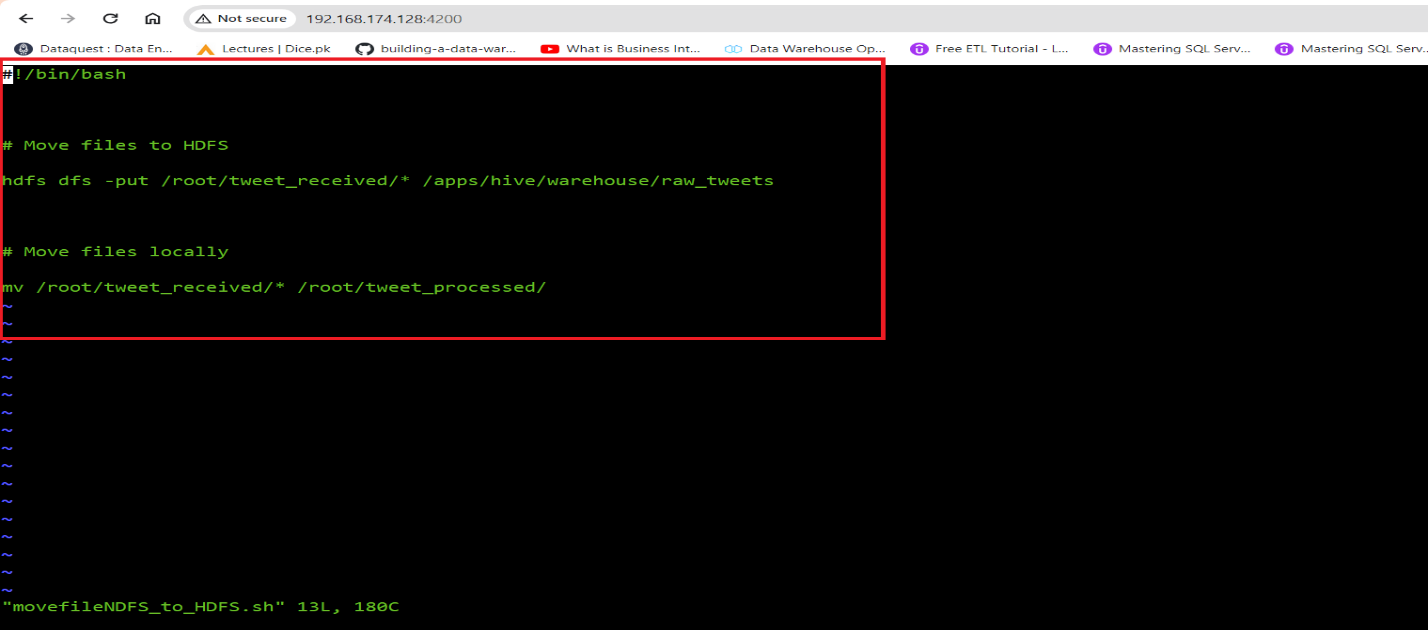
hdfs dfs -put /root/tsweet\_received/\* /apps/hive/warehouse/raw\_tweets && mv /root/tweet\_received/\* /root/tweet\_processed/



**Automation NDFS to HDFS**

****

****

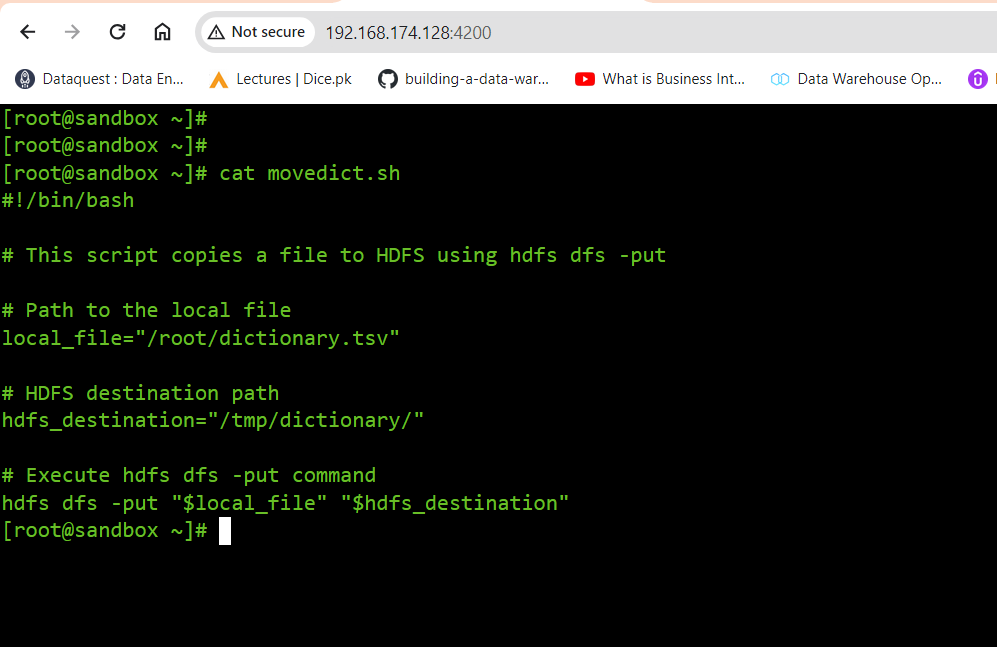
****

**Step:1 Transfer Dictionary window to NDFS:**

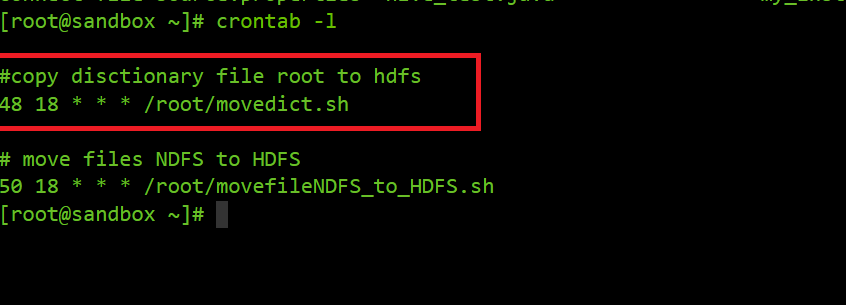
pscp -pw "jamali1234Ali@" "H:\projebg/dictionary.tsv" [root@192.168.174.128:/root/](mailto:root@192.168.174.128:/root/)

**Step:2 NDFS to HDFS**

hdfs dfs -put /root/dictionary.tsv /tmp/dictionary/

**Automation Dictionary move**

**Cron Job Dictionary**

****

**Create Table and Load Raw Tweets**

Now we can create the hive table to load tweets in it. In the query below I have only selected some columns from the .json file to show that we can just load the columns we require later in the process. Use the query or modify it to create the table where tweets will be loaded.

**Query:**

CREATE EXTERNAL TABLE IF NOT EXISTS raw\_tweets

(

created\_at string,

id string,

id\_str string,

text string,

source string,

truncated string,

user\_tw struct

<

id:string,

id\_str:string,

name:string,

screen\_name:string,

location:string,

url:string,

description:string,

translator\_type:string,

protected:string,

verified:string,

followers\_count:string,

friends\_count:string,

listed\_count:string,

favourites\_count:string,

created\_at:string,

utc\_offset:string,

time\_zone:string

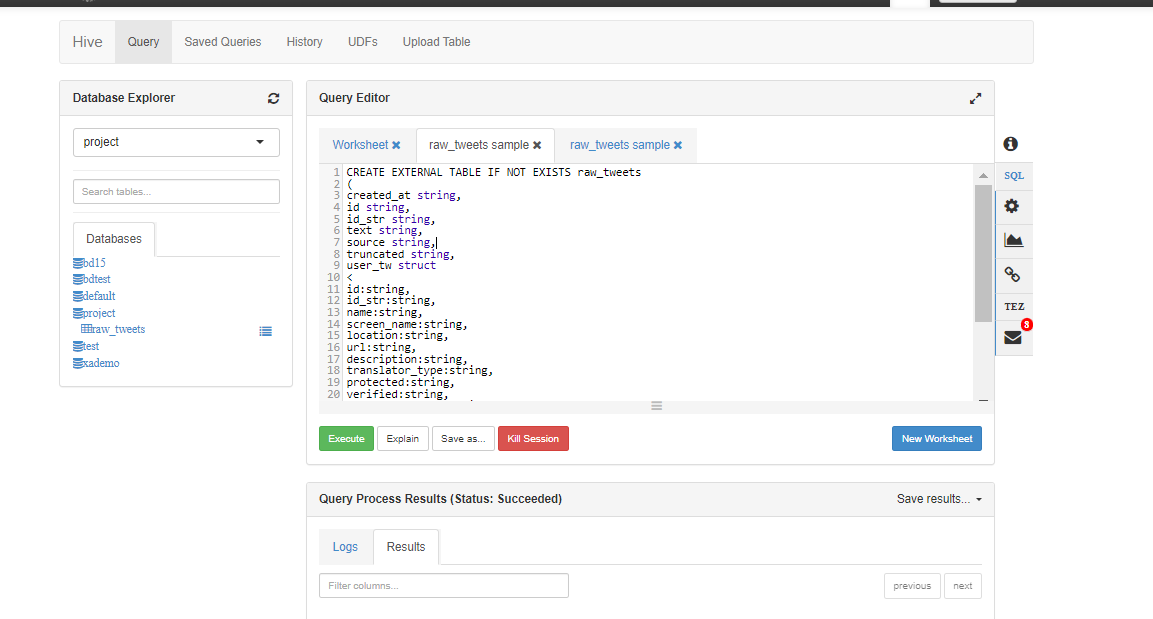
>

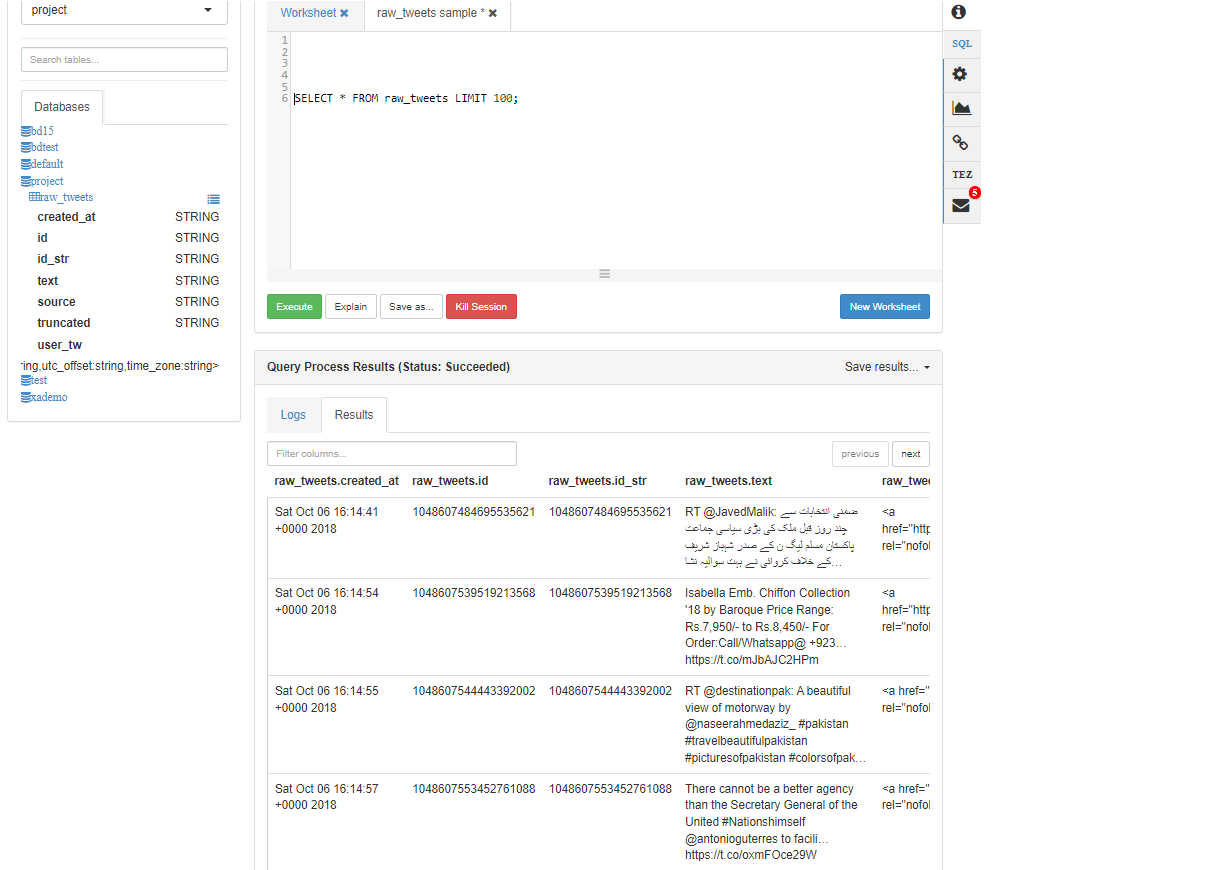
)

ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe'

STORED AS TEXTFILE

LOCATION '/apps/hive/warehouse/raw\_tweets/' ;





Create Table and Load Dictionary

Now we create table and Load Dictionary

CREATE EXTERNAL TABLE my\_dictionary (

type string,

length int,

word string,

pos string,

stemmed string,

polarity string

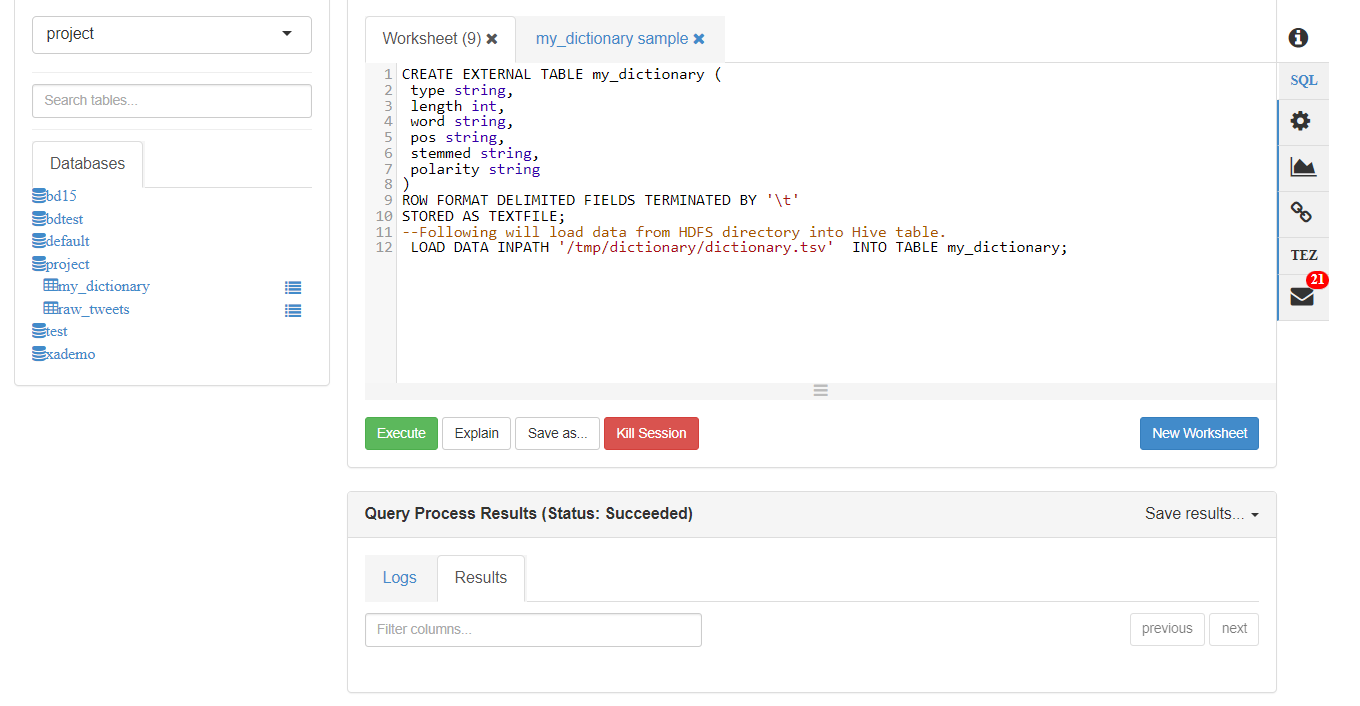
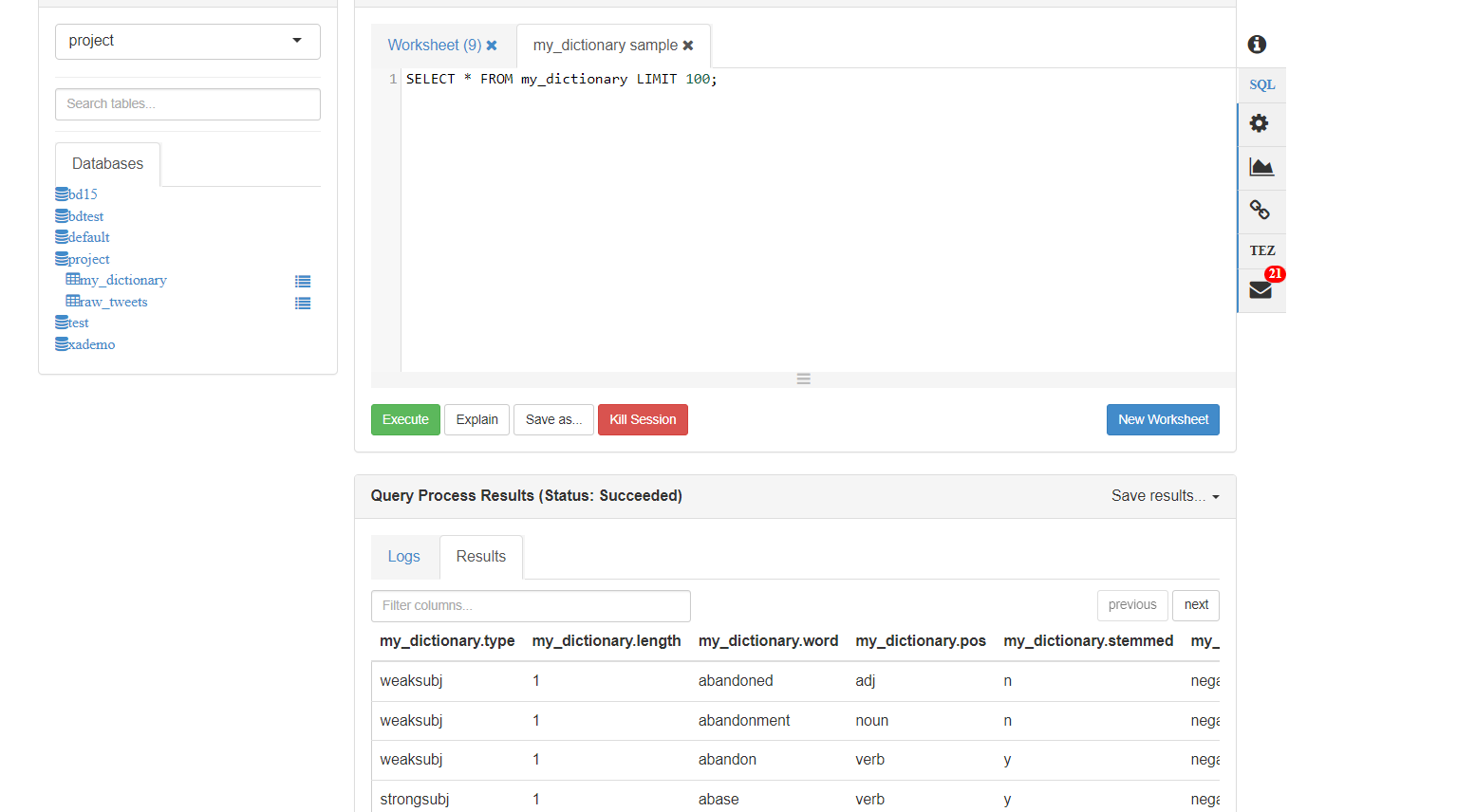
)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS TEXTFILE;

--Following will load data from HDFS directory into Hive table.

LOAD DATA INPATH '/tmp/dictionary/dictionary.tsv' INTO TABLE my\_dictionary;



**Create L1, L2 and L3 views:**

To analyze the sentiment of a tweet, we need to break it down into words so that we can match the sentiment of words from dictionary

table.

First, we will create Layer 1 view which will extract date elements, convert the text to lowercase and remove any new line characters.

Query:

CREATE OR REPLACE VIEW project.Layer1 AS

SELECT

created\_at,

SUBSTR(created\_at, 27, 4) AS years,

SUBSTR(created\_at, 5, 3) AS months,

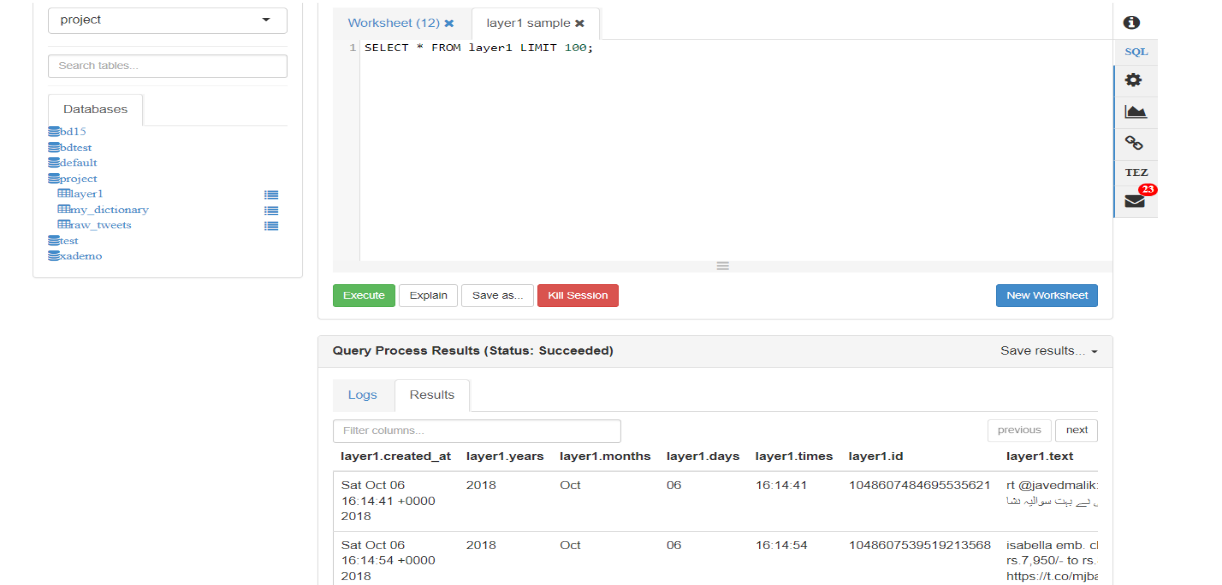
SUBSTR(created\_at, 9, 2) AS days,

SUBSTR(created\_at, 12, 8) AS times,

id,

LOWER(REGEXP\_REPLACE(text, '\n', '')) AS text

FROM raw\_tweets;



**Layer 2 view will explode the tweet by ID and separate each word into new line.**

Query:

CREATE OR REPLACE VIEW project.Layer2 AS

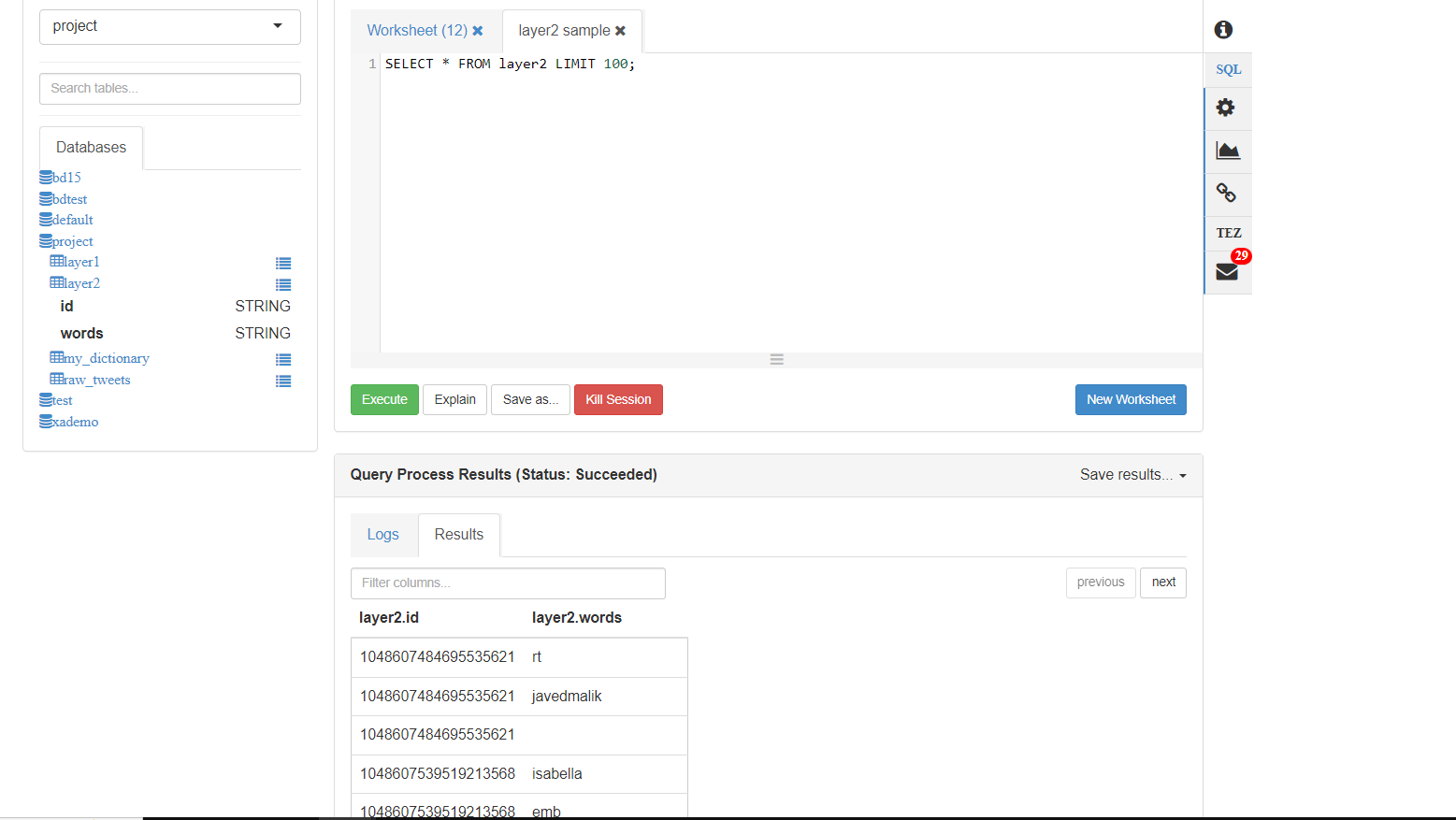
SELECT

id,

words

FROM

project.Layer1

LATERAL VIEW EXPLODE(SPLIT(text, '\\W+')) text AS words;

**Layer 3 view matches each word with the dictionary table picks whether polarity is negative or positive and assigns a value -1 or +1 respectively.**

**Query:**

**Create or Replace view project.Layer3 AS**

**select**

**id, L2.words,**

**case d.polarity**

**when 'negative' then -1**

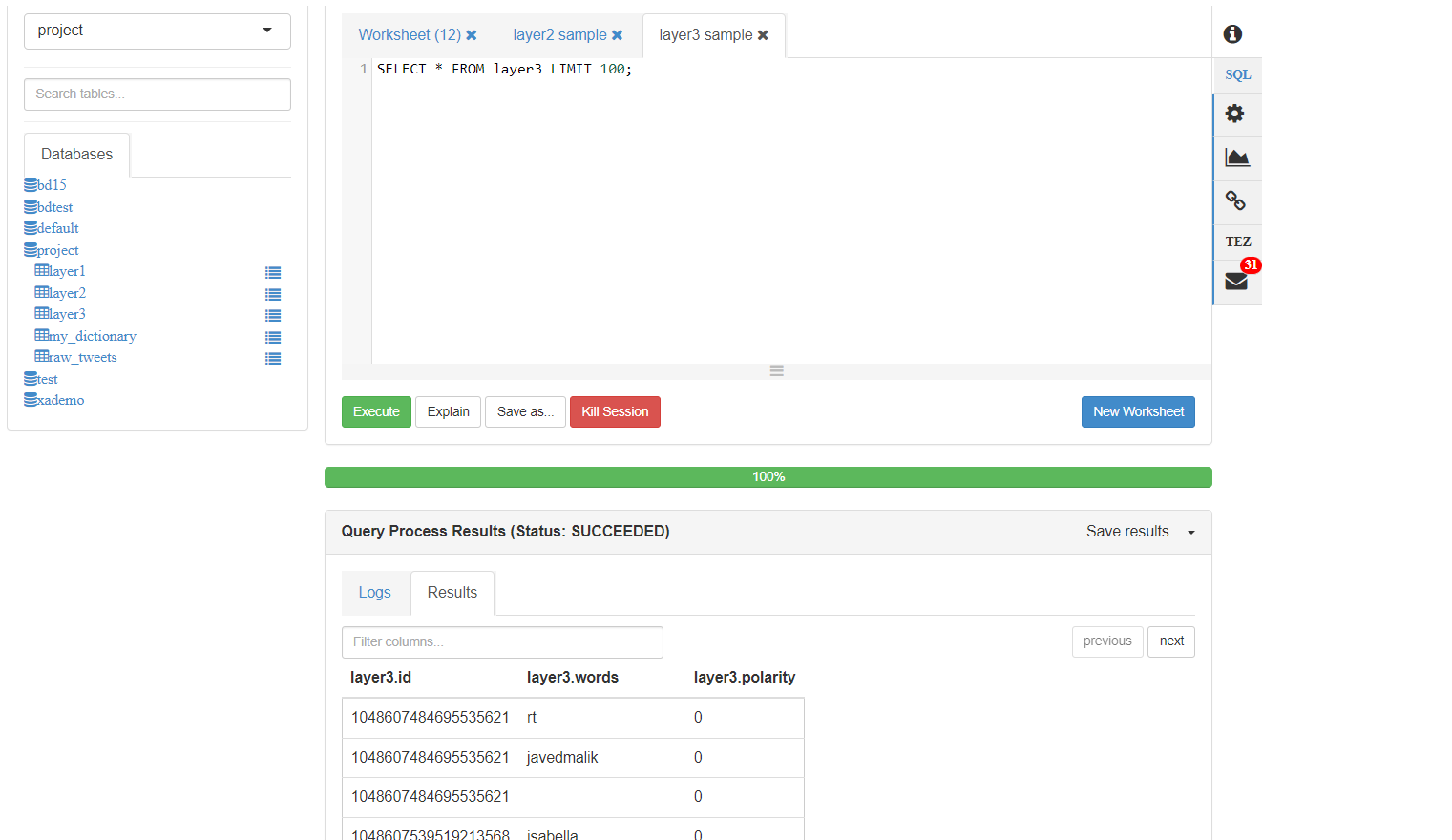
**when 'positive' then 1**

**else 0 end**

**as polarity**

**from Layer2 L2 left outer join my\_dictionary d**

**on L2.words=d.word;**



**Layer 4**

Finally, Layer 4 will sum up the polarity against a particular ID which generates the collective sentiment of the tweet with that ID. If you

are comfortable with writing complex queries, you can also combine Layer 3 and Layer 4 views and just create one view.

Query:

create or replace view project.sentiment as

select

id,

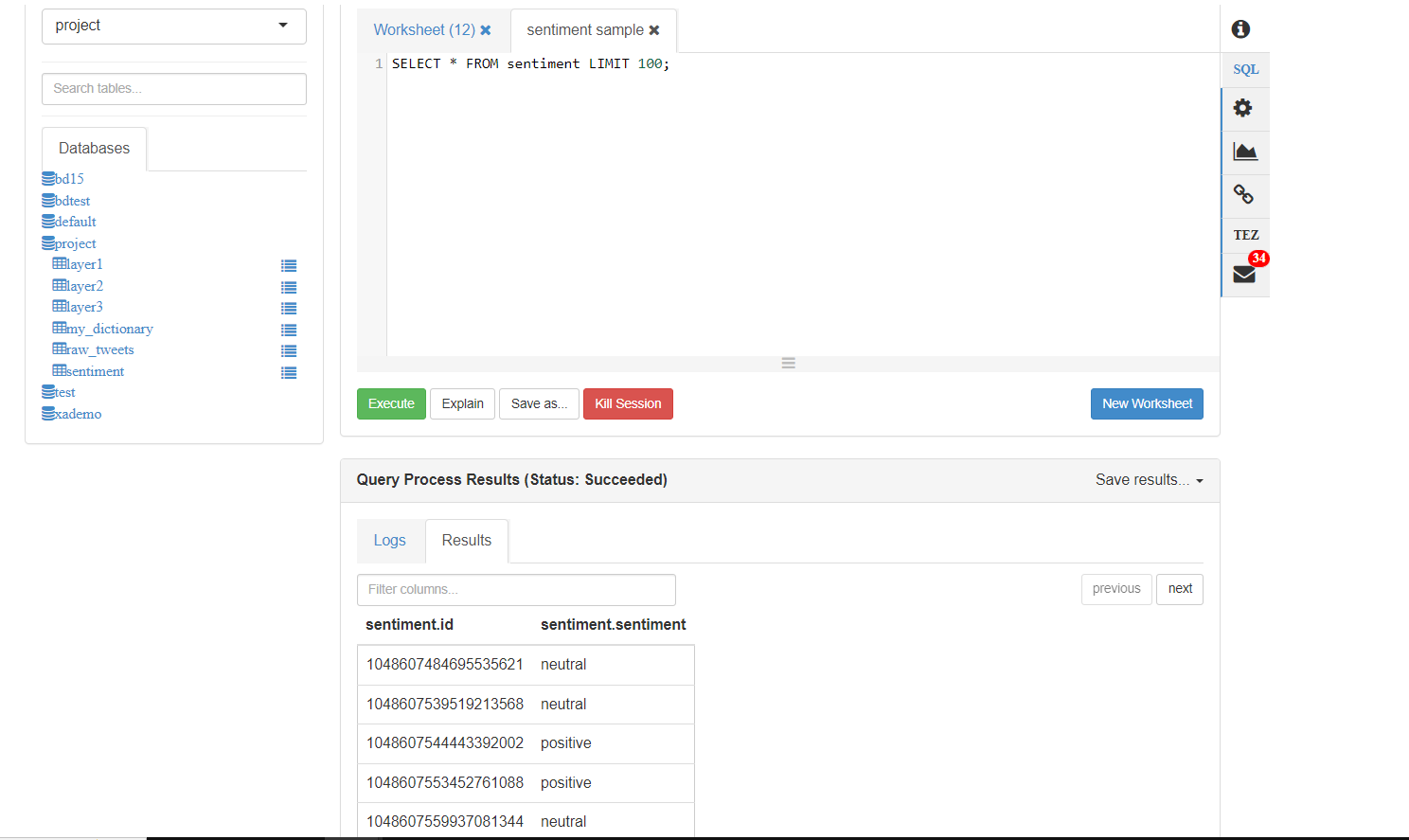
case

when sum( polarity ) > 0 then 'positive'

when sum( polarity ) < 0 then 'negative'

else 'neutral' end as sentiment

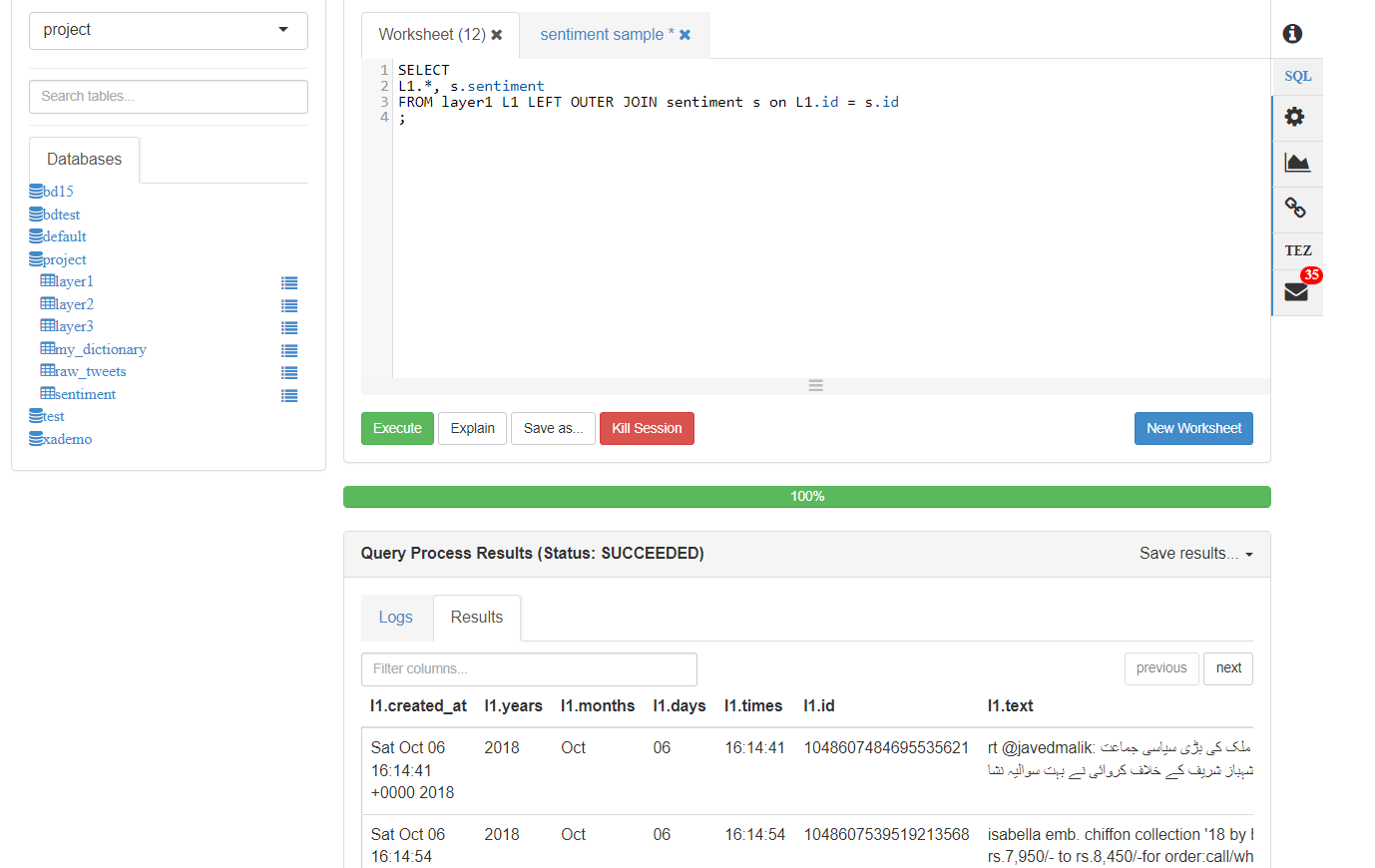
from layer3 l3 group by id;

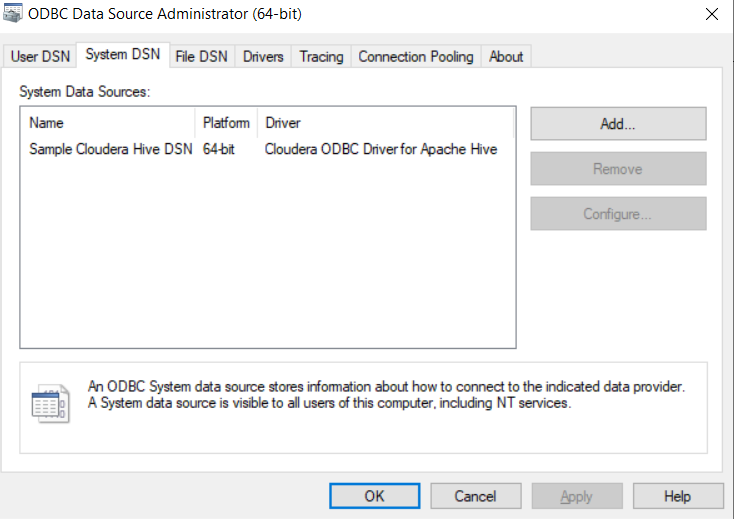
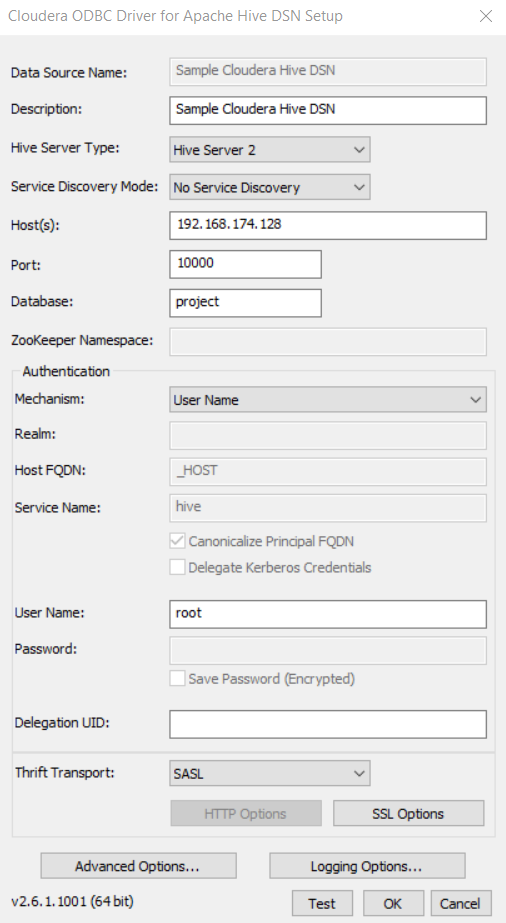


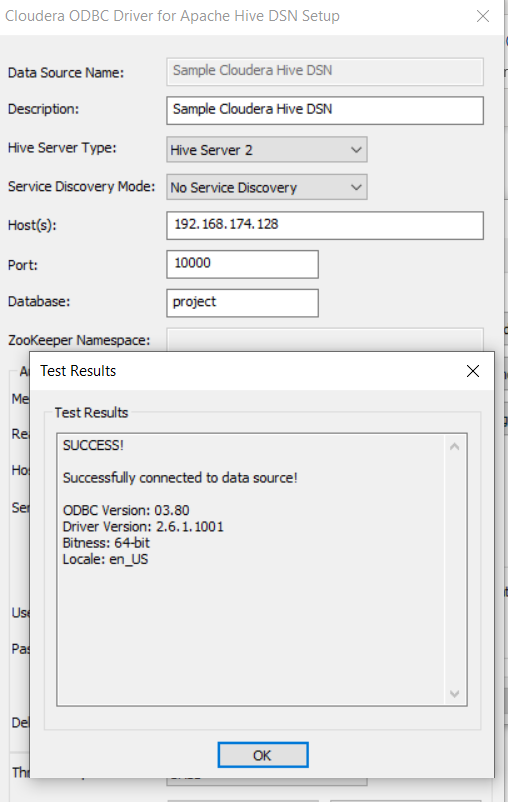
Now, We have the sentiment of each tweet in *project.sentiment* table/view grouped by each ID. If you wish to combine it with the Raw

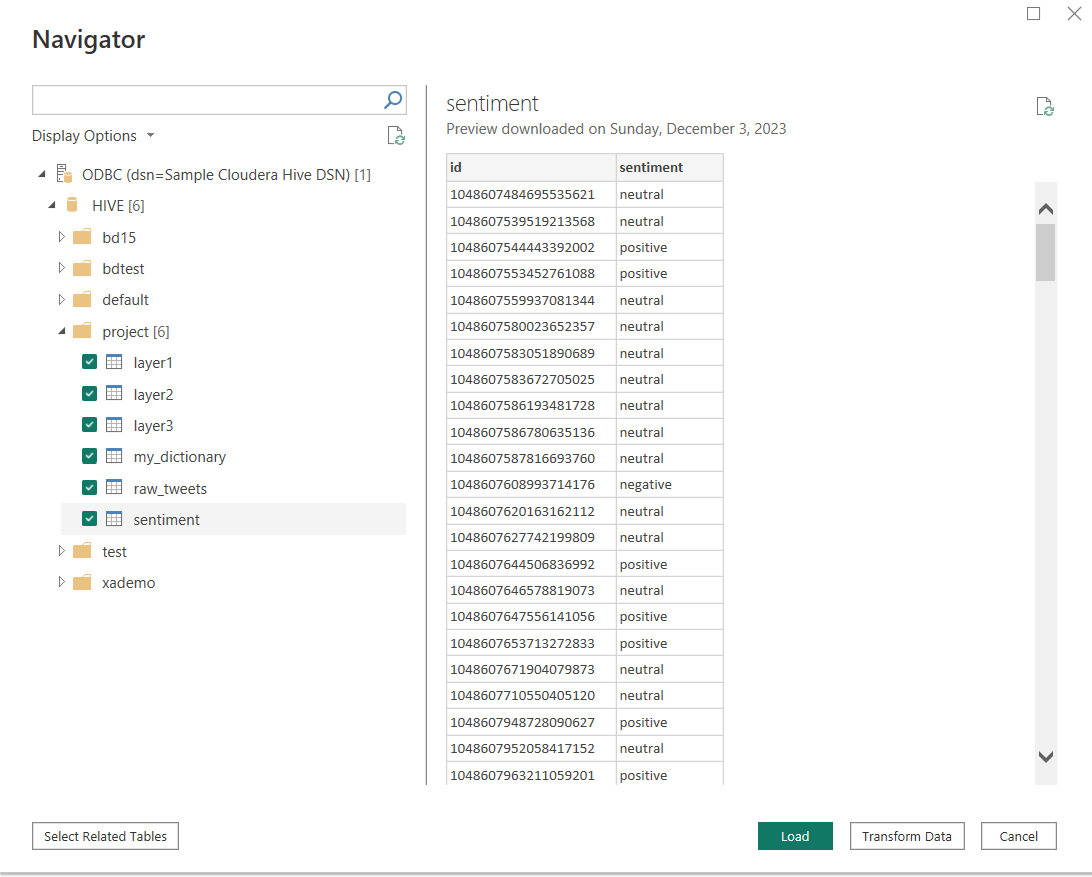
table or Layer 1 view/table to find the actual content of tweet. You can do so using the following query or create a final table using a

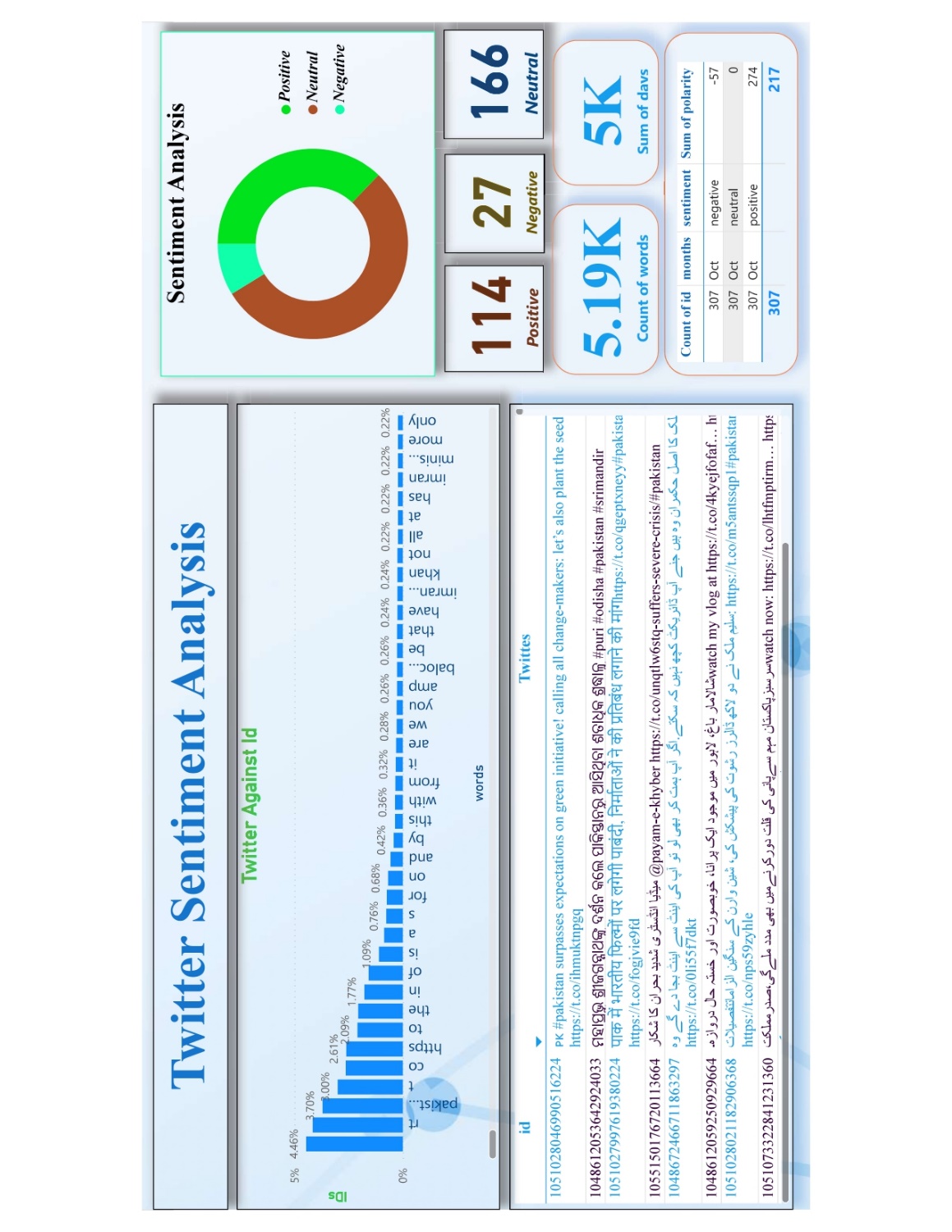
CTAS statement.



Connect Power BI with Hive:







Power BI Data Visualization

Please find All files and Assignments from GitHub:

<https://github.com/murtaza221/Twitter-Sentiment-Analysis-in-Hive-Big-Data-Analytics->