**Mid Term Report**

**Author: Manisha Ravindra Rao**

**Overview:**

An ETL setup has been established where the data from news-API has been imported for the keyword jobs and loaded into a CSV file that’s loaded into the hive table via hdfs. The loaded data is further used to extract insights, most said words, day-wise distribution of news articles, and so on.

**System Setup:**

For the process, we need kafka setup, where it’s done via zookeeper and confluent packages that are imported via URLs. The process of setting up can be referred to as an in-class activity exercise of week 4 on eConestoga. (https://conestoga.desire2learn.com/d2l/le/content/1001792/Home)

**Zookeeper setup:**

The zookeeper can be started using the below command.

bin/zkServer.sh start

A screenshot of a computer screen

Description automatically generated

**Running Kafka:**

The kafka setup is running fine and can be observed from the following command.

bin/kafka-server-start etc/kafka/server.properties

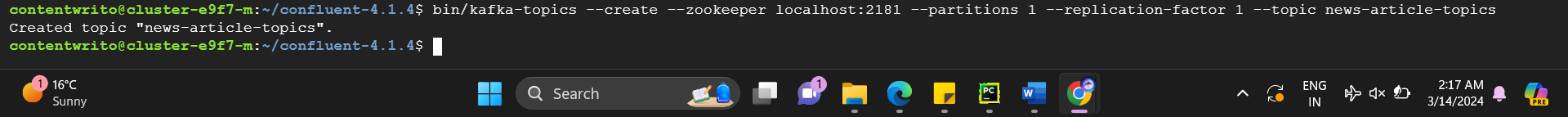
A screenshot of a computer

Description automatically generated

**Creating Kafka topic news-article-topics**

For the producer and consumer to communicate we will need a common topic where the data can be shared from producer to consumer via zookeeper.

bin/kafka-topics --create --zookeeper localhost:2181 --partitions 1 --replication-factor 1 --topic news-article-topics



**Running Producer:**

We want to run the kafka server so that the producer starts running and the data from the newsapi via api key is fetched for the sources like jobs in this case. The data with sources having jobs will be fetched every 60 seconds and passed to the consumer, making sure the fresh data is available.

Note: The producer and consumer code are in a Python file and the code is briefly commented on explaining the steps, as well as recorded in the video. I have selected a few columns only from api for my table. Before running the code we must pip install the kafka module and newsapi modules.

The python codeA screenshot of a computer

Description automatically generated

The kakfa server has to be started.

nohup bin/kafka-server-start etc/kafka/server.properties > /dev/null 2>&1 &

python news\_producer\_code.py

A screenshot of a computer screen

Description automatically generated

**Consumers running**

The Python code for consumers.

A screenshot of a computer

Description automatically generated

We want to run the kafka server so that the consumers start running, take the data from the producer, and convert it into CSV file.

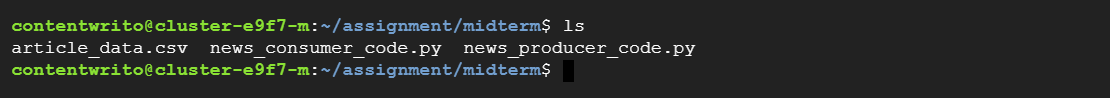
python news\_consumer\_code.py

A screenshot of a computer screen

Description automatically generated

**File output:**

article\_data.csv is created as the output of the file which is created by the consumer.



The sample output of the CSV is shown here.

A screenshot of a computer screen

Description automatically generated

**Hive Table:**

Table created in HIVE.

A sample table is created named news\_articles\_manisha, with the column names from the file.

CREATE EXTERNAL TABLE IF NOT EXISTS news\_articles\_manisha (

title STRING,

description STRING,

url STRING,

publishedAt STRING,

content STRING

)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE

LOCATION '/BigData/hive/midterm'

TBLPROPERTIES ('skip.header.line.count'='1');

A black screen with white text

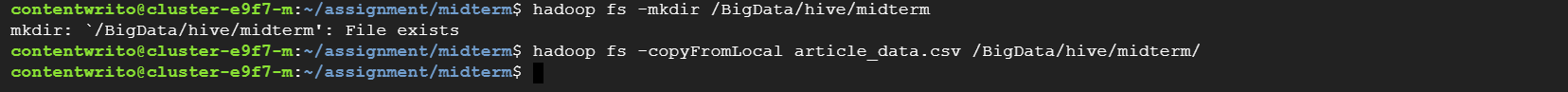
Description automatically generated

Moved the CSV file to a folder in Hadoop.

Created a folder midterm and copied the csv file from local to midterm folder.

hadoop fs -mkdir /BigData/hive/midterm/

hadoop fs -copyFromLocal article\_data.csv /BigData/hive/midterm/



Loaded the table from the file in the Hadoop folder

LOAD DATA INPATH '/BigData/hive/midterm/article\_data.csv'

INTO TABLE news\_articles\_manisha;

A screenshot of a computer

Description automatically generated

Displayed the data from the file.

SELECT \* FROM news\_articles\_manisha limit 5;

A screenshot of a computer screen

Description automatically generated

The data in Hive table can be used to derive multiple insights for data analysis some of my insights are as:

**Insights:**

1. **Most Common Words in Titles:**

The most common words from the title are extracted where we can observe that the most common topics were Biden and the government ignoring the articles. If we have additionally added more detailed select statements with sentiment analysis we can get fewer articles and more relatable words in the future.

SELECT word, COUNT(\*) AS word\_count

FROM (

SELECT EXPLODE(SPLIT(title, ' ')) AS word FROM news\_articles\_manisha

) words

WHERE word != ''

GROUP BY word

ORDER BY word\_count DESC

LIMIT 10;

A screenshot of a computer

Description automatically generated

1. **Articles Published by Year:**

The number of articles published per year is seen, to analyze how the number of articles are growing over the years.

SELECT YEAR(FROM\_UNIXTIME(UNIX\_TIMESTAMP(SUBSTR(publishedAt, 1, 8), 'yyyyMMdd'))) AS year, COUNT(\*) AS article\_count

FROM news\_articles\_manisha

GROUP BY YEAR(FROM\_UNIXTIME(UNIX\_TIMESTAMP(SUBSTR(publishedAt, 1, 8), 'yyyyMMdd')))

ORDER BY year;

A screenshot of a computer

Description automatically generated

1. **Day-wise Distribution of Published Articles:**

The articles per day are counted to see how the trends are over time. This helps us analyze how many articles could be expected over a certain time of the year in the long term making way for the traffic to websites.

SELECT DAY(FROM\_UNIXTIME(UNIX\_TIMESTAMP(SUBSTR(publishedAt, 1, 8), 'yyyyMMdd'))) AS day, COUNT(\*) AS article\_count

FROM news\_articles\_manisha

GROUP BY DAY(FROM\_UNIXTIME(UNIX\_TIMESTAMP(SUBSTR(publishedAt, 1, 8), 'yyyyMMdd')))

ORDER BY day;

A screenshot of a computer

Description automatically generated