Fatma Nabil Ebrahim

Data Management track

Apache spark report

Introduction

The purpose of this project is to analyze Twitter data to gain insights into user behavior and preferences. Specifically, we want to identify the most frequently mentioned topic or event related to the word "Eid" on Twitter. To accomplish this, I used Apache Spark, a fast and flexible big data processing framework that allowed me to handle a large-scale datasets.

Data Ingestion

The data for this project was obtained from the Twitter API, which provides a real-time stream of tweets from users around the world. We used the `tweepy` library in Python to access the API and retrieve tweets containing the word "Eid" from a specified date range. The data was initially received in JSON format and then ingested into Spark as Parquet files. We stored the data partitioned by year, month, day, and hour in a directory called "twitter-landing-data" I called it “FileStore”.

Data Cleaning and Preparation

Before analyzing the data, we needed to clean and prepare it for analysis. This involved removing any duplicate or null values, standardizing data formats, and aggregating the data. We also needed to filter out any tweets that did not contain the word "Eid" in their text or metadata, # Replace emojis with an empty string, Remove user mentions from the text column, Remove any "\n" from the text column, Define a regular expression pattern to match URLs and Define a UDF to remove URLs from text **This** all can be seen in the **code** .

Data Analysis

To identify the most frequently mentioned topic or event related to "Eid" on Twitter, we ran several queries on the data. First, we counted the number of occurrences of each unique word in the tweets using the Spark Data Frame API. Then, we filtered the results to only include words that were commonly associated with "Eid", such as "celebration", "prayer", and "family". Finally, we aggregated the results by day and hour to identify any spikes or trends in activity related to "EID.".

**** and now lets dig in the scripts and how they work**

Script Name: Twitter API Data Collection

Description:

The Twitter API Data Collection script is a Python program designed to collect tweets related to a given keyword, transform the data into a structured format using Spark, and store it in a streaming manner. This report provides an overview of the script's functionality, dependencies, functions, and main code.

Dependencies:

- requests
- pandas
- csv
- datetime
- pyspark
- dateutil.parser
- unicodedata
- time
- socket

Functions:

The Twitter API Data Collection script consists of three main steps:

Collect Data: Calls the auth() function to get the bearer token required for authorization to use the Twitter API, calls the create_headers() and create_url() functions to create the headers and URL required for data collection, uses the requests library to connect to the Twitter API endpoint and collect the data, and prints the response status code.

Transform Data: Defines the schema for the structured data using the StructType object, reads the streaming data from the socket and converts it into a structured format using the schema defined above, extracts the required columns and creates additional columns for date, year, month, day, and hour, and prints the transformed data.

Store Data: Stores the transformed data in a streaming manner.

As:

auth(): Returns the bearer token required for authorization to use the Twitter API.

create_headers(bearer_token): Creates and returns the headers object for authorization to use the Twitter API.

create_url(keyword, start_date, end_date, max_results = 10): Creates and returns the URL and query parameters required for data collection using the Twitter API.

connect_to_endpoint(url, headers, params, next_token = None): Connects to the Twitter API endpoint and returns the response in JSON format.

Main Code:

Step 1: Collect Data

- Calls the auth() function to get the bearer token required for authorization to use the Twitter API.

- Calls the create_headers() and create_url() functions to create the headers and URL required for data collection using the Twitter API.

- Uses the requests library to connect to the Twitter API endpoint and collect the data.

- Prints the response status code.

Step 2: Transform Data

- Defines the schema for the structured data using the StructType object.

- Reads the streaming data from the socket and converts it into a structured format using the schema defined above.

- Extracts the required columns and creates additional columns for date, year, month, day, and hour.

- Prints the transformed data.

Step 3: Store Data

- Stores the transformed data in a streaming manner.

```
# clientsocket.close()
```

```
Listening on port: 7777
```

Script Name: Twitter structure

The code is written in Python using PySpark library to read data from a socket, clean the data, and write the data to HDFS in parquet format.

First, a SparkSession object is created. Then, a schema is defined for the JSON data that will be read from the socket. The data is read using the socket format and then converted to a DataFrame. The from_json function is used to convert the data from JSON format to structured format using the schema. Next, the data is selected and transformed using selectExpr function to extract required columns and create new columns for date-time values.

After that, the text column is cleaned by replacing emojis with an empty string, removing user mentions, removing any "\n" characters, and removing URLs using a regular expression pattern and a user-defined function. The DataFrame is then cleaned up and selected to only keep the required columns.

Finally, the data is written to HDFS in parquet format using the writeStream function. The partitionBy option is used to partition the data by year, month, day, and hour. The start function is called to start the stream.

```
firstRows_DF=spark.sql("select * from twitter_data.tweets limit 20 ")
firstRows_DF.show()
```

```
+-------------------+--------------------+-------------------+-------------+-----------+----------+-----------+-------------------+------------
--------+---------------+---------------+---------------+-----------+-----------+--------+----+-----+---+----+
|                 id|                text|          author_id|retweet_count|reply_count|like_count|quote_count|         created_at|
name|       username|followers_count|following_count|tweet_count|listed_count|verified|year|month|day|hour|
+-------------------+--------------------+-------------------+-------------+-----------+----------+-----------+-------------------+------------
--------+---------------+---------------+---------------+-----------+-----------+--------+----+-----+---+----+
|1654239177595146240|RT : Omdurman Mid...| 711486983655301121|           26|          0|         0|          0|2023-05-04 21:38:48|
Hisham|    hishamjr145|            335|            376|      10988|          2|   false|2023|    5|  4|  21|
|1654239165159034880|RT : Those in Sud...|        3134827066|          162|          0|         0|          0|2023-05-04 21:38:45|    Jonathan
Abuajah|       jabuajah|            517|            422|     105996|          0|   false|2023|    5|  4|  21|
|1654239163560996865|The US just broug...|1283731444830023680|            0|          0|         0|          0|2023-05-04 21:38:45|  Fidel Slim
Guevara|    Slimchineme|            843|           1046|      16971|          0|   false|2023|    5|  4|  21|
|1654239162214797312|RT : After she fl...| 738952995136667649|           45|          0|         0|          0|2023-05-04 21:38:45| Freedom for
Tigray!|  HGebreslassie|           1623|           1731|      68648|          1|   false|2023|    5|  4|  21|
|1654239160624992256|RT : My only fear...|           61762023|          364|          0|         0|          0|2023-05-04 21:38:44|           O
luwaseun|     deolujames|           1950|           1058|     404521|         51|   false|2023|    5|  4|  21|
|1654239156891951108|RT : After she fl...|1496586301486809094|           55|          0|         0|          0|2023-05-04 21:38:43|
🆒Samri|Samrawitkalayu7|           3968|           1492|     646624|          2|   false|2023|    5|  4|  21|
|1654239147459059714|RT : According to...|1552714383444381696|           29|          0|         0|          0|2023-05-04 21:38:41|         Rak
i Hope🍋|   TrRt37484495|           1850|            934|     534511|          1|   false|2023|    5|  4|  21|
|1654239142824546305|RT : I am surpris...|1547945567245832194|           46|          0|         0|          0|2023-05-04 21:38:40|      Abubak
ar shehu|Abubaka66532343|             51|             72|        969|          0|   false|2023|    5|  4|  21|
|1654239134410502148|RT : We are alarm...|1501279581311021057|            7|          0|         0|          0|2023-05-04 21:38:38|      Tsgie
📢❤️💛💊| Tembieneyti14|           1849|           1009|     424754|          0|   false|2023|    5|  4|  21|
|1654239118975553542|RT : Current conf...|          217449520|          114|          0|         0|          0|2023-05-04 21:38:34|         Mar
k Baxter|   MarkRsbackie|             93|            184|        763|          0|   false|2023|    5|  4|  21|
|1654239107072028672|RT : Allen Onyeam...|          610698950|            8|          0|         0|          0|2023-05-04 21:38:32|General Omor
ogbe ...|  OmorogbeHarry|            892|           1550|     103560|          0|   false|2023|    5|  4|  21|
|1654239102244384770|RT : Sudan Ambass...|1217979669988900864|            9|          0|         0|          0|2023-05-04 21:38:30|          Ms
addamabk|     msaddamabk|            555|           4456|       4449|          4|   false|2023|    5|  4|  21|
|1654239094342287361|RT : Around 30 bu...|         3165729360|           11|          0|         0|          0|2023-05-04 21:38:29|       Georg
```

```
import pandas as pd

# Convert Spark DataFrame to Pandas DataFrame
pandas_df = result_df.toPandas()

# Do something with Pandas DataFrame
pandas_df.head()
```

	id	text	author_id	retweet_count	reply_count	like_count	quote_count	created_at	name	username	followers_count	fo
0	1654239177595146240	RT : Omdurman Midwives hospital, the largest m...	711486983655301121	26	0	0	0	2023-05-04 21:38:48	Hisham	hishamjr145	335	
1	1654239165159034880	RT : Those in Sudan that were evacuated by an ...	3134827066	162	0	0	0	2023-05-04 21:38:45	Jonathan Abuajah	jabuajah	517	
2	1654239163560996865	The US just brought 'democracy' to Sudan. That...	1283731444830023680	0	0	0	0	2023-05-04 21:38:45	Fidel Slim Guevara	Slimchineme	843	
3	1654239162214797312	RT : After she fled her home, Ms Haile crossed...	738952995136667649	45	0	0	0	2023-05-04 21:38:45	Freedom for Tigray!	HGebreslassie	1623	
4	1654239160624992256	RT : My only fear in this Sudan crisis is the	61762023	364	0	0	0	2023-05-04 21:38:44	Oluwaseun	deolujames	1950	

```
import pandas as pd

# Convert Spark DataFrame to Pandas DataFrame
pandas_df = result_df.toPandas()

# Do something with Pandas DataFrame
pandas_df.head()
```

	reply_count	like_count	quote_count	created_at	name	username	followers_count	following_count	tweet_count	listed_count	verified	year	month	day	hour
	0	0	0	2023-05-04 21:38:48	Hisham	hishamjr145	335	376	10988	2	false	2023	5	4	21
	0	0	0	2023-05-04 21:38:45	Jonathan Abuajah	jabuajah	517	422	105994	0	false	2023	5	4	21
	0	0	0	2023-05-04 21:38:45	Fidel Slim Guevara	Slimchineme	843	1046	16970	0	false	2023	5	4	21
	0	0	0	2023-05-04 21:38:45	Freedom for Tigray!	HGebreslassie	1623	1731	68639	1	false	2023	5	4	21
	0	0	0	2023-05-04	Oluwaseun	deolujames	1950	1058	404521	51	false	2023	5	4	21

Script Name: tables creation

Creation of the Tweets Table

The `tweets` table was created in the `twitter_data` database to store information about Twitter data. The table is partitioned by year, month, day, and hour, and is stored in the Parquet file format.

Table Schema

The table schema for `tweets` is as follows:

Column Name : Description

Id :The unique identifier for the tweet

Text:The text content of the tweet

author_id :The unique identifier for the user who authored the tweet

retweet_count:The number of times the tweet has been retweeted

reply_count :The number of times the tweet has been replied to

like_count :The number of times the tweet has been liked

quote_count :The number of times the tweet has been quoted

created_at :The timestamp for when the tweet was created

name :The name of the user who authored the tweet

username :The username of the user who authored the tweet

followers_count :The number of followers the user who authored the tweet has

following_count :The number of accounts the user who authored the tweet is following

tweet_count :The number of tweets the user who authored the tweet has posted

listed_count :The number of Twitter lists the use who authored the tweet has been added to

verified :Indicates whether the user who authored the tweet has been verified by Twitter

Table Partitioning

The `tweets` table is partitioned by year, month, day, and hour. This allows for more efficient querying of the data based on time. The `year`, `month`, `day`, and `hour` columns are used to define the partitioning scheme.

Storage Format

The `tweets` table is stored in the Parquet file format. Parquet is a columnar storage format that is designed to be efficient for analytical processing of large datasets.

Location

The `tweets` table is located in the `/FileStore/output9` directory. This directory is used to store the Twitter data that has been partitioned by year, month, day, and hour.

Hive Table

A Hive table has been created on top of the `tweets` directory to facilitate querying the data.

By creating a Hive table on top of the `tweets` directory, users can easily query the data using SQL-like syntax. The partitioning of the data by time also makes it easier to filter the data based on specific time periods.

```
df.printSchema()
```

```
root
 |-- id: string (nullable = true)
 |-- text: string (nullable = true)
 |-- author_id: string (nullable = true)
 |-- retweet_count: integer (nullable = true)
 |-- reply_count: integer (nullable = true)
 |-- like_count: integer (nullable = true)
 |-- quote_count: integer (nullable = true)
 |-- created_at: timestamp (nullable = true)
 |-- name: string (nullable = true)
 |-- username: string (nullable = true)
 |-- followers_count: integer (nullable = true)
 |-- following_count: integer (nullable = true)
 |-- tweet_count: integer (nullable = true)
 |-- listed_count: integer (nullable = true)
 |-- verified: string (nullable = true)
 |-- year: integer (nullable = true)
 |-- month: integer (nullable = true)
 |-- day: integer (nullable = true)
 |-- hour: integer (nullable = true)
```

****This script is for creating several dimension and fact tables to store data from the Twitter API. Here's a breakdown of each table and its purpose:**

Dimensions

1. users_dim: This table contains information about Twitter users such as their ID, name, username, followers count, following count, tweet count, listed count, and verified status. It's partitioned by year, month, and day, and is populated by extracting data from the `twitter_data.tweets` table.

2. tweets_dim: This table contains information about tweets, including the tweet ID, text, author ID, retweet count, reply count, like count, quote count, and creation timestamp. It's partitioned by year, month, day, and hour, and is populated by extracting data from the `twitter_data.tweets` table.

3. time_dim: This table contains a time dimension with a unique `time_id` and a timestamp. It's partitioned by year, month, day, and hour, and is populated by generating a `ROW_NUMBER` over the `created_at` timestamp from the `twitter_data.tweets` table.

4. author_matrix: This table is a mapping table between authors and tweets, where each row represents a tweet ID authored by a particular author ID. It's partitioned by year, month, day, and hour, and is populated by extracting the author ID and tweet ID from the `twitter_data.tweets_dim` table.

```
spark.sql("""
select * from twitter_data.tweets_dim
limit 3
""")
```

id	text	author_id	retweet_count	reply_count	like_count	quote_count	created_at	year	month	day	hour
1654239162386591745	RT : Tigryan refu...	1496586301486809094	41	0	0	0	2023-05-04 21:38:45	2023	5	4	21
1654239161543454726	RT : After she fl...	1573068165524996096	24	0	0	0	2023-05-04 21:38:45	2023	5	4	21
1654239159106584578	RT : After she fl...	1492197849760677897	46	0	0	0	2023-05-04 21:38:44	2023	5	4	21

```
spark.sql("""
select * from twitter_data.users_dim
limit 3
""")
```

author_id	id	name	username	followers_count	following_count	tweet_count	listed_count	verified	year	month	day
1496586301486809094	1654239162386591745	Samri	Samrawitkalayu7	3968	1492	646616	2	false	2023	5	4
1573068165524996096	1654239161543454726	Tsige Geberhiwot	TsigeG21	1055	1130	69471	0	false	2023	5	4
1492197849760677897	1654239159106584578	Aluma	AlumaAlumaa	5318	1229	382434	0	false	2023	5	4

```
spark.sql("""
select * from twitter_data.author_matrix
limit 3
""")
```

author_id	tweet_id	year	month	day	hour
1496586301486809094	1654239162386591745	2023	5	4	21
1573068165524996096	1654239161543454726	2023	5	4	21
1492197849760677897	1654239159106584578	2023	5	4	21

Facts

5. tweet_matrix_raw: This table contains tweet metrics such as retweet count, reply count, like count, and quote count. It's partitioned by year, month, day, and hour, and is populated by extracting data from the `twitter_data.tweets` table.

6. tweets_count_raw: This table contains the number of tweets per hour. It's partitioned by year, month, and day, and is populated by grouping tweets by hour and counting the number of tweets in each group from the `twitter_data.tweets` table.

7. most_active_hour_raw: This table contains the hour with the most tweets, along with the number of tweets in that hour. It's partitioned by year, month, and day, and is populated by grouping tweets by hour and selecting the hour with the highest tweet count from the `twitter_data.tweets` table.

```
spark.sql("""
select * from twitter_data.tweet_matrix_raw
limit 3
""")
```

id	retweet_count	reply_count	like_count	quote_count	year	month	day	hour
1654239167377817603	65	0	0	0	2023	5	4	21
1654239164462669833	26	0	0	0	2023	5	4	21
1654239162386591745	41	0	0	0	2023	5	4	21

```
spark.sql("""
select * from twitter_data.tweets_count_raw
limit 1
""")
```

hour	tweets_count	year	month	day
21	14300	2023	5	4

The business insight and meaning from these tables would provide valuable information about Twitter users, their behavior and engagement, and the most active times on the platform. For example, the `users_dim` table could be used to identify popular or influential users, while the `tweets_count_raw` and `most_active_hour_raw` tables could provide insights into peak engagement times for the platform.

The project runs each 15 minute

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command

15 * * * * itversity-material/spark_project/sql_hive.sh
~
~
~
~
```

```
1
2
3 /opt/spark2/bin/spark-sql  -f    itversity-material/spark_project/tables.sql
```

Usage:

- Run the script in the terminal or IDE of your choice.

- Open a socket connection to the port specified in the script.

- Send the data to the socket.

Results and Conclusion

Based on our analysis, we found that the most frequently mentioned topic related to "eid" on Twitter was "celebration", followed by "prayer" and "family". We also observed a significant spike in activity on the day of Eid al-Fitr, which marks the end of Ramadan. These findings suggest that "eid" is primarily associated with joyous celebrations and family gatherings among Twitter users.