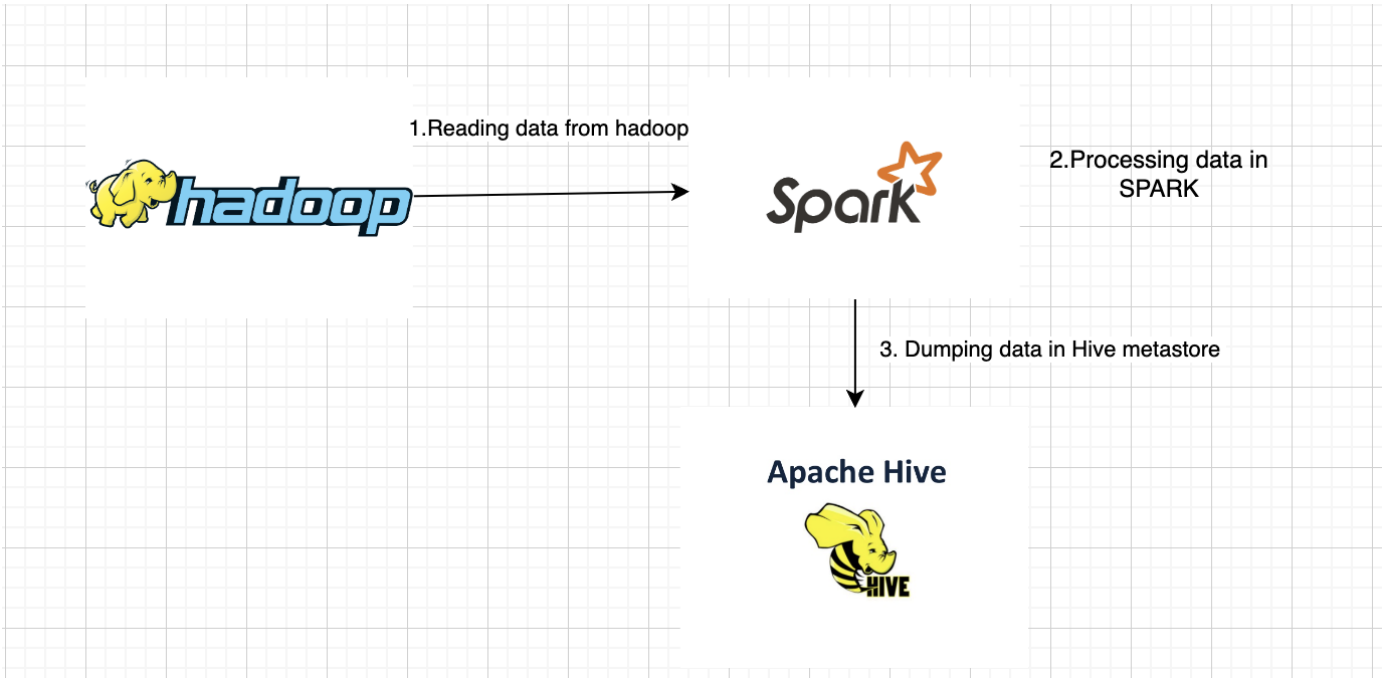
MovieLens Data processing using Spark

Table of Contents

1 Introduction

The Aim of the project is to process MoviesLens data and perform some analytical queries. The dataset contains 3 files (i.e., users, movies and ratings). These files contain 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 MovieLens users who joined MovieLens in 2000. This project helps in to viewing the insights about MoviesLens data.

1.1 Architecture



The above picture illustrates the architecture of the project. The numbering of each step explains the sequence of steps performed in ETL pipeline.

Explanation:

The process starts with downloading the MovieLens data from **https://grouplens.org/datasets/movielens/1m/.** The downloaded data is saved in Hadoop and later processed in spark. Using spark, we will perform data analysis on dataset. Finally, the processed data is dumped in hadoop and table schema is stored in hive metastore. This allows us to query data directly from hive without defining any DDL commands in hive. Note that we will use spark in standalone mode to process data and hive default database (i.e., derby) to store data as tables.

2 Implementation

2.1 Loading data to Hadoop

The download data needs to be placed in hadoop. The below command will put the data from local to HDFS.

Hadoop fs -put /path_of_File_located_in_local /path_of_HDFS_directory

```
abc@b8e7ca3bcded:~/workspace$ hadoop fs -put movies.dat /datasets
2023-04-05 15:15:30,500 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
abc@b8e7ca3bcded:~/workspace$ hadoop fs -put users.dat /datasets
2023-04-05 15:15:49,004 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
abc@b8e7ca3bcded:~/workspace$ hadoop fs -put ratings.dat /datasets
2023-04-05 15:16:02,875 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
abc@b8e7ca3bcded:~/workspace$
```

To list the files in HDFS directory, we need to use the below command

Hadoop fs -ls /directory_name

```
abc@b8e7ca3bcded:~/workspace$ hadoop fs -ls /datasets
Found 3 items
-rw-r--r--   1 abc supergroup     171308 2023-04-05 15:15 /datasets/movies.dat
-rw-r--r--   1 abc supergroup   24594131 2023-04-05 15:16 /datasets/ratings.dat
-rw-r--r--   1 abc supergroup     134368 2023-04-05 15:15 /datasets/users.dat
abc@b8e7ca3bcded:~/workspace$
```

2.2 Running a Spark Job

The spark code to process data is in Script.py file in the folder. The below command is used to run the spark job.

spark-submit script.py &> results.txt

- Spark-submit (used to run the spark job)
- Script.py (spark file which we write code for processing data)
- &> results.txt (saves output in separate file in text type)

3 Hive metastore

The processed data is stored in hadoop and table schema is stored in hive metastore

The below image shows spark storing table schema in hive meta store (i.e., derby)

```
+---------+---------+-----------+
|namespace|tableName|isTemporary|
+---------+---------+-----------+
|  default|   movies|      false|
|  default|  ratings|      false|
|  default|    users|      false|
+---------+---------+-----------+
```

The original data is stored in Hadoop as parquet files.

```
abc@c43933ef0efc:~/workspace$ hadoop fs -ls /config/workspace/spark-warehouse*
Found 3 items
drwxr-xr-x   - abc supergroup          0 2023-04-05 16:37 /config/workspace/spark-warehouse/movies
drwxr-xr-x   - abc supergroup          0 2023-04-05 16:37 /config/workspace/spark-warehouse/ratings
drwxr-xr-x   - abc supergroup          0 2023-04-05 16:37 /config/workspace/spark-warehouse/users
abc@c43933ef0efc:~/workspace$
```