Big data Project – Sqoop transfer

A project by Vinish Vijayan

vinishvijay.bigdata@icloud.com

+91 944 66 88 955

Project Name

Data transfer using SCOOP

Project Goal

The goal of this project is to develop a data pipeline that will transfer data from a relational database to a Hadoop cluster using SCOOP tool.

Design and development of a data pipeline using SQOOP

- Development of scripts to automate the data transfer process
- Testing and validation of the data pipeline
- Documentation of the data pipeline

General Problems

Difficult and time-consuming data transfer between RDBMS and HDFS.

Problem statement

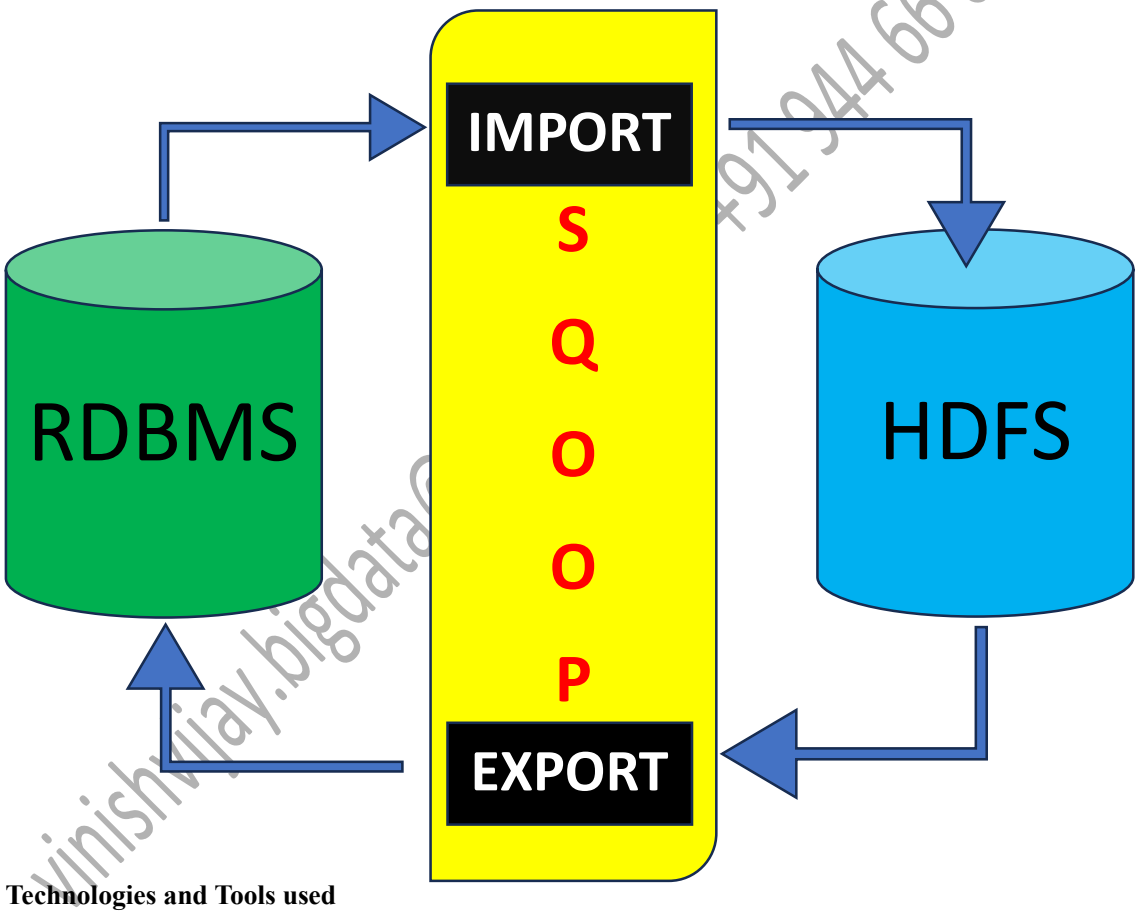
1. Importing weekly sales data of Dubai outlet from existing relational database (RDBMS) into a Hadoop Distributed File System (HDFS) directory can be a challenging task. The traditional approach of exporting data from the RDBMS into a flat file and then importing the file into HDFS can be time-consuming and inefficient.
2. Simple import of weekly sales data from Dubai outlet for cashew from a relational database (RDBMS) into a Hadoop Distributed File System (HDFS) directory is challenging task through the traditional approach.
3. Import of only cashew and pecan sales data from Dubai outlet from a relational database (RDBMS) into a Hadoop Distributed File System (HDFS) directory is challenging task through the traditional approach.
4. Import of only the sales data that has been added since 7th of April from a relational database (RDBMS) into a Hadoop Distributed File System (HDFS) directory is challenging task through the traditional approach.
5. Import of weekly sales data with 5 mappers from a relational database (RDBMS) into a Hadoop Distributed File System (HDFS) directory is a challenging task through the traditional approach.
6. Import of weekly sales data using secure password from a relational database (RDBMS) into a Hadoop Distributed File System (HDFS) directory is a challenging task through the traditional approach.
7. Import of weekly sales data from cloud into a Hadoop Distributed File System (HDFS) directory is challenging task through the traditional approach.
8. Import of weekly sales data as parquet file from a relational database (RDBMS) into a Hadoop Distributed File System (HDFS) directory is a challenging task through the traditional approach.
9. Import of weekly sales data that has been schema modified from a relational database (RDBMS) into a Hadoop Distributed File System (HDFS) directory is challenging task through the traditional approach.
10. Export of weekly sales data from a Hadoop Distributed File System (HDFS) directory into a relational database (RDBMS) is challenging task through the traditional approach.

11. Automating the import of weekly sales data from a relational database (RDBMS) into a Hadoop Distributed File System (HDFS) directory is a challenging task through the traditional approach.

Project Deliverables

1. A working model that can transfer data from RDBMS to HDFS.
2. A report on the model's performance.
3. Documentation of the model development process.

System Architecture



Technologies and Tools used

Sqoop

Project Module

1. Problem statement 1 - Import table to HDFS
    a. Explanation and Solution

Normal import command can be used here to perform import of weekly sales data to HDFS. specify the target directory while importing table data into HDFS using the Sqoop import tool.
Following command is used to import the sales data.

sqoop import \ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME> --username <USERNAME> \ --password <PASSWORD>\ --table <TABLENAME>\ --m <NUMBER> \ --delete <TARGETDIR> \ --target-dir <TARGET DIR>	hostname – localhost port –3306 database –dubaidata username –root password –cloudera table –dubai_sales mappers --1 delete target-dir target dir --/user/cloudera/nimport

2. Problem statement 2 - Import Portion data to HDFS using WHERE
    a. Explanation and Solution
    Portion import of the weekly sales data where sale date = 02-04-2017. Importing a subset of a table using the 'where' clause in Sqoop import tool. It executes the corresponding SQL query in the respective database server and stores the result in a target directory in HDFS.
    The following command is used to import a subset of dubai_sales table data. The subset query is to retrieve the sales data, on 02-04-2017 date.

sqoop import \ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME> --username <USERNAME> \ --password <PASSWORD>\ --table <TABLENAME>\ --m <NUMBER> \ --delete <TARGETDIR> \ --target-dir <TARGET DIR> \ --where <FILTER>	--where ''sale_date = '02-04-2017'""

3. Problem statement 3 - Import Portion data to HDFS using QUERY
    a. Explanation and Solution

sqoop import \ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME> --username <USERNAME> \ --password <PASSWORD>\ --table <TABLENAME>\ --m <NUMBER> \ --target-dir <TARGET DIR>\ --query "<QUERY> where \$ CONDITIONS;"	query - select a.*, b.products from zeyotab a join zeyoprod b on a.id-b.id

4. Problem statement 4 - Import incremental data to HDFS
    a. Explanation and Solution

sqoop import \ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME> --username <USERNAME> \ --password <PASSWORD>\ --table <TABLENAME>\ --m <NUMBER> \ --target-dir <TARGET DIR>\ --incremental <TYPE>\ --check-column <CLOUMNNAME>\ --last-value <NUMBER>	--incremental append\ --check-column id\ --last-value 4

5. Problem statement 5- Import data to HDFS using multiple mappers
    a. Explanation and Solution
    Here the file will be imported and will be split up into 5 different files. Mappers help to do the task faster, default no of mappers if no mappers mentioned is 4. There is no limit to the max number of mappers

sqoop import \ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME> --username <USERNAME> \ --password <PASSWORD>\ --table <TABLENAME>\ --m <NUMBER> \ --delete <TARGETDIR> \ --target-dir <TARGET DIR>	hostname – localhost port –3306 database –dubaidata username –root password –cloudera table –dubai_sales mappers --4 delete target-dir target dir --/user/cloudera/nimport

6. Problem statement 6 - Import data to HDFS using encrypted password
    a. Explanation and Solution
    Password when given along with the command is not secure and for that prepare a password in the file and root the command to get the password from the file. When preparing the password make sure there is no have any white line character.

sqoop import \ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME> --username <USERNAME> \ --password -file://<ABSOLUTEPATHFILE>\ --table <TABLENAME>\ --m <NUMBER> \ --target-dir <TARGET DIR>	Keep the Password in the file and root the command to read from this file. The file should not have NEW LINE CHARACTER is not there in file. echo -n <PASSWORD> > <absolute path FILENAME> echo -n cloudera > /home/cloudera/passfile

7. Problem statement 7 - Import data to HDFS from cloud
    a. Explanation and Solution
    For export from cloud require 3 new parameters access key, secret key, end point. All these details can be got from the cloud.

sqoop import \ -<ACCESSKEY>\ -<SECRETKEY>\ -<ENDPOINT>\ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME> --username <USERNAME> \ --password <PASSWORD>\ --table <TABLENAME>\ --m <NUMBER> \ --target-dir <TARGET DIR>\	Sqoop cloud import

8. Problem statement 8 - Import data to HDFS as serialized data
    a. Explanation and Solution
    In case of import of data to HDFS for special reasons we can get assistance from serialized file format. Some of the advantages of the different file format are listed below, based on the requirement and needs of the project we can decide on which file format to use.
    AVRO file where - schema evolution is required
    Row format
    40-60% compression
    Parquet file where - columnar format
    Faster querying
    Target system
    60-80% compression
    Predicate pushdown
    ORC file format - Columnar file format
    Faster querying not as fast as parquet
    Historical storage
    90-95% compression

sqoop import \ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME> --username <USERNAME> \ --password <PASSWORD>\ --table <TABLENAME>\ --m <NUMBER> \ --delete-target-dir \ --target-dir <TARGET DIR>\ --as-avrodatafile	Sqoop import with file format change

9. Problem statement 9 - Import schema modified data to HDFS
    a. Explanation and Solution
    In case of any modification of data we use modified data import.

sqoop import \ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME>	Modified import – incremental should be lastmodified,

--username <USERNAME> \ --password <PASSWORD>\ --table <TABLENAME>\ --m <NUMBER> \ --delete-target-dir \ --target-dir <TARGET DIR>\ --incremental lastmodified \ --check-column <COLUMNNAME> \ --last-value <VALUETOCONCIDER>\ --merge-key <MODIFIEDCOLUMN>	

10. Problem statement 10 - Export data from HDFS
    a. Explanation and Solution

    Export command is used to export the data from HDFS to RDBMS, new parameters used are export and export-dir commands. Here due to the risk of data loss due to any interruptions, staging-table is used. The data is first transferred to the staging table, once the process is completed then the data is transferred to the final table in RDBMS.

sqoop export \ --connect jdbc:mysql://<HOSTNAME><PORT> \ --username <USERNAME> \ --password <PASSWORD> \ --table <TABLENAME> \ --staging-table <TABLENAME> \ --m <NUMBEROFMAPPERS> \ --export-dir <DIRPATH>	Instead of import use the export command along with export-dir instead of export directory

11. Problem statement 11- Automate the Import using SQOOP JOB
    a. Explanation and Solution

    To automate the process of data transfer between RDBMS and HDFS SQOOP job is used. First time we execute the SQOOP job with last value as 0 and after that SQOOP will store the last value in the system. We only need to run exec command. To see the details of the SQOOP job use show command.

sqoop job –create <JOBNAME> \ --import \ --connect jdbc:mysql://<HOSTNAME>:<PORT>/<DATABASENAME> --username <USERNAME> \ --password <PASSWORD>\ --table <TABLENAME>\ --m <NUMBER> \ --target-dir <TARGET DIR>\ --incremental <TYPE>\ --check-column <CLOUMNNAME>\ --last-value 0	Sqoop incremental command with last value 0 sqoop job zeyojob

sqoop job –show <JOBNAME>	It will list the properties of the sqoop job sqoop job –show zeyojob
sqoop job –exec <JOBNAME>	It will start the sqoop job Sqoop job -exec zeyojob
sqoop job –list	List all the sqoop jobs

Project Result

The project was successful in developing a model that can transfer data from RDBMS to HDFS. The model was deployed to production and is being used to transfer data.

Project Lesson learned

The following are some of the lessons learned during the project:

The data can be transferred from RDBMS to HDFS using the import command and can be exported using the export command. All the process can be automated using the SCOOP job.

Project conclusion

The project was a success in developing data transfer from RDBMS to HDFS. The model is being used to transfer sales data. The project team learned a number of valuable lessons about the importance of SQOOP tool.