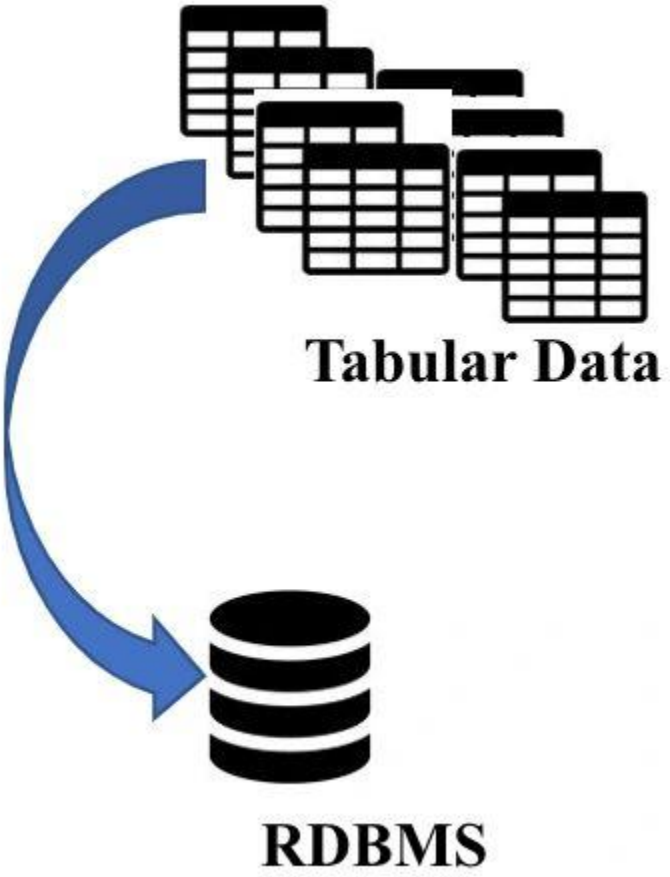
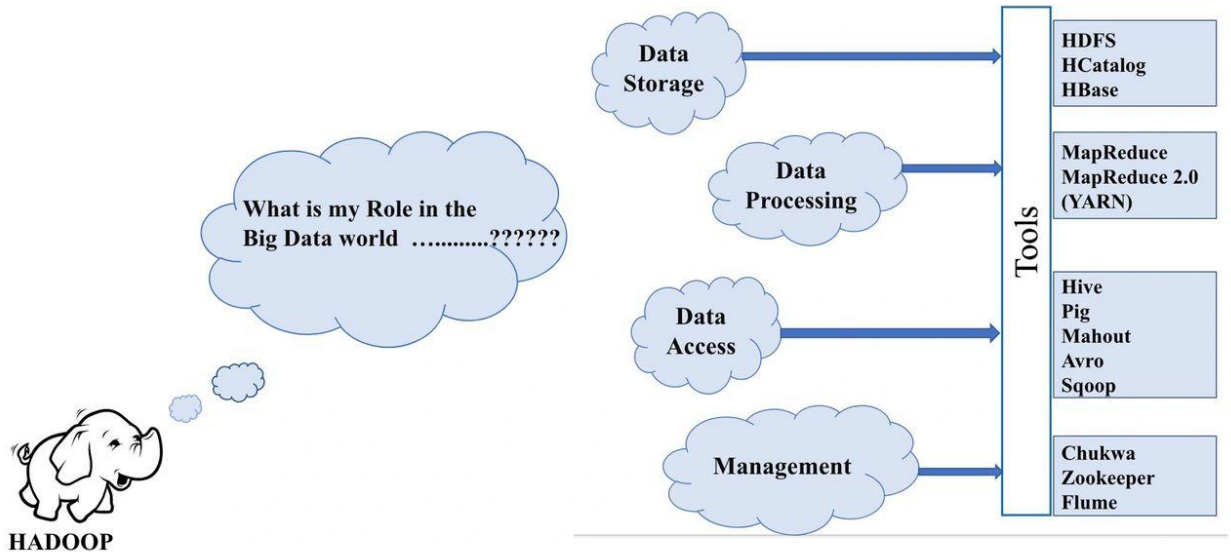
Let's Start...!!

Big Data is a hot topic in the present world and has transformed the face of every business field within a short period. Business giants like Netflix, Amazon and Facebook are few examples. Because of its volume, it needs specific and different tools to handle the Big data in each stage of its processing. Sqoop is one of the prominent big data tools and before getting started with Sqoop deeper, let's try to understand about Two terms - RDBMS & Hadoop, since these are the important components driving this Sqoop session.

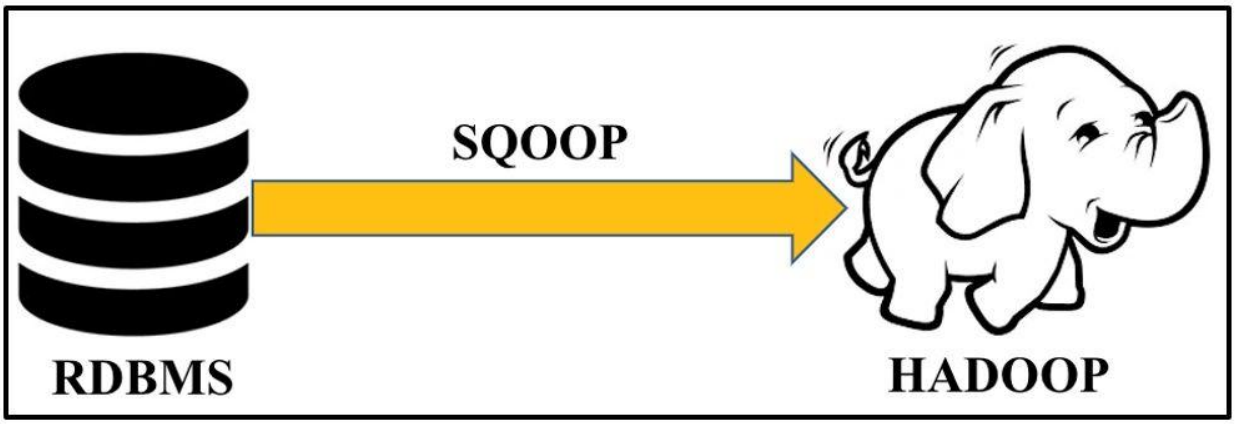


RDBMS: It stands for Relational DataBase Management system and it is one of the common database. It is designed for storing large amounts of data in a tabular form. A table is a collection of related data entries and contains records (rows) and fields (columns) to store data. The data is often stored in many tables and thus these tables may or may not have relationships.

HADOOP: Apache Hadoop is an open source framework that is used to efficiently store and process large volumes of data. Hadoop uses multiple computer clusters to do the assigned tasks in a parallel mode within a short span of time.

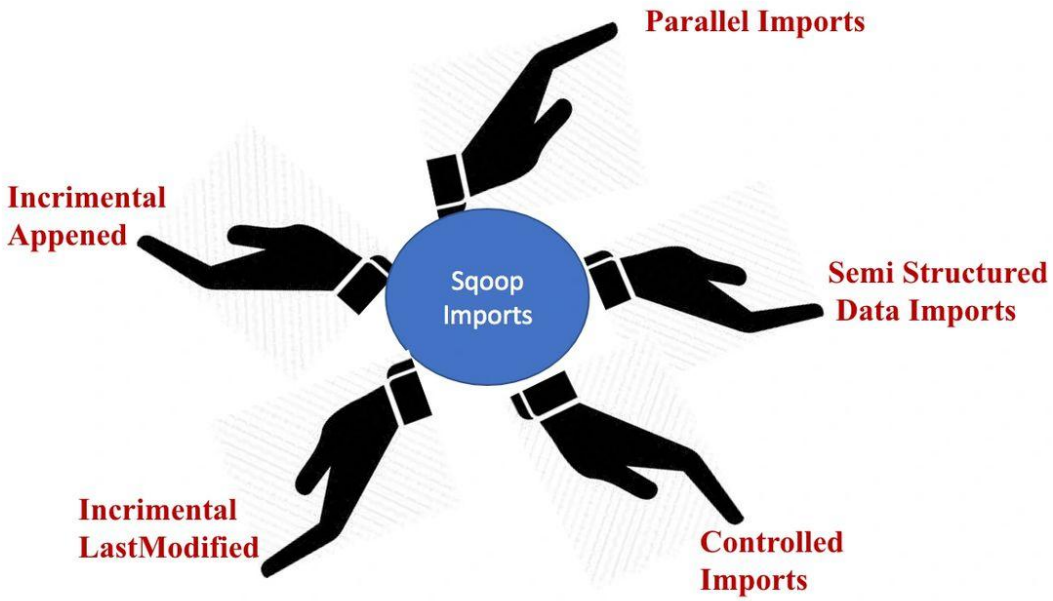


Apache Sqoop has the ability to connect the above discussed Two components - RDBMS and HADOOP. It is a popular big data tool for data import and export between Hadoop and external data stores such as relational databases. These transferred data will be used for data analytics. It uses the MapReduce mechanism for transmitting bulk data.



<u>Sqoop features</u>

Depending on the situation and requirements, it is possible to choose one of the following features, while doing data transmission.



And as we discussed above, the flow of data will be from RDBMS to HADOOP with the help of Sqoop. Initiatively, let's put some sample data in a tabular form in RDBMS. This table will be the input data in the upcoming examples.

Student Id	Student Name	Subject	Marks	ExamDate
1	John	Science	25	2021-07-23
2	Bobby	Science	30	2021-07-23
3	James	English	28	2021-07-23
4	Adam	English	35	2021-07-23
5	Kelvin	Science	30	2021-07-23

Sample data: Students

<u>Steps</u>

1. Create a Database

2. Create a sample Table in the Database to update the Student's Exam details. Table consisting of 5 columns - Student Id, Student Name, Subject, Marks, Exam Date.

3. Insert values into the Table.

Code

Step	Code
Database Creation	>CREATE DATABASE test
Table Creation	>USE test >CREATE TABLE Students (StudentId int, StudentName varchar(25), Subject varchar(10), Marks int, ExamDate date);
Insert values into the Table	>INSERT INTO Students VALUES (1, 'John','Science',25, now()); >INSERT INTO Students VALUES (2, 'Bobby','Science',30, now()); >INSERT INTO Students VALUES (3, 'James','English',28, now()); >INSERT INTO Students VALUES (4, 'Adam','English',35, now()); >INSERT INTO Students VALUES (5, 'Kelvin','Science',30, now());
Output verification	>SELECT * FROM Students

Thus we have placed some data on the RDBMS side. Now let's understand the Sqoop import features through the following segments.

- **About: Feature description**
- **Example: A sample problem with its complete solution architecture and code.**

Feature1: Parallel Imports

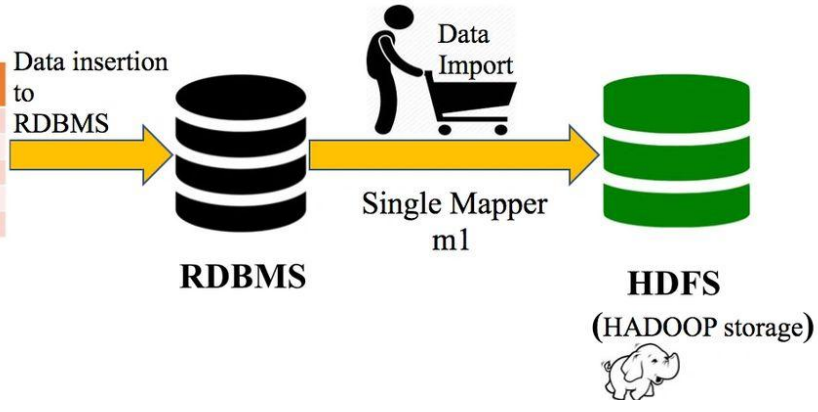
About: It is a common sqoop feature in use, especially to import large data. The argument *-m* representing the number of Mappers invoked (similar to how many people are working) to import the data. That is, mk - k indicates how many mapper tasks will be invoked by the sqoop command. For example, m3 means 3 mappers will be invoked by the sqoop command for importing data. The default number of mappers is 4. Once the task is complete, these mappers will place the results into the HDFS target directory. Single mapper can be used if the data size is small. And if the data volume is large, it is better to use multi mappers for performing the import process in parallel to achieve the task in a short time.

Example

Problem: Import the table Students from RDBMS to HDFS using single Mapper and using Three Mappers (Multi mappers or Parallelism).

Student Id	Student Name	Subject	Marks	ExamDate
1	John	Science	25	2021-07-23
2	Bobby	Science	30	2021-07-23
3	James	English	28	2021-07-23
4	Adam	English	35	2021-07-23
5	Kelvin	Science	30	2021-07-23

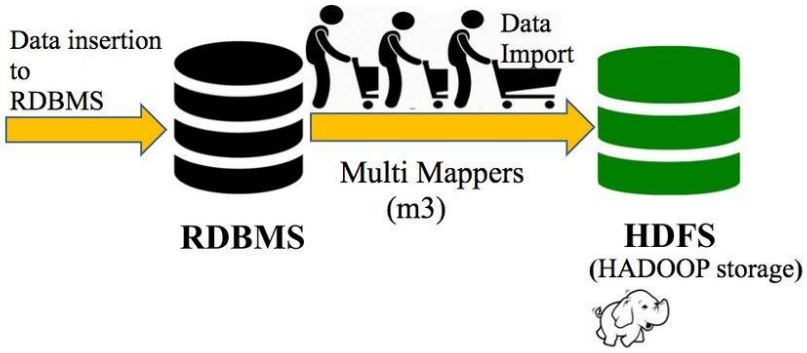
Input table: Students



Solution Architecture: Single Mapper

Student Id	Student Name	Subject	Marks	ExamDate
1	John	Science	25	2021-07-23
2	Bobby	Science	30	2021-07-23
3	James	English	28	2021-07-23
4	Adam	English	35	2021-07-23
5	Kelvin	Science	30	2021-07-23

Input table: Students



Solution Architecture: Multi Mappers

Code

Step	Code
	Db name: test Table name: Students Hadoop target directory: /user/cloudera/outputs
Import table1 from RDBMS to HDFS using single mapper.	sqoop import --connect jdbc:mysql://localhost/test --username ***--password ***--table Students --m 1 --delete-target-dir --target-dir /user/cloudera/outputs;
Import the data from RDBMS to HDFS using Multi mapper (m3)	sqoop import --connect jdbc:mysql://localhost/test --username ***--password ***--table Students --m 3 --split-by StudentID --delete-target-dir --target-dir /user/cloudera/outputs;
Output verification (*Print the output data in the target directory*)	hadoop fs -cat /user/cloudera/outputs/*

Expected Output: Student table data should present in the HDFS target directory - /user/cloudera/outputs

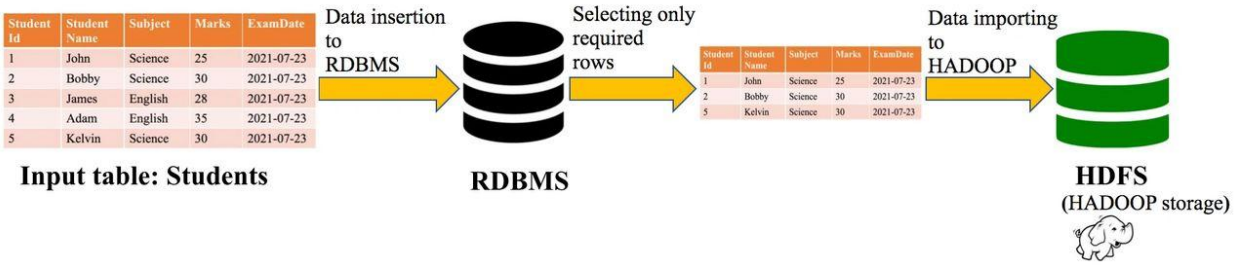
<u>Feature2: Controlled Imports</u>

<u>About</u>: In some situations, importing the entire table will not be necessary. In such a case, it is possible to import the required records by giving some condition over table rows (row subsetting) or table columns (column subsetting).

Row Subsetting: We can control the rows to be imported by using a SQL WHERE clause to the import statement or by giving a Query argument separately.

<u>Example</u>

Problem: From the table Students, import records with Science subject using single mapper.



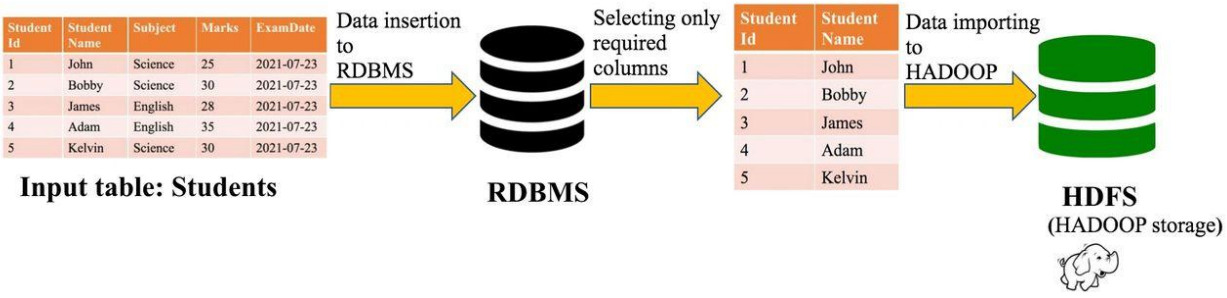
Solution Architecture
Code

Step	Code
	Db name: test Table name: Students Hadoop target directory: /user/cloudera/outputs
Import the data from RDBMS to HDFS based on the given condition. (using query clause)	sqoop import --connect jdbc:mysql://localhost/test --username ***--password *** --query 'select * from test.Students where Subject= "Science" and $CONDITIONS' --m 1 --delete-target-dir --target-dir /user/cloudera/outputs;
(using where clause)	sqoop import --connect jdbc:mysql://localhost/test --username ***--password *** --table Students --where "Subject='Science'" --m 1 --delete-target-dir --target-dir /user/cloudera/outputs;
Output verification (*Print the output data in the target directory*)	hadoop fs -cat /user/cloudera/outputs/*

Subsetting Columns: It is also possible to control the columns to be imported by using *-columns* argument.

<u>Example</u>

Problem: From table Students, import only student id and student name information.



Solution Architecture
Code

Step	Code
	Db name: test Table name: Students Hadoop target directory: /user/cloudera/outputs
Import the data from RDBMS to HDFS based on the given condition.	sqoop import --connect jdbc:mysql://localhost/test --username ***--password ***--table Students --columns "StudentID, StudentName" --m 1 --delete-target-dir --target-dir /user/cloudera/outputs;
Output verification (*Print the output data in the target directory*)	hadoop fs -cat /user/cloudera/outputs/*

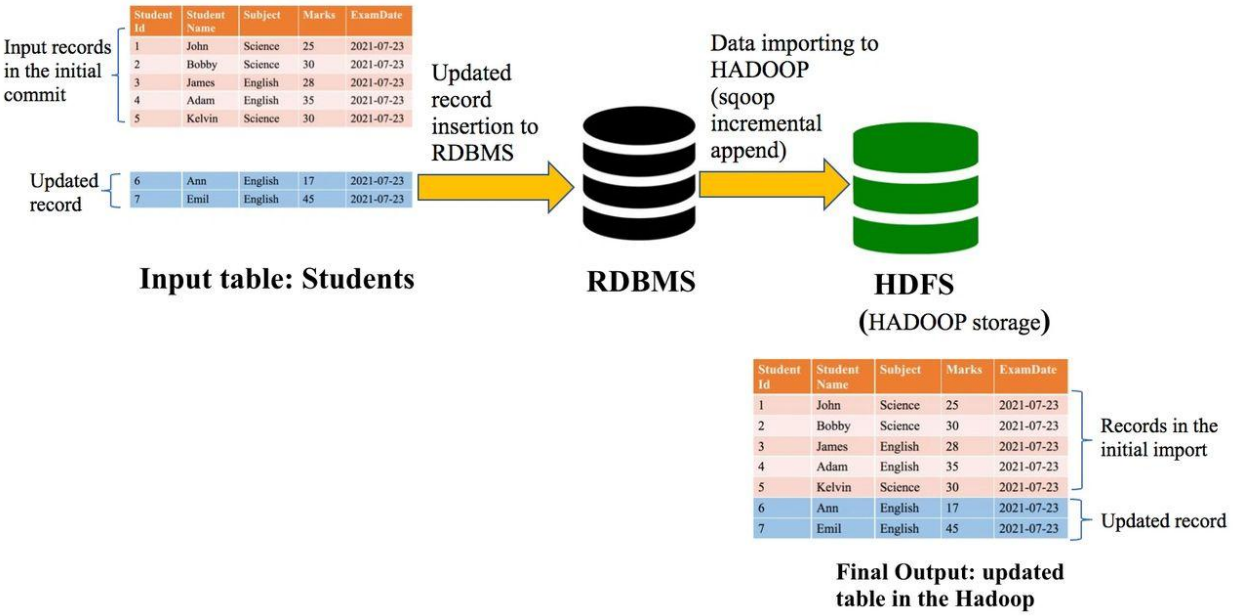
Expected Output: Student table data should be present in the HDFS target directory - /user/cloudera/outputs

Feature3: Incremental Append

About: In practice, new records are frequently added to respective tables. In such a situation, it is a tedious job to import the entire table in every import execution and thus there is a need to import only newly inserted or updated data from the database. Sqoop can handle this scenario by using -incremental append. It will import only records after a certain threshold value, instead of importing the entire table each time.

Example

Problem: Import additionally added new records in the table Students to the target directory.



Student Id	Student Name	Subject	Marks	ExamDate
1	John	Science	25	2021-07-23
2	Bobby	Science	30	2021-07-23
3	James	English	28	2021-07-23
4	Adam	English	35	2021-07-23
5	Kelvin	Science	30	2021-07-23
6	Ann	English	17	2021-07-23
7	Emil	English	45	2021-07-23

Solution Architecture

Code

Step	Code
	Db name: test Table name: Students Hadoop target directory: /user/cloudera/outputs
Import the updated records or new records into HDFS target directory	sqoop import --connect jdbc:mysql://localhost/test --username ***--password ***--table Students --m 1 --target-dir /user/cloudera/outputs --incremental append --check-column StudentID --last-value 5;
Output verification (*Print the output data in the target directory*)	hadoop fs -cat /user/cloudera/outputs/*

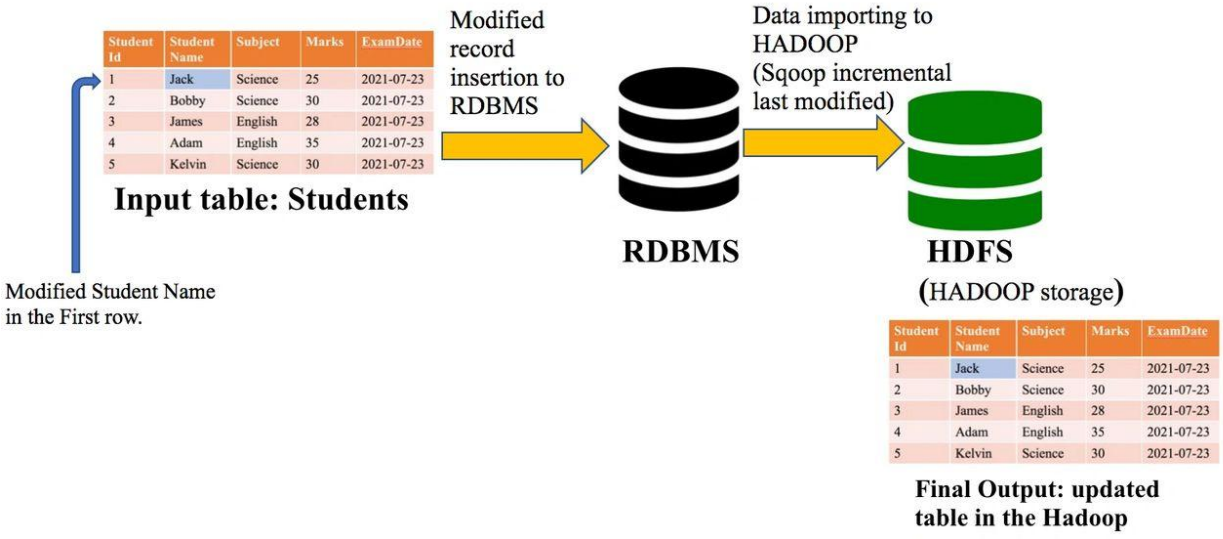
Expected Output: Student table data with updated records should be present in the HDFS target directory - /user/cloudera/outputs

Feature 4: Incremental Lastmodified

About: Similar to Incremental append, Incremental Last modified is also handling the changes in the RDBMS table. It helps to import the modified or updated records in the table. Sqoop can handle this scenario by using -incremental append. It will import only records after a certain threshold value, instead of importing the entire table each time.

Example

Problem: Import modified records in the table Students to the target directory.



Student Id	Student Name	Subject	Marks	ExamDate
1	Jack	Science	25	2021-07-23
2	Bobby	Science	30	2021-07-23
3	James	English	28	2021-07-23
4	Adam	English	35	2021-07-23
5	Kelvin	Science	30	2021-07-23

Solution Architecture

Code

Step	Code
	Db name: test Table name: Students Hadoop target directory: /user/cloudera/outputs
Import the modified records into HDFS target directory	sqoop import --connect jdbc:mysql://localhost/test --username ***--password *** --table Students --m 1 --target-dir /user/cloudera/outputs --incremental lastmodified --check-column ExamDate --last-value 1 --merge-key StudentID;
Output verification (*Print the output data in the target directory*)	hadoop fs -cat /user/cloudera/outputs/*

Expected Output

Student table data with modified records should be present in the HDFS target directory - /user/cloudera/outputs

Feature 5: SemiStructured Data Import

About: Sqoop can also efficiently transfer the data in a different semi-structured formats such as

1. PARQUET files

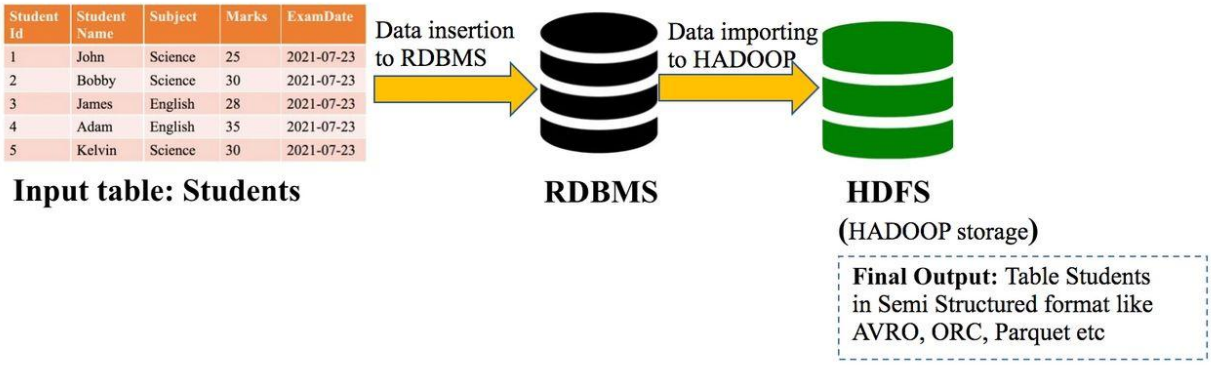
2. Sequence file

3. Avro files

4. Text file

Example

Problem: Import table Students to the target directory in the above mentioned formats.



Solution Architecture

Code

Step	Code
	Db name: test Table name: Students Hadoop target directory: /user/cloudera/outputs
Import the data from RDBMS to HDFS as Parquet file	sqoop import --connect jdbc:mysql://localhost/test --username ***--password ***--table Students --m 1 --delete-target-dir --target-dir /user/cloudera/outputs --as-parquetfile
Import the data from RDBMS to HDFS as Sequence file	sqoop import --connect jdbc:mysql://localhost/test --username ***--password ***--table Students --m 1 --delete-target-dir --target-dir /user/cloudera/outputs --as-sequencefile
Import the data from RDBMS to HDFS as Avro file	sqoop import --connect jdbc:mysql://localhost/test --username ***--password ***--table Students --m 1 --delete-target-dir --target-dir /user/cloudera/outputs --as-avrodatafile
Import the data from RDBMS to HDFS as Text file	sqoop import --connect jdbc:mysql://localhost/test --username ***--password ***--table Students --m 1 --delete-target-dir --target-dir /user/cloudera/outputs --as-textfile
Output verification (*Print the output data in the target directory*)	hadoop fs -cat /user/cloudera/outputs/*

Expected Output

Student table data should present in the given Semi structured format in the HDFS target directory - /user/cloudera/outputs

Summary

Sqoop has a significant role in this era and it is acting like a bridge for Big data transportation between RDBMS and HADOOP. This tool has its own features and it helps to improve this data transportation quality at different levels such as speed, cost etc. Several analysis and tasks can be performed over this imported data using different analytical tools, which will be the key solution for many business problems. Thus the performance of Sqoop influences the entire solution pipeline.