Aim

To use Sqoop to transfer data between relational databases and Hadoop, and to execute basic Sqoop commands to demonstrate this functionality.

Software Requirements

- Hadoop (HDFS and MapReduce)
- Sqoop
- MySQL (or another relational database)
- Java JDK

Theory

Introduction to Sqoop: Sqoop is a powerful tool that simplifies the process of transferring bulk data between Hadoop and structured databases like MySQL, Oracle, PostgreSQL, and others. It allows for efficient data import from relational databases into Hadoop's HDFS for processing with Hadoop tools like MapReduce, Hive, and Pig. Conversely, it can also export processed data back from HDFS to the relational database.

Sqoop Architecture:

- **Client:** Users interact with Sqoop through a command-line interface (CLI) to submit data transfer jobs.
- **Connectors:** Sqoop uses specific connectors to communicate with different databases via JDBC.
- **MapReduce Jobs:** Sqoop generates MapReduce jobs to perform data transfer in parallel, which ensures scalability and efficiency.
- **HDFS:** The destination for imported data or the source for exported data within the Hadoop ecosystem.

Key Commands and Functions:

1. **Import Data from RDBMS to Hadoop:**
    - This command transfers data from a database table into HDFS.
    - Example: `sqoop import --connect jdbc:mysql://localhost/db_name --username root --password password --table table_name --m 1`
2. **Export Data from Hadoop to RDBMS:**
    - This command exports data from HDFS back into a database table.
    - Example: `sqoop export --connect jdbc:mysql://localhost/db_name --username root --password password --table table_name --export-dir /user/hadoop/output_dir`

3. **List Databases:**
    - Lists all databases on the connected RDBMS.
    - Example: `sqoop list-databases --connect jdbc:mysql://localhost/ --username root --password password`
4. **List Tables in a Database:**
    - Lists all tables within a specified database.
    - Example: `sqoop list-tables --connect jdbc:mysql://localhost/db_name --username root --password password`
5. **Incremental Import:**
    - Imports only new or updated records since the last import.
    - Example: `sqoop import --connect jdbc:mysql://localhost/db_name --username root --password password --table table_name --incremental append --check-column id --last-value 100`
6. **Code Generation:**
    - Generates Java classes corresponding to the database schema for use in applications.
    - Example: `sqoop codegen --connect jdbc:mysql://localhost/db_name --username root --password password --table table_name`

Sqoop Data Transfer Scenarios:

- **From MySQL to HDFS:** Import data from MySQL to Hadoop for analysis.
- **From HDFS to Oracle:** Export processed data from Hadoop to an Oracle database.
- **From PostgreSQL to HDFS:** Import data from PostgreSQL into Hadoop.
- **From HDFS to MySQL:** Export data back to MySQL after processing.

Outcome

By performing the Sqoop commands, students will learn how to efficiently transfer data between various databases and Hadoop, showcasing Sqoop's flexibility and importance in big data processing.

Conclusion

Sqoop is a crucial tool in the Hadoop ecosystem, enabling seamless and scalable data transfers between relational databases and Hadoop. It plays an essential role in managing big data, ensuring that data is efficiently transferred for processing and storage across different platforms.