Workshop 6

Hadoop Data Ingestion

General Instructions:

Hadoop Data ingestion is the beginning of your data pipeline in a data lake. It means taking data from various silo databases and files and putting it into Hadoop. Hadoop is an open source framework; there are a variety of ways you can ingest data into Hadoop. It gives every developer the choice of using her/his favorite tool or language to ingest data into Hadoop. In this workshop we will focus on two particular tools to ingest data into Hadoop, Apache Sqoop and Apache Flume.

Apache Sqoop is a tool designed to efficiently transfer data between Hadoop and relational databases. We can use Sqoop to import data from a relational database table into HDFS. The import process is performed in parallel and thus generates multiple files in the format of delimited text, Avro, or SequenceFile. Besides, Sqoop generates a Java class that encapsulates one row of the imported table, which can be used in subsequent MapReduce processing of the data. Moreover, Sqoop can export the data (e.g. the results of MapReduce processing) back to the relational database for consumption by external applications or users.

Start your Cloudera QuickStart VM to complete the workshop

Online resources:

https://sqoop.apache.org/

Apache Sqoop

The purpose of these exercises is to familiarize you with the basic operations of Apache Sqoop. You will have the task of importing data from a MySQL database to Hive and vice versa.

Prerequisites:

To complete this exercise, you will have to install the **employees_DB** sample database on the MySQL server already preinstalled on your Cloudera environment. Once the database is installed you can use Apache Sqoop to import data from this database into Hadoop and export from Hadoop into this database.

To install the **employees_DB** database, follow the instructions below:

1. Open a new Terminal window
2. Check your MySQL server is running using this command:

```
$ sudo service mysqld status
```

This command will display something similar to this.

```
mysqld (pid  ~~1850~~) is running...
```

3. Create a local directory `sqoop` and copy the provided archive into this directory
4. Navigate to `sqoop` folder
5. Extract the provided archive to the `sqoop` directory using the following command:

   ```
   $ tar xvf employees_db-full-1.0.6.tar.bz2
   ```

6. Navigate to the new folder `employees_db` created into `sqoop` folder
7. Run the following command to create a new database `employees_DB` and populate it with sample data

   ```
   $ mysql -u root -p -t < employees.sql
   ```

   enter the password when prompted
   (user : **root**   password : **cloudera**)

8. Connect to the MySQL server using the following command:

   ```
   $ mysql -u root -p
   ```

   enter the password when prompted

9. At this level, your prompt should be **mysql>**. To check the new database, display all databases on the server:

   ```
   show databases;
   ```

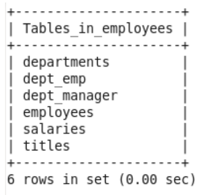
   should display a database named ***employees***

10. Activate the `employees` database and make it the current database

    ```
    use employees;
    ```

11. display all tables in the current database

    ```
    show tables;
    ```

    ```
    +---------------------+
    | Tables_in_employees |
    +---------------------+
    | departments         |
    | dept_emp            |
    | dept_manager        |
    | employees           |
    | salaries            |
    | titles              |
    +---------------------+
    6 rows in set (0.00 sec)
    ```

12. Now you need to give the cloudera user rights and permissions to access the employees database.

    ```
    grant ALL on employees.* to cloudera@localhost
    identified by 'cloudera';
    ```

13. You can now close the MySQL connection. To close the MySQL connection type **`exit ;`** and hit **`Enter`**

Everything now is done, you are ready to use Apache Sqoop

Exercise 1: Sqoop basic commands

In this exercise you will interact with Sqoop to list and import `employees` database tables into HDFS.

1. Open a new Terminal window
2. Use Sqoop to list all databases on the running MySQL local instance.

```
$ sqoop list-databases \
  --connect jdbc:mysql://localhost/ \
  --username cloudera --password cloudera
```

```
information_schema
employees
```

3. Use Sqoop to list all tables of the employees database

```
$ sqoop list-tables \
  --connect jdbc:mysql://localhost/employees \
  --username cloudera --password cloudera
```

```
departments
dept_emp
dept_manager
employees
employees_exp_stg
employees_export
salaries
titles
```

4. Import **employees** table into /user/cloudera/sqoop-mysql/employees HDFS directory. (use only one mapper)

```
$ sqoop import --connect jdbc:mysql://localhost/employees --
  username cloudera --password cloudera --table employees -m 1 --
  target-dir /user/cloudera/sqoop-mysql/employees
```

5. Check the target HDFS folder and determine the size of the imported table

Exercise 2: Import data into Hive

In this exercise, you will use Sqoop to import 'salaries table from 'employees' database into Hive.

1. Use Hive to create a new database emp_mysql
2. Use Sqoop to list all tables in the employees database
3. Import (using only one mapper) **`salaries`** table (**`employees`** database) and set :

- Hive target database as **emp_mysql**
- Hive target table as **salaries**.

4. Write a HiveQL query to list the first 20 rows in the salaries table
5. Write a HiveQL query to count all rows in salaries table
6. List recursively emp_mysql Hive database HDFS path
7. Determine the size on HDFS of the salaries table
8. Use the Linux wc command to count the rows in the physical salaries bucket file on HDFS (imported by Sqoop). Is there any difference with the original MySQL table?

Exercise 3: Import Hive partition

In this exercise you will use Sqoop to import data from MySQL employees table into a Hive partition 'gender'.

1. Open a new connection to MySQL server (password: cloudera)

```
$  mysql -u cloudera -p
```

2. to print all columns of the employees table enter this command:

```
describe employees.employees;
```

3. Write a query to group and count employees table rows by gender.

```
+--------+----------+
| gender | count(*) |
+--------+----------+
| M      |   179973 |
| F      |   120051 |
+--------+----------+
```

4. Use Sqoop to import only rows from employees table with 'gender'= '**M**'

```
database: emp_mysql
table:    employees_part
```

5. List recursively emp_mysql Hive database HDFS path
6. Write a HiveQL query to count the rows in the 'gender' partition. Is there a difference with the original MySQL table?

Exercise 4: Export Hive table

In this exercise, you will use Sqoop to export data from HDFS to MySQL server. For that we will export the Hive Bixi Stations table (created in previous workshop) to MySQL.

1. On MySQL create a new table stations (you can use the employees database)

```
CREATE TABLE stations_export_hive (code varchar(6),name
varchar(50),latitude DOUBLE,longitude DOUBLE);
```

2. Use Sqoop to export Hive stations table into MySQL stations_export_hive table
3. Write a SQL query to count the in stations_export_hive table. What do you notice?

Exercise 5: Import Other Format

In this exercise, you will use Sqoop to import 'salaries table from 'employees' database into Hive in avro format.

1. Use Sqoop to import the `salaries` table (`employees` database) using one mapper and set :
   - The file format to avro
   - The compression codec to snappy
   - The target directory as `/sqoop-avro-import`
2. Use the `avro-tools` command line tool to get the schema of the imported avro file
3. Place the schema on HDFS
4. Create a non-managed Hive table the read the avro file
5. Write a HiveQL query to list the first 20 rows in the salaries table
6. Write a HiveQL query to count all rows in salaries table
7. Determine the size on HDFS of the salaries table