Hortonworks Data Platform

Importing Data into HBase

(October 13, 2015)

docs.cloudera.com

Hortonworks Data Platform: Importing Data into HBase

The Hortonworks Data Platform, powered by Apache Hadoop, is a massively scalable and 100% open source platform for storing, processing and analyzing large volumes of data. It is designed to deal with data from many sources and formats in a very quick, easy and cost-effective manner. The Hortonworks Data Platform consists of the essential set of Apache Hadoop projects including MapReduce, Hadoop Distributed File System (HDFS), HCatalog, Pig, Hive, HBase, ZooKeeper and Ambari. Hortonworks is the major contributor of code and patches to many of these projects. These projects have been integrated and tested as part of the Hortonworks Data Platform release process and installation and configuration tools have also been included.

Unlike other providers of platforms built using Apache Hadoop, Hortonworks contributes 100% of our code back to the Apache Software Foundation. The Hortonworks Data Platform is Apache-licensed and completely open source. We sell only expert technical support, training and partner-enablement services. All of our technology is, and will remain, free and open source.

Please visit the Hortonworks Data Platform page for more information on Hortonworks technology. For more information on Hortonworks services, please visit either the Support or Training page. Feel free to contact us directly to discuss your specific needs.

Table of Contents

1. Importing Data into HBase

Bulk import bypasses the HBase API and writes contents, properly formatted as HBase data files (HFiles), directly to the file system. Bulk load uses fewer CPU and network resources than using the HBase API for similar work.

Prerequisite

To avoid problems with permissions while bulk loading data into HBase, configure secure bulk loading by adding the following values in your `hbase-site.xml` file:

- Add the SecureBulkLoadEndpoint coprocessor to the existing list of RegionServer coprocessors you have configured for <`hbase.coprocessor.region.classes`>
- Set the staging directory property, <`hbase.bulkload.staging.dir`>, to point to `/apps/hbase/staging`

These properties are bolded in the following example `hbase-site.xml` file:

```
<property>
     <name>hbase.bulkload.staging.dir</name>
     <value>/apps/hbase/staging</value>
</property>

<property>
     <name>hbase.coprocessor.region.classes</name>
     <value>org.apache.hadoop.hbase.security.token.TokenProvider,
            org.apache.hadoop.hbase.security.access.AccessController,
            org.apache.hadoop.hbase.security.access.SecureBulkLoadEndpoint
     </value>
</property>
```

To bulk load data into HBase using Pig:

1. Prepare the input file. The following `data.tsv` file is an example input file:

   ```
   row1 c1 c2
   row2 c1 c2
   row3 c1 c2
   row4 c1 c2
   row5 c1 c2
   row6 c1 c2
   row7 c1 c2
   row8 c1 c2
   row9 c1 c2
   row10 c1 c2
   ```

2. Make the data available on the cluster.

   ```
   hadoop fs -put $filename /tmp/
   ```

   For example:

   ```
   hadoop fs -put data.tsv /tmp/
   ```

3. Define the HBase schema for the data. Continuing with the `data.tsv` example, create a script file called `simple.ddl`, which contains the HBase schema for `data.tsv`:

```
CREATE TABLE simple_hcat_load_table (id STRING, c1 STRING, c2 STRING)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ( 'hbase.columns.mapping' = 'd:c1,d:c2' )
TBLPROPERTIES ( 'hbase.table.name' = 'simple_hcat_load_table'
);
```

4. Create and register the HBase table in HCatalog.

```
hcat -f $HBase_Table_Name
```

   The following HCatalog command-line command runs the DDL script `simple.ddl`:

```
hcat -f simple.ddl
```

5. Create the import file.

   The following example instructs Pig to load data from `data.tsv` and store it in `simple_hcat_load_table`. For the purposes of this example, assume that you have saved the following statement in a file named `simple.bulkload.pig`.

```
A = LOAD 'hdfs:///tmp/data.tsv' USING PigStorage('\t') AS (id:chararray,
 c1:chararray,
c2:chararray);
-- DUMP A;
STORE A INTO 'simple_hcat_load_table' USING org.apache.hive.hcatalog.pig.
HCatStorer();
```

Note

Modify the filenames and table schema for your environment.

6. Use Pig to populate the HBase table via HCatalog `bulkload`.

   Continuing with the example, execute the following command on your HBase Server machine:

```
pig -useHCatalog simple.bulkload.pig
```