Hortonworks Connector for Teradata

(Feb 14, 2014)

docs.hortonworks.com

Hortonworks Connector for Teradata

Copyright © 2014 Hortonworks, Inc. All rights reserved.

The software documented herein is Copyright (c) 2012-2014 Hortonworks, Inc., all rights reserved. It is distributed under the Hortonworks End User License Agreement (EULA), which you should read and agree to before using the software. You may not use this software except in compliance with the Hortonworks EULA.

Unless required by applicable law or agreed to in writing, software distributed under the EULA is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the EULA for the specific language governing permissions and limitations under the License.

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. Hortonworks Connector for Teradata

Introduction

Hortonworks Connector for Teradata (powered by the Teradata Connector for Hadoop) is an implementation of a Sqoop connector that enables those conversant with the Apache Sqoop tool to transfer data between the Teradata MPP DBMS and Apache Hadoop environments.

Background

Sqoop provides facilities for bulk transfer of data between external data stores and the Hadoop environment exploiting the Map Reduce paradigm. Sqoop depends on JDBC interfaces to access the external databases.

Most of the databases also have specialized access methods for high speed bulk data transfers for efficient batch processing needs, such as backups, etc.

To accommodate the varieties of database mechanisms to facilitate bulk transfer, Sqoop provides extensible base implementations of the data transfer functions utilizing the JDBC interface that can optionally be enhanced to suit a database-specific method of data transfer.

Terminology

Sqoop has the notion of *Connectors,* which contain the specialized logic to read and write to external systems.

- The *Hortonworks Connector for Teradata* ("Hortonworks Connector") is a Sqoop Connector implementation for Teradata.
- It is built on the *Teradata Connector for Hadoop,* a Teradata product.

Supported Features

The Hortonworks Connector has some features that are not directly supported by the Sqoop product; you must use the Hortonworks Connector mechanism for such features by specifying additional `-D` command line options.

For example to use ORFiles, RCFiles or HCatalog features, a command line option exposed by the Hortonworks Connector is needed to enable the features. The command line option for RCFiles is `-Dteradata.db.input.file.format=rcfile` (see Configuration Options for more information).

The Hortonworks Connector supports the following features:

- Import/Export tools that run Hadoop MR jobs to transfer data.
- Support for Text, Sequence, ORCFiles, and RCFiles as the source for export operations and target for import operations.
- Import table or query data from Teradata to:

- an existing partitioned or non-partitioned Hive table.
- a new partitioned or non-partitioned Hive table created by the connector.
- an HCatalog table.

- Export data from HDFS files, Hive or HCatalog tables to empty or non-empty Teradata tables.
- Facilities for mapping schemas between Teradata and Hive/HCatalog, including necessary data type conversions.

Connector Feature Checklist

Import all tables: Supported.

Incremental import: Sqoop options are not supported but can be emulated, as specified in the sample invocation Incremental Import.

BLOB and CLOB: Limited to 64 KB.

Import data to Sqoop

- **TextFormat, delimited:** Supported.
- **SequenceFile:** Supported.
- **RCFile:** Supported.

Hive arguments: Support for all standard Hive arguments. All data types except Union are supported.

Export from / import to HCatalog table: Supported.

Automatic schema mapping to/from HCatalog: Supported.

Import using a query: Supported.

Update table: Not supported.

Compression: Not supported.

Software Versions and Installation

Connector Version

This document discusses the Hortonworks Connector for Teradata ("Hortonworks Connector") built on version 1.2.0 of the Teradata Connector for Hadoop.

Supported Product Versions

This section lists the product versions supported in the current release of the Hortonworks Connector.

Teradata Database Versions

The following Teradata database versions are supported:

- Teradata Database 13.00
- Teradata Database 13.10
- Teradata Database 14.00
- Teradata Database 14.10

Hive Version

- Hive 0.11

Hadoop Version

- HDP 1.3.2 (prior versions are untested, but may also work)

Sqoop Versions

- Sqoop 1.4.3

Requirements and Dependencies

System Requirements

The Hortonworks Connector requires JRE/JDK 1.6 or later versions.

Dependencies

1. Teradata GSS Client Driver 14.10 or later versions (tdgssconfig)
2. Teradata JDBC Driver 14.10 or later versions (terajdbc)
3. Teradata Connector for Hadoop 1.2.0

Installation

Installation Dependencies

Sqoop must be installed first.

Installing the Software

1. Download the tarball from the "Add-Ons" for Hortonworks Data Platform 1.3 here: http://hortonworks.com/download.

2. Extract the contents of the tar archive to $SQOOP_HOME/lib. Sqoop will then distribute the contents of the tar to the necessary nodes.

Configuration

This section provides information about connection credentials and configuration options.

Database Connection Credentials

Refer to Sqoop documentation for the Teradata database connection credentials.

Documentation for Sqoop version 1.4.3 is available here: http://sqoop.apache.org/docs/1.4.3/index.html.

Configuration Options

The Hortonworks Connector defines many connector-specific options. A good selection of them is also available as Sqoop options (although not all Sqoop options are directly translatable to Hortonworks Connector options).

Configuration Option Precedence

Options can be specified using any of these techniques:

- a configuration file
- `-D` command line option
- Sqoop options (where applicable)

Therefore the following precedence is established:

1. If any Sqoop command line options are provided which have purposes similar to the Hortonworks Connector options, the Sqoop options have the highest precedence, overriding the `-D` command line options and the configuration file.
2. If `-D` command line options are provided, they override the configuration file values.
3. The value in the configuration file is the default.

As an example, if the configuration file sets the number of input mappers to 4 and the command line option (`-D com.teradata.db.input.num.mappers`) sets it to 5, but the Sqoop option `--num-mappers` is set to 6, then the import job will use 6 mappers.

In some cases, option constraints and the relationships between options affect the configuration value used. For example, import options job.type and file.format are interrelated because the 'hdfs' job type only supports the 'textfile' format, while the 'hcat' and 'hive' job types support file formats 'textfile', 'sequencefile', and 'rcfile'. So when the configuration file sets the job type to HCatalog ('hcat') and that value is not overridden by a command line option or Sqoop option, the file format can be text file, SequenceFile,

or RCfile. But if `-D com.teradata.db.input.job.type` sets the job type to HDFS in the command line and it is not overridden by a Sqoop option, then the file format will be text file regardless of the file.format setting. Finally, if the Sqoop option `--as-sequencefile` is set, then the job type cannot be 'hdfs' so that setting in the command line or configuration file will be overridden to be job type 'hive' (note: not 'hcat' even though the HCatalog job type also supports SequenceFile formats). These options are described in Connector Import Options.

Sqoop Options

The Sqoop option `--connection-manager` must be set as follows to use the Hortonworks Connector for Teradata (see the Sample Invocations):

```
--connection-manager org.apache.sqoop.teradata.TeradataConnManager
```

Some of the Sqoop options are unsupported in the current release of the Hortonworks Connector for Hadoop. See the Appendix for a list of unsupported Sqoop options.

Hortonworks Connector Options

The Appendix describes the Hortonworks Connector options, including Connector Import Options and Connector Export Options.

Data Type Support

The Hortonworks Connector data types depend on Teradata database types.

Support for Teradata Data Types

BIGINT	TIME (n)	INTERVAL HOUR (n) TO SECOND (m)
BYTEINT	TIMESTAMP (n)	INTERVAL MINUTE (n)
INTEGER	PERIOD (DATE)	INTERVAL MINUTE (n) TO SECOND (m)
SMALLINT	PERIOD (TIME (n))	INTERVAL SECOND (n)
DOUBLE PRECISION	PERIOD (TIMESTAMP (n))	*The following data types are supported with some limitations:*
FLOAT	INTERVAL YEAR (n)	• BYTE (n) [1]
REAL	INTERVAL YEAR (n) TO MONTH	• VARBYTE (n) [1]
DECIMAL (n,m)	INTERVAL MONTH (n)	• BLOB [1][2]
NUMERIC (n,m)	INTERVAL DAY (n)	• CLOB [2]
NUMBER (n,m)	INTERVAL DAY (n) TO HOUR	• ARRAY [3]
CHAR (n)	INTERVAL DAY (n) TO MINUTE	[1] *Converted to HEX string*
VARCHAR (n)	INTERVAL DAY (n) TO SECOND (m)	[2] *BLOB and CLOB datatypes are limited to 64 KB in length*
LONG VARCHAR	INTERVAL HOUR (n)	[3] *Can be used for InputFormat only*
DATE	INTERVAL HOUR (n) TO MINUTE	

Support for Hive Data Types

BIGINT INT SMALLINT TINYINT DOUBLE FLOAT STRING BOOLEAN	*The following Hive types are supported with some limitations:* • BINARY {A} • MAP {B} • ARRAY {B} • STRUCT {B} • TIMESTAMP {C} {A} *Supported with Hive 0.10.0 or later* {B} *Converted to/from VARCHAR in JSON format on the Teradata system* {C} *Custom formats are not supported*

Unsupported Data Types

These Teradata types are unsupported:	*This Hive type is unsupported:*
• GRAPHIC • VARGRAPHIC • LONG VARGRAPHIC	• UNION

Hive and HCatalog Support

Importing from Hive and HCatalog requires that HADOOP_CLASSPATH and LIB_JARS be specified before the **sqoop** command is run. This shows the environment variable setup:

```
export HADOOP_CLASSPATH=$(hcat -classpath)
HIVE_HOME=/usr/lib/hive
HCAT_HOME=/usr/lib/hcatalog

export LIB_JARS=$HCAT_HOME/share/hcatalog/hcatalog-core-0.12.0.2.0.6.0-76.jar,
\
                            $HIVE_HOME/lib/hive-metastore-0.12.0.2.0.6.0-76.
jar,\
                            $HIVE_HOME/lib/libthrift-0.9.0.jar,\
                            $HIVE_HOME/lib/hive-exec-0.12.0.2.0.6.0-76.jar,\
                            $HIVE_HOME/lib/libfb303-0.9.0.jar,\
                            $HIVE_HOME/lib/jdo2-api-3.0.1.jar,\
                            $HIVE_HOME/lib/slf4j-api-1.7.5.jar,\
                            $HIVE_HOME/lib/hive-cli-0.12.0.2.0.6.0-76.jar,\
                            $HIVE_HOME/lib/hive-builtins-0.12.0.2.0.6.0-76.jar
```

Note

Change the HIVE_HOME and HCAT_HOME variables as needed and change the versions of the jar to what is available under the directories mentioned.

Hive and HCatalog jobs can be run as shown in the next section.

Sample Invocations

The following examples assume that the SQOOP_HOME environment variable is set to the base directory of the Sqoop installation.

Import Data from Teradata to Hadoop and Hive

```
$SQOOP_HOME/bin/sqoop import \
    -libjars $LIB_JARS \
    -Dteradata.db.input.job.type=hive \
    -Dteradata.db.input.target.table=hive_table \
    -Dteradata.db.input.target.table.schema="emp_no int, birth_date string, \
      first_name string, last_name string, gender string, hire_date string" \
    --connect jdbc:teradata://td-host/Database=dbname \
    --connection-manager org.apache.sqoop.teradata.TeradataConnManager \
    --username tduser \
    --password tduserpass \
    --table tablename
```

Import Data from Teradata into an HCatalog Table

```
$SQOOP_HOME/bin/sqoop import \
    -libjars $LIB_JARS \
    -Dteradata.db.input.job.type=hcat \
    -Dteradata.db.input.target.table=hcat_table \
    --connect jdbc:teradata://td-host/Database=dbname \
    --connection-manager org.apache.sqoop.teradata.TeradataConnManager \
    --username tduser \
    --password tduserpass \
    --table tablename
```

Incremental Import

Teradata incremental import emulates the check-column and last value options. Here is an example for a table which has 'hire_date' as the date column to check against and 'name' as the column that can be used to partition the data.

```
export USER=dbc
export PASS=dbc
export HOST=<dbhost>
export DB=<dbuser>
export TABLE=<dbtable>
export JDBCURL=jdbc:teradata://$HOST/DATABASE=$DB
export IMPORT_DIR=<hdfs-dir to import>
export VERBOSE=--verbose
export MANAGER=org.apache.sqoop.teradata.TeradataConnManager
export CONN_MANAGER="--connection-manager $MANAGER"
export CONNECT="--connect $JDBCURL"
MAPPERS="--num-mappers 4"
DATE="'1990-12-31'"
FORMAT="'yyyy-mm-dd'"
```

```
LASTDATE="cast($DATE as date format $FORMAT)"
SQOOPQUERY="select * from employees where hire_date < $LASTDATE AND \
$CONDITIONS"
$SQOOP_HOME/bin/sqoop import $TDQUERY $TDSPLITBY $INPUTMETHOD $VERBOSE
 $CONN_MANAGER $CONNECT -query "$SQOOPQUERY" --username $USER --password $PASS
 --target-dir $IMPORT_DIR --split-by name
```

Export Data to Teradata

```
$SQOOP_HOME/bin/sqoop export \
    --connect jdbc:teradata://172.16.68.128/Database=employees \
    --connection-manager org.apache.sqoop.teradata.TeradataConnManager \
    --username dbc \
    --password dbc \
    --table employees2 \
    --export-dir /user/hrt_qa/test-sqoop/out \
    --batch
```

Known Issues and Workarounds

Stage Tables

Issue: The export option `--stage-table` does not work.

Cause: The behavior of stage tables is different between Hortonworks Connector and Sqoop, and this causes deadlocks during job cleanup if the Sqoop `-staging-table` option is used.

Workaround: Use the Hortonworks Connector option `teradata.db.output.stage.table.name` for specifying the stage table name.

Fastload

Issue: The export option `'fastload.soclet.host'` does not work.

Cause: The `internal.fastload` method used for Teradata exports can cause resource exhaustion (running out of database AMPs) if the number of reducers exceeds the number of available AMPs.

Workaround: Use the option `teradata.db.output.num.reducers` to restrict the resource usage.

Appendix: Configuration Options

This appendix describes the Hortonworks Connector configuration options and lists the Sqoop options that are currently unsupported.

- Sqoop Options
- Hortonworks Connector Options

Sqoop Options

To use the Hortonworks Connector, you must set the Sqoop option `--connection-manager` to `org.apache.sqoop.teradata.TeradataConnManager` as shown in the Sample Invocations.

Some of the Sqoop options are unsupported in the current release of the Hortonworks Connector for Hadoop. The tables below list the unsupported import and export options.

Note

Imports and exports are defined from the Hadoop perspective, that is, an import brings data into Hadoop from the database and an export moves data out of Hadoop into the database.

Unsupported Sqoop Import Options

Import Category	Unsupported Options

Control Options	`--as-avrodatafile`
	`--append`
	`--compression-codec`
	`--direct`
	`--direct-split-size`
	`--where`
	`--compress, -z`

Incremental Options	`--check-column`
	`--incremental`
	`--last-value`

Input Parsing Options	`--input-enclosed-by`
	`--input-escaped-by`
	`--input-lines-terminated-by`
	`--input-optionally-enclosed-by`

Output Formatting Options	`--mysql-delimiters` `--optionally-enclosed-by`

Hive Support Options	`--hive-delims-replacement`
	`--hive-drop-import-delims`
	`--hive-home`
	`--hive-overwrite`
	`--hive-partition-key`
	`--hive-partition-value`
	`--map-column-hive`

HBase Support Options	`--column-family` `--hbase-create-table`

Import Category	Unsupported Options
	`--hbase-row-key` `--hbase-table`
Data Mapping Options	`--map-column-java`

Unsupported Sqoop Export Options

Export Category	Unsupported Options

Control Options	`--batch`
	`--clear-staging-table`
	`--direct`
	`--update-key`
	`--update-mode`

Input Parsing Options	`--input-lines-terminated-by` `--input-optionally-enclosed-by`

Output Formatting Options	`--fields-terminated-by`
	`--lines-terminated-by`
	`--mysql-delimiters`
	`--optionally-enclosed-by`

Data Mapping Options	`--input-null-non-string`
	`--input-null-string`
	`--map-column-java`
	`--null-non-string`
	`--null-string`

Hortonworks Connector Options

This section describes configuration options provided by the Hortonworks Connector.

- Connector Import Options
- Connector Export Options

For information about how the options can be specified, see Configuration Option Precedence.

Note

Imports and exports are defined from the Hadoop perspective, that is, an import brings data into Hadoop from the database and an export moves data out of Hadoop into the database.

Connector Import Options

All option names below are prefixed by "**teradata.db.input.**" when specified in the configuration files or in the `-D` command line option.

For example, the `job.type` option is specified as `teradata.db.input.job.type`.

Connector Import Option (`teradata.db.input.*`)	Description	Overrriding Sqoop Option
num.partitions.in.staging	The number of partitions in the staging table generated for split.by.partition import job. If the number of mappers is larger than the number of partitions in staging, the value of number of mappers will be set to the number of partitions in staging. Required: no Supported values: an integer greater than 0 Default value:	N/A
job.type	The type of import job. Required: no Supported values: hcat, hive, hdfs Default value: hdfs	None for 'hcat' and 'hive' settings; also none for 'hdfs' when the file format is 'textfile'. But for file formats other than 'textfile' the 'hdfs' job type is reset to 'hive', therefore the following Sqoop option overrides a job.type of 'hdfs': `--as-sequencefile`

file.format	The format of a to-be-imported data file in HDFS. An 'hcat' or 'hive' job type supports 'orcfile', 'rcfile', 'sequencefile', and 'textfile' file formats; and an 'hdfs' job type supports only 'textfile' format.
	
	Required: no
	
	Supported values: rcfile, sequencefile, textfile
	
	Default value: textfile
		`--as-sequencefile`
		
		`--as-textfile`

target.paths	The directory with which to place the imported data. It is required for an 'hdfs' job, optional for a 'hive' job, and not valid for an 'hcat' job. For a 'hive' job, either specify this or the 'target.table' parameter but not both.
	
	Required: no
	
	Supported values: string
	
	Default value: The value of property 'mapred.output.dir'
		`--target-dir`
		
		`--warehouse-dir`

method	The method that the Hortonworks Connector uses to import data from a Teradata system. Required: no Supported values: split.by.hash, split.by.partition, split.by.value Default value: split.by.hash	

num.mappers	The number of mappers for the import job. It is also the number of splits the Hortonworks Connector will attempt to create.
	
	Required: no
	
	Supported values: an integer greater than 0
	
	Default value: 2
		`-m`
		
		`--num-mappers`

Connector Import Option (`teradata.db.input.*`)	Description	Overrriding Sqoop Option
source.query	The SQL query to select data from a Teradata database; either specify this or the 'source.table' parameter, but not both. Required: no Supported values: The select SQL query (Teradata database supported)	`--query`
source.count.query	A placeholder, not currently used.	N/A
source.table	The name of the source table in a Teradata system from which the data is imported. Either specify this or the 'source.query' parameter, but not both. Required: no Supported values: string	`--table`
source.field.names	The names of columns to import from the source table in a Teradata system, in comma-separated format. The order of the source field names must match exactly the order of the target field names for schema mapping. This parameter must be present when the 'target.field.names' parameter is specified. If not specified, then all columns from the source table will be retrieved. Required: no Supported values: string	`--columns`
target.database	The name of the target database in Hive or HCatalog. It is optional with a 'hive' or 'hcat' job and not valid with an 'hdfs' job. Required: no Supported values: string Default value: default	N/A
target.table	The name of the target table in Hive or HCatalog. It is required with an 'hcat' job, optional with a 'hive' job, and not valid with an 'hdfs' job. For a 'hive' job, either specify this parameter or the 'target.paths' parameter, but not both. Required: no Supported values: string	`--hive-table`
target.table.schema	The column schema of the target table, including the partition schema, in comma-separated format. Required: no Supported values: string	
target.partition.schema	The partition schema of the target table in Hive or HCatalog, in comma-separated format. This parameter is applicable with 'hive' job only, and 'target.table.schema' must be specified with it. Required: no Supported values: string	N/A

Connector Import Option (`teradata.db.input.*`)	Description	Overrriding Sqoop Option
target.field.names	The names of fields to write to the target file in HDFS, or to the target Hive or HCatalog table, in comma separated format. The order of the target field names must match exactly the order of the source field names for schema mapping. This parameter must be provided when the 'source.field.names' parameter is specified. Required: no Supported values: string	Driven by the imported columns
batch.size	The number of rows a Hortonworks Connector fetches each time from the Teradata system, up to a 1 MB buffer size limit. Required: no Supported values: an integer greater than 0 Default value: 10000	`--fetch-size`
separator	The field separator to use with the imported files. This parameter is only applicable with the 'textfile' file format. Required: no Supported values: string Default value: \t	`--fields-terminated-by`
split.by.column	The name of a table column to be used for splitting import tasks. It is optional with the 'split.by.hash' and 'split.by.value' methods, and not valid with the 'split.by.partition' method. If this parameter is not specified, the first column of the table's primary key or primary index will be used. Required: no Supported values: a valid table column name	`--split-by`
stage.force	If set to true, then staging is used even if the source table is a PPI table. It is valid with the 'split.by.partition' method only. Required: no Supported values: true, false Default value: false	N/A
stage.database	The database which the Hortonworks Connector uses to create a staging table. Required: no Supported values: the name of a database in the Teradata system Default value: the current logon database in the JDBC connection	N/A
stage.table.name	The name which Hortonworks Connector uses to create a staging table in the Teradata system, if staging is required. Its length cannot exceed 20 characters. It should be used when the	N/A

Connector Import Option (`teradata.db.input.*`)	Description	Overrriding Sqoop Option
	source Teradata table has a name exceeding 24 characters. Required: no Supported values: string	
teradata.db.hive.configuration.file	The path to the Hive configuration file in the HDFS. It is required for a 'hive' or 'hcat' job launched through remote execution or on data nodes. Required: no Supported values: string	

Connector Export Options

All option names below are prefixed by "**teradata.db.output.**" when specified in the configuration files or in the `-D` command line option.

For example, `target.table` is specified as `teradata.db.output.target.table`.

Connector Export Option (`teradata.db.output.*`)	Description	Overrriding Sqoop Option
num.partitions.in.staging	The number of partitions in the staging table generated for split.by.partition import job. If the number of mappers is larger than the number of partitions in staging, the value of number of mappers will be set to the number of partitions in staging. Required: no Supported values: an integer greater than 0 Default value:	N/A
target.table	The name of the target table in a Teradata system. Required: yes Supported values: string	`--table`
job.type	The type of export job. Required: no Supported values: hcat, hive, hdfs Default value: hdfs	N/A
file.format	The format of a to-be exported data file in HDFS. An 'hcat' or 'hive' job type supports 'rcfile', 'sequencefile', and 'textfile' formats; and an 'hdfs' job type supports only 'textfile' format. Required: no Supported values: rcfile, sequencefile, textfile Default value: textfile	N/A

Connector Export Option (`teradata.db.output.*`)	**Description**	**Overrriding Sqoop Option**
source.paths	The directory of to-be exported source files in HDFS. It is required for an 'hdfs' job, optional with a 'hive' job, and not valid with an 'hcat' job. For a 'hive' job, either specify this or the 'source.table' parameter but not both. Required: no Supported values: string	`--export-dir`
method	The method that the Hortonworks Connector uses to export data to a Teradata system. Required: no Supported values: batch.insert, multiple.fastload, internal.fastload Default value: batch.insert	N/A

num.mappers	The maximum number of output mapper tasks. If the value is zero, then the number of mappers will be the same as the number of file blocks in HDFS. Use either this parameter or 'num.reducers', but not both.
	
	Required: no
	
	Supported values: an integer greater than or equal to zero
	
	Default value: 2
		`-m`
		
		`--num-mappers`

num.reducers	The maximum number of output reducer tasks if export is done in the reduce phase. Use either this parameter or 'num.mappers', but not both. Required: no Supported values: an integer greater than or equal to zero Default value: 0	N/A
source.table	The name of the source table in Hive or HCatalog from which the data is exported. It is required for an 'hcat' job, optional with a 'hive' job, and not valid with an 'hdfs' job. For a 'hive' job, either specify this or the 'source.paths' parameter but not both. Required: no Supported values: string	N/A
source.table.schema	The full column schema of the source table in Hive or HCatalog, not including the partition schema if the table has any, in comma-separated format. When using the data mapping feature with an 'hdfs' job operating on a text file, use this parameter to describe the file content schema.	N/A

Connector Export Option (`teradata.db.output.*`)	Description	Overrriding Sqoop Option
	Required: no Supported values: string	
source.partition.schema	The full partition schema of the source table in Hive, in comma-separated format. It is valid with a 'hive' job only. When this parameter is used, the 'source.table.schema' parameter must also be specified. Required: no Supported values: string	N/A
source.field.names	The names of fields to export from the source HDFS files, or from the source Hive and HCatalog tables, in comma-separated format. The order of the source field names must match the order of the target field names for schema mapping. This parameter must be provided when the 'target.field.names' parameter is specified. Required: no Supported values: string	N/A
target.field.names	The names of fields to export to the target table in a Teradata system, in comma-separated format. The order of the target field names must match the order of the source field names for schema mapping. This parameter must be provided when the 'source.field.names' parameter is specified. Required: no Supported values: string	`--columns`
target.field.count	The number of fields to export to the target table in the Teradata system. Either specify this or the 'target.field.names' parameter, but not both. Required: no Supported values: integer Default value: 0	N/A
fastload.socket.host	The job client host name or IP address that fastload tasks communicate with to synchronize states. This parameter is valid with the 'internal.fastload' method only. If this parameter is not specified, the Hortonworks Connector will automatically lookup for the node that the job is launched on and the configuration values of the 'dfs.datanode.dns.interface' parameter or the 'mapred.tasktracker.dns.interface' parameter, if these are configured. Otherwise, the Hortonworks Connector will select the IP address of the node's first network interface.	N/A

Connector Export Option (`teradata.db.output.*`)	Description	Overrriding Sqoop Option
	Required: no Supported values: resolveable host name or IP address	
fastload.socket.port	The host port that fastload tasks will communicate with to synchronize states. This parameter is valid with the 'internal.fastload' method only. If this parameter is not specified, the Hortonworks Connector will automatically select an available port starting from 8678. Required: no Supported values: integer	N/A
batch.size	The number of rows the Hortonworks Connector will send each time to the Teradata system. Required: no Supported values: integer Default value: 10000	N/A
separator	The separator of fields in the source to-be-exported files. This parameter is only valid with 'textfile' file format. Required: no Supported values: string Default value: \t	`--input-fields-terminated-by`
stage.database	The database which the Hortonworks Connector uses to create staging tables. Required: no Supported values: the name of a database in the Teradata system Default value: the current logon database in the JDBC connection	N/A
stage.table.name	The name which the Hortonworks Connector uses to create a staging table in the Teradata system, if staging is required. Its length cannot exceed 20 characters. It should be used when the target Teradata table has a name exceeding 24 characters. Required: no Supported values: string less than 17 characters	
teradata.db.hive.configuration.file	The path to the Hive configuration file in the HDFS. It is required for a 'hive' or 'hcat' job launched through remote execution or on data nodes. Required: no Supported values: string	N/A