..

Teradata Connectors User Guide

Date published: 2017-08-24
Date modified: 2025-05-08

https://docs.cloudera.com/

Legal Notice

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Cloudera Connector Powered by Teradata User Guide

Cloudera Connector Powered by Teradata provides high-speed data transfer between Teradata and Cloudera.

This connector allows various Sqoop tools, such as sqoop-import and sqoop-export, to operate in highly efficient direct modes, and expose options that are specific to Teradata.

This guide describes how to install, configure, and use the connector in a Sqoop 1 installation and provides reference information for connector operation. This guide is intended for:

- System and application programmers
- System administrators
- Database administrators
- Data analysts
- Data engineers

Version Scheme for Teradata Connector

This topic describes the versioning scheme used for Cloudera Connector Powered by Teradata. The version string consists of the following parts:

$MAJOR_VERSION.$MINOR_VERSIONc$MAJOR_CDH_VERSION

- MAJOR VERSION and MINOR VERSION: Identify the major and minor version of the connector project.
- MAJOR CDH VERSION: The major Cloudera version for which the connector has been compiled and tested.

For example:

- 1.7.c6 - Seventh revision of a Sqoop 1-based connector that is compatible with CDP 6.

Cloudera Connector Powered by Teradata Release Notes

This section summarizes the high level changes and most important new features in the Cloudera Connectors for Teradata.

New Features in Cloudera Connector Powered by Teradata

The following new features are included in Cloudera Connector Powered by Teradata.

CDP compatible version of Cloudera Connector Powered by Teradata version 1.8.5.1c7p1 and the TDCH library to version 1.8.5.1

The versions 1.8.5.1c7p1 and 1.8.5.1c7 are identical in functionality. However, 1.8.5.1c7p1 can be installed on Ubuntu 22 and Ubuntu 24 via Cloudera Manager, whereas 1.8.5.1c7 cannot.

CDP compatible version of Cloudera Connector Powered by Teradata version 1.8.5.1c7 and the TDCH library to version 1.8.5.1

Cloudera Connector Powered by Teradata includes ORC support in the Sqoop-Connector-Teradata component. In this release, you can use Teradata Manager to import data from Teradata server to Hive in ORC format.

To leverage this feature, it is essential to have a compatible version of Sqoop. Ensure that the Sqoop installation meets the following minimum requirements:

- CDP Private Cloud Base 7.1.9 and later

Extended Cloudera Data Platform (CDP) runtime compatibility also includes

- CDP Public Cloud 7.2.9 and later

- CDP Private Cloud Base 7.1.7 and later

Upgraded TDCH version and Teradata driver

The connector has been upgraded to use TDCH version 1.8.5.1 and Teradata driver version 20.00.00.10. This update ensures better performance, compatibility, and includes bug fixes in the connector.

Supported commands for ORC imports

Here are some examples of supported commands that utilize the Cloudera Connector Powered by Teradata to import data to Hive in ORC format:

Example 1: Import without providing Teradata driver and connection details to TeradataManager

```
/opt/cloudera/parcels/CDH/bin/sqoop import \
--connect ... \
--username ... \
--password ... \
--table employees \
--warehouse-dir "..." \
--hive-import \
--delete-target-dir \
--hive-overwrite \
--as-orcfile \
--external-table-dir "..." \
--hs2-url "..." \
-m 1
```

Example 2: Import without providing Teradata driver and by providing the corresponding TeradataManager connection details

```
/opt/cloudera/parcels/CDH/bin/sqoop import \
--connect ... \
--username ... \
--password ... \
--table employees \
--warehouse-dir "..." \
--hive-import \
--delete-target-dir \
--hive-overwrite \
--as-orcfile \
--external-table-dir "..." \
--hs2-url "..." \
--connection-manager
 com.cloudera.connector.teradata.TeradataManager \
-m 1
```

Note: The examples provided in this section should be updated with the actual connection details and configuration specific to the corresponding environment.

Supported combinations of --driver and --connection-manager parameters

The compatibility matrix for driver and connection manager parameters remains unchanged and is the same as in the Sqoop-Teradata-Connector 1.8.3.2c7p1 release.

Other changes introduced through the TDCH upgrade to 1.8.5.1

The following features are added since the TDCH 1.8.3.2 version:

Features included in the 1.8.5.1 release

- TDCH-2005: Update Teradata JDBC driver to 20.00.00.10
- TDCH-2004: Certify TDCH 1.8.x on CDP 7.1.7 SP2
- TDCH-1994: Add support for using both -targetpaths and -targettable for Hive import jobs

- TDCH-2020: Discontinue using internal undocumented interfaces in TDJDBC

Features included in the 1.8.4.1 release

- TDCH-1989: Certify TDCH on CDP 7.1.8
- TDCH-1976: Fix Black Duck Security Issues
- TDCH-1993: Add support for custom staging directory instead of the default /user/<username>/<temp_directory> location
- TDCH-1962: Handling HASHAMP range of SQL query when AMP goes down in DBS
- TDCH-1998: Update Teradata JDBC driver to 17.20.00.12
- TDCH-1997: Include OSS License file (.pdf) in the rpm installation

CDP compatible version of Cloudera Connector Powered by Teradata Version 1.8.3.2c7p1

The changes introduced in this release do not contain a new version of TDCH library, Teradata driver, or any additional required changes from Sqoop; therefore, the CDP compatibility is the same as in Cloudera Connector Powered by Teradata version 1.8.3.2.c7.

Extended Cloudera Data Platform (CDP) runtime compatibility also includes

- CDP Public Cloud 7.2.9 and later
- CDP Private Cloud Base 7.1.7 and later

Parquet support on CDP versions for HDFS or Hive imports

Some changes made earlier between TDCH 1.8.3.1 and 1.8.3.2 addressed an issue related to Hive API breakage, which made the latest Sqoop Teradata Connector 1.8.3.2c7 incompatible with TDCH 1.8.3.2 resulting in a bug in the Hive import process in Parquet file format.

This release focuses on resolving this incompatibility issue, so that the Teradata Manager can be used to run Hive imports in Parquet file format successfully as it was supposed to in Sqoop Teradata Connector 1.8.3.2c7 release.

Additional step required for importing Teradata to Hive in Parquet format

The --hs2-url argument must be provided explicitly as a Sqoop argument to support Hive JDBC connection with HiveServer2 (HS2) through TDCH.

Configuring user/password based authentication

From this release onwards, you can configure user/password based authentication (like LDAP) when connecting to Hive using Teradata Manager. For this, you must provide the required credentials either in the --hs2-url argument or explicitly using the --hs2-user and --hive-password Sqoop arguments.

Supported commands for Parquet imports

You can use the Parquet feature under the following conditions:

- You can import from Teradata to Hive in Parquet format using one of the following commands:

```
sqoop import --connect “jdbc:teradata://host/database” --con
nection-manager org.apache.sqoop.manager.GenericJdbcManager --
driver com.teradata.jdbc.TeraDriver --table table1 --target-di
r hdfs://ns1/tmp/table1 --hive-import --as-parquetfile
```

```
sqoop import --connect “jdbc:teradata://host/database” --con
nection-manager com.cloudera.connector.teradata.TeradataMana
ger --table table1 --target-dir hdfs://nsq/tmp/table1 --hive-
import --as-parquetfile --hs2-url “jdbc:hive2://…"
```

```
sqoop import --connect “jdbc:teradata://host/database” --con
nection-manager com.cloudera.connector.teradata.TeradataMana
ger --table table1 --target-dir hdfs://nsq/tmp/table1 --hive
```

```
-import --as-parquetfile --hs2-url “jdbc:hive2://…;user=foo;
password=bar”
```

```
sqoop import --connect “jdbc:teradata://host/database” --con
nection-manager com.cloudera.connector.teradata.TeradataMana
ger --table table1 --target-dir hdfs://nsq/tmp/table1 --hive-
import --as-parquetfile --hs2-url “jdbc:hive2://…” --hs2-user
 foo --hive-password bar
```

- You can import from Teradata to HDFS in Parquet format using only Generic JDBC connection manager and with the following options:

  ```
  sqoop import --connect “jdbc:teradata://host/database” --
  connection-manager org.apache.sqoop.manager.GenericJdbcManager
   --driver com.teradata.jdbc.TeraDriver --table table1 --targ
  et-dir hdfs://ns1/tmp/table1 --as-parquetfile
  ```

  Any version of the Sqoop Teradata connector supports this command.

Supported combinations of --driver and --connection-manager parameters

The driver and connection manager compatibility matrix has not changed since the previous release and is same as in the Sqoop Teradata Connector 1.8.3.2c7 version.

CDP compatible version of Cloudera Connector Powered by Teradata version 1.8.3.2c7 and the TDCH library to version 1.8.3.2

Cloudera Connector Powered by Teradata implements Parquet support in the Sqoop-Connector-Teradata component. In this release, you can use TeradataManager to import Parquet files.

- CDP Private Cloud 7.1.8 and later compatibility

  If you install this connector version on CDP Private Cloud 7.1.8, you can import data from the Teradata server to Hive in Parquet using Teradata Manager.
- CDP Public Cloud 7.2.13 and later compatibility

  If you install this connector version on CDP Public Cloud 7.2.13, you can import data from the Teradata server to Hive in Parquet format using Teradata Manager.

Extended Cloudera Data Platform (CDP) runtime compatibility also includes

- CDP Public Cloud 7.2.9 and later
- CDP Private Cloud Base 7.1.7 and later

Parquet support on CDP versions for HDFS or Hive imports

You can use the Parquet feature to import data from the Teradata server to HDFS or Hive in Parquet format using GenericJdbcManager with the Teradata JDBC driver under the following conditions:

- Sqoop Teradata Connector 1.8.1c7 (older connector) or 1.8.3.2c7 (latest connector) is installed on one the following CDP versions:
  - CDP Public Cloud 7.2.9 - 7.2.12 (earlier CDP Public Cloud version)
  - CDP Private Cloud Base 7.1.7 (earlier CDP Private Cloud Base version)
- Sqoop Teradata Connector 1.8.1c7 (earlier connector) is installed on one of the following CDP versions:
  - CDP Public Cloud 7.2.13 (latest Public Cloud version) and later compatibility
  - CDP Private Cloud 7.1.8 (latest Private Cloud Base version) and later compatibility

As shown above, the latest connector is backward compatible for use on earlier CDP versions and the earlier connector is forward-compatible for use on the later CDP versions.

You must use the supported combinations of --driver and --connection-manager parameters shown below in "Supported combinations of --driver and --connection-manager parameters".

Supported commands for Parquet imports

You can use the Parquet feature under the following conditions:

- You can import from Teradata to Hive in Parquet format using one of the following commands:

  ```
  sqoop import --connect "jdbc:teradata://host/database" --connection-manager org.apache.sqoop.manager.GenericJdbcManager --driver com.teradata.jdbc.TeraDriver --table table1 --target-dir hdfs://ns1/tmp/table1 --hive-import --as-parquetfile
  ```

  ```
  sqoop import --connect "jdbc:teradata://host/database" --connection-manager com.cloudera.connector.teradata.TeradataManager --table table1 --target-dir hdfs://nsq/tmp/table1 --hive-import --as-parquetfile
  ```

- You can import from Teradata to HDFS in Parquet format using only the following options:

  ```
  sqoop import --connect "jdbc:teradata://host/database" --connection-manager org.apache.sqoop.manager.GenericJdbcManager --driver com.teradata.jdbc.TeraDriver --table table1 --target-dir hdfs://ns1/tmp/table1 --as-parquetfile
  ```

  Any version of the Sqoop Teradata connector supports this command.

Supported combinations of --driver and --connection-manager parameters

The following table describes supported combinations of parameters when importing Parquet data from Teradata to HDFS:

--driver	**--connection-manager**
-	-
com.teradata.jdbc.TeraDriver	-
com.teradata.jdbc.TeraDriver	org.apache.sqoop.manager.GenericJdbcManager

The following table describes supported combinations when importing Parquet data from Teradata to Hive:

--driver	**--connection-manager**
-	-
com.teradata.jdbc.TeraDriver	-
com.teradata.jdbc.TeraDriver	org.apache.sqoop.manager.GenericJdbcManager
-	com.cloudera.connector.teradata.TeradataManager

Other features

The following features have been added to the 1.8.3.2 version:

TDCH-1972

Certify TDCH on CDP 7.1.7 SP1 and add support for Hive JDBC with HiveServer2.

The following features are included in the 1.8.3.1 version:

TDCH-1919

TDCH support for Kerberos enabled Advanced SQL Engine (TDBMS)

TDCH-1921

Add more debug statements for "split.by.hash"

TDCH-1922

Add more debug statements for "split.by.value"

TDCH-1923

Add more debug statements for "split.by.partition"

TDCH-1924

Add more debug statements for "split.by.amp"

TDCH-1925

Add more debug statements for "batch.insert"

TDCH-1950

Certify TDCH with TDJDBC 17.10

The following features are included in the 1.8.2 release:

TDCH-1571

Add Timestamp Support for Parquet in TDCH

TDCH-1858

Certify TDCH with Advanced SQL Engine (TDBMS) 17.10

TDCH-1892

Adding more debug statements for fastload and fastexport methods for better debugging

TDCH-1897

Display the error at exact CLI option instead of generic message

The following changes in this version are related to the new Teradata JDBC connector incorporated into the Sqoop Teradata Connector 1.8.3.2c7:

- Supported Teradata Database versions
  - Teradata Database 16.00Teradata Database 16.10
  - Teradata Database 16.20
  - Teradata Database 17.00
  - Teradata Database 17.05
  - Teradata Database 17.10
- Supported Hadoop versions
  - Hadoop 3.1.1
- Supported Hive versions
  - Hive 3.1.1
  - Hive 3.1.3
- Certified Hadoop distributions
  - Cloudera Data Platform (CDP) Private Cloud Base (CDP Datacenter) 7.1.7
- Supported Teradata Wallet versions
  - Teradata Wallet 16.20 - since TD Wallet supports multiple versions installed on the system, TD Wallet 16.20 must be installed to use TD Wallet functionality.

CDP compatible version of Cloudera Connector Powered by Teradata Version 1.8.1c7 and the TDCH library to version 1.8.1

Cloudera Connector Powered by Teradata includes the following new features:

- Extended Cloudera Data Platform (CDP) compatibility
  - CDP Public Cloud 7.2.9 and later
  - CDP Private Cloud Base 7.1.7 and later

CDP compatible version of Cloudera Connector Powered by Teradata Version 1.8c7 and the TDCH library to version 1.8.0

Cloudera Connector Powered by Teradata includes the following new features:

- Extended Cloudera Data Platform (CDP) compatibility
  - CDP Public Cloud 7.2.0 - 7.2.8
  - CDP Private Cloud Base 7.1.0 - 7.1.6
- Support for sqoop import options --incremental lastmodified and --last-value

Note: Cloudera Connector Powered by Teradata 1.8c7 and is not compatible with CDH 6.

CDH 6 compatible version of Cloudera Connector Powered by Teradata 1.7.1c6 Available

Cloudera Connector Powered by Teradata 1.7.1c6 is compatible with CDH 6. It does not contain new features or changes.

Note: Cloudera Connector Powered by Teradata 1.7c5 and lower are not compatible with CDH 6.

CDH 6 compatible version of Cloudera Connector Powered by Teradata 1.7c6 Available

Cloudera Connector Powered by Teradata 1.7c6 is compatible with CDH 6. It does not contain new features or changes.

Note: Cloudera Connector Powered by Teradata 1.7c5 and lower are not compatible with CDH 6.

New Features in Cloudera Connector Powered by Teradata Version 1.7c5

Cloudera Connector Powered by Teradata now supports Teradata 16.x. This release upgrades the JDBC driver to version 16.10.00.05 and the TDCH library to version 1.5.4.

Cloudera Connector Powered by Teradata now supports importing tables without split-by column specified when the number of mappers is set to 1.

Cloudera Connector Powered by Teradata now supports the internal.fastexport input method. For table import, the following values for the --input-method option are valid:

- split.by.partition
- split.by.hash
- split.by.value
- split.by.amp
- internal.fastexport

Note that the query import still only supports the split.by.partition input method.

The internal.fastexport method implements coordination between the mappers and a coordinator process (running on the edge node where the job was submitted). The host name and the port of this process are automatically resolved, but there are new options introduced for manual configuration:

- --fastexport-socket-hostname: Configures the host of the coordinator process. It sets the tdch.input.teradata.fastexport.coordinator.socket.host Java property exposed by the underlying Teradata Connector for Hadoop (TDCH) library.
- --fastexport-socket-port: Configures the port of the coordinator process. It sets the tdch.input.teradata.fastexport.coordinator.socket.port Java property exposed by the underlying Teradata Connector for Hadoop (TDCH) library.

For more information on these properties, see the Teradata Connector for Hadoop tutorial provided by Teradata.

New Features in Cloudera Connector Powered by Teradata Version 1.6.1c5

- Adds support for SLES 12.

New Features in Cloudera Connector Powered by Teradata Version 1.6c5

- Upgrades the JDBC driver to version 15.10.00.22 and the TDCH library to version 1.5.0. These libraries contain several bug fixes and improvements.
- Adds the --schema argument, used to override the <td-instance> value in the connection string of the Sqoop command. For example, if the connection string in the Sqoop command is jdbc:teradata://<td-host>/DATABASE=database1, but you specify --schema database2, your data is imported from database2 and not database1.
  If the connection string does not contain the DATABASE parameter — for example jdbc:teradata://<td-host>/CHARSET=UTF8) — you can also use the --schema database argument to have Sqoop behave as if you specified the jdbc:teradata://<td-host>/DATABASE=databasename,CHARSET=UTF8 connection string.

New Features in Cloudera Connector Powered by Teradata Version 1.5c5

New features:

- Fixed compatibility issue with CDH 5.5.0 and higher.

New Features in Cloudera Connector Powered by Teradata Versions 1.4c5

New features:

- Added support for JDK 8.
- Added --error-database option.
- Added ability to specify format of date, time, and timestamp types when importing into CSV.
- Import method split.by.amp now supports views.
- Upgraded Teradata connector for Hadoop to version 1.3.4.

New Features and Changes in Cloudera Connector Powered by Teradata 1.3c5

New features:

- Upgraded Teradata Connector for Hadoop to version 1.3.3.
- Parcel distribution now contains Teradata JDBC driver; manual download no longer required.
- Added support for query import into Avro file format.

Changes:

- Export method multiple.fastload has been removed.

New Features in Cloudera Connector Powered by Teradata Versions 1.2c5

New features:

- Upgraded Teradata Connector for Hadoop to version 1.2.1.

- Added support for Avro.
- Added support for Incremental import.
- Added support for --where argument.
- Added support for Hive import.
- Added support for Importing all tables using import-all-tables.
- Added support for Query Bands.
- Added new import method split.by.amp (supported only on Teradata 14.10 and higher).

New Features in Cloudera Connector Powered by Teradata Version 1.0.0

This is the first release of this new connector. This connector features:

- Support for secondary indexes.
- Especially fast performance in most cases.

Limitations for Cloudera Connector Powered by Teradata

Learn about the functional limitations of the Cloudera Connector Powered by Teradata.

Cloudera Connector Powered by Teradata has the following limitations:

- Does not support HCatalog.
- Does not support import into HBase.
- Does not support upsert functionality (parameter --update-mode allowinsert).
- Does not support the --boundary-query option.
- Does not support Parquet file format.
- Does not support export to Teradata VIEWs.
- Does not support Kerberos authentication.
- By default speculative execution is disabled for the Teradata Connector. This avoids placing redundant load on the Teradata database.

Known Issues and Workarounds

Learn about the known issues in Cloudera Connector Powered by Teradata, the impact or changes to the functionality, and the workaround.

There are no known issues for customers using the following releases:

- CDP Private Cloud Base 7.1.5 or later
- CDP Public Cloud 7.2.6 or later

Failure to import and export Hive using Sqoop

For customers using earlier CDP releases, Hive imports and exports using Sqoop will fail. To work around this issue, put the Hive common jar in the Sqoop library as follows:

Copy hive-common-<version>.jar from /opt/cloudera/parcels/CDH/jars to /opt/cloudera/parcels/CDH/lib/sqoop/lib.

Getting Support

Cloudera Support provides assistance for the Cloudera Connector for Teradata. You can contact Support by logging a case on the Cloudera Support Portal.

Prerequisites for Teradata Connector

The prerequisites for using the Cloudera Connector Powered by Teradata are:

- To download the connector, you must have an active subscription agreement along with the required authentication credentials (namely, the username and password).

- The authentication credentials are provided in an email sent to the customer account from Cloudera when a new license is issued. If you do not have the authentication credentials, contact your account representative.
- You must have a functioning Cloudera installation, either CDP or CDH, including Sqoop components.
- Depending on how Sqoop is installed, you may need administrative privileges to create or modify configuration files.
- The Teradata connector uses catalog tables and views to look up metadata information. Therefore, the user making the connection must have the SELECT privilege on the DBC database. Check with your Teradata administrators or operators if you are not sure whether you have the required privileges. You must have SELECT privileges on at least one of the following DBC database object types:
    - DBC.columns
    - DBC.databases
    - DBC.tables
    - DBC.indices
- Depending on the input method used, the Teradata connector might need to create temporary tables or temporary views to import data. Check with your Teradata administrators or operators to determine if you have the required privileges.
- The Cloudera Connector Powered by Teradata requires the following additional permissions to use the following *.fastload data methods:
    - DBC.table_levelconstraints
    - DBC.triggers
    - DBC.tvm
    - DBC.dbase
    - DBC.referencingtbls

Cloudera Connector powered by Teradata version 1.8.1c7

- Teradata versions:
    - Teradata Database 15.10
    - Teradata Database 16.00
    - Teradata Database 16.10
    - Teradata Database 16.20
    - Teradata Database 17.00
    - Teradata Database 17.05
- CDP Public Cloud 7.2.9 and later
- CDP Private Cloud 7.1.7 and later

Cloudera Connector powered by Teradata version 1.8c7

- Teradata versions:
    - Teradata Database 15.10
    - Teradata Database 16.00
    - Teradata Database 16.10
    - Teradata Database 16.20
    - Teradata Database 17.00
- CDP Public Cloud 7.2.0 - 7.2.8
- CDP Private Cloud 7.1.0 - 7.1.6

Cloudera Connector powered by Teradata version 1.7c6

- Teradata 13.00 and higher
- CDH 6.0 and higher

- Sqoop 1.4 and higher, but not compatible with Sqoop2

Installing the Teradata Connector

You can install the connector for Teradata using Cloudera Manager if you have a CDH 6 or CDP 7 cluster.

Installation with Cloudera Manager

Prerequisites

In the past, if you manually installed a Teradata connector, you must remove it before installing the Cloudera Connector Powered by Teradata as follows:

1. Go to /var/lib/sqoop and search for "tera".
2. Delete the matching files.
3. Delete the /usr/lib/sqoop/conf directory.

Step 1: Adding the Sqoop Client Gateway

The Sqoop1 Client Gateway sets up and manages the connector configuration for the hosts where you execute Sqoop1 commands. If you do not already have the Sqoop1 Client Gateway deployed in your cluster, deploy it before proceeding.

⚠ **Important:** The Sqoop 1 Client Gateway is required for the Teradata Connector to work correctly. Cloudera recommends installing the Sqoop 1 Client Gateway role on any host used to execute the Sqoop CLI. If you do not already have the Sqoop Client service running on your cluster, see Managing the Sqoop 1 Client for instructions on how to add the service using the Cloudera Manager Admin Console.

Step 2: Download, Distribute, and Activate the Sqoop Parcels

Parcels for Sqoop connectors are prefixed by SQOOP_, followed by the name of the connector.

If you have a CDH cluster, follow the instructions in Managing Parcels to download, distribute, and activate Sqoop parcels. After activating the parcels, you must redeploy the Sqoop client configuration.

If you have a CDP cluster, you can install the connector as a parcel using Cloudera Manager.

To install the Cloudera Connector Powered by Teradata as a parcel:

1. In Cloudera Manager left navigation, click Parcels.
2. Click Parcel Repository & Network Settings.
3. Click Add Another to add a remote parcel repository URL.
4. Copy/paste the parcel URL corresponding to your CDP version from the Connector Compatibility Matrix.
5. Click Save & Verify Configuration.
6. When the URL you added appears with the indicator that it was successfully accessed and the manifest downloaded and validated, click Download > Distribute > Activate.

Connector Compatibility Matrix

The connector Compatiblity Matrix are:

Connector, Cloudera Runtime, and parcel mapping

If you have a CDP cluster, you can install the connector as a parcel using Cloudera Manager. You install the connector that matches your Cloudera Runtime in the list of parcels packed into Cloudera. You also obtain driver information.

The following list shows Teradata parcels packed into Cloudera for each version of the Sqoop Connector for Teradata.

- Connector version 1.8.5.1c7p1

  Parcel URL: https://archive.cloudera.com/p/sqoop-teradata-connector1/1.8.5.1c7p1/parcels/

  TDCH version: 1.8.5.1
- Connector version 1.8.5.1c7

  Parcel URL: https://archive.cloudera.com/p/sqoop-teradata-connector1/1.8.5.1c7/parcels/

  TDCH version: 1.8.5.1
- Connector version 1.8.3.2c7p1

  Parcel URL: https://archive.cloudera.com/p/sqoop-teradata-connector1/1.8.3.2c7p1/parcels/

  TDCH version: 1.8.3.2
- Connector version 1.8.3.2c7

  Parcel URL: https://archive.cloudera.com/p/sqoop-teradata-connector1/1.8.3.2c7/parcels/

  TDCH version: 1.8.3.2
- Connector version 1.8.1c7

  Parcel URL: https://archive.cloudera.com/p/sqoop-teradata-connector1/1.8.1c7/parcels/

  TDCH version: 1.8.1
- Connector version 1.8.0c7

  Parcel URL: https://archive.cloudera.com/p/sqoop-teradata-connector1/1.8.0c7/parcels/

  TDCH version: 1.8.0
- Connector version 1.7.1c6

  Parcel URL: https://archive.cloudera.com/p/sqoop-teradata-connector1/1.7.1c6/parcels/

  TDCH version: 1.5.4
- Connector version 1.7c6

  Parcel URL: https://archive.cloudera.com/p/sqoop-teradata-connector1/1.7c6/parcels/

  TDCH version: 1.5.4

The table below provides the following information for each version of the Sqoop Connector for Teradata:

- Teradata JDBC driver packed into Cloudera
- CDH release compatible with the specific Teradata Connector version
- Teradata server versions, some of which are not supported by all Sqoop Teradata Connector releases

Table 1:

Sqoop Connector for Teradata Version	Teradata JDBC driver version	CDH/CDP Compatibility	Teradata DB compatibility
1.8.5.1c7p1	20.00.00.10	• CDP Public Cloud 7.2.9 and later • CDP Private Cloud Base 7.1.7 and later	16.20, 17.00, 17.10, 17.20, and 20.00
1.8.5.1c7	20.00.00.10	• CDP Public Cloud 7.2.9 and later • CDP Private Cloud Base 7.1.7 and later	16.20, 17.00, 17.10, 17.20, and 20.00
1.8.3.2c7p1	17.10.00.22	• CDP Public Cloud 7.2.9 and later • CDP Private Cloud Base 7.1.7 and later	16.00, 16.10, 16.20, 17.00, 17.05, and 17.10

Sqoop Connector for Teradata Version	Teradata JDBC driver version	CDH/CDP Compatibility	Teradata DB compatibility
1.8.3.2c7	17.10.00.22	• CDP Public Cloud 7.2.9 and later • CDP Private Cloud Base 7.1.7 and later	16.00, 16.10, 16.20, 17.00, 17.05, and 17.10
1.8.1c7	17.00.00.02	• CDP Public Cloud 7.2.9 and later • CDP Private Cloud Base 7.1.7 and later	15.00, 15.10, 16.10, 16.20, and 17.00
1.8.0c7	17.00.00.02	• CDP Public Cloud 7.2.0 - 7.2.8 • CDP Private Cloud Base 7.1.0 - 7.1.6	15.00, 15.10, 16.10, 16.20, and 17.00
1.7.1c6	16.10.00.05	CDH 6	14.00, 14.10, 15.00, 15.10, 16.10, and 16.20
1.7c6	16.10.00.05	CDH 6	14.00, 14.10, 15.00, 15.10, 16.10, and 16.20

Upgrading the Teradata Connector

Use these instructions if you are upgrading one of the connectors to a newer version (for example, if you need to upgrade Cloudera Connector Powered by Teradata from version 1.2c4 to 1.3c4).

Upgrading with CDH 6 and CDP 7

Step 1: Distributing the Sqoop Connector Parcels

1. In the Cloudera Manager Admin Console, click Hosts in the top navigation bar and then go to the Parcels tab. Parcels for the Sqoop connectors are listed on this page, prefixed by "SQOOP_", followed by the name of the connector.
2. Click Download for the connectors you want to install.
3. After the parcels have been downloaded, click Distribute to distribute and unpack the connectors on all hosts in your Hadoop cluster.
4. After the parcels have been distributed, click Activate to make them available to the cluster. Sqoop connectors are listed as Activated on the Parcels page. You must redeploy the client configuration (Step 3) for activation to take effect.

Step 2: Deploying Client Configuration Files

1. In the Cloudera Manager Admin Console, go to the Sqoop Client service.
2. From the Actions menu at the top right of the service page, select Deploy Client Configuration.
3. Click Deploy Client Configuration to confirm redeployment of the client configuration.

Using the Cloudera Connector Powered by Teradata

Connection string format and key commands

After you have installed the connector and copied the JDBC drivers for Teradata to the lib directory of the Sqoop installation, use this connector by invoking Sqoop tools with the appropriate connection string.

The connection string format is jdbc:teradata://<td-host>/DATABASE=<td-instance>:

- <td-host> is the hostname of the machine on which the Teradata server runs.
- <td-instance> is the Teradata database instance name.

For example, the following command invokes the Sqoop import tool with three mappers:

```
$ sqoop import --connect jdbc:teradata://localhost/DATABASE=sqooptest \
--username sqooptest --password xxxxx --table MY_TABLE --num-mappers 3 \
--target-dir /user/sqooptest/MY_TABLE
```

The following command invokes the Sqoop export tool with three mappers:

```
$ sqoop export --connect jdbc:teradata://localhost/DATABASE=sqooptest \
--username sqooptest --password xxxxx --export-dir /user/sqooptest/MY_TABLE
\
--table MY_TABLE_TARGET --num-mappers 3
```

You can control the behavior of the connector by using extra arguments. Extra arguments must appear at the end of the command. Use a double-dash separator (--) to indicate the end of the standard arguments and the beginning of the extra arguments. For example, the following command uses the double-dash (--) separator (in bold for emphasis) to separate the standard arguments --table and --num-mappers from the extra arguments --input-method and --query-band:

```
$ sqoop ... --table MY_TABLE --num-mappers 3 -- --input-method split.by.amp
 --query-band DC=BP\;Location=Europe
```

Table 2: Teradata Connector Feature Support

Parameter	Tool	Description
--staging-table	import and export	Override the default staging table name. This parameter applies only if staging tables are used during data transfer.
--staging-database	import and export	Override the default staging database name. This parameter applies only if staging tables are used during the data transfer.
--staging-force	import and export	Force the connector to create the staging table if the input/ output method supports staging tables.
--input-method	import	Specify the input method used to transfer data from Teradata to Hadoop.
--output-method	export	Specify the output method used to transfer data from Hadoop to Teradata.
--batch-size	import and export	Specify the number of rows processed together in one batch.
--access-lock	import	Improve concurrency. When used, the import job is not blocked by concurrent accesses to the same table.
--query-band	import and export	Allow arbitrary query bands to be set for all queries that are run by the connector. The expected format is a semicolon-separated key=value pair list. A final semicolon is required after the last key=value pair. For example, Data_Center=XO;Location=Europe;.
--error-table	export (only for internal.fastload)	Specify a prefix for created error tables.
--error-database	export (only for internal.fastload)	Override the default error database name.
--fastload-socket-hostname	export (only for internal.fastload)	Hostname or IP address of the host on which you are running Sqoop, one that is visible from the Hadoop cluster. The connector can autodetect the interface. This parameter overrides the autodection routine.

Parameter	Tool	Description
--keep-staging-table	import	By default, the connector drops all automatically created staging tables when export fails. This option leaves the staging tables with partially imported data in the database.
--num-partitions-for-staging-table	import (only for split.by.partition)	Number of partitions to use for the automatically created staging table. The connector automatically generates the value based on the number of mappers used.
--skip-xviews	import and export	By default, the connector uses Teradata system views to obtain metadata. With this parameter, the connector switches to XViews instead.
--date-format	import and export	Use custom format for columns of date type. The parameter uses SimpleDateFormat formatting options.
--time-format	import and export	Use custom format for columns of time type. The parameter uses SimpleDateFormat formatting options.
--timestamp-format	import and export	Use custom format for columns of timestamp type. The parameter uses SimpleDateFormat formatting options.

Input Methods

Cloudera Connector Powered by Teradata supports the following methods for importing data from Teradata to Hadoop:

- split.by.amp
- split.by.value
- split.by.partition
- split.by.hash

split.by.amp Method

This optimal method retrieves data from Teradata. The connector creates one mapper per available Teradata AMP, and each mapper subsequently retrieves data from each AMP. As a result, no staging table is required. This method requires Teradata 14.10 or higher.

split.by.value Method

This method creates input splits as ranges on the split by column (usually the table's primary key). Each split is subsequently processed by a single mapper to transfer the data using SELECT queries. All splits can access all AMPs to retrieve data, so you should set the number of mappers between 20 and 30 because there is a limit for all-AMP concurrently running operations on the Teradata appliance. Ensure that users transferring data have sufficient spool space available for the SELECT queries.

split.by.partition Method

This method is preferred for extracting a large amount of data from the Teradata system. Behavior of this method depends whether source table is partitioned or not.

split.by.hash Method

This input method is similar to the split.by.partition method. Instead of directly operating on value ranges of one column, this method operates on the hash of the column. You can use importing by hash to extract data in situations where split.by.value and split.by.partition are not appropriate. Each mapper can access all AMPs available in the system, so set the number of mappers between 20 and 30 because there is a limit for all-AMP concurrent jobs on the Teradata appliance.

The following example shows import using input method split.by.hash:

```
$ sqoop import --connect jdbc:teradata://localhost/DATABASE=sqooptest \
--username sqooptest --password xxxxx --table MY_TABLE --num-mappers 3 \
--target-dir /user/sqooptest/MY_TABLE - --input-method split.by.hash
```

If your input table is not partitioned, the connector creates a partitioned staging table and runs an INSERT into SELECT query to move data from the source table into the staging table. Subsequently, each mapper transfers data from one partition, resulting in a single AMP operation. With a single AMP, you can use a large number of mappers to obtain optimal performance. The amount of available permanent space must be as large as your source table and the amount of spool space required to run the SELECT queries.

If your table is already partitioned, no extra staging table is created. However, you can force the connector to re-partition your data using the --staging-force parameter to achieve better performance. Without forcing repartition of the data, this method opens all-AMP operation, so you should use between 20 and 30 mappers. If your source table is a PI table, and your split by column is the table's primary key, the connector creates a single AMP operation, and you can use high number of mappers.

Output Methods

Cloudera Connector Powered by Teradata supports the following output methods to export data from Hadoop to Teradata:

- batch.insert
- internal.fastload

batch.insert Method

This method uses JDBC batch jobs to export data to the Teradata appliance. This method should be used only when other methods are not a viable. It creates a partitioned stating table before the export job, and then subsequently each mapper transfers data from Hadoop to one partition. After all mappers end, the INSERT into SELECT statement is called to transfer the data from staging to table to final destination. Ensure that you have sufficient permanent space for two copies of your data. You also need sufficient spool space for running the INSERT into SELECT query. The number of mappers that this method can use is limited only by the number of concurrent sessions allowed on the Teradata appliance.

internal.fastload Method

This method requires a partitioned staging table. Data is first exported by each mapper into a different partition and then moved to the target table, using the INSERT into SELECT statement. Make sure that you have sufficient permanent and spool space to store two copies of your data and to move them from the staging table to the target table. All mappers participate in one FastLoad job coordinated by an internal protocol. This is the fastest method for exporting data from Hadoop to a Teradata appliance. Because all mappers participate in the one FastLoad job, only one Teradata utility slot is used for the export job. The number of mappers is limited only by total number of AMPs available in your system.

The Teradata server is started on the machine where the sqoop command is running, and all mappers started by this sqoop command must connect to it. Because a mapper can be run on any hosts in the cluster, all hosts must have access to the machine running the sqoop command.

Uninstalling Teradata Connectors

If you have a CDP cluster, Cloudera Manager handles the Teradata Connector completely. Use Cloudera Manager to deactivate and remove the parcel.

Uninstallation with CDH 6 and CDP 7

Perform the following steps to uninstall the Sqoop connectors for Teradata using Cloudera Manager:

1. **Removing the Sqoop Connector Parcels**:
   - **a.** In the Cloudera Manager Admin Console, click **Hosts** in the top navigation bar and then go to the **Parcels** tab. Parcels for the Sqoop connectors are listed on this page, prefixed by "SQOOP_", followed by the name of the connector.
   - **b.** The Sqoop connectors are listed as **Activated**. To deactivate a parcel, click **Actions** on an activated parcel and select **Deactivate**.
   - **c.** To remove the parcel, click the down arrow to the right of the **Activate** button and select **Remove from Hosts**.
2. **Redeploy client configuration:**
   - **a.** In the Cloudera Manager Admin Console, go to the Sqoop Client service.
   - **b.** From the Actions menu at the top right of the service page, select **Deploy Client Configuration**.
   - **c.** Click **Deploy Client Configuration** to confirm redeployment of client configuration.