Microsoft® SQL Server® Connector for Apache Hadoop Version 1.0

User Guide

October 3, 2011

Contents

Legal Notice

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred. This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

Some information relates to pre-released product which may be substantially modified before it’s commercially released. Microsoft makes no warranties, express or implied, with respect to the information provided here.

Introduction

What is SQL Server-Hadoop Connector?

Microsoft SQL Server Connector for Apache Hadoop (SQL Server-Hadoop Connector) is a Sqoop-based connector that facilitates efficient data transfer between SQL Server 2008 R2 and Hadoop. Sqoop supports several databases.

This connector extends JDBC-based Sqoop connectivity to facilitate data transfer between SQL Server and Hadoop, and also supports the JDBC features as mentioned in the SQOOP User Guide on the Cloudera website. In addition to this, this connector provides support for *nchar* and *nvarchar* data types.

With SQL Server-Hadoop Connector, you import data from:

- ✓ tables in SQL Server to delimited text files on HDFS
- ✓ tables in SQL Server to SequenceFiles files on HDFS
- ✓ tables in SQL Server to tables in Hive*
- ✓ result of queries executed on SQL Server to delimited text files on HDFS
- ✓ result of queries executed on SQL Server to SequenceFiles files on HDFS
- ✓ result of queries executed on SQL Server to tables in Hive*

Note: importing data from SQL Server into HBase is not supported in this release.

With SQL Server-Hadoop Connector, you can export data from:

- ✓ delimited text files on HDFS to SQL Server
- ✓ sequenceFiles on HDFS to SQL Server
- ✓ hive Tables* to tables in SQL Server

* Hive is a data warehouse infrastructure built on top of Hadoop (http://wiki.apache.org/hadoop/Hive). We recommend to use hive-0.7.0-cdh3u0 version of Cloudera Hive.

What is Sqoop?

Sqoop is an open source connectivity framework that facilitates transfer between multiple Relational Database Management Systems (RDBMS) and HDFS. Sqoop uses MapReduce programs to import and export data; the imports and exports are performed in parallel with fault tolerance.

Supported File Types

The *Source* / *Target* files being used by Sqoop can be delimited text files (for example, with commas or tabs separating each field), or binary SequenceFiles containing serialized record data. Please refer to section 7.2.7 in the Sqoop User Guide for more details on supported file types. For information on SequenceFile format, please refer to the Hadoop API page.

Before You Install SQL Server-Hadoop Connector

The following requirements and steps explain how to prepare your system before installing SQL Server-Hadoop Connector.

Requirements

This User Guide assumes your environment has both Linux (for Hadoop setup) and Windows (with SQL Server setup). Both are required to use the SQL Server-Hadoop Connector.

Step 1: Install and Configure Cloudera's Distribution Including Hadoop

The first installation step is to install and configure Cloudera's Distribution Including Hadoop Update 1 (CDH3U1) on Linux. This is available for download from the Cloudera site at www.cloudera.com/downloads.

We also support Cloudera's CDH3U0 distribution of Hadoop for this connector, but we recommend Cloudera's CDH3U1 distribution of Hadoop. Set the HADOOP_HOME environment variable to the parent directory where Hadoop is installed.

Step 2: Install and Configure Sqoop

The next step is to install and configure Sqoop, if not already installed, on the master node of the Hadoop cluster. We recommend downloading and installing SQOOP 1.3.0-cdh3u1 (sqoop-1.3.0-cdh3u1.tar.gz) from http://archive.cloudera.com/cdh/3/.

For detailed instructions about using Sqoop, see the Sqoop User Guide at http://archive.cloudera.com/cdh/3/sqoop-1.3.0-cdh3u1/SqoopUserGuide.html . SQL Server – Hadoop Connector has backward compatibility with Sqoop-1.2.0, but, we recommended using Sqoop 1.3.0.

After installing and configuring Sqoop, verify the following environment variables are set on the machine with Sqoop installation, as described in the following table. These must be set for SQL Server-Hadoop Connector to work correctly.

Environment Variable	Value to Assign
SQOOP_HOME	Absolute path to the Sqoop installation directory
SQOOP_CONF_DIR	$SQOOP_HOME/conf

Step 3: Download and install the Microsoft JDBC Driver

Sqoop and SQL Server-Hadoop Connector use JDBC technology to establish connections to remote RDBMS servers and therefore needs the JDBC driver for SQL Server. To install this driver on Linux node where Sqoop is already installed:

- Visit http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=21599 and download "sqljdbc_<version>_enu.tar.gz".
- Copy it on the machine with Sqoop installation.
- Unpack the tar file using following command: *tar –zxvf sqljdbc_<version>_enu.tar.gz.* This will create a directory "sqljdbc_3.0" in current directory.

- Copy the driver jar (sqljdbc_3.0/enu/sqljdbc4.jar) file to the $SQOOP_HOME/lib directory on machine with Sqoop installation.

Download and Install SQL Server-Hadoop Connector

After all of the previous steps have completed, you are ready to download, install and configure the SQL Server-Hadoop Connector on the machine with Sqoop installation.

The SQL Server–Hadoop connector is distributed as a compressed tar archive named sqoop-sqlserver-1.0.tar.gz. Download the tar archive from http://download.microsoft.com , and save the archive on the same machine where Sqoop is installed.

This archive is composed of the following files and directories:

File / Directory	Description
install.sh	Is a shell script that installs the SQL Server – Hadoop Connector files into the Sqoop directory structure.
Microsoft SQL Server-Hadoop Connector User Guide.pdf	Contains instructions to deploy and execute SQL Server – Hadoop Connector.
lib/	Contains the sqoop-sqlserver-1.0.jar file
conf/	Contains the configuration files for SQL Server – Hadoop Connector.
THIRDPARTYNOTICES FOR HADOOP-BASED CONNECTORS.txt	Contains the third party notices.
SQL Server Connector for Apache Hadoop MSLT.pdf	EULA for the SQL Server Connector for Apache Hadoop

To install SQL Server–Hadoop Connector:

1. Login to the machine where Sqoop is installed as a user who has permission to install files
2. Extract the archive with the command: "*tar –zxvf sqoop-sqlserver-1.0.tar.gz*". This will create "*sqoop-sqlserver-1.0*" directory in current directory
3. Change directory (cd) to "*sqoop-sqlserver-1.0*"
4. Ensure that the MSSQL_CONNECTOR_HOME environment variable is set to the absolute path of the *sqoop -sqlserver-1.0* directory.
5. Run the shell script *install.sh* with no additional arguments.
6. Installer will copy the connector jar and configuration file under existing Sqoop installation

Example Import Commands

You're now ready to use SQL Server-Hadoop Connector. The following examples import data from SQL Server to HDFS or Hive.

The assumption is that you are running the commands from the $SQOOP_HOME directory on the master node of the Hadoop Cluster, where Sqoop is installed.

Example 1: Import to delimited text files on HDFS

The following command imports data from TPCH lineitem table in SQL Server to delimited text files in /data/lineitemData directory on HDFS:

```
$bin/sqoop import --connect
'jdbc:sqlserver://10.80.181.127;username=dbuser;password=dbpasswd;database=tpch'  --table lineitem --
target-dir /data/lineitemData
```

Example 2: Import with the split-by option

The following command specifies split-by column to compute the splits for mappers:

```
$bin/sqoop import --connect
'jdbc:sqlserver://10.80.181.127;username=dbuser;password=dbpasswd;database=tpch'  --table lineitem --
target-dir /data/lineitemData --split-by  L_ORDERKEY -m  3
```

Example 3: Import to SequenceFiles on HDFS

The following command imports data in SequenceFiles on HDFS:

```
$bin/sqoop import --connect
'jdbc:sqlserver://10.80.181.127;username=dbuser;password=dbpasswd;database=tpch'  --table lineitem --
target-dir /data/lineitemData  --as-sequencefile
```

Example 4: Import to tables in Hive

The following command imports data from lineitem tables in SQL Server to a table in Hive:

```
$bin/sqoop import --connect
'jdbc:sqlserver://10.80.181.127;username=dbuser;password=dbpasswd;database=tpch'  --table lineitem --
hive-import
```

For using Hive import, ensure that hive is installed and HIVE_HOME is set to the parent directory where hive is installed.

Example Export Commands

The following examples export data from HDFS or Hive to SQL Server. The assumption is that you are running the commands from the $SQOOP_HOME directory on the master node of the Hadoop Cluster, where Sqoop is installed.

Example 1: Export data from a delimited text on HDFS

The following command exports data from a delimited text file /data/lineitemData on HDFS to lineitem table in tpch database on SQL Server .

```
$bin/sqoop export --connect
'jdbc:sqlserver://10.80.181.127;username=dbuser;password=dbpasswd;database=tpch' --table lineitem --
export-dir /data/lineitemData
```

Example 2: Export data from a delimited text file or Sequence File on HDFS with a user-defined number of mappers.

The following command exports data from a delimited text file on HDFS with user defined number of mappers.

```
$bin/sqoop export --connect
'jdbc:sqlserver://10.80.181.127;username=dbuser;password=dbpasswd;database=tpch' --table lineitem --
export-dir /data/lineitemData -m 3
```

The following command exports data from a sequential file on HDFS. In the following example, the "--jar-file <ORM_JAR_FILE> --classname <ORM_ClassName> " parameters specify the jar file and the appropriate class name that needs to loaded from this jar file. For more details on these options, see the **Sqoop User Guide**.

```
$bin/sqoop export --connect
'jdbc:sqlserver://10.80.181.127;username=dbuser;password=dbpasswd;database=tpch' --table lineitem --
export-dir /data/lineitemData -m 3 --class-name <ORM_ClassName> --jar-file <ORM_JAR_FILE>
```

Example 3: Export data from delimited text or sequence file on HDFS using a staging table

The following command uses a staging table and specifies to first clear the staging table before starting the export.

```
$bin/sqoop export --connect
'jdbc:sqlserver://10.80.181.127;username=dbuser;password=dbpasswd;database=tpch' --table lineitem
--export-dir /data/lineitemData --staging-table lineitem_stage --clear-staging-table
```

Note: For current release, using "--direct" option for running Sqoop import / export tools would make no difference in execution of import / export flow.

Data Types

The following table summarizes the data types supported by this version of the SQL Server – Hadoop Connector. All other SQL Server types (e.g., XML, geography, geometry, sql_variant) not mentioned in the table below are not supported at this time.

Data type Category	SQL Server Data Type	SQL server data type Range	Sqoop Data Type	Sqoop Data type Range
Exact numeric	bigint	-2^63 to 2^63-1	Long	MAX_VALUE: 2^63-1 (9223372036854775807) MIN_VALUE: -2^63 (-9223372036854775808)
	bit	0 or 1	Boolean	1-bit
	decimal	- 10^38 +1 to 10^38 - 1	java.math.BigDecimal	No range specification found (non-lossy)
	int	-2^31 to 2^31-1	Integer	MAX_VALUE: 2^31-1 (2147483647) MIN_VALUE: -2^31 (-2147483648)
	money	-922,337,203,685,477.5808 to 922,337,203,685,477.5807	java.math.BigDecimal	No range specification found (non-lossy)
	smallint	-2^15 to 2^15-1	Integer	MAX_VALUE: 2^31-1 (2147483647) MIN_VALUE: -2^31 (-2147483648)
	smallmoney	- 214,748.3648 to 214,748.3647	java.math.BigDecimal	No range specification found (non-lossy)
	tinyint	0 to 255	Integer	MAX_VALUE: 2^31-1 (2147483647) MIN_VALUE: -2^31 (-2147483648)
Approximate numeric	float	- 1.79E+308 to -2.23E-308, 0 and 2.23E-308 to 1.79E+308	Double	MAX_VALUE: (2-2^-52)·2^1023 or (1.7976931348623157E308d) MIN_VALUE: 2^-1074 or (4.9E-324d)
	real	- 3.40E + 38 to -1.18E - 38, 0 and 1.18E - 38 to 3.40E The ISO synonym for real is float(24)	Float	MAX_VALUE: (2-2^-23)·2^127 or (3.4028234663852886E38f) MIN_VALUE: 2^-149 or (1.401298464324817E-45f)
Date and time	date	0001-01-01 through 9999-12-31 January 1, 1 A.D. through December 31, 9999 A.D.	java.sql.Date	int year, int month, int date: Year - the year minus 1900; Must be 0 to 8099. (Note that 8099 is 9999 minus 1900.) month - 0 to 11 day - 1 to 31

Data type Category	SQL Server Data Type	SQL server data type Range	Sqoop Data Type	Sqoop Data type Range
	datetime2	**Date Range:** 0001-01-01 through 9999-12-31 January 1,1 A.D. through December 31, 9999 A.D. **Time Range:** 00:00:00 through 23:59:59.9999999	java.sql.Timestamp	int year, int month, int date, int hour, int minute, int second, int nano: year - the year minus 1900 month - 0 to 11 date - 1 to 31 hour - 0 to 23 minute - 0 to 59 second - 0 to 59 nano - 0 to 999,999,999
	smalldatetime	**Date Range:** 1900-01-01 through 2079-06-06 January 1, 1900, through June 6, 2079 **Time Range:** 00:00:00 through 23:59:59 2007-05-09 23:59:59 will round to 2007-05-10 00:00:00	java.sql.Timestamp	int year, int month, int date, int hour, int minute, int second, int nano: year - the year minus 1900 month - 0 to 11 date - 1 to 31 hour - 0 to 23 minute - 0 to 59 second - 0 to 59 nano - 0 to 999,999,999
	datetime	**Date Range:** January 1, 1753, through December 31, 9999 **Time Range:** 00:00:00 through 23:59:59.997	java.sql.Timestamp	int year, int month, int date, int hour, int minute, int second, int nano: year - the year minus 1900 month - 0 to 11 date - 1 to 31 hour - 0 to 23 minute - 0 to 59 second - 0 to 59 nano - 0 to 999,999,999
	time	00:00:00.0000000 through 23:59:59.9999999	java.sql.Time	int hour, int minute, int second: hour - 0 to 23 minute - 0 to 59 second - 0 to 59
Character strings	char	Fixed-length, non-Unicode character data With a length of n bytes. n must be a value from 1 through 8,000.	String	Up to 8,000 characters
	varchar	Variable-length, non-Unicode character data. n can be a value from 1 through 8,000. Varchar(max) not supported.	String	Up to 8,000 characters

Data type Category	SQL Server Data Type	SQL server data type Range	Sqoop Data Type	Sqoop Data type Range
Unicode character strings	nchar	Fixed-length Unicode character data of n characters. n must be a value From 1 through 4,000.	String	Up to 4,000 unicode characters
	nvarchar	Variable-length Unicode character data, n can be a Value from 1 through 4,000. Nvarchar(max) not supported.	String	Up to 4,000 unicode characters
Binary strings	binary	Fixed-length binary data with a length of n bytes, where n is a value from 1 through 8,000.	BytesWritable.java	Up to 8,000 bytes
	varbinary	Variable-length binary data. n can be a value from 1 through 8,000. Varbinary(max) not supported.	BytesWritable.java	Up to 8,000 bytes

Known Issues

This JDBC-based connector is an extension of Sqoop and the open issues in Sqoop also occur in this connector. For a detailed description of Sqoop known issues, see https://issues.apache.org/jira/browse/SQOOP .

The use of *--driver* switch does not function correctly and hence avoid using *--driver* switch for the SQL Server connector to work. Use the *--connect* switch instead.

Troubleshooting and Support

This JDBC-based connector is an extension of Sqoop. For troubleshooting and support details with respect to Sqoop, see the Sqoop User Guide on the Cloudera site.

Security Notes

- For secure communication between the Hadoop nodes, we recommend users to configure IPsec or similar technologies. This will help prevent the Man-In the Middle attack.

  You can refer the following link:
  https://help.ubuntu.com/community/IPSecHowTo

- We recommend using the "`escaped-by`" and "`enclosed-by`" switches provided in Sqoop.

- To ensure secure communication between the Hadoop nodes and SQL Server use "encrypt=true" in the connection string. For details refer to the following link http://msdn.microsoft.com/en-us/library/bb879949.aspx on MSDN. This is recommended but is not tested with the current release.