cloudera
Ask Bigger Questions

Cloudera JDBC Driver for Apache Hive
Version 2.5.16

Important Notice

Cloudera, the Cloudera logo, Cloudera Impala, Impala, and any other product or service names or slogans contained in this document, except as otherwise disclaimed, are trademarks of Cloudera and its suppliers or licensors, and may not be copied, imitated or used, in whole or in part, without the prior written permission of Cloudera or the applicable trademark holder.

Hadoop and the Hadoop elephant logo are trademarks of the Apache Software Foundation. All other trademarks, registered trademarks, product names and company names or logos mentioned in this document are the property of their respective owners. Reference to any products, services, processes or other information, by trade name, trademark, manufacturer, supplier or otherwise does not constitute or imply endorsement, sponsorship or recommendation thereof by us.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Cloudera.

Cloudera may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Cloudera, the furnishing of this document does not give you any license to these patents, trademarks copyrights, or other intellectual property.

The information in this document is subject to change without notice. Cloudera shall not be liable for any damages resulting from technical errors or omissions which may be present in this document, or from use of this document.

Cloudera, Inc.
1001 Page Mill Road, Building 2
Palo Alto, CA 94304-1008
info@cloudera.com
US: 1-888-789-1488
Intl: 1-650-843-0595
www.cloudera.com

Release Information

Version: 2.5.16

Date: November 12, 2015

Table of Contents

Introduction

The Cloudera JDBC Driver for Apache Hive is used for direct SQL and HiveQL access to Apache Hadoop / Hive distributions, enabling Business Intelligence (BI), analytics, and reporting on Hadoop / Hive-based data. The driver efficiently transforms an application's SQL query into the equivalent form in HiveQL, which is a subset of SQL-92. If an application is Hive-aware, then the driver is configurable to pass the query through to the database for processing. The driver interrogates Hive to obtain schema information to present to a SQL-based application. Queries, including joins, are translated from SQL to HiveQL. For more information about the differences between HiveQL and SQL, see "Features" on page 21.

The Cloudera JDBC Driver for Apache Hive complies with the JDBC 3.0, 4.0 and 4.1 data standards. JDBC is one of the most established and widely supported APIs for connecting to and working with databases. At the heart of the technology is the JDBC driver, which connects an application to the database. For more information about JDBC, see http://www.simba.com/resources/data-access-standards-library.

This guide is suitable for users who want to access data residing within Hive from their desktop environment. Application developers may also find the information helpful. Refer to your application for details on connecting via JDBC.

System Requirements

Each computer where you use the Cloudera JDBC Driver for Apache Hive must have Java Runtime Environment (JRE) installed. The version of JRE that must be installed depends on the version of the JDBC API you are using with the driver. Table 1 lists the required version of JRE for each version of the JDBC API.

Table 1. Driver System Requirements

JDBC API Version	JRE Version
3.0	4.0 or 5.0

JDBC API Version	JRE Version
4.0	6.0 or later
4.1	7.0 or later

The Cloudera JDBC Driver for Apache Hive supports Hive 0.11, 0.12, 0.13, 0.14, 1.0, and 1.1.

Cloudera JDBC Driver for Apache Hive Files

The Cloudera JDBC Driver for Apache Hive is delivered in the following ZIP archives, where *version* is the version number of the driver:

- Cloudera_HiveJDBC3_*version*.zip
- Cloudera_HiveJDBC4_*version*.zip
- Cloudera_HiveJDBC41_*version*.zip

Each archive contains the driver supporting the JDBC API version indicated in the archive name.

The archives contain the following file and folder structure, where *LibVersion* is the version number of the library and *APIVersion* is the JDBC API version that the driver supports:

- HiveJDBC*APIVersion*
  - hive_metastore.jar
  - hive_service.jar
  - HiveJDBC*APIVersion*.jar
  - libfb303-*LibVersion*.jar
  - libthrift-*LibVersion*.jar
  - log4j-*LibVersion*.jar
  - ql.jar
  - slf4j-api-*LibVersion*.jar
  - slf4j-log4j12-*LibVersion*.jar
  - TCLIServiceClient.jar
  - zookeeper-*LibVersion*.jar

Using the Cloudera JDBC Driver for Apache Hive

To access a Hive data warehouse using the Cloudera JDBC Driver for Apache Hive, you need to configure the following:

- Class path
- Driver or DataSource class
- Connection URL

For sample code that demonstrates how to use the driver, see "Java Sample Code" on page 8.

Important:

The Cloudera JDBC Driver for Apache Hive is a forward-only, read-only driver with no transaction support. Because the driver does not support transactions, auto-commit is always set to **true**

Setting the Class Path

To use the Cloudera JDBC Driver for Apache Hive, you must set the class path to include all the JAR files from the ZIP archive containing the driver that you are using.

The class path is the path that the Java Runtime Environment searches for classes and other resource files. For more information, see the topic *Setting the Class Path* in the Java SE Documentation at http://docs.oracle.com/javase/7/docs/technotes/tools/windows/classpath.html.

Initializing the Driver Class

Before connecting to the data store, you must initialize the appropriate class for the Hive server and your application.

The following is a list of the classes used to connect the Cloudera JDBC Driver for Apache Hive to Hive Server 1 and Hive Server 2 instances. The Driver classes extend java.sql.Driver, and the DataSource classes extend javax.sql.DataSource and javax.sql.ConnectionPoolDataSource.

To support JDBC 3.0, classes with the following fully-qualified class names (FQCNs) are available:

- com.cloudera.hive.jdbc3.HS1Driver
- com.cloudera.hive.jdbc3.HS2Driver
- com.cloudera.hive.jdbc3.HS1DataSource
- com.cloudera.hive.jdbc3.HS2DataSource

To support JDBC 4.0, classes with the following FQCNs are available:

- com.cloudera.hive.jdbc4.HS1Driver
- com.cloudera.hive.jdbc4.HS2Driver
- com.cloudera.hive.jdbc4.HS1DataSource
- com.cloudera.hive.jdbc4.HS2DataSource

To support JDBC 4.1, classes with the following FQCNs are available:

- com.cloudera.hive.jdbc41.HS1Driver
- com.cloudera.hive.jdbc41.HS2Driver
- com.cloudera.hive.jdbc41.HS1DataSource
- com.cloudera.hive.jdbc41.HS2DataSource

The following sample code shows how to use the DriverManager to establish a connection:

```
private static Connection connectViaDM() throws Exception
{
    Connection connection = null;
    Class.forName(DRIVER_CLASS);
    connection = DriverManager.getConnection(CONNECTION_URL);
    return connection;
}
```

The following sample code shows how to use the DataSource class to establish a connection:

```
private static Connection connectViaDS() throws Exception
{
    Connection connection = null;
    Class.forName(DRIVER_CLASS);
    DataSource ds = new com.cloudera.hive.jdbc4.HS1DataSource();
    ds.setURL(CONNECTION_URL);
    connection = ds.getConnection();
    return connection;
}
```

Building the Connection URL

Use the connection URL to supply connection information to the data source that you are accessing. The following is the format of the connection URL for the Cloudera JDBC Driver for Apache Hive, where *Subprotocol* is **hive** if you are connecting to a Hive Server 1 instance or **hive2** if you are connecting to a Hive Server 2 instance, and *Host* is the DNS or IP address of the Hive server:

```
jdbc:Subprotocol://Host
```

By default, the driver connects to port 10000, uses the schema named **default**, and authenticates the connection using the user name **hive**.

You can specify optional settings such as the number of the TCP port to connect to, the schema to use, or any of the connection properties supported by the driver. For a list of the properties available in the driver, see "Driver Configuration Options" on page 80.

The following is the format of a connection URL that specifies some optional settings:

```
jdbc:Subprotocol://Host:Port[/Schema];Property1=Value;
Property2=Value;...
```

For example, to connect to port 11000 on a Hive Server 2 instance installed on the local machine, use a schema named **default2**, and authenticate the connection using a user name and password, you would use the following connection URL:

```
jdbc:hive2://localhost:11000[/default2];AuthMech=3;
UID=cloudera;PWD=cloudera
```

Important:

Be aware of the following:

- Properties are case-sensitive.
- Do not duplicate properties in the connection URL.

Note:

Note the following:

- If you specify a schema in the connection URL, you can still issue queries on other schemas by explicitly specifying the schema in the query. To inspect your databases and determine the appropriate schema to use, type the **show databases** command at the Hive command prompt.
- If you specify a property that is not supported by the driver, then the driver attempts to apply the property as a Hive server-side property for the client session.

Java Sample Code

The following Java code provides an example demonstrating how to use the JDBC API to do the following:

- Register the Cloudera JDBC Driver for Apache Hive
- Establish a connection to a Hive database
- Query the database
- Parse a result set
- Handle exceptions
- Clean up to avoid memory leakage

Important:

To use the Cloudera JDBC Driver for Apache Hive in an application, you must include all the JAR files from the ZIP archive in the class path for your Java project.

```
// java.sql packages are required
import java.sql.*;
class ClouderaJDBCHiveExample {

    // Define a string as the fully qualified class name
    // (FQCN) of the desired JDBC driver
    static String JDBCDriver = "com.cloudera.hive.jdbc3.HS1Driver";
    // Define a string as the connection URL
```

```
private static final String CONNECTION_URL =
"jdbc:hive://192.168.1.1:10000";


public static void main(String[] args) {


    Connection con = null;
    Statement stmt = null;
    ResultSet rs = null;


    // Define a plain query
    String query = "SELECT first_name, last_name, emp_id FROM
    default.emp";
    // Define a parametrized query
    String prepQuery = "SELECT first_name, last_name, emp_id
    FROM default.emp where store_id = ?";


    try {


        // Register the driver using the class name
        Class.forName(JDBC_DRIVER);


        // Establish a connection using the connection
        // URL
        con = DriverManager.getConnection(CONNECTION_URL);


        // Create a Statement object for sending SQL
        // statements to the database
        stmt = con.createStatement();


        // Execute the SQL statement
        rs = stmt.executeQuery(query);


        // Display a header line for output appearing in
        // the Console View
        System.out.printf("%20s%20s%20s\r\n", "FIRST NAME",
        "LAST NAME" , "EMPLOYEE ID");


        // Step through each row in the result set
```

```
// returned from the database
while(rs.next()) {
     // Retrieve values from the row where the
     // cursor is currently positioned using
     // column names
     String FirstName = rs.getString("first_name");
     String LastName = rs.getString("last_name");
     String EmployeeID = rs.getString("emp_id");

     // Display values in columns 20 characters
     // wide in the Console View using the
     // Formatter
     System.out.printf("%20s%20s%20s\r\n", FirstName,
     LastName, EmployeeID);
}
// Create a prepared statement
PreparedStatement prep = con.prepareStatement
(prepQuery);

// Bind the query parameter with a value
prep.setInt(1, 204);
// Execute the query
prep.execute();
rs = prep.getResultSet();
// Step through each row in the result set
// returned from the database
while(rs.next()) {
     // Retrieve values from the row where the
     // cursor is currently positioned using
     // column names
     String FirstName = rs.getString("first_name");
     String LastName = rs.getString("last_name");
     String EmployeeID = rs.getString("emp_id");

     // Display values in columns 20 characters
     // wide in the Console View using the
     // Formatter
```

```
            System.out.printf("%20s%20s%20s\r\n", FirstName,
            LastName, EmployeeID);
        }

    } catch (SQLException se) {
        // Handle errors encountered during interaction
        // with the data source
    } catch (Exception e) {
        // Handle other errors
    } finally {
        // Perform clean up
        try {
            if (rs != null) {
                rs.close();
            }
        } catch (SQLException se1) {
            // Log this
        }

        try {
            if (stmt != null) {
                stmt.close();
            }
        } catch (SQLException se2) {
            // Log this
        }
        try {
            if (prep != null) {
                prep.close();
            }
        } catch (SQLException se3) {
            // Log this
        }

        try {
            if (con != null) {
                con.close();
```

```
                    }
                } catch (SQLException se4) {
                    // Log this
                } // End try
            } // End try
        } // End main
} // End ClouderaJDBCHiveExample
```

Configuring Authentication

The Cloudera JDBC Driver for Apache Hive supports the following authentication mechanisms:

- No Authentication
- Kerberos
- User Name
- User Name and Password

You configure the authentication mechanism that the driver uses to connect to Hive by specifying the relevant properties in the connection URL.

For information about selecting an appropriate authentication mechanism when using the Cloudera JDBC Driver for Apache Hive, see "Authentication Options" on page 14.

For information about the properties you can use in the connection URL, see "Driver Configuration Options" on page 80.

Note:

In addition to authentication, you can configure the driver to connect over SSL. For more information, see "Configuring SSL" on page 20.

Using No Authentication

Note:

When connecting to a Hive server of type Hive Server 1, you must use No Authentication.

To configure a connection without authentication:

- Set the AuthMech property to 0.

For example:

```
jdbc:hive2://localhost:10000;AuthMech=0
```

Using Kerberos

Kerberos must be installed and configured before you can use this authentication mechanism. For information about configuring and operating Kerberos on Windows, see "Configuring Kerberos Authentication for Windows" on page 16. For other operating systems, refer to the MIT Kerberos documentation.

> **Note:**
>
> This authentication mechanism is available only for Hive Server 2.

To configure Kerberos authentication:

1. Set the AuthMech property to 1.
2. To use the default realm defined in your Kerberos setup, do not set the KrbRealm property.

   If your Kerberos setup does not define a default realm or if the realm of your Hive server is not the default, then set the KrbRealm property to the realm of the Hive server.
3. Set the KrbHostFQDN property to the fully qualified domain name of the Hive server host.
4. Set the KrbServiceName property to the service name of the Hive server.

For example:

```
jdbc:hive2://localhost:10000;AuthMech=1;KrbRealm=EXAMPLE.COM;
KrbHostFQDN=hs2.example.com;KrbServiceName=hive
```

Using User Name

This authentication mechanism requires a user name but does not require a password. The user name labels the session, facilitating database tracking.

> **Note:**
>
> This authentication mechanism is available only for Hive Server 2. Most default configurations of Hive Server 2 require User Name authentication.

To configure User Name authentication:

1. Set the AuthMech property to 2.
2. Set the UID property to an appropriate user name for accessing the Hive server.

For example:

```
jdbc:hive2://localhost:10000;AuthMech=2;UID=hs2
```

Using User Name and Password

This authentication mechanism requires a user name and a password.

> **Note:**
>
> This authentication mechanism is available only for Hive Server 2.

To configure User Name and Password authentication:

1. Set the AuthMech property to 3.
2. Set the UID property to an appropriate user name for accessing the Hive server.
3. Set the PWD property to the password corresponding to the user name you provided in step 2.

For example:

```
jdbc:hive2://localhost:10000;AuthMech=3;UID=hs2;PWD=*****
```

Authentication Options

Hive Server 1 does not support authentication. You must configure the driver to use No Authentication.

Hive Server 2 supports the following authentication mechanisms:

- No Authentication
- Kerberos
- User Name
- User Name and Password

Most default configurations of Hive Server 2 require User Name authentication. If you are unable to connect to your Hive server using User Name authentication, then verify the authentication mechanism configured for your Hive server by examining the hive-site.xml file. Examine the following properties to determine which authentication mechanism your server is set to use:

- **hive.server2.authentication**: This property sets the authentication mode for Hive Server 2. The following values are available:
  - **NOSASL** disables the Simple Authentication and Security Layer (SASL).
  - **KERBEROS** enables Kerberos authentication.
  - **NONE** enables plain SASL transport. NONE is the default value.
  - **PLANSASL** enables user name and password authentication using a cleartext password mechanism.
- **hive.server2.enable.doAs**: If this property is set to the default value of **TRUE**, then Hive processes queries as the user who submitted the query. If this property is set to **FALSE**, then queries are run as the user that runs the hiveserver2 process.

Table 2 lists authentication mechanisms to configure for the driver based on the settings in the hive-site.xml file.

Table 2. Hive Authentication Mechanism Configurations

hive.server2.authentication	hive.server2.enable.doAs	Driver Authentication Mechanism
NOSASL	FALSE	No Authentication
KERBEROS	TRUE or FALSE	Kerberos
NONE	TRUE or FALSE	User Name
LDAP	TRUE or FALSE	User Name and Password

Note:

It is an error to set hive.server2.authentication to NOSASL and hive.server2.enable.doAs to true. This configuration will not prevent the service from starting up, but results in an unusable service.

For more information about authentication mechanisms, refer to the documentation for your Hadoop / Hive distribution. See also *Running Hadoop in Secure Mode* at http://hadoop.apache.org/docs/r0.23.7/hadoop-project-dist/hadoop-common/ClusterSetup.html#Running_Hadoop_in_Secure_Mode.

Using No Authentication

When hive.server2.authentication is set to NOSASL, you must configure your connection to use No Authentication.

Using Kerberos

When connecting to a Hive server of type Hive Server 2 and hive.server2.authentication is set to KERBEROS, you must configure your connection to use Kerberos authentication.

Using User Name

When connecting to a Hive server of type Hive Server 2 and hive.server2.authentication is set to NONE, you must configure your connection to use User Name authentication. Validation of the credentials that you include depends on hive.server2.enable.doAs:

- If hive.server2.enable.doAs is set to TRUE, then the user name in the driver configuration must be an existing operating system user on the host that is running Hive Server 2.
- If hive.server2.enable.doAs is set to FALSE, then the user name in the driver configuration is ignored.

If no user name is specified in the driver configuration, then the driver defaults to using "hive" as the user name.

Using User Name and Password

When connecting to a Hive server of type Hive Server 2 and the server is configured to use the SASL-PLAIN authentication mechanism with a user name and a password, you must configure your connection to use User Name and Password authentication.

Configuring Kerberos Authentication for Windows

You can configure your Kerberos setup so that you use the MIT Kerberos Ticket Manager to get the Ticket Granting Ticket (TGT), or configure the setup so that you can use the driver to get the ticket directly from the Key Distribution Center (KDC). Also, if a client application obtains a Subject with a TGT, it is possible to use that Subject to authenticate the connection.

Downloading and Installing MIT Kerberos for Windows

To download and install MIT Kerberos for Windows 4.0.1:

1. Download the appropriate Kerberos installer:
   - For a 64-bit computer, use the following download link from the MIT Kerberos web-site: http://web.mit.edu/kerberos/dist/kfw/4.0/kfw-4.0.1-amd64.msi.
   - For a 32-bit computer, use the following download link from the MIT Kerberos web-site: http://web.mit.edu/kerberos/dist/kfw/4.0/kfw-4.0.1-i386.msi.

   > **Note:**
   >
   > The 64-bit installer includes both 32-bit and 64-bit libraries. The 32-bit installer includes 32-bit libraries only.
2. To run the installer, double-click the .msi file that you downloaded in step 1.
3. Follow the instructions in the installer to complete the installation process.
4. When the installation completes, click **Finish**.

Using the MIT Kerberos Ticket Manager to Get Tickets

Setting the KRB5CCNAME Environment Variable

You must set the KRB5CCNAME environment variable to your credential cache file.

To set the KRB5CCNAME environment variable:

1. Click the **Start** button, then right-click **Computer**, and then click **Properties**.
2. Click **Advanced System Settings**.
3. In the System Properties dialog box, on the **Advanced** tab, click **Environment Variables**.
4. In the Environment Variables dialog box, under the System variables list, click **New**.
5. In the **New System Variable** dialog box, in the Variable name field, type **KRB5CCNAME**.
6. In the **Variable Value** field, type the path for your credential cache file. For example, type **C:\KerberosTickets.txt**.

7. Click **OK** to save the new variable.
8. Make sure that the variable appears in the System Variables list.
9. Click **OK** to close the Environment Variables dialog box, and then click **OK** to close the System Properties dialog box.
10. Restart your computer.

Getting a Kerberos Ticket

To get a Kerberos ticket:

1. Click the **Start** button, then click **All Programs**, and then click the **Kerberos for Windows (64-bit)** or **Kerberos for Windows (32-bit)** program group.
2. Click **MIT Kerberos Ticket Manager**.
3. In the MIT Kerberos Ticket Manager, click **Get Ticket**.
4. In the Get Ticket dialog box, type your principal name and password, and then click **OK**.

If the authentication succeeds, then your ticket information appears in the MIT Kerberos Ticket Manager.

Authenticating to the Hive Server

To authenticate to the Hive server:

- Use a connection string that has the following properties defined:
  - AuthMech
  - KrbHostFQDN
  - KrbRealm
  - KrbServiceName

For detailed information about these properties, see "Driver Configuration Options" on page 80.

Using the Driver to Get Tickets

Deleting the KRB5CCNAME Environment Variable

To enable the driver to get Ticket Granting Tickets (TGTs) directly, you must ensure that the KRB5CCNAME environment variable has not been set.

To delete the KRB5CCNAME environment variable:

1. Click the **Start** button, then right-click **Computer**, and then click **Properties**.
2. Click **Advanced System Settings**.
3. In the System Properties dialog box, click the **Advanced** tab and then click **Environment Variables**.
4. In the Environment Variables dialog box, check if the KRB5CCNAME variable appears in the System variables list. If the variable appears in the list, then select the variable and click **Delete**.

5. Click **OK** to close the Environment Variables dialog box, and then click **OK** to close the System Properties dialog box.

Setting Up the Kerberos Configuration File

To set up the Kerberos configuration file:

1. Create a standard krb5.ini file and place it in the C:\Windows directory.
2. Ensure that the KDC and Admin server specified in the krb5.ini file can be resolved from your terminal. If necessary, modify "C:\Windows\System32\drivers\etc\hosts".

Setting Up the JAAS Login Configuration File

To set up the JAAS login configuration file:

1. Create a JAAS login configuration file that specifies a keytab file and “doNotPrompt=true”

   For example:

```
Client {
     com.sun.security.auth.module.Krb5LoginModule required
          useKeyTab=true
          keyTab="PathToTheKeyTab"
          principal="cloudera@CLOUDERA"
          doNotPrompt=true;
};
```

2. Set the java.security.auth.login.config environment variable to the location of the JAAS file.

   For example: C:\KerberosLoginConfig.ini

Authenticating to the Hive Server

To authenticate to the Hive server:

- Use a connection string that has the following properties defined:
  - AuthMech
  - KrbHostFQDN
  - KrbRealm
  - KrbServiceName

For detailed information about these properties, see "Driver Configuration Options" on page 80.

Using an Existing Subject to Authenticate the Connection

If the client application obtains a Subject with a TGT, then that Subject can be used to authenticate the connection to the server.

To use an existing Subject to authenticate the connection:

1. Create a PrivilegedAction for establishing the connection to the database.

For example:

```
// Contains logic to be executed as a privileged action
public class AuthenticateDriverAction
implements PrivilegedAction<Void>
{
     // The connection, which is established as a
     // PrivilegedAction
     Connection con;

     // Define a string as the connection URL
     static String ConnectionURL =
     "jdbc:hive2://192.168.1.1:10000";

     /**
     * Logic executed in this method will have access to the
     * Subject that is used to "doAs". The driver will get
     * the Subject and use it for establishing a connection
     * with the server.
     */
     @Override
     public Void run()
     {
          try
          {
               // Establish a connection using the connection
               URL
               con = DriverManager.getConnection(ConnectionURL);
          }
          catch (SQLException e)
          {
               // Handle errors that are encountered during
               // interaction with the data source
               e.printStackTrace();
          }
          catch (Exception e)
          {
               // Handle other errors
```

```
                e.printStackTrace();
            }
            return null;
        }
    }
```

2. Run the PrivilegedAction using the existing Subject, and then use the connection.

   For example:

```
// Create the action
AuthenticateDriverAction authenticateAction = new
AuthenticateDriverAction();
// Establish the connection using the Subject for
// authentication.
Subject.doAs(loginConfig.getSubject(), authenticateAction);
// Use the established connection.
authenticateAction.con;
```

Configuring SSL

If you are connecting to a Hive server that has Secure Sockets Layer (SSL) enabled, then you can configure the driver to connect to an SSL-enabled socket.

SSL connections require a KeyStore and a TrustStore. You can create a TrustStore and configure the driver to use it, or allow the driver to use one of the default TrustStores. If you do not configure the driver to use a specific TrustStore, then the driver uses the Java TrustStore jssecacerts. If jssecacerts is not available, then the driver uses cacerts instead.

To configure SSL:

1. Create a KeyStore and configure the driver to use it:
   a) Create a KeyStore containing your signed, trusted SSL certificate.
   b) Set the SSLKeyStore property to the full path of the KeyStore, including the file name.
   c) Set the SSLKeyStorePwd property to the password for the KeyStore.
2. Optionally, create a TrustStore and configure the driver to use it:
   a) Create a TrustStore containing your signed, trusted SSL certificate.
   b) Set the SSLTrustStore property to the full path of the TrustStore, including the file name.
   c) Set the SSLTrustStorePwd property to the password for the TrustStore.
3. Set the SSL property to 1.
4. Optionally, to allow the SSL certificate used by the server to be self-signed, set the AllowSelfSignedCerts property to 1

5. Optionally, to allow the common name of a CA-issued certificate to not match the host name of the Hive server, set the CAIssuedCertNamesMismatch property to 1

   Note:

   For self-signed certificates, the driver always allows the common name of the certificate to not match the host name.

For example:

```
jdbc:hive2://localhost:10000;AuthMech=3;SSL=1;
SSLKeyStore=C:\\Users\\bsmith\\Desktop\\keystore.jks;
SSLKeyStorePwd=*****;UID=hs2;PWD=*****
```

Note:

For more information about the connection properties used in SSL connections, see "Driver Configuration Options" on page 80

Features

More information is provided on the following features of the Cloudera JDBC Driver for Apache Hive:

SQL Query versus HiveQL Query

The native query language supported by Hive is HiveQL. HiveQL is a subset of SQL-92. However, the syntax is different enough that most applications do not work with native HiveQL.

Data Types

The Cloudera JDBC Driver for Apache Hive supports many common data formats, converting between Hive, SQL, and Java data types.

Table 3 lists the supported data type mappings.

Table 3. Supported Data Types

Hive Type	SQL Type	Java Type
BIGINT	BIGINT	java.math.BigInteger
BINARY	VARBINARY	byte[]

Hive Type	SQL Type	Java Type
BOOLEAN	BOOLEAN	Boolean
CHAR (Available only in Hive 0.13.0 or later)	CHAR	String
DATE	DATE	java.sql.Date
DECIMAL (In Hive 0.13 and later, you can specify scale and precision when creating tables using the DECIMAL data type.)	DECIMAL	java.math.BigDecimal
DOUBLE	DOUBLE	Double
FLOAT	REAL	Float
INT	INTEGER	Long
SMALLINT	SMALLINT	Integer
TIMESTAMP	TIMESTAMP	java.sql.Timestamp
TINYINT	TINYINT	Short
VARCHAR (Available only in Hive 0.12.0 or later)	VARCHAR	String

The aggregate types (ARRAY, MAP, STRUCT, and UNIONTYPE) are not yet supported. Columns of aggregate types are treated as VARCHAR columns in SQL and STRING columns in Java.

Catalog and Schema Support

The Cloudera JDBC Driver for Apache Hive supports both catalogs and schemas to make it easy for the driver to work with various JDBC applications. Since Hive only organizes tables into schemas/databases, the driver provides a synthetic catalog called “HIVE” under which all of the schemas/databases are organized. The driver also maps the JDBC schema to the Hive schema/database.

Note:

Setting the CatalogSchemaSwitch connection property to 1 will cause Hive catalogs to be treated as schemas in the driver as a restriction for filtering.

Interfaces and Supported Methods

The Cloudera JDBC Driver for Apache Hive implements the following JDBC interfaces:

- "CallableStatement" on page 23
- "Connection" on page 33
- "DatabaseMetaData" on page 38
- "DataSource" on page 50
- "Driver" on page 51
- "ParameterMetaData" on page 52
- "PooledConnection" on page 53
- "PreparedStatement" on page 54
- "ResultSet" on page 59
- "ResultSetMetaData" on page 74
- "Statement" on page 75

However, the driver does not support every method from these interfaces. For information about whether a specific method is supported by the driver and which version of the JDBC API is the earliest version that supports the method, refer to the following sections.

The driver does **not** support the following JDBC features:

- Array
- Blob
- Clob
- Ref
- Savepoint
- SQLData
- SQLInput
- SQLOutput
- Struct

CallableStatement

The CallableStatement interface extends the PreparedStatement interface.

Table 4 lists the methods that belong to the CallableStatement interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the CallableStatement interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/java/sql/CallableStatement.html.

Table 4. Methods in the CallableStatement Class Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`Array getArray(int i)`	3.0	No	
`Array getArray(String parameterName)`	3.0	No	
`BigDecimal getBigDecimal (int parameterIndex)`	3.0	Yes	
`BigDecimal getBigDecimal (int parameterIndex, int scale)`	3.0	Yes	Deprecated
`BigDecimal getBigDecimal (String parameterName)`	3.0	Yes	
`Blob getBlob(int i)`	3.0	No	
`Blob getBlob(String parameterName)`	3.0	No	
`boolean getBoolean(int parameterIndex)`	3.0	Yes	
`boolean getBoolean (String parameterName)`	3.0	Yes	
`byte getByte(int parameterIndex)`	3.0	Yes	
`byte getByte(String parameterName)`	3.0	Yes	
`byte[] getBytes(int parameterIndex)`	3.0	Yes	
`byte[] getBytes(String parameterName)`	3.0	Yes	
`Clob getClob(int i)`	3.0	No	
`Clob getClob(String`	3.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
parameterName)			
Date getDate(int parameterIndex)	3.0	Yes	
Date getDate(int parameterIndex, Calendar cal)	3.0	Yes	
Date getDate(String parameterName)	3.0	Yes	
Date getDate(String parameterName, Calendar cal)	3.0	Yes	
double getDouble(int parameterIndex)	3.0	Yes	
double getDouble(String parameterName)	3.0	Yes	
float getFloat(int parameterIndex)	3.0	Yes	
float getFloat(String parameterName)	3.0	Yes	
int getInt(int parameterIndex)	3.0	Yes	
int getInt(String parameterName)	3.0	Yes	
long getLong(int parameterIndex)	3.0	Yes	
long getLong(String parameterName)	3.0	Yes	
Reader getNCharacterStream(int parameterIndex)	4.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`Reader getNCharacterStream (String parameterName)`	4.0	No	
`NClob getNClob(int parameterIndex)`	4.0	No	
`NClob getNClob(String parameterName)`	4.0	No	
`String getNString(int parameterIndex)`	4.0	No	
`String getNString(String parameterName)`	4.0	No	
`Object getObject(int parameterIndex)`	3.0	Yes	
`<T> T getObject(int parameterIndex, Class<T> type)`	4.1	No	
`Object getObject(int i, Map<String,Class<?>> map)`	3.0	No	
`Object getObject(String parameterName)`	3.0	Yes	
`<T> T getObject(String parameterName, Class<T> type)`	4.1	No	
`Object getObject(String parameterName, Map<String,Class<?>> map)`	3.0	Yes	
`Ref getRef(int i)`	3.0	No	
`Ref getRef(String parameterName)`	3.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`RowId getRowId(int parameterIndex)`	4.0	No	
`RowId getRowId(String parameterName)`	4.0	No	
`short getShort(int parameterIndex)`	3.0	Yes	
`short getShort(String parameterName)`	3.0	Yes	
`SQLXML getSQLXML(int parameterIndex)`	4.0	No	
`SQLXML getSQLXML(String parameterName)`	4.0	No	
`String getString(int parameterIndex)`	3.0	Yes	
`String getString(String parameterName)`	3.0	Yes	
`Time getTime(int parameterIndex)`	3.0	Yes	
`Time getTime(int parameterIndex, Calendar cal)`	3.0	Yes	
`Time getTime(String parameterName)`	3.0	Yes	
`Time getTime(String parameterName, Calendar cal)`	3.0	Yes	
`Timestamp getTimestamp (int parameterIndex)`	3.0	Yes	
`Timestamp getTimestamp (int parameterIndex,`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`Calendar cal)`			
`Timestamp getTimestamp (String parameterName)`	3.0	Yes	
`Timestamp getTimestamp (String parameterName, Calendar cal)`	3.0	Yes	
`URL getURL(int parameterIndex)`	3.0	No	
`URL getURL(String parameterName)`	3.0	No	
`void registerOutParameter(int parameterIndex, int sqlType)`	3.0	Yes	
`void registerOutParameter(int parameterIndex, int sqlType, int scale)`	3.0	Yes	
`void registerOutParameter(int paramIndex, int sqlType, String typeName)`	3.0	Yes	
`void registerOutParameter (String parameterName, int sqlType)`	3.0	Yes	
`void registerOutParameter (String parameterName, int sqlType, int scale)`	3.0	Yes	
`void registerOutParameter (String parameterName, int sqlType, String`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
typeName)			
void setAsciiStream (String parameterName, InputStream x)	4.0	Yes	
void setAsciiStream (String parameterName, InputStream x, int length)	3.0	Yes	
void setAsciiStream (String parameterName, InputStream x, long length)	4.0	Yes	
void setBigDecimal (String parameterName, BigDecimal x)	3.0	Yes	
void setBinaryStream (String parameterName, InputStream x)	4.0	Yes	
setBinaryStream(String parameterName, InputStream x, int length)	3.0	Yes	
void setBinaryStream (String parameterName, InputStream x, long length)	4.0	Yes	
void setBlob(String parameterName, Blob x)	4.0	Yes	
void setBlob(String parameterName, InputStream inputStream)	4.0	Yes	
void setBlob(String parameterName, InputStream inputStream,	4.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`long length)`			
`void setBoolean(String parameterName, boolean x)`	3.0	Yes	
`void setByte(String parameterName, byte x)`	3.0	Yes	
`void setBytes(String parameterName, byte[] x)`	3.0	Yes	
`void setCharacterStream (String parameterName, Reader reader)`	4.0	Yes	
`void setCharacterStream (String parameterName, Reader reader, int length)`	3.0	Yes	
`void setCharacterStream (String parameterName, Reader reader, long length)`	4.0	Yes	
`void setClob(String parameterName, Clob x)`	4.0	Yes	
`void setClob(String parameterName, Reader reader)`	4.0	Yes	
`void setClob(String parameterName, Reader reader, long length)`	4.0	Yes	
`void setDate(String parameterName, Date x)`	3.0	Yes	
`void setDate(String parameterName, Date x, Calendar cal)`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void setDouble(String parameterName, double x)`	3.0	Yes	
`void setFloat(String parameterName, float x)`	3.0	Yes	
`void setInt(String parameterName, int x)`	3.0	Yes	
`void setLong(String parameterName, long x)`	3.0	Yes	
`void setNCharacterStream (String parameterName, Reader value)`	4.0	Yes	
`void setNCharacterStream (String parameterName, Reader value, long length)`	4.0	Yes	
`void setNClob(String parameterName, NClob value)`	4.0	Yes	
`void setNClob(String parameterName, Reader reader)`	4.0	Yes	
`void setNClob(String parameterName, Reader reader, long length)`	4.0	Yes	
`void setNString(String parameterName, String value)`	4.0	Yes	
`void setNull(String parameterName, int sqlType)`	3.0	Yes	
`void setNull(String parameterName, int`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
sqlType, String typeName)			
void setObject(String parameterName, Object x)	3.0	Yes	
void setObject(String parameterName, Object x, int targetSqlType)	3.0	Yes	
void setObject(String parameterName, Object x, int targetSqlType, int scale)	3.0	Yes	
void setRowId(String parameterName, RowId x)	4.0	Yes	
void setShort(String parameterName, short x)	3.0	Yes	
void setSQLXML(String parameterName, SQLXML xmlObject)	4.0	Yes	
void setString(String parameterName, String x)	3.0	Yes	
void setTime(String parameterName, Time x)	3.0	Yes	
void setTime(String parameterName, Time x, Calendar cal)	3.0	Yes	
void setTimestamp(String parameterName, Timestamp x)	3.0	Yes	
void setTimestamp(String parameterName, Timestamp x, Calendar cal)	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void setURL(String parameterName, URL val)`	3.0	Yes	
`boolean wasNull()`	3.0	Yes	
`boolean isWrapperFor (Class<?> iface)`	4.0	Yes	
`<T> T unwrap(Class<T> iface)`	4.0	Yes	

Connection

Table 5 lists the methods that belong to the Connection interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the Connection interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/java/sql/Connection.html.

Table 5. Methods in the Connection Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void clearWarnings()`	3.0	Yes	
`void close()`	3.0	Yes	
`void commit()`	3.0	Yes	Auto-commit cannot be set to false because it is hard-coded as true
`Array createArrayOf (String typeName, Object [] elements)`	4.0	No	
`Blob createBlob()`	4.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`Clob createClob()`	4.0	No	
`NClob createNClob()`	4.0	No	
`SQLXML createSQLXML()`	4.0	No	
`Statement createStatement ()`	3.0	Yes	
`Statement createStatement (int resultSetType, int resultSetConcurrency)`	3.0	No	
`Statement createStatement (int resultSetType, int resultSetConcurrency, int resultSetHoldability)`	3.0	No	
`Struct createStruct (String typeName, Object [] attributes)`	4.0	No	
`boolean getAutoCommit()`	3.0	Yes	Hard-coded to true
`String getCatalog()`	3.0	Yes	
`Properties getClientInfo ()`	4.0	Yes	
`String getClientInfo (String name)`	4.0	Yes	
`int getHoldability()`	3.0	Yes	Hard-coded to CLOSE_ CURSORS_AT_COMMIT
`DatabaseMetaData getMetaData()`	3.0	Yes	
`int getNetworkTimeout()`	4.1	No	
`String getSchema()`	4.1	Yes	The returned schema name does not always match the

Method	Supported Since JDBC Version	Supported by the Driver	Notes
			one used by statements. Statements use the schema name defined in the connection URL.
`int getTransactionIsolation()`	3.0	Yes	Hard-coded to TRANSACTION_READ_UNCOMMITTED
`Map<String,Class<?>> getTypeMap()`	3.0	No	
`SQLWarning getWarnings()`	3.0	Yes	
`boolean isClosed()`	3.0	Yes	
`boolean isReadOnly()`	3.0	Yes	Returns true
`boolean isValid(int timeout)`	4.0	Yes	
`String nativeSQL(String sql)`	3.0	Yes	
`CallableStatement prepareCall(String sql)`	3.0	No	
`CallableStatement prepareCall(String sql, int resultSetType, int resultSetConcurrency)`	3.0	No	
`CallableStatement prepareCall(String sql, int resultSetType, int resultSetConcurrency, int resultSetHoldability)`	3.0	No	
`PreparedStatement prepareStatement(String sql)`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`PreparedStatement prepareStatement(String sql, int autoGeneratedKeys)`	3.0	No	
`PreparedStatement prepareStatement(String sql, int[] columnIndexes)`	3.0	No	
`PreparedStatement prepareStatement(String sql, int resultSetType, int resultSetConcurrency)`	3.0	No	
`PreparedStatement prepareStatement(String sql, int resultSetType, int resultSetConcurrency, int resultSetHoldability)`	3.0	No	
`PreparedStatement prepareStatement(String sql, String[] columnNames)`	3.0	No	
`void releaseSavepoint (Savepoint savepoint)`	3.0	No	Savepoints are not available because transactions are not supported.
`void rollback()`	3.0	No	Savepoints are not available because transactions are not supported.
`void rollback(Savepoint savepoint)`	3.0	No	Savepoints are not available because transactions are not supported.
`void setAutoCommit (boolean autoCommit)`	3.0	Yes	Ignored because auto-commit is hard-coded to true
`void setCatalog(String catalog)`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void setClientInfo (Properties properties)`	4.0	Yes	
`void setClientInfo(String name, String value)`	4.0	Yes	
`void setHoldability(int holdability)`	3.0	Yes	
`void setNetworkTimeout (Executor executor, int milliseconds)`	4.1	No	
`void setReadOnly(boolean readOnly)`	3.0	Yes	
`Savepoint setSavepoint()`	3.0	No	Savepoints are not available because transactions are not supported.
`Savepoint setSavepoint (String name)`	3.0	No	Savepoints are not available because transactions are not supported.
`void setSchema(String schema)`	4.1	Yes	Does not actually change the schema name used by newly created statements; only changes the value returned by getSchema()
`void setTransactionIsolation (int level)`	3.0	Yes	
`void setTypeMap (Map<String,Class<?>> map)`	3.0	No	
`boolean isWrapperFor (Class<?> iface)`	4.0	Yes	
`<T> T unwrap(Class<T> iface)`	4.0	Yes	

DatabaseMetaData

Table 6 lists the methods that belong to the DatabaseMetaData interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the DatabaseMetaData interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/java/sql/DatabaseMetaData.html.

Table 6. Methods in the DatabaseMetaData Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`boolean allProceduresAreCallable()`	3.0	Yes	Returns true
`boolean allTablesAreSelectable()`	3.0	Yes	Returns true
`boolean autoCommitFailureClosesAllResultSets()`	4.0	Yes	Returns true
`boolean dataDefinitionCausesTransactionCommit()`	3.0	Yes	Returns false
`boolean dataDefinitionIgnoredInTransactions()`	3.0	Yes	Returns false
`boolean deletesAreDetected(int type)`	3.0	Yes	Returns true
`boolean doesMaxRowSizeIncludeBlobs()`	3.0	Yes	Returns false
`boolean generatedKeyAlwaysReturned()`	4.1	Yes	
`ResultSet getAttributes(String catalog, String schemaPattern, String typeNamePattern, String attributeNamePattern)`	3.0	Yes	
`ResultSet getBestRowIdentifier(String catalog, String schema, String table, int scope, boolean nullable)`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`ResultSet getCatalogs()`	3.0	Yes	
`String getCatalogSeparator()`	3.0	Yes	
`String getCatalogTerm()`	3.0	Yes	
`ResultSet getClientInfoProperties()`	4.0	Yes	
`ResultSet getColumnPrivileges(String catalog, String schema, String table, String columnNamePattern)`	3.0	Yes	
`ResultSet getColumns(String catalog, String schemaPattern, String tableNamePattern, String columnNamePattern)`	3.0	Yes	
`Connection getConnection()`	3.0	Yes	
`ResultSet getCrossReference(String primaryCatalog, String primarySchema, String primaryTable, String foreignCatalog, String foreignSchema, String foreignTable)`	3.0	Yes	
`int getDatabaseMajorVersion()`	3.0	Yes	
`int getDatabaseMinorVersion()`	3.0	Yes	
`String getDatabaseProductName()`	3.0	Yes	Hard-coded to Cloudera Impala
`String getDatabaseProductVersion()`	3.0	Yes	
`int getDefaultTransactionIsolation()`	3.0	Yes	Hard-coded to TRANSACTION_READ_UNCOMMITTED
`int getDriverMajorVersion()`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`int getDriverMinorVersion()`	3.0	Yes	
`String getDriverName()`	3.0	Yes	Hard-coded to ImpalaJDBC
`String getDriverVersion()`	3.0	Yes	
`ResultSet getExportedKeys(String catalog, String schema, String table)`	3.0	Yes	
`String getExtraNameCharacters()`	3.0	Yes	Returns an empty String.
`ResultSet getFunctionColumns(String catalog, String schemaPattern, String functionNamePattern, String columnNamePattern)`	4.0	Yes	
`ResultSet getFunctions(String catalog, String schemaPattern, String functionNamePattern)`	4.0	Yes	
`String getIdentifierQuoteString()`	3.0	Yes	Returns a backquote (`)
`ResultSet getImportedKeys(String catalog, String schema, String table)`	3.0	Yes	
`ResultSet getIndexInfo(String catalog, String schema, String table, boolean unique, boolean approximate)`	3.0	Yes	
`int getJDBCMajorVersion()`	3.0	Yes	
`int getJDBCMinorVersion()`	3.0	Yes	
`int getMaxBinaryLiteralLength()`	3.0	Yes	Returns 0
`int getMaxCatalogNameLength()`	3.0	Yes	Returns 128

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`int getMaxCharLiteralLength()`	3.0	Yes	Returns 0
`int getMaxColumnNameLength()`	3.0	Yes	Returns 128
`int getMaxColumnsInGroupBy()`	3.0	Yes	Returns 0
`int getMaxColumnsInIndex()`	3.0	Yes	Returns 0
`int getMaxColumnsInOrderBy()`	3.0	Yes	Returns 0
`int getMaxColumnsInSelect()`	3.0	Yes	Returns 0
`int getMaxColumnsInTable()`	3.0	Yes	Returns 0
`int getMaxConnections()`	3.0	Yes	Returns 0
`int getMaxCursorNameLength()`	3.0	Yes	Returns 0
`int getMaxIndexLength()`	3.0	Yes	Returns 0
`int getMaxProcedureNameLength()`	3.0	Yes	Returns 0
`int getMaxRowSize()`	3.0	Yes	Returns 0
`int getMaxSchemaNameLength()`	3.0	Yes	Returns 128
`int getMaxStatementLength()`	3.0	Yes	Returns 0
`int getMaxStatements()`	3.0	Yes	Returns 0
`int getMaxTableNameLength()`	3.0	Yes	Returns 128
`int getMaxTablesInSelect()`	3.0	Yes	Returns 0
`int getMaxUserNameLength()`	3.0	Yes	Returns 0
`String getNumericFunctions()`	3.0	Yes	Returns the Numeric Functions list

Method	Supported Since JDBC Version	Supported by the Driver	Notes
			from the specification related to the JDBC version of the driver.
`ResultSet getPrimaryKeys(String catalog, String schema, String table)`	3.0	Yes	
`ResultSet getProcedureColumns(String catalog, String schemaPattern, String procedureNamePattern, String columnNamePattern)`	3.0	Yes	
`ResultSet getProcedures(String catalog, String schemaPattern, String procedureNamePattern)`	3.0	Yes	
`String getProcedureTerm()`	3.0	Yes	Returns procedure
`ResultSet getPseudoColumns(String catalog, String schemaPattern, String tableNamePattern, String columnNamePattern)`	4.1	Yes	
`int getResultSetHoldability()`	3.0	Yes	Returns CLOSE_CURSORS_AT_COMMIT
`RowIdLifetime getRowIdLifetime()`	4.0	Yes	Returns ROWID_UNSUPPORTED
`ResultSet getSchemas()`	3.0	Yes	
`ResultSet getSchemas(String catalog, String schemaPattern)`	4.0	Yes	
`String getSchemaTerm()`	3.0	Yes	Returns schema

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`String getSearchStringEscape()`	3.0	Yes	Returns a backslash (\)
`String getSQLKeywords()`	3.0	Yes	Returns an empty String.
`int getSQLStateType()`	3.0	Yes	Returns sqlStateSQL99
`String getStringFunctions()`	3.0	Yes	Returns the String Functions list from the specification related to the JDBC version of the driver.
`ResultSet getSuperTables(String catalog, String schemaPattern, String tableNamePattern)`	3.0	Yes	
`ResultSet getSuperTypes(String catalog, String schemaPattern, String typeNamePattern)`	3.0	Yes	
`String getSystemFunctions()`	3.0	Yes	Returns DATABASE,IFNULL,USER
`ResultSet getTablePrivileges(String catalog, String schemaPattern, String tableNamePattern)`	3.0	Yes	
`ResultSet getTables(String catalog, String schemaPattern, String tableNamePattern, String[] types)`	3.0	Yes	
`ResultSet getTableTypes()`	3.0	Yes	
`String getTimeDateFunctions()`	3.0	Yes	Returns the

Method	Supported Since JDBC Version	Supported by the Driver	Notes
			Time and Date Functions list from the specification related to the JDBC version of the driver.
`ResultSet getTypeInfo()`	3.0	Yes	
`ResultSet getUDTs(String catalog, String schemaPattern, String typeNamePattern, int[] types)`	3.0	Yes	
`String getURL()`	3.0	Yes	
`String getUserName()`	3.0	Yes	
`ResultSet getVersionColumns(String catalog, String schema, String table)`	3.0	Yes	
`boolean insertsAreDetected(int type)`	3.0	Yes	
`boolean isCatalogAtStart()`	3.0	Yes	
`boolean isReadOnly()`	3.0	Yes	Returns true
`boolean locatorsUpdateCopy()`	3.0	Yes	Returns false
`boolean nullPlusNonNullIsNull()`	3.0	Yes	Returns true
`boolean nullsAreSortedAtEnd()`	3.0	Yes	Returns false
`boolean nullsAreSortedAtStart()`	3.0	Yes	Returns false
`boolean nullsAreSortedHigh()`	3.0	Yes	Returns false
`boolean nullsAreSortedLow()`	3.0	Yes	Returns true
`boolean othersDeletesAreVisible(int`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
type)			
boolean othersInsertsAreVisible(int type)	3.0	Yes	
boolean othersUpdatesAreVisible(int type)	3.0	Yes	
boolean ownDeletesAreVisible(int type)	3.0	Yes	
boolean ownInsertsAreVisible(int type)	3.0	Yes	
boolean ownUpdatesAreVisible(int type)	3.0	Yes	
boolean storesLowerCaseIdentifiers()	3.0	Yes	Returns false
boolean storesLowerCaseQuotedIdentifiers()	3.0	Yes	Returns false
boolean storesMixedCaseIdentifiers()	3.0	Yes	Returns true
boolean storesMixedCaseQuotedIdentifiers()	3.0	Yes	Returns true
boolean storesUpperCaseIdentifiers()	3.0	Yes	Returns false
boolean storesUpperCaseQuotedIdentifiers()	3.0	Yes	Returns false
boolean supportsAlterTableWithAddColumn()	3.0	Yes	Returns false
boolean supportsAlterTableWithDropColumn()	3.0	Yes	Returns false
boolean supportsANSI92EntryLevelSQL()	3.0	Yes	Returns true
boolean supportsANSI92FullSQL()	3.0	Yes	Returns false
boolean supportsANSI92IntermediateSQL	3.0	Yes	Returns false

Method	Supported Since JDBC Version	Supported by the Driver	Notes
()			
boolean supportsBatchUpdates()	3.0	Yes	Returns false
boolean supportsCatalogsInDataManipulation()	3.0	Yes	Returns true
boolean supportsCatalogsInIndexDefinitions()	3.0	Yes	Returns true
boolean supportsCatalogsInPrivilegeDefinitions()	3.0	Yes	Returns true
boolean supportsCatalogsInProcedureCalls()	3.0	Yes	Returns true
boolean supportsCatalogsInTableDefinitions()	3.0	Yes	Returns true
boolean supportsColumnAliasing()	3.0	Yes	Returns true
boolean supportsConvert()	3.0	Yes	Returns true
boolean supportsConvert(int fromType, int toType)	3.0	Yes	
boolean supportsCoreSQLGrammar()	3.0	Yes	Returns true
boolean supportsCorrelatedSubqueries()	3.0	Yes	Returns true
boolean supportsDataDefinitionAndDataManipulationTransactions()	3.0	Yes	Returns false
boolean supportsDataManipulationTransactionsOnly()	3.0	Yes	Returns false
boolean supportsDifferentTableCorrelationNames	3.0	Yes	Returns false

Method	Supported Since JDBC Version	Supported by the Driver	Notes
()			
boolean supportsExpressionsInOrderBy()	3.0	Yes	Returns true
boolean supportsExtendedSQLGrammar()	3.0	Yes	Returns false
boolean supportsFullOuterJoins()	3.0	Yes	Returns true
boolean supportsGetGeneratedKeys()	3.0	Yes	Returns false
boolean supportsGroupBy()	3.0	Yes	Returns true
boolean supportsGroupByBeyondSelect()	3.0	Yes	Returns true
boolean supportsGroupByUnrelated()	3.0	Yes	Returns false
boolean supportsIntegrityEnhancementFacility()	3.0	Yes	Returns false
boolean supportsLikeEscapeClause()	3.0	Yes	Returns true
boolean supportsLimitedOuterJoins()	3.0	Yes	Returns false
boolean supportsMinimumSQLGrammar()	3.0	Yes	Returns true
boolean supportsMixedCaseIdentifiers()	3.0	Yes	Returns false
boolean supportsMixedCaseQuotedIdentifiers()	3.0	Yes	Returns true
boolean supportsMultipleOpenResults()	3.0	Yes	Returns false
boolean supportsMultipleResultSets()	3.0	Yes	Returns false
boolean supportsMultipleTransactions()	3.0	Yes	Returns true
boolean supportsNamedParameters()	3.0	Yes	Returns false
boolean supportsNonNullableColumns()	3.0	Yes	Returns false

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`boolean supportsOpenCursorsAcrossCommit()`	3.0	Yes	Returns false
`boolean supportsOpenCursorsAcrossRollback()`	3.0	Yes	Returns false
`boolean supportsOpenStatementsAcrossCommit()`	3.0	Yes	Returns true
`boolean supportsOpenStatementsAcrossRollback()`	3.0	Yes	Returns true
`boolean supportsOrderByUnrelated()`	3.0	Yes	Returns false
`boolean supportsOuterJoins()`	3.0	Yes	Returns false
`boolean supportsPositionedDelete()`	3.0	Yes	Returns false
`boolean supportsPositionedUpdate()`	3.0	Yes	Returns false
`boolean supportsResultSetConcurrency (int type, int concurrency)`	3.0	Yes	
`boolean supportsResultSetHoldability (int holdability)`	3.0	Yes	
`boolean supportsResultSetType(int type)`	3.0	Yes	
`boolean supportsSavepoints()`	3.0	Yes	Returns false
`boolean supportsSchemasInDataManipulation()`	3.0	Yes	Returns true
`boolean supportsSchemasInIndexDefinitions()`	3.0	Yes	Returns true
`boolean supportsSchemasInPrivilegeDefinitions ()`	3.0	Yes	Returns true

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`boolean supportsSchemasInProcedureCalls()`	3.0	Yes	Returns false
`boolean supportsSchemasInTableDefinitions()`	3.0	Yes	Returns true
`boolean supportsSelectForUpdate()`	3.0	Yes	Returns false
`boolean supportsStatementPooling()`	3.0	Yes	Returns false
`boolean supportsStoredFunctionsUsingCallSyntax()`	4.0	Yes	Returns false
`boolean supportsStoredProcedures()`	3.0	Yes	Returns true
`boolean supportsSubqueriesInComparisons()`	3.0	Yes	Returns true
`boolean supportsSubqueriesInExists()`	3.0	Yes	Returns true
`boolean supportsSubqueriesInIns()`	3.0	Yes	Returns true
`boolean supportsSubqueriesInQuantifieds()`	3.0	Yes	Returns true
`boolean supportsTableCorrelationNames()`	3.0	Yes	Returns true
`boolean supportsTransactionIsolationLevel(int level)`	3.0	Yes	
`boolean supportsTransactions()`	3.0	Yes	Returns false
`boolean supportsUnion()`	3.0	Yes	Returns true
`boolean supportsUnionAll()`	3.0	Yes	Returns true
`boolean updatesAreDetected(int type)`	3.0	Yes	Returns true

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`boolean usesLocalFilePerTable()`	3.0	Yes	Returns false
`boolean usesLocalFiles()`	3.0	Yes	Returns false
`boolean isWrapperFor(Class<?> iface)`	4.0	Yes	
`<T> T unwrap(Class<T> iface)`	4.0	Yes	

DataSource

Table 7 lists the methods that belong to the DataSource interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the DataSource interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/javax/sql/DataSource.html.

Table 7. Methods in the DataSource Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`Connection getConnection ()`	3.0	Yes	
`Connection getConnection (String username, String password)`	3.0	Yes	
`int getLoginTimeout()`	3.0	Yes	
`PrintWriter getLogWriter ()`	3.0	Yes	
`Logger getParentLogger()`	4.1	No	The driver does not use java.util.logging

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void setLoginTimeout(int seconds)`	3.0	Yes	
`void setLogWriter (PrintWriter out)`	3.0	Yes	
`boolean isWrapperFor (Class<?> iface)`	4.0	Yes	
`<T> T unwrap(Class<T> iface)`	4.0	Yes	

Driver

Table 8 lists the methods that belong to the Driver interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the Driver interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/java/sql/Driver.html.

Table 8. Methods in the Driver Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`boolean acceptsURL (String url)`	3.0	Yes	
`Connection connect (String url, Properties info)`	3.0	Yes	
`int getMajorVersion()`	3.0	Yes	
`int getMinorVersion()`	3.0	Yes	
`Logger getParentLogger()`	4.1	No	
`DriverPropertyInfo[]`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`getPropertyInfo(String url, Properties info)`			
`boolean jdbcCompliant()`	3.0	Yes	

ParameterMetaData

Table 9 lists the methods that belong to the ParameterMetaData interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the ParameterMetaData interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/java/sql/ParameterMetaData.html.

Table 9. Methods in the ParameterMetaData Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`String getParameterClassName (int param)`	3.0	Yes	
`int getParameterCount()`	3.0	Yes	
`int getParameterMode(int param)`	3.0	Yes	
`int getParameterType(int param)`	3.0	Yes	
`String getParameterTypeName(int param)`	3.0	Yes	
`int getPrecision(int param)`	3.0	Yes	
`int getScale(int param)`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`int isNullable(int param)`	3.0	Yes	
`boolean isSigned(int param)`	3.0	Yes	
`boolean isWrapperFor (Class<?> iface)`	4.0	Yes	
`<T> T unwrap(Class<T> iface)`	4.0	Yes	

PooledConnection

Table 10 lists the methods that belong to the PooledConnection interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the PooledConnection interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/javax/sql/PooledConnection.html.

Table 10. Methods in the PooledConnection Interface

Method	Sup-ported Since JDBC Version	Sup-ported by the Driver	Notes
`void addConnectionEventListener (ConnectionEventListener listener)`	3.0	Yes	
`void addStatementEventListener (StatementEventListener listener)`	4.0	Yes	
`void close()`	3.0	Yes	

Method	Sup-ported Since JDBC Version	Sup-ported by the Driver	Notes
`Connection getConnection()`	3.0	Yes	
`void removeConnectionEventListener(ConnectionEventListener listener)`	3.0	Yes	
`void removeStatementEventListener(StatementEventListener listener)`	4.0	Yes	Removes the specified StatementEventListener from the list of components that will be notified when the driver detects that a PreparedStatement has been closed or is invalid.

PreparedStatement

The PreparedStatement interface extends the Statement interface.

Table 11 lists the methods that belong to the PreparedStatement interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the PooledConnection interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/java/sql/PreparedStatement.html.

Table 11. Methods in the PreparedStatement Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void addBatch()`	3.0	Yes	
`void clearParameters()`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`boolean execute()`	3.0	Yes	
`ResultSet executeQuery()`	3.0	Yes	
`int executeUpdate()`	3.0	Yes	
`ResultSetMetaData getMetaData()`	3.0	Yes	
`ParameterMetaData getParameterMetaData()`	3.0	Yes	
`void setArray(int parameterIndex, Array x)`	3.0	No	
`void setAsciiStream(int parameterIndex, InputStream x)`	4.0	Yes	
`void setAsciiStream(int parameterIndex, InputStream x, int length)`	3.0	Yes	
`void setAsciiStream(int parameterIndex, InputStream x, long length)`	4.0	Yes	
`void setBigDecimal(int parameterIndex, BigDecimal x)`	3.0	Yes	
`void setBinaryStream(int parameterIndex, InputStream x)`	4.0	Yes	
`void setBinaryStream(int parameterIndex, InputStream x, int length)`	3.0	Yes	
`void setBinaryStream(int`	4.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`parameterIndex, InputStream x, long length)`			
`void setBlob(int parameterIndex, Blob x)`	3.0	No	
`void setBlob(int parameterIndex, InputStream inputStream)`	4.0	No	
`void setBlob(int parameterIndex, InputStream inputStream, long length)`	4.0	No	
`void setBoolean(int parameterIndex, boolean x)`	3.0	Yes	
`void setByte(int parameterIndex, byte x)`	3.0	Yes	
`void setBytes(int parameterIndex, byte[] x)`	3.0	Yes	
`void setCharacterStream (int parameterIndex, Reader reader)`	4.0	Yes	
`void setCharacterStream (int parameterIndex, Reader reader, int length)`	3.0	Yes	
`void setCharacterStream (int parameterIndex, Reader reader, long length)`	4.0	Yes	
`void setClob(int parameterIndex, Clob x)`	3.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void setClob(int parameterIndex, Reader reader)`	4.0	No	
`void setClob(int parameterIndex, Reader reader, long length)`	4.0	No	
`void setDate(int parameterIndex, Date x)`	3.0	Yes	
`void setDate(int parameterIndex, Date x, Calendar cal)`	3.0	Yes	
`void setDouble(int parameterIndex, double x)`	3.0	Yes	
`void setFloat(int parameterIndex, float x)`	3.0	Yes	
`void setInt(int parameterIndex, int x)`	3.0	Yes	
`void setLong(int parameterIndex, long x)`	3.0	Yes	
`void setNCharacterStream (int parameterIndex, Reader value)`	4.0	No	
`void setNCharacterStream (int parameterIndex, Reader value, long length)`	4.0	No	
`void setNClob(int parameterIndex, NClob value)`	4.0	No	
`void setNClob(int parameterIndex, Reader reader)`	4.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void setNClob(int parameterIndex, Reader reader, long length)`	4.0	No	
`void setNString(int parameterIndex, String value)`	4.0	No	
`void setNull(int paramIndex, int sqlType, String typeName)`	3.0	Yes	
`void setObject(int parameterIndex, Object x)`	3.0	Yes	
`void setObject(int parameterIndex, Object x, int targetSqlType)`	3.0	Yes	
`void setObject(int parameterIndex, Object x, int targetSqlType, int scale)`	3.0	Yes	
`void setRef(int parameterIndex, Ref x)`	3.0	No	
`void setRowId(int parameterIndex, RowId x)`	4.0	No	
`void setShort(int parameterIndex, short x)`	3.0	No	
`void setSQLXML(int parameterIndex, SQLXML xmlObject)`	4.0	Yes	
`void setString(int parameterIndex, String x)`	3.0	Yes	
`void setTime(int parameterIndex, Time x)`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void setTime(int parameterIndex, Time x, Calendar cal)`	3.0	Yes	
`void setTimestamp(int parameterIndex, Timestamp x)`	3.0	Yes	
`void setTimestamp(int parameterIndex, Timestamp x, Calendar cal)`	3.0	Yes	
`void setUnicodeStream (int parameterIndex, InputStream x, int length)`	3.0	Yes	Deprecated
`void setURL(int parameterIndex, URL x)`	3.0	No	
`boolean isWrapperFor (Class<?> iface)`	4.0	Yes	
`<T> T unwrap(Class<T> iface)`	4.0	Yes	

ResultSet

Table 12 lists the methods that belong to the ResultSet interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the ResultSet interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/java/sql/ResultSet.html.

Table 12. Methods in the ResultSet Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`boolean absolute(int row)`	3.0	No	
`void afterLast()`	3.0	No	
`void beforeFirst()`	3.0	No	
`void cancelRowUpdates()`	3.0	No	Not valid because the driver is read-only.
`void clearWarnings()`	3.0	Yes	
`void close()`	3.0	Yes	
`void deleteRow()`	3.0	No	Not valid because the driver is read-only.
`int findColumn(String columnName)`	3.0	Yes	
`boolean first()`	3.0	No	
`Array getArray(int i)`	3.0	No	
`Array getArray(String colName)`	3.0	No	
`InputStream getAsciiStream(int columnIndex)`	3.0	Yes	
`InputStream getAsciiStream(String columnName)`	3.0	Yes	
`BigDecimal getBigDecimal (int columnIndex)`	3.0	Yes	
`BigDecimal getBigDecimal (int columnIndex, int scale)`	3.0	Yes	Deprecated

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`BigDecimal getBigDecimal (String columnName)`	3.0	Yes	
`BigDecimal getBigDecimal (String columnName, int scale)`	3.0	Yes	Deprecated
`InputStream getBinaryStream(int columnIndex)`	3.0	Yes	
`InputStream getBinaryStream(String columnName)`	3.0	Yes	
`Blob getBlob(int i)`	3.0	No	
`Blob getBlob(String colName)`	3.0	No	
`boolean getBoolean(int columnIndex)`	3.0	Yes	
`boolean getBoolean(String columnName)`	3.0	Yes	
`getByte(int columnIndex)`	3.0	Yes	
`byte getByte(String columnName)`	3.0	Yes	
`byte[] getBytes(int columnIndex)`	3.0	Yes	
`byte[] getBytes(String columnName)`	3.0	Yes	
`Reader getCharacterStream (int columnIndex)`	3.0	Yes	
`Reader getCharacterStream (String columnName)`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`Clob getClob(int i)`	3.0	No	
`Clob getClob(String colName)`	3.0	No	
`int getConcurrency()`	3.0	Yes	
`String getCursorName()`	3.0	Yes	
`Date getDate(int columnIndex)`	3.0	Yes	
`Date getDate(int columnIndex, Calendar cal)`	3.0	Yes	
`Date getDate(String columnName)`	3.0	Yes	
`Date getDate(String columnName, Calendar cal)`	3.0	Yes	
`double getDouble(int columnIndex)`	3.0	Yes	
`double getDouble(String columnName)`	3.0	Yes	
`int getFetchDirection()`	3.0	Yes	
`int getFetchSize()`	3.0	Yes	
`float getFloat(int columnIndex)`	3.0	Yes	
`float getFloat(String columnName)`	3.0	Yes	
`int getHoldability()`	4.0	Yes	
`int getInt(int columnIndex)`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`int getInt(String columnName)`	3.0	Yes	
`long getLong(int columnIndex)`	3.0	Yes	
`long getLong(String columnName)`	3.0	Yes	
`ResultSetMetaData getMetaData()`	3.0	Yes	
`Reader getNCharacterStream(int columnIndex)`	4.0	No	
`Reader getNCharacterStream (String columnLabel`	4.0	No	
`NClob getNClob(int columnIndex)`	4.0	No	
`NClob getNClob(String columnLabel)`	4.0	No	
`String getNString(int columnIndex)`	4.0	No	
`String getNString(String columnLabel)`	4.0	No	
`Object getObject(int columnIndex)`	3.0	Yes	
`<T> T getObject(int columnIndex, Class<T> type)`	4.1	No	
`Object getObject(int i, Map<String,Class<?>> map)`	3.0	No	
`Object getObject(String`	3.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
columnName)			
<T> T getObject(String columnName, Class<T> type)	4.1	No	
Object getObject(String colName, Map<String,Class<?>> map)	3.0	Yes	
Ref getRef(int i)	3.0	No	
Ref getRef(String colName)	3.0	No	
int getRow()	3.0	Yes	
RowId getRowId(int columnIndex)	4.0	No	
RowId getRowId(String columnLabel)	4.0	No	
short getShort(int columnIndex)	3.0	Yes	
short getShort(String columnName)	3.0	Yes	
SQLXML getSQLXML(int columnIndex)	4.0	No	
SQLXML getSQLXML(String columnLabel)	4.0	No	
Statement getStatement()	3.0	Yes	
String getString(int columnIndex)	3.0	Yes	
String getString(String columnName)	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`Time getTime(int columnIndex)`	3.0	Yes	
`Time getTime(int columnIndex, Calendar cal)`	3.0	Yes	
`Time getTime(String columnName)`	3.0	Yes	
`Time getTime(String columnName, Calendar cal)`	3.0	Yes	
`Timestamp getTimestamp (int columnIndex)`	3.0	Yes	
`Timestamp getTimestamp (int columnIndex, Calendar cal)`	3.0	Yes	
`Timestamp getTimestamp (String columnName)`	3.0	Yes	
`Timestamp getTimestamp (String columnName, Calendar cal)`	3.0	Yes	
`int getType()`	3.0	Yes	
`InputStream getUnicodeStream(int columnIndex)`	3.0	Yes	Deprecated
`InputStream getUnicodeStream(String columnName)`	3.0	Yes	Deprecated
`URL getURL(int columnIndex)`	3.0	No	
`URL getURL(String columnName)`	3.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`SQLWarning getWarnings()`	3.0	Yes	
`void insertRow()`	3.0	No	Not valid because the driver is read-only.
`boolean isAfterLast()`	3.0	Yes	
`boolean isBeforeFirst()`	3.0	Yes	
`boolean isClosed()`	4.0	Yes	
`boolean isFirst()`	3.0	Yes	
`boolean isLast()`	3.0	No	
`boolean last()`	3.0	No	
`void moveToCurrentRow()`	3.0	No	Not valid because the driver is read-only.
`void moveToInsertRow()`	3.0	No	Not valid because the driver is read-only.
`boolean next()`	3.0	Yes	
`boolean previous()`	3.0	No	
`void refreshRow()`	3.0	No	
`boolean relative(int rows)`	3.0	No	
`boolean rowDeleted()`	3.0	Yes	Hard-coded to false
`boolean rowInserted()`	3.0	Yes	Hard-coded to false
`boolean rowUpdated()`	3.0	Yes	Hard-coded to false
`void setFetchDirection (int direction)`	3.0	No	Not valid because the driver is forward-only.

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void setFetchSize(int rows)`	3.0	Yes	
`void updateArray(int columnIndex, Array x)`	3.0	No	
`void updateArray(String columnName, Array x)`	3.0	No	
`void updateAsciiStream (int columnIndex, InputStream x)`	4.0	No	Not valid because the driver is read-only.
`void updateAsciiStream (int columnIndex, InputStream x, int length)`	3.0	No	Not valid because the driver is read-only.
`void updateAsciiStream (int columnIndex, InputStream x, long length)`	4.0	No	Not valid because the driver is read-only.
`void updateAsciiStream (String columnName, InputStream x)`	4.0	No	Not valid because the driver is read-only.
`void updateAsciiStream (String columnName, InputStream x, int length)`	3.0	No	Not valid because the driver is read-only.
`void updateAsciiStream (String columnName, InputStream x, long length)`	4.0	No	Not valid because the driver is read-only.
`void updateBigDecimal(int columnIndex, BigDecimal x)`	3.0	No	Not valid because the driver is read-only.
`void updateBigDecimal (String columnName, BigDecimal x)`	3.0	No	Not valid because the driver is read-only.

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void updateBinaryStream (int columnIndex, InputStream x)`	4.0	No	Not valid because the driver is read-only.
`void updateBinaryStream (int columnIndex, InputStream x, int length)`	3.0	No	Not valid because the driver is read-only.
`void updateBinaryStream (int columnIndex, InputStream x, long length)`	4.0	No	Not valid because the driver is read-only.
`void updateBinaryStream (String columnName, InputStream x)`	4.0	No	Not valid because the driver is read-only.
`void updateBinaryStream (String columnName, InputStream x, int length)`	3.0	No	Not valid because the driver is read-only.
`void updateBinaryStream (String columnName, InputStream x, long length)`	4.0	No	Not valid because the driver is read-only.
`void updateBlob(int columnIndex, InputStream inputStream)`	4.0	No	
`void updateBlob(int columnIndex, Blob x)`	3.0	No	
`void updateBlob(int columnIndex, InputStream inputStream, long length)`	4.0	No	
`void updateBlob(String columnName, InputStream inputStream)`	4.0	No	
`void updateBlob(String`	3.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
columnName, Blob x)			
void updateBlob(String columnLabel, InputStream inputStream, long length)	4.0	No	
void updateBoolean(int columnIndex, boolean x)	3.0	No	Not valid because the driver is read-only.
void updateBoolean(String columnName, boolean x)	3.0	No	Not valid because the driver is read-only.
void updateByte(int columnIndex, byte x)	3.0	No	Not valid because the driver is read-only.
void updateByte(String columnName, byte x)	3.0	No	Not valid because the driver is read-only.
void updateBytes(int columnIndex, byte[] x)	3.0	No	Not valid because the driver is read-only.
void updateBytes(String columnName, byte[] x)	3.0	No	Not valid because the driver is read-only.
void updateCharacterStream(int columnIndex, Reader x, int length)	3.0	No	Not valid because the driver is read-only.
void updateCharacterStream (String columnName, Reader reader, int length)	3.0	No	Not valid because the driver is read-only.
void updateBlob(int columnIndex, InputStream inputStream)	4.0	No	
void updateClob(int columnIndex, Clob x)	3.0	No	
void updateBlob(int	4.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`columnIndex, InputStream inputStream, long length)`			
`void updateBlob(String columnName, InputStream inputStream)`	4.0	No	
`void updateClob(String columnName, Clob x)`	3.0	No	
`void updateBlob(String columnName, InputStream inputStream, long length)`	4.0	No	
`void updateDate(int columnIndex, Date x)`	3.0	No	Not valid because the driver is read-only.
`void updateDate(String columnName, Date x)`	3.0	No	Not valid because the driver is read-only.
`void updateDouble(int columnIndex, double x)`	3.0	No	Not valid because the driver is read-only.
`void updateDouble(String columnName, double x)`	3.0	No	Not valid because the driver is read-only.
`void updateFloat(int columnIndex, float x)`	3.0	No	Not valid because the driver is read-only.
`void updateFloat(String columnName, float x)`	3.0	No	Not valid because the driver is read-only.
`void updateInt(int columnIndex, int x)`	3.0	No	Not valid because the driver is read-only.
`void updateInt(String columnName, int x)`	3.0	No	Not valid because the driver is read-only.
`void updateLong(int columnIndex, long x)`	3.0	No	Not valid because the driver is read-only.
`void updateLong(String`	3.0	No	Not valid because the driver

Method	Supported Since JDBC Version	Supported by the Driver	Notes
columnName, long x)			is read-only.
void updateNCharacterStream (int columnIndex, Reader x)	4.0	No	
void updateNCharacterStream (int columnIndex, Reader x, long length)	4.0	No	
void updateNCharacterStream (String columnName, Reader reader)	4.0	No	
void updateNCharacterStream (String columnName, Reader reader, long length)	4.0	No	
void updateNClob(int columnIndex, NClob nClob)	4.0	No	
void updateNClob(int columnIndex, Reader reader)	4.0	No	
void updateNClob(int columnIndex, Reader reader, long length)	4.0	No	
void updateNClob(String columnName, NClob nClob)	4.0	No	
void updateNClob(String columnName, Reader reader)	4.0	No	
void updateNClob(String columnName, Reader reader, long length)	4.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void updateNString(int columnIndex, String nString)`	4.0	No	
`void updateNString(String columnName, String nString)`	4.0	No	
`void updateNull(int columnIndex)`	3.0	No	Not valid because the driver is read-only.
`void updateNull(String columnName)`	3.0	No	Not valid because the driver is read-only.
`void updateObject(int columnIndex, Object x)`	3.0	No	Not valid because the driver is read-only.
`void updateObject(int columnIndex, Object x, int scale)`	3.0	No	Not valid because the driver is read-only.
`void updateObject(String columnName, Object x)`	3.0	No	Not valid because the driver is read-only.
`void updateObject(String columnName, Object x, int scale)`	3.0	No	Not valid because the driver is read-only.
`void updateRef(int columnIndex, Ref x)`	3.0	No	Not valid because the driver is read-only.
`void updateRef(String columnName, Ref x)`	3.0	No	Not valid because the driver is read-only.
`void updateRow()`	3.0	No	Not valid because the driver is read-only.
`void updateRowId(int columnIndex, RowId x)`	4.0	No	
`void updateRowId(String columnName, RowId x)`	4.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void updateShort(int columnIndex, short x)`	3.0	No	Not valid because the driver is read-only.
`void updateShort(String columnName, short x)`	3.0	No	Not valid because the driver is read-only.
`void updateSQLXML(int columnIndex, SQLXML xmlObject)`	4.0	No	
`void updateSQLXML(String columnName, SQLXML xmlObject)`	4.0	No	
`void updateString(int columnIndex, String x)`	3.0	No	Not valid because the driver is read-only.
`void updateString(String columnName, String x)`	3.0	No	Not valid because the driver is read-only.
`void updateTime(int columnIndex, Time x)`	3.0	No	Not valid because the driver is read-only.
`void updateTime(String columnName, Time x)`	3.0	No	Not valid because the driver is read-only.
`void updateTimestamp(int columnIndex, Timestamp x)`	3.0	No	Not valid because the driver is read-only.
`void updateTimestamp (String columnName, Timestamp x)`	3.0	No	Not valid because the driver is read-only.
`boolean wasNull()`	3.0	Yes	
`boolean isWrapperFor (Class<?> iface)`	4.0	Yes	
`<T> T unwrap(Class<T> iface)`	4.0	Yes	

ResultSetMetaData

Table 13 lists the methods that belong to the ResultSetMetaData interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the ResultSetMetaData interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/java/sql/ResultSetMetaData.html.

Table 13. Methods in the ResultSetMetaData Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`String getCatalogName (int column)`	3.0	Yes	
`String getColumnClassName(int column)`	3.0	Yes	
`int getColumnCount()`	3.0	Yes	
`int getColumnDisplaySize (int column)`	3.0	Yes	
`String getColumnLabel (int column)`	3.0	Yes	
`String getColumnName(int column)`	3.0	Yes	
`int getColumnType(int column)`	3.0	Yes	
`String getColumnTypeName (int column)`	3.0	Yes	
`int getPrecision(int column)`	3.0	Yes	
`int getScale(int column)`	3.0	Yes	
`String getSchemaName(int column)`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`String getTableName(int column)`	3.0	Yes	
`boolean isAutoIncrement (int column)`	3.0	Yes	
`boolean isCaseSensitive (int column)`	3.0	Yes	
`boolean isCurrency(int column)`	3.0	Yes	
`boolean isDefinitelyWritable(int column)`	3.0	Yes	
`int isNullable(int column)`	3.0	Yes	
`boolean isReadOnly(int column)`	3.0	Yes	
`boolean isSearchable(int column)`	3.0	Yes	
`boolean isSigned(int column)`	3.0	Yes	
`boolean isWritable(int column)`	3.0	Yes	
`boolean isWrapperFor (Class<?> iface)`	4.0	Yes	
`<T> T unwrap(Class<T> iface)`	4.0	Yes	

Statement

Table 14 lists the methods that belong to the Statement interface, and describes whether each method is supported by the Cloudera JDBC Driver for Apache Hive and which version of the JDBC API is the earliest version that supports the method.

For detailed information about each method in the Statement interface, see the Java API documentation available at http://docs.oracle.com/javase/1.5.0/docs/api/java/sql/Statement.html.

Table 14. Methods in the Statement Interface

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`void addBatch(String sql)`	3.0	Yes	
`void cancel()`	3.0	Yes	
`void clearBatch()`	3.0	Yes	
`void clearWarnings()`	3.0	Yes	
`void close()`	3.0	Yes	
`void closeOnCompletion()`	4.1	Yes	
`boolean execute(String sql)`	3.0	Yes	
`boolean execute(String sql, int autoGeneratedKeys)`	3.0	No	
`boolean execute(String sql, int[] columnIndexes)`	3.0	No	
`boolean execute(String sql, String[] columnNames)`	3.0	No	
`int[]executeBatch()`	3.0	No	
`ResultSet executeQuery (String sql)`	3.0	Yes	
`int executeUpdate(String sql)`	3.0	Yes	
`int executeUpdate(String sql, int autoGeneratedKeys)`	3.0	No	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`int executeUpdate(String sql, int[] columnIndexes)`	3.0	No	
`int executeUpdate(String sql, String[] columnNames)`	3.0	No	
`Connection getConnection ()`	3.0	Yes	
`int getFetchDirection()`	3.0	Yes	
`int getFetchSize()`	3.0	Yes	
`ResultSet getGeneratedKeys()`	3.0	Yes	
`int getMaxFieldSize()`	3.0	Yes	
`int getMaxRows()`	3.0	Yes	
`boolean getMoreResults()`	3.0	Yes	
`boolean getMoreResults (int current)`	3.0	No	
`int getQueryTimeout()`	3.0	Yes	
`ResultSet getResultSet()`	3.0	Yes	
`int getResultSetConcurrency()`	3.0	Yes	Hard-coded to CONCUR_ READ_ONLY
`int getResultSetHoldability()`	3.0	Yes	Hard-coded to CLOSE_ CURSORS_AT_COMMIT
`int getResultSetType()`	3.0	Yes	Hard-coded to TYPE_ FORWARD_ONLY
`int getUpdateCount()`	3.0	Yes	

Method	Supported Since JDBC Version	Supported by the Driver	Notes
`SQLWarning getWarnings()`	3.0	Yes	
`boolean isClosed()`	4.0	Yes	
`boolean isCloseOnCompletion()`	4.1	Yes	
`boolean isPoolable()`	4.0	Yes	
`void setCursorName(String name)`	3.0	No	
`void setEscapeProcessing (boolean enable)`	3.0	Yes	
`void setFetchDirection (int direction)`	3.0	No	
`void setFetchSize(int rows)`	3.0	Yes	
`void setMaxFieldSize(int max)`	3.0	Yes	
`void setMaxRows(int max)`	3.0	Yes	
`void setPoolable(boolean poolable)`	4.0	Yes	
`void setQueryTimeout(int seconds)`	3.0	Yes	
`boolean isWrapperFor (Class<?> iface)`	4.0	Yes	
`<T> T unwrap(Class<T> iface)`	4.0	Yes	

Contact Us

If you are having difficulties using the driver, our Community Forum may have your solution. In addition to providing user to user support, our forums are a great place to share your questions, comments, and feature requests with us.

If you are a Subscription customer you may also use the Cloudera Support Portal to search the Knowledge Base or file a Case.

Important:

To help us assist you, prior to contacting Cloudera Support please prepare a detailed summary of the client and server environment including operating system version, patch level, and configuration.

Appendix A Driver Configuration Options

Appendix A lists and describes the properties that you can use to configure the behavior of the Cloudera JDBC Driver for Apache Hive.

You can set configuration properties using the connection URL. For more information, see "Building the Connection URL" on page 7.

AllowSelfSignedCerts

Default Value	Required
0	No

Description

When this property is set to 0, the SSL certificate used by the server cannot be self-signed.

When this property is set to 1, the SSL certificate used by the server can be self-signed.

Note:

This property is applicable only when SSL connections are enabled.

AuthMech

Default Value	Required
0	No

Description

The authentication mechanism to use. Set the value to one of the following numbers:

- **0** for No Authentication
- **1** for Kerberos
- **2** for User Name
- **3** for User Name and Password

CAIssuedCertNamesMismatch

Default Value	Required
0	No

Description

When this property is set to 0, the name of the CA-issued SSL certificate must match the host name of the Hive server.

When this property is set to 1, the names of the certificate and the host name of the server are allowed to mismatch.

Note:

This property is applicable only when SSL connections are enabled.

CatalogSchemaSwitch

Default Value	Required
0	No

Description

When this property is set to 1, the driver treats Hive catalogs as schemas as a restriction for filtering.

When this property is set to 0, Hive catalogs are treated as catalogs, and Hive schemas are treated as schemas.

DecimalColumnScale

Default Value	Required
10	No

Description

The maximum number of digits to the right of the decimal point for numeric data types.

DefaultStringColumnLength

Default Value	Required
255	No

Description

The maximum data length for STRING columns. The range of DefaultStringColumnLength is 0 to 32,767.

By default, the columns metadata for Hive does not specify a maximum data length for STRING columns.

DelegationUID

Default Value	Required
None	No

Description

Use this option to delegate all operations against Hive to a user that is different than the authenticated user for the connection.

Note:

This option is applicable only when connecting to a Hive Server 2 instance that supports this feature.

KrbHostFQDN

Default Value	Required
None	Yes, if AuthMech=1 (Kerberos)

Description

The fully qualified domain name of the Hive Server 2 host.

KrbRealm

Default Value	Required
Depends on Kerberos configuration.	No

Description

The realm of the Hive Server 2 host.

If your Kerberos configuration already defines the realm of the Hive Server 2 host as the default realm, then you do not need to configure this option.

KrbServiceName

Default Value	Required
None	Yes, if AuthMech=1 (Kerberos)

Description

The Kerberos service principal name of the Hive server.

PreparedMetaLimitZero

Default Value	Required
0	No

Description

When this property is set to 1, the PreparedStatement.getMetadata() call will request metadata from the server with “LIMIT 0”.

PWD

Default Value	Required
None	Yes, if AuthMech=3 (User Name and Password)

Description

The password corresponding to the user name that you provided using the property "UID" on page 86.

RowsFetchedPerBlock

Default Value	Required
10000	No

Description

The maximum number of rows that a query returns at a time.

Any positive 32-bit integer is a valid value, but testing has shown that performance gains are marginal beyond the default value of 10000 rows.

SocketTimeout

Default Value	Required
0	No

Description

The number of seconds after which Hive closes the connection with the client application if the connection is idle. The default value of 0 indicates that an idle connection is not closed.

SSL

Default Value	Required
0	No

Description

When this property is set to 1, the driver communicates with the Hive server through an SSL-enabled socket.

When this property is set to 0, the driver does not connect to SSL-enabled sockets.

Note:

SSL is configured independently of authentication. When authentication and SSL are both enabled, the driver performs the specified authentication method over an SSL connection.

SSLKeyStore

Default Value	Required
None	Yes, if SSL=1

Description

The full path and file name of the Java KeyStore containing an SSL certificate to use during authentication.

See also the property "SSLKeyStorePwd" on page 85.

SSLKeyStorePwd

Default Value	Required
None	Yes, if SSL=1

Description

The password for accessing the Java KeyStore that you specified using the property "SSLKeyStore" on page 84.

SSLTrustStore

Default Value	Required
jssecacerts, if it exists. If jssecacerts does not exist, then cacerts is used. The default location of cacerts is jre\lib\security\	No

Description

The full path and file name of the Java TrustStore containing an SSL certificate to use during authentication.

See also the property "SSLTrustStorePwd" on page 85.

SSLTrustStorePwd

Default Value	Required
None	Yes, if using a TrustStore.

Description

The password for accessing the Java TrustStore that you specified using the property "SSLTrustStore" on page 85.

UID

Default Value	Required
hive	Yes, if AuthMech=3 (User Name and Password) No, if AuthMech=2 (User Name)

Description

The user name that you use to access the Hive server.

UseNativeQuery

Default Value	Required
0	No

Description

When this option is enabled (1), the driver does not transform the queries emitted by an application, so the native query is used.

When this option is disabled (0), the driver transforms the queries emitted by an application and converts them into an equivalent form in HiveQL.

Note:

If the application is Hive-aware and already emits HiveQL, then enable this option to avoid the extra overhead of query transformation.

zk

Default Value	Required
None	No

Description

The connection string to one or more ZooKeeper quorums, written in the following format:

ZK_IP:*ZK_Port*/*ZK_Namespace*

For example:

```
jdbc:hive2://zk=192.168.0.1:2181/hiveserver2
```

Use this option to enable the Dynamic Service Discovery feature, which allows you to connect to Hive servers that are registered against a ZooKeeper service by connecting to the ZooKeeper service.

You can specify multiple quorums in a comma-separated list. If connection to a quorum fails, the driver will attempt to connect to the next quorum in the list.