



# **TIBCO® Data Virtualization**

## **Apache HBase Adapter Guide**

Version 8.7.0 | October 2023

# Contents

---

<b>Contents</b>	<b>2</b>
<b>HBase Adapter</b>	<b>4</b>
Getting Started	4
Basic Tab	5
Logging	6
Fine-Tuning Data Access	8
Using Kerberos	8
Changelog	10
Advanced Features	13
User Defined Views	14
SSL Configuration	17
Firewall and Proxy	17
Query Processing	18
Logging	18
SQL Compliance	21
SELECT Statements	23
SELECT INTO Statements	25
INSERT Statements	25
UPDATE Statements	26
UPSERT Statements	27
DELETE Statements	27
EXECUTE Statements	28
PIVOT and UNPIVOT	29
Data Model	30
Stored Procedures	30
Connection String Options	34
Authentication	39

Kerberos .....	42
SSL .....	46
Firewall .....	53
Proxy .....	56
Logging .....	63
Schema .....	63
Miscellaneous .....	65
<b>TIBCO Product Documentation and Support Services .....</b>	<b>75</b>
How to Access TIBCO Documentation .....	75
How to Contact TIBCO Support .....	76
Release Version Support .....	76
How to Join TIBCO Community .....	77
<b>Legal and Third-Party Notices .....</b>	<b>78</b>

# HBase Adapter

---

## HBase Version Support

The adapter models HBase data as a read/write, relational database. The adapter can connect to HBase REST Server 0.0.3, available in HBase version 0.98 and above.

## SQL Compliance

The [SQL Compliance](#) section shows the SQL syntax supported by the adapter and points out any limitations.

# Getting Started

## Connecting to HBase

[Basic Tab](#) shows how to authenticate to HBase and configure any necessary connection properties. Additional adapter capabilities can be configured using the available [Connection](#) properties on the Advanced tab. The Advanced Settings section shows how to set up more advanced configurations and troubleshoot connection errors.

## Deploying the HBase Adapter

To deploy the adapter, you can execute the `server_util` utility via the command line by

1. Unzip the `tdv.apachehbase.zip` file to the location of your choice.
2. Open a command prompt window.
3. Navigate to the `<TDV_install_dir>/bin`
4. Enter the `server_util` command with the `-deploy` option:

```
server_util -server <hostname> [-port <port>] -user <user> -  
password <password> -deploy -package <TDV_install_  
dir>/adapters/tdv.apachehbase/tdv.apachehbase.jar
```

Note: When deploying a build of an existing adapter, you will need to undeploy the existing adapter using the `server_util` command with the `-undeploy` option.

```
server_util -server <hostname> [-port <port>] -user <user> -password  
<password> -undeploy -version 1 -name ApacheHBase
```

## Basic Tab

### Before You Connect

Different Hadoop distributions contain different interfaces and means of starting and stopping the HBase REST server, along with different default port settings.

In most distributions, the HBase REST server can be started in the foreground by running the following command: *hbase rest start -p <port>*.

Please consult your Hadoop distribution's documentation for further information regarding the HBase REST server.

### Connecting to HBase

The HBase Adapter connects to Apache HBase via the HBase REST (Stargate) server. Set the following to connect to HBase:

- Server: This will typically be the host name or IP address of the server hosting HBase. If there are multiple nodes, use the host name or IP address of the machine running the REST (Stargate) server.
- Port: Set this to the port for the HBase REST (Stargate) server.

### Authenticating to HBase

The HBase Adapter supports the following authentication schemes:

- Anonymous
- Basic
- Negotiate (Kerberos)

## Anonymous

By default, no authentication (alternatively known as "anonymous" authentication) is used. Set AuthScheme to **None** to explicitly enforce no authentication.

## Basic

To use Basic authentication, set the following:

- AuthScheme: Set this to **Basic**.
- User: Set this to the HBase user.
- Password: Set this to the HBase password.

## Kerberos

To authenticate with Kerberos, set AuthScheme to **NEGOTIATE** and set the User and Password. Please see [Using Kerberos](#) for details on how to authenticate with Kerberos.

## Logging

The adapter uses TDV Server's logging (log4j) to generate log files. The settings within the TDV Server's logging (log4j) configuration file are used by the adapter to determine the type of messages to log. The following categories can be specified:

- Error: Only error messages are logged.
- Info: Both Error and Info messages are logged.
- Debug: Error, Info, and Debug messages are logged.

The Other property of the adapter can be used to set Verbosity to specify the amount of detail to be included in the log file, that is:

```
Verbosity=4;
```

You can use Verbosity to specify the amount of detail to include in the log within a category. The following verbosity levels are mapped to the log4j categories:

- 0 = Error
- 1-2 = Info
- 3-5 = Debug

For example, if the log4j category is set to DEBUG, the Verbosity option can be set to 3 for the minimum amount of debug information or 5 for the maximum amount of debug information.

Note that the log4j settings override the Verbosity level specified. The adapter never logs at a Verbosity level greater than what is configured in the log4j properties. In addition, if Verbosity is set to a level less than the log4j category configured, Verbosity defaults to the minimum value for that particular category. For example, if Verbosity is set to a value less than 3 and the Debug category is specified, the Verbosity defaults to 3.

The following list is an explanation of the Verbosity levels and the information that they log.

- 1 - Will log the query, the number of rows returned by it, the start of execution and the time taken, and any errors.
- 2 - Will log everything included in Verbosity 1 and HTTP headers.
- 3 - Will additionally log the body of the HTTP requests.
- 4 - Will additionally log transport-level communication with the data source. This includes SSL negotiation.
- 5 - Will additionally log communication with the data source and additional details that may be helpful in troubleshooting problems. This includes interface commands.

## Configure Logging for the HBase Adapter

By default, logging is turned on without debugging. If debugging information is desired, uncomment the following line in the TDV Server's log4j.properties file (default location of this file is: C:\Program Files\TIBCO\TDV Server <version>\conf\server):

```
log4j.logger.com.cdata=DEBUG
```

The TDV Server must be restarted after changing the log4j.properties file, which can be accomplished by running the composite.bat script located at: C:\Program Files\TIBCO\TDV Server <version>\bin. Note that reauthenticating to the TDV Studio is required after restarting the server.

Here is an example of the calls:

```
.\composite.bat monitor restart
```

All logs for the adapter are written to the "cs\_server\_dsrc.log" file as specified in the log4j properties.

**Note:** The "log4j.logger.com.cdata=DEBUG" option is not required if the **Debug Output Enabled** option is set to true within the TDV Studio. To set this option, navigate to **Administrator > Configuration**. Select **Server > Configuration > Debugging** and set the Debug Output Enabled option to **True**.

## Fine-Tuning Data Access

### Fine Tuning Data Access

You can use the following properties to configure automatic data type detection, which is enabled by default.

- TypeDetectionScheme: You can use this property to enable or disable automatic type detection based on the value specified in RowScanDepth.
- RowScanDepth: This property determines the number of rows that will be scanned to determine column data types.

## Using Kerberos

This section shows how to use the adapter to authenticate using Kerberos.

### Kerberos

To authenticate to HBase using Kerberos, set the following properties:

- AuthScheme: Set this to **NEGOTIATE**.
- KerberosKDC: Set this to the host name or IP Address of your Kerberos KDC machine.
- KerberosRealm: Set this to **the realm of the HBase Kerberos principal**. This will be the value after the '@' symbol (for instance, EXAMPLE.COM) of the **hbase.regionserver.kerberos.principal of the hbase-site.xml file** (for instance, hbase/MyHost@EXAMPLE.COM).
- KerberosSPN: Set this to the service and host of the HBase Kerberos Principal. This is



the value prior to the '@' symbol (for instance, hbase/MyHost) of the hbase.regionserver.kerberos.principal of the hbase-site.xml file (for instance, hbase/MyHost@EXAMPLE.COM).

## Retrieve the Kerberos Ticket

You can use one of the following options to retrieve the required Kerberos ticket.

### MIT Kerberos Credential Cache File

This option enables you to use the MIT Kerberos Ticket Manager or kinit command to get tickets. Note that you do not need to set the User or Password connection properties with this option.

1. Ensure that you have an environment variable created called **KRB5CCNAME**.
2. Set the **KRB5CCNAME** environment variable to a path pointing to your credential cache file (for instance, C:\krb\_cache\krb5cc\_0 or /tmp/krb5cc\_0). This file is created when generating your ticket with MIT Kerberos Ticket Manager.
3. To obtain a ticket, open the MIT Kerberos Ticket Manager application, click **Get Ticket**, enter your principal name and password, then click **OK**. If successful, ticket information appears in Kerberos Ticket Manager and is stored in the credential cache file.
4. Now that you have created the credential cache file, the adapter uses the cache file to obtain the Kerberos ticket to connect to HBase.

As an alternative to setting the **KRB5CCNAME** environment variable, you can directly set the file path using the KerberosTicketCache property. When set, the adapter uses the specified cache file to obtain the Kerberos ticket to connect to HBase.

### Keytab File

If the **KRB5CCNAME environment variable has not been set**, you can retrieve a Kerberos ticket using a Keytab File. To do so, set the User property to the desired username and set the KerberosKeytabFile property to a file path pointing to the keytab file associated with the user.

## User and Password

If both the **KRB5CCNAME** environment variable and the `KerberosKeytabFile` property have not been set, you can retrieve a ticket using a user and password combination. To do this, set the `User` and `Password` properties to the user/password combination that you use to authenticate with HBase.

## Cross-Realm

More complex Kerberos environments may require cross-realm authentication where multiple realms and KDC servers are used (e.g., where one realm/KDC is used for user authentication and another realm/KDC is used for obtaining the service ticket).

In such an environment, set the `KerberosRealm` and `KerberosKDC` properties to the values required for user authentication. Also set the `KerberosServiceRealm` and `KerberosServiceKDC` properties to the values required to obtain the service ticket.

# Changelog

## General Changes

Date	Build Number	Change Type	Description
12/14/2022	8383	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Added the Default column to the sys_procedureparameters table.</li> </ul>
09/30/2022	8308	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Added the IsPath column to the sys_procedureparameters table.</li> </ul>
08/17/2022	8264	General	<b>Changed</b> <ul style="list-style-type: none"> <li>We now support handling the keyword "COLLATE" as standard function name as</li> </ul>

well.			
03/16/2022	8110	HBase	<b>Changed</b> <ul style="list-style-type: none"> <li>RowScanDepth can now be used to scan all records when the value is set to less than or equal to 0.</li> </ul>
02/11/2022	8077	HBase	<b>Added</b> <ul style="list-style-type: none"> <li>Added the connection property ColumnEncoding</li> </ul>
09/16/2021	7929	HBase	<b>Added</b> <ul style="list-style-type: none"> <li>Added four connection properties SSLClientCert, SSLClientCertPassword, SSLClientCertSubject, SSLClientCertType.</li> </ul>
09/02/2021	7915	General	<b>Added</b> <ul style="list-style-type: none"> <li>Added support for the STRING_SPLIT table-valued function in the CROSS APPLY clause.</li> </ul>
08/07/2021	7889	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Added the KeySeq column to the sys_foreignkeys table.</li> </ul>
08/06/2021	7888	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Added the new sys_primarykeys system table.</li> </ul>
07/23/2021	7874	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Updated the Literal Function Names for relative date/datetime functions. Previously relative date/datetime functions resolved to a different value when used in the projection vs the predicate. I.e: SELECT LAST_MONTH() AS lm, Col FROM Table WHERE</li> </ul>

			<p>Col &gt; LAST_MONTH()). Formerly the two LAST_MONTH() methods would resolve to different datetimes. Now they will match.</p> <ul style="list-style-type: none"> <li>As a replacement for the previous behavior, the relative date/datetime functions in the criteria may have an 'L' appended to them. I.e: WHERE col &gt; L_LAST_MONTH(). This will continue to resolve to the same values that previously were calculated in the criteria. Note that the "L_" prefix will only work in the predicate - it not available for the projection.</li> </ul>
07/08/2021	7859	General	<p><b>Added</b></p> <ul style="list-style-type: none"> <li>Added the TCP Logging Module for the logging information happening on the TCP wire protocol. The transport bytes that are incoming and ongoing will be logged at verbosity=5.</li> </ul>
04/23/2021	7785	General	<p><b>Added</b></p> <ul style="list-style-type: none"> <li>Added support for handling client side formulas during insert / update. For example: UPDATE Table SET Col1 = Concat (Col1, " - ", Col2) WHERE Col2 LIKE 'A%'</li> </ul>
04/23/2021	7783	General	<p><b>Changed</b></p> <ul style="list-style-type: none"> <li>Updated how display sizes are determined for varchar primary key and foreign key columns so they will match the reported length of the column.</li> </ul>
04/16/2021	7776	General	<p><b>Added</b></p> <ul style="list-style-type: none"> <li>Non-conditional updates between two columns is now available to all drivers. For example: UPDATE Table SET Col1=Col2</li> </ul>

**Changed**

- Reduced the length to 255 for varchar primary key and foreign key columns.
- Updated implicit and metadata caching to improve performance and support for multiple connections. Old metadata caches are not compatible - you would need to generate new metadata caches if you are currently using CacheMetadata.
- Updated index naming convention to avoid duplicates
- Updated and standardized Getting Started connection help.
- Added the Advanced Features section to the help of all drivers.
- Categorized connection property listings in the help for all editions.

04/15 /2021

7775

General

**Changed**

- Kerberos authentication is updated to use TCP by default, but will fall back to UDP if a TCP connection cannot be established

## Advanced Features

This section details a selection of advanced features of the HBase adapter.

### User Defined Views

The adapter allows you to define virtual tables, called *user defined views*, whose contents are decided by a pre-configured query. These views are useful when you cannot directly control queries being issued to the drivers. See [User Defined Views](#) for an overview of creating and configuring custom views.

## SSL Configuration

Use [SSL Configuration](#) to adjust how adapter handles TLS/SSL certificate negotiations. You can choose from various certificate formats; see the [SSLServerCert](#) property under "Connection String Options" for more information.

## Firewall and Proxy

Configure the adapter for compliance with [Firewall and Proxy](#), including Windows proxies and HTTP proxies. You can also set up tunnel connections.

## Query Processing

The adapter offloads as much of the SELECT statement processing as possible to HBase and then processes the rest of the query in memory (client-side).

See [Query Processing](#) for more information.

## Logging

See [Logging](#) for an overview of configuration settings that can be used to refine CData logging. For basic logging, you only need to set two connection properties, but there are numerous features that support more refined logging, where you can select subsets of information to be logged using the [LogModules](#) connection property.

## User Defined Views

The HBase Adapter allows you to define a virtual table whose contents are decided by a pre-configured query. These are called *User Defined Views*, which are useful in situations where you cannot directly control the query being issued to the driver, e.g. when using the driver from a tool. The User Defined Views can be used to define predicates that are always applied. If you specify additional predicates in the query to the view, they are combined with the query already defined as part of the view.

There are two ways to create user defined views:

- Create a JSON-formatted configuration file defining the views you want.
- DDL statements.

## Defining Views Using a Configuration File

User Defined Views are defined in a JSON-formatted configuration file called *UserDefinedViews.json*. The adapter automatically detects the views specified in this file.

You can also have multiple view definitions and control them using the UserDefinedViews connection property. When you use this property, only the specified views are seen by the adapter.

This User Defined View configuration file is formatted as follows:

- Each root element defines the name of a view.
- Each root element contains a child element, called **query**, which contains the custom SQL query for the view.

For example:

```
{
  "MyView": {
    "query": "SELECT * FROM Account WHERE MyColumn = 'value'"
  },
  "MyView2": {
    "query": "SELECT * FROM MyTable WHERE Id IN (1,2,3)"
  }
}
```

Use the UserDefinedViews connection property to specify the location of your JSON configuration file. For example:

```
"UserDefinedViews",
"C:\\Users\\yourusername\\Desktop\\tmp\\UserDefinedViews.json"
```

## Defining Views Using DDL Statements

The adapter is also capable of creating and altering the schema via DDL Statements such as CREATE LOCAL VIEW, ALTER LOCAL VIEW, and DROP LOCAL VIEW.

### Create a View

To create a new view using DDL statements, provide the view name and query as follows:

```
CREATE LOCAL VIEW [MyViewName] AS SELECT * FROM Customers LIMIT 20;
```

If no JSON file exists, the above code creates one. The view is then created in the JSON configuration file and is now discoverable. The JSON file location is specified by the UserDefinedViews connection property.

## Alter a View

To alter an existing view, provide the name of an existing view alongside the new query you would like to use instead:

```
ALTER LOCAL VIEW [MyViewName] AS SELECT * FROM Customers WHERE  
TimeModified > '3/1/2020';
```

The view is then updated in the JSON configuration file.

## Drop a View

To drop an existing view, provide the name of an existing schema alongside the new query you would like to use instead.

```
DROP LOCAL VIEW [MyViewName]
```

This removes the view from the JSON configuration file. It can no longer be queried.

## Schema for User Defined Views

User Defined Views are exposed in the **UserViews** schema by default. This is done to avoid the view's name clashing with an actual entity in the data model. You can change the name of the schema used for UserViews by setting the UserViewsSchemaName property.

## Working with User Defined Views

For example, a SQL statement with a User Defined View called *UserViews.RCustomers* only lists customers in Raleigh:

```
SELECT * FROM Customers WHERE City = 'Raleigh';
```

An example of a query to the driver:

```
SELECT * FROM UserViews.RCustomers WHERE Status = 'Active';
```

Resulting in the effective query to the source:



```
SELECT * FROM Customers WHERE City = 'Raleigh' AND Status = 'Active';
```

That is a very simple example of a query to a User Defined View that is effectively a combination of the view query and the view definition. It is possible to compose these queries in much more complex patterns. All SQL operations are allowed in both queries and are combined when appropriate.

## SSL Configuration

### Customizing the SSL Configuration

By default, the adapter attempts to negotiate SSL/TLS by checking the server's certificate against the system's trusted certificate store.

To specify another certificate, see the [SSLServerCert](#) property for the available formats to do so.

## Firewall and Proxy

### Connecting Through a Firewall or Proxy

#### HTTP Proxies

To connect through the Windows system proxy, you do not need to set any additional connection properties. To connect to other proxies, set [ProxyAutoDetect](#) to false.

In addition, to authenticate to an HTTP proxy, set [ProxyAuthScheme](#), [ProxyUser](#), and [ProxyPassword](#), in addition to [ProxyServer](#) and [ProxyPort](#).

#### Other Proxies

Set the following properties:

- To use a proxy-based firewall, set [FirewallType](#), [FirewallServer](#), and [FirewallPort](#).
- To tunnel the connection, set [FirewallType](#) to TUNNEL.

- To authenticate, specify FirewallUser and FirewallPassword.
- To authenticate to a SOCKS proxy, additionally set FirewallType to SOCKS5.

## Query Processing

### Query Processing

CData has a client-side SQL engine built into the adapter library. This enables support for the full capabilities that SQL-92 offers, including filters, aggregations, functions, etc.

For sources that do not support SQL-92, the adapter offloads as much of SQL statement processing as possible to HBase and then processes the rest of the query in memory (client-side). This results in optimal performance.

For data sources with limited query capabilities, the adapter handles transformations of the SQL query to make it simpler for the adapter. The goal is to make smart decisions based on the query capabilities of the data source to push down as much of the computation as possible. The HBase Query Evaluation component examines SQL queries and returns information indicating what parts of the query the adapter is not capable of executing natively.

The HBase Query Slicer component is used in more specific cases to separate a single query into multiple independent queries. The client-side Query Engine makes decisions about simplifying queries, breaking queries into multiple queries, and pushing down or computing aggregations on the client-side while minimizing the size of the result set.

There's a significant trade-off in evaluating queries, even partially, client-side. There are always queries that are impossible to execute efficiently in this model, and some can be particularly expensive to compute in this manner. CData always pushes down as much of the query as is feasible for the data source to generate the most efficient query possible and provide the most flexible query capabilities.

### More Information

For a full discussion of how CData handles query processing, see [CData Architecture: Query Execution](#).

## Logging

Capturing adapter logging can be very helpful when diagnosing error messages or other unexpected behavior.

## Basic Logging

You will simply need to set two connection properties to begin capturing adapter logging.

- Logfile: A filepath which designates the name and location of the log file.
- Verbosity: This is a numerical value (1-5) that determines the amount of detail in the log. See the page in the Connection Properties section for an explanation of the five levels.
- MaxLogFileSize: When the limit is hit, a new log is created in the same folder with the date and time appended to the end. The default limit is 100 MB. Values lower than 100 kB will use 100 kB as the value instead.
- MaxLogFileCount: A string specifying the maximum file count of log files. When the limit is hit, a new log is created in the same folder with the date and time appended to the end and the oldest log file will be deleted. Minimum supported value is 2. A value of 0 or a negative value indicates no limit on the count.

Once this property is set, the adapter will populate the log file as it carries out various tasks, such as when authentication is performed or queries are executed. If the specified file doesn't already exist, it will be created.

## Log Verbosity

The verbosity level determines the amount of detail that the adapter reports to the Logfile. Verbosity levels from 1 to 5 are supported. These are described in the following list:

- |   |   |
|---|---|
| 1 | Setting <u>Verbosity</u> to 1 will log the query, the number of rows returned by it, the start of execution and the time taken, and any errors. |
| 2 | Setting <u>Verbosity</u> to 2 will log everything included in <u>Verbosity</u> 1 and additional information about the request.                  |
| 3 | Setting <u>Verbosity</u> to 3 will additionally log HTTP headers, as well as the body of the request and the response.                          |
| 4 | Setting <u>Verbosity</u> to 4 will additionally log transport-level communication with the data   |

---

source. This includes SSL negotiation.

---

- 5     Setting Verbosity to 5 will additionally log communication with the data source and additional details that may be helpful in troubleshooting problems. This includes interface commands.
- 

The Verbosity should not be set to greater than 1 for normal operation. Substantial amounts of data can be logged at higher verbosity levels, which can delay execution times.

To refine the logged content further by showing/hiding specific categories of information, see LogModules.

## Sensitive Data

Verbosity levels 3 and higher may capture information that you do not want shared outside of your organization. The following lists information of concern for each level:

- Verbosity 3: The full body of the request and the response, which includes all the data returned by the adapter
- Verbosity 4: SSL certificates
- Verbosity 5: Any extra transfer data not included at Verbosity 3, such as non human-readable binary transfer data

## Best Practices for Data Security

Although we mask sensitive values, such as passwords, in the connection string and any request in the log, it is always best practice to review the logs for any sensitive information before sharing outside your organization.

## Java Logging

When Java logging is enabled in Logfile, the Verbosity will instead map to the following logging levels.

- 0: Level.WARNING
- 1: Level.INFO
- 2: Level.CONFIG

- 3: Level.FINE
- 4: Level.FINER
- 5: Level.FINEST

## Advanced Logging

You may want to refine the exact information that is recorded to the log file. This can be accomplished using the LogModules property.

This property allows you to filter the logging using a semicolon-separated list of logging modules.

All modules are four characters long. **Please note that modules containing three letters have a required trailing blank space.** The available modules are:

- **EXEC:** Query Execution. Includes execution messages for original SQL queries, parsed SQL queries, and normalized SQL queries. Query and page success/failure messages appear here as well.
- **INFO:** General Information. Includes the connection string, driver version (build number), and initial connection messages.
- **HTTP:** HTTP Protocol messages. Includes HTTP requests/responses (including POST messages), as well as Kerberos related messages.
- **SSL :** SSL certificate messages.
- **OAUT:** OAuth related failure/success messages.
- **SQL :** Includes SQL transactions, SQL bulk transfer messages, and SQL result set messages.
- **META:** Metadata cache and schema messages.
- **TCP :** Incoming and Ongoing raw bytes on TCP transport layer messages.

An example value for this property would be.

```
LogModules=INFO;EXEC;SSL ;SQL ;META;
```

Note that these modules refine the information as it is pulled after taking the Verbosity into account.

## SQL Compliance

The HBase Adapter supports several operations on data, including querying, deleting, modifying, and inserting.

## SELECT Statements

See [SELECT Statements](#) for a syntax reference and examples.

See [Data Model](#) for information on the capabilities of the HBase API.

## INSERT Statements

See [INSERT Statements](#) for a syntax reference and examples.

## UPDATE Statements

The primary key Id is required to update a record. See [UPDATE Statements](#) for a syntax reference and examples.

## UPSERT Statements

An UPSERT updates a record if it exists and inserts the record if it does not. See [UPSERT Statements](#) for a syntax reference and examples.

## DELETE Statements

The primary key Id is required to delete a record. See [DELETE Statements](#) for a syntax reference and examples.

## EXECUTE Statements

Use EXECUTE or EXEC statements to execute stored procedures. See [EXECUTE Statements](#) for a syntax reference and examples.

## Names and Quoting

- Table and column names are considered identifier names; as such, they are restricted

to the following characters: [A-Z, a-z, 0-9, \_:@].

- To use a table or column name with characters not listed above, the name must be quoted using double quotes ("name") in any SQL statement.
- Strings must be quoted using single quotes (e.g., 'John Doe').

## SELECT Statements

A SELECT statement can consist of the following basic clauses.

- SELECT
- INTO
- FROM
- JOIN
- WHERE
- GROUP BY
- HAVING
- UNION
- ORDER BY
- LIMIT

## SELECT Syntax

The following syntax diagram outlines the syntax supported by the HBase adapter:

```
SELECT {
  [ TOP <numeric_literal> ]
  {
    *
    | {
      <expression> [ [ AS ] <column_reference> ]
      | { <table_name> | <correlation_name> } .*
    } [ , ... ]
  }
  [ INTO csv:// [ filename= ] <file_path> [ ;delimiter=tab ] ]
  {
    FROM <table_reference> [ [ AS ] <identifier> ]
```

```

    }
    [ WHERE <search_condition> ]
} | SCOPE_IDENTITY()
<expression> ::=
    | <column_reference>
    | @ <parameter>
    | ?
    | COUNT( * | { [ DISTINCT ] <expression> } )
    | { AVG | MAX | MIN | SUM | COUNT } ( <expression> )
    | NULLIF ( <expression> , <expression> )
    | COALESCE ( <expression> , ... )
    | CASE <expression>
        WHEN { <expression> | <search_condition> } THEN { <expression> |
NULL } [ ... ]
        [ ELSE { <expression> | NULL } ]
        END
    | <literal>
    | <sql_function>
<search_condition> ::=
    {
        <expression> { = | > | < | >= | <= | <> | != | AND | OR | LIKE |
NOT LIKE | IN | NOT IN } [ <expression> ]
    } [ { AND | OR } ... ]

```

## Examples

1. Return all columns:

```
SELECT * FROM Account
```

2. Rename a column:

```
SELECT "Name" AS MY_Name FROM Account
```

3. Cast a column's data as a different data type:

```
SELECT CAST(AnnualRevenue AS VARCHAR) AS Str_AnnualRevenue FROM
Account
```

4. Search data:



```
SELECT * FROM Account WHERE Industry = 'Floppy Disks'
```

5. The HBase APIs support the following operators in the WHERE clause: =, >, <, >=, <=, <>, !=, AND, OR, LIKE, NOT LIKE, IN, NOT IN.

```
SELECT * FROM Account WHERE Industry = 'Floppy Disks';
```

## SELECT INTO Statements

You can use the SELECT INTO statement to export formatted data to a file.

### Data Export with an SQL Query

The following query exports data into a file formatted in comma-separated values (CSV):

```
boolean ret = stat.execute("SELECT Id, Name INTO 'csv://c:/Account.txt'
FROM 'Account' WHERE Industry = 'Floppy Disks'");
System.out.println(stat.getUpdateCount()+" rows affected");
```

You can specify other file formats in the URI. The following example exports tab-separated values:

```
Statement stat = conn.createStatement();
boolean ret = stat.execute("SELECT * INTO 'Account' IN
'csv://filename=c:/Account.csv;delimiter=tab' FROM 'Account' WHERE
Industry = 'Floppy Disks'");
System.out.println(stat.getUpdateCount()+" rows affected");
```

## INSERT Statements

To create new records, use INSERT statements.

### INSERT Syntax

The INSERT statement specifies the columns to be inserted and the new column values. You can specify the column values in a comma-separated list in the VALUES clause, as shown in the following example:

```

INSERT INTO <table_name>
( <column_reference> [ , ... ] )
VALUES
( { <expression> | NULL } [ , ... ] )

<expression> ::=
    | @ <parameter>
    | ?
    | <literal>

```

You can use the `executeUpdate` method of the `Statement` and `PreparedStatement` classes to execute data manipulation commands and retrieve the rows affected.

```

String cmd = "INSERT INTO Account (Name) VALUES (?)";
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "Floppy Disks");
int count = pstmt.executeUpdate();
System.out.println(count+" rows were affected");
connection.close();

```

## UPDATE Statements

To modify existing records, use UPDATE statements.

### Update Syntax

The UPDATE statement takes as input a comma-separated list of columns and new column values as name-value pairs in the SET clause, as shown in the following example:

```

UPDATE <table_name> SET { <column_reference> = <expression> } [ , ... ]
WHERE { Id = <expression> } [ { AND | OR } ... ]

<expression> ::=
    | @ <parameter>
    | ?
    | <literal>

```

You can use the `executeUpdate` method of the `Statement` or `PreparedStatement` classes to execute data manipulation commands and retrieve the rows affected, as shown in the following example:

```

String cmd = "UPDATE Account SET Name='Floppy Disks' WHERE Id = ?";
PreparedStatement pstmt = connection.prepareStatement(cmd);

```

```
pstmt.setString(1, "1");
int count = pstmt.executeUpdate();
System.out.println(count + " rows were affected");
connection.close();
```

## UPSERT Statements

An UPSERT statement updates an existing record or creates a new record if an existing record is not identified.

### UPSERT Syntax

The UPSERT syntax is the same as for insert. HBase uses the input provided in the VALUES clause to determine whether the record already exists. If the record does not exist, all columns required to insert the record must be specified. See [Data Model](#) for any table-specific information.

```
UPSERT INTO <table_name>
( <column_reference> [ , ... ] )
VALUES
( { <expression> | NULL } [ , ... ] )

<expression> ::=
| @ <parameter>
| ?
| <literal>
```

You can use the executeUpdate method of the Statement and PreparedStatement classes to issue data manipulation commands and retrieve the rows affected, as shown in the following example:

```
String cmd = "UPSERT INTO Account (Name) VALUES (?)";
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "Floppy Disks");
int count = pstmt.executeUpdate();
System.out.println(count+" rows were affected");
connection.close();
```

## DELETE Statements

To delete information from a table, use DELETE statements.

## DELETE Syntax

The DELETE statement requires the table name in the FROM clause and the row's primary key in the WHERE clause, as shown in the following example:

```
<delete_statement> ::= DELETE FROM <table_name> WHERE { Id =
<expression> } [ { AND | OR } ... ]
<expression> ::=
    | @ <parameter>
    | ?
    | <literal>
```

You can use the executeUpdate method of the Statement or PreparedStatement classes to execute data manipulation commands and retrieve the number of affected rows, as shown in the following example:

```
Connection connection = DriverManager.getConnection
("jdbc:apachehbase:Server=127.0.0.1;Port=8080;");
String cmd = "DELETE FROM Account WHERE Id = ?";
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "1");
int count=pstmt.executeUpdate();
connection.close();
```

## EXECUTE Statements

To execute stored procedures, you can use EXECUTE or EXEC statements.

EXEC and EXECUTE assign stored procedure inputs, referenced by name, to values or parameter names.

### Stored Procedure Syntax

To execute a stored procedure as an SQL statement, use the following syntax:

```
{ EXECUTE | EXEC } <stored_proc_name>
{
    [ @ ] <input_name> = <expression>
} [ , ... ]
<expression> ::=
    | @ <parameter>
    | ?
    | <literal>
```

## Example Statements

Reference stored procedure inputs by name:

```
EXECUTE my_proc @second = 2, @first = 1, @third = 3;
```

Execute a parameterized stored procedure statement:

```
EXECUTE my_proc second = @p1, first = @p2, third = @p3;
```

## PIVOT and UNPIVOT

**PIVOT** and **UNPIVOT** can be used to change a table-valued expression into another table.

### PIVOT

PIVOT rotates a table-value expression by turning unique values from one column into multiple columns in the output. PIVOT can run aggregations where required on any column value.

### PIVOT Syntax

```
"SELECT 'AverageCost' AS Cost_Sorted_By_Production_Days, [0], [1], [2],
[3], [4]
FROM
(
SELECT DaysToManufacture, StandardCost
FROM Production.Product
) AS SourceTable
PIVOT
(
AVG(StandardCost)
FOR DaysToManufacture IN ([0], [1], [2], [3], [4])
) AS PivotTable;"
```

### UNPIVOT

UNPIVOT carries out nearly the opposite to PIVOT by rotating columns of a table-valued expressions into column values.

## UNPIVOT Syntax

```
"SELECT VendorID, Employee, Orders  
FROM  
(SELECT VendorID, Emp1, Emp2, Emp3, Emp4, Emp5  
FROM pvt) p  
UNPIVOT  
(Orders FOR Employee IN  
(Emp1, Emp2, Emp3, Emp4, Emp5)  
)AS unpvt;"
```

For further information on PIVOT and UNPIVOT, see [FROM clause plus JOIN, APPLY, PIVOT \(Transact-SQL\)](#)

## Data Model

The HBase Adapter models the HBase REST APIs as relational tables and stored procedures that can be accessed with standard SQL. This enables access from standards-based tools.

The table definitions are dynamically retrieved. When you connect, the adapter connects to HBase and gets the list of tables and the metadata for the tables by querying the HBase REST server. Any changes to the remote data are immediately reflected in your queries.

[Stored Procedures](#) are function-like interfaces to the data source. Stored procedures model actions that typically cannot be represented as SELECT, INSERT, UPDATE, or DELETE statements. The stored procedures of the adapter surface the capabilities of the DDL (data definition language) in HBase.

## Stored Procedures

Stored procedures are function-like interfaces that extend the functionality of the adapter beyond simple SELECT/INSERT/UPDATE/DELETE operations with HBase.

Stored procedures accept a list of parameters, perform their intended function, and then return, if applicable, any relevant response data from HBase, along with an indication of whether the procedure succeeded or failed.

### HBase Adapter Stored Procedures

Name	Description
CreateTable	Creates the specified table in Apache HBase.
DeleteTable	Deletes the specified table from Apache HBase.
ScanTable	Scan the specified table in Apache HBase.
UpdateTable	Updates (adds or modifies the column family) the specified table in Apache HBase.

## CreateTable

Creates the specified table in Apache HBase.

**Note:** This procedure makes use of **indexed parameters**. These input parameters are denoted with a '#' character at the end of their names.

Indexed parameters facilitate providing multiple instances a single parameter as inputs for the procedure.

Suppose there is an input parameter named Param#. Input multiple instances of an indexed parameter like this:

```
EXEC ProcedureName Param#1 = "value1", Param#2 = "value2", Param#3 = "value3"
```

## Input

Name	Type	Description
TableName	<i>String</i>	The name of the table you wish to create. If the table already exists in Apache HBase, the existing schema will be replaced with the one specified by ColumnFamily.
ColumnFamily#	<i>String</i>	The name of the column family to add to the table. At least one is required.

## Result Set Columns

Name	Type	Description
Success	<i>String</i>	Returns true if the operation is successful, else an exception is returned.

## DeleteTable

Deletes the specified table from Apache HBase.

### Input

Name	Type	Description
TableName	<i>String</i>	The name of the table you wish to delete.

## Result Set Columns

Name	Type	Description
Success	<i>String</i>	Returns true if the operation is successful, else an exception is returned.

## ScanTable

Scan the specified table in Apache HBase.



## Input

Name	Type	Description
TableName	<i>String</i>	The name of the table you wish to scan.
Parameter	<i>String</i>	The parameter of scan command.
Rows@next	<i>String</i>	This is used to page through multiple pages of results and should not be set manually.

## Result Set Columns

Name	Type	Description
ROW	<i>String</i>	The value of row.
COLUMN+CELL	<i>String</i>	The value of column and cell.

## UpdateTable

Updates (adds or modifies the column family) the specified table in Apache HBase.

**Note:** This procedure makes use of **indexed parameters**. These input parameters are denoted with a '#' character at the end of their names.

Indexed parameters facilitate providing multiple instances a single parameter as inputs for the procedure.

Suppose there is an input parameter named Param#. Input multiple instances of an indexed parameter like this:

```
EXEC ProcedureName Param#1 = "value1", Param#2 = "value2", Param#3 = "value3"
```

## Input

Name	Type	Description
TableName	<i>String</i>	The name of the table you wish to update. If the table does not exist, it will be created.
ColumnFamily#	<i>String</i>	The name of the column family to add or modify in the table. At least one is required.

## Result Set Columns

Name	Type	Description
Success	<i>String</i>	Returns true if the operation is successful, else an exception is returned.

# Connection String Options

The connection string properties are the various options that can be used to establish a connection. This section provides a complete list of the options you can configure in the connection string for this provider. Click the links for further details.

For more information on establishing a connection, see [Basic Tab](#).

## Authentication

Property	Description
<a href="#">AuthScheme</a>	The scheme used for authentication. Accepted entries are None, Basic, and Negotiate (Kerberos). None is the default.
<a href="#">Server</a>	The host name or IP address of the Apache HBase REST server.
<a href="#">Port</a>	The port for the Apache HBase REST (Stargate) server.
<a href="#">User</a>	The user who is authenticating to Apache HBase.
<a href="#">Password</a>	The password used to authenticate to Apache HBase.

## Kerberos

Property	Description
<a href="#">KerberosKDC</a>	The Kerberos Key Distribution Center (KDC) service used to authenticate the user.
<a href="#">KerberosRealm</a>	The Kerberos Realm used to authenticate the user.
<a href="#">KerberosSPN</a>	The service principal name (SPN) for the Kerberos Domain Controller.
<a href="#">KerberosKeytabFile</a>	The Keytab file containing your pairs of Kerberos principals and encrypted keys.
<a href="#">KerberosServiceRealm</a>	The Kerberos realm of the service.
<a href="#">KerberosServiceKDC</a>	The Kerberos KDC of the service.
<a href="#">KerberosTicketCache</a>	The full file path to an MIT Kerberos credential cache file.

## SSL

Property	Description
<a href="#">SSLClientCert</a>	The TLS/SSL client certificate store for SSL Client Authentication (2-way SSL).
<a href="#">SSLClientCertType</a>	The type of key store containing the TLS/SSL client certificate.
<a href="#">SSLClientCertPassword</a>	The password for the TLS/SSL client certificate.
<a href="#">SSLClientCertSubject</a>	The subject of the TLS/SSL client certificate.
<a href="#">SSLServerCert</a>	The certificate to be accepted from the server when connecting using TLS/SSL.

## Firewall

Property	Description
<a href="#">FirewallType</a>	The protocol used by a proxy-based firewall.
<a href="#">FirewallServer</a>	The name or IP address of a proxy-based firewall.
<a href="#">FirewallPort</a>	The TCP port for a proxy-based firewall.
<a href="#">FirewallUser</a>	The user name to use to authenticate with a proxy-based firewall.
<a href="#">FirewallPassword</a>	A password used to authenticate to a proxy-based firewall.

## Proxy

Property	Description
<a href="#">ProxyAutoDetect</a>	This indicates whether to use the system proxy settings or not. This takes precedence over other proxy settings, so you'll need to set ProxyAutoDetect to FALSE in order use custom proxy settings.

<a href="#">ProxyServer</a>	The hostname or IP address of a proxy to route HTTP traffic through.
<a href="#">ProxyPort</a>	The TCP port the ProxyServer proxy is running on.
<a href="#">ProxyAuthScheme</a>	The authentication type to use to authenticate to the ProxyServer proxy.
<a href="#">ProxyUser</a>	A user name to be used to authenticate to the ProxyServer proxy.
<a href="#">ProxyPassword</a>	A password to be used to authenticate to the ProxyServer proxy.
<a href="#">ProxySSLType</a>	The SSL type to use when connecting to the ProxyServer proxy.
<a href="#">ProxyExceptions</a>	A semicolon separated list of destination hostnames or IPs that are exempt from connecting through the ProxyServer .

## Logging

Property	Description
<a href="#">LogModules</a>	Core modules to be included in the log file.

## Schema

Property	Description
<a href="#">Location</a>	A path to the directory that contains the schema files defining tables, views, and stored procedures.

## Miscellaneous

Property	Description

<a href="#">ColumnEncoding</a>	The Encoding used for the column values. Accepted entries are None, UTF8, and BASE64. None is the default.
<a href="#">DatetimeFormat</a>	The format used when inserting datetime values into the database.
<a href="#">IncludeDisabledTables</a>	Specifies whether to retrieve disabled tables.
<a href="#">MaxRows</a>	Limits the number of rows returned rows when no aggregation or group by is used in the query. This helps avoid performance issues at design time.
<a href="#">Other</a>	These hidden properties are used only in specific use cases.
<a href="#">Pagesize</a>	The maximum number of results to return per page from Apache HBase.
<a href="#">Readonly</a>	You can use this property to enforce read-only access to Apache HBase from the provider.
<a href="#">ResetTableMetadataOnDelete</a>	Specifies whether to reset the cache table after executing the DELETE Statement.
<a href="#">RetrieveSelectedColumnsOnly</a>	Specifies whether to retrieve selected columns only when executing a SELECT statement.
<a href="#">RowScanDepth</a>	The maximum number of rows to scan to look for the columns available in a table.
<a href="#">ScannerCaching</a>	Specify a scanner cache that will be filled before the Scan result is returned, setting setCaching to the number of rows to cache before returning the result. By default, the caching setting on the table is used. The goal is to balance IO and network load.
<a href="#">Timeout</a>	The value in seconds until the timeout error is thrown, canceling the operation.
<a href="#">TypeDetectionScheme</a>	Determines how to determine the data type of columns.

---

<a href="#">UserDefinedViews</a>	A filepath pointing to the JSON configuration file containing your custom views.
<a href="#">UseSQLFiltering</a>	Specifies whether to use the ScannerFilter when executing the SELECT Statement.

---

## Authentication

This section provides a complete list of the Authentication properties you can configure in the connection string for this provider.

---

Property	Description
<a href="#">AuthScheme</a>	The scheme used for authentication. Accepted entries are None, Basic, and Negotiate (Kerberos). None is the default.
<a href="#">Server</a>	The host name or IP address of the Apache HBase REST server.
<a href="#">Port</a>	The port for the Apache HBase REST (Stargate) server.
<a href="#">User</a>	The user who is authenticating to Apache HBase.
<a href="#">Password</a>	The password used to authenticate to Apache HBase.

---

## AuthScheme

The scheme used for authentication. Accepted entries are None, Basic, and Negotiate (Kerberos). None is the default.

### Possible Values

None, Basic, Negotiate

### Data Type

string

## Default Value

"None"

## Remarks

This field is used to authenticate against the server. Use the following options to select your authentication scheme:

- None: No authentication is performed.
- Basic: Basic authentication is performed.
- Negotiate: If AuthScheme is set to Negotiate, the adapter will Negotiate an authentication mechanism with the server. Set AuthScheme to Negotiate if you want to use Kerberos authentication.

## Server

The host name or IP address of the Apache HBase REST server.

## Data Type

string

## Default Value

""

## Remarks

The host name or IP address of the HBase REST (Stargate) server.

To use SSL, you can prefix the host name or IP address with 'https://'.

## Port

The port for the Apache HBase REST (Stargate) server.



## Data Type

string

## Default Value

"8080"

## Remarks

The port for the HBase REST (Stargate) server.

## User

The user who is authenticating to Apache HBase.

## Data Type

string

## Default Value

""

## Remarks

The user who is authenticating to HBase.

## Password

The password used to authenticate to Apache HBase.

## Data Type

string

## Default Value

""

## Remarks

The password used to authenticate to HBase.

# Kerberos

This section provides a complete list of the Kerberos properties you can configure in the connection string for this provider.

Property	Description
<a href="#">KerberosKDC</a>	The Kerberos Key Distribution Center (KDC) service used to authenticate the user.
<a href="#">KerberosRealm</a>	The Kerberos Realm used to authenticate the user.
<a href="#">KerberosSPN</a>	The service principal name (SPN) for the Kerberos Domain Controller.
<a href="#">KerberosKeytabFile</a>	The Keytab file containing your pairs of Kerberos principals and encrypted keys.
<a href="#">KerberosServiceRealm</a>	The Kerberos realm of the service.
<a href="#">KerberosServiceKDC</a>	The Kerberos KDC of the service.
<a href="#">KerberosTicketCache</a>	The full file path to an MIT Kerberos credential cache file.

## KerberosKDC

The Kerberos Key Distribution Center (KDC) service used to authenticate the user.

## Data Type

string

## Default Value

""

## Remarks

The Kerberos properties are used when using SPNEGO or Windows Authentication. The adapter will request session tickets and temporary session keys from the Kerberos KDC service. The Kerberos KDC service is conventionally colocated with the domain controller.

If Kerberos KDC is not specified, the adapter will attempt to detect these properties automatically from the following locations:

- **KRB5 Config File (krb5.ini/krb5.conf):** If the KRB5\_CONFIG environment variable is set and the file exists, the adapter will obtain the KDC from the specified file. Otherwise, it will attempt to read from the default MIT location based on the OS: *C:\ProgramData\MIT\Kerberos5\krb5.ini* (Windows) or */etc/krb5.conf* (Linux).
- **Java System Properties:** Using the system properties *java.security.krb5.realm* and *java.security.krb5.kdc*.
- **Domain Name and Host:** If the Kerberos Realm and Kerberos KDC could not be inferred from another location, the adapter will infer them from the configured domain name and host.

**Note:** Windows authentication is supported in JRE 1.6 and above only.

## KerberosRealm

The Kerberos Realm used to authenticate the user.

## Data Type

string

## Default Value

""

## Remarks

The Kerberos properties are used when using SPNEGO or Windows Authentication. The Kerberos Realm is used to authenticate the user with the Kerberos Key Distribution Service (KDC). The Kerberos Realm can be configured by an administrator to be any string, but conventionally it is based on the domain name.

If Kerberos Realm is not specified, the adapter will attempt to detect these properties automatically from the following locations:

- **KRB5 Config File (krb5.ini/krb5.conf):** If the KRB5\_CONFIG environment variable is set and the file exists, the adapter will obtain the default realm from the specified file. Otherwise, it will attempt to read from the default MIT location based on the OS: *C:\ProgramData\MIT\Kerberos5\krb5.ini* (Windows) or */etc/krb5.conf* (Linux)
- **Java System Properties:** Using the system properties *java.security.krb5.realm* and *java.security.krb5.kdc*.
- **Domain Name and Host:** If the Kerberos Realm and Kerberos KDC could not be inferred from another location, the adapter will infer them from the user-configured domain name and host. This might work in some Windows environments.

**Note:** Kerberos-based authentication is supported in JRE 1.6 and above only.

## KerberosSPN

The service principal name (SPN) for the Kerberos Domain Controller.

### Data Type

string

### Default Value

""

## Remarks

If the SPN on the Kerberos Domain Controller is not the same as the URL that you are authenticating to, use this property to set the SPN.

## KerberosKeytabFile

The Keytab file containing your pairs of Kerberos principals and encrypted keys.

## Data Type

string

## Default Value

""

## Remarks

The Keytab file containing your pairs of Kerberos principals and encrypted keys.

## KerberosServiceRealm

The Kerberos realm of the service.

## Data Type

string

## Default Value

""

## Remarks

The KerberosServiceRealm is the specify the service Kerberos realm when using cross-realm Kerberos authentication.

In most cases, a single realm and KDC machine are used to perform the Kerberos authentication and this property is not required.

This property is available for complex setups where a different realm and KDC machine are used to obtain an authentication ticket (AS request) and a service ticket (TGS request).

## KerberosServiceKDC

The Kerberos KDC of the service.

## Data Type

string

## Default Value

""

## Remarks

The KerberosServiceKDC is used to specify the service Kerberos KDC when using cross-realm Kerberos authentication.

In most cases, a single realm and KDC machine are used to perform the Kerberos authentication and this property is not required.

This property is available for complex setups where a different realm and KDC machine are used to obtain an authentication ticket (AS request) and a service ticket (TGS request).

## KerberosTicketCache

The full file path to an MIT Kerberos credential cache file.

## Data Type

string

## Default Value

""

## Remarks

This property can be set if you wish to use a credential cache file that was created using the MIT Kerberos Ticket Manager or kinit command.

## SSL

This section provides a complete list of the SSL properties you can configure in the connection string for this provider.

Property	Description
<a href="#">SSLClientCert</a>	The TLS/SSL client certificate store for SSL Client Authentication (2-way SSL).
<a href="#">SSLClientCertType</a>	The type of key store containing the TLS/SSL client certificate.
<a href="#">SSLClientCertPassword</a>	The password for the TLS/SSL client certificate.
<a href="#">SSLClientCertSubject</a>	The subject of the TLS/SSL client certificate.
<a href="#">SSLServerCert</a>	The certificate to be accepted from the server when connecting using TLS/SSL.

## SSLClientCert

The TLS/SSL client certificate store for SSL Client Authentication (2-way SSL).

### Data Type

string

### Default Value

""

### Remarks

The name of the certificate store for the client certificate.

The [SSLClientCertType](#) field specifies the type of the certificate store specified by [SSLClientCert](#). If the store is password protected, specify the password in [SSLClientCertPassword](#).

`SSLClientCert` is used in conjunction with the `SSLClientCertSubject` field in order to specify client certificates. If `SSLClientCert` has a value, and `SSLClientCertSubject` is set, a search for a certificate is initiated. See `SSLClientCertSubject` for more information.

Designations of certificate stores are platform-dependent.

The following are designations of the most common User and Machine certificate stores in Windows:

MY	A certificate store holding personal certificates with their associated private keys.
CA	Certifying authority certificates.
ROOT	Root certificates.
SPC	Software publisher certificates.

In Java, the certificate store normally is a file containing certificates and optional private keys.

When the certificate store type is `PFXFile`, this property must be set to the name of the file. When the type is `PFXBlob`, the property must be set to the binary contents of a PFX file (for example, PKCS12 certificate store).

## SSLClientCertType

The type of key store containing the TLS/SSL client certificate.

### Possible Values

USER, MACHINE, PFXFILE, PFXBLOB, JKSFIL, JKSBLOB, PEMKEY\_FILE, PEMKEY\_BLOB, PUBLIC\_KEY\_FILE, PUBLIC\_KEY\_BLOB, SSHPUBLIC\_KEY\_FILE, SSHPUBLIC\_KEY\_BLOB, P7BFILE, PPKFILE, XMLFILE, XMLBLOB

### Data Type

string



## Default Value

"USER"

## Remarks

This property can take one of the following values:

USER - default	For Windows, this specifies that the certificate store is a certificate store owned by the current user. Note that this store type is not available in Java.
MACHINE	For Windows, this specifies that the certificate store is a machine store. Note that this store type is not available in Java.
PFXFILE	The certificate store is the name of a PFX (PKCS12) file containing certificates.
PFXBLOB	The certificate store is a string (base-64-encoded) representing a certificate store in PFX (PKCS12) format.
JKSFILE	The certificate store is the name of a Java key store (JKS) file containing certificates. Note that this store type is only available in Java.
JKSBLOB	The certificate store is a string (base-64-encoded) representing a certificate store in JKS format. Note that this store type is only available in Java.
PEMKEY_FILE	The certificate store is the name of a PEM-encoded file that contains a private key and an optional certificate.
PEMKEY_BLOB	The certificate store is a string (base64-encoded) that contains a private key and an optional certificate.
PUBLIC_KEY_FILE	The certificate store is the name of a file that contains a PEM- or DER-encoded public key certificate.
PUBLIC_KEY_BLOB	The certificate store is a string (base-64-encoded) that contains a PEM- or DER-encoded public key certificate.

SSHPUBLIC_KEY_FILE	The certificate store is the name of a file that contains an SSH-style public key.
SSHPUBLIC_KEY_BLOB	The certificate store is a string (base-64-encoded) that contains an SSH-style public key.
P7BFILE	The certificate store is the name of a PKCS7 file containing certificates.
PPKFILE	The certificate store is the name of a file that contains a PuTTY Private Key (PPK).
XMLFILE	The certificate store is the name of a file that contains a certificate in XML format.
XMLBLOB	The certificate store is a string that contains a certificate in XML format.

## SSLClientCertPassword

The password for the TLS/SSL client certificate.

### Data Type

string

### Default Value

""

### Remarks

If the certificate store is of a type that requires a password, this property is used to specify that password to open the certificate store.

## SSLClientCertSubject

The subject of the TLS/SSL client certificate.

## Data Type

string

## Default Value

"\*"

## Remarks

When loading a certificate the subject is used to locate the certificate in the store.

If an exact match is not found, the store is searched for subjects containing the value of the property. If a match is still not found, the property is set to an empty string, and no certificate is selected.

The special value "\*" picks the first certificate in the certificate store.

The certificate subject is a comma separated list of distinguished name fields and values. For example, "CN=www.server.com, OU=test, C=US, E=support@company.com". The common fields and their meanings are shown below.

Field	Meaning
CN	Common Name. This is commonly a host name like www.server.com.
O	Organization
OU	Organizational Unit
L	Locality
S	State
C	Country
E	Email Address

If a field value contains a comma, it must be quoted.

## SSLServerCert

The certificate to be accepted from the server when connecting using TLS/SSL.

### Data Type

string

### Default Value

""

### Remarks

If using a TLS/SSL connection, this property can be used to specify the TLS/SSL certificate to be accepted from the server. Any other certificate that is not trusted by the machine is rejected.

This property can take the following forms:

Description	Example
A full PEM Certificate (example shortened for brevity)	-----BEGIN CERTIFICATE----- MIICHTCCAe4CAQAwDQYJKoZIhvd.....Qw == -----END CERTIFICATE-----
A path to a local file containing the certificate	C:\cert.cer
The public key (example shortened for brevity)	-----BEGIN RSA PUBLIC KEY----- MIGfMA0GCSq.....AQAB -----END RSA PUBLIC KEY-----
The MD5 Thumbprint (hex values can also be either space or colon separated)	34e929226ae0819f2ec14b4a3d904f801c
The SHA1 Thumbprint (hex values can also be either space or colon separated)	bb150d

If not specified, any certificate trusted by the machine is accepted.

Certificates are validated as trusted by the machine based on the System's trust store. The trust store used is the 'javax.net.ssl.trustStore' value specified for the system. If no value is specified for this property, Java's default trust store is used (for example, JAVA\_HOME\lib\security\cacerts).

Use '\*' to signify to accept all certificates. Note that this is not recommended due to security concerns.

## Firewall

This section provides a complete list of the Firewall properties you can configure in the connection string for this provider.

---

Property	Description
<a href="#">FirewallType</a>	The protocol used by a proxy-based firewall.
<a href="#">FirewallServer</a>	The name or IP address of a proxy-based firewall.
<a href="#">FirewallPort</a>	The TCP port for a proxy-based firewall.
<a href="#">FirewallUser</a>	The user name to use to authenticate with a proxy-based firewall.
<a href="#">FirewallPassword</a>	A password used to authenticate to a proxy-based firewall.

---

## FirewallType

The protocol used by a proxy-based firewall.

### Possible Values

NONE, TUNNEL, SOCKS4, SOCKS5

### Data Type

string

## Default Value

"NONE"

## Remarks

This property specifies the protocol that the adapter will use to tunnel traffic through the [FirewallServer](#) proxy. Note that by default, the adapter connects to the system proxy; to disable this behavior and connect to one of the following proxy types, set [ProxyAutoDetect](#) to false.

Type	Default Port	Description
TUNNEL	80	When this is set, the adapter opens a connection to HBase and traffic flows back and forth through the proxy.
SOCKS4	1080	When this is set, the adapter sends data through the SOCKS 4 proxy specified by <a href="#">FirewallServer</a> and <a href="#">FirewallPort</a> and passes the <a href="#">FirewallUser</a> value to the proxy, which determines if the connection request should be granted.
SOCKS5	1080	When this is set, the adapter sends data through the SOCKS 5 proxy specified by <a href="#">FirewallServer</a> and <a href="#">FirewallPort</a> . If your proxy requires authentication, set <a href="#">FirewallUser</a> and <a href="#">FirewallPassword</a> to credentials the proxy recognizes.

To connect to HTTP proxies, use [ProxyServer](#) and [ProxyPort](#). To authenticate to HTTP proxies, use [ProxyAuthScheme](#), [ProxyUser](#), and [ProxyPassword](#).

## FirewallServer

The name or IP address of a proxy-based firewall.

## Data Type

string

## Default Value

""

## Remarks

This property specifies the IP address, DNS name, or host name of a proxy allowing traversal of a firewall. The protocol is specified by [FirewallType](#): Use [FirewallServer](#) with this property to connect through SOCKS or do tunneling. Use [ProxyServer](#) to connect to an HTTP proxy.

Note that the adapter uses the system proxy by default. To use a different proxy, set [ProxyAutoDetect](#) to false.

## FirewallPort

The TCP port for a proxy-based firewall.

## Data Type

int

## Default Value

0

## Remarks

This specifies the TCP port for a proxy allowing traversal of a firewall. Use [FirewallServer](#) to specify the name or IP address. Specify the protocol with [FirewallType](#).

## FirewallUser

The user name to use to authenticate with a proxy-based firewall.

## Data Type

string

## Default Value

""

## Remarks

The [FirewallUser](#) and [FirewallPassword](#) properties are used to authenticate against the proxy specified in [FirewallServer](#) and [FirewallPort](#), following the authentication method specified in [FirewallType](#).

## FirewallPassword

A password used to authenticate to a proxy-based firewall.

## Data Type

string

## Default Value

""

## Remarks

This property is passed to the proxy specified by [FirewallServer](#) and [FirewallPort](#), following the authentication method specified by [FirewallType](#).

## Proxy

This section provides a complete list of the Proxy properties you can configure in the connection string for this provider.

Property	Description
<a href="#">ProxyAutoDetect</a>	This indicates whether to use the system proxy settings or not. This takes precedence over other proxy settings, so you'll need to set



	ProxyAutoDetect to FALSE in order use custom proxy settings.
<a href="#">ProxyServer</a>	The hostname or IP address of a proxy to route HTTP traffic through.
<a href="#">ProxyPort</a>	The TCP port the ProxyServer proxy is running on.
<a href="#">ProxyAuthScheme</a>	The authentication type to use to authenticate to the ProxyServer proxy.
<a href="#">ProxyUser</a>	A user name to be used to authenticate to the ProxyServer proxy.
<a href="#">ProxyPassword</a>	A password to be used to authenticate to the ProxyServer proxy.
<a href="#">ProxySSLType</a>	The SSL type to use when connecting to the ProxyServer proxy.
<a href="#">ProxyExceptions</a>	A semicolon separated list of destination hostnames or IPs that are exempt from connecting through the ProxyServer .

## ProxyAutoDetect

This indicates whether to use the system proxy settings or not. This takes precedence over other proxy settings, so you'll need to set ProxyAutoDetect to FALSE in order use custom proxy settings.

### Data Type

bool

### Default Value

true

### Remarks

This takes precedence over other proxy settings, so you'll need to set ProxyAutoDetect to FALSE in order use custom proxy settings.

NOTE: When this property is set to True, the proxy used is determined as follows:

- A search from the JVM properties (**http.proxy**, **https.proxy**, **socksProxy**, etc.) is performed.
- In the case that the JVM properties don't exist, a search from **java.home/lib/net.properties** is performed.
- In the case that `java.net.useSystemProxies` is set to `True`, a search from **the SystemProxy** is performed.
- In Windows only, an attempt is made to retrieve these properties from the **Internet Options** in the **registry**.

To connect to an HTTP proxy, see [ProxyServer](#). For other proxies, such as SOCKS or tunneling, see [FirewallType](#).

## ProxyServer

The hostname or IP address of a proxy to route HTTP traffic through.

### Data Type

string

### Default Value

""

### Remarks

The hostname or IP address of a proxy to route HTTP traffic through. The adapter can use the HTTP, Windows (NTLM), or Kerberos authentication types to authenticate to an HTTP proxy.

If you need to connect through a SOCKS proxy or tunnel the connection, see [FirewallType](#).

By default, the adapter uses the system proxy. If you need to use another proxy, set [ProxyAutoDetect](#) to `false`.

## ProxyPort

The TCP port the ProxyServer proxy is running on.

## Data Type

int

## Default Value

80

## Remarks

The port the HTTP proxy is running on that you want to redirect HTTP traffic through. Specify the HTTP proxy in [ProxyServer](#). For other proxy types, see [FirewallType](#).

## ProxyAuthScheme

The authentication type to use to authenticate to the ProxyServer proxy.

## Possible Values

BASIC, DIGEST, NONE, NEGOTIATE, NTLM, PROPRIETARY

## Data Type

string

## Default Value

"BASIC"

## Remarks

This value specifies the authentication type to use to authenticate to the HTTP proxy specified by [ProxyServer](#) and [ProxyPort](#).

Note that the adapter will use the system proxy settings by default, without further configuration needed; if you want to connect to another proxy, you will need to set [ProxyAutoDetect](#) to false, in addition to [ProxyServer](#) and [ProxyPort](#). To authenticate, set [ProxyAuthScheme](#) and set [ProxyUser](#) and [ProxyPassword](#), if needed.

The authentication type can be one of the following:

- **BASIC:** The adapter performs HTTP BASIC authentication.
- **DIGEST:** The adapter performs HTTP DIGEST authentication.
- **NEGOTIATE:** The adapter retrieves an NTLM or Kerberos token based on the applicable protocol for authentication.
- **PROPRIETARY:** The adapter does not generate an NTLM or Kerberos token. You must supply this token in the Authorization header of the HTTP request.

If you need to use another authentication type, such as SOCKS 5 authentication, see [FirewallType](#).

## ProxyUser

A user name to be used to authenticate to the ProxyServer proxy.

### Data Type

string

### Default Value

""

### Remarks

The [ProxyUser](#) and [ProxyPassword](#) options are used to connect and authenticate against the HTTP proxy specified in [ProxyServer](#).

You can select one of the available authentication types in [ProxyAuthScheme](#). If you are using HTTP authentication, set this to the user name of a user recognized by the HTTP proxy. If you are using Windows or Kerberos authentication, set this property to a user name in one of the following formats:

```
user@domain  
domain\user
```

## ProxyPassword

A password to be used to authenticate to the ProxyServer proxy.

## Data Type

string

## Default Value

""

## Remarks

This property is used to authenticate to an HTTP proxy server that supports NTLM (Windows), Kerberos, or HTTP authentication. To specify the HTTP proxy, you can set [ProxyServer](#) and [ProxyPort](#). To specify the authentication type, set [ProxyAuthScheme](#).

If you are using HTTP authentication, additionally set [ProxyUser](#) and [ProxyPassword](#) to HTTP proxy.

If you are using NTLM authentication, set [ProxyUser](#) and [ProxyPassword](#) to your Windows password. You may also need these to complete Kerberos authentication.

For SOCKS 5 authentication or tunneling, see [FirewallType](#).

By default, the adapter uses the system proxy. If you want to connect to another proxy, set [ProxyAutoDetect](#) to false.

## ProxySSLType

The SSL type to use when connecting to the ProxyServer proxy.

## Possible Values

AUTO, ALWAYS, NEVER, TUNNEL

## Data Type

string

## Default Value

"AUTO"

## Remarks

This property determines when to use SSL for the connection to an HTTP proxy specified by [ProxyServer](#). This value can be AUTO, ALWAYS, NEVER, or TUNNEL. The applicable values are the following:

<b>AUTO</b>	Default setting. If the URL is an HTTPS URL, the adapter will use the TUNNEL option. If the URL is an HTTP URL, the component will use the NEVER option.
<b>ALWAYS</b>	The connection is always SSL enabled.
<b>NEVER</b>	The connection is not SSL enabled.
<b>TUNNEL</b>	The connection is through a tunneling proxy. The proxy server opens a connection to the remote host and traffic flows back and forth through the proxy.

## ProxyExceptions

A semicolon separated list of destination hostnames or IPs that are exempt from connecting through the ProxyServer .

### Data Type

string

### Default Value

""

## Remarks

The [ProxyServer](#) is used for all addresses, except for addresses defined in this property. Use semicolons to separate entries.

Note that the adapter uses the system proxy settings by default, without further configuration needed; if you want to explicitly configure proxy exceptions for this connection, you need to set [ProxyAutoDetect](#) = false, and configure [ProxyServer](#) and

[ProxyPort](#). To authenticate, set [ProxyAuthScheme](#) and set [ProxyUser](#) and [ProxyPassword](#), if needed.

## Logging

This section provides a complete list of the Logging properties you can configure in the connection string for this provider.

---

Property	Description
<a href="#">LogModules</a>	Core modules to be included in the log file.

---

## LogModules

Core modules to be included in the log file.

### Data Type

string

### Default Value

""

### Remarks

Only the modules specified (separated by ';') will be included in the log file. By default all modules are included.

See the [Logging](#) page for an overview.

## Schema

This section provides a complete list of the Schema properties you can configure in the connection string for this provider.

Property	Description
<a href="#">Location</a>	A path to the directory that contains the schema files defining tables, views, and stored procedures.

## Location

A path to the directory that contains the schema files defining tables, views, and stored procedures.

## Data Type

string

## Default Value

"%APPDATA%\\CData\\ApacheHBase Data Provider\\Schema"

## Remarks

The path to a directory which contains the schema files for the adapter (.rsd files for tables and views, .rsb files for stored procedures). The folder location can be a relative path from the location of the executable. The [Location](#) property is only needed if you want to customize definitions (for example, change a column name, ignore a column, and so on) or extend the data model with new tables, views, or stored procedures.

If left unspecified, the default location is "%APPDATA%\\CData\\ApacheHBase Data Provider\\Schema" with **%APPDATA%** being set to the user's configuration directory:

Platform	%APPDATA%
Windows	The value of the APPDATA environment variable
Mac	~/Library/Application Support
Linux	~/config



## Miscellaneous

This section provides a complete list of the Miscellaneous properties you can configure in the connection string for this provider.

Property	Description
<a href="#">ColumnEncoding</a>	The Encoding used for the column values. Accepted entries are None, UTF8, and BASE64. None is the default.
<a href="#">DatetimeFormat</a>	The format used when inserting datetime values into the database.
<a href="#">IncludeDisabledTables</a>	Specifies whether to retrieve disabled tables.
<a href="#">MaxRows</a>	Limits the number of rows returned rows when no aggregation or group by is used in the query. This helps avoid performance issues at design time.
<a href="#">Other</a>	These hidden properties are used only in specific use cases.
<a href="#">Pagesize</a>	The maximum number of results to return per page from Apache HBase.
<a href="#">Readonly</a>	You can use this property to enforce read-only access to Apache HBase from the provider.
<a href="#">ResetTableMetadataOnDelete</a>	Specifies whether to reset the cache table after executing the DELETE Statement.
<a href="#">RetrieveSelectedColumnsOnly</a>	Specifies whether to retrieve selected columns only when executing a SELECT statement.
<a href="#">RowScanDepth</a>	The maximum number of rows to scan to look for the columns available in a table.
<a href="#">ScannerCaching</a>	Specify a scanner cache that will be filled before the Scan

	result is returned, setting setCaching to the number of rows to cache before returning the result. By default, the caching setting on the table is used. The goal is to balance IO and network load.
<a href="#">Timeout</a>	The value in seconds until the timeout error is thrown, canceling the operation.
<a href="#">TypeDetectionScheme</a>	Determines how to determine the data type of columns.
<a href="#">UserDefinedViews</a>	A filepath pointing to the JSON configuration file containing your custom views.
<a href="#">UseSQLFiltering</a>	Specifies whether to use the ScannerFilter when executing the SELECT Statement.

## ColumnEncoding

The Encoding used for the column values. Accepted entries are None, UTF8, and BASE64. None is the default.

### Possible Values

None, UTF8, BASE64

### Data Type

string

### Default Value

"None"

### Remarks

This field is used to encode the column value. Use the following options to specify your column encoding:

- None: Automatically get the the appropriate encoding via

TypeDetectionScheme=RowScan.

- UTF8: Specify the encoding UTF8 for all columns.
- BASE64: Specify the encoding BASE64 for all columns, except the key column "RowKey".

## DatetimeFormat

The format used when inserting datetime values into the database.

### Data Type

string

### Default Value

"yyyy-MM-dd'T'HH:mm:ss.fffzzz"

### Remarks

This can be used to automatically format any datetime values entering a database.

## IncludeDisabledTables

Specifies whether to retrieve disabled tables.

### Data Type

bool

### Default Value

false

### Remarks

Specifies whether to retrieve disabled tables.

## MaxRows

Limits the number of rows returned rows when no aggregation or group by is used in the query. This helps avoid performance issues at design time.

### Data Type

int

### Default Value

-1

### Remarks

Limits the number of rows returned rows when no aggregation or group by is used in the query. This helps avoid performance issues at design time.

## Other

These hidden properties are used only in specific use cases.

### Data Type

string

### Default Value

""

### Remarks

The properties listed below are available for specific use cases. Normal driver use cases and functionality should not require these properties.

Specify multiple properties in a semicolon-separated list.

## Integration and Formatting

DefaultColumnSize	Sets the default length of string fields when the data source does not provide column length in the metadata. The default value is 2000.
ConvertDateTimeToGMT	Determines whether to convert date-time values to GMT, instead of the local time of the machine.
RecordToFile=filename	Records the underlying socket data transfer to the specified file.

## Pagesize

The maximum number of results to return per page from Apache HBase.

### Data Type

int

### Default Value

10000

### Remarks

The Pagesize property affects the maximum number of results to return per page from HBase. Setting a higher value may result in better performance at the cost of additional memory allocated per page consumed.

## Readonly

You can use this property to enforce read-only access to Apache HBase from the provider.

### Data Type

bool

## Default Value

false

## Remarks

If this property is set to true, the adapter will allow only SELECT queries. INSERT, UPDATE, DELETE, and stored procedure queries will cause an error to be thrown.

## ResetTableMetadataOnDelete

Specifies whether to reset the cache table after executing the DELETE Statement.

## Data Type

bool

## Default Value

false

## Remarks

Specifies whether to reset the cache table after executing the DELETE Statement.

## RetrieveSelectedColumnsOnly

Specifies whether to retrieve selected columns only when executing a SELECT statement.

## Data Type

bool

## Default Value

false

## Remarks

By default, the adapter will create a generic scanner to retrieve the results from HBase. This enables all values to be returned without HBase filtering out rows containing empty columns that were selected.

When set to 'True', the adapter will only retrieve the selected columns from HBase. Note in this scenario, HBase will not return rows where a selected column is empty. In such a case, this can cause a different number of rows to be returned for various queries. This setting can improve performance but it should only be used when you know that there are no empty values for the selected columns or in the case that you don't want these rows to be returned.

## RowScanDepth

The maximum number of rows to scan to look for the columns available in a table.

### Data Type

int

### Default Value

100

## Remarks

The columns in a table must be determined by scanning table rows. This value determines the maximum number of rows that will be scanned.

Setting a high value may decrease performance. Setting a low value may prevent the data type from being determined properly, especially when there is null data.

## ScannerCaching

Specify a scanner cache that will be filled before the Scan result is returned, setting `setCaching` to the number of rows to cache before returning the result. By default, the caching setting on the table is used. The goal is to balance IO and network load.

## Data Type

string

## Default Value

"0"

## Remarks

It means there is no cache for the scanner if it is less than or equal to 0. A suitable cache value can improve query efficiency, but it can also introduce additional memory usage.

## Timeout

The value in seconds until the timeout error is thrown, canceling the operation.

## Data Type

int

## Default Value

60

## Remarks

If Timeout = 0, operations do not time out. The operations run until they complete successfully or until they encounter an error condition.

If Timeout expires and the operation is not yet complete, the adapter throws an exception.

## TypeDetectionScheme

Determines how to determine the data type of columns.

## Possible Values

None, RowScan



## Data Type

string

## Default Value

"RowScan"

## Remarks

None	Setting <a href="#">TypeDetectionScheme</a> to None will return all columns as strings. The <a href="#">RowScanDepth</a> determines the number of rows to be scanned to retrieve the column listing.
RowScan	Setting <a href="#">TypeDetectionScheme</a> to RowScan will scan rows to heuristically determine the data type. The <a href="#">RowScanDepth</a> determines the number of rows to be scanned to retrieve the column listing and column data type.

## UserDefinedViews

A filepath pointing to the JSON configuration file containing your custom views.

## Data Type

string

## Default Value

""

## Remarks

User Defined Views are defined in a JSON-formatted configuration file called *UserDefinedViews.json*. The adapter automatically detects the views specified in this file.

You can also have multiple view definitions and control them using the UserDefinedViews connection property. When you use this property, only the specified views are seen by the adapter.

This User Defined View configuration file is formatted as follows:

- Each root element defines the name of a view.
- Each root element contains a child element, called **query**, which contains the custom SQL query for the view.

For example:

```
{
  "MyView": {
    "query": "SELECT * FROM Account WHERE MyColumn = 'value'"
  },
  "MyView2": {
    "query": "SELECT * FROM MyTable WHERE Id IN (1,2,3)"
  }
}
```

Use the UserDefinedViews connection property to specify the location of your JSON configuration file. For example:

```
"UserDefinedViews",
"C:\\Users\\yourusername\\Desktop\\tmp\\UserDefinedViews.json"
```

## UseSQLFiltering

Specifies whether to use the ScannerFilter when executing the SELECT Statement.

### Data Type

bool

### Default Value

false

### Remarks

If UseSQLFiltering = false, only the client filter will be used.

# TIBCO Product Documentation and Support Services

---

For information about this product, you can read the documentation, contact TIBCO Support, and join the TIBCO Community.

## How to Access TIBCO Documentation

Documentation for TIBCO products is available on the [TIBCO Product Documentation](#) website, mainly in HTML and PDF formats.

The [TIBCO Product Documentation](#) website is updated frequently and is more current than any other documentation included with the product.

## Product-Specific Documentation

The following documentation for this product is available on the [TIBCO® Data Virtualization](#) page.

- **Users**
  - TDV Getting Started Guide
  - TDV User Guide
  - TDV Web UI User Guide
  - TDV Client Interfaces Guide
  - TDV Tutorial Guide
  - TDV Northbay Example
- **Administration**
  - TDV Installation and Upgrade Guide
  - TDV Administration Guide
  - TDV Active Cluster Guide
  - TDV Security Features Guide
- **Data Sources**

TDV Adapter Guides

TDV Data Source Toolkit Guide (Formerly Extensibility Guide)

- **References**

TDV Reference Guide

TDV Application Programming Interface Guide

- **Other**

TDV Business Directory Guide

TDV Discovery Guide

- *TIBCO TDV and Business Directory Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

## How to Contact TIBCO Support

Get an overview of [TIBCO Support](#). You can contact TIBCO Support in the following ways:

- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the [TIBCO Support](#) website.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to [TIBCO Support](#) website. If you do not have a user name, you can request one by clicking **Register** on the website.

## Release Version Support

TDV 8.5 is designated as a Long Term Support (LTS) version. Some release versions of TIBCO® Data Virtualization products are selected to be long-term support (LTS) versions. Defect corrections will typically be delivered in a new release version and as hotfixes or service packs to one or more LTS versions. See also

[https://docs.tibco.com/pub/tdv/general/LTS/tdv\\_LTS\\_releases.htm](https://docs.tibco.com/pub/tdv/general/LTS/tdv_LTS_releases.htm).

## How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the [TIBCO Ideas Portal](#). For a free registration, visit [TIBCO Community](#).

# Legal and Third-Party Notices

---

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, TIBCO logo, TIBCO O logo, ActiveSpaces, Enterprise Messaging Service, Spotfire, TERR, S-PLUS, and S+ are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle Corporation and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the

readme file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 2002-2023 Cloud Software Group, Inc All Rights Reserved.