



# **TIBCO® Data Virtualization**

## **Twitter Adapter Guide**

Version 8.7.0 | October 2023

# Contents

---

<b>Contents</b>	<b>2</b>
<b>Twitter Adapter</b>	<b>4</b>
Getting Started	4
Basic Tab	5
Logging	5
Using OAuth Authentication	7
Changelog	13
Advanced Features	16
User Defined Views	17
SSL Configuration	20
Firewall and Proxy	20
Query Processing	21
Logging	21
SQL Compliance	24
SELECT Statements	26
SELECT INTO Statements	27
INSERT Statements	28
DELETE Statements	29
EXECUTE Statements	29
PIVOT and UNPIVOT	30
Data Model	31
Tables	32
Views	51
Stored Procedures	84
Connection String Options	93
Authentication	97
OAuth	99

SSL .....	107
Firewall .....	108
Proxy .....	112
Logging .....	118
Schema .....	119
Miscellaneous .....	120
<b>TIBCO Product Documentation and Support Services .....</b>	<b>129</b>
How to Access TIBCO Documentation .....	129
How to Contact TIBCO Support .....	130
Release Version Support .....	130
How to Join TIBCO Community .....	131
<b>Legal and Third-Party Notices .....</b>	<b>132</b>

# Twitter Adapter

---

## Twitter Version Support

The adapter models the behavior of version 1.1 of the Twitter REST API as bidirectional tables. A developer account is required to connect to Twitter APIs.

## SQL Compliance

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

# Getting Started

## Connecting to Twitter

[Basic Tab](#) shows how to authenticate to Twitter 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 Twitter Adapter

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

1. Unzip the `tdv.twitter.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.twitter/tdv.twitter.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 Twitter
```

## Basic Tab

### OAuth

The adapter includes embedded OAuth credentials in order to connect to Twitter. You can connect without setting any connection properties for your user credentials. When you connect the adapter opens the OAuth endpoint in your default browser. Log in and grant permissions to the application. The adapter then completes the OAuth process.

## 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 Twitter 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**.

## Using OAuth Authentication

OAuth requires the authenticating user to interact with Twitter using the browser. The adapter facilitates this in various ways as described below.

### Embedded Credentials

#### Desktop Applications

See [Embedded Credentials](#) to connect with the adapter's embedded credentials and skip creating a custom OAuth app.

#### Headless Machines

See [Headless Machines](#) to skip creating a custom OAuth app and authenticate an application running on a headless server or another machine where the adapter is not authorized to open a browser.

### Custom Credentials

Instead of connecting with the adapter's embedded credentials, you can register an app to obtain the OAuthClientId also called the consumer key and OAuthClientSecret also called the consumer secret.

## When to Create a Custom OAuth App

### Desktop Applications

Creating a custom OAuth app is optional as the adapter is already registered with Twitter and you can connect with its embedded credentials. You might want to create a custom OAuth app to change the information displayed when users log into the Twitter OAuth endpoint to grant permissions to the adapter.

### Headless Machines

Creating a custom OAuth app is optional to authenticate a headless machine; the adapter is already registered with Twitter and you can connect with its embedded credentials. In the headless OAuth flow, users need to authenticate via a browser on another machine. You might want to create a custom OAuth app to change the information displayed when users log into the Twitter OAuth endpoint to grant permissions to the adapter.

## Creating a Custom OAuth App

See [Creating a Custom OAuth App](#) for a procedure.

# Embedded Credentials

## Embedded OAuth Credentials

### Desktop Authentication with the Embedded OAuth App

You can connect without setting any connection properties for your user credentials. After setting the following, you are ready to connect: InitiateOAuth: Set this to GETANDREFRESH. You can use InitiateOAuth to avoid repeating the OAuth exchange and manually setting the OAuthAccessToken. When you connect the adapter opens the OAuth endpoint in your default browser. Log in and grant permissions to the application. The adapter then completes the OAuth process.

1. Extracts the access token from the callback URL and authenticates requests.
2. Obtains a new access token when the old one expires.
3. Saves OAuth values in OAuthSettingsLocation to be persisted across connections.

## Custom Credentials

### When to Use a Custom OAuth App

You might want to create a custom OAuth app to change the information displayed when users log into Twitter to grant permissions to the adapter.



## Desktop Authentication with a Custom OAuth App

Follow the steps below to authenticate with the credentials for a custom OAuth app. See [Creating a Custom OAuth App](#) for more information.

### Get and Refresh the OAuth Access Token

After setting the following connection properties, you are ready to connect:

- OAuthClientId: Set this to the consumer key in your app settings.
- OAuthClientSecret: Set this to the consumer secret in your app settings.
- CallbackURL: Set this to the OAuth Redirect URL that you specified to use your own OAuth app.
- InitiateOAuth: Set this to GETANDREFRESH. You can use InitiateOAuth to avoid repeating the OAuth exchange and manually setting the OAuthAccessToken connection property.

When you connect the adapter opens the OAuth endpoint in your default browser. Log in and grant permissions to the application. The adapter then completes the OAuth process:

1. Extracts the access token from the callback URL and authenticates requests.
2. Refreshes the access token when it expires.
3. Saves OAuth values in OAuthSettingsLocation to be persisted across connections.

## Headless Machines

### Using OAuth on a Headless Machine

To create Twitter data sources on headless servers or other machines on which the adapter cannot open a browser, you need to authenticate from another machine. Authentication is a two-step process.

1. Instead of installing the adapter on another machine, you can follow the steps below to obtain the OAuthVerifier value. Or, you can install the adapter on another machine and transfer the OAuth authentication values, after you authenticate through the usual browser-based flow.
2. You can then configure the adapter to automatically refresh the access token from the headless machine.

You can follow the headless OAuth authentication flow using the adapter's embedded

OAuth credentials or using the OAuth credentials for your custom OAuth app.

## Using the Embedded OAuth Credentials

### Obtain a Verifier Code

Follow the steps below to authenticate from another machine and obtain the OAuthVerifier connection property:

1. Obtain the OAuth Authorization URL via the [GetOAuthAuthorizationURL](#) stored procedure. Save the returned AuthToken and AuthKey.
2. Open the OAuth Authorization URL in your browser.
3. Log in and grant permissions to the adapter. You are then redirected to the callback URL, which contains the verifier code.
4. Save the value of the verifier code. You will set this in the OAuthVerifier connection property.

On the headless machine, set the following connection properties to obtain the OAuth authentication values.

- AuthToken: Set to the authentication token returned by the [GetOAuthAuthorizationURL](#) stored procedure.
- AuthKey: Set to the authentication key returned by the [GetOAuthAuthorizationURL](#) stored procedure.
- OAuthVerifier: Set this to the verifier code.
- InitiateOAuth: Set this to REFRESH.
- OAuthSettingsLocation: Set this to persist the encrypted OAuth authentication values to the specified file.

After the OAuth settings file is generated, set the following properties to connect to data:

- OAuthSettingsLocation: Set this to the file containing the encrypted OAuth authentication values. Make sure this file gives read and write permissions to the adapter to enable the automatic refreshing of the access token.
- InitiateOAuth: Set this to REFRESH.

### Transfer OAuth Settings

Follow the steps below to install the adapter on another machine, authenticate, and then transfer the resulting OAuth values.

On a second machine, install the adapter and connect with the following properties set:

- OAuthSettingsLocation: Set this to a writable text file.
- InitiateOAuth: Set this to GETANDREFRESH.

Test the connection to authenticate in the browser. The resulting authentication values are written, encrypted, to the path specified by OAuthSettingsLocation. Once you have successfully tested the connection, copy the OAuth settings file to your headless machine.

On the headless machine, set the following connection properties to connect to data:

- InitiateOAuth: Set this to REFRESH.
- OAuthSettingsLocation: Set this to the path to your OAuth settings file. Make sure this file gives read and write permissions to the adapter to enable the automatic refreshing of the access token.

## Using the Credentials for a Custom OAuth App

### Create a Custom OAuth App

Creating a custom OAuth app is optional in the headless OAuth flow; you can skip creating an app by connecting with the adapter's embedded OAuth credentials. You might want to create a custom OAuth app to change the information displayed when users log into Twitter to grant permissions to the adapter.

See [Creating a Custom OAuth App](#) for a procedure. You can then follow the procedures below to authenticate and connect to data.

### Obtain a Verifier Code

Set the following properties on the headless machine:

- InitiateOAuth: Set this to OFF.
- OAuthClientId: Set this to the Client Id in your app settings.
- OAuthClientSecret: Set this to the Client Secret in your app settings.

You can then follow the steps below to authenticate from another machine and obtain the OAuthVerifier connection property.

1. Call the [GetOAuthAuthorizationURL](#) stored procedure with the CallbackURL input

parameter set to the exact Redirect URI you specified in your app settings.

2. Open the returned URL in a browser. Log in and grant permissions to the adapter. You are then redirected to the callback URL, which contains the verifier code.
3. Save the value of the verifier code. You will set this in the OAuthVerifier connection property.

On the headless machine, set the following connection properties to obtain the OAuth authentication values:

- OAuthClientId: Set to the consumer key in your app settings.
- OAuthClientSecret: Set to the consumer secret in your app settings.
- AuthToken: Set to the authentication token returned by the [GetOAuthAuthorizationURL](#) stored procedure.
- AuthKey: Set to the authentication key returned by the [GetOAuthAuthorizationURL](#) stored procedure.
- OAuthVerifier: Set this to the verifier code.
- OAuthSettingsLocation: Set this to persist the encrypted OAuth authentication values to the specified file.
- InitiateOAuth: Set this to REFRESH.

After the OAuth settings file is generated, set the following properties to connect to data:

- OAuthClientId: Set this to the consumer key in your app settings.
- OAuthClientSecret: Set this to the consumer secret in your app settings.
- OAuthSettingsLocation: Set this to the file containing the encrypted OAuth authentication values. Make sure this file gives read and write permissions to the provider to enable the automatic refreshing of the access token.
- InitiateOAuth: Set this to REFRESH.

### Transfer OAuth Settings

Follow the steps below to install the adapter on another machine, authenticate, and then transfer the resulting OAuth values.

On a second machine, install the adapter and connect with the following properties set:

- OAuthSettingsLocation: Set this to a writable text file.

- InitiateOAuth: Set this to GETANDREFRESH.
- OAuthClientId: Set this to the client Id assigned when you registered your app.
- OAuthClientSecret: Set this to the client secret assigned when you registered your app.
- CallbackURL: Set this to the Redirect URI you specified in your app settings.

Test the connection to authenticate. The resulting authentication values are written, encrypted, to the path specified by OAuthSettingsLocation. Once you have successfully tested the connection, copy the OAuth settings file to your headless machine. On the headless machine, set the following connection properties to connect to data:

- InitiateOAuth: Set this to REFRESH.
- OAuthClientId: Set this to the consumer key in your app settings.
- OAuthClientSecret: Set this to the consumer secret in your app settings.
- OAuthSettingsLocation: Set this to the path to your OAuth settings file. Make sure this file gives read and write permissions to the adapter to enable the automatic refreshing of the access token.

## Creating a Custom OAuth App

## 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 well.</li> </ul>
08/17/2022	8264	Twitter	<b>Added</b> <ul style="list-style-type: none"> <li>Added binary input support for UploadMedia. Added Content, ContentSize, FileName and MediaCategory as inputs for UploadMedia.</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 Col &gt; LAST_MONTH(). Formerly the two LAST_MONTH() methods would resolve to different datetimes. Now they will match.</li> </ul>

			<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. Ie: 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> <p><b>Changed</b></p> <ul style="list-style-type: none"> <li>Reduced the length to 255 for varchar primary key and foreign key columns.</li> </ul>

---

			<ul style="list-style-type: none"> <li>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.</li> <li>Updated index naming convention to avoid duplicates</li> <li>Updated and standardized Getting Started connection help.</li> <li>Added the Advanced Features section to the help of all drivers.</li> <li>Categorized connection property listings in the help for all editions.</li> </ul>
04/15 /2021	7775	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Kerberos authentication is updated to use TCP by default, but will fall back to UDP if a TCP connection cannot be established</li> </ul>
02/02/2021	7703	Twitter	<b>Added</b> <ul style="list-style-type: none"> <li>Added the StreamReadDuration connection property to allow reading streaming data for a specific time period.</li> </ul>

---

## Advanced Features

This section details a selection of advanced features of the Twitter 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 Twitter 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 Twitter 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 Tweets 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 Twitter 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 Twitter Query Evaluation component examines SQL queries and returns information indicating what parts of the query the adapter is not capable of executing natively.

The Twitter 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 Twitter 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 Twitter API.

## INSERT Statements

See [INSERT Statements](#) for a syntax reference and examples, as well as retrieving the new records' Ids.

## UPDATE Statements

The primary key Id is required to update a record. See [UPDATE 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 Twitter 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 } [ <expression> ]
} [ { AND | OR } ... ]

```

## Examples

1. Return all columns:

```
SELECT * FROM Tweets
```

2. Rename a column:

```
SELECT "Text" AS MY_Text FROM Tweets
```

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

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

4. Search data:

```
SELECT * FROM Tweets WHERE From_User_Name = 'twitter'
```

5. The Twitter APIs support the following operators in the WHERE clause: =, AND.

```
SELECT * FROM Tweets WHERE From_User_Name = 'twitter';
```

## 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 From_User_Name, Text INTO
'csv://c:/Tweets.txt' FROM 'Tweets' WHERE From_User_Name = 'twitter'");
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 'Tweets' IN
'csv://filename=c:/Tweets.csv;delimiter=tab' FROM 'Tweets' WHERE From_
User_Name = 'twitter'");
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. To retrieve the Id of the last inserted record use getGeneratedKeys. Additionally, set the **RETURN\_GENERATED\_KEYS** flag of the Statement class when you call prepareStatement.

```
String cmd = "INSERT INTO Tweets (Text) VALUES (?)";
PreparedStatement pstmt = connection.prepareStatement(
    cmd, Statement.RETURN_GENERATED_KEYS);
pstmt.setString(1, "My twitter message 2");
int count = pstmt.executeUpdate();
System.out.println(count+" rows were affected");
ResultSet rs = pstmt.getGeneratedKeys();
while(rs.next()){
    System.out.println(rs.getString("Id"));
}
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:twitter:InitiateOAuth=GETANDREFRESH;"));
String cmd = "DELETE FROM Tweets WHERE Id = ?";
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "123456789");
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 Twitter Adapter models Twitter entities in relational Tables, Views, and Stored Procedures.

API limitations and requirements are documented in this section; you can use the SupportEnhancedSQL feature, set by default, to circumvent most of these limitations.

## Tables

[Tables](#) describes the available tables.

## Views

[Views](#) are tables that cannot be modified. Typically, data that are read-only and cannot be updated are shown as views.

## Stored Procedures

[Stored Procedures](#) are function-like interfaces to the data source. They can be used to search, update, and modify information in the data source.

# Tables

The adapter models the data in Twitter into a list of tables that can be queried using standard SQL statements.

Generally, querying Twitter tables is the same as querying a table in a relational database. Sometimes there are special cases, for example, including a certain column in the WHERE clause might be required to get data for certain columns in the table. This is typically needed for situations where a separate request must be made for each row to get certain columns. These types of situations are clearly documented at the top of the table page linked below.

## Twitter Adapter Tables

Name	Description
<a href="#">DirectMessages</a>	Send direct messages and query messages sent and received by the authenticated user.
<a href="#">Favorites</a>	Create, delete, and query a list of favorite tweets of the authenticated user and allow the user to favorite new tweets or remove existing favorites.
<a href="#">Following</a>	Create, delete, and query a list of users that the current Twitter account is following, otherwise known as friends.



**Tweets**

Create, delete, and query status updates and tweets from the authenticated user.

## DirectMessages

Send direct messages and query messages sent and received by the authenticated user.

### Table Specific Information

Direct messages that have been sent and received by the authenticated user will appear in [DirectMessages](#).

### Select

The Min\_Id and Max\_Id pseudo columns may be used to narrow down a range of direct messages to return, or to return only recent direct messages. The Id may be specified to return a specific direct message.

### Insert

New direct messages may be sent by performing an insert. Note that only Text and the Recipient\_Id may be specified in an insert. In order to attach a media to the message, simply specify the media file paths MediaFilePath or the MediaId for the media file in the INSERT command. It is possible to attach only one media to a message. For example:

```
INSERT INTO DirectMessages (Recipient_Id, Text, MediaFilePath) VALUES  
( '00000000000000000000', 'Hello World', 'C:\\myFile.jpg' )
```

```
INSERT INTO DirectMessages (Recipient_Id, Text, MediaId) VALUES  
( '1001073178217713668', 'Hello World', '1001072801250529280' )
```

### Update

The UPDATE operation is not available on this table.

## Delete

Direct messages may be deleted by using the DELETE operation.

## Columns

Name	Type	ReadOnly	Description
ID [KEY]	<i>String</i>	True	The Id of the direct message.
Created_At	<i>Datetime</i>	True	When the direct message was made.
Text	<i>String</i>	True	The text of the direct message.
Sender_Id	<i>String</i>	True	The Id for the sender of the message.
Source_App_Id	<i>String</i>	True	The name for the sender of the message.
Recipient_ID	<i>String</i>	True	The Id for the recipient of the message.
User_Mentions	<i>String</i>	True	Mentions of other users in the tweet, returned as an XML aggregate.
URLs	<i>String</i>	True	URLs in the tweet, returned as an XML aggregate.
Hashtags	<i>String</i>	True	Hashtags in the tweet, returned as an XML aggregate.
Attachment_Id	<i>String</i>	True	Identifier of the media attached to the message.

Attachment_Url	<i>String</i>	True	Url of the media attached to the message.
Attachment_Type	<i>String</i>	True	Type of media attached to the message.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

<b>Name</b>	<b>Type</b>	<b>Description</b>
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.
Min_ID	<i>String</i>	Specifies the lowest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only direct messages that are equal to or newer than the creation date of the specified Id will be returned. The Min_Id column must be a valid number but does not need to be a valid direct message Id.
Max_ID	<i>String</i>	Specifies the highest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only direct messages that are equal to or older than the creation date of the specified Id will be returned. The Max_Id column must be a valid number but does not need to be a valid direct message Id.
MediaId	<i>String</i>	The media Ids to associate with the tweet.
MediaFilePath	<i>String</i>	The media file path to associate with the tweet.

---

Include_Entities	Boolean	Boolean indicating whether or not to include entities such as URLs, hashtags, and user mentions in the response.  The default value is <i>false</i> .
------------------	---------	---

---

## Favorites

Create, delete, and query a list of favorite tweets of the authenticated user and allow the user to favorite new tweets or remove existing favorites.

### Table Specific Information

The authenticated user's favorites may be viewed from [Favorites](#).

### Select

The Min\_Id and Max\_Id pseudo columns may be used to narrow down a range of tweets to return, or to return only recent tweets. Additionally, the Include\_Entities pseudo column may be specified, which can reduce the size of the response if it is set to false.

### Insert

New tweets may be favorited by performing an insert and specifying the Id of the tweet.

### Update

The UPDATE operation is not available on this table.

### Delete

Favorites may be removed by performing a DELETE operation and specifying the Id of the tweet to remove from Favorites.

### Columns

Name	Type	ReadOnly	Description
ID [KEY]	<i>String</i>	False	The Id of the status update or tweet. Set this value when inserting to retweet an existing tweet.
IDLong	<i>Long</i>	False	The long type Id of the status update or tweet.
Created_At	<i>Datetime</i>	True	When the tweet was made.
Text	<i>String</i>	False	The text of the tweet.
Source	<i>String</i>	True	Source of the tweet.
Favorited	<i>Boolean</i>	True	Boolean indicating if this tweet has been favorited.
Retweet_Count	<i>Integer</i>	True	The number of times the tweet has been retweeted.
From_User_Id	<i>String</i>	True	Id of the user who made the tweet.
From_User	<i>String</i>	True	Screen name of the user who made the tweet.
From_User_Name	<i>String</i>	True	Name of the user who made the tweet.
From_User_Lang	<i>String</i>	True	Language code the from user is using.
From_User_Profile_URL	<i>String</i>	True	URL to the user who made the tweet.

From_User_Profile_Image_URL	<i>String</i>	True	URL to the profile image for the from user.
From_User_Location	<i>String</i>	True	The location of the user.
To_User_Id	<i>String</i>	True	Id of the user whom the tweet was sent to.
To_User_Screen_Name	<i>String</i>	True	Screen name of the user whom the tweet was sent to.
User_Mentions	<i>String</i>	True	Mentions of other users in the tweet, returned as an XML aggregate.
URLs	<i>String</i>	True	URLs in the tweet, returned as an XML aggregate.
Hashtags	<i>String</i>	True	Hashtags in the tweet, returned as an XML aggregate.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

<b>Name</b>	<b>Type</b>	<b>Description</b>
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.
Min_ID	<i>String</i>	Specifies the lowest Id to return results for. Ids are stored

		in increasing numerical order, so specifying this value means that only tweets that are equal to or newer than the creation date of the specified Id will be returned. The Min_Id must be a valid number but does not need to be a valid tweet Id.
Max_ID	<i>String</i>	Specifies the highest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only tweets that are equal to or older than the creation date of the specified Id will be returned. The Max_Id must be a valid number but does not need to be a valid tweet Id.
Include_Entities	<i>Boolean</i>	Boolean indicating whether or not to include entities such as URLs, hashtags, and user mentions in the response.  The default value is <i>false</i> .

## Following

Create, delete, and query a list of users that the current Twitter account is following, otherwise known as friends.

### Table Specific Information

Users whom the authenticated user is following will appear here. Additionally, users whom a specified user is following may be viewed from [Following](#).

### Select

By default, [Following](#) will return users whom the authenticated user is following. By specifying a Following\_User\_Id or Following\_Screen\_Name, users whom the specified user is following will be returned.

LookUpUsers refers to looking up additional information on the users returned from Twitter. Normally, only the Ids of users someone is following are returned from Twitter. If LookUpUsers is set to true, additional queries will be made to Twitter to retrieve user details, such as a Screen\_Name. LookUpUsers is true by default.

MaxUserLookup refers to the maximum number of users to additionally look up information on. A maximum of 100 user details may be returned from Twitter per request. Setting this value higher will cause details for more users to be returned, but at the cost of additional requests. MaxUserLookup defaults to 100.

## Insert

The authenticated user may follow another user by performing an insert based on either the User\_Id or the Screen\_Name.

## Update

The UPDATE operation is not available on this table.

## Delete

The authenticated user may stop following another user by performing a delete based on the Id.

## Columns

Name	Type	ReadOnly	Description
ID [KEY]	String	False	The Id of the user.
Name	String	True	The name of the user.
User_Id	String	False	The Id of the user.
Screen_Name	String	False	The screen name of the user.
Following_User_Id	String	False	Use this in the WHERE clause to



			retrieve the users a specific user is following other than the authenticated user.
Following_Screen_Name	<i>String</i>	False	Use this in the WHERE clause to retrieve the users a specific user is following other than the authenticated user.
Location	<i>String</i>	True	The location of the user.
Profile_URL	<i>String</i>	True	The URL for the user's profile.
Profile_Image_URL	<i>String</i>	True	The URL for the image of the user.
Protected	<i>Boolean</i>	True	The privacy flag of the user. If true, then the user's account is private and only their approved followers can read their tweets or see extended information about them.
Lang	<i>String</i>	True	The ISO language code of the user.
Created_At	<i>Datetime</i>	True	When the user account was created.
Friends_Count	<i>Integer</i>	True	The number of people this user is following.
Followers_Count	<i>Integer</i>	True	The number of followers the user has.
Favourites_Count	<i>Integer</i>	True	The number of favorites the user has.

Statuses_Count	<i>Integer</i>	True	The number of status updates or tweets the user has made.
UTC_Offset	<i>Integer</i>	True	The Coordinated Universal Time offset for the user in seconds.
Time_Zone	<i>String</i>	True	The time zone of the user.
Notifications	<i>Boolean</i>	True	Boolean indicating if the user has notifications enabled.
Geo_Enabled	<i>Boolean</i>	True	Boolean indicating if the user has geo-enabled turned on in their profile.
Verified	<i>Boolean</i>	True	Boolean indicating if the user account has been verified.
Following	<i>Boolean</i>	True	Boolean indicating if the user is following you.
Contributors_Enabled	<i>Boolean</i>	True	Boolean indicating if contributors are enabled for the account. Typically used in multiuser accounts.
Follow_Request_Sent	<i>Boolean</i>	True	If the user is a protected user, this column indicates if the authenticated user has sent a request to follow them.
Listed_Count	<i>Integer</i>	True	The number of public lists a user is listed in. -1 if unknown.
Is_Translator	<i>Boolean</i>	True	Boolean indicating if the user contributes to translating Twitter in other languages.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

Name	Type	Description
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.
LookUpUsers	<i>String</i>	A boolean indicating if you would like to additionally look up details about the user Ids returned. Normally only Ids will be returned from Twitter, but additional requests can be made to Twitter to retrieve user details.  The default value is <i>true</i> .
MaxUserLookup	<i>String</i>	The maximum number of users to perform a lookup on. Each user lookup is limited to a maximum of 100 Ids per request.  The default value is <i>100</i> .

## Tweets

Create, delete, and query status updates and tweets from the authenticated user.

### Table Specific Information

#### Select

Tweets posted on Twitter will appear here. Tweets may come from the authenticated user's home page, they may be searched for, or they may come from a specific user or list. Note that some columns will always be null unless the SearchTerms pseudo column is specified.

By default, [Tweets](#) will return tweets from the authenticated user's home page. By specifying `SearchTerms`, tweets that match the terms used in the search will return. Valid search terms may be found under the topic "Standard Search Operators", located here: <https://developer.twitter.com/en/docs/tweets/search/guides/standard-operators>.

*Note:* `SearchTerms` may not be used with `Screen_Name`, `User_Id`, `List_Id`, `Slug`, `Owner_User_Id`, or `Owner_Screen_Name`.

The `Screen_Name` and `User_Id` pseudo columns can be used to return tweets made by the specified user. Alternatively, use the `From_User_Name` and `From_User_Id` columns to do the same thing.

The `List_Id` pseudo column may be specified to return tweets made in a specific list. Alternatively, the list may be specified using the `Slug` and either `Owner_User_Id` or `Owner_Screen_Name`.

The `Min_Id` and `Max_Id` pseudo columns may be used to narrow down a range of tweets to return, or to return only recent tweets. Ids are created in increasing numerical order on Twitter. Specifying a `Min_Id` returns only results with a greater Id or tweets that were created more recently than the specified one. Setting a `Max_Id` will return only tweets that are older than the creation date of the specified Id. Note that while these values must be valid, non-negative numbers, they do not have to be Ids that exist.

## Full Archive and 30-day Searches

By default, Twitter only supports returning Tweets from searches that are up to 7 days old. However, customers with premium accounts may retrieve tweets older than this by executing either a 30-day or full archive search.

The `EnvType` and `DevEnvironment` pseudo columns may be used to query Premium Search APIs. `EnvType` specifies the environment type. It can take one of `30day` or `fullarchive` values. The default value is `30day`. The `DevEnvironment` is required in order to query the Premium Search API and must be set to the value of the dev environment label, created in your Twitter Developer Account. You should use a Premium or Enterprise app in your OAuth connection properties to use the feature. A `SearchTerms` value must be specified in order to use this feature.

## Insert

In order to insert a tweet, simply specify the text of the tweet in the `INSERT` command. For example:

```
INSERT INTO Tweets (Text) VALUES ('Hello World')
```

In order to insert a tweet with media, specify the text of the tweet and the media file paths MediaFilePath for every media file in the INSERT command. You may include up to 4 photos or 1 animated GIF or 1 video in a Tweet. For example:

```
INSERT INTO Tweets (Text, MediaFilePath) VALUES ('Hello
World', 'C:\\myfile.jpg,C:\\myfile2.jpg')
```

If you already have media ids you can insert them in a tweet by providing media ids MediaId of every media, in INSERT command. For example:

```
INSERT INTO Tweets (Text, MediaId) VALUES ('Hello
World', '123456789,123456789')
```

In order to retweet an existing tweet, specify the Id column when inserting the tweet. For example:

```
INSERT INTO Tweets (Id) VALUES ('123456789')
```

## Columns

Name	Type	ReadOnly	Description
ID [KEY]	<i>String</i>	False	The Id of the status update or tweet. Set this value when inserting to retweet an existing tweet.
IDLong	<i>String</i>	False	The long type Id of the status update or tweet.
SearchTerms	<i>String</i>	False	The SearchTerms to search against. This cannot be used with the Screen_Name or User_Id inputs. For more information on using the advanced query

			operators, see the Twitter API documentation here: <a href="https://dev.twitter.com/docs/using-search">https://dev.twitter.com/docs/using-search</a> . Roughly 1500 results can be returned using the SearchTerms.
Created_At	<i>Datetime</i>	True	When the tweet was made.
Text	<i>String</i>	False	The text of the tweet.
Lang	<i>String</i>	True	Language code the tweet was made in.
Source	<i>String</i>	True	Source of the tweet.
Favorited	<i>Boolean</i>	True	Boolean indicating if this tweet has been favorited.
Favorite_Count	<i>Integer</i>	True	The approximate number of times this tweet has been favorited.
Retweeted	<i>Boolean</i>	True	Boolean indicating if this tweet has been retweeted.
Retweet_Count	<i>Integer</i>	True	The number of times the tweet has been retweeted.
Retweeted_Status_Id	<i>String</i>	True	Id of the tweet which was retweeted by this one. Empty if the current tweet is not a retweet.
Truncated	<i>Boolean</i>	True	Boolean indicating if this tweet has been truncated.
Filter_Level	<i>String</i>	True	Indicates the maximum value of the Filter_Level parameter that

			can be used and still stream this tweet.
Possibly_Sensitive	<i>String</i>	True	This field is available only when a tweet contains a link. The meaning of the field does not pertain to the tweet content itself, but instead it is an indicator that the URL contained in the tweet may contain content or media identified as sensitive content.
Withheld_Copyright	<i>Boolean</i>	True	When present and set to true, indicates that this piece of content has been withheld due to a DMCA complaint.
Withheld_Scope	<i>String</i>	True	When present, indicates whether the content being withheld is the status or a user.
Withheld_In_Countries	<i>String</i>	True	A list of uppercase, two-letter country codes this content is withheld from.
Contributors	<i>String</i>	True	An XML collection of user objects (usually only one) indicating users who contributed to the authorship of the tweet, on behalf of the official tweet author.
Coordinates_Coordinates	<i>String</i>	True	The geographic coordinates of this tweet (longitude first, then latitude).
Coordinates_Type	<i>String</i>	True	The type of coordinate, if applicable.
Place_Full_Name	<i>String</i>	True	The full name of the location of this tweet (city and state).

Place_Country	<i>String</i>	True	The country of origin of this tweet.
Current_User_Retweet_Id	<i>String</i>	True	Details the tweet Id of the authenticated user's own retweet (if it exists) of this tweet.
Scopes	<i>String</i>	True	A set of key-value pairs indicating the intended contextual delivery of the containing tweet. Currently used by Twitter's promoted products.
In_Reply_To_Status_Id	<i>String</i>	True	Represents the Id of the original status if this tweet is in reply to another.
From_User_Id	<i>String</i>	True	Id of the user who made the tweet. Use this in the WHERE clause to get tweets for the specified user.
From_User_Screen_Name	<i>String</i>	True	Screen name of the user who made the tweet. Use this in the WHERE clause to get tweets for the specified user.
From_User_Name	<i>String</i>	True	Name of the user who made the tweet.
From_User_Location	<i>String</i>	True	Location of the user who made the tweet.
From_User_Profile_URL	<i>String</i>	True	URL to the user who made the tweet. This is not returned when a SearchTerms is specified.
From_User_Profile_Image_Url	<i>String</i>	True	URL to the profile image for the from user.



To_User_Id	<i>String</i>	True	Id of the user who made the tweet. Use this in the WHERE clause to get tweets for the specified user.
To_User_Screen_Name	<i>String</i>	True	Screen name of the user who made the tweet.
User_Mentions	<i>String</i>	True	Mentions of other users in the tweet, returned as an XML aggregate.
URLs	<i>String</i>	True	URLs in the tweet, returned as an XML aggregate. If SearchTerms is specified, set Include_Entities=true to retrieve URLs.
Hashtags	<i>String</i>	True	Hashtags in the tweet, returned as an XML aggregate. If SearchTerms is specified, set Include_Entities=true to retrieve Hashtags.
Media	<i>String</i>	True	Media in the tweet, returned as an XML aggregate. If SearchTerms is specified, set Include_Entities=true to retrieve Media.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

<b>Name</b>	<b>Type</b>	<b>Description</b>
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the

		original request are still specified when using the NextPageToken.
Min_ID	<i>String</i>	Specifies the lowest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only tweets that are equal to or newer than the creation date of the specified Id will be returned. The Min_Id must be a valid number but does not need to be a valid tweet Id.
Max_ID	<i>String</i>	Specifies the highest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only tweets that are equal to or older than the creation date of the specified Id will be returned. The Max_Id must be a valid number but does not need to be a valid tweet Id.
Include_Entities	<i>Boolean</i>	Boolean indicating whether or not to include entities such as URLs, hashtags, and user mentions in the response.  The default value is <i>false</i> .
Include_Retweets	<i>Boolean</i>	Boolean indicating whether or not to include retweets in the result set. Only useful when no filters are specified (listing own tweets), From_User_Id, From_User_Screen_Name are specified (listing a user's tweets) or when List_Id is specified (listing the tweets from a list of User ids). Has no effect when used with SearchTerms. By default twitter includes retweets.  The default value is <i>true</i> .
Result_Type	<i>String</i>	Indicates the type of results to return when using a SearchTerms. Currently Twitter has only popular and recent results.  The allowed values are <i>popular</i> , <i>recent</i> .
List_Id	<i>String</i>	Used to specify the list Id to retrieve tweets from. This value does not work with SearchTerms, Screen_

		Name, or User_Id.
Slug	<i>String</i>	Alternative method of specifying a list. Use this in with an Owner_User_Id and Owner_Screen_Name to specify a list without a List_Id.
Owner_User_Id	<i>String</i>	Alternative method of specifying a list. Use this with a Slug to specify a list without the list Id.
Owner_Screen_Name	<i>String</i>	Alternative method of specifying a list. Use this with a Slug to specify a list without the List_Id.
Geocode	<i>String</i>	If this parameter is used with a SearchTerms, the query will return tweets from the specified geographical location. The geocode is specified in this manner: Lat,Long,Radius. For example: 37.781157,-122.398720,25mi
RetweetId	<i>String</i>	When making a new tweet, specify this value to retweet the specified tweet.
MediaId	<i>String</i>	A comma-separated value of media Ids to associate with the tweet. You may include up to 4 photos or 1 animated GIF or 1 video in a tweet.
MediaFilePath	<i>String</i>	A comma-separated value of media file paths to associate with the tweet. You may include up to 4 photos or 1 animated GIF or 1 video in a tweet.
EnvType	<i>String</i>	The environment type you want to use.  The allowed values are <i>30day</i> , <i>fullarchive</i> .  The default value is <i>30day</i> .
DevEnvironment	<i>String</i>	To begin using the new Premium APIs, you need to setup one or more dev environments for the endpoint and connect it to an app.

## Views

Views are composed of columns and pseudo columns. Views are similar to tables in the way that data is represented; however, views do not support updates. Entities that are represented as views are typically read-only entities. Often, a stored procedure is available to update the data if such functionality is applicable to the data source.

Queries can be executed against a view as if it were a normal table, and the data that comes back is similar in that regard.

Dynamic views, such as queries exposed as views, and views for looking up specific combinations of project\_team work items are supported.

## Twitter Adapter Views

Name	Description
<a href="#">AccountSettings</a>	Query account settings about the currently authenticated user.
<a href="#">Followers</a>	Query a list of users following the current Twitter account.
<a href="#">ListMembers</a>	Query the members of a specified list.
<a href="#">Lists</a>	Query Twitter list information based on a set of criteria.
<a href="#">ListSubscribers</a>	Query the subscribers to a specified list.
<a href="#">Mentions</a>	Query the most recent mentions (tweet containing @username) for the authenticating user.
<a href="#">Retweets</a>	Query a list of retweets of the authenticated user.
<a href="#">Trends</a>	Query the daily trending topics from Twitter.
<a href="#">TweetStream</a>	Query public data flowing through Twitter.
<a href="#">Users</a>	Query a list of users based on the SearchTerms, Id, or Screen_Name.

## AccountSettings

Query account settings about the currently authenticated user.

## Columns

Name	Type	Description
Screen_Name [KEY]	<i>String</i>	The screen name of the currently authenticated user.
Always_Use_Https	<i>Boolean</i>	A boolean indicating if the user has specified in their user settings to always use HTTPS URLs.
Discoverable_By_Email	<i>Boolean</i>	A boolean indicating if the user can be found by email. This can be enabled in the 'let others find me by my email address' check box.
Discoverable_By_Mobile	<i>Boolean</i>	A boolean indicating if the user can be found by their mobile number.
Geo_Enabled	<i>Boolean</i>	A boolean indicating if the user has enabled adding locations to their tweets.
Language	<i>String</i>	The default language code for the user. For example: en.
Protected	<i>Boolean</i>	A boolean indicating if the user has selected the Protect My Tweets setting.
Show_All_Inline_Media	<i>Boolean</i>	A boolean indicating if the user has enabled all media to be displayed in tweets.
Sleep_Time_Enabled	<i>Boolean</i>	A boolean indicating if the user has selected to turn off updates during certain hours if they have added a mobile phone to their account.
Sleep_Time_Start_Time	<i>String</i>	The start time for the range when Twitter updates will not be submitted to the user's mobile phone.

Sleep_Time_End_Time	<i>String</i>	An ending time for the range when Twitter updates will not be submitted to the user's mobile phone.
Time_Zone_Name	<i>String</i>	The name of the time zone the user is located in.
Time_Zone_TZInfo_Name	<i>String</i>	A more specific location for the time zone the user is located in.
Time_Zone_Utc_Offset	<i>Int</i>	The Coordinated Universal Time offset in seconds from GMT.
Trend_Location	<i>String</i>	An XML aggregate of trending locations for the user.

## Followers

Query a list of users following the current Twitter account.

### View Specific Information

The authenticated user's followers and a specified user's followers may be viewed from [Followers](#).

By default, [Followers](#) will return the followers of the authenticated user. By specifying a Followers\_Of\_User\_Id or Followers\_Of\_Screen\_Name, the followers of the specified user will be returned.

LookUpUsers refers to looking up additional information about the followers returned from Twitter. Normally only the Ids of followers are returned from Twitter. If LookUpUsers is set to true, additional queries will be made to Twitter to retrieve user details, such as the Screen\_Name. LookUpUsers is true by default.

MaxUserLookup refers to the maximum number of users to additionally look up information on. A maximum of 100 user details may be returned from Twitter per request. Setting this value higher will cause details for more users to be returned, but at the cost of additional requests. MaxUserLookup defaults to 100.

## Columns

Name	Type	Description
ID [KEY]	<i>String</i>	The Id of the user.
Name	<i>String</i>	The name of the user.
User_Id	<i>String</i>	The Id of the user.
Screen_Name	<i>String</i>	The screen name of the user.
Followers_Of_User_Id	<i>String</i>	Use this in the WHERE clause to retrieve the users a specific user is following other than the authenticated user.
Followers_Of_Screen_Name	<i>String</i>	Use this in the WHERE clause to retrieve the users a specific user is following other than the authenticated user.
Location	<i>String</i>	The location of the user.
Profile_URL	<i>String</i>	The URL for the user's profile.
Profile_Image_URL	<i>String</i>	The URL for the image of the user.
Protected	<i>Boolean</i>	The privacy flag of the user. If true, then the user's account is private and only their approved followers can read their tweets or see extended information about them.

Lang	<i>String</i>	The ISO language code of the user.
Created_At	<i>Datetime</i>	When the user account was created.
Friends_Count	<i>Integer</i>	The number of people this user is following.
Followers_Count	<i>Integer</i>	The number of followers the user has.
Favourites_Count	<i>Integer</i>	The number of favorites the user has.
Statuses_Count	<i>Integer</i>	The number of status updates or tweets the user has made.
UTC_Offset	<i>Integer</i>	The Coordinated Universal Time offset for the user in seconds.
Time_Zone	<i>String</i>	The time zone of the user.
Notifications	<i>Boolean</i>	Boolean indicating if the user has notifications enabled.
Geo_Enabled	<i>Boolean</i>	Boolean indicating if the user has geo-enabled turned on in their profile.
Verified	<i>Boolean</i>	Boolean indicating if the user account has been verified.
Following	<i>Boolean</i>	Boolean indicating if the user is following you.
Contributors_Enabled	<i>Boolean</i>	Boolean indicating if contributors are enabled for the account. Typically used in multiuser accounts.



Follow_Request_Sent	<i>Boolean</i>	If the user is a protected user, indicates if the authenticated user has sent a request to follow them.
Listed_Count	<i>Integer</i>	The number of public lists a user is listed in. -1 if unknown.
Is_Translator	<i>Boolean</i>	Boolean indicating if the user contributes to translating Twitter in other languages.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

<b>Name</b>	<b>Type</b>	<b>Description</b>
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.
LookUpUsers	<i>String</i>	A boolean indicating if you would like to additionally look up details about the user Ids returned. Normally only Ids will be returned from Twitter, but additional requests can be made to Twitter to retrieve user details.  The default value is true.
MaxUserLookup	<i>String</i>	The maximum number of users to perform a lookup on. Each user lookup is limited to a maximum of 100 Ids per request.  The default value is 100.

## ListMembers

Query the members of a specified list.

## View Specific Information

Members of a specified list can be found under [ListMembers](#).

The column List\_Id can be used to specify the Id of a list you wish to obtain the members of.

## Columns

Name	Type	Description
ID [KEY]	<i>String</i>	The Id of the user.
List_Id	<i>String</i>	The Id of the list. Used to specify the list Id to retrieve tweets from.  The default value is 2031945.
Name	<i>String</i>	The name of the user.
Screen_Name	<i>String</i>	The screen name of the user.
Location	<i>String</i>	The location of the user.
Profile_URL	<i>String</i>	The URL for the user's profile.
Profile_Image_URL	<i>String</i>	The URL for the image of the user.
Protected	<i>Boolean</i>	The privacy flag of the user. If true, then the user's account is private and only their approved followers can read their tweets or see extended information about them.

Lang	<i>String</i>	The ISO language code of the user.
Created_At	<i>Datetime</i>	When the user account was created.
Friends_Count	<i>Integer</i>	The number of people this user is following.
Followers_Count	<i>Integer</i>	The number of followers the user has.
Favourites_Count	<i>Integer</i>	The number of favorites the user has.
Statuses_Count	<i>Integer</i>	The number of status updates or tweets the user has made.
UTC_Offset	<i>Integer</i>	The Coordinated Universal Time offset for the user in seconds.
Time_Zone	<i>String</i>	The time zone of the user.
Notifications	<i>Boolean</i>	Boolean indicating if the user has notifications enabled.
Geo_Enabled	<i>Boolean</i>	Boolean indicating if the user has geo-enabled turned on in their profile.
Verified	<i>Boolean</i>	Boolean indicating if the user account has been verified.
Following	<i>Boolean</i>	Boolean indicating if the user is following you.
Contributors_Enabled	<i>Boolean</i>	Boolean indicating if contributors are enabled for the account. Typically used in multiuser accounts.
Follow_Request_Sent	<i>Boolean</i>	If the user is a protected user, this column indicates

		if the authenticated user has sent a request to follow them.
Listed_Count	<i>Integer</i>	The number of public lists a user is listed in. -1 if unknown.
Is_Translator	<i>Boolean</i>	Boolean indicating if the user contributes to translating Twitter in other languages.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

Name	Type	Description
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.

## Lists

Query Twitter list information based on a set of criteria.

### View Specific Information

Information on Twitter lists may be obtained in [Lists](#). By default, lists that the authenticated user is subscribed to will appear. Lists may returned in several different ways using both pseudo columns and normal columns.

Specify the Subscriber\_Id or Subscriber\_Screen\_Name in the WHERE clause of the request to return lists that the specified Twitter user is subscribed to. Other pseudo columns may not be used when specifying these values.

Specify the Member\_Id or Member\_Screen\_Name in the WHERE clause of the request to return lists that the specified Twitter user is a member of. Other pseudo columns may not be used when specifying these values.

Specify both the Slug and Owner\_User\_Id or only the Owner\_Screen\_Name to return one specific list. This is an alternative to specifying the Id of the list and will return only one result.

## Columns

Name	Type	Description
ID [KEY]	<i>String</i>	The Id of the list.
Name	<i>String</i>	The name of the list.
Full_Name	<i>String</i>	The full name of the list.
Slug	<i>String</i>	The Slug or name of the list.
Description	<i>String</i>	A description of the list.
Subscriber_Count	<i>Integer</i>	The number of subscribers to the list.
Member_Count	<i>Integer</i>	The number of members in the list.
Created_At	<i>Datetime</i>	When the list was created.
Following	<i>Boolean</i>	Boolean indicating if the authenticated user is following the list.

Mode	<i>String</i>	What mode the list is set to.
Owner_Id	<i>String</i>	User Id for the owner of the list.
Owner_Name	<i>String</i>	Name for the owner of the list.
Owner_Screen_Name	<i>String</i>	Screen name for the owner of the list.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

<b>Name</b>	<b>Type</b>	<b>Description</b>
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.
Subscriber_Id	<i>String</i>	Used to retrieve lists the specified User_Id is subscribed to.
Subscriber_Screen_Name	<i>String</i>	Used to retrieve lists the specified User_Screen_Name is subscribed to.
Member_Id	<i>String</i>	Used to retrieve lists the specified User_Id is a member of.
Member_Screen_Name	<i>String</i>	Used to retrieve lists the specified User_Screen_Name is a member of.

Owner_User_Id	<i>String</i>	Alternative method of specifying a list. Use this with a Slug to specify a list without the List_Id.
Filter_To_Owned_Lists	<i>Boolean</i>	When set to true or 1 , will return just lists the authenticating user owns, and the user represented by user_id or screen_name is a member of.

## ListSubscribers

Query the subscribers to a specified list.

### View Specific Information

Subscribers of a specified list can be found under [ListSubscribers](#).

The column List\_Id can be used to specify the Id of a list you wish to obtain the members of.

### Columns

Name	Type	Description
ID [KEY]	<i>String</i>	The Id of the user.
List_Id	<i>String</i>	Used to specify the list Id to retrieve tweets from. The default value is 2031945.
Name	<i>String</i>	The name of the user.
Screen_Name	<i>String</i>	The screen name of the user.

Location	<i>String</i>	The location of the user.
Profile_URL	<i>String</i>	The URL for the user's profile.
Profile_Image_URL	<i>String</i>	The URL for the image of the user.
Protected	<i>Boolean</i>	The privacy flag of the user. If true, then the user's account is private and only their approved followers can read their tweets or see extended information about them.
Lang	<i>String</i>	The ISO language code of the user.
Created_At	<i>Datetime</i>	When the user account was created.
Friends_Count	<i>Integer</i>	The number of people this user is following.
Followers_Count	<i>Integer</i>	The number of followers the user has.
Favourites_Count	<i>Integer</i>	The number of favorites the user has.
Statuses_Count	<i>Integer</i>	The number of status updates or tweets the user has made.
UTC_Offset	<i>Integer</i>	The Coordinated Universal Time offset for the user in seconds.
Time_Zone	<i>String</i>	The time zone of the user.
Notifications	<i>Boolean</i>	Boolean indicating if the user has notifications enabled.



Geo_Enabled	<i>Boolean</i>	Boolean indicating if the user has geo-enabled turned on in their profile.
Verified	<i>Boolean</i>	Boolean indicating if the user account has been verified.
Following	<i>Boolean</i>	Boolean indicating if the user is following you.
Contributors_Enabled	<i>Boolean</i>	Boolean indicating if contributors are enabled for the account. Typically used in multiuser accounts.
Follow_Request_Sent	<i>Boolean</i>	If the user is a protected user, this column indicates if the authenticated user has sent a request to follow them.
Listed_Count	<i>Integer</i>	The number of public lists a user is listed in. -1 if unknown.
Is_Translator	<i>Boolean</i>	Boolean indicating if the user contributes to translating Twitter in other languages.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

<b>Name</b>	<b>Type</b>	<b>Description</b>
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.

## Mentions

Query the most recent mentions (tweet containing @username) for the authenticating user.

## Columns

Name	Type	Description
ID [KEY]	<i>String</i>	The Id of the status update or tweet. Set this value when inserting to instead retweet an existing tweet.
IDLong	<i>Long</i>	The long type Id of the status update or tweet.
Created_At	<i>Datetime</i>	When the tweet was made.
Text	<i>String</i>	The text of the tweet.
Source	<i>String</i>	Source of the tweet.
Favorited	<i>Boolean</i>	Boolean indicating if this tweet has been favorited.
Favorite_Count	<i>Integer</i>	The approximate number of times this tweet has been favorited.
Retweet_Count	<i>Integer</i>	The number of times the tweet has been retweeted.
Retweeted_Status_Id	<i>String</i>	Id of the tweet which was retweeted by this one. Empty if the current tweet is not a retweet.
User_Id	<i>String</i>	Id of the user who made the tweet.

User_Name	<i>String</i>	Name of the user who made the tweet.
User_Screen_Name	<i>String</i>	Screen name of the user who made the tweet.
User_Location	<i>String</i>	Location of the user who made the tweet.
User_Profile_URL	<i>String</i>	URL to the profile of the user who made the tweet.
User_Profile_Image_URL	<i>String</i>	URL to the user's profile image.
User_Mentions	<i>String</i>	Mentions of other users in the tweet, returned as an XML aggregate.
URLs	<i>String</i>	URLs in the tweet, returned as an XML aggregate.
Hashtags	<i>String</i>	Hashtags in the tweet, returned as an XML aggregate.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

<b>Name</b>	<b>Type</b>	<b>Description</b>
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.
Min_ID	<i>String</i>	Specifies the lowest Id to return results for. Ids are stored

		in increasing numerical order, so specifying this value means that only tweets that are equal to or newer than the creation date of the specified Id will be returned. The Min_Id must be a valid number but does not need to be a valid tweet Id.
Max_ID	<i>String</i>	Specifies the highest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only tweets that are equal to or older than the creation date of the specified Id will be returned. The Max_Id must be a valid number but does not need to be a valid tweet Id.
Include_Entities	<i>Boolean</i>	Boolean indicating whether or not to include entities such as URLs, hashtags, and user mentions in the response.  The default value is false.
Include_Retweets	<i>Boolean</i>	Boolean indicating whether or not to include retweets in the result set.  The default value is true.

## Retweets

Query a list of retweets of the authenticated user.

### View Specific Information

#### Select

Tweets from the authenticated user that have been retweeted by other users will appear in [Retweets](#) by default.

The Min\_Id and Max\_Id pseudo columns may be used to narrow down a range of retweets to return, or to return only recent retweets. Ids are created in increasing numerical order on Twitter. Specifying a Min\_Id will return only results with a greater Id or tweets that were created more recently than the specified one. Setting a Max\_Id will return only tweets that

are older than the creation date of the specified Id. Note that while these values must be valid, non-negative numbers, they do not have to be Ids that exist.

## Columns

Name	Type	Description
ID [KEY]	<i>String</i>	The Id of the status update or tweet.
IDLong	<i>Long</i>	The long type Id of the status update or tweet.
Created_At	<i>Datetime</i>	When the tweet was made.
Text	<i>String</i>	The text of the tweet.
Lang	<i>String</i>	Language code the tweet was made in.
Source	<i>String</i>	Source of the tweet.
Favorited	<i>Boolean</i>	Boolean indicating if this tweet has been favorited.
Favorite_Count	<i>Integer</i>	The approximate number of times this tweet has been favorited.
Retweeted	<i>Boolean</i>	Boolean indicating if this tweet has been retweeted.
Retweet_Count	<i>Integer</i>	The number of times the tweet has been retweeted.
Retweeted_Status_Id	<i>String</i>	Id of the tweet which was retweeted by this one.

		Empty if the current tweet is not a retweet.
Truncated	<i>Boolean</i>	Boolean indicating if this tweet has been truncated.
Filter_Level	<i>String</i>	Indicates the maximum value that may be used and still stream this tweet.
Possibly_Sensitive	<i>String</i>	This field is available only when a tweet contains a link. The meaning of the field does not pertain to the tweet content itself, but instead it is an indicator that the URL contained in the tweet may contain content or media identified as sensitive content.
Withheld_Copyright	<i>Boolean</i>	When present and set to true, it indicates that this piece of content has been withheld due to a DMCA complaint.
Withheld_Scope	<i>String</i>	When present, this column indicates whether the content being withheld is the status or a user.
Withheld_In_Countries	<i>String</i>	A list of uppercase, two-letter country codes of the countries this content is withheld from.
Contributors	<i>String</i>	An XML collection of user objects (usually only one) indicating users who contributed to the authorship of the tweet, on behalf of the official tweet author.
Coordinates_Coordinates	<i>String</i>	The geographic coordinates of this tweet (longitude first, then latitude).
Coordinates_Type	<i>String</i>	The type of coordinate, if applicable.
Place_Full_Name	<i>String</i>	The full name of the location of this tweet (city and state).
Place_Country	<i>String</i>	The country of origin of this tweet.

Current_User_Retweet_Id	<i>String</i>	Details the tweet Id of the authenticated user's own retweet (if it exists) of this tweet.
Scopes	<i>String</i>	A set of key-value pairs indicating the intended contextual delivery of the containing tweet. Currently used by Twitter's promoted products.
In_Reply_To_Status_Id	<i>String</i>	Represents the Id of the original status if this tweet is in reply to another.
User_Id	<i>String</i>	Id of the user who made the tweet. Use this in the WHERE clause to get retweets for a specific user other than the authenticated user.
User_Name	<i>String</i>	Name of the user who made the tweet.
User_Screen_Name	<i>String</i>	Screen name of the user who made the tweet.
User_Location	<i>String</i>	Location of the user who made the tweet.
User_Profile_URL	<i>String</i>	URL to the user who made the tweet.
User_Profile_Image_Url	<i>String</i>	URL to the profile image for the user who made the tweet.
User_Mentions	<i>String</i>	Mentions of other users in the tweet, returned as an XML aggregate.
URLs	<i>String</i>	URLs in the tweet, returned as an XML aggregate.
Hashtags	<i>String</i>	Hashtags in the tweet, returned as an XML aggregate.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

Name	Type	Description
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.
Min_ID	<i>String</i>	Specifies the lowest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only retweets that are equal to or newer than the creation date of the specified Id will be returned. The Min_Id must be a valid number but does not need to be a valid retweet Id.
Max_ID	<i>String</i>	Specifies the highest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only retweets that are equal to or older than the creation date of the specified Id will be returned. The Max_Id must be a valid number but does not need to be a valid retweet Id.
Include_Entities	<i>Boolean</i>	Boolean indicating whether or not to include entities such as URLs, hashtags, and user mentions in the response.  The default value is false.

## Trends

Query the daily trending topics from Twitter.

## Columns



Name	Type	Description
ID [KEY]	<i>String</i>	A unique Id for the trending topic, denoted by the syntax Date   #
Date	<i>Datetime</i>	The date for the trending topic.
Topic	<i>String</i>	The name of the trending topic.
Woeld	<i>String</i>	A Where on Earth Id for the location the topic is trending. Input this to keep from making an extra request to get the woeld from Twitter each time.
Country	<i>String</i>	The country the topic is trending in.
CountryCode	<i>String</i>	The country code the topic is trending in.
Latitude	<i>String</i>	The latitude where trends are being searched for. The default value is 40.7142.
Longitude	<i>String</i>	The longitude where trends are being searched for. The default value is -74.0064.
Search_Terms	<i>String</i>	The search terms you would need to search for this trending topic.
Events	<i>String</i>	Any events associated with the topic.
Url	<i>String</i>	A URL to the Twitter search for this topic.
Promoted_Content	<i>String</i>	Any promoted content that is associated with the topic.

# TweetStream

Query public data flowing through Twitter.

## View Specific Information

TweetStream is a stream that will continuously read public Tweets until the operation is killed.

## Select

Below are the fields that can be specified to filter the results.

<b>Follow</b>	A comma-separated list of user Ids, indicating the users you are following whose Tweets should be delivered on the stream.
<b>Track</b>	A comma-separated list of phrases that will be used to deliver the Tweets on the stream containing the phrases.
<b>Language</b>	Return Tweets that have been detected as being written in the specified list of comma-separated languages. You must use BCP 47 language identifiers, such as 'en','fr','ja'. Twitter will respond with a "406 Not Acceptable" error if the coordinates entered were incorrect.
<b>Locations</b>	A comma-separated list of longitude and latitude pairs specifying a set of bounding boxes to filter Tweets by. You must use at least two pairs of longitude and latitude coordinates, with the first pair indicating the southwest bound of the region you will search in.
<b>BackfillCount</b>	When reconnecting to a streaming endpoint, the BackfillCount parameter may be included to attempt to backfill missed messages that occurred during the disconnect period. The supplied value can be an integer from 1 to 150000 or from -1 to -150000. If a positive number is specified, the stream will transition to live values once the backfilled values have been delivered to the client. If a negative number is specified, the stream will disconnect once the backfilled values have been delivered to the client. This filter is only allowed to users with these elevated access roles: Firehose, Links, BirdDog and Shadow. Twitter will respond with a "416

	Requested Range Not Satisfiable" error if your role does not support this filter.
<b>Filter_Level</b>	The level of tweets returned. The default value is none, which includes all available Tweets.

```
SELECT * FROM TweetStream WHERE Follow = '51192312'
```

If either Follow, Track, or Language, is not specified, a small random sample of all public Tweets will be returned, and using any other filter is not going to affect Twitter's streaming results.

Retrieve a large number of tweets that contain the word 'Assembly' in the tweet's body:

```
SELECT * FROM TweetStream WHERE Track = 'Assembly' AND Filter_
Level='low'
```

The following filter will retrieve all tweets originating from New York City:

```
SELECT * FROM TweetStream WHERE Locations = '-74,40,-73,41'
```

Retrieve tweets in the French or English language containing the word 'fillet', and if disconnected, stop retrieving after 50 tweets:

```
SELECT * FROM TweetStream WHERE Track = 'fillet' AND Language = 'fr,en'
AND BackfillCount='-50'
```

## Columns

Name	Type	Description
ID	<i>String</i>	The Id of the status update or tweet. Set this value when inserting to retweet an existing tweet.

Created_At	<i>Datetime</i>	When the tweet was made.
Text	<i>String</i>	The text of the tweet.
Lang	<i>String</i>	Language code the tweet was made in.
Source	<i>String</i>	Source of the tweet.
Favorited	<i>Boolean</i>	Boolean indicating if this tweet has been favorited.
Favorite_Count	<i>Integer</i>	The approximate number of times this tweet has been favorited.
Retweeted	<i>Boolean</i>	Boolean indicating if this tweet has been retweeted.
Retweet_Count	<i>Integer</i>	The number of times the tweet has been retweeted.
Retweeted_Status_Id	<i>String</i>	Id of the tweet which was retweeted by this one. Empty if the current tweet is not a retweet.
Truncated	<i>Boolean</i>	Boolean indicating if this tweet has been truncated.
Filter_Level	<i>String</i>	Indicates the maximum value of the Filter_Level parameter that can be used and still stream this tweet.
Possibly_Sensitive	<i>String</i>	This field is available only when a tweet contains a link. The meaning of the field does not pertain to the tweet content itself, but instead it is an indicator that the URL contained in the tweet may contain content

		or media identified as sensitive content.
Contributors	<i>String</i>	A JSON collection of user objects (usually only one) indicating users who contributed to the authorship of the tweet, on behalf of the official tweet author.
Coordinates_Coordinates	<i>String</i>	The geographic coordinates of this tweet (longitude first, then latitude).
Coordinates_Type	<i>String</i>	The type of coordinate, if applicable.
Place_Full_Name	<i>String</i>	The full name of the location of this tweet (city and state).
Place_Country	<i>String</i>	The country of origin of this tweet.
Current_User_Retweet_Id	<i>String</i>	Details the tweet Id of the authenticated users own retweet (if it exists) of this tweet.
In_Reply_To_Status_Id	<i>String</i>	Represents the Id of the original status if this tweet is in reply to another.
In_Reply_To_User_Id	<i>String</i>	Represents the Id of the original user if this tweet is in reply to another.
From_User_Id	<i>String</i>	Id of the user who made the tweet. Use this in the WHERE clause to get tweets for the specified user.
From_User_Screen_Name	<i>String</i>	Screen name of the user who made the tweet. Use this in the WHERE clause to get tweets for the specified user.
From_User_Name	<i>String</i>	Name of the user who made the tweet.
From_User_Location	<i>String</i>	Location of the user who made the tweet.

From_User_Profile_URL	<i>String</i>	URL to the user who made the tweet. This is not returned when a SearchTerms is specified.
From_User_Profile_Image_Url	<i>String</i>	URL to the profile image for the from user.
User_Mentions	<i>String</i>	Mentions of other users in the tweet, returned as an JSON aggregate.
URLs	<i>String</i>	URLs in the tweet, returned as an XML aggregate.
Hashtags	<i>String</i>	Hashtags in the tweet, returned as an XML aggregate.
Follow	<i>String</i>	A comma separated list of user IDs, indicating the users whose Tweets should be delivered on the stream. Following protected users is not supported.
Track	<i>String</i>	A comma separated list of phrases which will be used to determine what Tweets will be delivered on the stream.
Language	<i>String</i>	Setting this parameter to a comma-separated list of BCP 47 language identifiers corresponding to any of the languages listed on Twitter's advanced search page will only return Tweets that have been detected as being written in the specified languages.
Locations	<i>String</i>	A comma separated list of longitude,latitude pairs specifying a set of bounding boxes to filter Tweets by. Only geolocated Tweets falling within the requested bounding boxes will be included unlike the Search API, the location of the user field is not used to filter Tweets.

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

Name	Type	Description
BackfillCount	<i>String</i>	When reconnecting to a streaming endpoint, the count parameter may be included to attempt to backfill missed messages which occurred during the disconnect period. The supplied value may be an integer from 1 to 150000 or from -1 to -150000. If a positive number is specified, the stream will transition to live values once the backfill has been delivered to the client. If a negative number is specified, the stream will disconnect once the backfill has been delivered to the client.

## Users

Query a list of users based on the SearchTerms, Id, or Screen\_Name.

### View Specific Information

Users may be obtained via a search or specified in a list of Ids or screen names.

SearchTerms may be used to search for users in a method that is similar to the Find People search on Twitter.com. Up to the first 1000 matches will be available using SearchTerms.

You can specify either the Screen\_Name or User\_Id pseudo columns to perform a lookup. Up to 100 screen names or Ids may be specified by using a comma-separated list. Id may also be specified using a comma-separated list and will perform the same lookup.

The Min\_Id and Max\_Id pseudo columns may be used to narrow down a range of users to return, or to return only recently created users. Ids are created in increasing numerical order on Twitter. Specifying a Min\_Id returns only results with a greater Id or tweets that were created more recently than the specified one. Setting a Max\_Id returns only tweets that are older than the creation date of the specified Id to return. Note that while these values must be valid, non-negative numbers, they do not have to be Ids that exist.

## Columns

Name	Type	Description
ID [KEY]	<i>String</i>	The Id of the user. A comma-separated list of user Ids may be used in the WHERE clause to get data about multiple users.
SearchTerms	<i>String</i>	A SearchTerms to use while searching users. This can return up to 1000 results.
Name	<i>String</i>	The name of the user.
Screen_Name	<i>String</i>	The screen name of the user.
Location	<i>String</i>	The location of the user.
Profile_URL	<i>String</i>	The URL for the user's profile.
Profile_Image_URL	<i>String</i>	The URL for the image of the user.
Protected	<i>Boolean</i>	The privacy flag of the user. If true, then the user's account is private and only their approved followers can read their tweets or see extended information about them.
Lang	<i>String</i>	The ISO language code of the user.
Created_At	<i>Datetime</i>	When the user account was created.



Friends_Count	<i>Integer</i>	The number of people this user is following.
Followers_Count	<i>Integer</i>	The number of followers the user has.
Favourites_Count	<i>Integer</i>	The number of favorites the user has.
Statuses_Count	<i>Integer</i>	The number of status updates or tweets the user has made.
UTC_Offset	<i>Integer</i>	The Coordinated Universal Time offset for the user in seconds.
Time_Zone	<i>String</i>	The time zone of the user.
Notifications	<i>Boolean</i>	Boolean indicating if the user has notifications enabled.
Geo_Enabled	<i>Boolean</i>	Boolean indicating if the user has geo-enabled turned on in their profile.
Verified	<i>Boolean</i>	Boolean indicating if the user account has been verified.
Following	<i>Boolean</i>	Boolean indicating if the user is following you.
Contributors_Enabled	<i>Boolean</i>	Boolean indicating if contributors are enabled for the account. Typically used in multiuser accounts.
Follow_Request_Sent	<i>Boolean</i>	If the user is a protected user, this column indicates if the authenticated user has sent a request to follow them.

Listed_Count	<i>Integer</i>	The number of public lists a user is listed in. -1 if unknown.
Is_Translator	<i>Boolean</i>	Boolean indicating if the user contributes to translating Twitter in other languages.
Description	<i>String</i>	The description of the user.
Url	<i>String</i>	A URL to the user page on Twitter.
Default_Profile	<i>Boolean</i>	Boolean indicating if the user is using the default profile design/theme.
Default_Profile_Image	<i>Boolean</i>	Boolean indicating if the user is using the default Twitter profile image.
Profile_Background_Color	<i>String</i>	Background color for the user's theme.
Profile_Background_Image_Url	<i>String</i>	HTTP URL for the user's background image in their theme settings.
Profile_Background_Image_Url_Https	<i>String</i>	HTTPS URL for the user's background image in their theme settings.
Profile_Background_Tile	<i>Boolean</i>	Boolean indicating if the user has used the Tile Background checkbox in their theme settings.
Profile_Image_Url_Https	<i>String</i>	HTTPS URL for the user's profile image.
Profile_Link_Color	<i>String</i>	Hexadecimal color code for the user's links.

Profile_Sidebar_Border_Color	<i>String</i>	The sidebar border color for the user.
Profile_Sidebar_Fill_Color	<i>String</i>	The sidebar fill color for the user.
Profile_Text_Color	<i>String</i>	The hexadecimal color code for text in the user's settings.
Profile_Use_Background_Image	<i>String</i>	Boolean indicating if the user is using the background image.
Show_All_Inline_Media	<i>String</i>	Boolean indicating if the user has enabled viewing all in-line media (pictures, videos, etc).

## Pseudo-Columns

Pseudo column fields are used in the WHERE clause of SELECT statements and offer a more granular control over the tuples that are returned from the data source.

<b>Name</b>	<b>Type</b>	<b>Description</b>
NextPageToken	<i>String</i>	An identifier to retrieve the next page of results. Specify only if an error occurs during the original request. Make sure any criteria specified in the original request are still specified when using the NextPageToken.
User_ID	<i>String</i>	The Id of the user to return results for. This can be used in the WHERE clause of an SQL statement as a comma-separated list. Only up to 100 user Ids can be submitted per request.
Min_ID	<i>String</i>	Specifies the lowest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only users that are equal to or newer than the creation date of the specified Id will be returned. The Min_Id must be

		a valid number but does not need to be a valid user Id.
Max_ID	<i>String</i>	Specifies the highest Id to return results for. Ids are stored in increasing numerical order, so specifying this value means that only users that are equal to or older than the creation date of the specified Id will be returned. The Max_Id must be a valid number but does not need to be a valid user Id.

## Stored Procedures

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

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

### Twitter Adapter Stored Procedures

Name	Description
<a href="#">GeoSearch</a>	Run a geo-search against the Twitter service.
<a href="#">GetOAuthAccessToken</a>	Obtains the OAuth access token to be used for authentication with Twitter. If using a Windows application, set AuthMode to APP. If using a Web App, set AuthMode to WEB and specify the AuthToken, AuthKey, and Verifier returned by GetOAuthAuthorizationUrl.
<a href="#">GetOAuthAuthorizationURL</a>	Gets the authorization URL, AuthToken, and AuthKey from Twitter. Navigate to the authorization URL in an Internet browser. This will return a verifier token that you will need to use as input along with the AuthToken for the GetOAuthAccessToken stored procedure.
<a href="#">GetRemainingRequests</a>	Returns the number of remaining requests that can be made to Twitter and how long until the request limit is reset.

<a href="#">ReverseGeoSearch</a>	Reverse-searches a place based on the specified latitude and longitude.
<a href="#">UploadMedia</a>	Uploads an image or video. Only JPG, PNG, GIF, WEBP, MP4 media types are supported.

## GeoSearch

Run a geo-search against the Twitter service.

### Input

Name	Type	Description
Latitude	<i>String</i>	The latitude to search around. This parameter will be ignored unless it is inside the range -90.0 to 90.0 inclusive. (North is positive.) It will also be ignored if there is not a corresponding Longitude input.
Longitude	<i>String</i>	The longitude to search around. The valid ranges for longitude are -180.0 to 180.0 inclusive. (East is positive). This parameter will be ignored if it is outside that range, if it is not a number, if Geo_Enabled is disabled, or if there not a corresponding Latitude input.
IP	<i>String</i>	An IP address. Used when attempting to fix geo-location based off of the user.
Accuracy	<i>String</i>	A hint on the region in which to search. If a number, then this is a radius in meters, but it can also take a string that is suffixed with ft to specify feet. If this is not passed in, then it is assumed to be 0m.
ContainedWithin	<i>String</i>	The place Id that you would like to restrict the search results to. Setting this value means only places within the given place Id will be found.

Granularity	<i>String</i>	The minimal granularity of place types to return. Valid values are poi, neighborhood, city, admin, or country. If no granularity is provided for the request, neighborhood is assumed. Setting this to city, for example, will find places that have a type of city, admin, or country.
MaxResults	<i>String</i>	A hint as to the number of results to return. This does not guarantee that the number of results returned will equal MaxResults, but instead informs how many nearby results to return.
Query	<i>String</i>	Free-form text to match against while executing a geo-based query. This parameter is best suited for finding nearby locations by name. Note that the query must be URL encoded.

## Result Set Columns

<b>Name</b>	<b>Type</b>	<b>Description</b>
Id	<i>String</i>	The place Id.
Name	<i>String</i>	Name of the place.
Country	<i>String</i>	The country name where the coordinates are located.
CountryCode	<i>String</i>	The code for the country.
FullName	<i>String</i>	Full name of the place.
Latitude	<i>String</i>	The latitude of the place.

Longitude	<i>String</i>	The longitude of the place.
PlaceType	<i>String</i>	The type of place (city, neighborhood, etc.)

## GetOAuthAccessToken

Obtains the OAuth access token to be used for authentication with Twitter. If using a Windows application, set AuthMode to APP. If using a Web App, set AuthMode to WEB and specify the AuthToken, AuthKey, and Verifier returned by GetOAuthAuthorizationUrl.

### Input

Name	Type	Description
AuthMode	<i>String</i>	Enter either APP or WEB. The type of authentication mode to use. Set to APP to get authentication tokens via a Windows app (.exe). Set to WEB to get authentication tokens via a Web app.  The default value is <i>APP</i> .
AuthToken	<i>String</i>	The authentication token returned by GetOAuthAuthorizationUrl. Required for only the Web AuthMode.
AuthKey	<i>String</i>	The AuthKey returned by GetOAuthAuthorizationUrl. Required for only the Web AuthMode.
Verifier	<i>String</i>	The verifier token returned by Twitter after using the URL obtained with GetOAuthAuthorizationURL. Required for only the Web AuthMode.
State	<i>String</i>	Any value that you wish to be sent with the callback.

## Result Set Columns

Name	Type	Description
OAuthAccessToken	<i>String</i>	The OAuth access token.
OAuthAccessTokenSecret	<i>String</i>	The OAuth access token secret.
ExpiresIn	<i>String</i>	The remaining lifetime on the access token. A -1 denotes that it will not expire.

## GetOAuthAuthorizationURL

Gets the authorization URL, AuthToken, and AuthKey from Twitter. Navigate to the authorization URL in an Internet browser. This will return a verifier token that you will need to use as input along with the AuthToken for the GetOAuthAccessToken stored procedure.

## Input

Name	Type	Description
CallbackURL	<i>String</i>	The URL that Twitter will return to after the user has authorized your app.
State	<i>String</i>	Any value that you wish to be sent with the callback.

## Result Set Columns

Name	Type	Description
------	------	-------------



URL	<i>String</i>	The URL to be entered into a Web browser to obtain the verifier token and authorize your Twitter app with.
AuthToken	<i>String</i>	A token used as input for the GetOAuthAccessToken stored procedure to verify the request and obtain the OAuth access token.
AuthKey	<i>String</i>	A key used as input for the GetOAuthAccessToken stored procedure to verify the request and obtain the OAuth access token.

## GetRemainingRequests

Returns the number of remaining requests that can be made to Twitter and how long until the request limit is reset.

### Input

Name	Type	Description
Resources	<i>String</i>	A comma-separated list of the resources you want to request rate limit information about. For example: search,users,statuses

### Result Set Columns

Name	Type	Description
Remaining_Requests	<i>String</i>	The number of remaining API requests.
Reset_Time	<i>String</i>	When your available API requests will reset.

Reset_Time_Seconds	<i>String</i>	The time in seconds since 1/1/1970 when your remaining API requests will be reset.
Limit	<i>String</i>	Your limit of total API requests per window. At the moment, each window is 15 minutes.
Api_Request	<i>String</i>	The API request the rate limit information returned goes with.

## ReverseGeoSearch

Reverse-searches a place based on the specified latitude and longitude.

### Input

Name	Type	Description
Accuracy	<i>String</i>	A hint on the region in which to search. If a number, then this is a radius in meters, but it can also take a string that is suffixed with ft to specify feet. If this is not passed in, then it is assumed to be 0m.
Granularity	<i>String</i>	This is the minimal granularity of place types to return. Valid values are poi, neighborhood, city, admin, or country. If no granularity is provided for the request, neighborhood is assumed. Setting this to city, for example, will find places that have a type of city, admin, or country.
Latitude	<i>String</i>	The latitude to search around. This parameter will be ignored unless it is inside the range -90.0 to 90.0 inclusive. (North is positive.) It will also be ignored if there is not a corresponding Longitude input.
Longitude	<i>String</i>	The longitude to search around. The valid ranges for longitude are -180.0 to 180.0 inclusive. (East is positive.) This parameter will be ignored if it is outside that range, if it is not a number, if Geo_Enabled is disabled, or if there not a corresponding

Latitude input.		
MaxResults	<i>String</i>	A hint as to the number of results to return. This does not guarantee that the number of results returned will equal MaxResults, but instead informs how many nearby results to return.

## Result Set Columns

Name	Type	Description
Id	<i>String</i>	The place Id.
Name	<i>String</i>	Name of the place.
Country	<i>String</i>	The country name where the coordinates are located.
CountryCode	<i>String</i>	The code of the country.
FullName	<i>String</i>	Full name of the place.
Latitude	<i>String</i>	The latitude of the place.
Longitude	<i>String</i>	The longitude of the place.
PlaceType	<i>String</i>	The type of place (i.e., city, neighborhood, etc.)

## UploadMedia

Uploads an image or video. Only JPG, PNG, GIF, WEBP, MP4 media types are supported.

## Stored Procedure Specific Info

### Upload Media

Use this procedure to upload images or videos to Twitter. It returns media ids, which can be used to create a Tweet with an attached photo or video. The uploaded media expires in 24 hours after the upload. The video size should not be more than 15 MB and 30 seconds of the length.

Set MediaFilePath for every media file. For example, the set of media files to upload would be the following:

```
MediaFilePath - path/to/file1,path/to/file2
```

The result of the procedure is a list of media ids, which can be retrieved by MediaId for every media file uploaded.

### Input

Name	Type	Accepts Input Streams	Description
MediaFilePath	String	False	A comma-separated list of media file paths to upload. Takes precedence over 'Content' inputs. You can include up to 4 photos or 1 animated GIF or 1 video in a Tweet/DM with INSERT statements.
Content	String	True	The image or video data as an InputStream (JPG, PNG, GIF, WEBP or MP4).
FileName	String	False	The file name including extension to be used with the Content input. FileName is required for Content input.

ContentSize	<i>String</i>	<i>False</i>	The total size in bytes for the Content InputStream. Required when the Content input is used instead of MediaFilePath.
MediaCategory	<i>String</i>	<i>False</i>	A comma-separated list of media categories for each file path to upload or a single media category for a Content input.

## Result Set Columns

Name	Type	Description
MediaId	<i>String</i>	The Ids of the media uploaded.

## 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="#">AccountId</a>	Identifier of the advertising account. This Id is used to get analytic stats for the account that is identified by this Id. The AddAccounts view can be used to list available accounts with their specific Ids.
<a href="#">UseAppOnlyAuthentication</a>	A boolean that indicates whether or not you would like to use app-only authentication.

## OAuth

Property	Description
<a href="#">InitiateOAuth</a>	Set this property to initiate the process to obtain or refresh the OAuth access token when you connect.
<a href="#">OAuthClientId</a>	The client Id assigned when you register your application with an OAuth authorization server.
<a href="#">OAuthClientSecret</a>	The client secret assigned when you register your application with an OAuth authorization server.
<a href="#">OAuthAccessToken</a>	The access token for connecting using OAuth.
<a href="#">OAuthAccessTokenSecret</a>	The OAuth access token secret for connecting using OAuth.
<a href="#">OAuthSettingsLocation</a>	The location of the settings file where OAuth values are saved when InitiateOAuth is set to GETANDREFRESH or REFRESH. Alternatively, this can be held in memory by specifying a value starting with memory://.
<a href="#">CallbackURL</a>	The OAuth callback URL to return to when authenticating. This value must match the callback URL you specify in your app settings.
<a href="#">OAuthVerifier</a>	The verifier code returned from the OAuth authorization URL.
<a href="#">AuthToken</a>	The authentication token used to request and obtain the OAuth Access Token.
<a href="#">AuthKey</a>	The authentication key used to request and obtain the OAuth Access Token.
<a href="#">OAuthExpiresIn</a>	The lifetime in seconds of the OAuth AccessToken.
<a href="#">OAuthTokenTimestamp</a>	The Unix epoch timestamp in milliseconds when the current Access Token was created.

## SSL

Property	Description
<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="#">IsPremiumTwitter</a>	When true, enables access to the Premium Search API. This is false by default.



<a href="#">MaxRateLimitDelay</a>	The maximum amount of time to delay (in seconds) before submitting a request if it would be rate limited.
<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="#">Readonly</a>	You can use this property to enforce read-only access to Twitter from the provider.
<a href="#">SearchTerms</a>	Default SearchTerms if none are specified. Used for some tables such as Tweets where SearchTerms may be specified.
<a href="#">StreamPageSize</a>	The number of results to return per page of data retrieved from the Twitter stream.
<a href="#">StreamReadDuration</a>	This property represents the maximum time to read streaming data, in seconds.
<a href="#">StreamTimeout</a>	The maximum number of seconds to continue waiting for results while streaming. When this value is reached and no tweets are returned, then the connection will be closed.
<a href="#">Timeout</a>	The value in seconds until the timeout error is thrown, canceling the operation.
<a href="#">UserDefinedViews</a>	A filepath pointing to the JSON configuration file containing your custom views.

## 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="#">AccountId</a>	Identifier of the advertising account. This Id is used to get analytic stats for the account that is identified by this Id. The AddAccounts view can be used to list available accounts with their specific Ids.
<a href="#">UseAppOnlyAuthentication</a>	A boolean that indicates whether or not you would like to use app-only authentication.

---

## AccountId

Identifier of the advertising account. This Id is used to get analytic stats for the account that is identified by this Id. The AddAccounts view can be used to list available accounts with their specific Ids.

### Data Type

string

### Default Value

""

### Remarks

Identifier of the advertising account. This Id is used to get analytic stats for the account that is identified by this Id. The AddAccounts view can be used to list available accounts with their specific Ids.

## UseAppOnlyAuthentication

A boolean that indicates whether or not you would like to use app-only authentication.

### Data Type

bool

## Default Value

false

## Remarks

Set this to true to have your Twitter app log in to Twitter instead of a user.

## OAuth

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

Property	Description
<a href="#">InitiateOAuth</a>	Set this property to initiate the process to obtain or refresh the OAuth access token when you connect.
<a href="#">OAuthClientId</a>	The client Id assigned when you register your application with an OAuth authorization server.
<a href="#">OAuthClientSecret</a>	The client secret assigned when you register your application with an OAuth authorization server.
<a href="#">OAuthAccessToken</a>	The access token for connecting using OAuth.
<a href="#">OAuthAccessTokenSecret</a>	The OAuth access token secret for connecting using OAuth.
<a href="#">OAuthSettingsLocation</a>	The location of the settings file where OAuth values are saved when InitiateOAuth is set to GETANDREFRESH or REFRESH. Alternatively, this can be held in memory by specifying a value starting with memory://.
<a href="#">CallbackURL</a>	The OAuth callback URL to return to when authenticating. This value must match the callback URL you specify in your app settings.
<a href="#">OAuthVerifier</a>	The verifier code returned from the OAuth authorization URL.

<code>AuthToken</code>	The authentication token used to request and obtain the OAuth Access Token.
<code>AuthKey</code>	The authentication key used to request and obtain the OAuth Access Token.
<code>OAuthExpiresIn</code>	The lifetime in seconds of the OAuth AccessToken.
<code>OAuthTokenTimestamp</code>	The Unix epoch timestamp in milliseconds when the current Access Token was created.

## InitiateOAuth

Set this property to initiate the process to obtain or refresh the OAuth access token when you connect.

### Possible Values

OFF, GETANDREFRESH, REFRESH

### Data Type

string

### Default Value

"OFF"

### Remarks

The following options are available:

1. **OFF:** Indicates that the OAuth flow will be handled entirely by the user. An OAuthAccessToken will be required to authenticate.
2. **GETANDREFRESH:** Indicates that the entire OAuth Flow will be handled by the adapter. If no token currently exists, it will be obtained by prompting the user via the browser. If a token exists, it will be refreshed when applicable.
3. **REFRESH:** Indicates that the adapter will only handle refreshing the

OAuthAccessToken. The user will never be prompted by the adapter to authenticate via the browser. The user must handle obtaining the OAuthAccessToken and OAuthRefreshToken initially.

## OAuthClientId

The client Id assigned when you register your application with an OAuth authorization server.

### Data Type

string

### Default Value

""

### Remarks

As part of registering an OAuth application, you will receive the OAuthClientId value, sometimes also called a consumer key, and a client secret, the [OAuthClientSecret](#).

## OAuthClientSecret

The client secret assigned when you register your application with an OAuth authorization server.

### Data Type

string

### Default Value

""

## Remarks

As part of registering an OAuth application, you will receive the [OAuthClientId](#), also called a consumer key. You will also receive a client secret, also called a consumer secret. Set the client secret in the [OAuthClientSecret](#) property.

## OAuthAccessToken

The access token for connecting using OAuth.

### Data Type

string

### Default Value

""

## Remarks

The [OAuthAccessToken](#) property is used to connect using OAuth. The [OAuthAccessToken](#) is retrieved from the OAuth server as part of the authentication process. It has a server-dependent timeout and can be reused between requests.

The access token is used in place of your user name and password. The access token protects your credentials by keeping them on the server.

## OAuthAccessTokenSecret

The OAuth access token secret for connecting using OAuth.

### Data Type

string

### Default Value

""

## Remarks

The `OAuthAccessTokenSecret` property is used to connect and authenticate using OAuth. The `OAuthAccessTokenSecret` is retrieved from the OAuth server as part of the authentication process. It is used with the `OAuthAccessToken` and can be used for multiple requests until it times out.

## OAuthSettingsLocation

The location of the settings file where OAuth values are saved when `InitiateOAuth` is set to `GETANDREFRESH` or `REFRESH`. Alternatively, this can be held in memory by specifying a value starting with `memory://`.

## Data Type

string

## Default Value

"%APPDATA%\\CData\\Twitter Data Provider\\OAuthSettings.txt"

## Remarks

When `InitiateOAuth` is set to `GETANDREFRESH` or `REFRESH`, the adapter saves OAuth values to avoid requiring the user to manually enter OAuth connection properties and allowing the credentials to be shared across connections or processes.

Alternatively to specifying a file path, memory storage can be used instead. Memory locations are specified by using a value starting with `'memory://'` followed by a unique identifier for that set of credentials (ex: `memory://user1`). The identifier can be anything you choose but should be unique to the user. Unlike with the file based storage, you must manually store the credentials when closing the connection with memory storage to be able to set them in the connection when the process is started again. The OAuth property values can be retrieved with a query to the `sys_connection_props` system table. If there are multiple connections using the same credentials, the properties should be read from the last connection to be closed.

If left unspecified, the default location is "%APPDATA%\\CData\\Twitter Data Provider\\OAuthSettings.txt" 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

## CallbackURL

The OAuth callback URL to return to when authenticating. This value must match the callback URL you specify in your app settings.

### Data Type

string

### Default Value

""

### Remarks

During the authentication process, the OAuth authorization server redirects the user to this URL. This value must match the callback URL you specify in your app settings.

## OAuthVerifier

The verifier code returned from the OAuth authorization URL.

### Data Type

string

### Default Value

""



## Remarks

The verifier code returned from the OAuth authorization URL. This can be used on systems where a browser cannot be launched such as headless systems.

## Authentication on Headless Machines

See to obtain the OAuthVerifier value.

Set [OAuthSettingsLocation](#) along with OAuthVerifier. When you connect, the adapter exchanges the OAuthVerifier for the OAuth authentication tokens and saves them, encrypted, to the specified file. Set [InitiateOAuth](#) to GETANDREFRESH automate the exchange.

Once the OAuth settings file has been generated, you can remove OAuthVerifier from the connection properties and connect with [OAuthSettingsLocation](#) set.

To automatically refresh the OAuth token values, set [OAuthSettingsLocation](#) and additionally set [InitiateOAuth](#) to REFRESH.

## AuthToken

The authentication token used to request and obtain the OAuth Access Token.

## Data Type

string

## Default Value

""

## Remarks

This property is required only when performing headless authentication in OAuth 1.0. It can be obtained from the GetOAuthAuthorizationUrl stored procedure.

It can be supplied alongside the [AuthKey](#) in the GetOAuthAccessToken stored procedure to obtain the [OAuthAccessToken](#).

## AuthKey

The authentication key used to request and obtain the OAuth Access Token.

### Data Type

string

### Default Value

""

### Remarks

This property is required only when performing headless authentication in OAuth 1.0. It can be obtained from the GetOAuthAuthorizationUrl stored procedure.

It can be supplied alongside the [AuthToken](#) in the GetOAuthAccessToken stored procedure to obtain the [OAuthAccessToken](#).

## OAuthExpiresIn

The lifetime in seconds of the OAuth AccessToken.

### Data Type

string

### Default Value

""

### Remarks

Pair with OAuthTokenTimestamp to determine when the AccessToken will expire.

## OAuthTokenTimestamp

The Unix epoch timestamp in milliseconds when the current Access Token was created.

## Data Type

string

## Default Value

""

## Remarks

Pair with OAuthExpiresIn to determine when the AccessToken will expire.

# 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="#">SSLServerCert</a>	The certificate to be accepted from the server when connecting using TLS/SSL.

---

## 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	Description
------	---------	-------------

---

	Port	
TUNNEL	80	When this is set, the adapter opens a connection to Twitter 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\\Twitter 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\\Twitter 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="#">IsPremiumTwitter</a>	When true, enables access to the Premium Search API. This is false by default.
<a href="#">MaxRateLimitDelay</a>	The maximum amount of time to delay (in seconds) before submitting a request if it would be rate limited.



<b>MaxRows</b>	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.
<b>Other</b>	These hidden properties are used only in specific use cases.
<b>ReadOnly</b>	You can use this property to enforce read-only access to Twitter from the provider.
<b>SearchTerms</b>	Default SearchTerms if none are specified. Used for some tables such as Tweets where SearchTerms may be specified.
<b>StreamPageSize</b>	The number of results to return per page of data retrieved from the Twitter stream.
<b>StreamReadDuration</b>	This property represents the maximum time to read streaming data, in seconds.
<b>StreamTimeout</b>	The maximum number of seconds to continue waiting for results while streaming. When this value is reached and no tweets are returned, then the connection will be closed.
<b>Timeout</b>	The value in seconds until the timeout error is thrown, canceling the operation.
<b>UserDefinedViews</b>	A filepath pointing to the JSON configuration file containing your custom views.

## IsPremiumTwitter

When true, enables access to the Premium Search API. This is false by default.

### Data Type

bool

### Default Value

false

## Remarks

With this property enabled, you can return tweets from searches older than 7 days ago.

The EnvType and DevEnvironment pseudo columns may be used to query Premium Search APIs. EnvType specifies the environment type. It can take one of 30day or fullarchive values.

The default value is 30day. The DevEnvironment is required in order to query the Premium Search API and must be set to the value of the dev environment label, created in your Twitter Developer Account.

## MaxRateLimitDelay

The maximum amount of time to delay (in seconds) before submitting a request if it would be rate limited.

## Data Type

string

## Default Value

"60"

## Remarks

Twitter uses different rate limits for total number of requests for different endpoints. These can range from as few as 15 per 15 minute window, up to 900 for a 15 minute window. Internally the Twitter Adapter keeps track of if a given request would result in a rate limit exception. If a rate limit would occur, the Twitter Adapter can internally delay submitting a request until the limit is up. However, this could also result in waiting for several minutes before requesting data, which is also not a good behavior.

The MaxRateLimitDelay gives control over the maximum amount of time the Twitter Adapter will wait once it detects a rate limit would occur. Since the amount of time the Twitter Adapter needs to wait can be calculated, if it would have to wait longer than the MaxRateLimitDelay, it will simply error immediately when it sees the time would take too long.

## 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.

## Readonly

You can use this property to enforce read-only access to Twitter 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.

## SearchTerms

Default SearchTerms if none are specified. Used for some tables such as Tweets where SearchTerms may be specified.

### Data Type

string

## Default Value

""

## Remarks

Default SearchTerms if none are specified. Used for some tables such as Tweets where SearchTerms may be specified.

## StreamPageSize

The number of results to return per page of data retrieved from the Twitter stream.

## Data Type

string

## Default Value

"50"

## Remarks

The number of results to return per page of data retrieved from the Twitter stream.

## StreamReadDuration

This property represents the maximum time to read streaming data, in seconds.

## Data Type

int

## Default Value

0

## Remarks

Set `StreamReadDuration` property to consume streaming data for a specific time period. When `StreamReadDuration` is reached and `StreamTimeout` is 0, then the connection will be closed.

## StreamTimeout

The maximum number of seconds to continue waiting for results while streaming. When this value is reached and no tweets are returned, then the connection will be closed.

### Data Type

string

### Default Value

"0"

## Remarks

Set the value of `StreamTimeout` to 0 in order to keep the connection open indefinitely. Note that, if the value of this property is greater than zero, the value of the [StreamPageSize](#) property, will be overwritten and will be set to one(1).

## 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.

## 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 Tweets 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"
```



# 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.