



TIBCO® Data Virtualization

Google Calendar Adapter Guide

Version 8.7.0 | October 2023

Contents

Contents	2
Google Calendar Adapter	5
Google Calendar Version Support	5
SQL Compliance	5
Getting Started	5
Connecting to Google Calendar	5
Deploying the Google Calendar Adapter	5
Basic Tab	6
Logging	7
Using OAuth Authentication	9
Advanced Settings	17
SQL Compliance	19
SELECT Statements	19
INSERT Statements	19
UPDATE Statements	19
DELETE Statements	20
EXECUTE Statements	20
Names and Quoting	20
Transactions and Batching	20
SELECT Statements	20
SELECT INTO Statements	22
INSERT Statements	23
UPDATE Statements	24
DELETE Statements	24
EXECUTE Statements	25
Data Model	26
Tables	26

Views	26
Stored Procedures	27
Tables	27
Views	44
Stored Procedures	50
Connection String Options	56
Auth Scheme	59
Firewall Password	59
Firewall Port	60
Firewall Server	60
Firewall Type	61
Firewall User	62
Initiate OAuth	63
Location	63
Log Modules	64
Max Rows	65
OAuth Access Token	65
OAuth Client Id	66
OAuth Client Secret	66
OAuth Expires In	67
OAuth JWT Cert	67
OAuth JWT Cert Password	68
OAuth JWT Cert Subject	69
OAuth JWT Cert Type	70
OAuth JWT Issuer	72
OAuth JWT Subject	72
OAuth Refresh Token	73
OAuth Settings Location	73
OAuth Token Timestamp	75
OAuth Verifier	75
Other	76
Proxy Auth Scheme	77

Proxy Auto Detect	78
Proxy Exceptions	79
Proxy Password	79
Proxy Port	80
Proxy Server	80
Proxy SSL Type	81
Proxy User	82
Readonly	83
SSL Server Cert	83
Timeout	84
TIBCO Product Documentation and Support Services	86
How to Access TIBCO Documentation	86
Release Version Support	87
How to Contact TIBCO Support	87
How to Join TIBCO Community	88
Legal and Third-Party Notices	89

Google Calendar Adapter

Google Calendar Version Support

The adapter defaults to version 3 of the Google Calendar API.

SQL Compliance

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

Getting Started

Connecting to Google Calendar

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

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

1. Unzip the `tdv.googlecalendar.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.googlecalendar/tdv.googlecalendar.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 GoogleCalendar
```

Basic Tab

Authenticate via OAuth Authentication

Use the OAuth authentication standard to connect to Google Calendar. You can authenticate with a user account or with a service account. A service account is required to grant organization-wide access scopes to the adapter. The adapter facilitates these authentication flows as described below.

Authenticate with a User Account

Authenticate with a Service Account

Service accounts have silent authentication, without user authentication in the browser. You can also use a service account to delegate enterprise-wide access scopes to the adapter.

You need to create an OAuth application in this flow. See [Creating a Custom OAuth App](#) to create and authorize an app. You can then connect to Google Calendar data that the service account has permission to access.

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

- InitiateOAuth: Set this to GETANDREFRESH.
- OAuthJWTCertType: Set this to "PFXFILE".

- OAuthJWTCert: Set this to the path to the .p12 file you generated.
- OAuthJWTCertPassword: Set this to the password of the .p12 file.
- OAuthJWTCertSubject: Set this to "*" to pick the first certificate in the certificate store.
- OAuthJWTIssuer: In the service accounts section, click Manage Service Accounts and set this field to the email address displayed in the service account Id field.
- OAuthJWTSubject: Set this to your enterprise Id if your subject type is set to "enterprise" or your app user Id if your subject type is set to "user".

When you connect the adapter completes the OAuth flow for a service account.

1. Creates and signs the JWT with the claim set required by the adapter.
2. Exchanges the JWT for the access token.
3. Saves OAuth values in OAuthSettingsLocation to be persisted across connections.
4. Submits the JWT for a new access token when the token expires.

Logging

The adapter uses log4j to generate log files. The settings within the 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 a breakdown 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 Google Calendar 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_cdata.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

Use the OAuth authentication standard to connect to Google Calendar. You can authenticate with a user account or a service account. The adapter facilitates this as described below.

Using a User Account to Authenticate to Google Calendar

The user account flow requires the authenticating user to interact with Google Calendar via the browser.

Embedded Credentials

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

Custom Credentials

Instead of connecting with the adapter's embedded credentials, you can register an app to obtain the [OAuthClientId](#) and [OAuthClientSecret](#).

When to Create a Custom OAuth App

Creating a custom OAuth app is optional as the adapter is already registered with Google Calendar 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 Google Calendar OAuth endpoint to grant permissions to the adapter.

Using a Service Account to Connect to Google Calendar

Service accounts have silent authentication, without user authentication in the browser. You can also use a service account to delegate enterprise-wide access scopes to the adapter.

You need to create an OAuth application in this flow. You can then connect to Google Calendar data that the service account has permission to access. See [Custom Credentials](#) for an authentication guide.

Creating a Custom OAuth App

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

Embedded Credentials

Authenticate using the Embedded OAuth Credentials

Desktop Authentication with the Embedded OAuth App

You can connect without setting any connection properties for your user credentials. After setting `InitiateOAuth` to `GETANDREFRESH`, you are ready to connect. 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

You can use a custom OAuth app to authenticate a service account or a user account. See [Using OAuth Authentication](#) for more information.

Authenticate with a User Account

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](#).

Get and Refresh the OAuth Access Token

After setting the following, you are ready to connect:

- 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.
- 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. Refreshes the access token when it expires.
3. Saves OAuth values in OAuthSettingsLocation to be persisted across connections.

Authenticate with a Service Account

Service accounts have silent authentication, without user authentication in the browser. You can also use a service account to delegate enterprise-wide access scopes to the adapter.

You need to create an OAuth application in this flow. See [Creating a Custom OAuth App](#) to create and authorize an app. You can then connect to Google Calendar data that the service account has permission to access.

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

- InitiateOAuth: Set this to GETANDREFRESH.
- OAuthJWTCertType: Set this to "PFXFILE".
- OAuthJWTCert: Set this to the path to the .p12 file you generated.

- OAuthJWTCertPassword: Set this to the password of the .p12 file.
- OAuthJWTCertSubject: Set this to "*" to pick the first certificate in the certificate store.
- OAuthJWTIssuer: In the service accounts section, click Manage Service Accounts and set this field to the email address displayed in the service account Id field.
- OAuthJWTSubject: Set this to your enterprise Id if your subject type is set to "enterprise" or your app user Id if your subject type is set to "user".

When you connect the adapter completes the OAuth flow for a service account.

1. Creates and signs the JWT with the claim set required by the adapter.
2. Exchanges the JWT for the access token.
3. Saves OAuth values in OAuthSettingsLocation to be persisted across connections.
4. Submits the JWT for a new access token when the token expires.

Headless Machines

Using OAuth on a Headless Machine

The following sections show how to authenticate a headless server or another machine on which the adapter cannot open a browser. You can authenticate with a user account or with a service account.

Authenticate with a User Account

To authenticate with a user account, 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. Click the following link to open the [Google Calendar OAuth endpoint](#) in your browser.
2. 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.

- 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 Google Calendar 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:

- 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:

- 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.

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.
- 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.

Authenticate with a Service Account

Service accounts have silent authentication, without user authentication in the browser. You can also use a service account to delegate enterprise-wide access scopes to the adapter.

You need to create an OAuth application in this flow. See [Creating a Custom OAuth App](#) to create and authorize an app. You can then connect to Google Calendar data that the service account has permission to access.

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

- InitiateOAuth: Set this to GETANDREFRESH.
- OAuthJWTCertType: Set this to "PFXFILE".
- OAuthJWTCert: Set this to the path to the .p12 file you generated.
- OAuthJWTCertPassword: Set this to the password of the .p12 file.
- OAuthJWTCertSubject: Set this to "*" to pick the first certificate in the certificate store.
- OAuthJWTIssuer: In the service accounts section, click Manage Service Accounts and set this field to the email address displayed in the service account Id field.
- OAuthJWTSubject: Set this to your enterprise Id if your subject type is set to "enterprise" or your app user Id if your subject type is set to "user".

When you connect the adapter completes the OAuth flow for a service account.

1. Creates and signs the JWT with the claim set required by the adapter.
2. Exchanges the JWT for the access token.
3. Saves OAuth values in OAuthSettingsLocation to be persisted across connections.
4. Submits the JWT for a new access token when the token expires.

Creating a Custom OAuth App

You can use a custom OAuth app to authenticate a service account or a user account. See [Using OAuth Authentication](#) for more information.

Create an OAuth App for User Account Authentication

Follow the procedure below to register an app and obtain the OAuthClientId and OAuthClientSecret.

Create a Custom OAuth App: Desktop

1. Log into the Google API Console.
2. Click Create Project or select an existing project.
3. In the API Manager, click Credentials -> Create Credentials -> OAuth Client Id -> Other.
4. Click Create. The OAuthClientId and OAuthClientSecret are displayed.
5. Click Library -> Google Calendar API -> Enable API.

Create an OAuth App for Service Account Authentication

Follow the steps below to create an OAuth application and generate a private key. You will then authorize the service account.

1. Log into the Google API Console and open a project. Select the API Manager from the main menu.
2. Click Create Credentials -> Service Account Key.
3. In the Service Account menu, select New Service Account or select an existing service account.
4. If you are creating a new service account, additionally select one or more roles. You can assign primitive roles at the project level in the IAM and Admin section; other roles enable you to further customize access to Google APIs.
5. In the Key Type section, select the P12 key type.
6. Create the app to download the key pair. The private key's password is displayed: Set this in OAuthJWTCertPassword.
7. In the service accounts section, click Manage Service Accounts and set OAuthJWTIssuer to the email address displayed in the service account Id field.
8. Click Library -> Google Calendar API -> Enable API.

Advanced Settings

The following sections detail adapter settings that may be needed in advanced integrations.

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.

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.

Troubleshooting the Connection

To show adapter activity from query execution to network traffic, use Logfile and Verbosity. The examples of common connection errors below show how to use these properties to get more context. Contact the support team for help tracing the source of an error or circumventing a performance issue.

- **Authentication errors:** Typically, recording a Logfile at Verbosity 4 is necessary to get full details on an authentication error.
- **Queries time out:** A server that takes too long to respond will exceed the adapter's

client-side timeout. Often, setting the [Timeout](#) property to a higher value will avoid a connection error. Another option is to disable the timeout by setting the property to 0. Setting [Verbosity](#) to 2 will show where the time is being spent.

- **The certificate presented by the server cannot be validated:** This error indicates that the adapter cannot validate the server's certificate through the chain of trust. If you are using a self-signed certificate, there is only one certificate in the chain.

To resolve this error, you must verify yourself that the certificate can be trusted and specify to the adapter that you trust the certificate. One way you can specify that you trust a certificate is to add the certificate to the trusted system store; another is to set [SSLServerCert](#).

SQL Compliance

The Google Calendar 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 Google Calendar API.

INSERT Statements

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

UPDATE Statements

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

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').

Transactions and Batching

Transactions are not currently supported.

Additionally, the adapter does not support batching of SQL statements. To execute multiple commands, you can create multiple instances and execute each separately.

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 Google Calendar 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> ]
  [
    LIMIT <expression>
  ]
} | 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 MyCalendar
```

2. Rename a column:

```
SELECT "Description" AS MY_Description FROM MyCalendar
```

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

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

4. Search data:

```
SELECT * FROM MyCalendar WHERE Status = 'confirmed';
```

5. The Google Calendar APIs support the following operators in the WHERE clause:
=,AND,>,<,<=,>=.

```
SELECT * FROM MyCalendar WHERE Status = 'confirmed';
```

SELECT INTO Statements

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

Data Export with an SQL Query

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

```
boolean ret = stat.execute("SELECT Id, Description INTO
'csv://c:/MyCalendar.txt' FROM 'MyCalendar' WHERE Status =
'confirmed'");
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 'MyCalendar' IN
'csv://filename=c:/MyCalendar.csv;delimiter=tab' FROM 'MyCalendar' WHERE
Status = 'confirmed'");
System.out.println(stat.getUpdateCount()+" rows affected");
```

INSERT Statements

To create new records, use INSERT statements.

INSERT Syntax

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

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

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

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

```
String cmd = "INSERT INTO MyCalendar (Description) VALUES (?)";
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "My Calendar 2");
int count = pstmt.executeUpdate();
System.out.println(count+" rows were affected");
connection.close();
```

UPDATE Statements

To modify existing records, use UPDATE statements.

Update Syntax

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

```
UPDATE <table_name> SET { <column_reference> = <expression> } [ , ... ]
WHERE { Id = <expression> } [ { AND | OR } ... ]
<expression> ::=
    | @ <parameter>
    | ?
    | <literal>
```

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

```
String cmd = "UPDATE MyCalendar SET Description='My Calendar 2' WHERE Id = ?";
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "S");
int count = pstmt.executeUpdate();
System.out.println(count + " rows were affected");
connection.close();
```

DELETE Statements

To delete information from a table, use DELETE statements.

DELETE Syntax

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

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

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

```
Connection connection = DriverManager.getConnection
("jdbc:googlecalendar:InitiateOAuth=GETANDREFRESH;",);
String cmd = "DELETE FROM MyCalendar WHERE Id = ?";
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "S");
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;
```

Data Model

The Google Calendar Adapter models Google Calendar APIs as 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

The adapter dynamically retrieves table definitions for the calendars accessible to your account. [Tables](#) describes the columns for a sample calendar as well as the [Calendars](#) table, which can be used to list, create, modify, and delete calendars.

Views

[Views](#) offer additional metadata information from Google Calendar to surface other aspects of a calendar.

Stored Procedures

[Stored Procedures](#) are function-like interfaces to the data source.

Tables

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

Generally, querying Google Calendar 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.

Google Calendar Adapter Tables

Name	Description
AllCalendars	Create, update, delete, and query all calendar events in your Google Account.
Calendars	Create, update, delete, and query calendars in Google.
MyCalendar	Create, update, delete, and query events of the calendar.

AllCalendars

Create, update, delete, and query all calendar events in your Google Account.

Table-Specific Information

This is an example on how all calendar events in your account are exposed in a single table.

Select

Query events of all calendars.

```
Select * FROM [AllCalendars]
```

Insert

Create a new event in a certain calendar. At least StartDateTime, EndDateTime and CalendarId must be specified.

```
INSERT INTO [AllCalendars] (Summary, Description, StartDateTime,
EndDateTime, CalendarId) VALUES ('Great Event', 'Description for event',
'8/27/2017', '8/28/2017', 'calendarid@gmail.com')
```

Update

Update details of a specific event. At least the Id of the event being updated must be specified.

```
UPDATE AllCalendars SET Summary='Test Event' WHERE
id='6bjelf33p0al4d8ei5ft5ghqjs' AND CalendarId='clanedarId@cdata.com'
```

Delete

Delete an event from AllCalendars by specifying at least its Id.

```
Delete FROM [AllCalendars] WHERE Id='8ba774m3anenroqcephi7ka6ok' AND
CalendarId='clanedarId@cdata.com'
```

Order Events

When you query from AllCalendars table, the events will not be ordered by the StartDate, but rather by the CalendarId. You can order the calendars by either the StartDate or StartDateTime column, depending if the event is an AllDayEvent or not. Alternatively, you can order both Event types using the example query below.

```

SELECT CalendarId, Id, Summary,
CASE
    WHEN StartDateTime IS NULL THEN startDate
    ELSE StartDateTime
END AS EventDate
FROM AllCalendars ORDER BY EventDate ASC

```

Columns

Name	Type	ReadOnly	Description
Id [KEY]	<i>String</i>	True	The Id of the event.
CalendarId [KEY]	<i>String</i>	True	Calendar Id
Summary	<i>String</i>	False	The title of the event.
Description	<i>String</i>	False	The description of the event.
Location	<i>String</i>	False	The geographic location of the event as free-form text.
AllDayEvent	<i>Boolean</i>	False	This value sets whether or not the event is an all-day event.
StartDate	<i>Date</i>	False	The start date if this is an all-day event..
StartDateTime	<i>Datetime</i>	False	The (inclusive) start time of the event. For a recurring event, this value is the start

			time of the first instance.
StartDateTimeZone	<i>String</i>	False	The time zone in which the start date time is specified.
EndDate	<i>Datet</i>	False	The end date if this is an all-day event.
EndTime	<i>Datetime</i>	False	The (exclusive) end time of the event. For a recurring event, this value is the end time of the first instance.
EndTimeZone	<i>String</i>	False	The time zone in which the end date time is specified.
OriginalStartTimeDateTime	<i>Datetime</i>	False	For an instance of a recurring event, this value is the time when the event would start according to the recurrence data in the recurring event identified by RecurringEventId.
SendNotification	<i>Boolean</i>	False	This value sets whether to send a notification when performing an insert or update.
Kind	<i>String</i>	True	The type of the resource, returned in the format calendar#event.
ETag	<i>String</i>	True	The ETag of the resource.
Status	<i>String</i>	False	The status of the event.
HTMLLink	<i>String</i>	True	The absolute link to the

			event in the Google Calendar Web UI.
Created	<i>Datetime</i>	True	The creation time of the event.
Updated	<i>Datetime</i>	True	The latest modification time of the event.
ColorId	<i>Integer</i>	False	The color of the event. This value is an Id referring to an entry in the event section of the colors definitions.
CreatorEmail	<i>String</i>	True	The creator's email address, if available.
CreatorDisplayName	<i>String</i>	True	The creator's name, if available.
OrganizerEmail	<i>String</i>	False	The organizer's email address, if available.
OrganizerDisplayName	<i>String</i>	False	The organizer's name, if available.
Recurrences	<i>String</i>	False	A pipe-separated list of RRULE, EXRULE, RDATE, and EXDATE lines for a recurring event. This field is omitted for single events or instances of recurring events. OriginalStartTimeDateTime must be set in order to modify this value.
RecurringEventId	<i>String</i>	True	For an instance of a recurring event, this value is the event Id of the recurring

			event itself.
Transparency	<i>String</i>	False	This value sets whether the event blocks time on the calendar. If set to transparent, the event does not block time on the calendar. If set to opaque, the event blocks time; this is the default value.
Visibility	<i>String</i>	False	The visibility of the event.
ICalUid	<i>String</i>	True	The event Id in the iCalendar format.
Sequence	<i>String</i>	False	The sequence number as per iCalendar.
AttendeesEmails	<i>String</i>	False	A comma-separated list of attendee's email addresses, if available.
AttendeesDisplayNames	<i>String</i>	False	A comma-separated list of attendee's names, if available.
AttendeesOmitted	<i>Boolean</i>	True	This field sets whether attendees have been omitted from the event's representation. When updating an event, this field can be used to update only the participant's response. When retrieving an event, the attendees that are returned are restricted to only the participant by the MaxAttendees query

			parameter.
ExtendedPropertiesPrivateKey	<i>String</i>	False	This field contains properties that are private to the copy of the event that appears on the calendar.
ExtendedPropertiesPrivateValue	<i>String</i>	False	This field contains properties that are private to the copy of the event that appears on the calendar.
ExtendedPropertiesSharedKey	<i>String</i>	False	This field contains properties that are shared between copies of the event on other attendees' calendars.
ExtendedPropertiesSharedValue	<i>String</i>	False	This field contains properties that are shared between copies of the event on other attendees' calendars.
AnyoneCanAddSelf	<i>Boolean</i>	True	This value sets whether anyone can invite themselves to the event.
GuestsCanInviteOthers	<i>Boolean</i>	False	This value sets whether attendees other than the organizer can invite others to the event.
GuestsCanSeeOtherGuests	<i>Boolean</i>	False	This value sets whether attendees other than the organizer can see who the event's attendees are.
GuestsCanModify	<i>Boolean</i>	False	Whether attendees other than the organizer can

			modify the event.
PrivateCopy	<i>Boolean</i>	True	This value sets whether this is a private event copy where changes are not shared with other copies on other calendars.
RemindersUseDefault	<i>Boolean</i>	False	This value sets whether the default reminders of the calendar apply to the event.
ReminderOverrideMethods	<i>String</i>	False	A comma-separated list of the methods used by the reminder. The possible values are EMAIL, SMS, and POPUP.
ReminderOverrideMinutes	<i>String</i>	False	A comma-separated list of the minutes before the start of the event when the corresponding ReminderOverrideMethod should trigger.

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
SearchTerms	<i>String</i>	Free text search terms to find events that match these terms in any field, except for extended properties.
SingleEvents	<i>Boolean</i>	Whether to expand recurring events into instances, default value is false.

Calendars

Create, update, delete, and query calendars in Google.

Table-Specific Information

Select

When selecting calendars no fields are required. In addition an Id can be specified for filtering the result. For example:

```
SELECT Id, Summary, Description FROM Calendars
```

Insert

To insert a calendar, issue an INSERT statement and specify a value for at least the Summary column. For example:

```
INSERT INTO Calendars (Summary) VALUES ('My Custom Calendar')
```

Update

To update a calendar, the Id column must be specified. Only the Description, Location, Summary, and Timezone columns are updateable. For example:

```
UPDATE Calendars SET Description='Updated Description' WHERE  
Id='8ba774m3anenroqcepf7ka6ok'
```

Delete

Delete a calendar by specifying its Id. For example:

```
DELETE FROM Calendars WHERE Id='123456789'
```

Columns

Name	Type	ReadOnly	Description
Id [KEY]	<i>String</i>	True	The calendar identifier.
Summary	<i>String</i>	False	The title of the calendar.
Description	<i>String</i>	False	The description of the calendar.
Kind	<i>String</i>	True	The type of the resource.
ETag	<i>String</i>	True	The ETag of the resource.
Location	<i>String</i>	False	The geographic location of the calendar as free-form text.
Timezone	<i>String</i>	False	The time zone of the calendar.
SummaryOverride	<i>String</i>	False	The summary that the authenticated user has set for the calendar.
ColorId	<i>Integer</i>	False	The color of the calendar. This is an Id referring to an entry in the 'calendar' section of the colors definition.
Hidden	<i>Boolean</i>	False	This field sets whether the calendar has been hidden from the list.

Selected	<i>Boolean</i>	False	This field sets whether the calendar content shows up in the calendar UI.
AccessRole	<i>String</i>	True	The effective access role that the authenticated user has on the calendar.
ReminderMethods	<i>String</i>	False	A semicolon-separated list of the methods used by the reminder. Possible values are: EMAIL, SMS, and POPUP.
ReminderMinutes	<i>String</i>	False	A semicolon-separated list of minutes before the start of the event when the corresponding ReminderOverrideMethod should trigger.

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
ShowHidden	<i>String</i>	<p>This value sets whether to show hidden calendars.</p> <p>The allowed values are <i>TRUE</i>, <i>FALSE</i>.</p> <p>The default value is <i>False</i>.</p>

MyCalendar

Create, update, delete, and query events of the calendar.

Table-Specific Information

This is an example on how a calendar is exposed as table.

Select

Query events of the specified calendar.

```
Select * FROM [MyCalendar]
```

Insert

Create a new event in the calendar. At least StartDateTime and EndDateTime must be specified.

```
INSERT INTO [MyCalendar] (Summary, Description, StartDateTime,
EndDateTime) VALUES ('Great Event', 'Description for event',
'8/27/2017', '8/28/2017')
```

When inserting a new event, multiple values can be specified for AttendeesEmails and AttendeesDisplayNames.

```
Insert Into [MyCalendar]
(StartDateTime,EndDateTime,AttendeesEmails#1,AttendeesEmails#2) VALUES
('2017-03-15 15:00','2017-03-15 20:00','test@test.com','test1@test.com')
```

Update

Update details of a specific event. The Id of the event being updated must be specified.

```
UPDATE [MyCalendar] SET Summary='Updated Summary 2' WHERE
Id='8ba774m3anenroqcepf7ka6ok'
```

Delete

Delete an event by specifying its Id.

```
Delete FROM [MyCalendar] WHERE Id='8ba774m3anenroqcepf7ka6ok'
```

Columns

Name	Type	ReadOnly	Description
Id [KEY]	<i>String</i>	True	The Id of the event.
CalendarId	<i>String</i>	True	Calendar Id
Summary	<i>String</i>	False	The title of the event.
Description	<i>String</i>	False	The description of the event.
Location	<i>String</i>	False	The geographic location of the event as free-form text.
AllDayEvent	<i>Boolean</i>	False	This value sets whether or not the event is an all-day event.
StartDate	<i>Date</i>	False	The start date if this is an all-day event..
StartDateTime	<i>Datetime</i>	False	The (inclusive) start time of the event. For a recurring event, this value is the start time of the first instance.
StartDateTimeZone	<i>String</i>	False	The time zone in which the start date time is specified.
EndDate	<i>Datet</i>	False	The end date if this is an all-day event.
EndTime	<i>Datetime</i>	False	The (exclusive) end time of the event. For a recurring event, this value is the end time of the first instance.

EndTimeZone	<i>String</i>	False	The time zone in which the end date time is specified.
OriginalStartTimeDateTime	<i>Datetime</i>	False	For an instance of a recurring event, this value is the time when the event would start according to the recurrence data in the recurring event identified by RecurringEventId.
SendNotification	<i>Boolean</i>	False	This value sets whether to send a notification when performing an insert or update.
Kind	<i>String</i>	True	The type of the resource, returned in the format calendar#event.
ETag	<i>String</i>	True	The ETag of the resource.
Status	<i>String</i>	False	The status of the event.
HTMLLink	<i>String</i>	True	The absolute link to the event in the Google Calendar Web UI.
Created	<i>Datetime</i>	True	The creation time of the event.
Updated	<i>Datetime</i>	True	The latest modification time of the event.
ColorId	<i>Integer</i>	False	The color of the event. This value is an Id referring to an entry in the event section of the colors definitions.

CreatorEmail	<i>String</i>	True	The creator's email address, if available.
CreatorDisplayName	<i>String</i>	True	The creator's name, if available.
OrganizerEmail	<i>String</i>	False	The organizer's email address, if available.
OrganizerDisplayName	<i>String</i>	False	The organizer's name, if available.
Recurrences	<i>String</i>	False	A pipe-separated list of RRULE, EXRULE, RDATE, and EXDATE lines for a recurring event. This field is omitted for single events or instances of recurring events. OriginalStartTimeDateTime must be set in order to modify this value.
RecurringEventId	<i>String</i>	True	For an instance of a recurring event, this value is the event Id of the recurring event itself.
Transparency	<i>String</i>	False	This value sets whether the event blocks time on the calendar. If set to transparent, the event does not block time on the calendar. If set to opaque, the event blocks time; this is the default value.
Visibility	<i>String</i>	False	The visibility of the event.

ICalUid	<i>String</i>	True	The event Id in the iCalendar format.
Sequence	<i>String</i>	False	The sequence number as per iCalendar.
AttendeesEmails	<i>String</i>	False	A comma-separated list of attendee's email addresses, if available.
AttendeesDisplayNames	<i>String</i>	False	A comma-separated list of attendee's names, if available.
AttendeesOmitted	<i>Boolean</i>	True	This field sets whether attendees have been omitted from the event's representation. When updating an event, this field can be used to update only the participant's response. When retrieving an event, the attendees that are returned are restricted to only the participant by the MaxAttendees query parameter.
ExtendedPropertiesPrivateKey	<i>String</i>	False	This field contains properties that are private to the copy of the event that appears on the calendar.
ExtendedPropertiesPrivateValue	<i>String</i>	False	This field contains properties that are private to the copy of the event that appears on the calendar.
ExtendedPropertiesSharedKey	<i>String</i>	False	This field contains properties that are shared

			between copies of the event on other attendees' calendars.
ExtendedPropertiesSharedValue	<i>String</i>	False	This field contains properties that are shared between copies of the event on other attendees' calendars.
AnyoneCanAddSelf	<i>Boolean</i>	True	This value sets whether anyone can invite themselves to the event.
GuestsCanInviteOthers	<i>Boolean</i>	False	This value sets whether attendees other than the organizer can invite others to the event.
GuestsCanSeeOtherGuests	<i>Boolean</i>	False	This value sets whether attendees other than the organizer can see who the event's attendees are.
GuestsCanModify	<i>Boolean</i>	False	Whether attendees other than the organizer can modify the event.
PrivateCopy	<i>Boolean</i>	True	This value sets whether this is a private event copy where changes are not shared with other copies on other calendars.
RemindersUseDefault	<i>Boolean</i>	False	This value sets whether the default reminders of the calendar apply to the event.
ReminderOverrideMethods	<i>String</i>	False	A comma-separated list of the methods used by the

			reminder. The possible values are EMAIL, SMS, and POPUP.
ReminderOverrideMinutes	<i>String</i>	False	A comma-separated list of the minutes before the start of the event when the corresponding ReminderOverrideMethod should trigger.

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
SearchTerms	<i>String</i>	Free text search terms to find events that match these terms in any field, except for extended properties.
SingleEvents	<i>Boolean</i>	Whether to expand recurring events into instances, default value is false.

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. To find out more about tables and stored procedures, please navigate to their corresponding entries in this help document.

Google Calendar Adapter Views

Name	Description
Colors	Query colors for Google calendars and events.
EventInstances	Query event instances in a Google calendar.

Colors

Query colors for Google calendars and events.

Table-Specific Information

Select

Returns the color definitions for calendars and events.

```
SELECT * FROM Colors
```

Columns

Name	Type	Description
Id [KEY]	String	The unique color identifier composed of the color type and Id separated by the pipe character.
EntityType [KEY]	String	The type of the entity the color is specified for. It can be either Calendar or Event

ColorBackground	<i>String</i>	The background color for the calendar color.
ColorForeground	<i>String</i>	The foreground color for the calendar color.
Updated	<i>Datetime</i>	The last time the list of colors was updated.

EventInstances

Query event instances in a Google calendar.

Table-Specific Information

Select

Returns instances of the specified recurring event. CalendarId and EventId must be specified.

A recurring event consists of several instances: its particular occurrences at different times. These instances act as events themselves.

```
SELECT * FROM EventInstances WHERE CalendarId='thecalendarid' AND
EventId='sq12g65ed3nk0n3n7sc6q6k0q8'
```

Columns

Name	Type	Description
Id [KEY]	<i>String</i>	The Id of the event.

CalendarId	<i>String</i>	The calendar identifier.
EventId	<i>String</i>	The event identifier.
Summary	<i>String</i>	The title of the event.
Description	<i>String</i>	The description of the event.
Location	<i>String</i>	The geographic location of the event as free-form text.
AllDayEvent	<i>Boolean</i>	This value sets whether or not the event is an all-day event.
StartDateTime	<i>Datetime</i>	The (inclusive) start time of the event. For a recurring event, this value is the start time of the first instance.
EndDateTime	<i>Datetime</i>	The (exclusive) end time of the event. For a recurring event, this value is the end time of the first instance.
OriginalStartTimeDateTime	<i>Datetime</i>	For an instance of a recurring event, this value is the time when the event would start according to the recurrence data in the recurring event identified by RecurringEventId.
SendNotification	<i>Boolean</i>	This value sets whether to send a notification when performing an insert or update.
Kind	<i>String</i>	The type of the resource, returned in the format calendar#event.
ETag	<i>String</i>	The ETag of the resource.

Status	<i>String</i>	The status of the event.
HTMLLink	<i>String</i>	The absolute link to the event in the Google Calendar Web UI.
Created	<i>Datetime</i>	The creation time of the event.
Updated	<i>Datetime</i>	The latest modification time of the event.
ColorId	<i>Integer</i>	The color of the event. This value is an Id referring to an entry in the event section of the colors definitions.
CreatorEmail	<i>String</i>	The creator's email address, if available.
CreatorDisplayName	<i>String</i>	The creator's name, if available.
OrganizerEmail	<i>String</i>	The organizer's email address, if available.
OrganizerDisplayName	<i>String</i>	The organizer's name, if available.
RecurringEventId	<i>String</i>	For an instance of a recurring event, this value is the event Id of the recurring event itself.
Transparency	<i>String</i>	This value sets whether the event blocks time on the calendar. If set to transparent, the event does not block time on the calendar. If set to opaque, the event blocks time; this is the default value.

Visibility	<i>String</i>	The visibility of the event.
iCalUid	<i>String</i>	The event Id in the iCalendar format.
Sequence	<i>String</i>	The sequence number as per iCalendar.
AttendeesAggregate	<i>String</i>	An XML aggregate of the event's attendees. This includes the values Email, DisplayName, Organizer, Self, Resource, Optional, ResponseStatus, Comment, and AddtnlGuests.
AttendeesEmails	<i>String</i>	A comma-separated list of attendee's email addresses, if available.
AttendeesDisplayNames	<i>String</i>	A comma-separated list of attendee's names, if available.
AttendeesResponseStatus	<i>String</i>	A comma-separated list of attendee's response status, if available.
AttendeesOmitted	<i>Boolean</i>	This field sets whether attendees have been omitted from the event's representation. When updating an event, this field can be used to update only the participant's response. When retrieving an event, the attendees that are returned are restricted to only the participant by the MaxAttendees query parameter.
ExtendedPropertiesPrivateKey	<i>String</i>	This field contains properties that are private to the copy of the event that appears on the calendar.
ExtendedPropertiesPrivateValue	<i>String</i>	This field contains properties that are private to the copy of the event that appears on the calendar.

ExtendedPropertiesSharedKey	<i>String</i>	This field contains properties that are shared between copies of the event on other attendees' calendars.
ExtendedPropertiesSharedValue	<i>String</i>	This field contains properties that are shared between copies of the event on other attendees' calendars.
AnyoneCanAddSelf	<i>Boolean</i>	This value sets whether anyone can invite themselves to the event.
GuestsCanInviteOthers	<i>Boolean</i>	This value sets whether attendees other than the organizer can invite others to the event.
GuestsCanSeeOtherGuests	<i>Boolean</i>	This value sets whether attendees other than the organizer can see who the event's attendees are.
PrivateCopy	<i>Boolean</i>	This value sets whether this is a private event copy where changes are not shared with other copies on other calendars.
RemindersUseDefault	<i>Boolean</i>	This value sets whether the default reminders of the calendar apply to the event.
ReminderOverrideMethods	<i>String</i>	A comma-separated list of the methods used by the reminder. The possible values are EMAIL, SMS, and POPUP.
ReminderOverrideMinutes	<i>String</i>	A comma-separated list of the minutes before the start of the event when the corresponding ReminderOverrideMethod should trigger.

Stored Procedures

Stored procedures are available to complement the data available from the [Data Model](#). It may be necessary to update data available from a view using a stored procedure because

the data does not provide for direct, table-like, two-way updates. In these situations, the retrieval of the data is done using the appropriate view or table, while the update is done by calling a stored procedure. Stored procedures take a list of parameters and return back a dataset that contains the collection of tuples that constitute the response.

Google Calendar Adapter Stored Procedures

Name	Description
GetOAuthAccessToken	Obtains the OAuth access token to be used for authentication with various Google services.
GetOAuthAuthorizationURL	Obtains the OAuth authorization URL used for authentication with various Google services.
RefreshOAuthAccessToken	Obtains the OAuth access token to be used for authentication with various Google services.

GetOAuthAccessToken

Obtains the OAuth access token to be used for authentication with various Google services.

Input

Name	Type	Required	Description
AuthMode	<i>String</i>	<i>True</i>	The type of authentication mode to use. The allowed values are <i>APP</i> , <i>WEB</i> . The default value is <i>WEB</i> .
Verifier	<i>String</i>	<i>False</i>	The verifier code returned by Google after permission for the app to connect has been

			granted. WEB AuthMode only.
Scope	String	True	<p>The scope of access to Google APIs. By default, access to all APIs used by this data provider will be specified.</p> <p>The default value is https://www.googleapis.com/auth/calendar.</p>
CallbackURL	String	False	<p>This field determines where the response is sent. The value of this parameter must exactly match one of the values registered in the APIs Console, including the HTTP or HTTPS schemes, capitalization, and trailing forward slash ('/').</p>
Prompt	String	True	<p>This field indicates the prompt to present the user. The default is <code>CONSENT</code>, so a given user will see a consent page every time, even if they have previously given consent to the application for a given set of scopes. If it is set to <code>SELECT_ACCOUNT</code>, the user will be prompted to select the account to connect to. Lastly, if it is set to <code>NONE</code>, no authentication or consent screens will be displayed to the user.</p> <p>The allowed values are <code>NONE</code>, <code>CONSENT</code>, <code>SELECT_ACCOUNT</code>.</p> <p>The default value is <code>CONSENT</code>.</p>
AccessType	String	True	<p>This field indicates if your application needs to access a Google API when the user is not present at the browser. This parameter defaults to <code>OFFLINE</code>. If your application needs to refresh access tokens when the user is not present at the browser, then use <code>OFFLINE</code>. This will result in your application obtaining a refresh token the first time your application exchanges an authorization code for a user.</p>

			<p>The allowed values are <i>ONLINE</i>, <i>OFFLINE</i>.</p> <p>The default value is <i>OFFLINE</i>.</p>
State	String	False	<p>This field indicates any state that may be useful to your application upon receipt of the response. Your application receives the same value it sent, as this parameter makes a round-trip to Google authorization server and back. Uses include redirecting the user to the correct resource in your site, using nonces, and mitigating cross-site request forgery.</p>

Result Set Columns

Name	Type	Description
OAuthAccessToken	String	The authentication token returned from Google. This can be used in subsequent calls to other operations for this particular service.
OAuthRefreshToken	String	A token that may be used to obtain a new access token.
ExpiresIn	String	The remaining lifetime on the access token.

GetOAuthAuthorizationURL

Obtains the OAuth authorization URL used for authentication with various Google services.

Input

Name	Type	Required	Description
Scope	<i>String</i>	<i>True</i>	<p>The scope of access to Google APIs. By default, access to all APIs used by this data provider will be specified.</p> <p>The default value is https://www.googleapis.com/auth/calendar.</p>
CallbackURL	<i>String</i>	<i>False</i>	<p>This field determines where the response is sent. The value of this parameter must exactly match one of the values registered in the APIs Console, including the HTTP or HTTPS schemes, case, and trailing forward slash ('/').</p>
Prompt	<i>String</i>	<i>True</i>	<p>This field indicates the prompt to present the user. The default is <i>CONSENT</i>, so a given user will see a consent page every time, even if they have previously given consent to the application for a given set of scopes. If it is set to <i>SELECT_ACCOUNT</i>, the user will be prompted to select the account to connect to. Lastly, if it is set to <i>NONE</i>, no authentication or consent screens will be displayed to the user.</p> <p>The allowed values are <i>NONE</i>, <i>CONSENT</i>, <i>SELECT_ACCOUNT</i>.</p> <p>The default value is <i>CONSENT</i>.</p>
AccessType	<i>String</i>	<i>True</i>	<p>This field indicates if your application needs to access a Google API when the user is not present at the browser. This parameter defaults to <i>OFFLINE</i>. If your application needs to refresh access tokens when the user is not present at the browser, then use <i>OFFLINE</i>. This will result in your application obtaining a refresh token the first time your application exchanges an authorization code for a user.</p> <p>The allowed values are <i>ONLINE</i>, <i>OFFLINE</i>.</p>

			The default value is <i>OFFLINE</i> .
State	String	False	This field indicates any state that may be useful to your application upon receipt of the response. Your application receives the same value it sent, as this parameter makes a round-trip to the Google authorization server and back. Possible uses include redirecting the user to the correct resource in your site, using nonces, and mitigating cross-site request forgery.

Result Set Columns

Name	Type	Description
URL	String	The URL to complete user authentication.

RefreshOAuthAccessToken

Obtains the OAuth access token to be used for authentication with various Google services.

Input

Name	Type	Required	Description
OAuthRefreshToken	String	True	The refresh token returned from the original authorization code exchange.

Result Set Columns

Name	Type	Description
OAuthAccessToken	String	The authentication token returned from Google. This can be used in subsequent calls to other operations for this particular service.
OAuthRefreshToken	String	The authentication token returned from Google. This can be used in subsequent calls to other operations for this particular service.
ExpiresIn	String	The remaining lifetime on the access token.

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.

Auth Scheme	The type of authentication to use when connecting to Google Calendar.
Firewall Password	A password used to authenticate to a proxy-based firewall.
Firewall Port	The TCP port for a proxy-based firewall.
Firewall Server	The name or IP address of a proxy-based firewall.
Firewall Type	The protocol used by a proxy-based firewall.
Firewall User	The user name to use to authenticate with a proxy-based firewall.

Initiate OAuth	Set this property to initiate the process to obtain or refresh the OAuth access token when you connect.
Location	A path to the directory that contains the schema files defining tables, views, and stored procedures.
Log Modules	Core modules to be included in the log file.
Max Rows	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.
OAuth Access Token	The access token for connecting using OAuth.
OAuth Client Id	The client ID assigned when you register your application with an OAuth authorization server.
OAuth Client Secret	The client secret assigned when you register your application with an OAuth authorization server.
OAuth Expires In	The lifetime in seconds of the OAuth AccessToken.
OAuth JWT Cert	The JWT Certificate store.
OAuth JWT Cert Password	The password for the OAuth JWT certificate.
OAuth JWT Cert Subject	The subject of the OAuth JWT certificate.
OAuth JWT Cert Type	The type of key store containing the JWT Certificate.
OAuth JWT Issuer	The issuer of the Java Web Token.
OAuth JWT	The user subject for which the application is requesting delegated access.

Subject	
OAuth Refresh Token	The OAuth refresh token for the corresponding OAuth access token.
OAuth Settings Location	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://.
OAuth Token Timestamp	The Unix epoch timestamp in milliseconds when the current Access Token was created.
OAuth Verifier	The verifier code returned from the OAuth authorization URL.
Other	These hidden properties are used only in specific use cases.
Proxy Auth Scheme	The authentication type to use to authenticate to the ProxyServer proxy.
Proxy Auto Detect	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.
Proxy Exceptions	A semicolon separated list of destination hostnames or IPs that are exempt from connecting through the ProxyServer .
Proxy Password	A password to be used to authenticate to the ProxyServer proxy.
Proxy Port	The TCP port the ProxyServer proxy is running on.
Proxy Server	The hostname or IP address of a proxy to route HTTP traffic through.
Proxy SSL Type	The SSL type to use when connecting to the ProxyServer proxy.
Proxy User	A user name to be used to authenticate to the ProxyServer proxy.
Readonly	You can use this property to enforce read-only access to Google Calendar

	from the provider.
SSL Server Cert	The certificate to be accepted from the server when connecting using TLS/SSL.
Timeout	The value in seconds until the timeout error is thrown, canceling the operation.

Auth Scheme

The type of authentication to use when connecting to Google Calendar.

Data Type

string

Default Value

"Auto"

Remarks

- Auto: Lets the driver decide automatically based on the other connection properties you have set.
- OAuth: Set this to perform OAuth authentication using a standard user account.
- OAuthJWT: Set this to perform OAuth authentication using an OAuth service account.
- GCPIstanceAccount: Set this to get Access Token from Google Cloud Platform instance.

Firewall Password

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](#).

Firewall Port

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](#).

Firewall Server

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.

Firewall Type

The protocol used by a proxy-based firewall.

Data Type

string

Default Value

"NONE"

Remarks

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

Type	Default Port	Description
TUNNEL	80	When this is set, the adapter opens a connection to Google Calendar 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 FirewallServer and FirewallPort and passes the FirewallUser 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 FirewallServer and FirewallPort . If your proxy requires authentication, set FirewallUser and FirewallPassword to credentials the proxy recognizes.

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

Firewall User

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](#).

Initiate OAuth

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

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.

Location

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

Data Type

string

Default Value

"%APPDATA%\\CData\\GoogleCalendar 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\\GoogleCalendar 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

Log Modules

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.

Max Rows

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.

OAuth Access Token

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.

OAuth Client Id

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](#).

OAuth Client Secret

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.

OAuth Expires In

The lifetime in seconds of the OAuth AccessToken.

Data Type

string

Default Value

""

Remarks

Pair with OAuthTokenTimestamp to determine when the AccessToken will expire.

OAuth JWT Cert

The JWT Certificate store.

Data Type

string

Default Value

""

Remarks

The name of the certificate store for the client certificate.

The [OAuthJWTCertType](#) field specifies the type of the certificate store specified by [OAuthJWTCert](#). If the store is password protected, specify the password in [OAuthJWTCertPassword](#).

[OAuthJWTCert](#) is used in conjunction with the [OAuthJWTCertSubject](#) field in order to specify client certificates. If [OAuthJWTCert](#) has a value, and [OAuthJWTCertSubject](#) is set, a search for a certificate is initiated. Please refer to the [OAuthJWTCertSubject](#) field for details.

Designations of certificate stores are platform-dependent.

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

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

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

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

OAuth JWT Cert Password

The password for the OAuth JWT certificate.

Data Type

string

Default Value

""

Remarks

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

This is not required when using the GOOGLEJSON [OAuthJWTCertType](#). Google JSON keys are not encrypted.

OAuth JWT Cert Subject

The subject of the OAuth JWT certificate.

Data Type

string

Default Value

"*"

Remarks

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

If an exact match is not found, the store is searched for subjects containing the value of the property.

If a match is still not found, the property is set to an empty string, and no certificate is selected.

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

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

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

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

OAuth JWT Cert Type

The type of key store containing the JWT Certificate.

Data Type

string

Default Value

"USER"

Remarks

This property can take one of the following values:

USER - default	For Windows, this specifies that the certificate store is a certificate store owned by the current user. <i>Note:</i> This store type is not available in Java.
MACHINE	For Windows, this specifies that the certificate store is a machine store. <i>Note:</i> this store type is not available in Java.
PFXFILE	The certificate store is the name of a PFX (PKCS12) file containing certificates.
PFXBLOB	The certificate store is a string (base-64-encoded) representing a certificate store in PFX (PKCS12) format.
JKSFILE	The certificate store is the name of a Java key store (JKS) file containing certificates. <i>Note:</i> this store type is only available in Java.
JKSBLOB	The certificate store is a string (base-64-encoded) representing a certificate store in Java key store (JKS) format. <i>Note:</i> this store type is only available in Java.
PEMKEY_FILE	The certificate store is the name of a PEM-encoded file that contains a private key and an optional certificate.
PEMKEY_BLOB	The certificate store is a string (base64-encoded) that contains a private key and an optional certificate.
PUBLIC_KEY_FILE	The certificate store is the name of a file that contains a PEM- or DER-encoded public key certificate.
PUBLIC_KEY_BLOB	The certificate store is a string (base-64-encoded) that contains a PEM- or DER-encoded public key certificate.
SSHPUBLIC_KEY_FILE	The certificate store is the name of a file that contains an SSH-style public key.
SSHPUBLIC_KEY_BLOB	The certificate store is a string (base-64-encoded) that contains an SSH-style public key.
P7BFILE	The certificate store is the name of a PKCS7 file containing

	certificates.
PPKFILE	The certificate store is the name of a file that contains a PPK (PuTTY Private Key).
XMLFILE	The certificate store is the name of a file that contains a certificate in XML format.
XMLBLOB	The certificate store is a string that contains a certificate in XML format.
GOOGLEJSON	The certificate store is the name of a JSON file containing the service account information. Only valid when connecting to a Google service.

OAuth JWT Issuer

The issuer of the Java Web Token.

Data Type

string

Default Value

""

Remarks

The issuer of the Java Web Token. This is typically either the Client ID or Email Address of the OAuth Application.

This is not required when using the GOOGLEJSON [OAuthJWTCertType](#). Google JSON keys contain a copy of the issuer account.

OAuth JWT Subject

The user subject for which the application is requesting delegated access.

Data Type

string

Default Value

""

Remarks

The user subject for which the application is requesting delegated access. Typically, the user account name or email address.

OAuth Refresh Token

The OAuth refresh token for the corresponding OAuth access token.

Data Type

string

Default Value

""

Remarks

The OAuthRefreshToken property is used to refresh the [OAuthAccessToken](#) when using OAuth authentication.

OAuth Settings Location

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\\GoogleCalendar 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\\GoogleCalendar 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

OAuth Token Timestamp

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.

OAuth Verifier

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 [Getting Started](#) 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.

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.

Proxy Auth Scheme

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

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](#).

Proxy Auto Detect

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](#).

Proxy Exceptions

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.

Proxy Password

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.

Proxy Port

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](#).

Proxy Server

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.

Proxy SSL Type

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

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:

AUTO	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.
ALWAYS	The connection is always SSL enabled.
NEVER	The connection is not SSL enabled.
TUNNEL	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.

Proxy User

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
```

Readonly

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

SSL Server Cert

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----- MIICHTCCAe4CAQAwDQYJKoZIhvc... ...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)	cc58a1e9e09828d70e4
The SHA1 Thumbprint (hex values can also be either space or colon separated)	34a629326a081f2ec14b4a3d904f801cbb150d

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.

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.

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.

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.

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.

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, Two-Second Advantage, TIBCO Spotfire, TIBCO ActiveSpaces, TIBCO Spotfire Developer, TIBCO EMS, TIBCO Spotfire Automation Services, TIBCO Enterprise Runtime for R, TIBCO Spotfire Server, TIBCO Spotfire Web Player, TIBCO Spotfire Statistics Services, S-PLUS, and TIBCO Spotfire 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.