



# **TIBCO® Data Virtualization**

## **Google Directory Adapter Guide**

Version 8.7.0 | October 2023



# Contents

---

<b>Contents</b>	<b>2</b>
<b>Google Directory Adapter</b>	<b>4</b>
Getting Started	4
Basic Tab	5
Logging	10
Creating a Custom OAuth App	12
Changelog	14
Advanced Features	17
User Defined Views	18
SSL Configuration	21
Firewall and Proxy	21
Query Processing	22
Logging	22
SQL Compliance	25
SELECT Statements	26
SELECT INTO Statements	28
INSERT Statements	29
UPDATE Statements	30
DELETE Statements	30
EXECUTE Statements	31
PIVOT and UNPIVOT	32
Data Model	33
Tables	33
Views	60
Stored Procedures	79
Connection String Options	86
Authentication	90



OAuth .....	91
JWT OAuth .....	99
SSL .....	105
Firewall .....	106
Proxy .....	110
Logging .....	116
Schema .....	117
Miscellaneous .....	118
<b>TIBCO Product Documentation and Support Services .....</b>	<b>126</b>
How to Access TIBCO Documentation .....	126
How to Contact TIBCO Support .....	127
Release Version Support .....	127
How to Join TIBCO Community .....	128
<b>Legal and Third-Party Notices .....</b>	<b>129</b>



# Google Directory Adapter

---

## Google Directory Version Support

### SQL Compliance

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

## Getting Started

### Connecting to Google Directory

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

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

1. Unzip the `tdv.googledirectory.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.googledirectory/tdv.googledirectory.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 GoogleDirectory
```

## Basic Tab

### Authenticating to Google Directory

The adapter supports using user accounts, service accounts and GCP instance accounts for authentication.

The following sections discuss the available authentication schemes for Google Directory:

- User Accounts (OAuth)
- Service Account (OAuthJWT)
- GCP Instance Account

### User Accounts (OAuth)

AuthScheme must be set to **OAuth** in all user account flows.

### Web Applications

When connecting via a Web application, you need to create and register a custom OAuth application with Google Directory. You can then use the adapter to acquire and manage the OAuth token values. See [Creating a Custom OAuth App](#) for more information about custom applications.

#### Get an OAuth Access Token

Set the following connection properties to obtain the OAuthAccessToken:

- OAuthClientId: Set this to the Client Id in your application settings.
- OAuthClientSecret: Set this to the Client Secret in your application settings.

Then call stored procedures to complete the OAuth exchange:

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



the Callback URL you specified in your application settings. The stored procedure returns the URL to the OAuth endpoint.

2. Navigate to the URL that the stored procedure returned in Step 1. Log in to the custom OAuth application and authorize the web application. Once authenticated, the browser redirects you to the callback URL.
3. Call the [GetOAuthAccessToken](#) stored procedure. Set AuthMode to **WEB** and the Verifier input to the "code" parameter in the query string of the callback URL.

Once you have obtained the access and refresh tokens, you can connect to data and refresh the OAuth access token either automatically or manually.

### Automatic Refresh of the OAuth Access Token

To have the driver automatically refresh the OAuth access token, set the following on the first data connection:

- InitiateOAuth: Set this to **REFRESH**.
- OAuthClientId: Set this to the Client Id in your application settings.
- OAuthClientSecret: Set this to the Client Secret in your application settings.
- OAuthAccessToken: Set this to the access token returned by [GetOAuthAccessToken](#).
- OAuthRefreshToken: Set this to the refresh token returned by [GetOAuthAccessToken](#).
- OAuthSettingsLocation: Set this to the path where the adapter saves the OAuth token values, which persist across connections.

On subsequent data connections, the values for OAuthAccessToken and OAuthRefreshToken are taken from OAuthSettingsLocation.

### Manual Refresh of the OAuth Access Token

The only value needed to manually refresh the OAuth access token when connecting to data is the OAuth refresh token.

Use the [RefreshOAuthAccessToken](#) stored procedure to manually refresh the OAuthAccessToken after the ExpiresIn parameter value returned by [GetOAuthAccessToken](#) has elapsed, then set the following connection properties:

- OAuthClientId: Set this to the Client Id in your application settings.
- OAuthClientSecret: Set this to the Client Secret in your application settings.



Then call [RefreshOAuthAccessToken](#) with [OAuthRefreshToken](#) set to the OAuth refresh token returned by [GetOAuthAccessToken](#). After the new tokens have been retrieved, open a new connection by setting the [OAuthAccessToken](#) property to the value returned by [RefreshOAuthAccessToken](#).

Finally, store the OAuth refresh token so that you can use it to manually refresh the OAuth access token after it has expired.

## Headless Machines

To configure the driver to use OAuth with a user account on a headless machine, you need to authenticate on another device that has an internet browser.

1. Choose one of two options:
  - Option 1: Obtain the [OAuthVerifier](#) value as described in "Obtain and Exchange a Verifier Code" below.
  - Option 2: Install the adapter on a machine with an Internet browser and transfer the OAuth authentication values after you authenticate through the usual browser-based flow, as described in "Transfer OAuth Settings" below.
2. Then configure the adapter to automatically refresh the access token on the headless machine.

### Option 1: Obtain and Exchange a Verifier Code

To obtain a verifier code, you must authenticate at the OAuth authorization URL.

Follow the steps below to authenticate from the machine with an Internet browser and obtain the [OAuthVerifier](#) connection property.

1. Choose one of these options:
  - If you are using the Embedded OAuth Application click [Google Directory OAuth endpoint](#) to open the endpoint in your browser.
  - If you are using a custom OAuth application, create the Authorization URL by setting the following properties:
    - [InitiateOAuth](#): Set to **OFF**.
    - [OAuthClientId](#): Set to the client Id assigned when you registered your application.



- OAuthClientSecret: Set to the client secret assigned when you registered your application.

Then call the [GetOAuthAuthorizationURL](#) stored procedure with the appropriate CallbackURL. Open the URL returned by the stored procedure in a 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. Later you will set this in the OAuthVerifier connection property.

Next, you need to exchange the OAuth verifier code for OAuth refresh and access tokens. Set the following properties:

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

- InitiateOAuth: Set this to **REFRESH**.
- OAuthVerifier: Set this to the verifier code.
- OAuthClientId: (custom applications only) Set this to the Client Id in your custom OAuth application settings.
- OAuthClientSecret: (custom applications only) Set this to the Client Secret in the custom OAuth application settings.
- OAuthSettingsLocation: Set this to persist the encrypted OAuth authentication values to the specified file.

After the OAuth settings file is generated, you need to re-set the following properties to connect:

- InitiateOAuth: Set this to **REFRESH**.
- OAuthClientId: (custom applications only) Set this to the client Id assigned when you registered your application.
- OAuthClientSecret: (custom applications only) Set this to the client secret assigned when you registered your application.
- 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.

## Option 2: Transfer OAuth Settings



Prior to connecting on a headless machine, you need to create and install a connection with the driver on a device that supports an Internet browser. Set the connection properties as described in "Desktop Applications" above.

After completing the instructions in "Desktop Applications", the resulting authentication values are encrypted and written to the path specified by OAuthSettingsLocation. The default filename is *OAuthSettings.txt*.

Once you have successfully tested the connection, copy the OAuth settings file to your headless machine.

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

- InitiateOAuth: Set this to **REFRESH**.
- OAuthClientId: (custom applications only) Set this to the client Id assigned when you registered your application.
- OAuthClientSecret: (custom applications only) Set this to the client secret assigned when you registered your application.
- 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.

## Service Accounts (OAuthJWT)

To authenticate using a service account, you must create a new service account and have a copy of the accounts certificate. If you do not already have a service account, you can create one by following the procedure in [Creating a Custom OAuth App](#).

For a JSON file, set these properties:

- AuthScheme: Set this to **OAuthJWT**.
- InitiateOAuth: Set this to **GETANDREFRESH**.
- OAuthJWTCertType: Set this to **GOOGLEJSON**.
- OAuthJWTCert: Set this to the path to the *.json* file provided by Google.
- OAuthJWTSubject: (optional) Only set this value if the service account is part of a GSuite domain and you want to enable delegation. The value of this property should be the email address of the user whose data you want to access.

For a PFX file, set these properties instead:



- AuthScheme: Set this to **OAuthJWT**.
- InitiateOAuth: Set this to **GETANDREFRESH**.
- OAuthJWTCertType: Set this to **PFXFILE**.
- OAuthJWTCert: Set this to the path to the *.pfx* file provided by Google.
- OAuthJWTCertPassword: (optional) Set this to the *.pfx* file password. In most cases you must provide this since Google encrypts PFX certificates.
- OAuthJWTCertSubject: (optional) Set this only if you are using a OAuthJWTCertType which stores multiple certificates. Should not be set for PFX certificates generated by Google.
- OAuthJWTIssuer: Set this to the email address of the service account. This address will usually include the domain **iam.gserviceaccount.com**.
- OAuthJWTSubject: (optional) Only set this value if the service account is part of a GSuite domain and you want to enable delegation. The value of this property should be the email address of the user whose data you want to access.

## GCP Instance Accounts

When running on a GCP virtual machine, the adapter can authenticate using a service account tied to the virtual machine. To use this mode, set AuthScheme to **GCPInstanceAccount**.

## Logging

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

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

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

```
Verbosity=4;
```



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

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

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

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

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

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

## Configure Logging for the Google Directory Adapter

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

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

The TDV Server must be restarted after changing the log4j.properties file, which can be accomplished by running the composite.bat script located at: C:\Program Files\TIBCO\TDV



Server <version>\bin. Note that reauthenticating to the TDV Studio is required after restarting the server.

Here is an example of the calls:

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

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

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

## Creating a Custom OAuth App

### Introduction

You can use a custom OAuth app to authenticate a service account or a user account. You can always create a custom OAuth application, but note that desktop and headless connections support embedded OAuth, which simplifies the process of authentication.

### When To Create a Custom OAuth Application

CData embeds OAuth Application Credentials with CData branding that can be used when connecting via either a Desktop Application or from a Headless Machine.

You may choose to use your own OAuth Application Credentials when you want to

- control branding of the Authentication Dialog
- control the redirect URI that the application redirects the user to after the user authenticates
- customize the permissions that you are requesting from the user

### Enable the Directory API

Follow these steps to enable the Directory API:

1. Navigate to the [Google Cloud Console](#).



2. Select **Library** from the left-hand navigation menu. This opens the **Library** page.
3. In the search field, enter "Directory API" and select **Directory API** from the search results.
4. On the **Directory API** page, click **ENABLE**.

## Create an OAuth Application for User Accounts (OAuth)

When using AuthScheme=OAuth, and you're using a web application, you must create an OAuth Client ID Application. For desktop and headless flows, creating a custom OAuth application is optional.

Follow these steps to create a custom OAuth application:

1. Navigate to the [Google Cloud Console](#).
2. If you have not done so, follow the steps in the console to create an OAuth consent screen.
3. Select **Credentials** from the left-hand navigation menu.
4. On the **Credentials** page, select **Create Credentials > OAuth Client ID**.
5. In the **Application Type** menu, select **Web application**.
6. Specify a name for your OAuth custom web application.
7. Under **Authorized redirect URIs**, click **ADD URI** and enter a redirect URI. Press **Enter**.
8. Click **CREATE**, which returns you to the **Credentials** page.
9. A window opens that displays your client Id and client secret. Although the client secret is accessible from from the Google Cloud Console, we recommend you write down the client secret. You need both the client secret and client Id to specify the OAuthClientId and OAuthClientSecret connection properties.

## Create an OAuth Application for Service Accounts (OAuthJWT)

When using AuthScheme=OAuthJWT, you must create a Service Account Application. Follow these steps:

1. Navigate to the [Google Cloud Console](#).
2. If you have not done so, follow the steps in the console to create an OAuth consent



screen.

3. Select **Credentials** from the left-hand navigation menu.
4. On the **Credentials** page, select **Create Credentials > Service account**.
5. On the **Create service account** page, enter the Service account name, the Service account ID, and, optionally, a description.
6. Click **DONE**. This returns you to the **Credentials** page.

## Changelog

### General Changes

Date	Build Number	Change Type	Description
12/14/2022	8383	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Added the Default column to the sys_procedureparameters table.</li> </ul>
09/30/2022	8308	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Added the IsPath column to the sys_procedureparameters table.</li> </ul>
09/14/2022	8292	Google Directory	<b>Changed</b> <ul style="list-style-type: none"> <li>Changed the default Scope to be readonly (<a href="https://www.googleapis.com/auth/admin.directory.user.readonly">https://www.googleapis.com/auth/admin.directory.user.readonly</a> <a href="https://www.googleapis.com/auth/admin.directory.user.alias.readonly">https://www.googleapis.com/auth/admin.directory.user.alias.readonly</a> <a href="https://www.googleapis.com/auth/admin.directory.rolemanagement.readonly">https://www.googleapis.com/auth/admin.directory.rolemanagement.readonly</a> <a href="https://www.googleapis.com/auth/admin.directory.orgunit.readonly">https://www.googleapis.com/auth/admin.directory.orgunit.readonly</a>)</li> </ul>



			<a href="https://www.googleapis.com/auth/admin.directory.group.member.readonly">https://www.googleapis.com/auth/admin.directory.group.member.readonly</a> <a href="https://www.googleapis.com/auth/admin.directory.group.readonly">https://www.googleapis.com/auth/admin.directory.group.readonly</a> <a href="https://www.googleapis.com/auth/admin.directory.domain.readonly">https://www.googleapis.com/auth/admin.directory.domain.readonly</a> <a href="https://www.googleapis.com/auth/admin.directory.device.chromeos.readonly">https://www.googleapis.com/auth/admin.directory.device.chromeos.readonly</a> <a href="https://www.googleapis.com/auth/admin.directory.device.mobile.readonly">https://www.googleapis.com/auth/admin.directory.device.mobile.readonly</a>
08/17/2022	8264	General	<b>Changed</b> <ul style="list-style-type: none"> <li>We now support handling the keyword "COLLATE" as standard function name as well.</li> </ul>
09/02/2021	7915	General	<b>Added</b> <ul style="list-style-type: none"> <li>Added support for the STRING_SPLIT table-valued function in the CROSS APPLY clause.</li> </ul>
08/07/2021	7889	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Added the KeySeq column to the sys_foreignkeys table.</li> </ul>
08/06/2021	7888	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Added the new sys_primarykeys system table.</li> </ul>
07/23/2021	7874	General	<b>Changed</b> <ul style="list-style-type: none"> <li>Updated the Literal Function Names for relative date/datetime functions. Previously relative date/datetime functions resolved to a different value when used in the projection vs te predicate. Ie: SELECT LAST_MONTH() AS lm, Col FROM Table WHERE Col &gt; LAST_MONTH(). Formerly the two LAST_MONTH() methods would resolve to different datetimes. Now they will match.</li> <li>As a replacement for the previous behavior, the relative date/datetime functions in the criteria may</li> </ul>



			<p>have an 'L' appended to them. I.e: WHERE col &gt; L_LAST_MONTH(). This will continue to resolve to the same values that previously were calculated in the criteria. Note that the "L_" prefix will only work in the predicate - it not available for the projection.</p>
07/08/2021	7859	General	<p><b>Added</b></p> <ul style="list-style-type: none"> <li>Added the TCP Logging Module for the logging information happening on the TCP wire protocol. The transport bytes that are incoming and ongoing will be logged at verbosity=5.</li> </ul>
06/18/2021	7839	Google Directory	<p><b>Added</b></p> <ul style="list-style-type: none"> <li>Added support for the GOOGLEJSONBLOB JWT certificate type. This works like the existing GOOGLEJSON certificate type except that the certificate is provided as JSON text instead of as a file path.</li> </ul>
04/23/2021	7785	General	<p><b>Added</b></p> <ul style="list-style-type: none"> <li>Added support for handling client side formulas during insert / update. For example: UPDATE Table SET Col1 = Concat(Col1, " - ", Col2) WHERE Col2 LIKE 'A%'</li> </ul>
04/23/2021	7783	General	<p><b>Changed</b></p> <ul style="list-style-type: none"> <li>Updated how display sizes are determined for varchar primary key and foreign key columns so they will match the reported length of the column.</li> </ul>
04/16/2021	7776	General	<p><b>Added</b></p> <ul style="list-style-type: none"> <li>Non-conditional updates between two columns is now available to all drivers. For example: UPDATE Table SET Col1=Col2</li> </ul> <p><b>Changed</b></p> <ul style="list-style-type: none"> <li>Reduced the length to 255 for varchar primary key and foreign key columns.</li> </ul>



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

04/15 /2021 7775

General

**Changed**

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

## Advanced Features

This section details a selection of advanced features of the Google Directory adapter.

### User Defined Views

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

### SSL Configuration

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



## Firewall and Proxy

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

## Query Processing

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

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

## Logging

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

## User Defined Views

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

There are two ways to create user defined views:

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

### Defining Views Using a Configuration File

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



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

This User Defined View configuration file is formatted as follows:

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

For example:

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

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

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

## Defining Views Using DDL Statements

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

### Create a View

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

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

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



## Alter a View

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

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

The view is then updated in the JSON configuration file.

## Drop a View

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

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

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

## Schema for User Defined Views

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

## Working with User Defined Views

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

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

An example of a query to the driver:

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

Resulting in the effective query to the source:

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

That is a very simple example of a query to a User Defined View that is effectively a combination of the view query and the view definition. It is possible to compose these



queries in much more complex patterns. All SQL operations are allowed in both queries and are combined when appropriate.

## SSL Configuration

### Customizing the SSL Configuration

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

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

## Firewall and Proxy

### Connecting Through a Firewall or Proxy

#### HTTP Proxies

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

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

#### Other Proxies

Set the following properties:

- To use a proxy-based firewall, set [FirewallType](#), [FirewallServer](#), and [FirewallPort](#).
- To tunnel the connection, set [FirewallType](#) to TUNNEL.
- To authenticate, specify [FirewallUser](#) and [FirewallPassword](#).
- To authenticate to a SOCKS proxy, additionally set [FirewallType](#) to SOCKS5.



# Query Processing

## Query Processing

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

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

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

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

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

## More Information

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

## Logging

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



## Basic Logging

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

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

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

## Log Verbosity

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

- |   |   |
|---|---|
| 1 | Setting <u>Verbosity</u> to 1 will log the query, the number of rows returned by it, the start of execution and the time taken, and any errors. |
| 2 | Setting <u>Verbosity</u> to 2 will log everything included in <u>Verbosity</u> 1 and additional information about the request.                  |
| 3 | Setting <u>Verbosity</u> to 3 will additionally log HTTP headers, as well as the body of the request and the response.                          |
| 4 | Setting <u>Verbosity</u> to 4 will additionally log transport-level communication with the data source. This includes SSL negotiation.          |
| 5 | Setting <u>Verbosity</u> to 5 will additionally log communication with the data source and  |



---

additional details that may be helpful in troubleshooting problems. This includes interface commands.

---

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

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

## Sensitive Data

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

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

## Best Practices for Data Security

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

## Java Logging

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

- 0: Level.WARNING
- 1: Level.INFO
- 2: Level.CONFIG
- 3: Level.FINE
- 4: Level.FINER



- 5: Level.FINEST

## Advanced Logging

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

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

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

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

An example value for this property would be.

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

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

## SQL Compliance

The Google Directory 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 Directory 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').

## 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 Directory adapter:

```

SELECT {
  [ TOP <numeric_literal> ]
  {
    *
    | {
      <expression> [ [ AS ] <column_reference> ]
      | { <table_name> | <correlation_name> } .*
    } [ , ... ]
  }
  [ INTO csv:// [ filename= ] <file_path> [ ;delimiter=tab ] ]
  {
    FROM <table_reference> [ [ AS ] <identifier> ]
  }
  [ WHERE <search_condition> ]
} | SCOPE_IDENTITY()
<expression> ::=
  | <column_reference>
  | @ <parameter>
  | ?
  | COUNT( * | { [ DISTINCT ] <expression> } )
  | { AVG | MAX | MIN | SUM | COUNT } ( <expression> )
  | NULLIF ( <expression> , <expression> )
  | COALESCE ( <expression> , ... )

```



```

    | CASE <expression>
      WHEN { <expression> | <search_condition> } THEN { <expression> |
NULL } [ ... ]
    [ ELSE { <expression> | NULL } ]
    END
    | <literal>
    | <sql_function>
<search_condition> ::=
{
  <expression> { = } [ <expression> ]
} [ { AND | OR } ... ]

```

## Examples

1. Return all columns:

```
SELECT * FROM Group
```

2. Rename a column:

```
SELECT "Email" AS MY_Email FROM Group
```

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

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

4. Search data:

```
SELECT * FROM Group WHERE CustomerId = 'eyt593htjgh'
```

5. The Google Directory APIs support the following operators in the WHERE clause: =.

```
SELECT * FROM Group WHERE CustomerId = 'eyt593htjgh';
```

## 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 Name, Email INTO 'csv://c:/Group.txt'
FROM 'Group' WHERE CustomerId = 'eyt593htjgh'");
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 'Group' IN
'csv://filename=c:/Group.csv;delimiter=tab' FROM 'Group' WHERE
CustomerId = 'eyt593htjgh'");
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 Group (Email) VALUES (?)";
PreparedStatement pstmt = connection.prepareStatement(cmd);
```



```
pstmt.setString(1, "cook@northwind.net");
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 Group SET Email='cook@northwind.net' 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:googledirectory:InitiateOAuth=GETANDREFRESH;",);
String cmd = "DELETE FROM Group 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;
```

## PIVOT and UNPIVOT

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

### PIVOT

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

### PIVOT Syntax

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

### UNPIVOT

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

### UNPIVOT Syntax



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

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

## Data Model

The Google Directory Adapter models Google Directory APIs as relational tables, views, and stored procedures. API limitations and requirements for the available objects are documented in this section.

### Tables

The adapter provider models the Google Directory API as relational [Tables](#).

### Views

[Views](#) offer additional information from Google Directory.

### Stored Procedures

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

## Tables

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

Generally, querying Google Directory 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 Directory Adapter Tables

Name	Description
<a href="#">ChromeOsDevices</a>	Retrieve all Chrome devices for an account.
<a href="#">DomainAliases</a>	Create, update, and query aliases of a domain.
<a href="#">Domains</a>	Create, delete, and query the domains for a user.
<a href="#">GroupAliases</a>	Create, delete, and query aliases for a group.
<a href="#">GroupMembers</a>	Create, update, delete, and query the members for a group.
<a href="#">Groups</a>	Create, update, delete, and query groups.
<a href="#">Notifications</a>	Update, delete, and query notifications for a customer.
<a href="#">OrganizationUnits</a>	Create, update, delete, and query the organization units for a customer.
<a href="#">RoleAssignments</a>	Create, delete, and query roles assigned to users.
<a href="#">Tokens</a>	Query and delete tokens for a user.
<a href="#">UserAliases</a>	Lists aliases, which are alternative email addresses for a user.
<a href="#">Users</a>	Query user information.

## ChromeOsDevices

Retrieve all Chrome devices for an account.



## Table Specific Information

### Select

To get a list of all Chrome devices for an account, the CustomerId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM ChromeOsDevices
```

### Insert

Inserts are not supported for this table.

### Update

Updates a device's annotatedUser, annotatedLocation, or notes properties. To update a notification, the following columns are required: CustomerId and Id. If not specified, the CustomerId of the current account will be used.

```
UPDATE ChromeOsDevices SET AnnotatedUser='User_2' WHERE Id = '12345' AND  
CustomerId = '1234'.
```

### Delete

Deletes are not supported for this table.

### Columns

Name	Type	ReadOnly	Description
Id [KEY]	String	True	The unique ID of the Chrome device.



SerialNumber	<i>String</i>	True	The Chrome device serial number entered when the device was enabled. This value is the same as the Admin console's Serial Number in the Chrome OS Devices tab.
Model	<i>String</i>	True	The device's model information. If the device does not have this information, this property is not included in the response.
MEID	<i>String</i>	True	The Mobile Equipment Identifier (MEID) for the 3G mobile card in a mobile device. A MEID is typically used when adding a device to a wireless carrier's post-pay service plan. If the device does not have this information, this property is not included in the response.
LastSync	<i>Datetime</i>	True	The date and time the device was last enrolled.
AnnotatedUser	<i>String</i>	False	The user of the device as noted by the administrator. Maximum length is 100 characters. Empty values are allowed.
AnnotatedLocation	<i>String</i>	False	The address or location of the device as noted by the administrator. Maximum length is 200 characters. Empty values are allowed.
AnnotatedAssetId	<i>String</i>	False	The asset identifier as noted by an administrator or specified during enrollment.
Notes	<i>String</i>	False	Notes about this device added by the administrator. This property can



			be searched with the list method's query parameter. Maximum length is 500 characters. Empty values are allowed.
OrgUnitPath	<i>String</i>	False	The full parent path with the organizational unit's name associated with the device. Path names are case insensitive. If the parent organizational unit is the top-level organization, it is represented as a forward slash, /. This property can be updated using the API
OrderNumber	<i>String</i>	True	The device's order number. Only devices directly purchased from Google have an order number.
MacAddress	<i>String</i>	True	The device's wireless MAC address. If the device does not have this information, it is not included in the response.
WillAutoRenew	<i>Boolean</i>	True	Determines if the device will auto renew its support after the support end date. This is a read-only property.
OsVersion	<i>String</i>	True	The Chrome device's operating system version.
PlatformVersion	<i>String</i>	True	The Chrome device's platform version.
FirmwareVersion	<i>String</i>	True	The Chrome device's firmware version.
BootMode	<i>String</i>	True	The boot mode for the device.



LastEnrollmentTime	<i>String</i>	True	The date and time the device was last enrolled.
TmpVersionInfoAggr	<i>String</i>	True	Trusted Platform Module (TPM).
ActiveTimeRangesAggr	<i>String</i>	True	List of active time ranges.
RecentUsersAggr	<i>String</i>	True	List of recent device users, in descending order, by last login time.
DeviceFilesAggr	<i>String</i>	True	List of device files to download.
ETag	<i>String</i>	True	ETag of the resource.

## Pseudo-Columns

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

<b>Name</b>	<b>Type</b>	<b>Description</b>
CustomerId	<i>String</i>	Id of the customer

## DomainAliases

Create, update, and query aliases of a domain.



## Table Specific Information

### Select

To get a list of all the aliases for a domain, the CustomerId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM DomainAliases
```

### Insert

To insert an alias, the following columns are required: CustomerId, ParentDomain, and DomainAliasName.

```
INSERT INTO DomainAliases (CustomerId, DomainAliasName, ParentDomain)
VALUES ('12345','Alias', 'parentdomain.com')
```

### Update

Updates are not supported for this table.

### Delete

To delete an alias, the following columns are required: CustomerId and DomainAliasName.

```
DELETE FROM DomainAliases WHERE CustomerId='C020vaw0q' AND
DomainAliasName='Alias'
```

### Columns

Name	Type	ReadOnly	Description



DomainAliasName [KEY]	<i>String</i>	False	The domain alias name.
ParentDomain	<i>String</i>	False	The parent domain name that the domain alias is associated with.
IsVerified	<i>Boolean</i>	True	Indicates the verification state of a domain alias.
CreationDate	<i>Timestamp</i>	True	Creation date timestamp of the domain alias in milliseconds.
ETag	<i>String</i>	True	ETag of the resource

## Pseudo-Columns

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

<b>Name</b>	<b>Type</b>	<b>Description</b>
CustomerId	<i>String</i>	Id of the customer

## Domains

Create, delete, and query the domains for a user.



## Table Specific Information

### Select

To get a list of all the domains, the CustomerId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the CustomerId of the current account will be used, as in the following query.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM Domains
```

### Insert

To insert a domain, the following columns are required: CustomerId and DomainName.

```
INSERT INTO Domains (CustomerId, DomainName) VALUES ('12345',  
'exampledomain.com')
```

### Update

Updates are not supported for this table.

### Delete

To delete a domain, the DomainName column is required.

```
DELETE FROM Domains WHERE DomainName='exampledomain.com'
```

## Columns

Name	Type	ReadOnly	Description
DomainName [KEY]	String	False	The domain name.



---

IsPrimary	<i>Boolean</i>	True	Indicates if the domain is a primary domain.
IsVerified	<i>Boolean</i>	True	Indicates the verification state of a domain.
CreationDate	<i>Datetime</i>	True	The creation date of the domain.
Aliases	<i>String</i>	True	The aliases of the domain.
ETag	<i>String</i>	True	ETag of the resource.

---

## 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
CustomerId	<i>String</i>	Id of the customer

---

## GroupAliases

Create, delete, and query aliases for a group.



## Table Specific Information

### Select

To get a list of all the aliases for a group, the GroupId column is required. If not specified, the GroupId of the first group from the Groups table will be used.

The following query shows the only filter processed server side by the Google Directory API:

```
SELECT * FROM GroupAliases WHERE GroupId = '12345'
```

### Insert

To insert an alias, the following columns are required: GroupId and Alias.

```
INSERT INTO GroupAliases (GroupId, Alias) VALUES ('12345', 'Alias')
```

### Update

Updates are not supported for this table.

### Delete

To delete an alias, the following columns are required: GroupId and Alias.

```
DELETE FROM GroupAliases WHERE GroupId = '12345' AND Alias = 'Alias'
```

### Columns

Name	Type	ReadOnly	Description
Alias [KEY]	String	False	The alias email address.



GroupId	<i>String</i>	True	Id of the group.
PrimaryEmail	<i>String</i>	True	PrimaryEmail of the group.
ETag	<i>String</i>	True	ETag of the resource.

## GroupMembers

Create, update, delete, and query the members for a group.

### Table Specific Information

#### Select

To get a list of all the members of a group, the GroupId column is required. If not specified, the Id of the first group from the Groups table will be used.

The following query shows the only filter processed server side by the Google Directory API:

```
SELECT * FROM GroupMembers WHERE GroupId = '12345'
```

#### Insert

To insert a member, the following columns are required: Email and GroupId. The Role column only accepts the following values: MEMBER, MANAGER, and OWNER.

```
INSERT INTO GroupMembers (Email, GroupId, Role) VALUES
('john@example.com', '12345', 'MEMBER')
```

#### Update

To update a member, the following columns are required: GroupId and Id.



```
UPDATE GroupMembers SET Role='MEMBER' WHERE GroupId='1234' AND
Id='12345'
```

## Delete

To delete a member, the following columns are required: GroupId and Id.

```
DELETE FROM GroupMembers WHERE GroupId='1234' AND Id='12345'
```

## Columns

Name	Type	ReadOnly	Description
Id [KEY]	String	False	The unique identifier for the member.
GroupId	String	True	The unique identifier for the member.
Email	String	False	The email of the member.
Role	String	False	The name of the member.
Status	String	True	The status of the member.
Type	String	True	The type of members.
ETag	String	True	ETag of the resource

## Groups



Create, update, delete, and query groups.

## Table Specific Information

### Select

To get a list of all the groups, the CustomerId is required. You can either set it in the connection string or in the WHERE clause condition. If not specified, the CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory API:

```
SELECT * FROM Groups
```

### Insert

To insert a group, the Email column is required.

```
INSERT INTO Groups (Email, Name, Description) VALUES  
('group@example.com', 'Group Example Name', 'Example Description')
```

### Update

To update a group, the Id is required.

```
UPDATE Groups SET Email = 'group@example.com', Name = 'Group',  
Description = 'Description' WHERE Id = 1231
```

### Delete

To delete a group, the Id column is required.

```
DELETE FROM Groups WHERE Id='12345'
```

### Columns



Name	Type	ReadOnly	Description
Id [KEY]	<i>String</i>	True	The unique identifier for the group.
Email	<i>String</i>	False	The email of the group.
Name	<i>String</i>	False	The name of the group.
MembersCount	<i>Long</i>	True	The number of members.
Description	<i>String</i>	False	Description of the group.
Aliases	<i>String</i>	True	Aliases of the group.
AdminCreated	<i>Boolean</i>	True	Indicates if the group was created by an admin.
ETag	<i>String</i>	True	ETag of the resource

## 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
CustomerId	<i>String</i>	The customer Id of the group.



---

Domain	<i>String</i>	Domain name.
--------	---------------	--------------

---

## Notifications

Update, delete, and query notifications for a customer.

### Table Specific Information

#### Select

To get a list of all the notifications, the CustomerId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM Notifications
```

#### Insert

Inserts are not supported for this table.

#### Update

To update a notification, the following columns are required: CustomerId and Id. If not specified, the CustomerId of the current account will be used.

```
UPDATE Notifications SET IsUnread = true WHERE Id = '12345' AND  
CustomerId = '1234'.
```

#### Delete

To delete a notification, the following columns are required: CustomerId and Id. If not specified, the CustomerId of the current account will be used.



```
DELETE FROM Notifications WHERE CustomerId = '1234' AND Id = '12345'
```

## Columns

Name	Type	ReadOnly	Description
Id [KEY]	<i>String</i>	True	Id of the notification
Subject	<i>String</i>	False	Subject of the notification
Body	<i>String</i>	True	The body of the notification
SendDate	<i>Datetime</i>	True	The date when the notification was sent
FromAddress	<i>String</i>	True	The address from which the notification is recieved
IsUnread	<i>Boolean</i>	False	Indicates wether the notification is unread or not
ETag	<i>String</i>	True	ETag of the resource

## 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
------	------	-------------



---

CustomerId	<i>String</i>	Id of the customer
------------	---------------	--------------------

---

## OrganizationUnits

Create, update, delete, and query the organization units for a customer.

### Table Specific Information

#### Select

To get a list of all the organization units, the CustomerId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM OrganizationUnits
```

#### Insert

To insert an organization unit, the following columns are required: CustomerId, Name, and ParentOrgUnitId. If not specified, the CustomerId of the current account will be used.

```
INSERT INTO OrganizationUnits (CustomerId, Name, Description,
OrgUnitPath, ParentOrgUnitId, ParentOrgUnitPath) VALUES ('12345',
'OrgUnit Name', 'OrgUnit Description', 'Path', '123456', 'ParentPath',
'1234')
```

#### Update

To update an organization unit, the following columns are required: CustomerId and Id. If not specified, the CustomerId of the current account will be used.



```
UPDATE OrganizationUnits SET Name='OrgUnit Name', 'Description =
'OrgUnit Description', OrgUnitPath = 'Path', ParentOrgUnitId = '123456',
ParentOrgUnitPath = 'ParentPath' WHERE CustomerId='1234' AND Id='12345'
```

## Delete

To delete an organization unit, the following columns are required: CustomerId and Id. If not specified, the CustomerId of the current account will be used.

```
DELETE FROM OrganizationUnits WHERE CustomerId = '1234' AND Id =
'12345'
```

## Columns

Name	Type	ReadOnly	Description
Id [KEY]	String	True	Id of the Organization Unit.
Name	String	False	Name of the Organization Unit.
Description	String	False	Description of the Organization Unit.
OrgUnitPath	String	False	Path of the OrgOrganization Unit.Unit
ParentOrgUnitPath	String	False	Path of the Organization Unit's parent.
ParentOrgUnitId	String	False	Id of the Organization Unit's parent
ETag	String	True	ETag of the resource.



## 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
CustomerId	<i>String</i>	Id of the customer

## RoleAssignments

Create, delete, and query roles assigned to users.

### Table Specific Information

#### Select

To get a list of all the roles assigned to users, the CustomerId column is required. If not specified, the CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM RoleAssignments
```

#### Insert

To assign a role to a user, the following columns are required: RoleId, UserId, ScopeType, and CustomerId. If not specified, the CustomerId of the current account will be used. ScopeType has only two acceptable values : CUSTOMER and ORG\_UNIT.

```
INSERT INTO RoleAssignments (RoleId, UserId, ScopeType) VALUES ('12345',  
'123456', 'CUSTOMER')
```



## Update

Updates are not supported for this table.

## Delete

To remove an assigned role from a user, the Id and CustomerId columns are required. If not specified, the CustomerId of the current account will be used.

```
DELETE FROM RoleAssignments WHERE Id = '12345'
```

## Columns

Name	Type	ReadOnly	Description
Id [KEY]	String	True	The unique identifier of the role assignment.
RoleId	String	False	The Id of the role that is assigned.
UserId	String	False	The Id of the user this role is assigned to.
OrgUnitId	String	False	If the role is restricted to an organizational unit, this contains the ID for the organizational unit the exercise of this role is restricted to.
ScopeType	String	False	The scope in which this role is assigned. Acceptable values are
Etag	String	True	Etag of the resource.

## 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
CustomerId	<i>String</i>	Id of the customer

## Tokens

Query and delete tokens for a user.

### Table Specific Information

#### Select

To get a current set of tokens a specified user has issued to 3rd party applications, the UserId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the Id of the first user from the Users table will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM Tokens
```

#### Insert

Inserts are not supported for this table.

#### Update

Updates are not supported for this table.

#### Delete

To delete a token, the UserId and Id columns are required.



```
DELETE FROM Tokens WHERE UserId='12345' AND Id='123456'
```

## Columns

Name	Type	ReadOnly	Description
Id [KEY]	<i>String</i>	False	The Client ID of the application the token is issued to.
UserId	<i>String</i>	False	Aggregate of child privileges.
DisplayText	<i>String</i>	False	The displayable name of the application the token is issued to.
IsAnonymous	<i>Boolean</i>	False	Indicates if the name of the privilege.
IsNativeApp	<i>Boolean</i>	False	Indicates if the token is issued to an installed application.
ScopesAggregate	<i>String</i>	False	Aggregate of child privileges.
Etag	<i>String</i>	False	Etag of the resource.

## UserAliases

Lists aliases, which are alternative email addresses for a user.



## Table Specific Information

### Select

To get a list of all the aliases for a user, the UserId column is required. It can be set in the connection string or in the WHERE clause condition. Otherwise, the adapter will automatically use the Id of the first user from the Users table.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM Aliases
```

### Insert

To insert an alias, the following columns are required: UserId and Alias.

```
INSERT INTO Aliases (UserId, Alias) VALUES ('12345', 'Alias')
```

### Update

Updates are not supported for this table.

### Delete

To delete an alias, the following columns are required: UserId and Alias.

```
DELETE FROM Aliases WHERE Id = '12345' AND Alias = 'Alias'
```

### Columns

Name	Type	ReadOnly	Description
Alias [KEY]	String	False	The alias email address.



UserId	<i>String</i>	True	Id of the user.
PrimaryEmail	<i>String</i>	True	PrimaryEmail of the user.
ETag	<i>String</i>	True	ETag of the resource.

## Users

Query user information.

### Table Specific Information

#### Select

To get a list of all the users, CustomerId is required. You can either set it in the connection string or in the WHERE clause condition. If not specified, the CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM Users
```

#### Insert

To insert a user, the following columns are required: PrimaryEmail, FirstName, Surname, and Password.

```
INSERT INTO Users (PrimaryEmail, FirstName, Surname, Password,
Suspended) VALUES ('john@example.com', 'John', 'Doe', '12345', true)
```

#### Update

To update a user, the Id column is required.



```
UPDATE Users SET PrimaryEmail = 'john@example.com', FirstName = 'John' ,
Surname = 'Doe', Suspended = true WHERE Id = 1231
```

## Delete

To delete users, the Id column is required.

```
DELETE FROM Users WHERE Id = '12345'
```

## Columns

Name	Type	ReadOnly	Description
Id [KEY]	<i>String</i>	True	The unique identifier for the user.
CustomerId	<i>String</i>	True	The customer Id of the user.
PrimaryEmail	<i>String</i>	False	The primary email of the user.
FirstName	<i>String</i>	False	The first name of the user.
Surname	<i>String</i>	False	The surname of the user.
Aliases	<i>String</i>	True	The aliases of the user.
IsAdmin	<i>Boolean</i>	True	Indicates if the user is an admin.
IsDelegatedAdmin	<i>Boolean</i>	True	Indicates if the user is a delegated admin.



LastLoginDate	<i>Datetime</i>	True	Last time the user logged on.
CreationDate	<i>Datetime</i>	True	Creation date of the user.
DeletionDate	<i>Datetime</i>	True	Deletion date of the user.
AgreedToTerms	<i>Boolean</i>	True	Indicates if the user agreed to the terms or not.
Suspended	<i>Boolean</i>	False	Indicates if the user got suspended.
SuspensionReason	<i>String</i>	True	The reason the user got suspended.
OrgUnitPath	<i>String</i>	False	The full path of the parent organization associated with the user. If the parent organization is the top-level, it is represented as a forward slash (/).
IsMailBoxSetup	<i>Boolean</i>	True	Indicates if the user's Google mailbox is created. This property is only applicable if the user has been assigned a Gmail license.
IsEnrolledIn2Sv	<i>Boolean</i>	True	Indicates if the user is enrolled in 2-step verification.
IsEnforcedIn2Sv	<i>Boolean</i>	True	Indicates if the user is enforced in 2-step verification.
IncludeInGlobalAddressList	<i>Boolean</i>	True	Indicates if the user's profile is visible in the G Suite global address list when the contact



			sharing feature is enabled for the domain.
ThumbnailPhotoUrl	<i>String</i>	True	Photo Url of the user

## Pseudo-Columns

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

<b>Name</b>	<b>Type</b>	<b>Description</b>
Password	<i>String</i>	The password of the user.
Domain	<i>String</i>	Domain name

## Views

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

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

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

## Google Directory Adapter Views



Name	Description
<a href="#">AppSpecificPasswords</a>	Lists all Application Specific Passwords (passwords that are used with applications that do not accept verification codes) issued by a user.
<a href="#">MobileDevices</a>	Lists mobile devices for an account.
<a href="#">Privileges</a>	Lists all Privileges.
<a href="#">Roles</a>	Lists roles in a domain.
<a href="#">UserAddresses</a>	Lists the addresses for a user.
<a href="#">UserEmails</a>	Query the emails for a user.
<a href="#">UserInstantMessagingAccounts</a>	Query the IM accounts for a user.
<a href="#">UserLocations</a>	Query the locations for a user.
<a href="#">UserOrganizations</a>	Query the organizations for a user.
<a href="#">UserPhones</a>	Query the phone numbers for a user.
<a href="#">UserWebsites</a>	Retrieve a list of the websites of a user.
<a href="#">VerificationCodes</a>	Query verification codes for a user.

## AppSpecificPasswords

Lists all Application Specific Passwords (passwords that are used with applications that do not accept verification codes) issued by a user.



## Table Specific Information

### Select

To get a list of all the application specific tokens issued by a user, the UserId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the Id of the first user from the Users table will be used.

For example, the following query is processed server side by the Google Directory API:

```
SELECT * FROM AppSpecificPasswords
```

### Columns

Name	Type	Description
Id [KEY]	String	The unique identifier of the ASP.
UserId	String	The unique identifier of the user who issued the ASP.
Name	String	Name of the ASP.
CreationDate	Datetime	The date when the ASP was created.
LastTimeUsed	Datetime	The time when the ASP was last used.
Etag	String	Etag of the resource.

## MobileDevices

Lists mobile devices for an account.



## Table Specific Information

### Select

To get a list of all mobile devices for an account, the CustomerId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM MobileDevices
```

### Columns

Name	Type	Description
Id [KEY]	String	The serial number for a Google Sync mobile device. For Android and iOS devices, this is a software generated unique identifier.
ResourceId	String	The unique ID the API service uses to identify the mobile device.
Name	String	List of the owner's usernames.
AccountsList	String	List of accounts added on device.
Email	String	List of the owner's emails.
Model	String	The mobile device's model name.
OS	String	The mobile device's operating system



Type	<i>String</i>	The type of mobile device.
Status	<i>String</i>	The device's status.
HardwareId	<i>String</i>	The IMEI/MEID unique identifier for Android hardware.
FirstSyncDate	<i>Datetime</i>	The date and time the device was initially synchronized with the policy settings in the Admin console.
LastSyncDate	<i>Datetime</i>	The date and time the device was last synchronized with the policy settings in the Admin console.
UserAgent	<i>String</i>	Gives information about the device such as os version.
SerialNumber	<i>String</i>	The device's serial number.
IMEI	<i>String</i>	The device's IMEI number.
MEID	<i>String</i>	The device's MEID number.
WiFiMacAddress	<i>String</i>	The device's MAC address on Wi-Fi networks.
NetworkOperator	<i>String</i>	Mobile Device mobile or network operator.
DefaultLanguage	<i>String</i>	The default language used on the device.



DeviceCompromisedStatus	<i>String</i>	The compromised device status.
BuildNumber	<i>String</i>	The device's operating system build number.
KernelVersion	<i>String</i>	The device's kernel version.
BasebandVersion	<i>String</i>	The device's baseband version.
Manufacturer	<i>String</i>	The device's manufacturer.
ReleaseVersion	<i>String</i>	Mobile Device release version version.
SecurityPatchLevel	<i>String</i>	The device's security patch level.
Brand	<i>String</i>	The device's brand.
BootloaderVersion	<i>String</i>	The device's bootloader version.
Hardware	<i>String</i>	The device's hardware.
EncryptionStatus	<i>String</i>	The device's encryption status.
DevicePasswordStatus	<i>String</i>	The device's password status
Privilege	<i>String</i>	DMAgentPermission.
UnknownSourcesStatus	<i>Boolean</i>	Indicates if unknown sources are enabled or



		disabled on device
AdbStatus	<i>Boolean</i>	Indicates if adb(USB debugging) is enabled or disabled on device
IsOnOwnerProfile	<i>Boolean</i>	Indicates if this account is on owner/primary profile or not.
SupportsWorkProfile	<i>Boolean</i>	Indicates if work profile is supported on device.
Etag	<i>String</i>	Etag of the resource

## Pseudo-Columns

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

<b>Name</b>	<b>Type</b>	<b>Description</b>
CustomerId	<i>String</i>	The Id of the customer

## Privileges

Lists all Privileges.

## Table Specific Information

### Select

To get a list of all privileges for an account, the CustomerId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the



CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM Privileges
```

## Columns

Name	Type	Description
ServiceId [KEY]	String	The obfuscated ID of the service this privilege is for.
ServiceName [KEY]	String	The name of the service this privilege is for.
PrivilegeName	String	The name of the privilege.
ParentServiceId	String	The service Id of the parent privilege.
ParentPrivilegeName	String	The privilege name of the parent privilege.
IsOrganizationUnitRestrictable	Boolean	Indicates if the privilege can be restricted to an organization unit.
Etag	String	Etag of the resource.

## 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
CustomerId	<i>String</i>	Id of the customer

## Roles

Lists roles in a domain.

### Table Specific Information

#### Select

To get a list of all the roles, the CustomerId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the CustomerId of the current account will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM Roles
```

#### Columns

Name	Type	Description
Id [KEY]	<i>String</i>	The unique identifier for the role.
Name	<i>String</i>	Name of the role.
Description	<i>String</i>	A short description of the role.



PrivilegeName	<i>String</i>	The name of the privilege.
ServiceId	<i>String</i>	The ID of the service the privilege is for.
IsSystemRole	<i>Boolean</i>	Indicates if it is a pre-defined system role.
IsSuperAdminRole	<i>Boolean</i>	Indicates if the role is a super admin role.
Etag	<i>String</i>	Etag of the resource.

## Pseudo-Columns

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

<b>Name</b>	<b>Type</b>	<b>Description</b>
CustomerId	<i>String</i>	Id of the customer

## UserAddresses

Lists the addresses for a user.



## Table Specific Information

### Select

To get a list of addresses for a user, the `UserId` column is required. It can be set in the connection string or in the `WHERE` clause condition. Otherwise, the adapter will automatically use the `UserId` of the first user from the `Users` table.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM UserAddresses
```

### Columns

Name	Type	Description
UserId	<i>String</i>	The unique identifier for the user.
Type	<i>String</i>	The address type.
CustomType	<i>String</i>	The custom type of the address.
FormattedAddress	<i>String</i>	The full unstructured postal address.
PoBox	<i>String</i>	Post office box of the address.
ExtendedAddress	<i>String</i>	The extended address
StreetAddress	<i>String</i>	The street address
Locality	<i>String</i>	The town or city of the address.



Region	<i>String</i>	The abbreviated province or state.
PostalCode	<i>String</i>	The ZIP or postal code, if applicable.
Country	<i>String</i>	Country in the address.
CountryCode	<i>String</i>	The country code. Uses the ISO 3166-1 standard.
IsPrimary	<i>Boolean</i>	Indicates if this is the primary address of the user

## UserEmails

Query the emails for a user.

### Table Specific Information

#### Select

To get a list of email addresses for a user, the `UserId` column is required. It can be set in the connection string or in the `WHERE` clause condition. Otherwise, the adapter will automatically use the `Id` of the first user from the `Users` table.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM UserEmails
```

#### Columns



Name	Type	Description
Address	<i>String</i>	The user's email address
UserId	<i>String</i>	The unique identifier for the user.
IsPrimary	<i>String</i>	Indicates if this is the user's primary email.
CustomType	<i>String</i>	The custom type of the email.
Type	<i>String</i>	The type of the email account.

## UserInstantMessagingAccounts

Query the IM accounts for a user.

### Table Specific Information

#### Select

To get a list of IM accounts for a user, the UserId column is required. It can be set in the connection string or in the WHERE clause condition. Otherwise, the adapter will automatically use the Id of the first user from the Users table.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM UserInstantMessagingAccounts
```

#### Columns



Name	Type	Description
IM	<i>String</i>	The user's IM network ID.
UserId	<i>String</i>	The unique identifier for the user.
Protocol	<i>String</i>	The IM protocol identifies the IM network.
CustomProtocol	<i>String</i>	The custom type of the IM protocol.
IsPrimary	<i>String</i>	Indicates if this is the user's primary IM.
CustomType	<i>String</i>	The custom type of the IM account.
Type	<i>String</i>	The type of the IM account.

## UserLocations

Query the locations for a user.

### Table Specific Information

#### Select

To get a list of locations for a user, the UserId column is required. It can be set in the connection string or in the WHERE clause condition. Otherwise, the adapter will automatically use the Id of the first user from the Users table.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM UserLocations
```



## Columns

Name	Type	Description
Area	<i>String</i>	Textual location of the user.
UserId	<i>String</i>	The unique identifier for the user.
BuildingId	<i>String</i>	The building identifier.
DeskCode	<i>String</i>	The desk location.
FloorName	<i>String</i>	The floor name/number
FloorSection	<i>String</i>	The floor section.
CustomType	<i>String</i>	The custom type of the location.
Type	<i>String</i>	The type of the location.

## UserOrganizations

Query the organizations for a user.



## Table Specific Information

### Select

To get a list of organizations for a user, the UserId column is required. It can be set in the connection string or in the WHERE clause condition. Otherwise, the adapter will automatically use the Id of the first user from the Users table.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM UserOrganizations
```

### Columns

Name	Type	Description
Name	<i>String</i>	The name of the organization.
UserId	<i>String</i>	The unique identifier for the user.
OrganizationDomain	<i>String</i>	The domain the organization belongs to.
Department	<i>String</i>	Specifies the department within the organization.
Description	<i>String</i>	The description of the organization.
Title	<i>String</i>	The user's title within the organization.
CostCenter	<i>String</i>	The cost center of the user's organization.
Location	<i>String</i>	The physical location of the organization.



IsPrimary	<i>Boolean</i>	Indicates if this is the user's primary organization.
Symbol	<i>String</i>	Text string symbol of the organization.
Type	<i>String</i>	Country in the address.
CustomType	<i>String</i>	If the value of type is custom, this property contains the custom type.

## UserPhones

Query the phone numbers for a user.

### Table Specific Information

#### Select

To get a list of phones for a user, the UserId column is required. It can be set in the connection string or in the WHERE clause condition. Otherwise, the adapter will automatically use the Id of the first user from the Users table.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM UserPhones
```

#### Columns

Name	Type	Description
PhoneNumber	<i>String</i>	The user's phone number.



UserId	<i>String</i>	The unique identifier for the user.
IsPrimary	<i>String</i>	Indicates if this is the user's primary IM.
CustomType	<i>String</i>	The custom type of the phone number.
Type	<i>String</i>	The type of the phone number.

## UserWebsites

Retrieve a list of the websites of a user.

### Table Specific Information

#### Select

To get a list of websites for a user, the UserId column is required. It can be set in the connection string or in the WHERE clause condition. Otherwise, the adapter will automatically use the Id of the first user from the Users table.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM UserWebsites
```

#### Columns

Name	Type	Description
URL	<i>String</i>	The URL of the website.



UserId	<i>String</i>	The unique identifier for the user.
CustomType	<i>String</i>	The custom type of the website.
Type	<i>String</i>	The type of the website.
IsPrimary	<i>Boolean</i>	Indicates if this is the user's primary website or not

## VerificationCodes

Query verification codes for a user.

### Table Specific Information

#### Select

To get a current set of valid backup verification codes for a specified user, the UserId column is required. It can be set in the connection string or in the WHERE clause condition. If not specified, the Id of the first user from the Users table will be used.

For example, the following query is processed server side by the Google Directory APIs:

```
SELECT * FROM VerificationCodes
```

#### Columns

Name	Type	Description
UserId	<i>String</i>	The unique ID of the user.



---

VerificationCode	String	A current verification code for the user.
Etag	String	Etag of the resource.

---

## Stored Procedures

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

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

### Google Directory Adapter Stored Procedures

Name	Description
<a href="#">ChangeAdminStatus</a>	Change the admin status of a user.
<a href="#">GenerateVerificationCodes</a>	Generate verification codes for a user.
<a href="#">GetOAuthAccessToken</a>	Obtains the OAuth access token to be used for authentication with various Google services.
<a href="#">GetOAuthAuthorizationURL</a>	Obtains the OAuth authorization URL used for authentication with various Google services.
<a href="#">InvalidateVerificationCodes</a>	Invalidate verification codes for a user.
<a href="#">RefreshOAuthAccessToken</a>	Obtains the OAuth access token to be used for authentication with various Google services.



## ChangeAdminStatus

Change the admin status of a user.

### Input

Name	Type	Description
UserId	<i>String</i>	Id of the user. The value can be the user's primary email address, alias email address, or unique user ID.
Status	<i>String</i>	New admin status of the user.

## GenerateVerificationCodes

Generate verification codes for a user.

### Input

Name	Type	Description
UserId	<i>String</i>	Id of the user.

## GetOAuthAccessToken

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

### Input



Name	Type	Description
AuthMode	String	<p>The type of authentication mode to use.</p> <p>The allowed values are <i>APP</i>, <i>WEB</i>.</p> <p>The default value is <i>WEB</i>.</p>
Verifier	String	The verifier code returned by Google after permission for the app to connect has been granted. WEB AuthMode only.
Scope	String	<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</p> <p><a href="https://www.googleapis.com/auth/admin.directory.rolemanagement.readonly">https://www.googleapis.com/auth/admin.directory.rolemanagement.readonly</a></p> <p><a href="https://www.googleapis.com/auth/admin.directory.domain.readonly">https://www.googleapis.com/auth/admin.directory.domain.readonly</a></p> <p><a href="https://www.googleapis.com/auth/admin.directory.user.readonly">https://www.googleapis.com/auth/admin.directory.user.readonly</a></p> <p><a href="https://www.googleapis.com/auth/admin.directory.user.alias.readonly">https://www.googleapis.com/auth/admin.directory.user.alias.readonly</a></p> <p><a href="https://www.googleapis.com/auth/admin.directory.group.member.readonly">https://www.googleapis.com/auth/admin.directory.group.member.readonly</a></p> <p><a href="https://www.googleapis.com/auth/admin.directory.group.readonly">https://www.googleapis.com/auth/admin.directory.group.readonly</a></p> <p><a href="https://www.googleapis.com/auth/admin.directory.orgunit.readonly">https://www.googleapis.com/auth/admin.directory.orgunit.readonly</a></p> <p><a href="https://www.googleapis.com/auth/admin.directory.device.chromeos.readonly">https://www.googleapis.com/auth/admin.directory.device.chromeos.readonly</a></p> <p><a href="https://www.googleapis.com/auth/admin.directory.device.mobile.readonly">https://www.googleapis.com/auth/admin.directory.device.mobile.readonly</a>.</p>
CallbackURL	String	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 ('/').
Prompt	String	This field indicates the prompt to present the user. It accepts one of the following values: NONE, CONSENT, SELECT ACCOUNT. The default is SELECT_ACCOUNT, so a given user will be prompted to select the account to connect to. If it is set to CONSENT, the user will see a consent page every time, even if they have previously given consent to the application for a given set of scopes. Lastly, if



		<p>it is set to <code>NONE</code>, no authentication or consent screens will be displayed to the user.</p> <p>The default value is <code>SELECT_ACCOUNT</code>.</p>
<code>AccessType</code>	<i>String</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 <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 <code>ONLINE</code>, <code>OFFLINE</code>.</p> <p>The default value is <code>OFFLINE</code>.</p>
<code>State</code>	<i>String</i>	<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
<code>OAuthAccessToken</code>	<i>String</i>	The authentication token returned from Google. This can be used in subsequent calls to other operations for this particular service.
<code>OAuthRefreshToken</code>	<i>String</i>	A token that may be used to obtain a new access token.
<code>ExpiresIn</code>	<i>String</i>	The remaining lifetime on the access token.



## GetOAuthAuthorizationURL

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

### Input

Name	Type	Description
Scope	<i>String</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</p> <p><code>https://www.googleapis.com/auth/admin.directory.rolemanagement.readonly</code></p> <p><code>https://www.googleapis.com/auth/admin.directory.domain.readonly</code></p> <p><code>https://www.googleapis.com/auth/admin.directory.user.readonly</code></p> <p><code>https://www.googleapis.com/auth/admin.directory.user.alias.readonly</code></p> <p><code>https://www.googleapis.com/auth/admin.directory.group.member.readonly</code></p> <p><code>https://www.googleapis.com/auth/admin.directory.group.readonly</code></p> <p><code>https://www.googleapis.com/auth/admin.directory.orgunit.readonly</code></p> <p><code>https://www.googleapis.com/auth/admin.directory.device.chromeos.readonly</code></p> <p><code>https://www.googleapis.com/auth/admin.directory.device.mobile.readonly</code>.</p>
CallbackURL	<i>String</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>	<p>This field indicates the prompt to present the user. It accepts one of the following values: NONE, CONSENT, SELECT ACCOUNT. The default is SELECT_ACCOUNT, so a given user will be prompted to select the account to connect to. If it is set to CONSENT, the user will see a consent page every time, even if they have previously given consent to the application for a given set of scopes. Lastly, if it is set to NONE, no authentication or consent screens will be</p>



displayed to the user.

The default value is *SELECT\_ACCOUNT*.

AccessType	String	<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 OFFLINE. If your application needs to refresh access tokens when the user is not present at the browser, then use OFFLINE. 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	<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 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.</p>

## Result Set Columns

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

## InvalidateVerificationCodes

Invalidate verification codes for a user.

### Input



Name	Type	Description
UserId	<i>String</i>	Id of the user.

## RefreshOAuthAccessToken

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

### Input

Name	Type	Description
OAuthRefreshToken	<i>String</i>	The refresh token returned from the original authorization code exchange.

### Result Set Columns

Name	Type	Description
OAuthAccessToken	<i>String</i>	The authentication token returned from Google. This can be used in subsequent calls to other operations for this particular service.
OAuthRefreshToken	<i>String</i>	The authentication token returned from Google. This can be used in subsequent calls to other operations for this particular service.
ExpiresIn	<i>String</i>	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.

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

## Authentication

---

Property	Description
<a href="#">AuthScheme</a>	The type of authentication to use when connecting to Google Directory.

## OAuth

---

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

---



	settings.
<a href="#">Scope</a>	Specify scope to obtain the initial access and refresh token.
<a href="#">OAuthVerifier</a>	The verifier code returned from the OAuth authorization URL.
<a href="#">OAuthRefreshToken</a>	The OAuth refresh token for the corresponding OAuth access token.
<a href="#">OAuthExpiresIn</a>	The lifetime in seconds of the OAuth AccessToken.
<a href="#">OAuthTokenTimestamp</a>	The Unix epoch timestamp in milliseconds when the current Access Token was created.

## JWT OAuth

Property	Description
<a href="#">OAuthJWTCert</a>	The JWT Certificate store.
<a href="#">OAuthJWTCertType</a>	The type of key store containing the JWT Certificate.
<a href="#">OAuthJWTCertPassword</a>	The password for the OAuth JWT certificate.
<a href="#">OAuthJWTCertSubject</a>	The subject of the OAuth JWT certificate.
<a href="#">OAuthJWTIssuer</a>	The issuer of the Java Web Token.
<a href="#">OAuthJWTSubject</a>	The user subject for which the application is requesting delegated access.

## SSL

Property	Description



---

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

---

## Firewall

---

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

---

## Proxy

---

Property	Description
<a href="#">ProxyAutoDetect</a>	This indicates whether to use the system proxy settings or not. This takes precedence over other proxy settings, so you'll need to set ProxyAutoDetect to FALSE in order use custom proxy settings.
<a href="#">ProxyServer</a>	The hostname or IP address of a proxy to route HTTP traffic through.
<a href="#">ProxyPort</a>	The TCP port the ProxyServer proxy is running on.
<a href="#">ProxyAuthScheme</a>	The authentication type to use to authenticate to the ProxyServer proxy.
<a href="#">ProxyUser</a>	A user name to be used to authenticate to the ProxyServer proxy.

---



<a href="#">ProxyPassword</a>	A password to be used to authenticate to the ProxyServer proxy.
<a href="#">ProxySSLType</a>	The SSL type to use when connecting to the ProxyServer proxy.
<a href="#">ProxyExceptions</a>	A semicolon separated list of destination hostnames or IPs that are exempt from connecting through the ProxyServer .

## Logging

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

## Schema

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

## Miscellaneous

Property	Description
<a href="#">CustomerId</a>	Restrict query results to this customer.
<a href="#">Domain</a>	Restrict queries to this domain.
<a href="#">GroupId</a>	Restrict query results to this group.
<a href="#">MaxRows</a>	Limits the number of rows returned rows when no aggregation or group by is used in the query. This helps avoid performance issues at



	design time.
<a href="#">Other</a>	These hidden properties are used only in specific use cases.
<a href="#">Pagesize</a>	The maximum number of results to return per page from Google Directory.
<a href="#">Readonly</a>	You can use this property to enforce read-only access to Google Directory from the provider.
<a href="#">Timeout</a>	The value in seconds until the timeout error is thrown, canceling the operation.
<a href="#">UserDefinedViews</a>	A filepath pointing to the JSON configuration file containing your custom views.
<a href="#">UserId</a>	Restrict query results to this user.

## Authentication

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

Property	Description
<a href="#">AuthScheme</a>	The type of authentication to use when connecting to Google Directory.

## AuthScheme

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

### Possible Values

Auto, OAuth, OAuthJWT, GCPIInstanceAccount



## 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.

## OAuth

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

Property	Description
<a href="#">InitiateOAuth</a>	Set this property to initiate the process to obtain or refresh the OAuth access token when you connect.
<a href="#">OAuthClientId</a>	The client Id assigned when you register your application with an OAuth authorization server.
<a href="#">OAuthClientSecret</a>	The client secret assigned when you register your application with an OAuth authorization server.
<a href="#">OAuthAccessToken</a>	The access token for connecting using OAuth.



<a href="#">OAuthSettingsLocation</a>	The location of the settings file where OAuth values are saved when InitiateOAuth is set to GETANDREFRESH or REFRESH. Alternatively, this can be held in memory by specifying a value starting with memory://.
<a href="#">CallbackURL</a>	The OAuth callback URL to return to when authenticating. This value must match the callback URL you specify in your app settings.
<a href="#">Scope</a>	Specify scope to obtain the initial access and refresh token.
<a href="#">OAuthVerifier</a>	The verifier code returned from the OAuth authorization URL.
<a href="#">OAuthRefreshToken</a>	The OAuth refresh token for the corresponding OAuth access token.
<a href="#">OAuthExpiresIn</a>	The lifetime in seconds of the OAuth AccessToken.
<a href="#">OAuthTokenTimestamp</a>	The Unix epoch timestamp in milliseconds when the current Access Token was created.

## InitiateOAuth

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

### Possible Values

OFF, GETANDREFRESH, REFRESH

### Data Type

string

### Default Value

"OFF"



## Remarks

The following options are available:

1. **OFF:** Indicates that the OAuth flow will be handled entirely by the user. An OAuthAccessToken will be required to authenticate.
2. **GETANDREFRESH:** Indicates that the entire OAuth Flow will be handled by the adapter. If no token currently exists, it will be obtained by prompting the user via the browser. If a token exists, it will be refreshed when applicable.
3. **REFRESH:** Indicates that the adapter will only handle refreshing the OAuthAccessToken. The user will never be prompted by the adapter to authenticate via the browser. The user must handle obtaining the OAuthAccessToken and OAuthRefreshToken initially.

## OAuthClientId

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

### Data Type

string

### Default Value

""

## Remarks

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

## OAuthClientSecret

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



## Data Type

string

## Default Value

""

## Remarks

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

## OAuthAccessToken

The access token for connecting using OAuth.

## Data Type

string

## Default Value

""

## Remarks

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

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

## OAuthSettingsLocation

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



value starting with memory://.

## Data Type

string

## Default Value

"%APPDATA%\CDATA\GoogleDirectory 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\GoogleDirectory Data Provider\OAuthSettings.txt" with %**APPDATA**% being set to the user's configuration directory:

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

## CallbackURL



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

## Data Type

string

## Default Value

""

## Remarks

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

## Scope

Specify scope to obtain the initial access and refresh token.

## Data Type

string

## Default Value

""

## Remarks

Specify scope to obtain the initial access and refresh token.

## OAuthVerifier

The verifier code returned from the OAuth authorization URL.



## Data Type

string

## Default Value

""

## Remarks

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

## Authentication on Headless Machines

See to obtain the [OAuthVerifier](#) value.

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

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

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

## OAuthRefreshToken

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.

## OAuthExpiresIn

The lifetime in seconds of the OAuth AccessToken.

## Data Type

string

## Default Value

""

## Remarks

Pair with `OAuthTokenTimestamp` to determine when the AccessToken will expire.

## OAuthTokenTimestamp

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

## Data Type

string

## Default Value

""

## Remarks

Pair with `OAuthExpiresIn` to determine when the AccessToken will expire.



## JWT OAuth

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

Property	Description
<a href="#">OAuthJWTCert</a>	The JWT Certificate store.
<a href="#">OAuthJWTCertType</a>	The type of key store containing the JWT Certificate.
<a href="#">OAuthJWTCertPassword</a>	The password for the OAuth JWT certificate.
<a href="#">OAuthJWTCertSubject</a>	The subject of the OAuth JWT certificate.
<a href="#">OAuthJWTIssuer</a>	The issuer of the Java Web Token.
<a href="#">OAuthJWTSubject</a>	The user subject for which the application is requesting delegated access.

### OAuthJWTCert

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).

## OAuthJWTCertType

The type of key store containing the JWT Certificate.

### Possible Values

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



## Data Type

string

## Default Value

"USER"

## Remarks

This property can take one of the following values:

USER	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.
GOOGLEJSONBLOB	The certificate store is a string that contains the service account JSON. Only valid when connecting to a Google service.

## OAuthJWTCertPassword

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.

## OAuthJWTCertSubject

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.

## OAuthJWTIssuer

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.

## OAuthJWTSubject



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.

# SSL

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

---

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

---

## SSLServerCert

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

## Data Type

string



## Default Value

""

## Remarks

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

This property can take the following forms:

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

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

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

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

## Firewall



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

---

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

---

## FirewallType

The protocol used by a proxy-based firewall.

### Possible Values

NONE, TUNNEL, SOCKS4, SOCKS5

### Data Type

string

### Default Value

"NONE"

### Remarks

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



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

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

## FirewallServer

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

### Data Type

string

### Default Value

""

### Remarks

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



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

## FirewallPort

The TCP port for a proxy-based firewall.

### Data Type

int

### Default Value

0

### Remarks

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

## FirewallUser

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

### Data Type

string

### Default Value

""

### Remarks

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



## FirewallPassword

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

### Data Type

string

### Default Value

""

### Remarks

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

## Proxy

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

Property	Description
<a href="#">ProxyAutoDetect</a>	This indicates whether to use the system proxy settings or not. This takes precedence over other proxy settings, so you'll need to set ProxyAutoDetect to FALSE in order use custom proxy settings.
<a href="#">ProxyServer</a>	The hostname or IP address of a proxy to route HTTP traffic through.
<a href="#">ProxyPort</a>	The TCP port the ProxyServer proxy is running on.
<a href="#">ProxyAuthScheme</a>	The authentication type to use to authenticate to the ProxyServer proxy.
<a href="#">ProxyUser</a>	A user name to be used to authenticate to the ProxyServer proxy.



<a href="#">ProxyPassword</a>	A password to be used to authenticate to the ProxyServer proxy.
<a href="#">ProxySSLType</a>	The SSL type to use when connecting to the ProxyServer proxy.
<a href="#">ProxyExceptions</a>	A semicolon separated list of destination hostnames or IPs that are exempt from connecting through the ProxyServer .

## ProxyAutoDetect

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

### Data Type

bool

### Default Value

true

### Remarks

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

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

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



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

## ProxyServer

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

### Data Type

string

### Default Value

""

### Remarks

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

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

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

## ProxyPort

The TCP port the ProxyServer proxy is running on.

### Data Type

int

### Default Value

80



## Remarks

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

## ProxyAuthScheme

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

### Possible Values

BASIC, DIGEST, NONE, NEGOTIATE, NTLM, PROPRIETARY

### Data Type

string

### Default Value

"BASIC"

## Remarks

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

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

The authentication type can be one of the following:

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



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

## ProxyUser

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

### Data Type

string

### Default Value

""

### Remarks

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

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

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

## ProxyPassword

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

### Data Type

string



## Default Value

""

## Remarks

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

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

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

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

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

## ProxySSLType

The SSL type to use when connecting to the [ProxyServer](#) proxy.

## Possible Values

AUTO, ALWAYS, NEVER, TUNNEL

## Data Type

string

## Default Value

"AUTO"

## Remarks

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



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

## ProxyExceptions

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

### Data Type

string

### Default Value

""

### Remarks

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

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

## Logging



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

---

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

---

## LogModules

Core modules to be included in the log file.

### Data Type

string

### Default Value

""

### Remarks

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

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

## Schema

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

---

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

---



## Location

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

## Data Type

string

## Default Value

"%APPDATA%\\CData\\GoogleDirectory 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\\GoogleDirectory Data Provider\\Schema" with **%APPDATA%** being set to the user's configuration directory:

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

## Miscellaneous

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



Property	Description
<a href="#">CustomerId</a>	Restrict query results to this customer.
<a href="#">Domain</a>	Restrict queries to this domain.
<a href="#">GroupId</a>	Restrict query results to this group.
<a href="#">MaxRows</a>	Limits the number of rows returned rows when no aggregation or group by is used in the query. This helps avoid performance issues at design time.
<a href="#">Other</a>	These hidden properties are used only in specific use cases.
<a href="#">Pagesize</a>	The maximum number of results to return per page from Google Directory.
<a href="#">Readonly</a>	You can use this property to enforce read-only access to Google Directory from the provider.
<a href="#">Timeout</a>	The value in seconds until the timeout error is thrown, canceling the operation.
<a href="#">UserDefinedViews</a>	A filepath pointing to the JSON configuration file containing your custom views.
<a href="#">UserId</a>	Restrict query results to this user.

## CustomerId

Restrict query results to this customer.

### Data Type

string

### Default Value

"my\_customer"



## Remarks

This property can be set in the connection string or query. Otherwise, the adapter will use the Customer Id of the authenticated user. You can also get this value from the Users table.

## Domain

Restrict queries to this domain.

## Data Type

string

## Default Value

""

## Remarks

The domain name (e.g., cdata.com). Use this connection property to get results from only one domain.

## GroupId

Restrict query results to this group.

## Data Type

string

## Default Value

""

## Remarks

This property must be set in the connection string or query. Otherwise, the adapter will use the first found Group Id. You can get this value from the Groups table.



## MaxRows

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

### Data Type

int

### Default Value

-1

### Remarks

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

## Other

These hidden properties are used only in specific use cases.

### Data Type

string

### Default Value

""

### Remarks

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

Specify multiple properties in a semicolon-separated list.

## Integration and Formatting



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

## Pagesize

The maximum number of results to return per page from Google Directory.

### Data Type

int

### Default Value

100

### Remarks

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

## Readonly

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

## Timeout

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

## Data Type

int

## Default Value

60

## Remarks

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

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

## UserDefinedViews

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

## Data Type

string

## Default Value

""



## Remarks

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

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

This User Defined View configuration file is formatted as follows:

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

For example:

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

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

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

## Userld

Restrict query results to this user.

## Data Type

string

## Default Value

""



## Remarks

The Id of the user. If not specified, the first user from the Users table will be used.



# TIBCO Product Documentation and Support Services

---

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

## How to Access TIBCO Documentation

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

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

## Product-Specific Documentation

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

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



TDV Adapter Guides

TDV Data Source Toolkit Guide (Formerly Extensibility Guide)

- **References**

TDV Reference Guide

TDV Application Programming Interface Guide

- **Other**

TDV Business Directory Guide

TDV Discovery Guide

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

## How to Contact TIBCO Support

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

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

## Release Version Support

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

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



## How to Join TIBCO Community

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



# Legal and Third-Party Notices

---

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

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

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

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

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

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

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



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

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

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

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

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

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