



Tibco Data Virtualization[®]

Google Contact Adapter Guide

Version 8.7.0
April 2023



Contents

Contents	2
Google Contacts Adapter	4
Getting Started	4
Basic Tab	5
Logging	10
Creating a Custom OAuth App	12
Changelog	14
Advanced Features	18
User Defined Views	19
Inserting Parent and Child Records	22
SSL Configuration	26
Firewall and Proxy	26
Query Processing	27
Logging	28
SQL Compliance	31
SELECT Statements	32
SELECT INTO Statements	34
INSERT Statements	35
UPDATE Statements	35
DELETE Statements	36
EXECUTE Statements	37
PIVOT and UNPIVOT	37
Data Model	39
Tables	39
Views	62
Stored Procedures	79
Connection String Options	89

Authentication	93
OAuth	94
JWT OAuth	102
SSL	108
Firewall	109
Proxy	113
Logging	119
Schema	120
Miscellaneous	121
TIBCO Product Documentation and Support Services	126
How to Access TIBCO Documentation	126
How to Contact TIBCO Support	127
Release Version Support	127
How to Join TIBCO Community	128
Legal and Third-Party Notices	129

Google Contacts Adapter

Google Contacts Version Support

The adapter leverages the People API to enable bidirectional access to Google Contacts.

SQL Compliance

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

Getting Started

Connecting to Google Contacts

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

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

1. Unzip the `tdv.googlecontacts.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.googlecontacts/tdv.googlecontacts.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 GoogleContacts
```

Basic Tab

Connecting to GoogleContacts

Authorize the adapter to access Google APIs on behalf of individual users or on behalf of a domain.

Use the OAuth authentication standard to connect to Google APIs.

Authenticating to Google Contacts

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

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

- 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 Contacts. 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 Contacts 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 Contacts 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 People API

Follow these steps to enable the People 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 "People API" and select **People API** from the search results.
4. On the **People 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 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	Changed <ul style="list-style-type: none"> Added the Default column to the sys_procedureparameters table.
09/30/2022	8308	General	Changed <ul style="list-style-type: none"> Added the IsPath column to the sys_procedureparameters table.
08/30/2022	8277	Google Contacts	Added

			<ul style="list-style-type: none"> Added Content input to support inputting binary data in the UploadAttachment stored procedure.
08/17/2022	8264	General	Changed <ul style="list-style-type: none"> We now support handling the keyword "COLLATE" as standard function name as well.
10/27/2021	7970	Google Contacts	Added <ul style="list-style-type: none"> Added the following columns to the Contacts table: Country, City, CountryCode, ExtendedAddress, PostalCode, Region, StreetAddress.
10/25/2021	7968	Google Contacts	Added <ul style="list-style-type: none"> Added the following columns to the Contacts and PeopleConnections tables: PhoneticNamePrefix, PhoneticFullName , PhoneticGivenName , PhoneticMiddleName, PhoneticFamilyName, PhoneticNameSuffix. Added the following columns to the OtherContacts table: PhoneticNamePrefix, PhoneticFullName , PhoneticGivenName , PhoneticMiddleName, PhoneticFamilyName, PhoneticNameSuffix.
09/02/2021	7915	General	Added <ul style="list-style-type: none"> Added support for the STRING_SPLIT table-valued function in the CROSS APPLY clause.
08/07/2021	7889	General	Changed <ul style="list-style-type: none"> Added the KeySeq column to the sys_foreignkeys table.

08/06/2021	7888	General	Changed <ul style="list-style-type: none"> Added the new sys_primarykeys system table.
07/23/2021	7874	General	Changed <ul style="list-style-type: none"> 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 > LAST_MONTH(). Formerly the two LAST_MONTH() methods would resolve to different datetimes. Now they will match. As a replacement for the previous behavior, the relative date/datetime functions in the criteria may have an 'L' appended to them. Ie: WHERE col > 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.
07/08/2021	7859	General	Added <ul style="list-style-type: none"> 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.
06/18/2021	7839	Google Contacts	Added <ul style="list-style-type: none"> 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.			
06/11/2021	7833	Google Contacts	<p>Added</p> <ul style="list-style-type: none"> The stored procedures UpdateOrDeleteContactPhoto, and CopyOtherContactsToMyContacts have been added. The tables GroupMembers, OtherContacts, and PeopleConnections have been added. <p>Changed</p> <ul style="list-style-type: none"> The driver has been updated to the People API due to the sunset of the Google Contacts API. ContactGroups is now writable. Default scopes in the GetOAuthAccessToken and GetOAuthAuthorizationUrls stored procedures are changed due to the move to the People API. Significantly altered the available columns in the existing tables due to required changes in transitioning to the People API.
04/23/2021	7785	General	<p>Added</p> <ul style="list-style-type: none"> Added support for handling client side formulas during insert / update. For example: UPDATE Table SET Col1 = Concat (Col1, " - ", Col2) WHERE Col2 LIKE 'A%'
04/23/2021	7783	General	<p>Changed</p> <ul style="list-style-type: none"> Updated how display sizes are determined for varchar primary key and foreign key columns so they will match the reported length of the column.

04/16/2021	7776	General	Added <ul style="list-style-type: none"> Non-conditional updates between two columns is now available to all drivers. For example: UPDATE Table SET Col1=Col2 Changed <ul style="list-style-type: none"> Reduced the length to 255 for varchar primary key and foreign key columns. Updated implicit and metadata caching to improve performance and support for multiple connections. Old metadata caches are not compatible - you would need to generate new metadata caches if you are currently using CacheMetadata. Updated index naming convention to avoid duplicates Updated and standardized Getting Started connection help. Added the Advanced Features section to the help of all drivers. Categorized connection property listings in the help for all editions.
04/15 /2021	7775	General	Changed <ul style="list-style-type: none"> 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 Contacts 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 Contacts 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 Contacts 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 [My Contacts] 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.

Inserting Parent and Child Records

Use Case

When inserting records, often there is a need to fill in details about child records that have a dependency on a parent.

For instance, when dealing with a CRM system, Invoices often may not be entered without at least one line item.

Invoices may have many line items, with each line item made up of several fields. This presents a unique challenge when offering the data as relational tables.

When reading the data, it is easy enough to model an Invoice and an InvoiceLineItem table with a foreign key connecting the two.

But during inserts, the CRM system will require both the Invoice and the InvoiceLineItems to be created in a single submission.

To solve this sort of problem, our tools offer child collection columns on the parent.

These columns may be used to submit insert statements that include details of both the parent and the child records.

From our example, the Invoice table may have a column called InvoiceLineItems.

During the insert, we can pass in the details of the records that would need to be inserted

to the InvoiceLineItems table into the InvoiceLineItems column of the Invoice record. This can be done using the following methods:

Methods for Inserting Parent/Child Records

The adapter facilitates two methods for inserting parent/child records: temporary table insertion and XML/JSON aggregate insertion.

Temporary (#TEMP) tables

The simplest way to enter data would be to use a #TEMP table, or temporary table, which the adapter will store in memory.

Reference the #TEMP table with the following syntax:

TableName#TEMP

#TEMP tables are stored in memory for the duration of a connection.

Therefore, in order to use them, you cannot close the connection between submitting inserts to them, and they cannot be used in environments where a different connection may be used for each query.

Within that single connection, the table remains in memory until the bulk insert is successful, at which point the temporary table will be wiped from memory.

For example:

```
INSERT INTO InvoiceLineItems#TEMP (ReferenceNumber, Item, Quantity,
Amount) VALUES ('INV001', 'Basketball', 10, 9.99)
INSERT INTO InvoiceLineItems#TEMP (ReferenceNumber, Item, Quantity,
Amount) VALUES ('INV001', 'Football', 5, 12.99)
```

Once the InvoiceLineItems table is populated, the #TEMP table may be referenced during an insert into the Invoice table:

```
INSERT INTO Invoices (ReferenceNumber, Customer, InvoiceLines) VALUES
('INV001', 'John Doe', 'InvoiceLineItems#TEMP')
```

Under the hood, the adapter will read in values from the #TEMP table.

Notice that the ReferenceNumber was used to identify what Invoice the lines are tied to. This is because the #TEMP table may be populated and used with a bulk insert, where you will have different lines for different Invoices.

This allows the #TEMP tables to be used with a bulk insert. For example:

```

INSERT INTO Invoices#TEMP (ReferenceNumber, Customer, InvoiceLines)
VALUES ('INV001', 'John Doe', 'InvoiceLineItems#TEMP')
INSERT INTO Invoices#TEMP (ReferenceNumber, Customer, InvoiceLines)
VALUES ('INV002', 'Jane Doe', 'InvoiceLineItems#TEMP')
INSERT INTO Invoices SELECT ReferenceNumber, Customer, InvoiceLines FROM
Invoices#TEMP

```

In this case, we are inserting two different Invoices. The ReferenceNumber is how we determine which Lines go with which Invoice.

Note: The tables and columns presented here are an example of how the adapter works in general. The specific table and column names may be different in the adapter.

XML/JSON Aggregates

As an alternative to #TEMP tables, direct XML/JSON may be used. Since #TEMP tables are not used to construct them, it does not matter if you use the same connection or close the connection after insert.

For example:

```

[
  {
    "Item", "Basketball",
    "Quantity": 10
    "Amount": 9.99
  },
  {
    "Item", "Football",
    "Quantity": 5
    "Amount": 12.99
  }
]

```

OR

```

<Row>
  <Item>Basketball</Item>
  <Quantity>10</Quantity>
  <Amount>9.99</Amount>
</Row>
<Row>
  <Item>Football</Item>
  <Quantity>5</Quantity>

```



```
<Amount>12.99</Amount>
</Row>
```

Note that the ReferenceNumber is not present in these examples.

That is because the XML/JSON by its nature is not being passed by reference, but passed in full per insert against the parent record.

There is no need to provide something to tie the child back to the parent since the complete XML/JSON must be constructed and submitted for each row.

Then, insert the values:

```
INSERT INTO Invoices (ReferenceNumber, Customer, InvoiceLines) VALUES
('INV001', 'John Doe', '{...}')
```

OR

```
INSERT INTO Invoices (ReferenceNumber, Customer, InvoiceLines) VALUES
('INV001', 'John Doe', '<Row>...</Row>')
```

Example for Google Contacts

Note: the key references such as Id may be different in your environment:

```
// Execute bulk insert
// Perform a bulk insert into the parent table [My Contacts].
INSERT INTO ContactOrganizations#TEMP
(Current,Department,Domain,JobDescription,Location,Name,StartDate) VALUES
(true,'Engineering','QA','Big Data Engineer','Bangalore','Tech
Mahindra','03-01-2021')
INSERT INTO ContactOrganizations#TEMP
(Current,Department,Domain,JobDescription,Location,Name,StartDate,EndDat
e) VALUES (false,'Engineering','QA','Software
Consultant','Bangalore','CData','05-23-2019','02-17-2020')
INSERT INTO [My Contacts] (GivenName, Organizations) VALUES
('John','ContactOrganizations#TEMP')

// Perform a bulk insert into the table ContactGroups.
INSERT INTO ContactGroups#TEMP (Name) VALUES ('Test Bulk')
INSERT INTO ContactGroups#TEMP (Name) VALUES ('Test Bulk 2')
INSERT INTO ContactGroups (Name) SELECT (Name) from ContactGroups#TEMP

// Perform a bulk insert into the table ContactGroups.
```

```

INSERT INTO ClientData#TEMP (ClientDataKey, ClientDataValue) VALUES
('Client Data 1','sfsdfsds45rf')
INSERT INTO ClientData#TEMP (ClientDataKey, ClientDataValue) VALUES
('Client Data 2','2jfk24g78w2jbdk')
INSERT INTO ContactGroups (Name,ClientData) VALUES ('Insert ClientData
to a Contact Group','ClientData#TEMP')

// Peform a bulk insert into the table [My Contacts].
INSERT INTO MyContacts#TEMP(GivenName, HomeAddresses) VALUES
('Elexandar', 'Bangalore India')
INSERT INTO MyContacts#TEMP(GivenName, HomeAddresses) VALUES ('Altin',
'Bangalore India')
INSERT INTO [My Contacts] (GivenName, HomeAddresses) SELECT GivenName,
HomeAddresses FROM MyContacts#TEMP

// Execute bulk update
INSERT INTO MyContacts#TEMP(Id, HomeAddresses) VALUES
('c970377457588673806', 'Bangalore India')
INSERT INTO MyContacts#TEMP(Id, HomeAddresses) VALUES
('c928048274343381622', 'Bangalore India')
UPDATE [My Contacts] (Id, HomeAddresses) SELECT Id, HomeAddresses FROM
MyContacts#TEMP

// Execute bulk delete
INSERT INTO MyContacts#TEMP(Id) VALUES ('c970377457588673806')
INSERT INTO MyContacts#TEMP(Id) VALUES ('c928048274343381622')
DELETE FROM [My Contacts] WHERE EXISTS SELECT Id FROM MyContacts#TEMP

```

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 Contacts 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 Contacts 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 Contacts 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 Verbosity 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 Verbosity to 2 will log everything included in Verbosity 1 and additional information about the request.
3	Setting Verbosity to 3 will additionally log HTTP headers, as well as the body of the request and the response.
4	Setting Verbosity to 4 will additionally log transport-level communication with the data source. This includes SSL negotiation.
5	Setting Verbosity to 5 will additionally log communication with the data source and additional details that may be helpful in troubleshooting problems. This includes interface commands.

The [Verbosity](#) should not be set to greater than 1 for normal operation. Substantial amounts of data can be logged at higher 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 Contacts 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 Contacts 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 Contacts adapter:


```

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

```

Examples

1. Return all columns:

```
SELECT * FROM [My Contacts]
```

2. Rename a column:

```
SELECT "HomeEmailAddresses" AS MY_HomeEmailAddresses FROM [My
Contacts]
```

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

```
SELECT CAST(AnnualRevenue AS VARCHAR) AS Str_AnnualRevenue FROM [My
Contacts]
```

4. Search data:

```
SELECT * FROM [My Contacts] WHERE Id = 'c7782206569106794554'
```

5. The Google Contacts APIs support the following operators in the WHERE clause: =, AND, IN.

```
SELECT * FROM [My Contacts] WHERE Id = 'c7782206569106794554';
```

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 GivenName, HomeEmailAddresses INTO
'csv://c:/[My Contacts].txt' FROM "[My Contacts]" WHERE Id =
'c7782206569106794554'");
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 "[My Contacts]" IN
'csv://filename=c:/[My Contacts].csv;delimiter=tab' FROM "[My Contacts]"
WHERE Id = 'c7782206569106794554'");
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 [My Contacts] (HomeEmailAddresses) VALUES
(?)";
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "John");
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 [My Contacts] SET HomeEmailAddresses='John' WHERE
Id = ?";
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "1");
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:googlecontacts:InitiateOAuth=GETANDREFRESH;",);
String cmd = "DELETE FROM [My Contacts] WHERE Id = ?";
```

```
PreparedStatement pstmt = connection.prepareStatement(cmd);
pstmt.setString(1, "1");
int count=pstmt.executeUpdate();
connection.close();
```

EXECUTE Statements

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

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

Stored Procedure Syntax

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

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

Example Statements

Reference stored procedure inputs by name:

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

Execute a parameterized stored procedure statement:

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

PIVOT and UNPIVOT

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

PIVOT

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

PIVOT Syntax

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

UNPIVOT

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

UNPIVOT Syntax

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

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

Data Model

Overview

This section shows the available API objects and provides more information on executing SQL to Google Contacts APIs.

Key Features

- The adapter models Google Contacts entities like documents, folders, and groups as relational views, allowing you to write SQL to query Google Contacts data.
- Stored procedures allow you to execute operations to Google Contacts.
- Live connectivity to these objects means any changes to your Google Contacts account are immediately reflected when using the adapter.

Tables

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

Views

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

Stored Procedures

[Stored Procedures](#) are function-like interfaces to Google Contacts. Stored procedures allow you to execute operations to Google Contacts, including downloading documents and moving envelopes.

Tables

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

Generally, querying Google Contacts 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 Contacts Adapter Tables

Name	Description
ContactGroups	Create, update, delete and query Contact Groups.
Contacts	Set the name of this table to any of your ContactGroups(Group Name) to create, update, delete, and query Google contacts.

ContactGroups

Create, update, delete and query Contact Groups.

Table Specific Information

Select

The ContactGroups table supports the Id and MemberCount columns in the WHERE clause. The supported operators are '=', in' for Id and '=' for MemberCount. For example:

```
SELECT * FROM [ContactGroups] WHERE Id = 'myContacts'  
SELECT * FROM [ContactGroups] WHERE Id = '45a15a878be3580a'  
SELECT * FROM [ContactGroups] WHERE Id = '4fe7dbbd0d26cc95' AND  
MemberCount = 10
```

Insert

To create a new ContactGroup, the Name is required. You can also insert data to ClientData columns

```
INSERT INTO [ContactGroups] (Name) VALUES ('Contacts Group Test')
```


To insert ClientData for a contactGroup, you can use the #TEMP table or create the JSON structure to provide the values to insert.

```
INSERT INTO ClientData#TEMP (ClientDataKey, ClientDataValue) VALUES
('Client Data 1','sfsdfsds45rf')
INSERT INTO ClientData#TEMP (ClientDataKey, ClientDataValue) VALUES
('Client Data 2','2jfk24g78w2jbdk')
INSERT INTO ContactGroups (Name,ClientData) VALUES ('Insert ClientData
to a Contact Group 2','ClientData#TEMP')
```

```
INSERT INTO ContactGroups (Name, ClientData) VALUES ('Testing Insert
ClientData in Contact Group via client data columns','[{"key": "Client
Data 1","value": "sdfjkbq36"},{"key": "Client Data 2","value":
"df32jkbk"}]')
```

Update

You can update the Name, ClientData of a ContactGroups row (except for System Groups (GroupType As SYSTEM_CONTACT_GROUP) like My Contacts, Coworkers, and Friends) by specifying the Id column.

```
UPDATE [ContactGroups] SET Name = 'Salsa Friends' WHERE Id =
'674ddb258de3ef81'
```

Delete

To delete a ContactGroup, the Id is required.

```
DELETE FROM [ContactGroups] WHERE Id = '674ddb258de3ef81'
```

Columns

Name	Type	ReadOnly	Description
Id [KEY]	String	True	The Id of the contact group.

ResourceName	<i>String</i>	True	The ResourceName of the contact group.
ETag	<i>String</i>	True	The HTTP entity tag of the resource.
Name	<i>String</i>	False	The name of the contact group.
GroupType	<i>String</i>	True	The contact group type.
FormattedName	<i>String</i>	True	The group name formatted in the viewer's account locale.
MemberCount	<i>Integer</i>	True	The total number of contacts in the group irrespective of max members in specified in the request.
MemberResourceNames	<i>String</i>	True	The list of contact person resource names that are members of the contact group.
MetadataUpdateTime	<i>Datetime</i>	True	The metadata of the contact group.
MetadataDeleted	<i>Boolean</i>	True	The metadata of the contact group.
ClientData	<i>String</i>	False	The group's client data.

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
ClientDataKey	<i>String</i>	The client specified key of the client data.
ClientDataValue	<i>String</i>	The client specified value of the client data.

Contacts

Set the name of this table to any of your ContactGroups(Group Name) to create, update, delete, and query Google contacts.

Select

The Contacts table supports the Id in the WHERE clause. The supported operators are '=', IN' for Id. For example:

```
SELECT * FROM [My Contacts] WHERE Id = '567931ee88efc109'
SELECT * FROM [My Contacts] WHERE Id IN
('567931ee88efc109','79733d1ee88efc109')
SELECT * FROM [My Contacts] WHERE SearchTerms = 'Yash'
```

Insert

You can insert any field into the Contacts table that is not read-only. You can add multiple values for few specific fields separated by semi-colon.

```
INSERT INTO [My Contacts](GivenName, FamilyName, HomeEmailAddresses,
WorkEmailAddresses, WorkPhoneNumbers, HomePhoneNumbers, HomeAddresses)
VALUES ('Elizabeth', 'Bennet', 'liz@gmail.com', 'liz@example.org', '
(206)555-1212' , '(206)555-1213', '1600 Amphitheatre Pkwy Mountain
View')
```

To insert organization details for a contact, Use the #Temp table to insert the fields of the organization. The fields of the organization is present in ContactOrganizations view. For Example:

```

INSERT INTO ContactOrganizations#TEMP (Current, Department, Domain,
JobDescription, Location, Name, StartDate) VALUES (true, 'Engineering',
'QA', 'Big Data Engineer', 'Bangalore', 'Tech Mahindra', '2021-03-01')
INSERT INTO ContactOrganizations#TEMP (Current, Department, Domain,
JobDescription, Location, Name, StartDate, EndDate) VALUES (false,
'Engineering', 'QA', 'Software Consultant', 'Bangalore', 'CData', '2019-
05-23', '2020-02-17')
INSERT INTO [My Contacts] (GivenName, Organizations) VALUES ('Karan',
'ContactOrganizations#TEMP')

```

To insert multiple contacts at once, use the #TEMP table to insert. For Example:

```

INSERT INTO MyContacts#TEMP (GivenName, FamilyName, HomeAddresses)
VALUES ('Ankit', 'Singh', 'Bangalore India; Indore India')
INSERT INTO MyContacts#TEMP (GivenName, MiddleName, HomeAddresses)
VALUES ('Aman', '', 'Bangalore India; Indore India')
INSERT INTO [My Contacts] (GivenName, HomeAddresses) SELECT
GivenName, HomeAddresses FROM MyContacts#TEMP

```

Update

You can update any field in the Contacts table that is not read-only. You can add multiple values for few specific fields separated by semi-colon.

```

UPDATE [My Contacts] SET GivenName = 'Elizabeth', FamilyName = 'Bennet',
MobilePhoneNumbers = '+3556969699999;+355676555001' WHERE Id =
'48b8b9158b1db34d'
UPDATE [My Contacts] SET HomeAddresses='Electronic City Bangalore India;
Brilliant Solitaire Indore India' WHERE Id='c6545017396039868174'

```

To update details of multiple contacts at once, use the #TEMP table. For Example:

```

INSERT INTO MyContacts#TEMP (Id, GivenName, HomeAddresses) VALUES
('c9197613024342508599', 'Ankit', 'Bangalore India; Indore India')
INSERT INTO MyContacts#TEMP (Id, GivenName, HomeAddresses) VALUES
('c2042847992816525584', 'Aman', 'Bangalore India; Indore India')
UPDATE [My Contacts] (Id, GivenName, HomeAddresses) SELECT Id,
GivenName, HomeAddresses FROM MyContacts#TEMP

```

Delete

To delete a Contact, the Id is required.

```
DELETE FROM [My Contacts] WHERE Id = '567931ee88efc109'
```

To delete multiple contacts at once, use the #TEMP table. For Example:

```
INSERT INTO MyContacts#TEMP(Id) VALUES ('c9197613024342508599')
Insert INTO MyContacts#TEMP(Id) VALUES ('c2042847992816525584')
DELETE FROM [My Contacts] WHERE EXISTS SELECT Id FROM MyContacts#TEMP
```

Columns

Name	Type	ReadOnly	Description
Id [KEY]	<i>String</i>	True	The Id of the contact.
ResourceName	<i>String</i>	True	The ResourceName of the contact.
Etag	<i>String</i>	True	The ETag of the resource. (Note that the quotation marks are part of the ETag.)
NamePrefix	<i>String</i>	True	The honorific prefixes, such as Mrs. or Dr.
FullName	<i>String</i>	True	The display name formatted according to the locale specified by the viewer's account
GivenName	<i>String</i>	False	The given name of the contact.
MiddleName	<i>String</i>	False	The middle name of the contact.
FamilyName	<i>String</i>	False	The family name of the contact.

NameSuffix	<i>String</i>	False	The honorific suffixes, such as Jr.
PhoneticNamePrefix	<i>String</i>	True	The Phonetic honorific prefixes, such as Mrs. or Dr.
PhoneticFullName	<i>String</i>	False	The Phonetic display name formatted according to the locale specified by the viewer's account.
PhoneticGivenName	<i>String</i>	False	The Phonetic given name of the contact.
PhoneticMiddleName	<i>String</i>	False	The Phonetic middle name of the contact.
PhoneticFamilyName	<i>String</i>	False	The Phonetic family name of the contact.
PhoneticNameSuffix	<i>String</i>	False	The Phonetic honorific suffixes, such as Jr.
NickNames	<i>String</i>	False	The nickname of the contact.
Birthday	<i>Date</i>	False	The birthday of the contact. The format of the date is yyyy-mm-dd.
GenderAddressMeAs	<i>String</i>	False	The type of pronoun that should be used to address the contact.
Gender	<i>String</i>	False	The gender for the contact.
Photos	<i>String</i>	True	The contact's photo.
UnlabeledEmailAddresses	<i>String</i>	False	The unlabeled email addresses of the contact. Multiple email addresses should be separated by

			semi-colon.
HomeEmailAddresses	<i>String</i>	False	The home email addresses of the contact. Multiple email addresses should be separated by semi-colon.
WorkEmailAddresses	<i>String</i>	False	The work email addresses of the contact. Multiple email addresses should be separated by semi-colon.
OtherEmailAddresses	<i>String</i>	False	The other email addresses of the contact. Multiple email addresses should be separated by semi-colon.
CustomEmailAddressTypes	<i>String</i>	False	The custom type of the email address. This attribute should be used along with CustomEmailAddresses. Multiple custom types should be separated by semi-colon.
CustomEmailAddresses	<i>String</i>	False	The custom value of the email address. This attribute should be used along with CustomEmailAddressTypes. Multiple custom emailAddresses should be separated by semi-colon.
UnlabeledPhoneNumbers	<i>String</i>	False	The unlabeled phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
HomePhoneNumbers	<i>String</i>	False	The home phone numbers of the contacts. Multiple phone numbers should be separated by semi-

			colon.
WorkPhoneNumbers	<i>String</i>	False	The work phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
MobilePhoneNumbers	<i>String</i>	False	The mobile phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
OtherPhoneNumbers	<i>String</i>	False	The homeFax phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
HomeFaxes	<i>String</i>	False	The workFax phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
WorkFaxes	<i>String</i>	False	The otherFax phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
OtherFaxes	<i>String</i>	False	The pager phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
Pagers	<i>String</i>	False	The workMobile phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
WorkMobilePhoneNumbers	<i>String</i>	False	The workPager phone numbers of the contacts. Multiple phone numbers should be separated by

			semi-colon.
WorkPagers	<i>String</i>	False	The main phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
MainPhoneNumbers	<i>String</i>	False	The googleVoice phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
GoogleVoicePhoneNumbers	<i>String</i>	False	The other phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
CustomPhoneNumberTypes	<i>String</i>	False	The custom type of the phone number. This attribute should be used along with CustomPhoneNumbers. Multiple custom types should be separated by semi-colon.
CustomPhoneNumbers	<i>String</i>	False	The custom value of the phone number. This attribute should be used along with CustomPhoneNumberTypes. Multiple custom phone numbers should be separated by semi-colon.
UnlabeledAddresses	<i>String</i>	False	The unlabeled addresses of the contact. Multiple addresses should be separated by semi-colon.
HomeAddresses	<i>String</i>	False	The home addresses of the contact. Multiple addresses should be separated by semi-

			colon.
WorkAddresses	<i>String</i>	False	The work addresses of the contact. Multiple addresses should be separated by semi-colon.
OtherAddresses	<i>String</i>	False	The other addresses of the contact. Multiple addresses should be separated by semi-colon.
CustomAddressTypes	<i>String</i>	False	The custom type of the address. This attribute should be used along with CustomAddresses. Multiple custom types should be separated by semi-colon.
CustomAddresses	<i>String</i>	False	The custom value of the address. This attribute should be used along with CustomAddressTypes. Multiple custom emailAddresses should be separated by semi-colon.
AgeRanges	<i>String</i>	True	An age range of a contact.
Biography	<i>String</i>	False	The short biography of the contact.
BiographyContentType	<i>String</i>	False	The content type of the biography of the contact.
HomeCalendarUrls	<i>String</i>	True	The home calendar urls of the contact. Multiple calendar urls should be separated by semi-colon.
FreeBusyCalendarUrls	<i>String</i>	True	The freeBusy calendar urls of the

			contact. Multiple calendar urls should be separated by semi-colon.
WorkCalendarUrls	<i>String</i>	True	The work calendar urls of the contact. Multiple calendar urls should be separated by semi-colon.
CustomCalendarUrlTypes	<i>String</i>	True	The custom type of the address. This attribute should be used along with CustomCalendarUrls. Multiple custom types should be separated by semi-colon.
CustomCalendarUrls	<i>String</i>	True	The custom value of the address. This attribute should be used along with CustomCalendarUrlTypes. Multiple custom calendar urls should be separated by semi-colon.
ClientData	<i>String</i>	False	The group's client data.
CoverPhotos	<i>String</i>	True	A contact's cover photo.
UnlabeledEvents	<i>String</i>	False	The dates of unlabeled events. Multiple dates should be separated by semi-colon.
AnniversaryEvents	<i>String</i>	False	The dates of anniversary events. Multiple dates should be separated by semi-colon.
OtherEvents	<i>String</i>	False	The dates of other events. Multiple dates should be separated by semi-colon.

CustomEventTypes	<i>String</i>	False	The custom type of an event. Multiple types should be separated by semi-colon.
CustomEvents	<i>String</i>	False	The custom value of an event. Multiple custom event dates should be separated by semi-colon.
AccountExternalIds	<i>String</i>	False	The account externalIds of the contact. Multiple externalIds should be separated by semi-colon.
CustomerExternalIds	<i>String</i>	False	The customer externalIds of the contact. Multiple externalIds should be separated by semi-colon.
LoginIdExternalIds	<i>String</i>	False	The loginId externalIds of the contact. Multiple externalIds should be separated by semi-colon.
NetworkExternalIds	<i>String</i>	False	The network externalIds of the contact. Multiple externalIds should be separated by semi-colon.
OrganizationExternalIds	<i>String</i>	False	The organization externalIds of the contact. Multiple externalIds should be separated by semi-colon.
CustomExternalIdTypes	<i>String</i>	False	The custom type of the externalId. This attribute should be used along with CustomExternalIds. Multiple types should be separated by semi-colon.

CustomExternalIds	<i>String</i>	False	The custom value of the externalId. This attribute should be used along with CustomExternalIdTypes. Multiple externalIds should be separated by semi-colon.
FileAses	<i>String</i>	True	The name that should be used to sort the contact in a list.
ImClientsProtocols	<i>String</i>	True	The semi-colon separated list of ImClients Protocols. Multiple protocols should be separated by semi-colon.
ImClientsUsernames	<i>String</i>	True	The semi-colon separated list of ImClients Usernames. Multiple usernames should be separated by semi-colon.
Interests	<i>String</i>	False	The contact's interests.
Locales	<i>String</i>	False	The contact's locales.
DeskLocations	<i>String</i>	False	The desk locations of the contact. Multiple locations should be separated by semi-colon.
GrewUpLocations	<i>String</i>	False	The grewUp locations of the contact. Multiple locations should be separated by semi-colon.
CustomLocationTypes	<i>String</i>	False	The custom types of the location. The attribute should be used along with CustomLocations. Multiple locations should be separated by semi-colon.

CustomLocations	<i>String</i>	False	The custom value of the location. The attribute should be used along with CustomLocationsTypes. Multiple locations should be separated by semi-colon.
Memberships	<i>String</i>	True	The resource name for the contact group, assigned by the server. Only contactGroupName can be used for modifying memberships. Any contact group membership can be removed, but only user group or 'myContacts' or 'starred' system groups memberships can be added. A contact must always have at least one contact group membership.
MiscKeywordTypes	<i>String</i>	False	<p>The miscellaneous keyword types. Multiple types should be separated by semi-colon. Allowed Values are : TYPE_UNSPECIFIED, OUTLOOK_BILLING_INFORMATION, OUTLOOK_DIRECTORY_SERVER, OUTLOOK_KEYWORD, OUTLOOK_MILEAGE, OUTLOOK_PRIORITY, OUTLOOK_SENSITIVITY, OUTLOOK_SUBJECT, OUTLOOK_USER, HOME, WORK, OTHER</p> <p>The allowed values are <i>TYPE_UNSPECIFIED, OUTLOOK_BILLING_INFORMATION, OUTLOOK_DIRECTORY_SERVER, OUTLOOK_KEYWORD, OUTLOOK_MILEAGE, OUTLOOK_PRIORITY, OUTLOOK_SENSITIVITY, OUTLOOK_SUBJECT,</i></p>

			<i>OUTLOOK_USER, HOME, WORK, OTHER.</i>
MiscKeywordValues	<i>String</i>	False	The miscellaneous keywords. Multiple miscellaneous keywords should be separated by semi-colon.
Occupations	<i>String</i>	False	The contact's occupations
Organizations	<i>String</i>	False	The organizations of the contact.
RelationshipTypes	<i>String</i>	True	The contact's relation to the other person. This attribute should be used along with RelationValues. Multiple types should be separated by semi-colon.
RelationshipValues	<i>String</i>	True	The name of the other person this relation refers to. This attribute should be used along with RelationTypes. Multiple names should be separated by semi-colon.
UnlabeledSipAddresses	<i>String</i>	False	The dates of anniversary events. Multiple dates should be separated by semi-colon.
HomeSipAddresses	<i>String</i>	False	The home sipAddresses of the contact. Multiple sipAddresses should be separated by semi-colon.
WorkSipAddresses	<i>String</i>	False	The work sipAddresses of the contact. Multiple sipAddresses should be separated by semi-colon.

MobileSipAddresses	<i>String</i>	False	The mobile sipAddresses of the contact. Multiple sipAddresses should be separated by semi-colon.
OtherSipAddresses	<i>String</i>	False	The other sipAddresses of the contact. Multiple sipAddresses should be separated by semi-colon.
CustomSipAddressTypes	<i>String</i>	False	The custom type of the sipAddress. This attribute should be used along with CustomSipAddresses. Multiple types should be separated by semi-colon.
CustomSipAddresses	<i>String</i>	False	The custom value of the sipAddress. This attribute should be used along with CustomSipAddressTypes. Multiple sipAddresses should be separated by semi-colon.
Skills	<i>String</i>	True	The skills that the contact has.
UnlabeledUrls	<i>String</i>	False	The unlabeled website urls of the contact. Multiple urls should be separated by semi-colon.
HomeUrls	<i>String</i>	False	The home website urls of the contact. Multiple urls should be separated by semi-colon.
WorkUrls	<i>String</i>	False	The work website urls of the contact. Multiple urls should be separated by semi-colon.
BlogUrls	<i>String</i>	False	The blog website urls of the

			contact. Multiple urls should be separated by semi-colon.
ProfileUrls	<i>String</i>	False	The profile website urls of the contact. Multiple urls should be separated by semi-colon.
HomePageUrls	<i>String</i>	False	The homePage website urls of the contact. Multiple urls should be separated by semi-colon.
FtpUrls	<i>String</i>	False	The ftp website urls of the contact. Multiple urls should be separated by semi-colon.
ReservationsUrls	<i>String</i>	False	The reservations website urls of the contact. Multiple urls should be separated by semi-colon.
OtherUrls	<i>String</i>	False	The appInstall Page website urls of the contact. Multiple urls should be separated by semi-colon.
AppInstallPageUrls	<i>String</i>	False	The other website urls of the contact. Multiple urls should be separated by semi-colon.
CustomUrlTypes	<i>String</i>	False	The custom type of the website url. This attribute values should be used along with CustomUrls. Multiple urls should be separated by semi-colon.
CustomUrls	<i>String</i>	False	The custom value of the website url. This attribute values should be used along with CustomUrlTypes. Multiple urls should be separated by semi-colon.

UnlabeledCountry	<i>String</i>	True	The unlabeled country of the contact. Multiple addresses should be separated by semi-colon.
UnlabeledCity	<i>String</i>	True	The unlabeled city of the contact. Multiple addresses should be separated by semi-colon.
UnlabeledCountryCode	<i>String</i>	True	The unlabeled country code of the contact. Multiple addresses should be separated by semi-colon.
UnlabeledExtendedAddress	<i>String</i>	True	The unlabeled extended address of the contact. Multiple addresses should be separated by semi-colon.
UnlabeledPostalCode	<i>String</i>	True	The unlabeled postal code of the contact. Multiple addresses should be separated by semi-colon.
UnlabeledRegion	<i>String</i>	True	The unlabeled region of the contact. Multiple addresses should be separated by semi-colon.
UnlabeledStreetAddress	<i>String</i>	True	The unlabeled street address of the contact. Multiple addresses should be separated by semi-colon.
HomeCountry	<i>String</i>	True	The home country of the contact. Multiple addresses should be separated by semi-colon.
HomeCity	<i>String</i>	True	The home city of the contact. Multiple addresses should be

			separated by semi-colon.
HomeCountryCode	<i>String</i>	True	The home country code of the contact. Multiple addresses should be separated by semi-colon.
HomeExtendedAddress	<i>String</i>	True	The home extended address of the contact. Multiple addresses should be separated by semi-colon.
HomePostalCode	<i>String</i>	True	The home postal code of the contact. Multiple addresses should be separated by semi-colon.
HomeRegion	<i>String</i>	True	The home region of the contact. Multiple addresses should be separated by semi-colon.
HomeStreetAddress	<i>String</i>	True	The home street address of the contact. Multiple addresses should be separated by semi-colon.
WorkCountry	<i>String</i>	True	The work country of the contact. Multiple addresses should be separated by semi-colon.
WorkCity	<i>String</i>	True	The work city of the contact. Multiple addresses should be separated by semi-colon.
WorkCountryCode	<i>String</i>	True	The work country code of the contact. Multiple addresses should be separated by semi-colon.
WorkExtendedAddress	<i>String</i>	True	The work extended address of the

			contact. Multiple addresses should be separated by semi-colon.
WorkPostalCode	<i>String</i>	True	The work postal code of the contact. Multiple addresses should be separated by semi-colon.
WorkRegion	<i>String</i>	True	The work region of the contact. Multiple addresses should be separated by semi-colon.
WorkStreetAddress	<i>String</i>	True	The work street address of the contact. Multiple addresses should be separated by semi-colon.
OtherCountry	<i>String</i>	True	The other country of the contact. Multiple addresses should be separated by semi-colon.
OtherCity	<i>String</i>	True	The other city of the contact. Multiple addresses should be separated by semi-colon.
OtherCountryCode	<i>String</i>	True	The other country code of the contact. Multiple addresses should be separated by semi-colon.
OtherExtendedAddress	<i>String</i>	True	The other extended address of the contact. Multiple addresses should be separated by semi-colon.
OtherPostalCode	<i>String</i>	True	The other postal code of the contact. Multiple addresses should be separated by semi-colon.

OtherRegion	<i>String</i>	True	The other region of the contact. Multiple addresses should be separated by semi-colon.
OtherStreetAddress	<i>String</i>	True	The other street address of the contact. Multiple addresses should be separated by semi-colon.
CustomCountry	<i>String</i>	True	The custom country of the contact. Multiple addresses should be separated by semi-colon.
CustomCity	<i>String</i>	True	The custom city of the contact. Multiple addresses should be separated by semi-colon.
CustomCountryCode	<i>String</i>	True	The custom country code of the contact. Multiple addresses should be separated by semi-colon.
CustomExtendedAddress	<i>String</i>	True	The custom extended address of the contact. Multiple addresses should be separated by semi-colon.
CustomPostalCode	<i>String</i>	True	The custom postal code of the contact. Multiple addresses should be separated by semi-colon.
CustomRegion	<i>String</i>	True	The custom region of the contact. Multiple addresses should be separated by semi-colon.
CustomStreetAddress	<i>String</i>	True	The custom street address of the contact. Multiple addresses should be separated by semi-colon.

Pseudo-Columns

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

Name	Type	Description
SearchTerms	<i>String</i>	The query matches on a contact's names, nickNames, emailAddresses, phoneNumbers, and organizations fields that are from the CONTACT source. Maximum 10 matching rows will be returned.
ClientDataKey	<i>String</i>	The client specified key of the client data.
ClientDataValue	<i>String</i>	The client specified value of the client data.

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 Contacts Adapter Views

Name	Description
------	-------------

ContactOrganizations	This view specifies all the details of the Contact's Organizations
GroupMembers	Query contact group resource names
OtherContacts	The view specifies all the details of Other Contacts
PeopleConnections	This view specifies all the details of People Connections

ContactOrganizations

This view specifies all the details of the Contact's Organizations

Select

The ContactOrganizations view supports the Id in the WHERE clause. The supported operators are '=', IN' for Id. For example:

```
SELECT * FROM [ContactOrganizations] WHERE Id = '567931ee88efc109'
SELECT * FROM [ContactOrganizations] WHERE Id IN
('567931ee88efc109', '79733d1ee88efc109')
```

Columns

Name	Type	Description
Id	<i>String</i>	The Id of the contact.
ResourceName	<i>String</i>	The resource name of the contact.
Current	<i>Boolean</i>	True if organization is the person's current organization
Department	<i>String</i>	The person's department at the organization.

Domain	<i>String</i>	The domain name associated with the organization
FormattedType	<i>String</i>	The type of the organization translated and formatted in the viewer's account locale
JobDescription	<i>String</i>	The person's job description at the organization.
Location	<i>String</i>	The location of the organization office the person works at.
Name	<i>String</i>	The name of the organization.
PhoneticName	<i>String</i>	The phonetic name of the organization.
StartDate	<i>Date</i>	The start date when the person joined the organization.
EndDate	<i>Date</i>	The end date when the person left the organization.
Symbol	<i>String</i>	The symbol associated with the organization.
Title	<i>String</i>	The contact's job title at the organization.
Type	<i>String</i>	The type of the organization.

GroupMembers

Query contact group resource names

Table Specific Information

Select

The GroupMembers view supports the Id and MemberCount columns in the WHERE clause. Id is required for this view. For example:

```
SELECT * FROM [GroupMembers] WHERE Id = '4fe7dbbd0d26cc95'
```

Columns

Name	Type	Description
Id	<i>String</i>	The Id of the contact group.
MemberCount	<i>Integer</i>	The total number of contacts in the group irrespective of max members in specified in the request.
ContactId	<i>String</i>	The list of contact person resource names that are members of the contact group.

OtherContacts

The view specifies all the details of Other Contacts

Table Specific Information

Select

For example:

```
SELECT * FROM [OtherContacts] WHERE SearchTerms = 'Yash'
```

Columns

Name	Type	Description
Id [KEY]	<i>String</i>	The Id of the contact.
ResourceName	<i>String</i>	The ResourceName of the contact.
Etag	<i>String</i>	The ETag of the resource. (Note that the quotation marks are part of the ETag.)
NamePrefix	<i>String</i>	The honorific prefixes, such as Mrs. or Dr.
FullName	<i>String</i>	The display name formatted according to the locale specified by the viewer's account
GivenName	<i>String</i>	The given name of the contact.
MiddleName	<i>String</i>	The middle name of the contact.
FamilyName	<i>String</i>	The family name of the contact.
NameSuffix	<i>String</i>	The honorific suffixes, such as Jr.
PhoneticNamePrefix	<i>String</i>	The Phonetic honorific prefixes, such as Mrs. or Dr.
PhoneticFullName	<i>String</i>	The Phonetic display name formatted according to the locale specified by the viewer's account.
PhoneticGivenName	<i>String</i>	The Phonetic given name of the contact.

PhoneticMiddleName	<i>String</i>	The Phonetic middle name of the contact.
PhoneticFamilyName	<i>String</i>	The Phonetic family name of the contact.
PhoneticNameSuffix	<i>String</i>	The Phonetic honorific suffixes, such as Jr.
UnlabeledEmailAddresses	<i>String</i>	The unlabeled email addresses of the contact. Multiple email addresses should be separated by comma.
HomeEmailAddresses	<i>String</i>	The home email addresses of the contact. Multiple email addresses should be separated by comma.
WorkEmailAddresses	<i>String</i>	The work email addresses of the contact. Multiple email addresses should be separated by comma.
OtherEmailAddresses	<i>String</i>	The other email addresses of the contact. Multiple email addresses should be separated by comma.
CustomEmailAddressTypes	<i>String</i>	The custom type of the email address. This attribute should be used along with CustomEmailAddresses. Multiple custom types should be separated by comma.
CustomEmailAddresses	<i>String</i>	The custom value of the email address. This attribute should be used along with CustomEmailAddressTypes. Multiple custom emailAddresses should be separated by comma.
UnlabeledPhoneNumbers	<i>String</i>	The unlabeled phone numbers of the contacts. Multiple phone numbers should be separated by comma.
HomePhoneNumbers	<i>String</i>	The home phone numbers of the contacts. Multiple phone numbers should be separated by

		comma.
WorkPhoneNumbers	<i>String</i>	The work phone numbers of the contacts. Multiple phone numbers should be separated by comma.
MobilePhoneNumbers	<i>String</i>	The mobile phone numbers of the contacts. Multiple phone numbers should be separated by comma.
HomeFaxes	<i>String</i>	The homeFax phone numbers of the contacts. Multiple phone numbers should be separated by comma.
WorkFaxes	<i>String</i>	The workFax phone numbers of the contacts. Multiple phone numbers should be separated by comma.
OtherFaxes	<i>String</i>	The otherFax phone numbers of the contacts. Multiple phone numbers should be separated by comma.
Pagers	<i>String</i>	The pager phone numbers of the contacts. Multiple phone numbers should be separated by comma.
WorkMobilePhoneNumbers	<i>String</i>	The workMobile phone numbers of the contacts. Multiple phone numbers should be separated by comma.
WorkPagers	<i>String</i>	The workPager phone numbers of the contacts. Multiple phone numbers should be separated by comma.
MainPhoneNumbers	<i>String</i>	The main phone numbers of the contacts. Multiple phone numbers should be separated by comma.
GoogleVoicePhoneNumbers	<i>String</i>	The googleVoice phone numbers of the contacts. Multiple phone numbers should be separated by

		comma.
OtherPhoneNumbers	<i>String</i>	The other phone numbers of the contacts. Multiple phone numbers should be separated by comma.
CustomPhoneNumberTypes	<i>String</i>	The custom type of the phone number. This attribute should be used along with CustomPhoneNumbers. Multiple custom types should be separated by comma.
CustomPhoneNumbers	<i>String</i>	The custom value of the phone number. This attribute should be used along with CustomPhoneNumberTypes. Multiple custom phone numbers should be separated by comma.

Pseudo-Columns

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

Name	Type	Description
SearchTerms	<i>String</i>	The query matches on a contact's names, emailAddresses, and phoneNumbers fields that are from the OTHER_CONTACT source. Maximum 10 matching rows will be returned.

PeopleConnections

This view specifies all the details of People Connections

Columns

Name	Type	Description
Id [KEY]	<i>String</i>	The Id of the contact.
ResourceName	<i>String</i>	The ResourceName of the contact.
Etag	<i>String</i>	The ETag of the resource. (Note that the quotation marks are part of the ETag.)
NamePrefix	<i>String</i>	The honorific prefixes, such as Mrs. or Dr.
FullName	<i>String</i>	The display name formatted according to the locale specified by the viewer's account
GivenName	<i>String</i>	The given name of the contact.
MiddleName	<i>String</i>	The middle name of the contact.
FamilyName	<i>String</i>	The family name of the contact.
NameSuffix	<i>String</i>	The honorific suffixes, such as Jr.
PhoneticNamePrefix	<i>String</i>	The Phonetic honorific prefixes, such as Mrs. or Dr.
PhoneticFullName	<i>String</i>	The Phonetic display name formatted according to the locale specified by the viewer's account.
PhoneticGivenName	<i>String</i>	The Phonetic given name of the contact.
PhoneticMiddleName	<i>String</i>	The Phonetic middle name of the contact.

PhoneticFamilyName	<i>String</i>	The Phonetic family name of the contact.
PhoneticNameSuffix	<i>String</i>	The Phonetic honorific suffixes, such as Jr.
NickNames	<i>String</i>	The nickname of the contact.
Birthday	<i>Date</i>	The birthday of the contact. The format of the date is yyyy-mm-dd.
GenderAddressMeAs	<i>String</i>	The type of pronoun that should be used to address the contact.
Gender	<i>String</i>	The gender for the contact.
Photos	<i>String</i>	The contact's photo.
UnlabeledEmailAddresses	<i>String</i>	The unlabeled email addresses of the contact. Multiple email addresses should be separated by semi-colon.
HomeEmailAddresses	<i>String</i>	The home email addresses of the contact. Multiple email addresses should be separated by semi-colon.
WorkEmailAddresses	<i>String</i>	The work email addresses of the contact. Multiple email addresses should be separated by semi-colon.
OtherEmailAddresses	<i>String</i>	The other email addresses of the contact. Multiple email addresses should be separated by semi-colon.
CustomEmailAddressTypes	<i>String</i>	The custom type of the email address. This

		attribute should be used along with CustomEmailAddresses. Multiple custom types should be separated by semi-colon.
CustomEmailAddresses	<i>String</i>	The custom value of the email address. This attribute should be used along with CustomEmailAddressTypes. Multiple custom emailAddresses should be separated by semi-colon.
UnlabeledPhoneNumbers	<i>String</i>	The unlabeled phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
HomePhoneNumbers	<i>String</i>	The home phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
WorkPhoneNumbers	<i>String</i>	The work phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
MobilePhoneNumbers	<i>String</i>	The mobile phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
OtherPhoneNumbers	<i>String</i>	The homeFax phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
HomeFaxes	<i>String</i>	The workFax phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
WorkFaxes	<i>String</i>	The otherFax phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
OtherFaxes	<i>String</i>	The pager phone numbers of the contacts. Multiple phone numbers should be separated by

		semi-colon.
Pagers	<i>String</i>	The workMobile phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
WorkMobilePhoneNumbers	<i>String</i>	The workPager phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
WorkPagers	<i>String</i>	The main phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
MainPhoneNumbers	<i>String</i>	The googleVoice phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
GoogleVoicePhoneNumbers	<i>String</i>	The other phone numbers of the contacts. Multiple phone numbers should be separated by semi-colon.
CustomPhoneNumberTypes	<i>String</i>	The custom type of the phone number. This attribute should be used along with CustomPhoneNumbers. Multiple custom types should be separated by semi-colon.
CustomPhoneNumbers	<i>String</i>	The custom value of the phone number. This attribute should be used along with CustomPhoneNumberTypes. Multiple custom phone numbers should be separated by semi-colon.
UnlabeledAddresses	<i>String</i>	The unlabeled addresses of the contact. Multiple addresses should be separated by semi-colon.
HomeAddresses	<i>String</i>	The home addresses of the contact. Multiple addresses should be separated by semi-colon.
WorkAddresses	<i>String</i>	The work addresses of the contact. Multiple

		addresses should be separated by semi-colon.
OtherAddresses	<i>String</i>	The other addresses of the contact. Multiple addresses should be separated by semi-colon.
CustomAddressTypes	<i>String</i>	The custom type of the address. This attribute should be used along with CustomAddresses. Multiple custom types should be separated by semi-colon.
CustomAddresses	<i>String</i>	The custom value of the address. This attribute should be used along with CustomAddressTypes. Multiple custom emailAddresses should be separated by semi-colon.
AgeRanges	<i>String</i>	An age range of a contact.
Biography	<i>String</i>	The short biography of the contact.
BiographyContentType	<i>String</i>	The content type of the biography of the contact.
HomeCalendarUrls	<i>String</i>	The home calendar urls of the contact. Multiple calendar urls should be separated by semi-colon.
FreeBusyCalendarUrls	<i>String</i>	The freeBusy calendar urls of the contact. Multiple calendar urls should be separated by semi-colon.
WorkCalendarUrls	<i>String</i>	The work calendar urls of the contact. Multiple calendar urls should be separated by semi-colon.
CustomCalendarUrlTypes	<i>String</i>	The custom type of the address. This attribute should be used along with CustomCalendarUrls. Multiple custom types should be separated by semi-colon.
CustomCalendarUrls	<i>String</i>	The custom value of the address. This attribute

		should be used along with CustomCalendarUrlTypes. Multiple custom calendar urls should be separated by semi-colon.
ClientData	<i>String</i>	The group's client data.
CoverPhotos	<i>String</i>	A contact's cover photo.
UnlabeledEvents	<i>String</i>	The dates of unlabeled events. Multiple dates should be separated by semi-colon.
AnniversaryEvents	<i>String</i>	The dates of anniversary events. Multiple dates should be separated by semi-colon.
OtherEvents	<i>String</i>	The dates of other events. Multiple dates should be separated by semi-colon.
CustomEventTypes	<i>String</i>	The custom type of an event. Multiple types should be separated by semi-colon.
CustomEvents	<i>String</i>	The custom value of an event. Multiple custom event dates should be separated by semi-colon.
AccountExternalIds	<i>String</i>	The account externalIds of the contact. Multiple externalIds should be separated by semi-colon.
CustomerExternalIds	<i>String</i>	The customer externalIds of the contact. Multiple externalIds should be separated by semi-colon.
LoginIdExternalIds	<i>String</i>	The loginId externalIds of the contact. Multiple externalIds should be separated by semi-colon.
NetworkExternalIds	<i>String</i>	The network externalIds of the contact. Multiple externalIds should be separated by semi-colon.
OrganizationExternalIds	<i>String</i>	The organization externalIds of the contact. Multiple externalIds should be separated by semi-colon.

CustomExternalIdTypes	<i>String</i>	The custom type of the externalId. This attribute should be used along with CustomExternalIds. Multiple types should be separated by semi-colon.
CustomExternalIds	<i>String</i>	The custom value of the externalId. This attribute should be used along with CustomExternalIdTypes. Multiple externalIds should be separated by semi-colon.
FileAses	<i>String</i>	The name that should be used to sort the contact in a list.
ImClientsProtocols	<i>String</i>	The custom type of the imClient. This attribute should be used along with CustomImClients. Multiple custom types should be separated by semi-colon.
ImClientsUsernames	<i>String</i>	The custom value of the imClient. This attribute should be used along with CustomImClientTypes. Multiple custom imClients should be separated by semi-colon.
Interests	<i>String</i>	The contact's interests.
Locales	<i>String</i>	The contact's locales.
DeskLocations	<i>String</i>	The desk locations of the contact. Multiple locations should be separated by semi-colon.
GrewUpLocations	<i>String</i>	The grewUp locations of the contact. Multiple locations should be separated by semi-colon.
CustomLocationTypes	<i>String</i>	The custom types of the location. The attribute should be used along with CustomLocations. Multiple locations should be separated by semi-colon.

CustomLocations	<i>String</i>	The custom value of the location. The attribute should be used along with CustomLocationsTypes. Multiple locations should be separated by semi-colon.
Memberships	<i>String</i>	The resource name for the contact group, assigned by the server. Only contactGroupName can be used for modifying memberships. Any contact group membership can be removed, but only user group or 'myContacts' or 'starred' system groups memberships can be added. A contact must always have at least one contact group membership.
MiscKeywordTypes	<i>String</i>	<p>The miscellaneous keyword types. Multiple types should be separated by semi-colon. Allowed Values are : TYPE_UNSPECIFIED, OUTLOOK_BILLING_INFORMATION, OUTLOOK_DIRECTORY_SERVER, OUTLOOK_KEYWORD, OUTLOOK_MILEAGE, OUTLOOK_PRIORITY, OUTLOOK_SENSITIVITY, OUTLOOK_SUBJECT, OUTLOOK_USER, HOME, WORK, OTHER</p> <p>The allowed values are <i>TYPE_UNSPECIFIED, OUTLOOK_BILLING_INFORMATION, OUTLOOK_DIRECTORY_SERVER, OUTLOOK_KEYWORD, OUTLOOK_MILEAGE, OUTLOOK_PRIORITY, OUTLOOK_SENSITIVITY, OUTLOOK_SUBJECT, OUTLOOK_USER, HOME, WORK, OTHER.</i></p>
MiscKeywordValues	<i>String</i>	The miscellaneous keywords. Multiple miscellaneous keywords should be separated by semi-colon.
Occupations	<i>String</i>	The contact's occupations
Organizations	<i>String</i>	The organizations of the contact.

RelationshipTypes	<i>String</i>	The contact's relation to the other person. This attribute should be used along with RelationValues. Multiple types should be separated by semi-colon.
RelationshipValues	<i>String</i>	The name of the other person this relation refers to. This attribute should be used along with RelationTypes. Multiple names should be separated by semi-colon.
UnlabeledSipAddresses	<i>String</i>	The unlabeled sipAddresses of the contact. Multiple sipAddresses should be separated by semi-colon.
HomeSipAddresses	<i>String</i>	The home sipAddresses of the contact. Multiple sipAddresses should be separated by semi-colon.
WorkSipAddresses	<i>String</i>	The work sipAddresses of the contact. Multiple sipAddresses should be separated by semi-colon.
MobileSipAddresses	<i>String</i>	The mobile sipAddresses of the contact. Multiple sipAddresses should be separated by semi-colon.
OtherSipAddresses	<i>String</i>	The other sipAddresses of the contact. Multiple sipAddresses should be separated by semi-colon.
CustomSipAddressTypes	<i>String</i>	The custom type of the sipAddress. This attribute should be used along with CustomSipAddresses. Multiple types should be separated by semi-colon.
CustomSipAddresses	<i>String</i>	The custom value of the sipAddress. This attribute should be used along with CustomSipAddressTypes. Multiple sipAddresses should be separated by semi-colon.
Skills	<i>String</i>	The skills that the contact has.
UnlabeledUrls	<i>String</i>	The unlabeled website urls of the contact.

		Multiple urls should be separated by semi-colon.
HomeUrls	<i>String</i>	The home website urls of the contact. Multiple urls should be separated by semi-colon.
WorkUrls	<i>String</i>	The work website urls of the contact. Multiple urls should be separated by semi-colon.
BlogUrls	<i>String</i>	The blog website urls of the contact. Multiple urls should be separated by semi-colon.
ProfileUrls	<i>String</i>	The profile website urls of the contact. Multiple urls should be separated by semi-colon.
HomePageUrls	<i>String</i>	The homePage website urls of the contact. Multiple urls should be separated by semi-colon.
FtpUrls	<i>String</i>	The ftp website urls of the contact. Multiple urls should be separated by semi-colon.
ReservationsUrls	<i>String</i>	The reservations website urls of the contact. Multiple urls should be separated by semi-colon.
OtherUrls	<i>String</i>	The appInstall Page website urls of the contact. Multiple urls should be separated by semi-colon.
AppInstallPageUrls	<i>String</i>	The other website urls of the contact. Multiple urls should be separated by semi-colon.
CustomUrlTypes	<i>String</i>	The custom type of the website url. This attribute values should be used along with CustomUrls. Multiple urls should be separated by semi-colon.
CustomUrls	<i>String</i>	The custom value of the website url. This attribute values should be used along with CustomUrlTypes. Multiple urls should be separated by semi-colon.

Stored Procedures

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

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

Google Contacts Adapter Stored Procedures

Name	Description
CopyOtherContactsToMyContacts	Copies an 'Other contact' to a new contact in the user's 'myContacts' group
GetOAuthAccessToken	Gets an authentication token from GoogleContactsV2.
GetOAuthAuthorizationURL	Gets the authorization URL that must be opened separately by the user to grant access to your application. Only needed when developing Web apps. You will request the OAuthAccessToken from this URL.
ModifyContactGroupMembers	Modify the members of a contact group owned by the authenticated user.
RefreshOAuthAccessToken	Refreshes the OAuth access token used for authentication with GoogleContactsV2.
UpdateOrDeleteContactPhoto	Update or Delete a contact's photo.

CopyOtherContactsToMyContacts

Copies an 'Other contact' to a new contact in the user's 'myContacts' group

Stored Procedure Specific Information

Process of Copy Other Contacts To My Contacts

Google Contacts allows only a small subset of columns to be used in the Exec query. These columns can typically be used with only = comparison. The available columns for CopyOtherContactsToMyContacts are Id, FieldsToCopy and Sources. For example:

```
EXECUTE CopyOtherContactsToMyContacts Id = 'c2583603548064626314',
FieldsToCopy = 'emailAddresses,names,phoneNumbers', Sources = 'READ_
SOURCE_TYPE_CONTACT'
```

Input

Name	Type	Required	Description
Id	String	True	The Id of the 'Other contact' to copy.
FieldsToCopy	String	True	The fields to restrict which fields are copied into the new contact. Valid values are: emailAddresses,names,phoneNumbers
Sources	String	True	A mask of what source types to return. Valid values are: READ_SOURCE_TYPE_UNSPECIFIED, READ_SOURCE_TYPE_PROFILE, READ_SOURCE_TYPE_CONTACT, READ_SOURCE_TYPE_DOMAIN_CONTACT

Result Set Columns

Name	Type	Description
Success	String	Returns True if contact group has been modified.

GetOAuthAccessToken

Gets an authentication token from GoogleContactsV2.

Input

Name	Type	Required	Description
AuthMode	String	True	<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	False	<p>The verifier code returned by Google after permission for the app to connect has been granted. WEB Authmode only.</p>
Scope	String	True	<p>The scope of access to Google APIs. By default, access to all APIs used by this data provider will be specified.</p> <p>The default value is</p> <p>https://www.googleapis.com/auth/contacts</p> <p>https://www.googleapis.com/auth/contacts.other.readonly</p> <p>https://www.googleapis.com/auth/contacts.readonly</p> <p>https://www.googleapis.com/auth/user.gender.read</p> <p>https://www.googleapis.com/auth/user.emails.read</p> <p>https://www.googleapis.com/auth/user.birthday.read</p> <p>https://www.googleapis.com/auth/user.addresses.read</p> <p>https://www.googleapis.com/auth/profile.language.read</p> <p>https://www.googleapis.com/auth/profile.age.range.read</p> <p>https://www.googleapis.com/auth/user.organization.read</p> <p>https://www.googleapis.com/auth/user.phonenumbers.read.</p>

CallbackURL	String	False	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	True	<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 displayed to the user.</p> <p>The default value is <i>SELECT_ACCOUNT</i>.</p>
AccessType	String	True	<p>This field indicates if your application needs to access a Google API when the user is not present at the browser. This parameter defaults to 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	False	This field indicates any state which 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. Uses include redirecting the user to the correct resource in your site, using nonces, and mitigating cross-site request forgery.

Result Set Columns

Name	Type	Description
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>	A token that may be used to obtain a new access token.
ExpiresIn	<i>String</i>	The remaining lifetime on the access token.

GetOAuthAuthorizationURL

Gets the authorization URL that must be opened separately by the user to grant access to your application. Only needed when developing Web apps. You will request the OAuthAccessToken from this URL.

Input

Name	Type	Required	Description
Scope	<i>String</i>	<i>True</i>	<p>The scope of access to Google APIs. By default, access to all APIs used by this data provider will be specified.</p> <p>The default value is https://www.googleapis.com/auth/contacts https://www.googleapis.com/auth/contacts.other.readonly https://www.googleapis.com/auth/contacts.readonly https://www.googleapis.com/auth/user.gender.read https://www.googleapis.com/auth/user.emails.read https://www.googleapis.com/auth/user.birthday.read</p>

			https://www.googleapis.com/auth/user.addresses.read https://www.googleapis.com/auth/profile.language.read https://www.googleapis.com/auth/profile.ageRange.read https://www.googleapis.com/auth/user.organization.read https://www.googleapis.com/auth/user.phonenumbers.read .
CallbackURL	String	False	<p>This field determines where the response is sent. The value of this parameter must exactly match one of the values registered in the APIs Console, including the HTTP or HTTPS schemes, capitalization, and trailing forward slash, ('/').</p>
Prompt	String	True	<p>This field indicates the prompt to present the user. 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 displayed to the user.</p> <p>The default value is <i>SELECT_ACCOUNT</i>.</p>
AccessType	String	True	<p>This field indicates if your application needs to access a Google API when the user is not present at the browser. This parameter defaults to 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	<i>String</i>	<i>False</i>	This field indicates any state which 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. Uses include redirecting the user to the correct resource in your site, using nonces, and mitigating cross-site request forgery.
-------	---------------	--------------	---

Result Set Columns

Name	Type	Description
URL	<i>String</i>	The authorization URL, entered into a Web browser to obtain the verifier token and authorize your app.

ModifyContactGroupMembers

Modify the members of a contact group owned by the authenticated user.

Stored Procedure Specific Information

Process of Modify Contact Group Members

Google Contacts allows only a small subset of columns to be used in the Exec query. These columns can typically be used with only = comparison. The available columns for ModifyContactGroupMembers are Id, ResourceIdsToAdd and ResourceIdsToRemove.

Note: The only system contact groups that can have members added are contactGroups/myContacts and contactGroups/starred. Other system contact groups are deprecated and can only have contacts removed. For example:

```
EXECUTE ModifyContactGroupMembers Id = 'starred', ResourceIdsToAdd =
'c8493601355484697130,c8493601355484697130', ResourceIdsToRemove = ''
```

Input

Name	Type	Required	Description
Id	<i>String</i>	<i>True</i>	The Id of the contact group to modify.
ResourceIdsToAdd	<i>String</i>	<i>False</i>	A comma separated list of ids of the contact people to add.
ResourceIdsToRemove	<i>String</i>	<i>False</i>	A comma separated list of ids of the contact people to remove.

Result Set Columns

Name	Type	Description
Success	<i>String</i>	Returns True if contact group has been modified.

RefreshOAuthAccessToken

Refreshes the OAuth access token used for authentication with GoogleContactsV2.

Input

Name	Type	Required	Description
OAuthRefreshToken	<i>String</i>	<i>True</i>	Set this to the token value that expired.

Result Set Columns

Name	Type	Description
OAuthAccessToken	String	The authentication token returned from GoogleContactsV2. This can be used in subsequent calls to other operations for this particular service.
OAuthRefreshToken	String	This is the same as the access token.
ExpiresIn	String	The remaining lifetime on the access token.

UpdateOrDeleteContactPhoto

Update or Delete a contact's photo.

Stored Procedure Specific Information

Process of Update or Delete Contact Photo

Google Contacts allows only a small subset of columns to be used in the Exec query. These columns can typically be used with only = comparison. The available columns for UpdateOrDeleteContactPhoto are Id, Action, FileSource and Url.

Note: Id is required column. The Action attribute defines the operation to perform. The supported operators are 'Update, Delete' for Action. You need to specify either FileSource or URL to update the contact photo. For example:

```
EXECUTE UpdateOrDeleteContactPhoto Id = 'c2845916184580148264', Action =
'Update', FileSource = 'D:\GooglePeople_API_Outputs\contactphoto.png'
EXECUTE UpdateOrDeleteContactPhoto Id = 'c2845916184580148264', Action =
'Update', Url = 'https://play-lh.googleusercontent.com/-
Xqd3k7aJqZY/AAAAAAAAAAI/AAAAAAAAAA/AMZuuckWFzqX627ygMhiilKbqmIA-T_
AsQ/photo.jpg'
EXECUTE UpdateOrDeleteContactPhoto Id = 'c2845916184580148264', Action =
'Delete'
```


Input

Name	Type	Required	Description
Id	<i>String</i>	<i>True</i>	The Id of the contact.
Action	<i>String</i>	<i>True</i>	Update or Delete operation you want to perform on Contact Photo. Possible values: Update, Delete
FileSource	<i>String</i>	<i>False</i>	The complete filepath of the photo to be uploaded. You need to specify either this field or URL.
URL	<i>String</i>	<i>False</i>	An accessible URL the image will be downloaded from and then posted to the target. You need to specify either this field or FileSource.

Result Set Columns

Name	Type	Description
Success	<i>String</i>	Returns True if contact group has been modified.

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
AuthScheme	The type of authentication to use when connecting to GoogleContacts.

OAuth

Property	Description
InitiateOAuth	Set this property to initiate the process to obtain or refresh the OAuth access token when you connect.
OAuthClientId	The client Id assigned when you register your application with an OAuth authorization server.
OAuthClientSecret	The client secret assigned when you register your application with an OAuth authorization server.
OAuthAccessToken	The access token for connecting using OAuth.
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://.
CallbackURL	The OAuth callback URL to return to when authenticating. This value must match the callback URL you specify in your app settings.
Scope	Specify scope to obtain the initial access and refresh token.
OAuthVerifier	The verifier code returned from the OAuth authorization URL.
OAuthRefreshToken	The OAuth refresh token for the corresponding OAuth access token.
OAuthExpiresIn	The lifetime in seconds of the OAuth AccessToken.

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

JWT OAuth

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

SSL

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

Firewall

Property	Description
FirewallType	The protocol used by a proxy-based firewall.

FirewallServer	The name or IP address of a proxy-based firewall.
FirewallPort	The TCP port for a proxy-based firewall.
FirewallUser	The user name to use to authenticate with a proxy-based firewall.
FirewallPassword	A password used to authenticate to a proxy-based firewall.

Proxy

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

Logging

Property	Description
LogModules	Core modules to be included in the log file.

Schema

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

Miscellaneous

Property	Description
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.
Other	These hidden properties are used only in specific use cases.
Readonly	You can use this property to enforce read-only access to GoogleContacts from the provider.
Timeout	The value in seconds until the timeout error is thrown, canceling the operation.
UserDefinedViews	A filepath pointing to the JSON configuration file containing your custom views.

Authentication

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

Property	Description
AuthScheme	The type of authentication to use when connecting to GoogleContacts.

AuthScheme

The type of authentication to use when connecting to GoogleContacts.

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.
- GCPIInstanceAccount: 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
InitiateOAuth	Set this property to initiate the process to obtain or refresh the OAuth access token when you connect.
OAuthClientId	The client Id assigned when you register your application with an OAuth authorization server.
OAuthClientSecret	The client secret assigned when you register your application with an OAuth authorization server.
OAuthAccessToken	The access token for connecting using OAuth.
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://.
CallbackURL	The OAuth callback URL to return to when authenticating. This value must match the callback URL you specify in your app settings.
Scope	Specify scope to obtain the initial access and refresh token.
OAuthVerifier	The verifier code returned from the OAuth authorization URL.
OAuthRefreshToken	The OAuth refresh token for the corresponding OAuth access token.
OAuthExpiresIn	The lifetime in seconds of the OAuth AccessToken.
OAuthTokenTimestamp	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\\GoogleContacts 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\\GoogleContacts 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

"https://www.googleapis.com/auth/contacts
 https://www.googleapis.com/auth/contacts.other.readonly
 https://www.googleapis.com/auth/contacts.readonly
 https://www.googleapis.com/auth/user.gender.read
 https://www.googleapis.com/auth/user.emails.read"

```
https://www.googleapis.com/auth/user.birthday.read  
https://www.googleapis.com/auth/user.addresses.read  
https://www.googleapis.com/auth/profile.language.read  
https://www.googleapis.com/auth/profile.age.range.read  
https://www.googleapis.com/auth/user.organization.read  
https://www.googleapis.com/auth/user.phonenumbers.read"
```

Remarks

Specify scope to obtain the initial access and refresh token.

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
OAuthJWTCert	The JWT Certificate store.
OAuthJWTCertType	The type of key store containing the JWT Certificate.
OAuthJWTCertPassword	The password for the OAuth JWT certificate.
OAuthJWTCertSubject	The subject of the OAuth JWT certificate.
OAuthJWTIssuer	The issuer of the Java Web Token.
OAuthJWTSubject	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, JKSFILe, JKSBLOB, PEMKEY_FILE, PEMKEY_BLOB, PUBLIC_KEY_FILE, PUBLIC_KEY_BLOB, SSHPUBLIC_KEY_FILE, SSHPUBLIC_KEY_BLOB, P7BFILE, PPKFILE, XMLFILE, XMLBLOB

Data Type

string

Default Value

"USER"

Remarks

This property can take one of the following values:

USER	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.

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.

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.

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
SSLServerCert	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
FirewallType	The protocol used by a proxy-based firewall.
FirewallServer	The name or IP address of a proxy-based firewall.
FirewallPort	The TCP port for a proxy-based firewall.
FirewallUser	The user name to use to authenticate with a proxy-based firewall.
FirewallPassword	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 Contacts and traffic flows back and forth through the proxy.
SOCKS4	1080	When this is set, the adapter sends data through the SOCKS 4 proxy specified by FirewallServer and FirewallPort and passes the FirewallUser value to the proxy, which determines if the connection request should be granted.
SOCKS5	1080	When this is set, the adapter sends data through the SOCKS 5 proxy specified by FirewallServer and FirewallPort . If your proxy requires authentication, set FirewallUser and FirewallPassword to credentials the proxy recognizes.

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

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
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.
ProxyServer	The hostname or IP address of a proxy to route HTTP traffic through.
ProxyPort	The TCP port the ProxyServer proxy is running on.
ProxyAuthScheme	The authentication type to use to authenticate to the ProxyServer proxy.
ProxyUser	A user name to be used to authenticate to the ProxyServer proxy.

ProxyPassword	A password to be used to authenticate to the ProxyServer proxy.
ProxySSLType	The SSL type to use when connecting to the ProxyServer proxy.
ProxyExceptions	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:

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

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
LogModules	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
Location	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\\GoogleContacts 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\\GoogleContacts 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
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.
Other	These hidden properties are used only in specific use cases.
Readonly	You can use this property to enforce read-only access to GoogleContacts from the provider.
Timeout	The value in seconds until the timeout error is thrown, canceling the operation.
UserDefinedViews	A filepath pointing to the JSON configuration file containing your custom views.

MaxRows

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

Data Type

int

Default Value

-1

Remarks

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

Other

These hidden properties are used only in specific use cases.

Data Type

string

Default Value

""

Remarks

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

Specify multiple properties in a semicolon-separated list.

Integration and Formatting

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

Readonly

You can use this property to enforce read-only access to GoogleContacts 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 [My Contacts] WHERE MyColumn = 'value'"
  },
  "MyView2": {
    "query": "SELECT * FROM MyTable WHERE Id IN (1,2,3)"
  }
}
```

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

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

TIBCO Product Documentation and Support Services

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

How to Access TIBCO Documentation

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

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

Product-Specific Documentation

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

- **Users**

- TDV Getting Started Guide

- TDV User Guide

- TDV Web UI User Guide

- TDV Client Interfaces Guide

- TDV Tutorial Guide

- TDV Northbay Example

- **Administration**

- TDV Installation and Upgrade Guide

- TDV Administration Guide

- TDV Active Cluster Guide

- TDV Security Features Guide

- **Data Sources**

- TDV Adapter Guides

- TDV Data Source Toolkit Guide (Formerly Extensibility Guide)

- **References**

- TDV Reference Guide

- TDV Application Programming Interface Guide

- **Other**

- TDV Business Directory Guide

- TDV Discovery Guide

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

How to Contact TIBCO Support

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

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

Release Version Support

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

https://docs.tibco.com/pub/tdv/general/LTS/tdv_LTS_releases.htm.

How to Join TIBCO Community

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

Legal and Third-Party Notices

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

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

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

TIBCO, TIBCO logo, Two-Second Advantage, TIBCO Spotfire, TIBCO ActiveSpaces, TIBCO Spotfire Developer, TIBCO EMS, TIBCO Spotfire Automation Services, TIBCO Enterprise Runtime for R, TIBCO Spotfire Server, TIBCO Spotfire Web Player, TIBCO Spotfire Statistics Services, S-PLUS, and TIBCO Spotfire S+ are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

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

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

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

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

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

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

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

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