Denodo Google Sheets Custom Wrapper User Manual

Revision 20221129

NOTE
This document is confidential and proprietary of **Denodo Technologies**.
No part of this document may be reproduced in any form by any means without prior written authorization of **Denodo Technologies**.

Copyright © 2023
Denodo Technologies Proprietary and Confidential

Denodo Inc.: 525 University Avenue, Suite 31, Palo Alto, CA 94301. USA www.denodo.com

CONTENTS

1 OVERVIEW

Google Sheets is a web-based application for creating and editing spreadsheets. With Google Sheets, you can create and edit spreadsheets directly in your web browser or mobile devices. Multiple people can work simultaneously, you can see people's changes as they make them, and every change is saved automatically.

With the Denodo Google Sheets Custom Wrapper you will be able to query the data in your spreadsheets directly in Denodo. With a datasource based on this CW, you can create base views that point to the desired sheets and query their data.

2 REQUIREMENTS

2.1 GET OAUTH2 CREDENTIALS

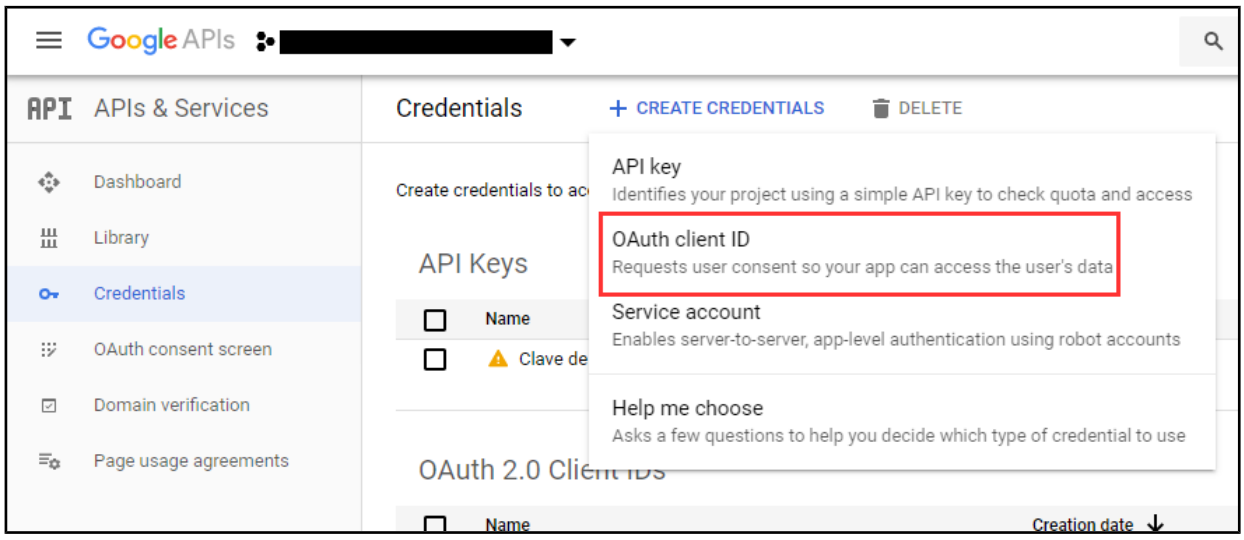
2.1.1 Google API configuration

Before using the Google Sheets API with this Custom Wrapper, it is necessary to have a Google Developers Account configured to be able to authenticate with OAuth 2.0:

1. Access the Google Developers Console at https://console.developers.google.com.

2. Go to the "`Credentials`" tab and create a new Project clicking:

    a. `Create Credentials > OAuth Client ID`.

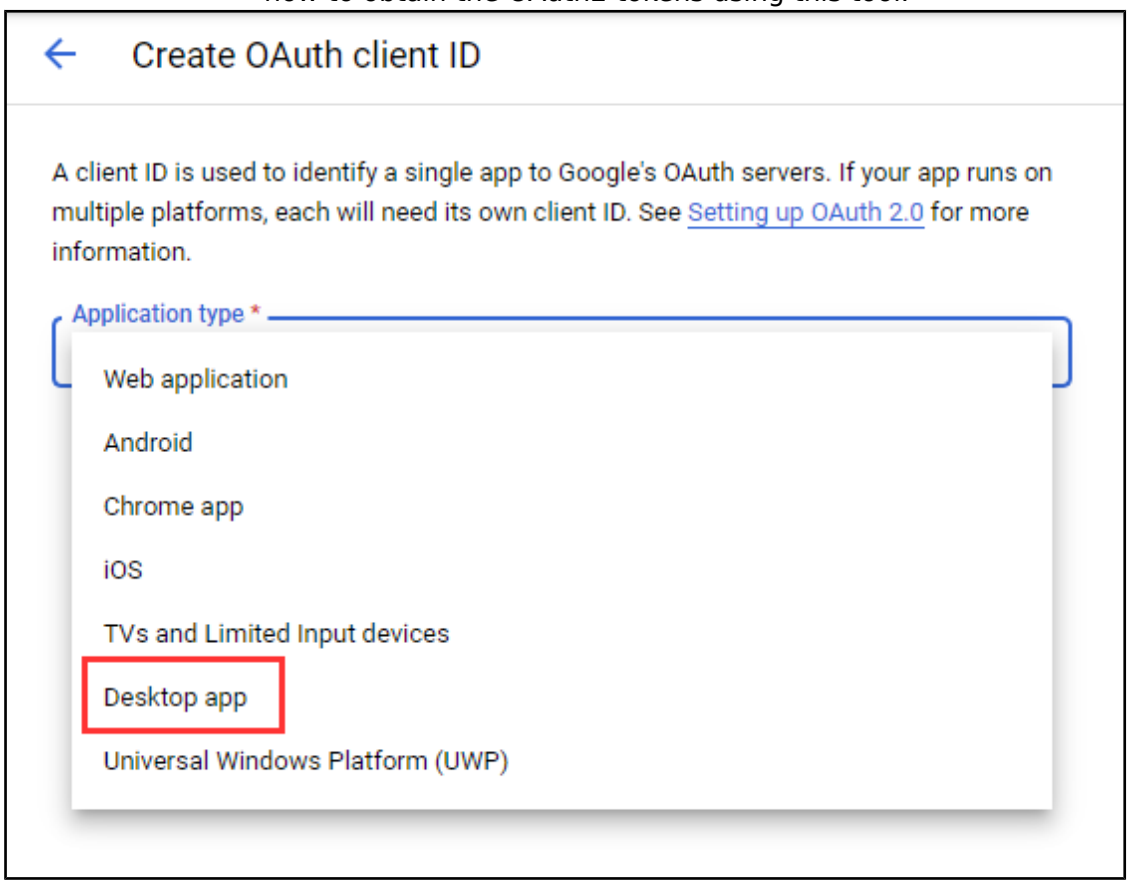


    b. Choose `Application type` : "`Desktop app`" and give it a name.

        i. Note that if you are accessing VDP from a different domain than `localhost`, in this step you will need to select "`Web application`" and fill the `Authorized redirect URLs` field with the appropriate URL(s) which should be something like: `http://<host_name>:<port>/oauth/2.0/redirectURL.jsp`
        Otherwise, if you use "`Desktop app`", you'll be prompted with a `"Error 400: redirect_uri_mismatch`" error.

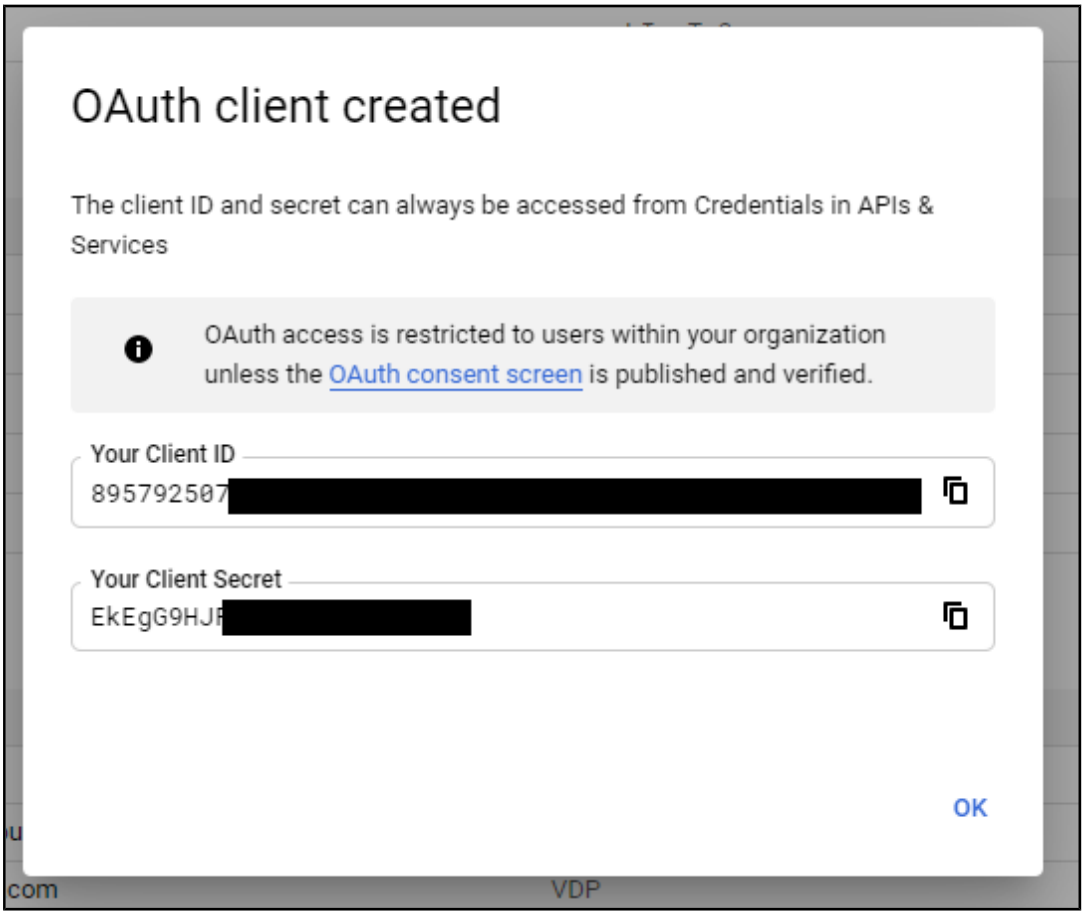
        In this scenario, using the `Google OAuth 2.0 Playground tool` is the preferred way. If you use this tool, you will also need to add to the `Authorized redirect URLs` this URL

https://developers.google.com/oauthplayground, so the refresh token can be used to obtain new access tokens. See **"Alternative method: Google OAuth 2.0 Playground" section** to check out how to obtain the OAuth2 tokens using this tool.

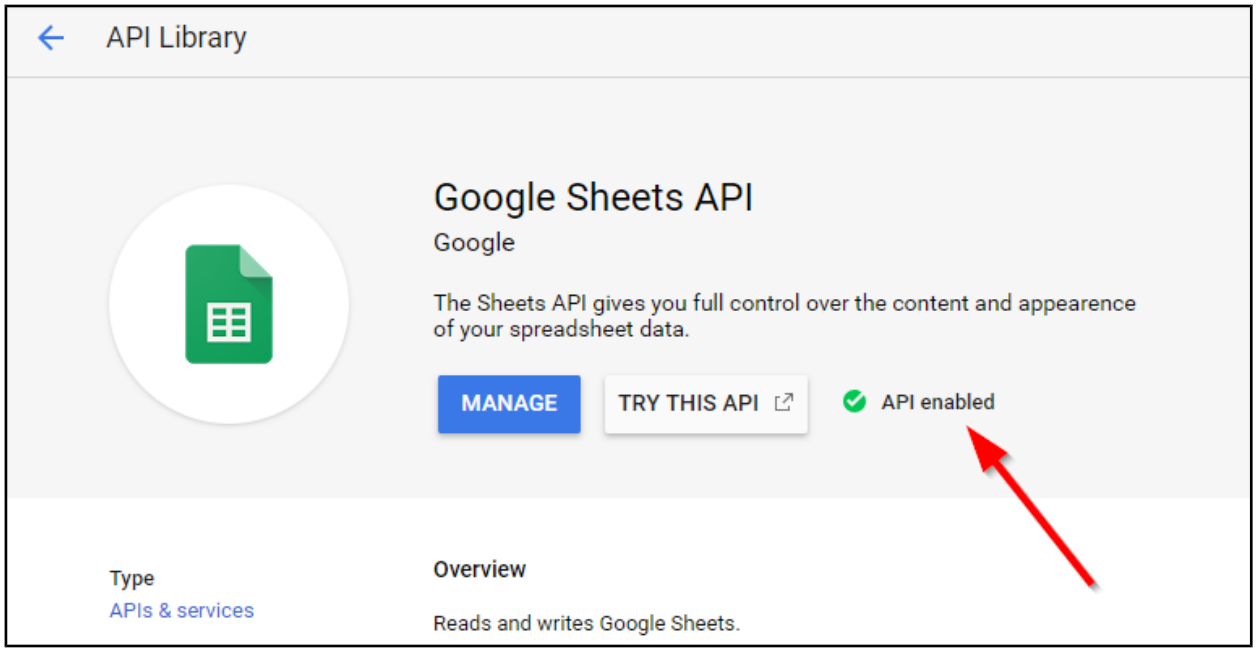


3. Once the Developer account is configured, two values will be used for the OAuth authentication. They will be known both to Google and the application:

    a. `Client ID`.
    b. `Client Secret`.

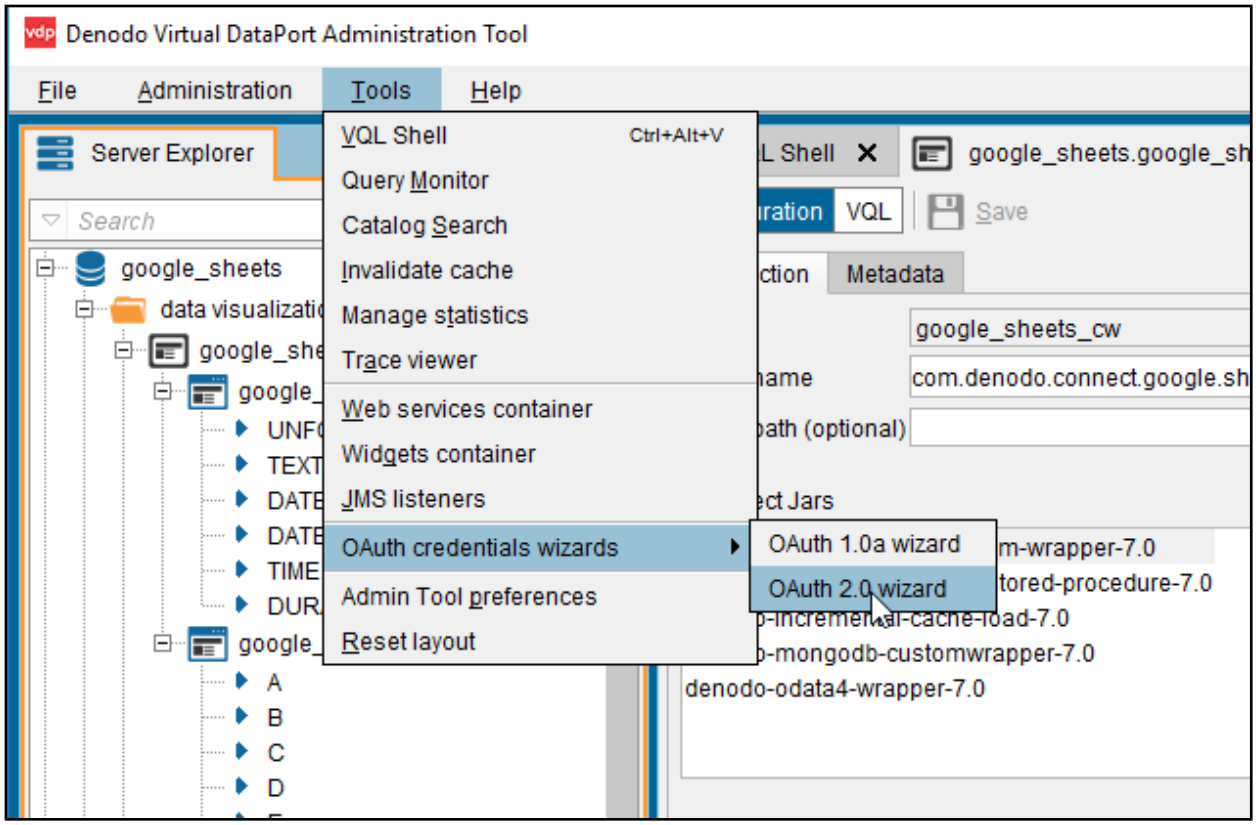


4. Make sure the Google Sheets API is enabled for the account. To do so, go to the `Dashboard` section in the Developer Console and click on `Enable` if needed.



2.1.2 Generate OAuth2 tokens in VDP

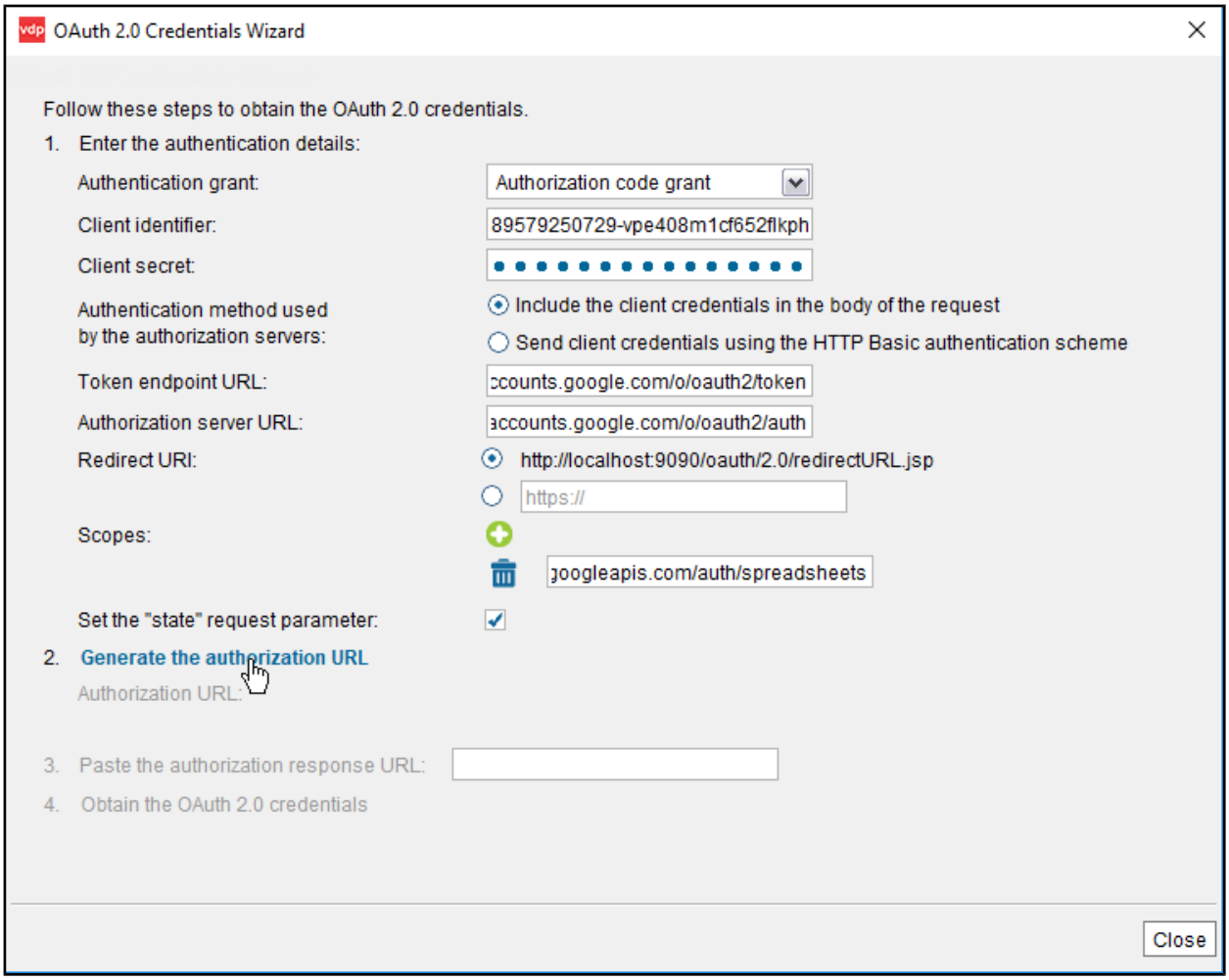
Now, we can create the tokens needed to authenticate the Denodo data sources that the stored procedure will create. To make this, we will use the OAuth credentials wizards from VDP. Note that this method should be used only when the application created in the previous chapter is a `Desktop app` type. If you created a `web application`, you should go to the **Alternative method: Google OAuth 2.0 Playground" section** in order to obtain the OAuth2 tokens.



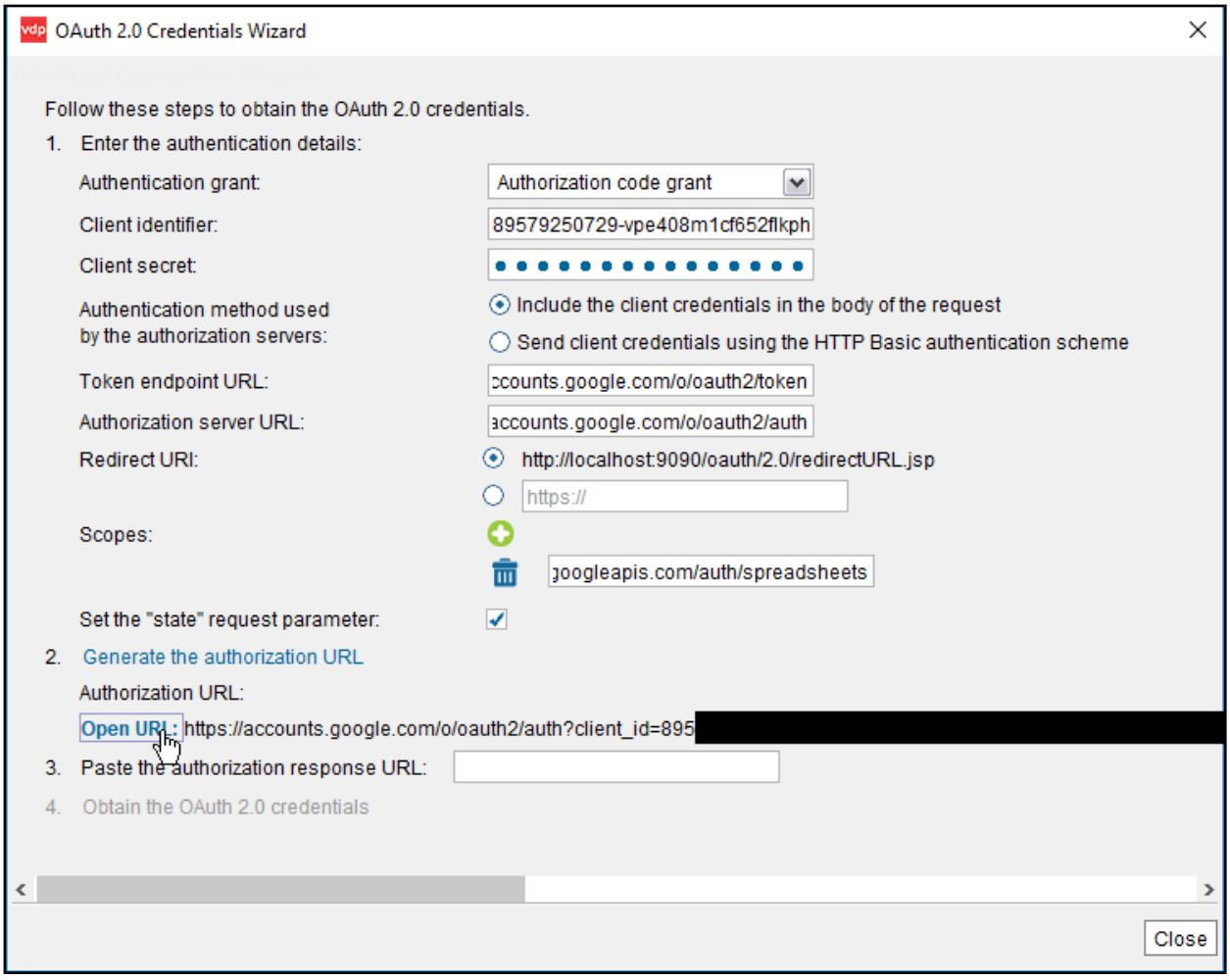
You have to fill in the form with the following parameters:

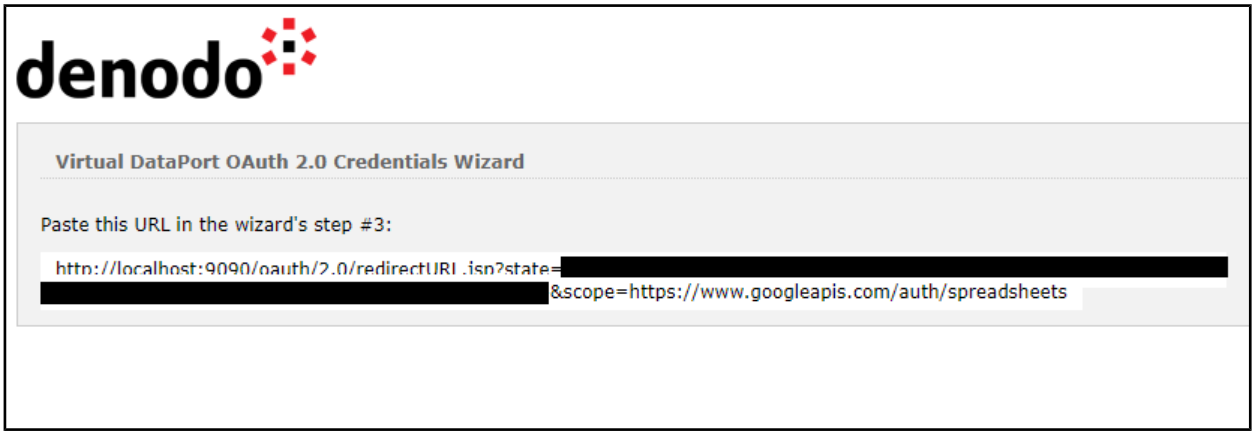
- **`Authentication grant`**: Authorization code grant
- **`Client identifier`**: Obtained from the **"Google API configuration" section**
- **`Client secret`**: Obtained from the **"Google API configuration" section**
- **`Authentication method`**: Include the client credentials in the body of the request
- **`Token endpoint URL`**: https://accounts.google.com/o/oauth2/token.
- **`Authorization server URL`**: https://accounts.google.com/o/oauth2/auth
- **`Scopes`**: https://www.googleapis.com/auth/spreadsheets

Once the form is completed, you have to click on `Generate the authorization URL`:

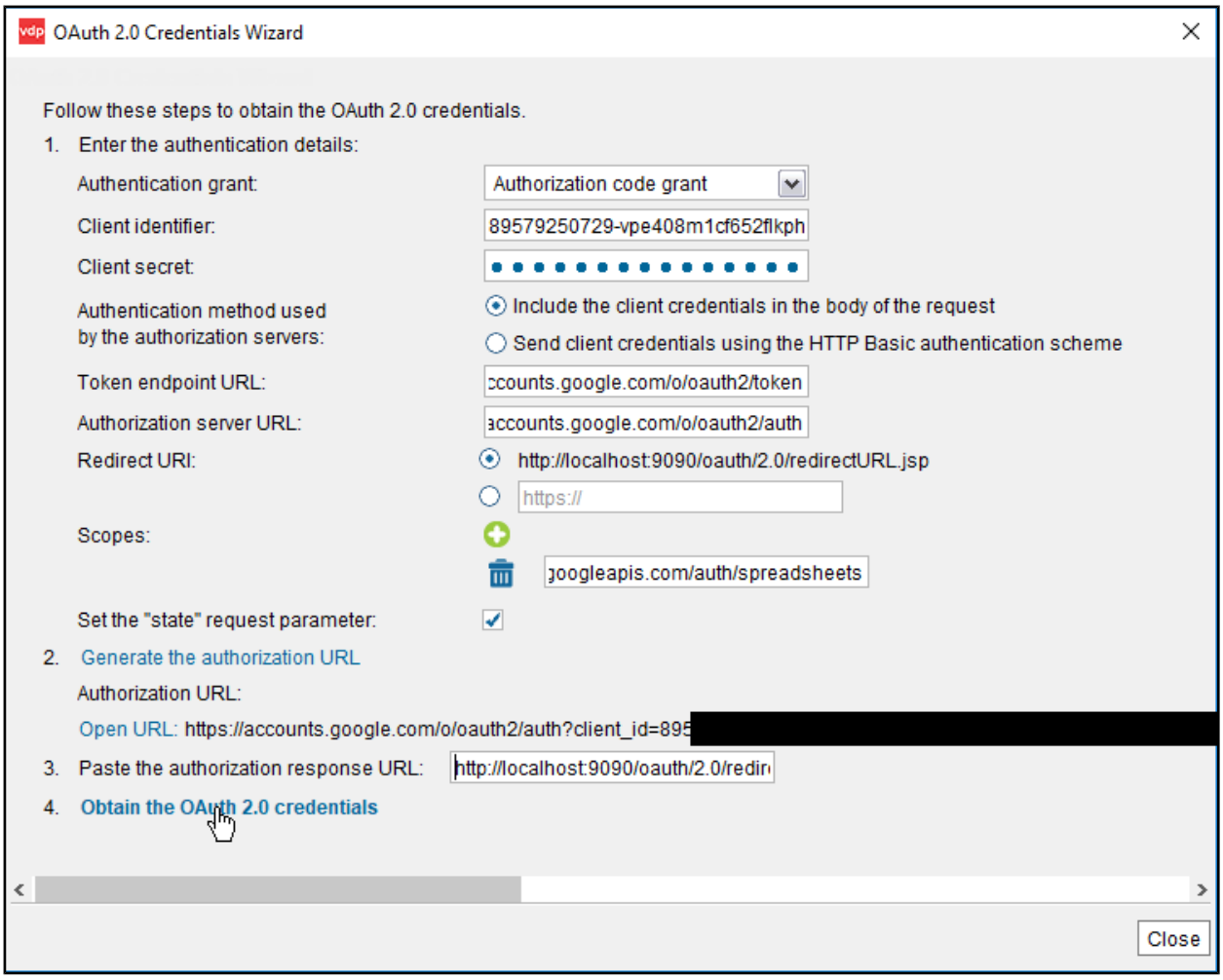


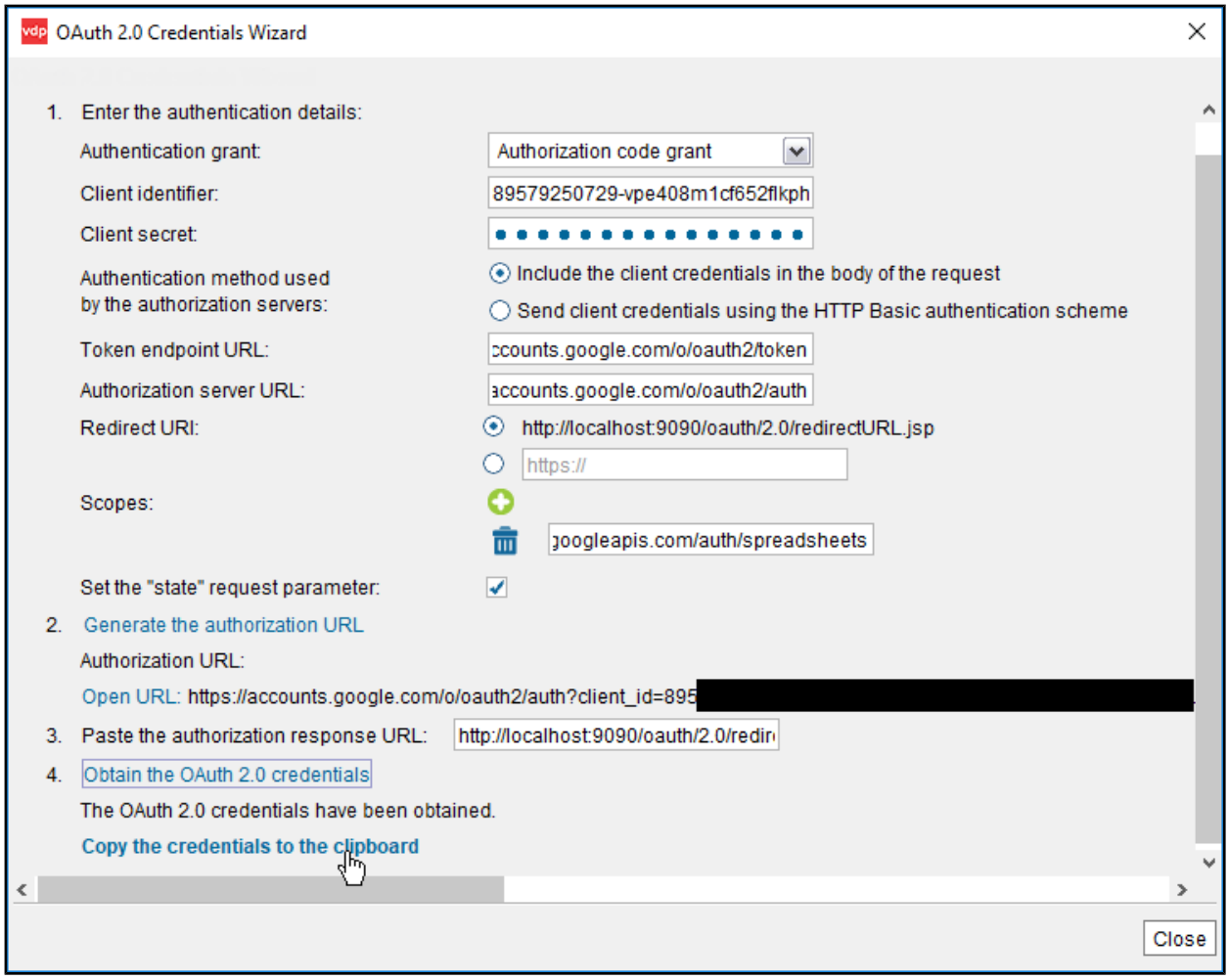
The wizard generates a URL. Click in Open URL and a webpage will be displayed:





As shown, you now have to copy the URL generated and paste it into step number 3. Now click on `Obtain the OAuth 2.0 credentials` and then click on `Copy the credentials to the clipboard`. Paste them into a text file, they will be part of the input parameters of the Denodo Google Sheets Tool Custom Wrapper data source.


OAuth 2.0 Credentials Wizard
Follow these steps to obtain the OAuth 2.0 credentials.
1. Enter the authentication details:
Authentication grant:
Authorization code grant
Client identifier:
89579250729-vpe408m1cf652flkph
Client secret:
Authentication method used by the authorization servers:
Include the client credentials in the body of the request
Send client credentials using the HTTP Basic authentication scheme
Token endpoint URL:
ccounts.google.com/o/oauth2/token
Authorization server URL:
accounts.google.com/o/oauth2/auth
Redirect URI:
http://localhost:9090/oauth/2.0/redirectURL.jsp
https://
Scopes:
googleapis.com/auth/spreadsheets
Set the "state" request parameter:
2. Generate the authorization URL
Authorization URL:
Open URL: https://accounts.google.com/o/oauth2/auth?client_id=895
3. Paste the authorization response URL:
http://localhost:9090/oauth/2.0/redir
4. Obtain the OAuth 2.0 credentials
Close



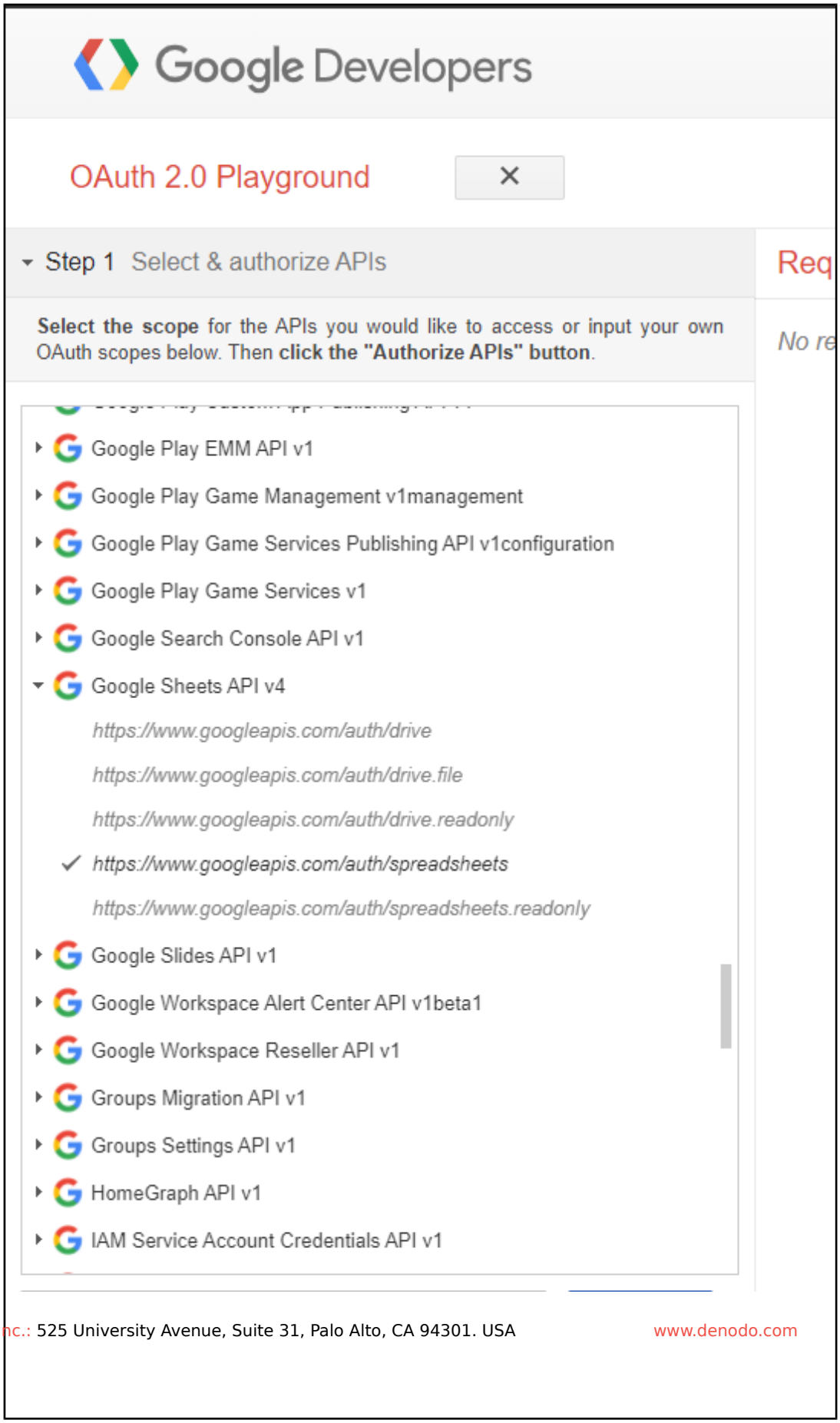
2.1.3 Alternative method: Google OAuth 2.0 Playground

As a first step to get the OAuth2 tokens using the `Google OAuth 2.0 Playground`, you'll need to make sure the Google Sheets API is enabled for the account. This is explained in the **"Google API configuration" section** in step 4.

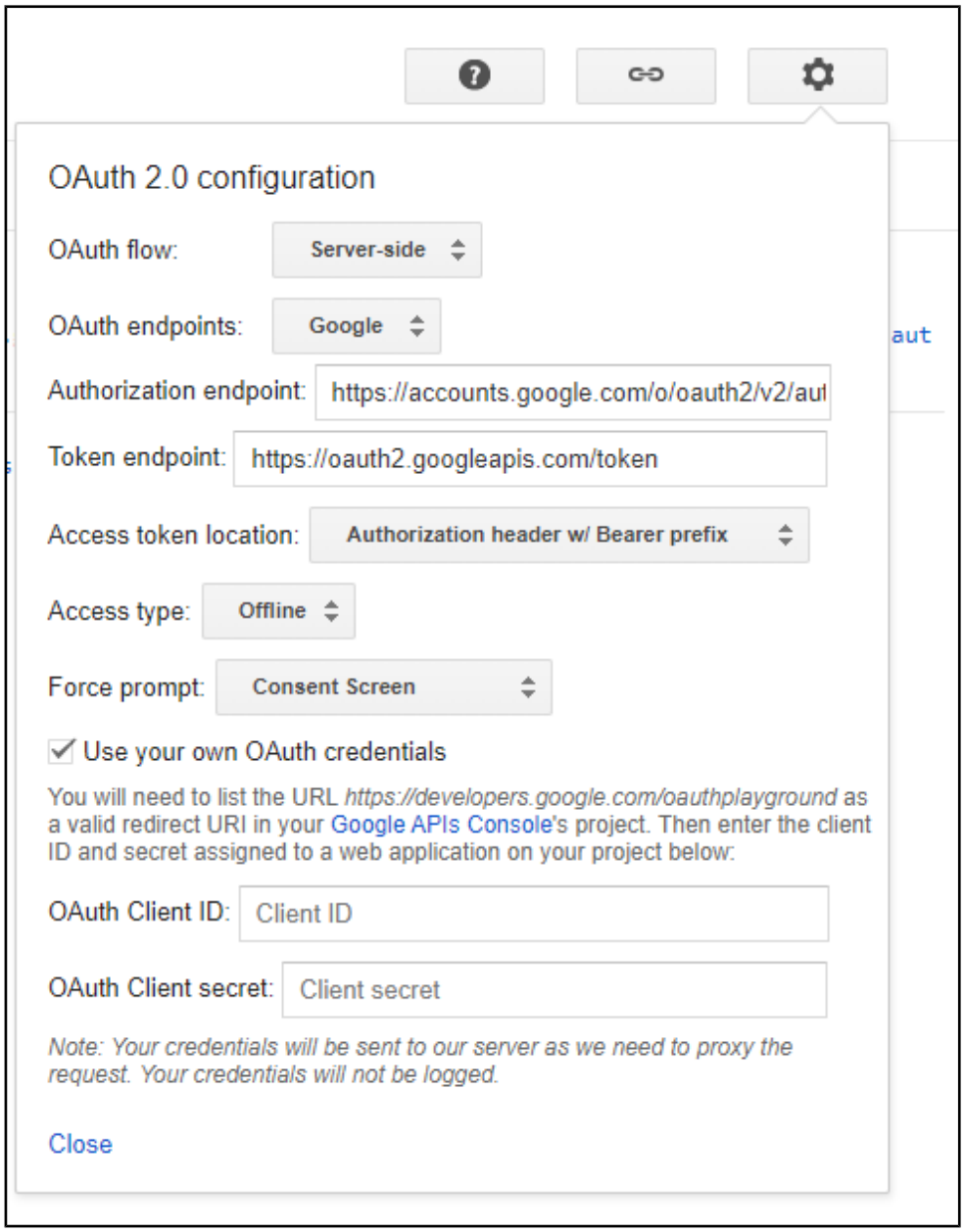
After that, access https://developers.google.com/oauthplayground/ and do the following:

1. Select & authorize APIs: Check the https://www.googleapis.com/auth/spreadsheets option and click on `Authorize`

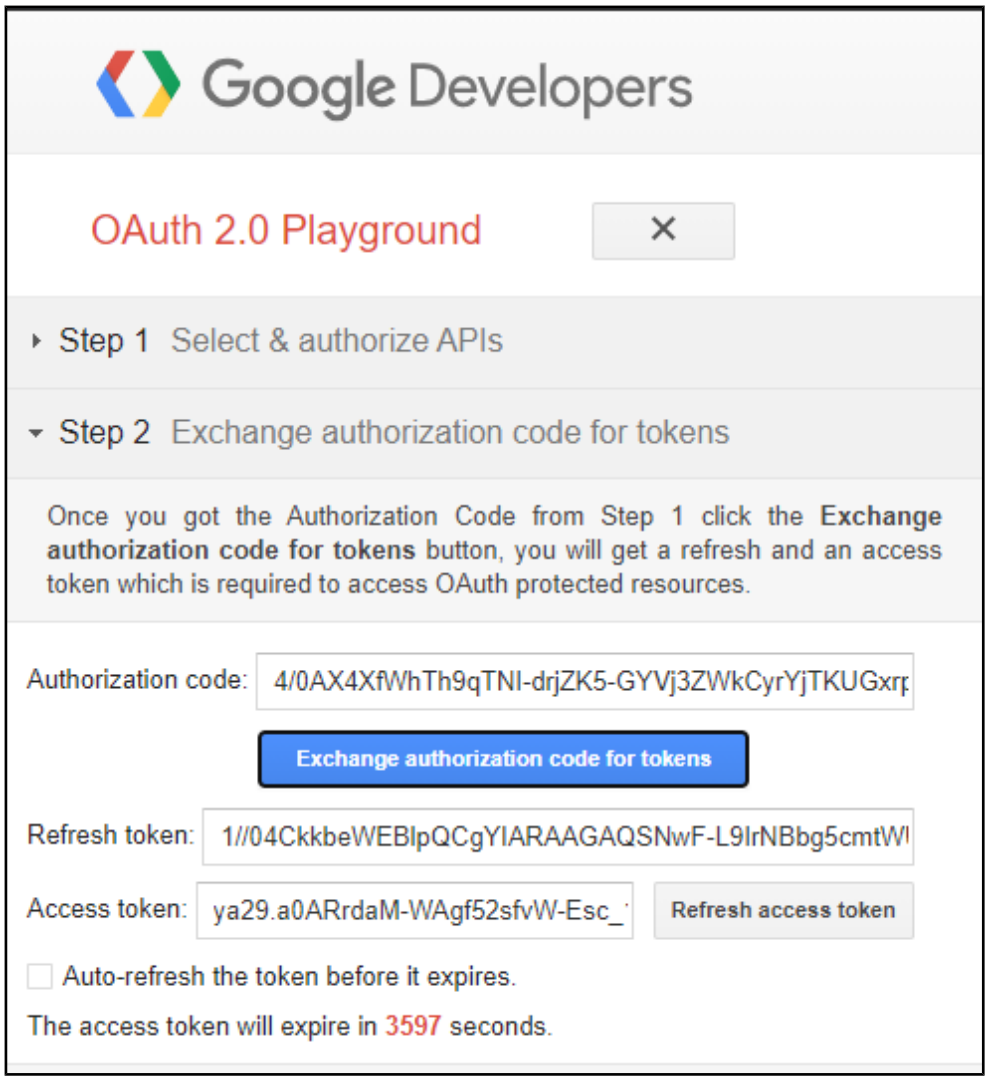
APIs.


Google Developers
OAuth 2.0 Playground
Step 1 Select & authorize APIs
Select the scope for the APIs you would like to access or input your own OAuth scopes below. Then click the "Authorize APIs" button.
Google Play EMM API v1
Google Play Game Management v1management
Google Play Game Services Publishing API v1configuration
Google Play Game Services v1
Google Search Console API v1
Google Sheets API v4
https://www.googleapis.com/auth/drive
https://www.googleapis.com/auth/drive.file
https://www.googleapis.com/auth/drive.readonly
https://www.googleapis.com/auth/spreadsheets
https://www.googleapis.com/auth/spreadsheets.readonly
Google Slides API v1
Google Workspace Alert Center API v1beta1
Google Workspace Reseller API v1
Groups Migration API v1
Groups Settings API v1
HomeGraph API v1
IAM Service Account Credentials API v1

Note that you should check the Use your own OAuth credentials checkbox under OAuth 2.0 configuration options, as well as fill the OAuth Client ID and OAuth Client secret parameters with the val



2. Google will ask you to select the account you're going to use to authenticate to the service. Select the desired account and accept the permissions request.
3. You'll be redirected back to the Google OAuth 2.0 Playground with an Authorization Code. Now click on the Exchange authorization code for tokens button. The application will generate, finally, an access token and a refresh token.


Google Developers
OAuth 2.0 Playground
Step 1 Select & authorize APIs
Step 2 Exchange authorization code for tokens
Once you got the Authorization Code from Step 1 click the Exchange authorization code for tokens button, you will get a refresh and an access token which is required to access OAuth protected resources.
Authorization code: 4/0AX4XfWhTh9qTNI-drjZK5-GYVj3ZWkCyrYjTKUGxr
Exchange authorization code for tokens
Refresh token: 1//04CkkbeWEBlpQCgYIARAAGAQSNwF-L9IrNBbg5cmtW
Access token: ya29.a0ARrdaM-WAgf52sfvW-Esc_
Refresh access token
Auto-refresh the token before it expires.
The access token will expire in 3597 seconds.

3 INSTALLATION

The Denodo Google Sheets Custom Wrapper distribution consists of:

- `/dist`:
  - `denodo-google-sheets-custom-wrapper-{denodo-version}-{version}.jar`. The custom wrapper.
  - `denodo-google-sheets-custom-wrapper-{denodo-version}-{version}-jar-with-dependencies.jar`. The custom wrapper plus its dependencies. This is the wrapper we recommend to use, as it is easier to install in VDP.
  - `denodo-google-sheets-custom-wrapper-{denodo-version}-{version}-sources`. The custom wrapper source code.

3.1 IMPORT THE CUSTOM WRAPPER

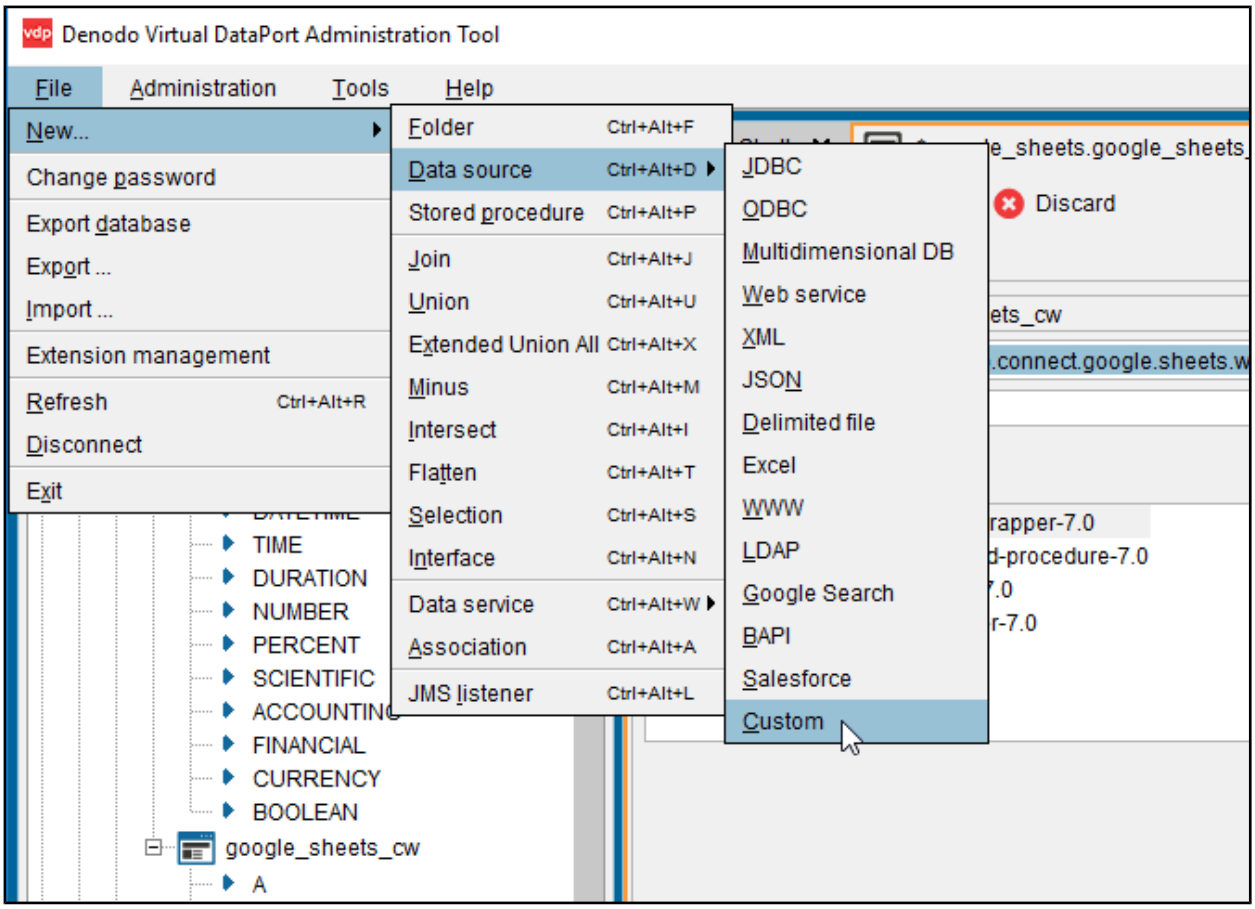
To import the custom wrapper, follow these steps:

1. In the VDP Administration Tool, go to:
   - Denodo 6.0: `File → Jar management`
   - From Denodo 7.0: `File → Extension management`
2. Click on "`Create`" button and select the "`denodo-google-sheets-custom-wrapper-{denodo-version}-{version}-jar-with-dependencies.jar`" file, located in the `dist` folder of the Denodo Google Sheets Custom Wrapper distribution, downloaded from the Denodo Support Site.

3.2 CREATE THE GOOGLE SHEETS DATA SOURCE

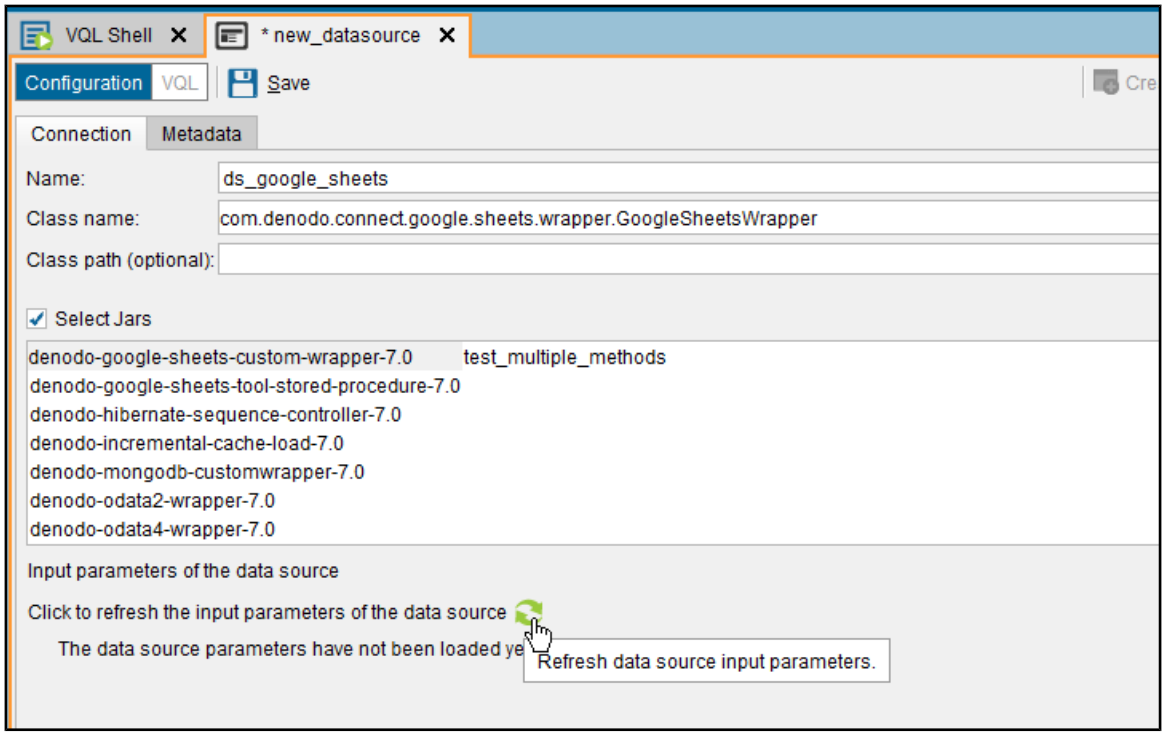
To create a new Google Sheets custom data source:

1. In the VDP Administration Tool, go to: `File` → `New…` → `Data source` → `Custom`

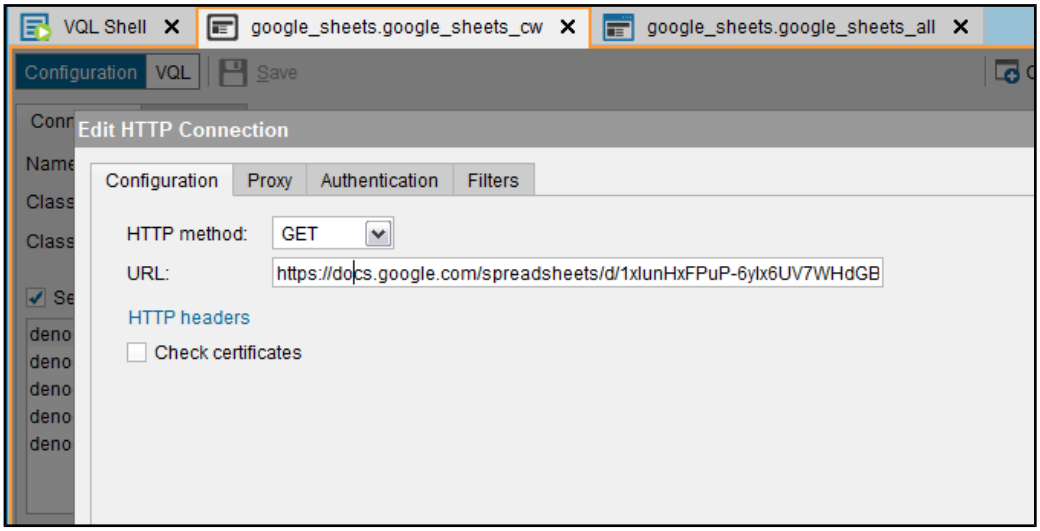


2. In the form displayed, do the following:

   - Set a name for the new Google Sheets data source in the "`Name`" field.
   - Click on "`Select Jars`" and select the file imported in the previous section.
   - The "`Class name`" field must be filled with: `com.denodo.connect.google.sheets.wrapper.GoogleSheetsWrapper`
   - Click on the `Refresh` button. The `Route` parameter will be displayed.



- To configure the `Route` parameter:
  - **`Configuration`**:



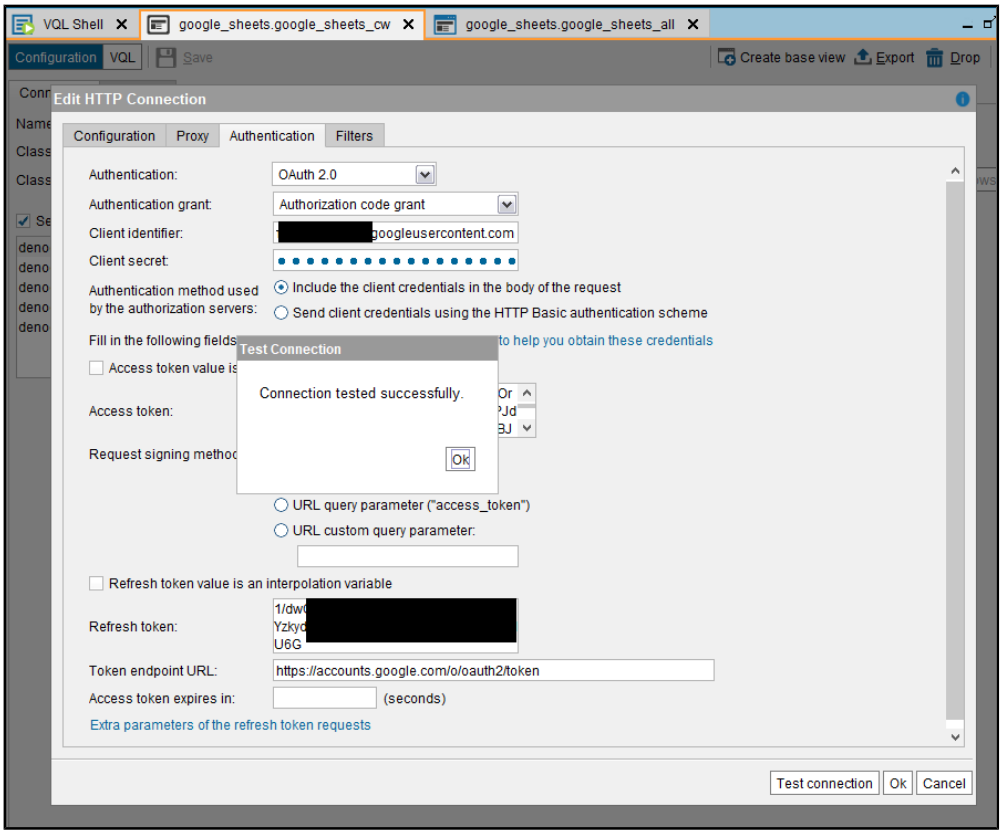
    i. **`HTTP method`** (mandatory): GET.
    ii. **`URL`** (mandatory): Here you have to insert the URL used to edit the Google sheet. You can get it directly by copying and pasting it from the browser navigation bar. Note that you should copy only the part of the URL until the identifier. The rest of the url should be discarded.

    One example of spreadsheet URL could be: https://docs.google.com/spreadsheets/d/<spreadsheet_id>

/edit#gid=0.

What you should set for the `URL` input parameter would be: https://docs.google.com/spreadsheets/d/<spreadsheet_id>/

- **`Authentication`**:
  i. If the spreadsheet is set to public and no permissions are required to access it, you can set Authentication to "`Off`" (the default option). But if the spreadsheet is protected you'll have to configure the authentication as follows, using the data generated in the **"Generate OAuth2 tokens in VDP section"**.



1. **`Authentication`**: OAuth 2.0
2. **`Authentication grant`**: Authentication code grant.
3. **`Client identifier`**: Generated in the **"Google API configuration" section**.
4. **`Client secret`**: Generated in the **"Google API configuration" section**.
5. **`Authentication method`**: Include the client credentials in the body of the request.
6. **`Access token`**: Generated in the **"Generate OAuth2 tokens in VDP" section**.
7. **`Request signing method`**: "Authorization" request header.

8. **Refresh token**: Generated in the **"Generate OAuth2 tokens in VDP" section**.
9. **Token endpoint URL**: https://accounts.google.com/o/oauth2/token.

   Note that, if you are using the Google OAuth2 Playground tool to obtain the tokens, the token endpoint URL must be: https://oauth2.googleapis.com/token

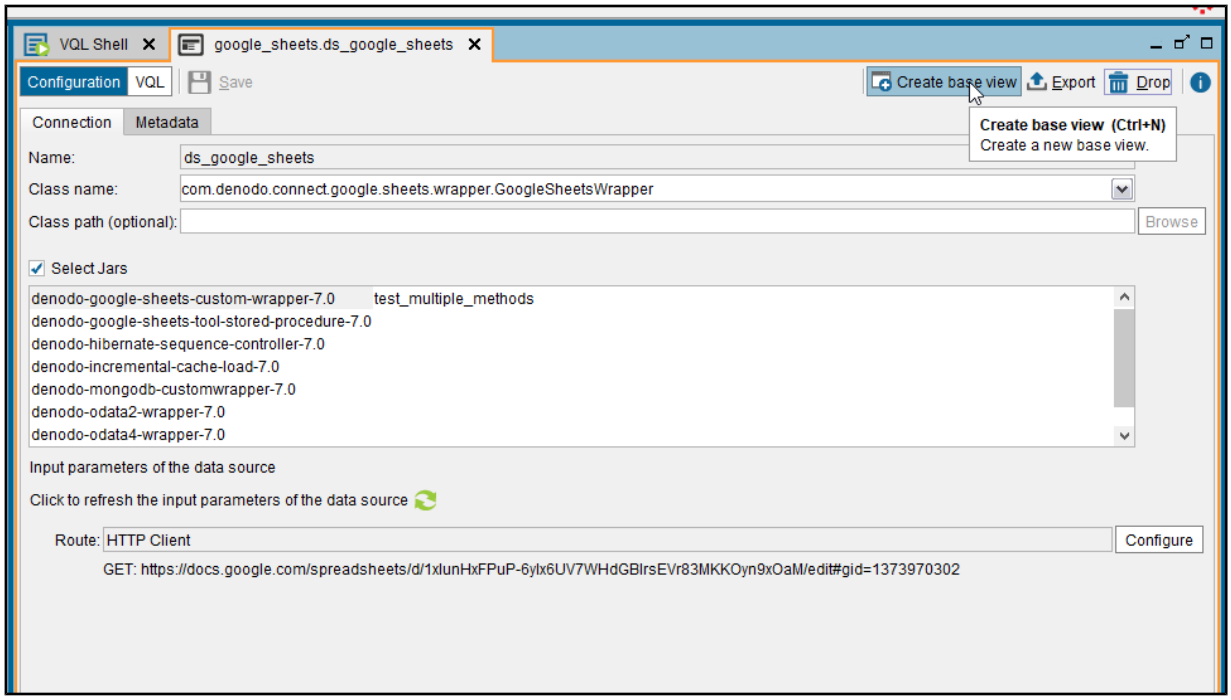
ii. You can test the connection to check if all the parameters were set properly and then click "OK" to close the Route configuration.

3. Click on the "Save" button.

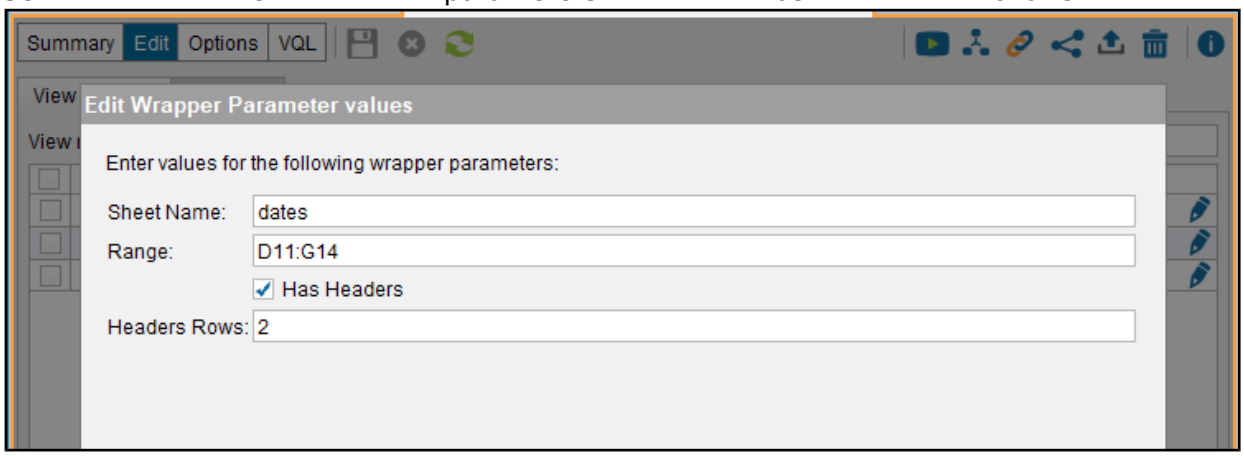
4 USAGE

To create a new base view using the Google Sheets data source created in the previous section:

1. Double-click on the Google Sheets data source and then click on "`Create base view`".



2. Set the parameters as follows:

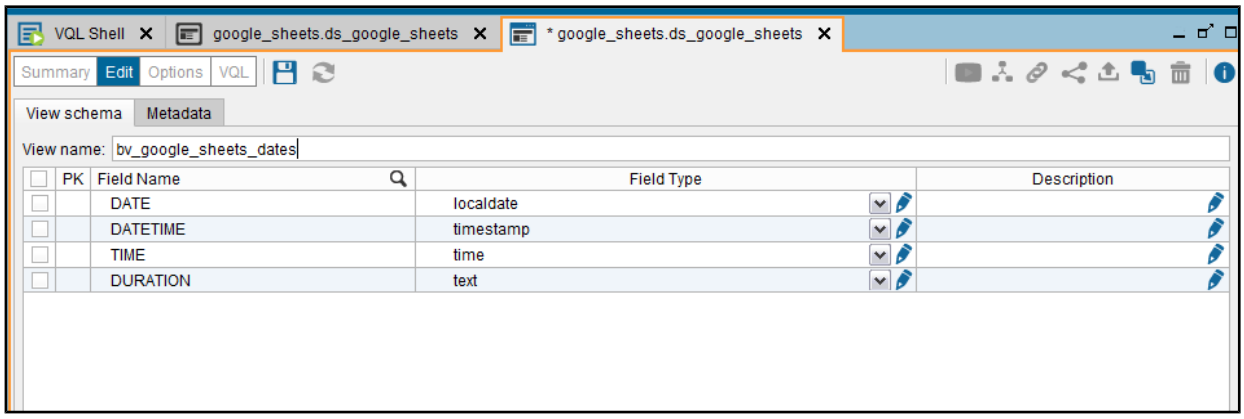


   - **`Sheet Name`** (mandatory): The name of the sheet of the desired Google spreadsheet.
   - **`Range`** (optional): By default, the Custom Wrapper will read all the columns with data in the sheet. In this parameter you can set the specific

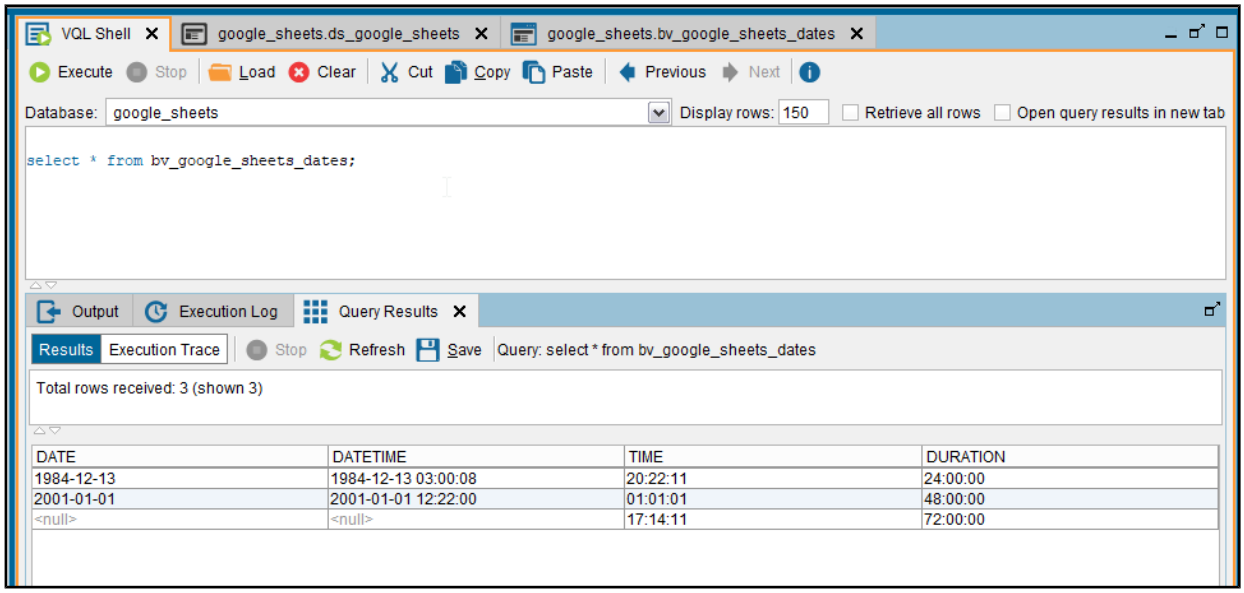
range of cells that will be retrieved in the base view. You can find more information about this parameter in the **"Range parameter" section**.

- **Has Headers** (optional): If checked, the names of the columns of the base view will be taken from the first row of the Google sheet.
- **Headers Rows** (optional): If the Has Headers parameter is checked, you can specify manually the number of rows the columns headers have.

3. Now you can see the new base view. You can configure the last properties, such as the view name, and then click on "Save".



4. Once saved, you can start querying the new base view. For example, `select * from bv_google_sheets_dates`:



4.1 RANGE PARAMETER

As said before, with the `Range` parameter you can specify the exact portion of the sheet that you want for your base view. These are some examples of ranges that you can define:

- A1:B10 - A range from cell A1 through B10
- 5:7 - Rows 5-7
- D:F - Columns D-F
- A:A70 - The first 70 cells in column A
- A70:A - Column A from row 70 to the end
- B5:5 - B5 to the end of row 5
- D3:D - D3 to the end of column D
- C:C10 - From the beginning of column C to C10

Note that with this parameter you can get only the cells you want for one view. But you are also able to, for example, get different tables from the same sheet.

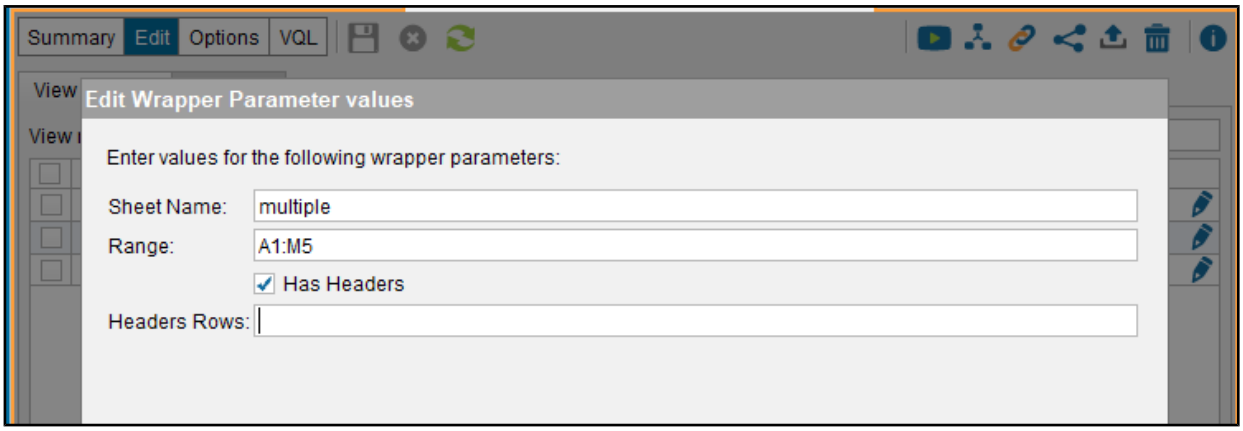
4.1.1 Example: Multiple tables in one sheet

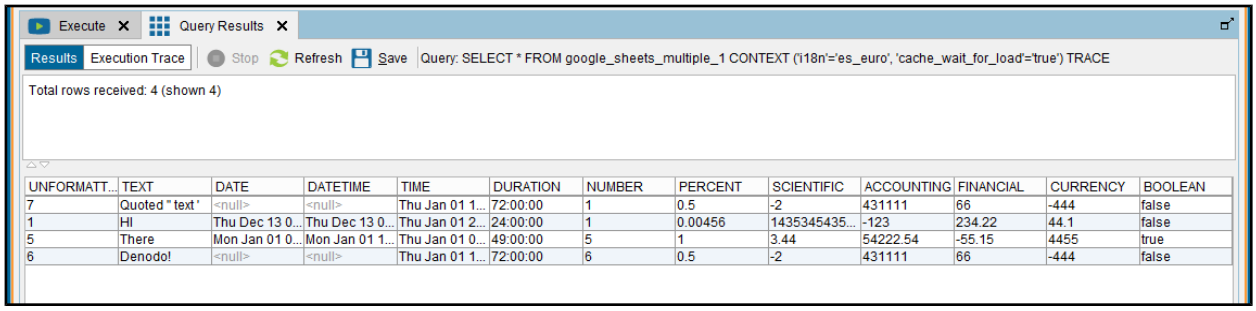
Let's consider this sheet, called `Multiple`, with two tables inside it:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	UNFORMATTED	TEXT	DATE	DATETIME	TIME	DURATION	NUMBER	PERCENT	SCIENTIFIC	ACCOUNTING	FINANCIAL	CURRENCY	BOOLEAN
2	7	Quoted " text '			17:14:11	72:00:00	1,00	50,00%	-2,00E+00	431.111,00 €	66,00	-444 €	☐
3	1	HI	13/12/1984	13/12/1984 3:00:08	20:22:11	24:00:00	1,00	0,46%	1,44E+13	- 123,00 €	234,22	44 €	☐
4	5	There	1/01/2001	1/01/2001 12:22:00	1:01:01	49:00:00	5,00	100,00%	3,44E+00	54.222,54 €	(55,15)	4.455 €	☑
5	6	Denodo!			17:14:11	72:00:00	6,00	50,00%	-2,00E+00	431.111,00 €	66,00	-444 €	☐
6													
7													
8													
9													
10													
11				DATE	DATETIME	TIME	DURATION						
12				13/12/1984	13/12/1984 3:00:08	20:22:11	24:00:00						
13				1/01/2001	1/01/2001 12:22:00	1:01:01	48:00:00						
14						17:14:11	72:00:00						
15													

We can have two different base views by setting up the right range in the base view creation:

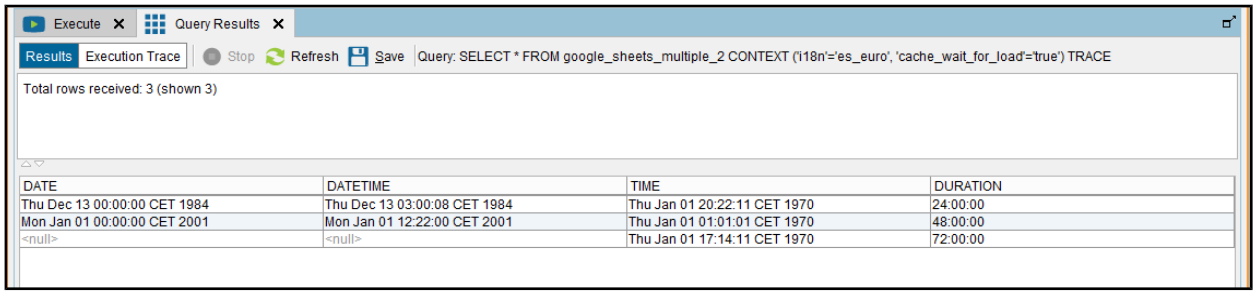
- BV 1:





UNFORMATT...	TEXT	DATE	DATETIME	TIME	DURATION	NUMBER	PERCENT	SCIENTIFIC	ACCOUNTING	FINANCIAL	CURRENCY	BOOLEAN
7	Quoted " text "	<null>	<null>	Thu Jan 01 1...	72:00:00	1	0.5	-2	431111	66	-444	false
1	HI	Thu Dec 13 0...	Thu Dec 13 0...	Thu Jan 01 2...	24:00:00	1	0.00456	1435345435...	-123	234.22	44.1	false
5	There	Mon Jan 01 0...	Mon Jan 01 1...	Thu Jan 01 0...	49:00:00	5	1	3.44	54222.54	-55.15	4455	true
6	Denodo!	<null>	<null>	Thu Jan 01 1...	72:00:00	6	0.5	-2	431111	66	-444	false

- BV 2:

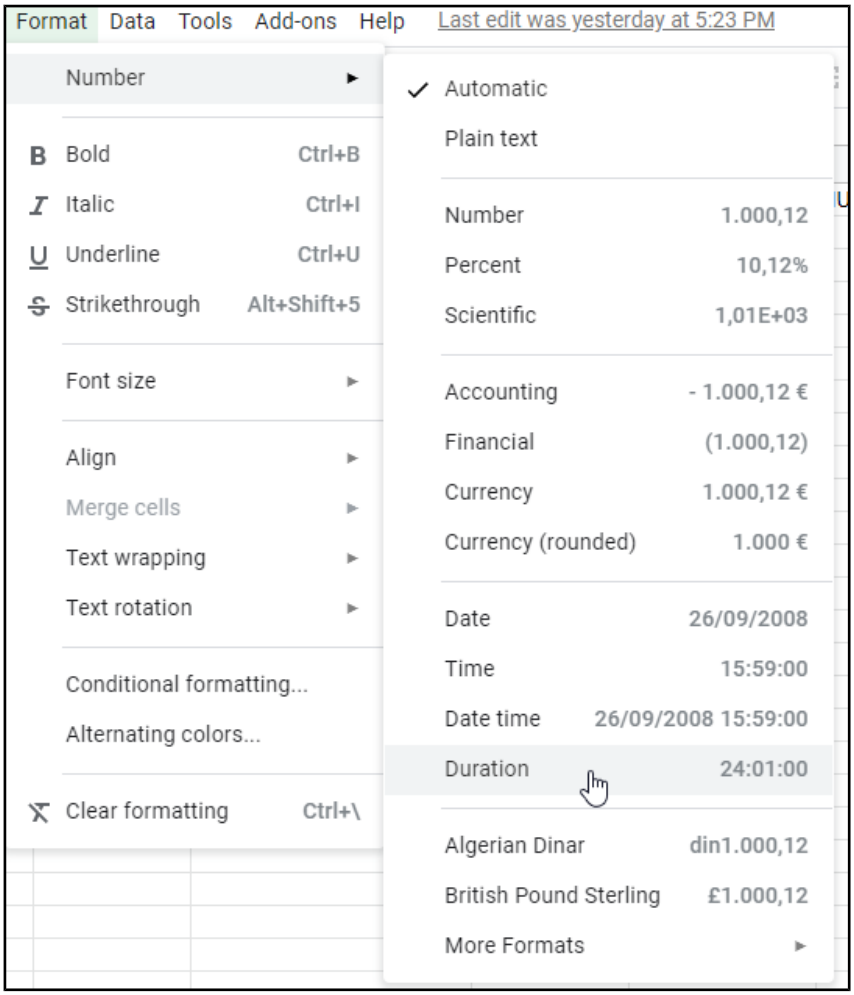


DATE	DATETIME	TIME	DURATION
Thu Dec 13 00:00:00 CET 1984	Thu Dec 13 03:00:08 CET 1984	Thu Jan 01 20:22:11 CET 1970	24:00:00
Mon Jan 01 00:00:00 CET 2001	Mon Jan 01 12:22:00 CET 2001	Thu Jan 01 01:01:01 CET 1970	48:00:00
<null>	<null>	Thu Jan 01 17:14:11 CET 1970	72:00:00

5 LIMITATIONS

5.1 TIME AND DURATION TYPES

Google Sheets supports by default these data types:



They are all supported in the Denodo Google Sheets Custom Wrapper, but we have some limitations with the `Time` and `Duration` types:

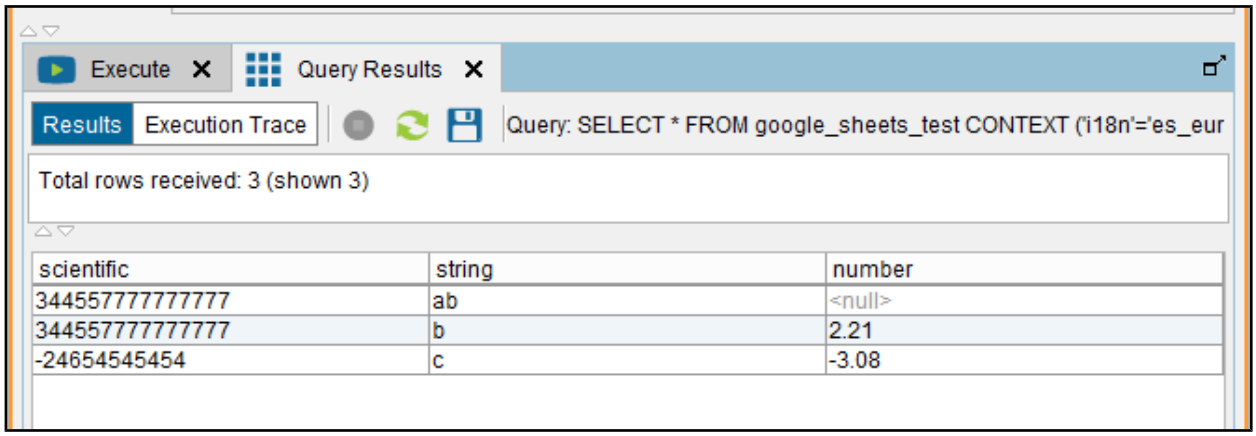
- Denodo 6 and later:
  - `Duration`: In the base views, fields will be defined as `Text` and it's not allowed to filter queries using parameters of this type.
- Denodo 6:
  - `Time`: In the base views, fields will be defined as `Date` and it's not allowed to filter queries using parameters of this type.

5.2 WRONG FORMAT / TYPE IN A COLUMN

In the previous explained limitation we could see the available data types for Google Sheets. Anyway, you can insert values that don't match the data type set for a column directly into the sheet. For example:

A	B	C
SCIENTIFIC	STRING	NUMBER
3,45E+14	ab	a
3,45E+14	b	2,21
-2,47E+10	c	-3,08

The "`NUMBER`" column is defined as `Number` type, but you can insert a `String` as shown. For this kind of situations, the **Google Sheets Custom Wrapper will set that values to null**, as they don't match the type defined for that column in the base view:



5.3 SET A SHEET NAME THAT DOES NOT EXIST

It's mandatory to set a sheet name when you create a base view with this Custom Wrapper. But Google Sheets will not validate if a sheet exists with that name in the spreadsheet, so **if you provide a sheet name that doesn't exist, the API will return the first sheet of the spreadsheet**.

5.4 DELEGATED OPERATORS

Due to restrictions in the underlying API, only the following operators will be delegated to Google Sheets: <=, <, >, >=, =, !=, <>, as well as combinations of these operators using `and`, `or` and `not`. All other filtering combinations will be computed in memory by VDP.

5.5 FILTER VIEWS

Due to restrictions in the underlying API, no support is provided for querying filter views defined in Google Sheets.