Denodo Google Sheets Tool User Manual

Revision 20200514

Denodo North America & APAC: 525 University Avenue, Suite 31, Palo Alto, CA 94301. USA
Denodo Iberia & Latino América: Montalbán 5, 28014 Madrid, Spain
Denodo EMEA: 21st Floor, Portland House, Bressenden Place, London SW1E 5RS. UK
Denodo DACH: Karlstraße 10, 80333 München. Germany
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 Tool Stored Procedure you'll be able to query the data in your spreadsheets directly in Denodo. With a simple call of the procedure, several data sources and views will be created, but the most important one will be a derived view that will represent the data the same way you can see it directly in Google Sheets, along with some custom configuration.

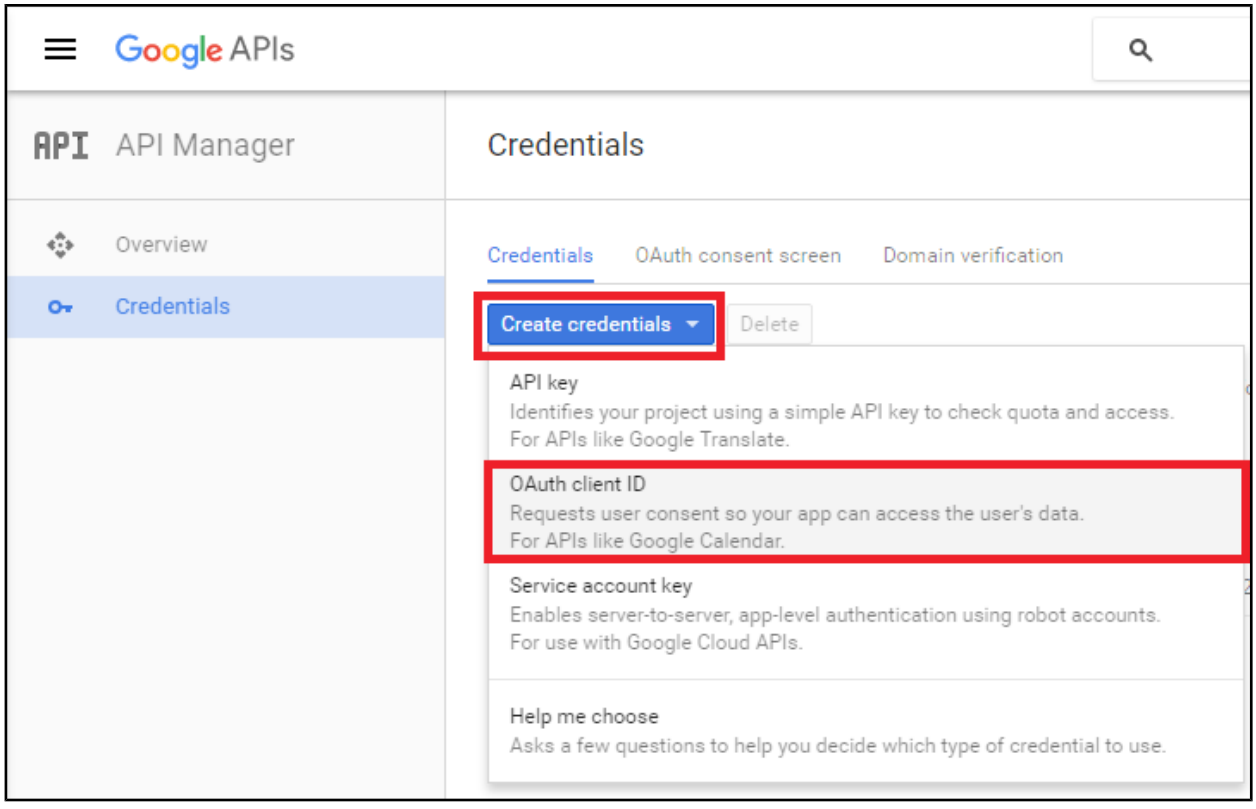
2 REQUIREMENTS

2.1 GET OAUTH2 CREDENTIALS

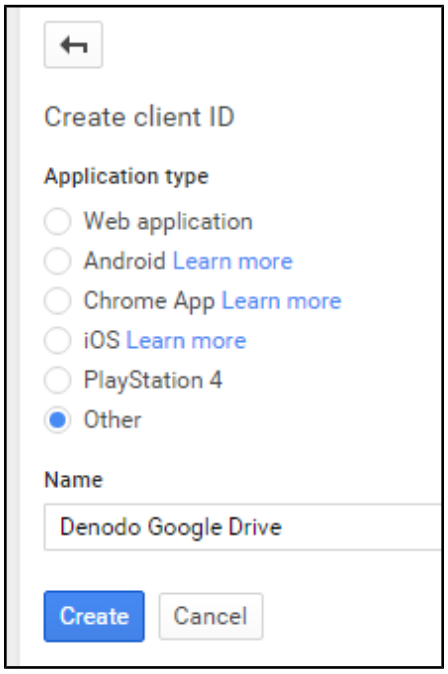
2.1.1 Google API configuration

Google Sheets offers a REST API with capabilities to perform CRUD operations. Before executing the stored procedure, which will create several data sources and views using this API, 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`.



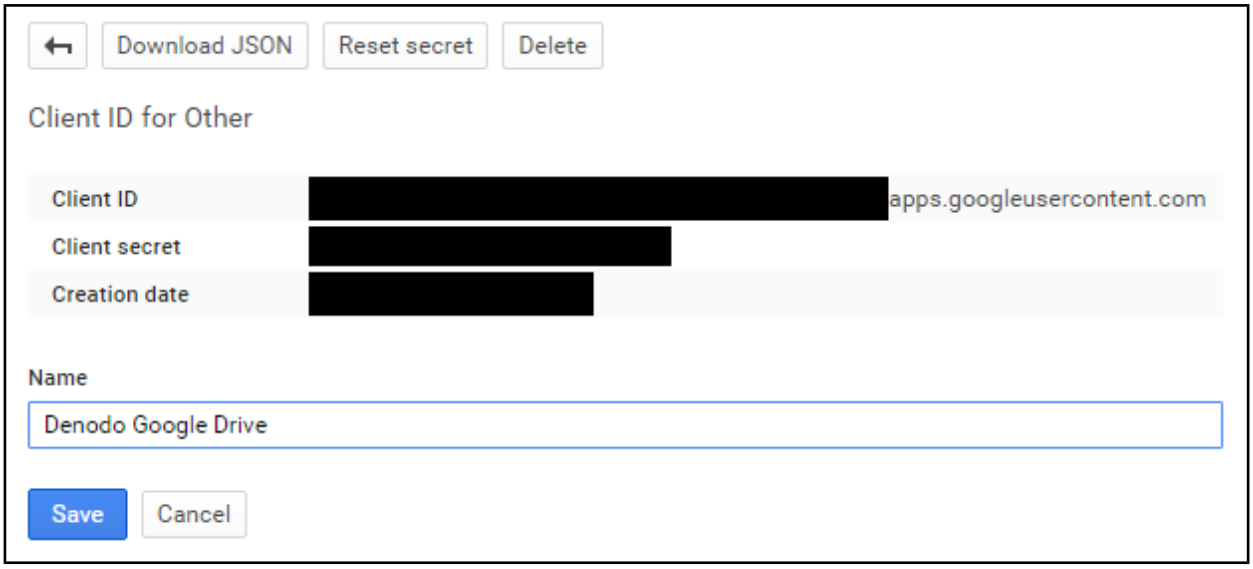
    b. Choose `Application type` : "`Other`" and give it a name.



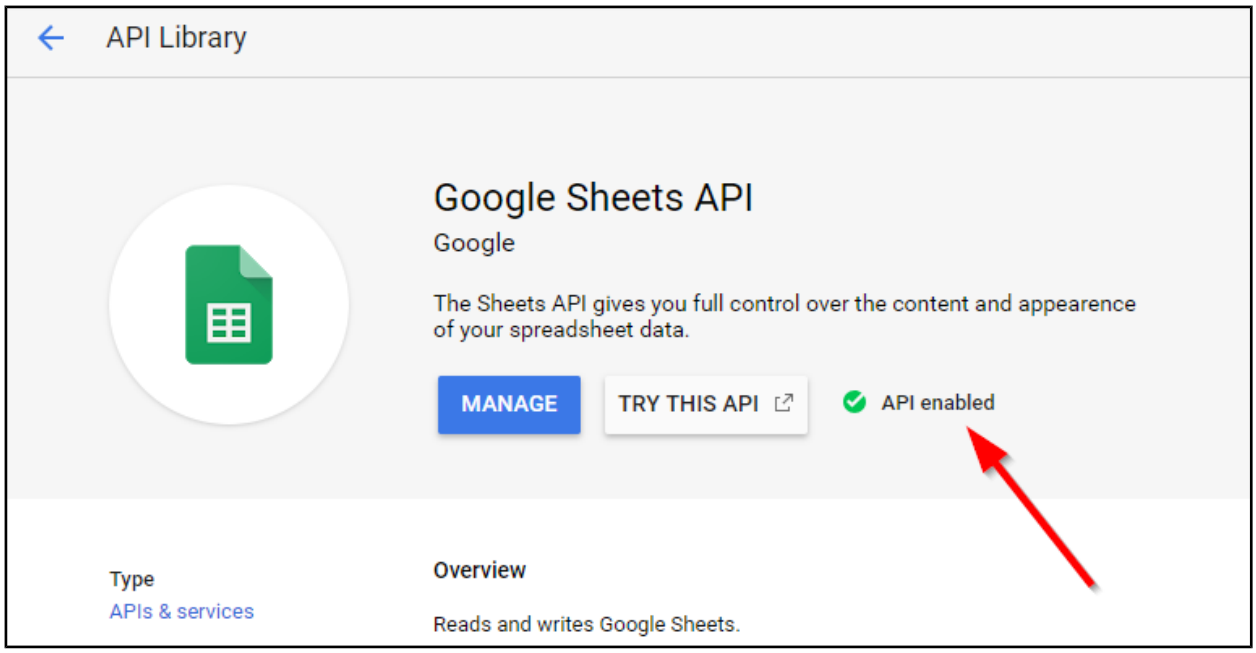
3. Once the Developer account is configured two values will be used for the Oauth authentication, they will be known both to Google and the application:

    a. `Client ID`.

    b. `Client Secret`.

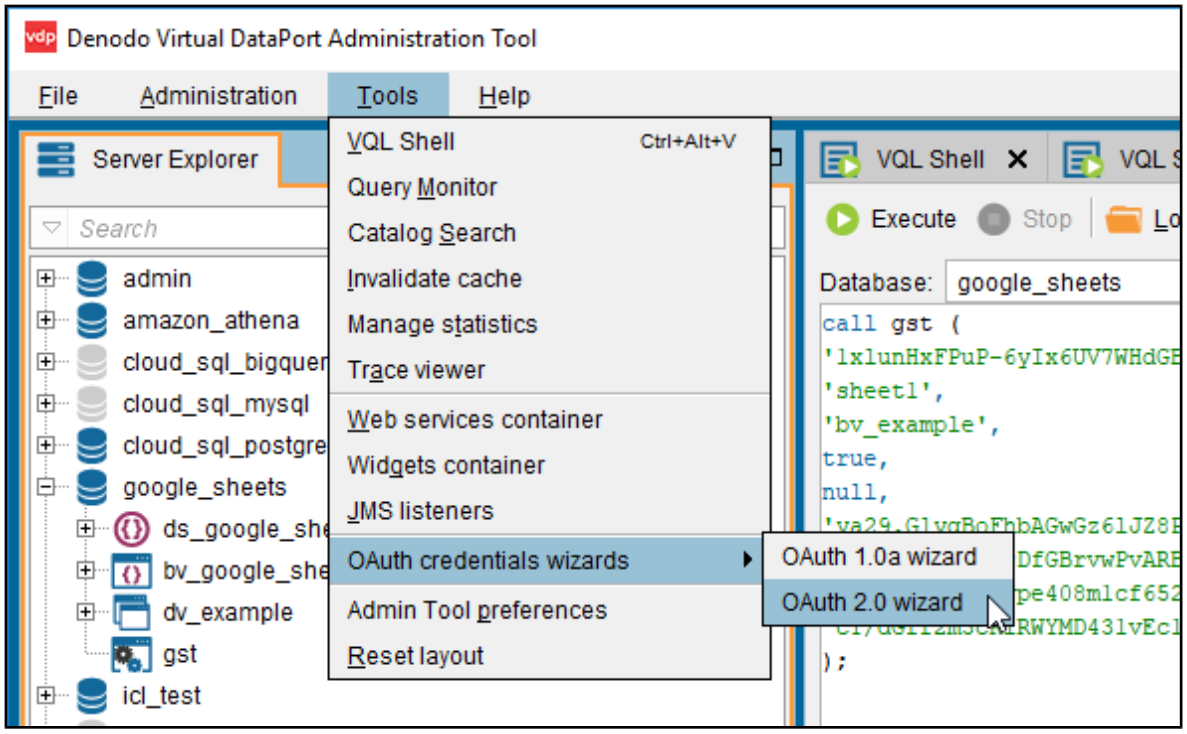


4. Make sure the Google Sheets API is enabled for the account. To do so, go to the `Dashboard` section in the Developer Console and click on `Enable` if needed.



2.1.2 Generate OAuth2 tokens in VDP

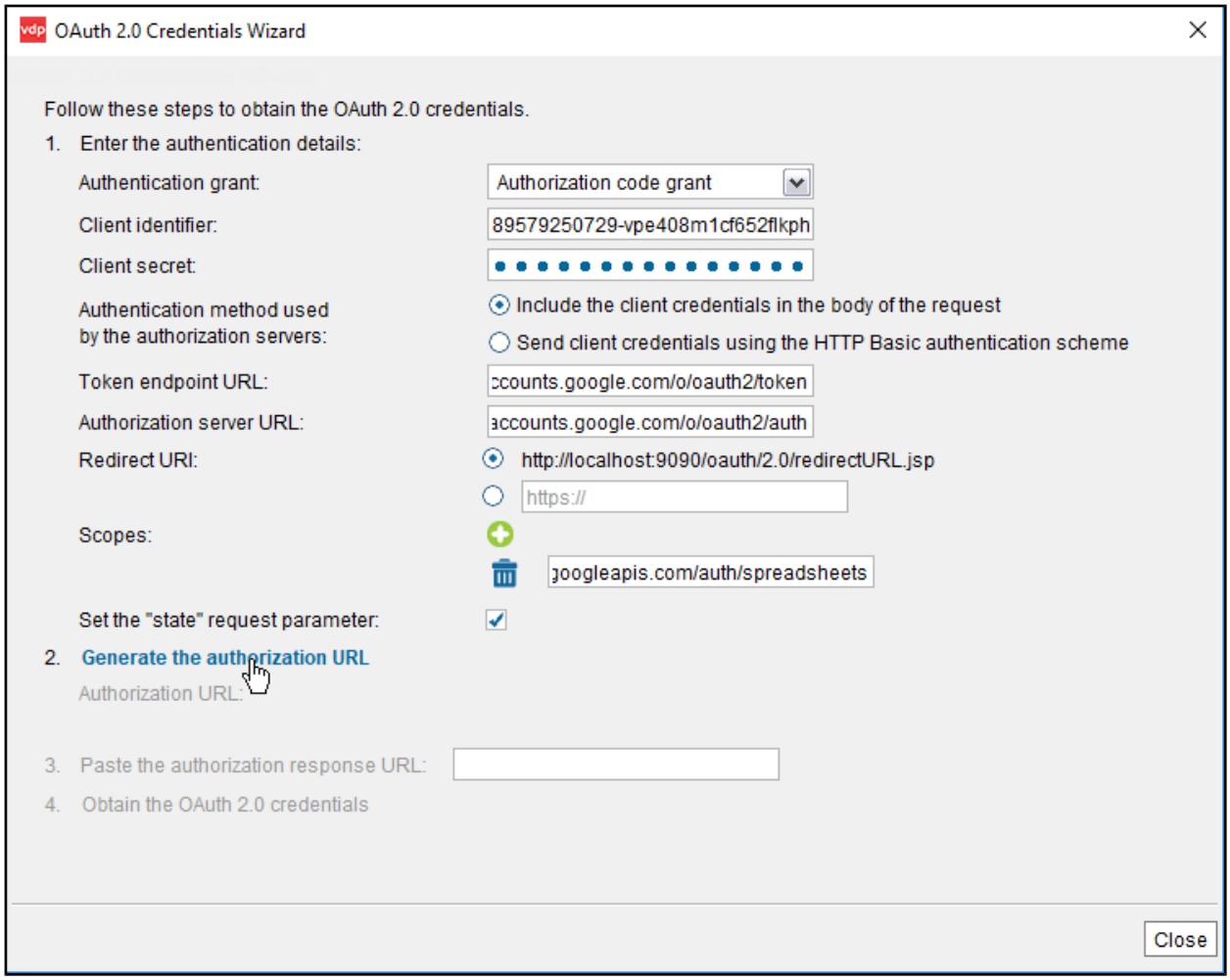
Now, we can create the tokens needed to authenticate the Denodo data sources that the stored procedure will create. To make this, we will use the OAuth credentials wizards from VDP.



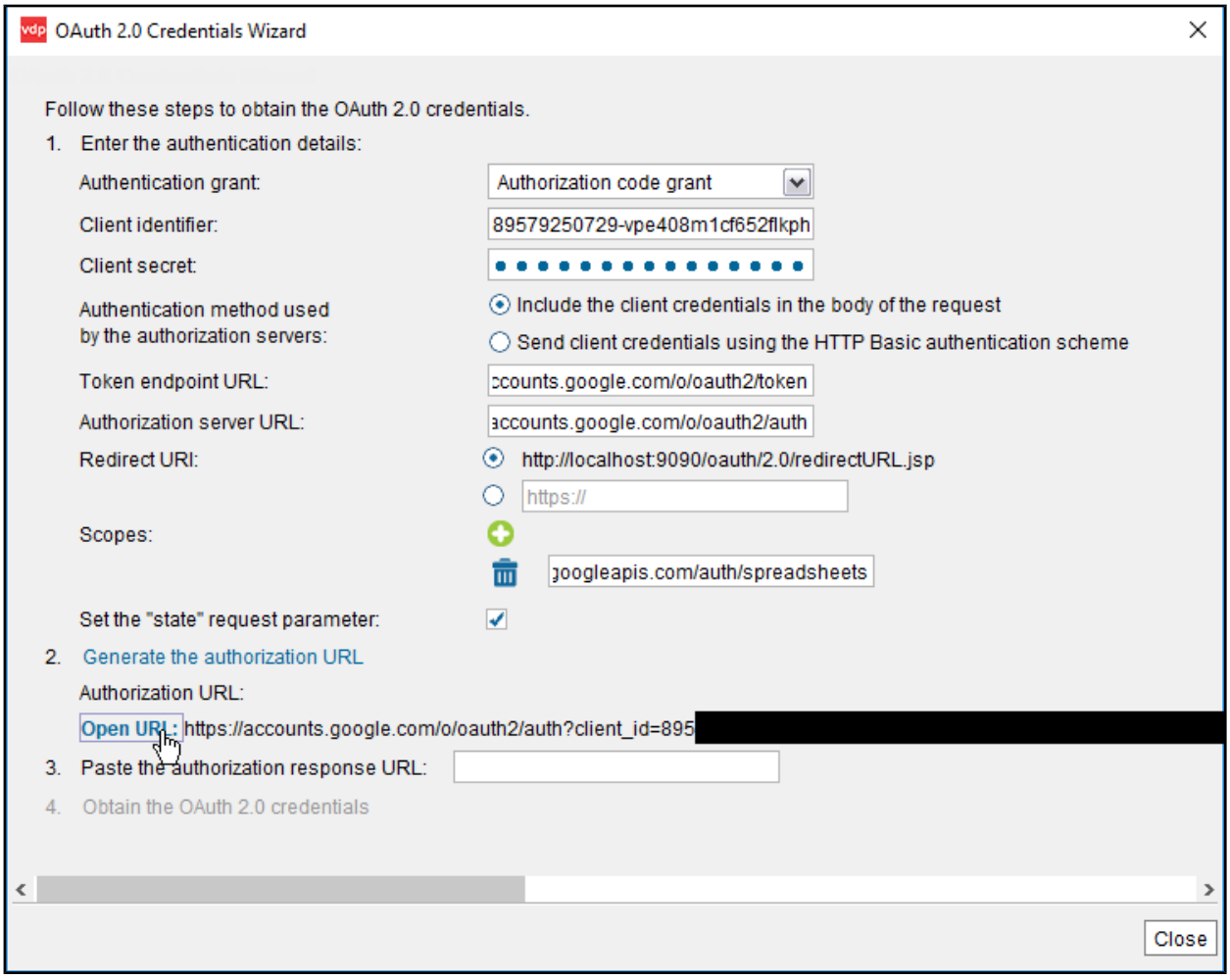
You have to fill in the form with the following parameters:

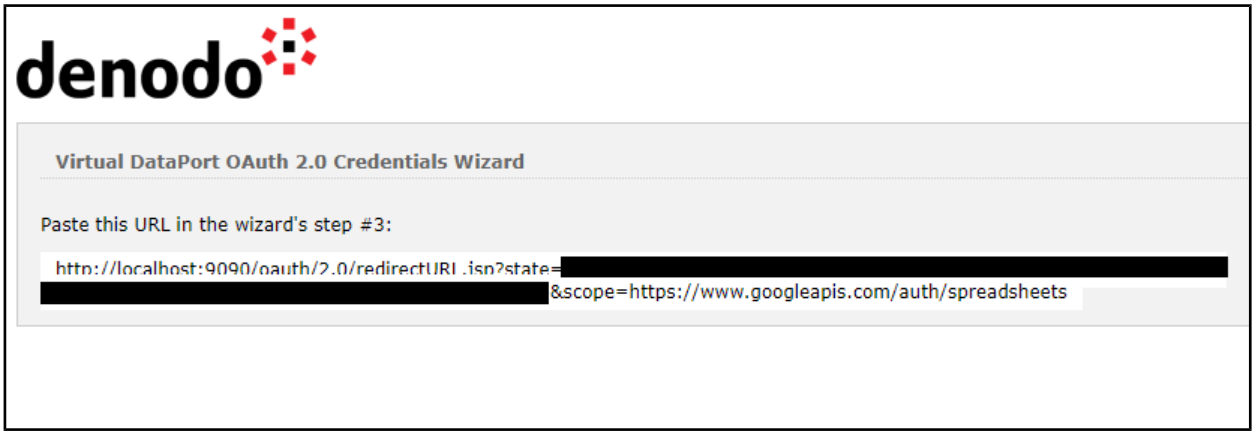
- **`Authentication grant`**: Authorization code grant
- **`Client identifier`**: Obtained from the Credentials section of the Google sheets API
- **`Client secret`**: Obtained from the Credentials section of the Google sheets API
- **`Authentication method`**: Include the client credentials in the body 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`:

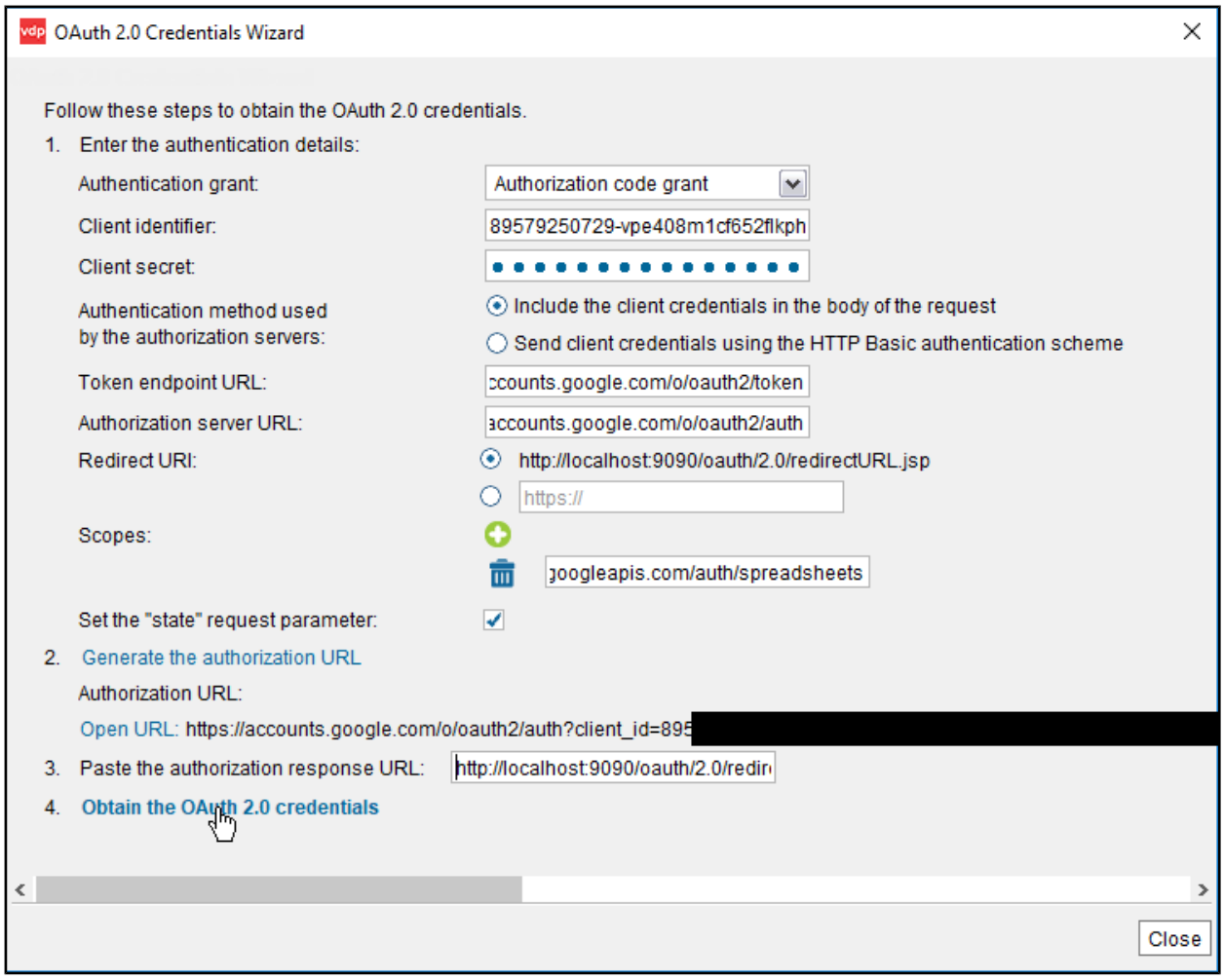


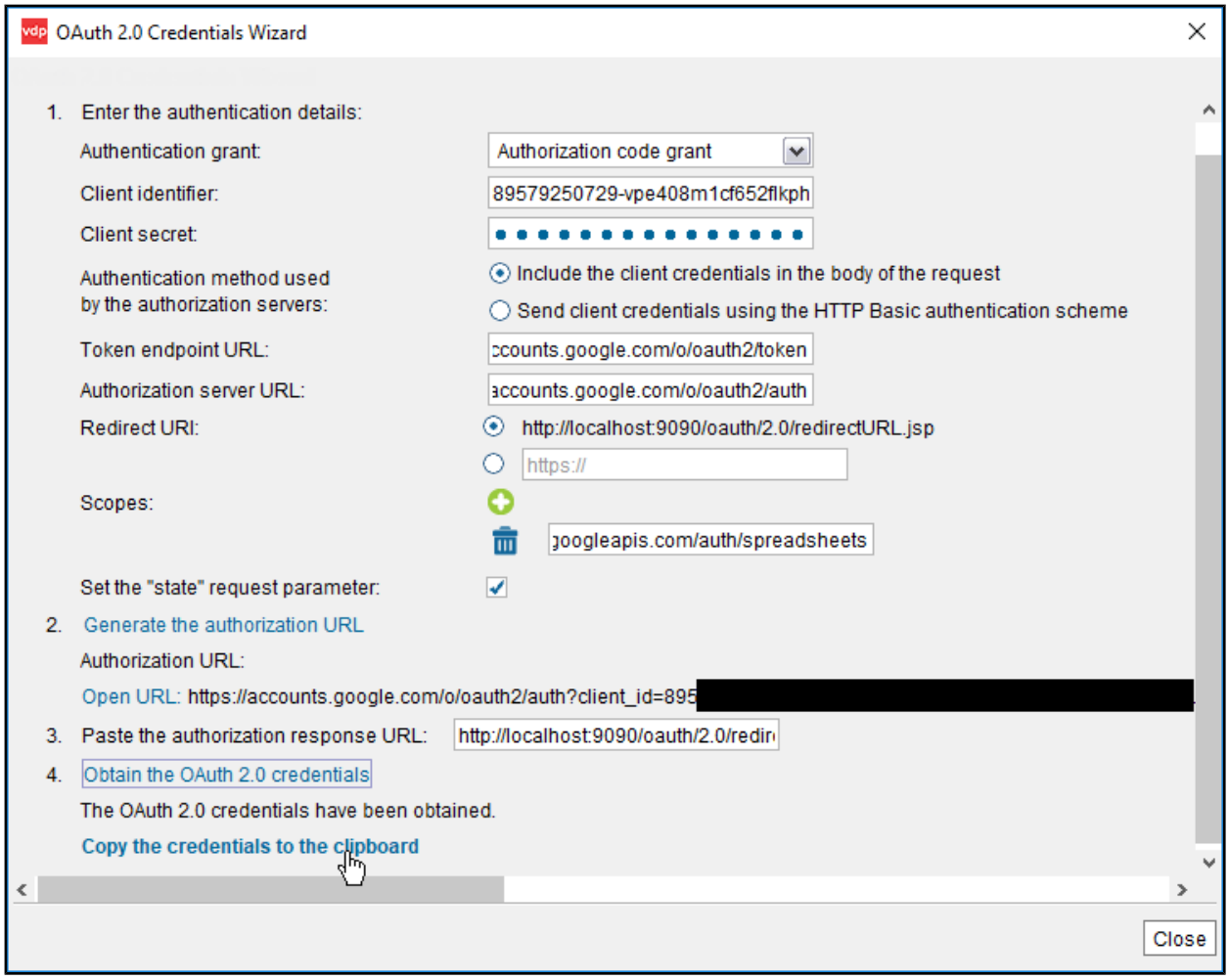
The wizard generates a URL. Click in `Open URL` and a webpage will be displayed:





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


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



Note: It's recommended to generate the tokens in VDP like explained in this section. Using other external tools, including Google's OAuth Playground, may result in not working refresh tokens as they might be generated with a different token endpoint version than the used in VDP.

3 INSTALLATION

3.1 IMPORTING THE STORED PROCEDURE

For running the Denodo Google Sheets Tool Stored Procedure you have to load the `denodo-google-sheets-tool-{vdpversion}-{version}-jar-with-dependencies.jar` file, using the `File > Jar Management` menu of the VDP Administration Tool (or `File > Extension management` in Denodo 7.0 and later).

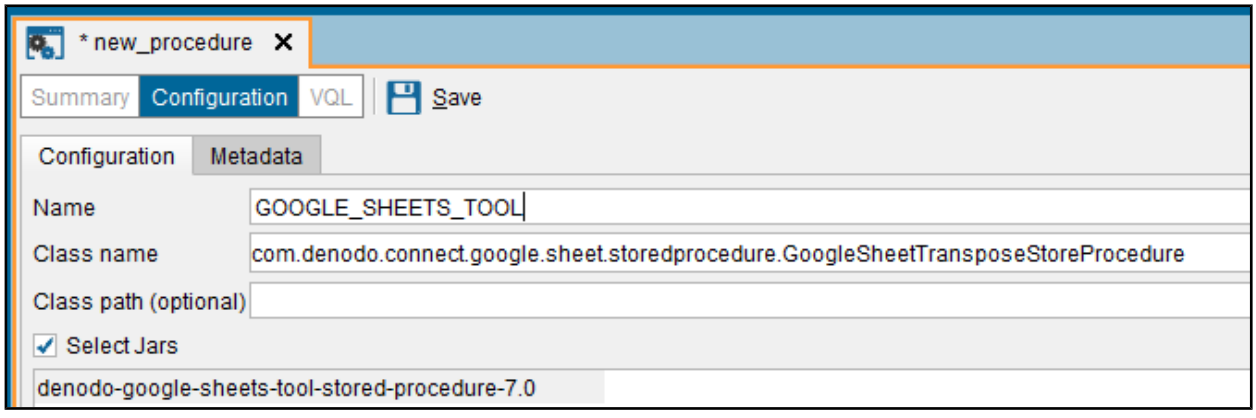
3.2 ADDING THE STORED PROCEDURE: VQL SHELL

You can add the stored procedure with the statement `CREATE PROCEDURE`:

```
CREATE [OR REPLACE] PROCEDURE <name:identifier>
CLASSNAME='com.denodo.connect.google.sheet.storedprocedure.GoogleSheetsTo
olStoreProcedure'
      JARS 'denodo-google-sheets-tool-<vdpversion>';
      [ FOLDER = <literal> ]
      [ DESCRIPTION = <literal> ]
```

3.3 ADDING THE STORED PROCEDURE: VIRTUAL DATAPORT ADMINISTRATION TOOL MENU

Alternatively to using VQL, you can add the stored procedure clicking `Stored procedure` on the menu `File > New`:



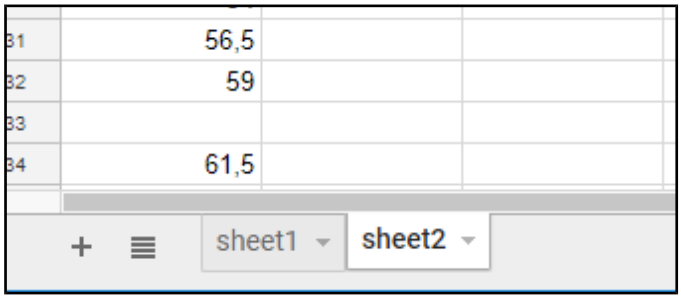
You must set a name in the `Name` field and select the `Select Jars` checkbox in order to use the Jar file, `denodo-google-sheets-tool-stored-procedure-<vdpversion>`.

4 USAGE

The stored procedure requires the following input parameters:

- **SPREADSHEET_ID** (mandatory): the identifier of the Google spreadsheet. You can extract it from the url, which will be something like `https://docs.google.com/spreadsheets/d/<SPREADSHEET_ID>/edit`

- **SHEET_NAME** (mandatory): You'll find it in the tabs of the spreadsheet. In this example, it would be `sheet2`.



- **VIEW NAME** (mandatory): The name you want for the derived view which is going to represent your Google sheet in VDP.

- **FIRST_ROW_COLUMN_NAMES** (mandatory): If set to true, the column names in the view will be extracted from the first row of the Google sheet. If false, they will be named with the same alphabetic sequence used in any spreadsheet.

- **NUM_COLUMNS**: If set to null, the number of columns will be determined by the number of columns with data in the sheet. If a number is specified, the view will have exactly that number of columns.

- **ACCESS TOKEN** (mandatory): Obtained in **"Generate OAuth2 tokens in VDP" section**.

- **REFRESH TOKEN** (mandatory): Obtained in **"Generate OAuth2 tokens in VDP" section**.

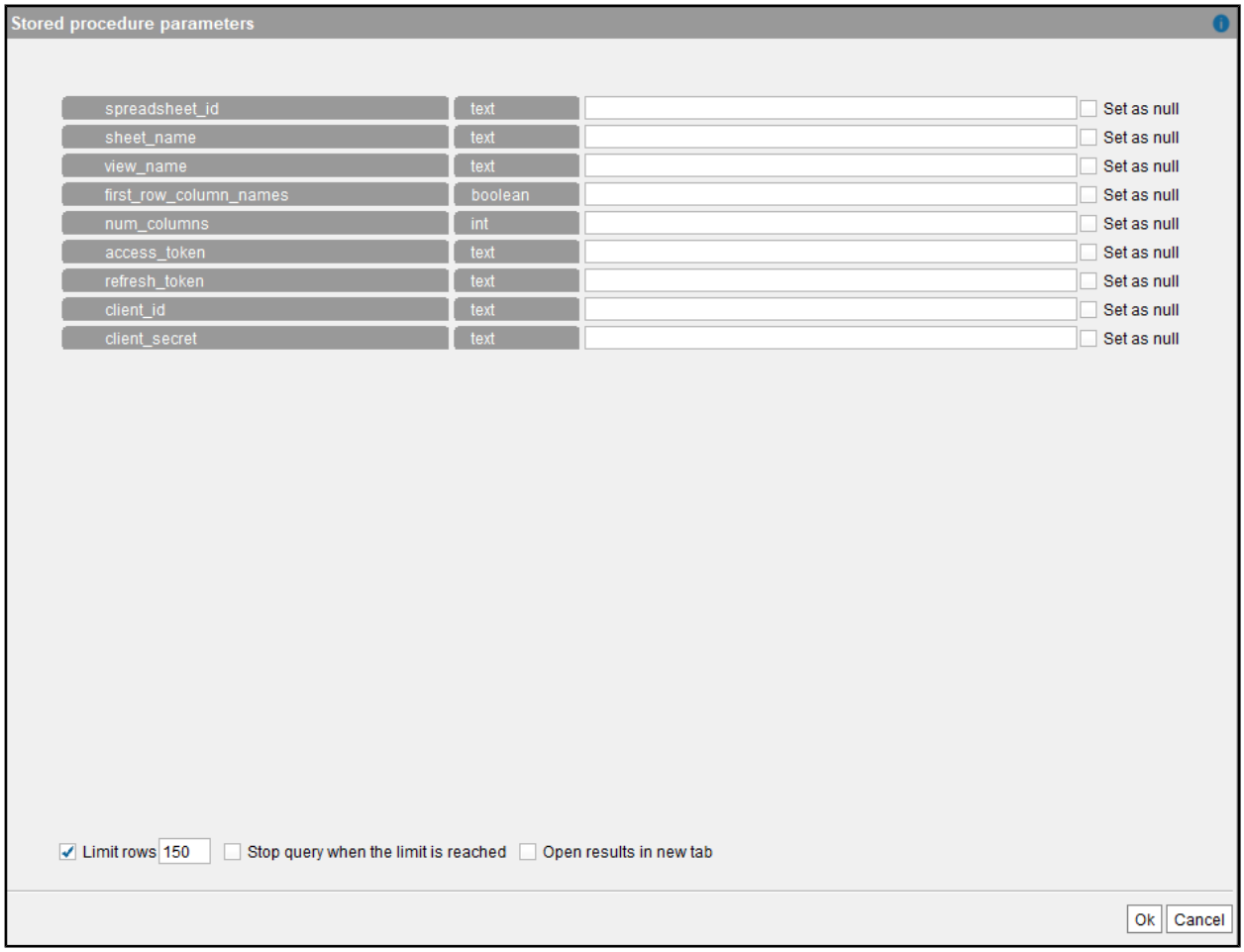
- **CLIENT ID** (mandatory): Obtained in **"Generate OAuth2 tokens in VDP" section**.

- **CLIENT SECRET** (mandatory): Obtained in **"Generate OAuth2 tokens in VDP" section**.

4.1 EXECUTING THE STORED PROCEDURE

There are four possibilities:

1. Click the `Execute` button in the dialog that displays the schema of the stored procedure. The VDP Administration Tool will show a dialog to enter the input values.



2. Execute the `CALL` statement from the VQL Shell:

```
CALL GOOGLE_SHEETS_TOOL(
      '<spreadsheet_id>',
      '<sheet_name>',
      '<view_name>',
      '<first_row_column_names',
      '<num_columns',
      '<access_token',
      '<refresh_token>',
      '<client_id>',
```

```
            '<client_secret>'
      );
```

3. Execute as a SELECT statement in the VQL Shell:

```
      SELECT *
      FROM
      GOOGLE_SHEETS_TOOL(
            '<spreadsheet_id>',
            '<sheet_name>',
            '<view_name>',
            '<first_row_column_names>',
            '<num_columns>',
            '<access_token>',
            '<refresh_token>',
            '<client_id>',
            '<client_secret>'
      );
```

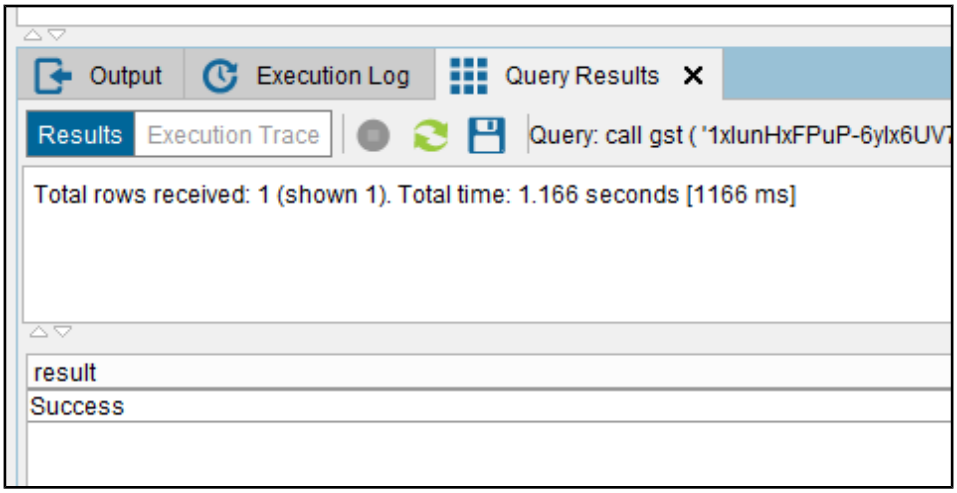
4. Execute as a SELECT statement in the VQL Shell:

```
      SELECT *
      FROM GOOGLE_SHEETS_TOOL()
      WHERE SPREADSHEET_ID = 'spreadsheet_id'
              and SHEET_NAME = 'sheet_name'
              and VIEW_NAME = 'view_name'
                    and    FIRST_ROW_COLUMN_NAMES    =
'first_row_column_names'
              and NUM_COLUMNS = 'num_columns'
              and ACCESS_TOKEN = 'access_token'
              and REFRESH_TOKEN = 'refresh_token'
              and CLIENT_ID = 'client_id'
              and CLIENT_SECRET = 'client_secret';
```

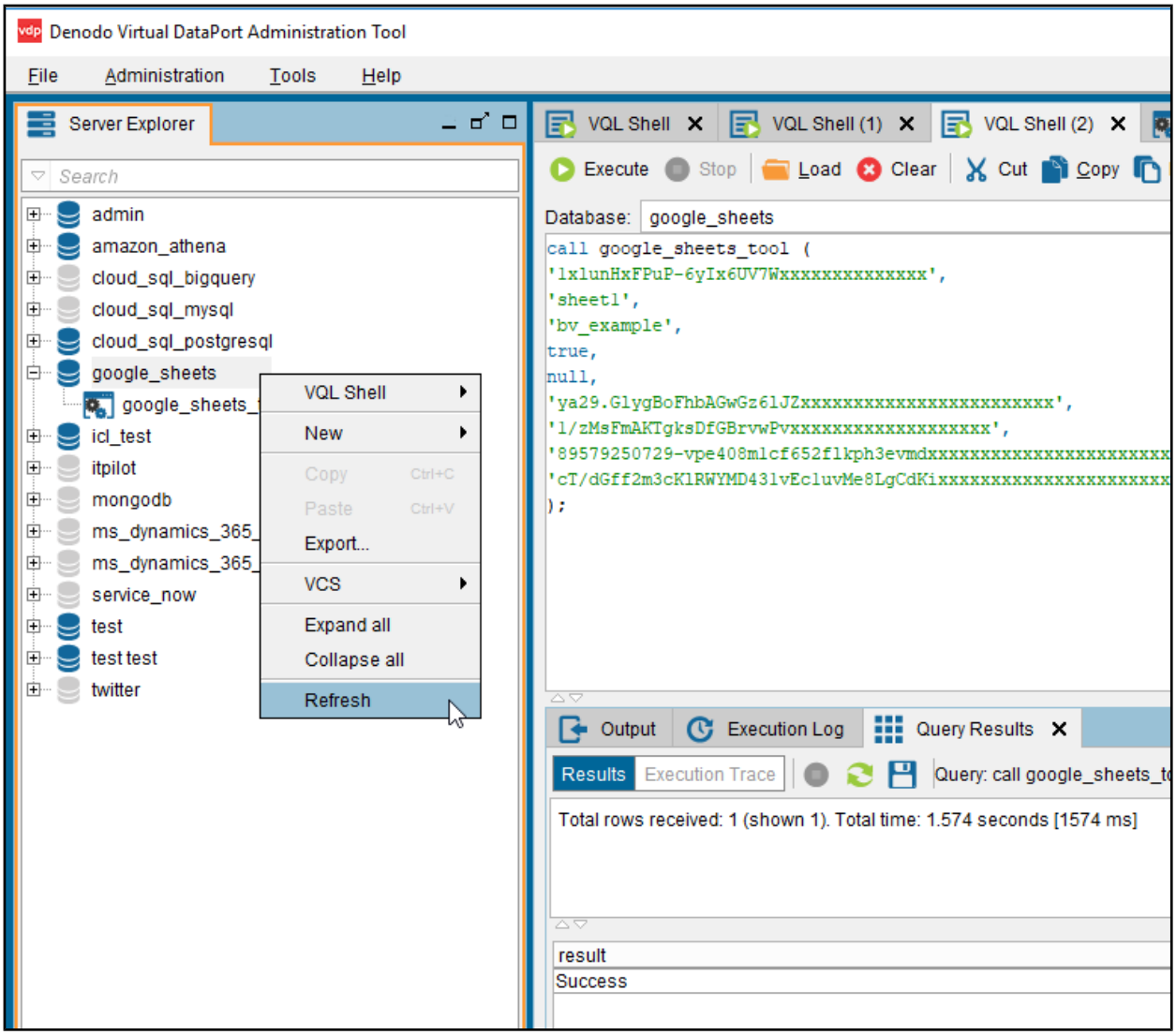
The Denodo Google Sheets Tool Stored Procedure **has an output parameter**, RESULT, that shows if the process ended successfully or not.

4.2 REFRESH THE VDP INTERFACE

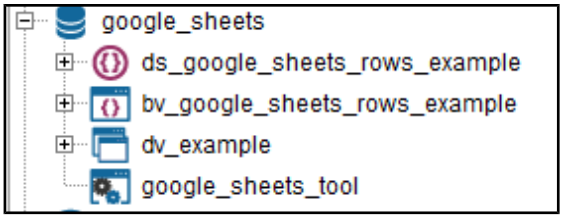
Once the execution finishes OK, you'll see a message like this:



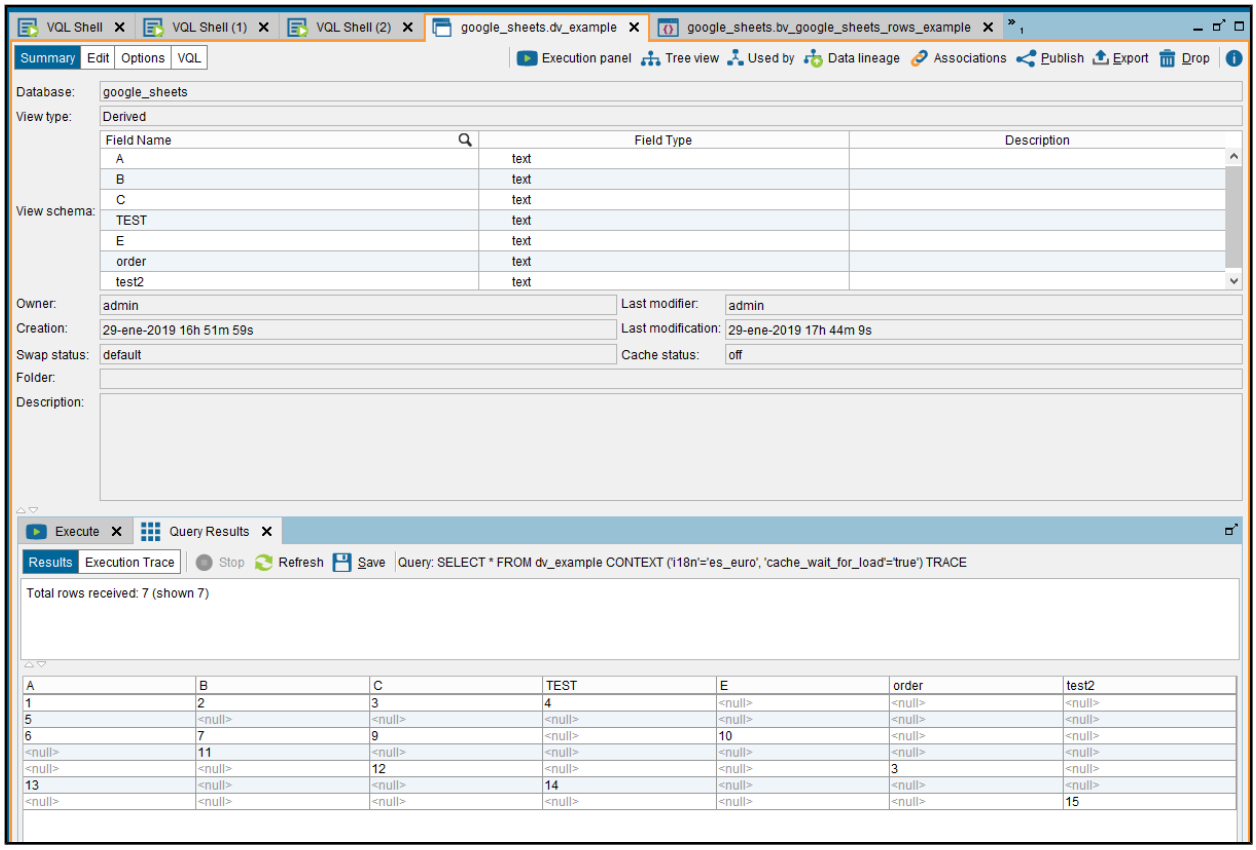
This means that the Google Sheets elements were successfully created, but you have to refresh the VDP interface in order to see them. To get this done, just right click in the database where you have executed the Stored Procedure and select `Refresh`:



Now you can finally see **three new elements**: the final derived view, which is the view that will represent the selected Google Sheet, and a data source and a base view, structures needed for the derived view to work. In the current example, `dv_example` is the mentioned final view:



If we execute it, we can see the data stored in the Google Sheet:


VQL Shell
VQL Shell (1)
VQL Shell (2)
google_sheets.dv_example
google_sheets.bv_google_sheets_rows_example
Summary
Edit
Options
VQL
Execution panel
Tree view
Used by
Data lineage
Associations
Publish
Export
Drop
Database: google_sheets
View type: Derived
View schema:
Field Name
Field Type
Description
A text
B text
C text
TEST text
E text
order text
test2 text
Owner: admin
Last modifier: admin
Creation: 29-ene-2019 16h 51m 59s
Last modification: 29-ene-2019 17h 44m 9s
Swap status: default
Cache status: off
Folder:
Description:
Execute
Query Results
Results
Execution Trace
Stop
Refresh
Save
Query: SELECT * FROM dv_example CONTEXT ('i18n'='es_euro', 'cache_wait_for_load'='true') TRACE
Total rows received: 7 (shown 7)
A B C TEST E order test2
1 2 3 4 <null> <null> <null>
5 <null> <null> <null> <null> <null> <null>
6 7 9 <null> 10 <null> <null>
<null> 11 <null> <null> <null> <null> <null>
<null> <null> 12 <null> <null> 3 <null>
13 <null> <null> 14 <null> <null> <null>
<null> <null> <null> <null> <null> <null> 15

5 LIMITATIONS

5.1 THE VIEWS WILL ONLY HAVE STRING TYPES

The Google Sheets API returns JSON responses where all parameters are defined as Strings, even if they are numbers or other data types. Because of this, the parameters of the views in Denodo will also be text types.

5.2 USING FIRST ROW OF GOOGLE SHEET TO EXTRACT COLUMN NAMES

This feature requires reading the whole sheet as we execute a base view that uses the Google Sheets API method that retrieves the whole data in rows, so the filter is applied in memory.