WhereScape Source Enablement Pack - Google Drive

This is a guide for installing Source Enablement Packs for WhereScape RED 8.6.1.x

Prerequisites

- Python 3.8 or higher
  - Download python installer from https://www.python.org/downloads/
  - Select "Add Python 3.8 to PATH" from installation Window
- PIP Manager
  - From Command Prompt (Run As Administrator) run command:

PIP Manager Install

```
python -m pip install --upgrade pip
```

- Google Drive
  - Enable Google API and setup from https://console.cloud.google.com/home/dashboard
    NOTE: Refer to section Google Drive API Connection Steps below for more details.
  - Download authentication file credentials.json
  - Install Google API packages:

Google API Packages

```
pip install --upgrade google-api-python-client google-auth-httplib2 google-auth-oauthlib
pip install google-api-python-client
pip install --upgrade google-api-python-client oauth2client
```

NOTE:Use 64-bit powershell terminal

Source Enablement Pack Setup Scripts

The Enablement Pack Install process is entirely driven by scripts. The below table outlines these scripts, their purpose and if "Run as Administrator" is required.

#	Enablement Pack Setup Scripts	Script Purpose	Run as Admin	Intended Application
1	install_Source_Enablement_Pack.ps1	Install Python scripts and UI Config Files for browsing files from Amazon S3, Azure Data Lake Gen2,Google Drive	Yes	New and Existing installations

Powershell script above provides some help at the command line, this can be output by passing the "-help" parameter to the script.

NOTE: Note that on some systems executing Windows Powershell scripts is disabled by default, see troubleshooting for workarounds.

Source Enablement Pack Installation

- Run Windows Powershell as Administrator

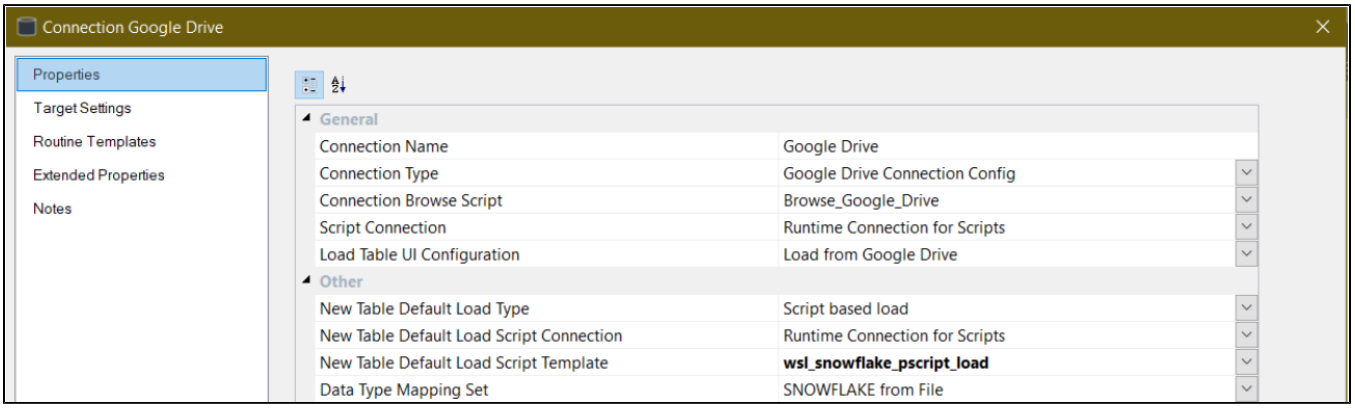
Install Source Connectivity Packs

```
<Script1 Location > Powershell -ExecutionPolicy Bypass -File .\install_Source_Enablement_Pack.ps1
```

- If prompted enter source enablement pack as **'Google'**

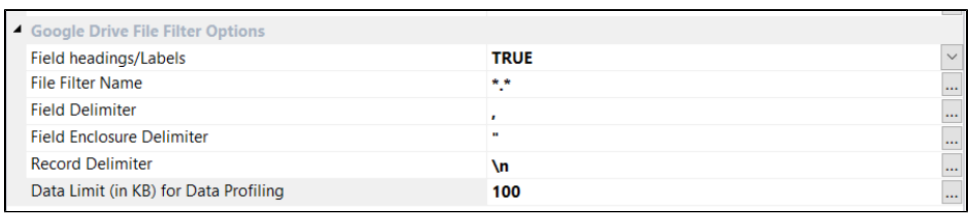
Google Drive Connection Setup

1. Login to RED
2. Check in **Host Script** Browse_Google_Drive in objects list.
3. Check **UI Configurations** in Menu, Tools UI Configurations Maintain UI Configurations
4. Create new connection in RED
5. Select properties as shown in below screenshot



- Property Section **Google Drive Settings**
    - Drive Folder : Source file location on google drive.Path till single Child folder supported, for example: 'rootFolder\\subFolder' are considered. Path for shared files are supported.The token used to read directory name in the script is *$WSL_SRCCFG_googleDriveFolderPath$*
- Property Section **Google Drive Auth File Path**
    - Authentication Files Path : File location where credentials.json file is stored. This path will also be used to create and store token. json file for authentication,The token used to read Authentication Files Path in the scripts is *$WSL_SRCCFG_googleDriveCredentialsFilePath$*

- Property Section **Google Drive File Filter Options**
    - Field Headings/Labels : Indicates whether the first line of source file contains a heading/label for each field, which is not regarded as data so it should not be loaded.
      The token used to reader field header boolean value in the script is *$WSL_SRCCFG_googleDriveFirstLineHeader$*
    - File Filter Name: Indicates source file name.Provide Google Drive filename pattern. The file list filters with file extensions,file name patterns
        - *.*
        - *.<File Extension>
        - <File Name>.<File Extension>
        - <File Name Start>*
        - The token used to read File Filter Name in the scripts is *$WSL_SRCCFG_googlDriveFileFilterName$*



    - Field Delimiter : This is a character that separates the fields within each record of the source file.The field delimiter identifies end of each field.For Example, comma (,),pipe(|).
      The token used to reader field delimiter in the script is *$WSL_SRCCFG_googleDriveFieldDelimiter$*
    - Field Enclosure Delimiter: This is a character that delimits BOTH start and end of field value i.e. encapsulates value. A double quote is common enclosure delimiter.

The token used to reader field enclosure delimiter in the script is *$WSL_SRCCFG_googleDriveFieldEnclosureDelimiter$*

- Record Delimiter : This is to identify how each line/record in source file is ended/terminated/delineated.Default is '\n' .
  The token used to read record delimiter value in the script is *$WSL_SRCCFG_googleDriveRecordDelimiter$*

- Data Limit (in KB) for Data Profiling : Kilobytes (KB's) to scan for Data Profiling.Data profiling is used to get the column names and data types from the source file.By default 100 KB will be scanned. The token used to read record delimiter value in the script is *$WSL_SRCCFG_googleDriveDataLimit$*

NOTE
• Files with extension .dat are read as a single column because google drive reads .dat files as application/octet-stream file type • Folders present in Google Drive must have unique names. Example: "REDFolder" and "REDFolder1" similar folder name should be avoided • Files which are deleted from 'Google Drive Folders' are still browsable. To avoid such files, delete them from 'Bin\Trash' as well. • Sometimes, it might take 2-5 minutes for files or directories that were recently added/uploaded to 'Google Drive' to be browse.

Google Drive API Setup Steps - Optional

These steps are for google API connection setup.If the API is already set and credentials.json is ready then these steps are not required.

1. Visit this URL: https://console.cloud.google.com/home/dashboard and Click on 'Select a project' 'New Project'
2. Enter project name and click on 'Create'
3. Go to 'APIs & Services' and then to 'Library'
4. Search for "Drive API" and click on 'Enable'
5. Click on 'Credentials' on left and then click on '+ CREATE CREDENTIALS' and select 'OAuth client ID'
6. Click on 'CONFIGURE CONSENT SCREEN' and select 'User Type' as 'External'
7. In 'App Information' add App name, User support email (current email), and Developer Contact info (current email)
8. In 'Scope Section' click on 'ADD OR REMOVE SCOPES'
9. Search for 'Drive' and select "https://www.googleapis.com/auth/drive" scope
10. Click on check-box next to selected scope and click update
11. Click on 'SAVE AND CONTINUE'
12. In "Test Users Section", click on "+ ADD USERS"
13. Add Users who can connect with these APIs. Note: Users who are listed here can only log in and browse files. Maximum 100 users can be added. This section can be edited anytime.
14. Click on 'Save and Continue'
15. Go back to 'Credentials' and click on '+ CREATE CREDENTIALS' and select 'OAuth client ID'
16. Select 'Application type' as "Desktop app" and enter name, example: 'Gdrive_RED'
17. In 'Credentials Section' under "OAuth 2.0 Client IDs" verify the auth app which was just created and press download button which is at end of that row.
18. Save this file with name "credentials" **ONLY**
19. Save the '**credentials.json**' file in a folder.
20. Login to **RED** and create Google drive connection as mentioned in the above section Google Drive Connection Setup
21. Right click and browse the connection.
22. It will be directed to default browser, select 'Google Id' which you want to connect with drive.
23. On "Google hasn't verified this app" screen, click 'Continue'
24. Click "Allow" to grant necessary permission.
25. A success message will be displayed on both, browser and RED. Close browser windows and browse connection again in RED.

Troubleshooting and Tips

Run As Administrator

Press the Windows Key on your keyboard and start typing cmd.exe, when the cmd.exe icon shows up in the search list right click it to bring up the context menu, select "Run As Administrator"

Now you have an admin prompt navigate to to the folder where you have unpacked your WhereScape Source Enablement Pack to using the 'cd' command:

```
C:\Windows\system32> cd <full path to the unpacked folder>
```

Run Powershell (.ps1) scripts from the administrator prompt by typing the Powershell run script command, for example:

```
C:\temp\EnablementPack>Powershell -ExecutionPolicy Bypass -File .\install_Source_Enablement_Pack.ps1
```

NOTE: In the event you can not bypass the Powershell execution policy due to group policies you can instead try "-ExecutionPolicy RemoteSigned" which should allow unsigned local scripts.

Windows Powershell Script Execution

On some systems Windows Powershell script execution is disabled by default. There are a number of workarounds for this which can be found by searching the term "Powershell Execution Policy".

Here is the most common workaround which WhereScape suggests, which does not permanently change the execution rights:

Start a Windows CMD prompt as Administrator, change directory to your script directory and run the WhereScape Powershell scripts with this command:

- cmd:>Powershell -ExecutionPolicy Bypass -File .\<script_file_name.ps1>

Restarting failed scripts

Some of the setup scripts will track each step and output the step number when there is a failure. To restart from the failed step (or to skip the step) provide the parameter "-startAtStep <step number>" to the script.

Example:

Powershell -ExecutionPolicy Bypass -File .\<script_file_name.ps1> -startAtStep 123

Tip: to avoid having to provide all the parameters again you can copy the full command line with parameters from the first "INFO" message from the beginning of the console output.

If a valid RED installation can not be found

If you have Red 8.6.1.x or higher installed but the script (install_Source_Enablement_Pack.ps1) fails to find it on you system then you are most likely running PowerShell (x86) version which does not show installed 64 bit apps by default. Please open a 64 bit version of PowerShell instead and re-run the script.