



Denodo Templates for SharePoint - Quick Use Guide

Revision 20221219

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 © 2024
Denodo Technologies Proprietary and Confidential

CONTENTS

1 OVERVIEW.....	4
2 THE SHAREPOINT ONLINE TEMPLATES.....	5
3 GETTING OAUTH 2.0 CREDENTIALS.....	6
3.1.1 CREATE OAUTH 2.0 CREDENTIALS FOR ODATA ENTITIES TEMPLATES.....	6
3.1.2 CREATE OAUTH 2.0 CREDENTIALS FOR REST API TEMPLATES.....	6
4 SHAREPOINT REST API TEMPLATES.....	7
4.1 IMPORTING ARTIFACTS.....	7
4.2 REFRESH.....	9
4.3 QUERY EXAMPLE.....	12
5 SHAREPOINT ODATA TEMPLATES.....	15
5.1 IMPORTING ARTIFACTS.....	15
5.2 REFRESH.....	17
5.3 QUERY EXAMPLE.....	18
6 REFERENCES.....	20

1 OVERVIEW

SharePoint Online is a cloud-based service that helps organizations share and manage content, knowledge and applications to:

- Empower teamwork
- Quickly find information
- Seamlessly collaborate across the organization

You can configure Denodo to retrieve OData entities from SharePoint Online by creating a data source using the Denodo OData2 Custom Wrapper and creating a base view for each OData entity. In addition, you can configure Denodo to retrieve data from your SharePoint Online file system by creating JSON data sources and base views in Denodo using the SharePoint API REST.

As SharePoint Online manages a very large amount of data, it may be hard work to add all the required data sources and base views in Denodo. In order to make this process easier and faster, we distribute templates: a set of predefined VQL and properties files. You just have to configure a few parameters in the properties file and import these files into your Denodo server to get access to Sharepoint.

In the case that you require information not available through the templates, have a look at the summary table of all possible accesses to Sharepoint through Denodo of the [How to integrate Denodo with Sharepoint Online](#) document.

2 THE SHAREPOINT ONLINE TEMPLATES

The SharePoint Online templates are divided in two different script files:

- sharepoint_odata_entities_templates_denodo.vql
- sharepoint_api_rest_templates_denodo.vql

The first one contains the data sources and base views for the following SharePoint OData entities:

- Appdata
- Appfiles
- Attachments
- Books
- ComposedLooks
- ConvertedForms
- Documents
- Events
- EventsCategory
- FormTemplates
- ListTemplateGallery
- MaintenanceLogLibrary
- MasterPageGallery
- MasterPageGalleryCompatibleSearchDataTypes
- MasterPageGalleryCompatibleUIVersionS
- MasterPageGalleryStandalone
- MasterPageGalleryTargetControlType
- MasterPageGalleryTargetControlTypeSearch
- MasterPageGalleryTemplateLevel
- SearchConfigList
- SearchConfigListScope
- SharePointHomeOrgLinks
- SitePagesSitePageFlags
- SitePages
- SolutionGallery
- StyleLibrary
- TaxonomyHiddenList
- ThemeGallery
- UserInformationList
- WebPartGallery
- WebPartGalleryGroup
- WebPartGalleryRecommendationSettings

The second contains base views created with the JSON datasources that allow you to navigate through your SharePoint Online file system.

Please note: importing these scripts will create a database called sharepoint in your Virtual DataPort installation, and will replace any existing database previously existing with that exact name.

3 GETTING OAUTH 2.0 CREDENTIALS

In order to be able to configure the templates to connect Denodo to your Sharepoint app you must see the **Configuring the application in Azure** section from [How to integrate Denodo with Sharepoint Online](#) document.

3.1.1 Create OAuth 2.0 credentials for OData entities templates

In order to create OAuth 2.0 credentials for OData entities templates you must see the **Connecting to Sharepoint using OData /Obtain the OAuth tokens** section from [How to integrate Denodo with Sharepoint Online](#) document.

3.1.2 Create OAuth 2.0 credentials for REST API templates

You can use the credentials created in the previous step to get the access token and the refresh token through the OAuth 2.0 wizard of the VDP as explained in the **Get the OAuth tokens using the OAuth credentials wizard** section of the [How to integrate Denodo with Sharepoint Online](#) document.

4 SHAREPOINT REST API TEMPLATES

SharePoint includes a REST service that is comparable to the existing SharePoint client object models. Now, you can interact directly with SharePoint objects by using any technology that supports standard REST capabilities.

The VQL configures access to the SharePoint file system using the SharePoint REST API.

4.1 IMPORTING ARTIFACTS

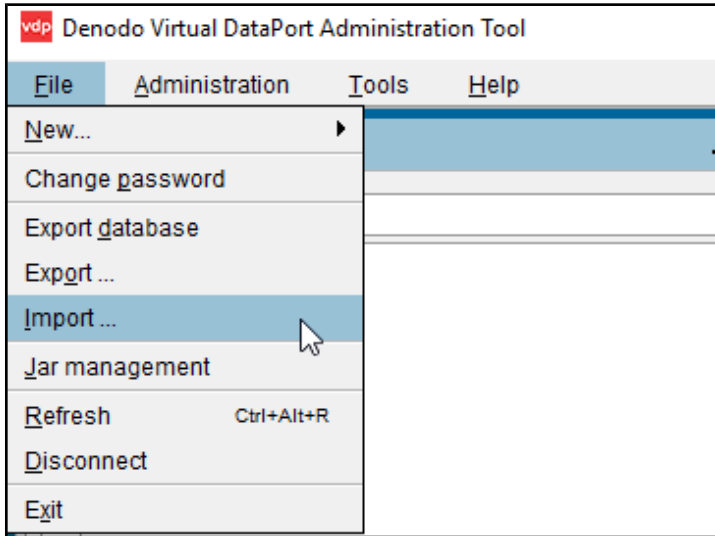
Before importing the VQL in Denodo, you have to set the following parameters defined in the `sharepoint_api_rest_templates_denodo.properties` file:

```
sharepoint.tenant.name=  
sharepoint.realm=  
sharepoint.client.id=  
sharepoint.client.secret=  
access.token=  
refresh.token=  
redirect.uri=  
sharepoint.site.name=  
view.prefix=
```

- `sharepoint.tenant.name`: Your SharePoint Online service name.
- `sharepoint.realm`: The Directory tenant ID.
- `sharepoint.client.id`: The client id generated in the Adobe Developer Console in the previous step.
- `sharepoint.client.secret`: The client secret generated in the Adobe Developer Console in the previous step.
- `redirect.uri`: The redirect uri you have set when you have registered the app in Azure.
- `access.token`: Provided by VDP Wizard for OAuth 2.0.
- `refresh.token`: Provided by VDP Wizard for OAuth 2.0.
- `sharepoint.site.name`: The site name of the specific SharePoint Online site you want to retrieve lists and folders content.
- `view.prefix`: You can configure the name prefix of every view created in the VDP in order to be able to relate VDP views to SharePoint Online sites. The max length of this parameter is 10 characters.

You can import as many `sharepoint_api_rest_templates_denodo.vql` scripts as you want for retrieving data from all your SharePoint Online sites. If a long time elapses between importing one script and another, you will have to regenerate the tokens again.

Note: We have used “site2” as the value of the `view.prefix` and “SampleSite” as the value of the `sharepoint.site.name` configuration parameters in the examples shown below.



Once you have set the properties file, you only have to import both the properties and the `sharepoint_api_rest_templates_denodo.vql` file using the Import option of the VDP Administration Tool:

Import
✕

VQL file Browse

Use properties file

Properties File Browse

Save output

Output File Browse

Report only commands that were not executed successfully

Use custom password for sensitive data decryption

Password:

Import in current server

Default database: ▼

Import in multiple servers. If selected, you can specify a list of servers using the table below so the VQL will be imported into all the selected servers.

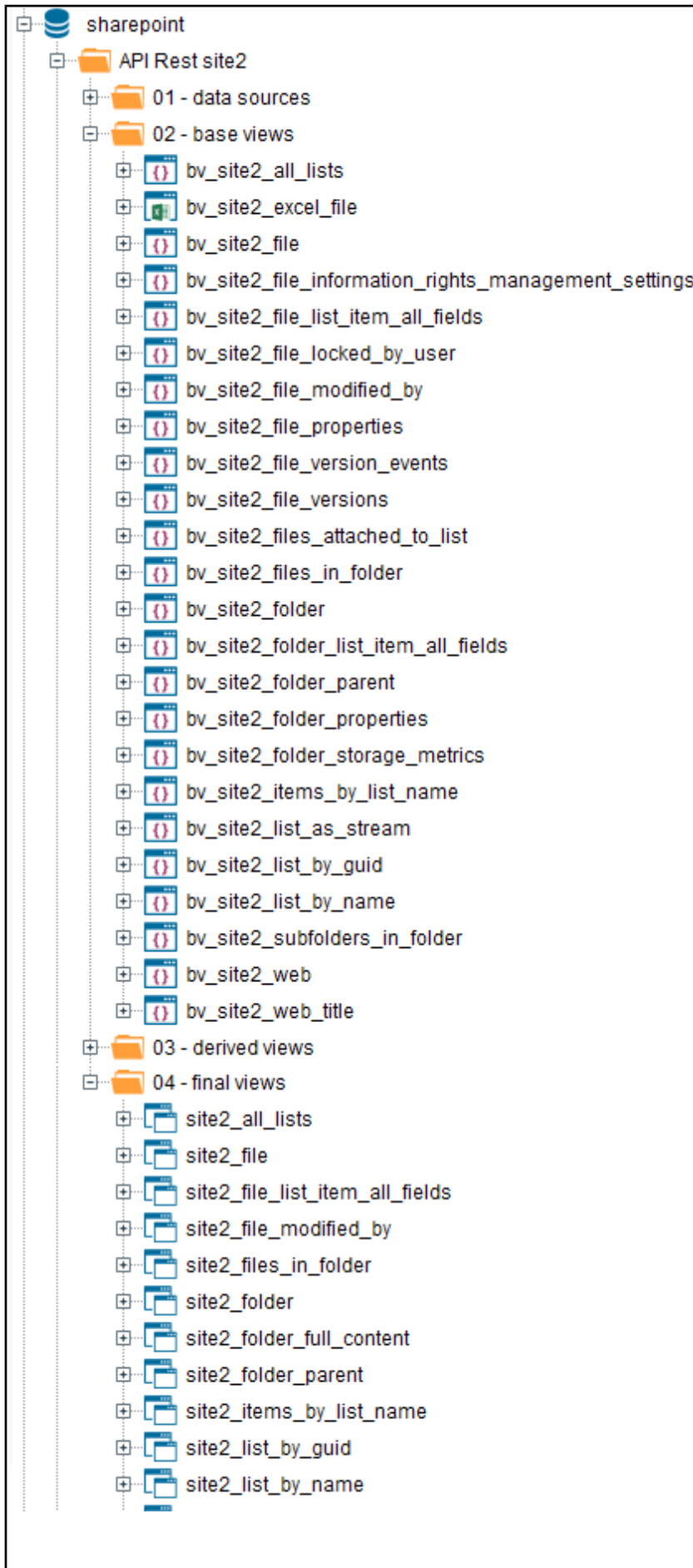
	Host	Port	Database	User	Password	Query Timeout	Chunk Timeout	Chunk Size
(empty)								

Add Remove selected Add from file

Ok Cancel

4.2 REFRESH

After the import operation finishes, refresh the Server Explorer by selecting File > Refresh and you will be able to see the data sources and views created to access the specified SharePoint Online site lists and folders content.



- `${view.prefix}_all_lists`: Retrieving all the lists of the SharePoint site specified in the `sharepoint.site.name` property.
- `${view.prefix}_file`: Retrieving a file.
 - Required parameters:
 - `file_path`: The path to the file.
- `${view.prefix}_file_list_item_all_fields`: Retrieving the value of the `ListItemAllFields` property of the file.
 - Required parameters:
 - `file_path`: path to the file.
- `${view.prefix}_file_modified_by`: Retrieving the value of the `ModifiedBy` property of the file.
 - Required parameters:
 - `file_path`: path to the file.
- `bv_${view.prefix}_file_version_events`: Retrieving the value of the `VersionEvents` property of the file.
 - Required parameters:
 - `file_path`: path to the file.
- `bv_${view.prefix}_file_versions`: Retrieving the value of the `Version` property of the file.
 - Required parameters:
 - `file_path`: path to the file.
- `bv_${view.prefix}_files_attached_to_list`: Retrieving all the files that are attached to a list item.
 - Required parameters:
 - `list_name`: the name of the list.
 - `item_id`: the id of the item.
- `${view.prefix}_files_in_folder`: Retrieving the files in a folder.
 - Required parameters:
 - `folder_path`: the path of the folder..
- `${view.prefix}_folder`: Retrieving info about the folder.
 - Required parameters:
 - `folder_path`: the path to the folder.
- `${view.prefix}_folder_parent`: Retrieving the value of the `ParentFolder` property of the folder.
 - Required parameters:
 - `folder_path`: the path to the folder.
 - `property_name`: the name of the property.
- `bv_${view.prefix}_items_by_list_name`: Retrieving all items in a list.
 - Required parameters:
 - `list_name`: the name of the list.
- `bv_${view.prefix}_list_as_stream`: Retrieving items as a stream.
 - Required parameters:
 - `list_name`: the name of the list

- `${view.prefix}_list_by_guid`: Retrieving the details of a list by GUID.
 - Required parameters:
 - `guid`: The list guid.
- `${view.prefix}_list_by_name`: Retrieve a list by name.
 - Required parameters:
 - `list_name`: the name of the list
- `${view.prefix}_subfolders_in_folder`: Retrieving the folders in a folder.
 - Required parameters:
 - `folder_path`: the path to the folder.
- `${view.prefix}_web`: Retrieving the details of the site specified in the `sharepoint.site.name` property.
- `bv_${view.prefix}_web_title`: Retrieving the title of the site specified in the `sharepoint.site.name` property
- `bv_${view.prefix}_excel_file`: Retrieving an excel file.
 - Required parameters:
 - `file_path`: the path of the file to retrieve.
- `${view.prefix}_folder_full_content`: Retrieving both the files and the folders of the folder.
 - Required parameters:
 - `folder_path`: the path of the folder.

Please note: All paths are relative to `/sites/${sharepoint.site.name}/`.

4.3 QUERY EXAMPLE

Now, you can execute queries on the SharePoint Online views that have been created:

Query 1: Retrieving all the content of the folder 'MyLibrary':



Execute Query Results

Query: SELECT folder_path AS folder_path, folders_name AS folders_name, folders_itemcount AS folders_itemcount, f

Total rows received: 4 (shown 4)

folder_path	folders_na...	folders_ite...	files_name	files_title	files_chec...	files_chec...	files_cust...	files_etag	files_exist...	files_irme...	files_length
MyLibrary	<null>	<null>	doc1.docx		2	0	"{2AA30AA...	true	false		16878
MyLibrary	<null>	<null>	Excel1.xlsx		2	0	"{CDBDED...	true	false		14601
MyLibrary	MyFolder	1	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>
MyLibrary	Forms	0	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>

Query 2: Retrieving the files attached to the item 2 attached to 'MyList' list:

Execute Query Results

Quick Query Specify Where Expression

Current sentence:

```
SELECT * FROM by_site2_files_attached_to_list WHERE item_id = 2 and list_name = 'MyList' CONTEXT ('il8n'='us_pst', 'cache_wait_for_load'='true')
```

Do not use cache
 Invalidate existing results
 Display rows: 150
 Open results in new tab
 Store results in cache
 Replace rows with the same PK value
 Do not use swap
 Retrieve all rows

Conditions

item_id	=	2
list_name	=	'MyList'

Execute Query Results

Query: SELECT list_name AS list_name, item_id AS item_id, __metadata AS __metadata, filename AS filename, filenameaspath AS filer

Total rows received: 1 (shown 1)

list_name	item_id	__metadata	filename	filenameaspath	serverrelativepath	serverrelativeurl
MyList	2	[Register]...	list_data_as_stream_b...	[Register]...	[Register]...	/sites/SampleSite/Lists/...

Query 3: Retrieving an excel file:

Execute Query Results

Quick Query Specify Where Expression

Execute Query plan

Current sentence:

```
SELECT * FROM bv_site2_excel_file WHERE file_path = 'MyLibrary/Excel1.xlsx' CONTEXT ('il8n'='us_pst', 'cache_wait_for_load'='true')
```

Do not use cache
 Invalidate existing results
 Display rows: 150
 Open results in new tab
 Store results in cache
 Replace rows with the same PK value
 Do not use swap
 Retrieve all rows

Conditions +

file_path = MyLibrary/Excel1.xlsx

Execute Query Results

Results Execution Trace

Query: SELECT h1 AS h1, h2 AS h2, h3 AS h3, file_path AS file_path FROM bv_site2_excel_file AS bv_site2_excel_file W

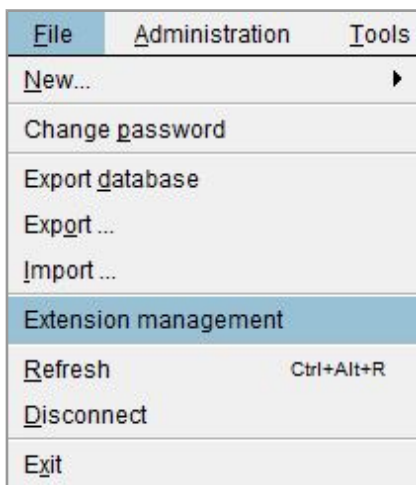
Total rows received: 5 (shown 5)

h1	h2	h3	file_path
11	12	13	MyLibrary/Excel1.xlsx
21	22	23	MyLibrary/Excel1.xlsx
31	32	33	MyLibrary/Excel1.xlsx
41	42	43	MyLibrary/Excel1.xlsx
<null>	<null>	<null>	MyLibrary/Excel1.xlsx

5 SHAREPOINT ODATA TEMPLATES

5.1 IMPORTING ARTIFACTS

Before importing the VQL into Denodo, you have to download the **Denodo OData 2 Custom Wrapper** and add the `denodo-odata2-wrapper-8.0-
{version}-jar-with-dependencies.jar` using the Extension management option of the VDP Administration Tool:



Before importing the VQL in Denodo, you have to set the following parameters defined in the `sharepoint_odata_entities_templates_denodo.properties` file:

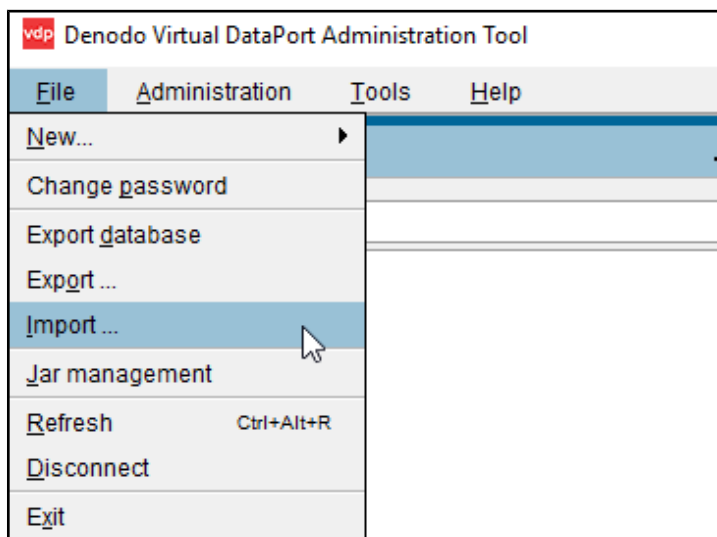
```
sharepoint.tenant.name=
sharepoint.realm=
sharepoint.audience.principal.id=
sharepoint.client.id=
sharepoint.client.secret=
access.token=
refresh.token=
sharepoint.site.name=
view.prefix=
```

- `sharepoint.tenant.name`: Your SharePoint Online service name.
- `sharepoint.realm`: The realm obtained in the **Get the Realm of the site** section from [How to integrate Denodo with Sharepoint Online](#) document.

- `sharepoint.audience.principal.id`: The value of the client id obtained in the **Get the Realm of the site** section from [How to integrate Denodo with Sharepoint Online](#) document.
- `sharepoint.client.id`: The client id generated in the Adobe Developer Console in the previous step.
- `sharepoint.client.secret`: The client secret generated in the Adobe Developer Console in the previous step.
- `access.token`: provided by VDP Wizard for OAuth 2.0.
- `refresh.token`: provided by VDP Wizard for OAuth 2.0.
- `sharepoint.site.name`: The site name of the specific SharePoint Online site you want to retrieve lists and folders content.
- `view.prefix`: You can configure the name prefix of every view created in the VDP in order to be able to relate VDP views to SharePoint Online sites. The max length of this parameter is 10 characters.

You can import as many `sharepoint_odata_entities_templates_denodo.vql` scripts as you want for retrieving data from all your SharePoint Online sites. If a long time elapses between importing one script and another, you will have to regenerate the tokens again.

Note: We have used “site2” as the value of the `view.prefix` and “SampleSite” as the value of the `sharepoint.site.name` configuration parameters in the examples shown below.



Once you have set the properties file, you only have to import both the properties and the `sharepoint_odata_entities_templates_denodo.vql` file using the Import option of the VDP Administration Tool:

Import
✕

VQL file Browse

Use properties file

Properties File Browse

Save output

Output File Browse

Report only commands that were not executed successfully

Use custom password for sensitive data decryption

Password:

Import in current server

Default database: ▼

Import in multiple servers. If selected, you can specify a list of servers using the table below so the VQL will be imported into all the selected servers.

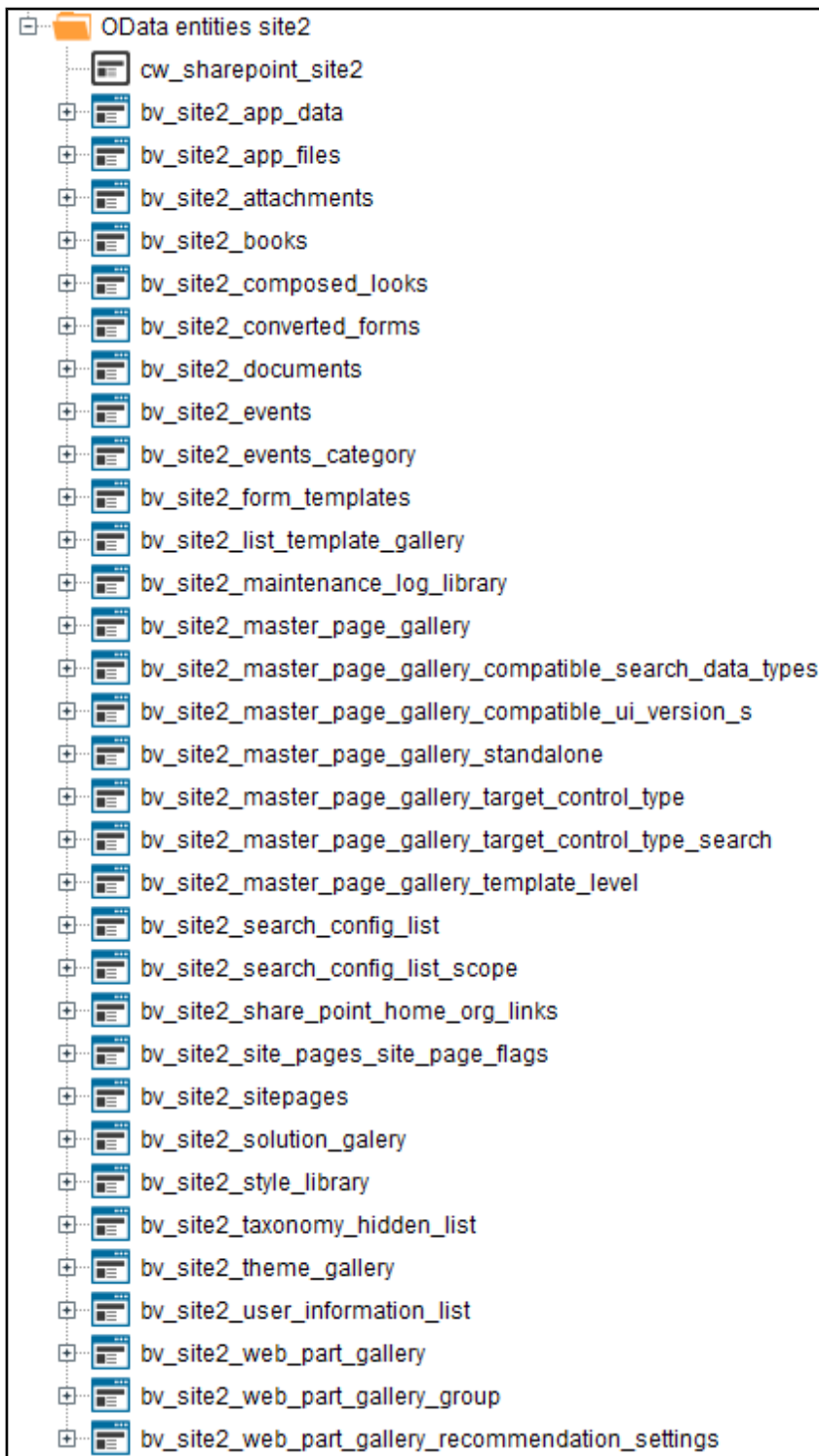
	Host	Port	Database	User	Password	Query Timeout	Chunk Timeout	Chunk Size
(empty)								

Add Remove selected Add from file

Ok Cancel

5.2 REFRESH

After the import operation finishes, refresh the Server Explorer by selecting File > Refresh and you will be able to see the data sources and views created to access the SharePoint OData entities.



5.3 QUERY EXAMPLE

Now, you can execute queries on the SharePoint Online views that have been created.

Example, executing the `bv_${view.prefix}_sitepages` view:

Execute Query Results

Quick Query Specify Where Expression

Execute Query plan

Current sentence:

```
SELECT * FROM by_site2_sitepages CONTEXT ('118n='us_pst', 'cache_wait_for_load'='true')
```

Do not use cache
 Invalidate existing results
 Display rows: 150
 Open results in new tab
 Store results in cache
 Replace rows with the same PK value
 Do not use swap
 Retrieve all rows
 Conditions +

Execute Query Results

Results Execution Trace

Query: SELECT * FROM by_site2_sitepages LIMIT 150 CONTEXT ('118n='us_pst', 'cache_wait_for_load'='true') TRACE

Total rows received: 8 (shown 8)

contenttyp...	name	complianc...	wikicontent	title	bannerima...	description	promoteds...	firstpublis...	topicheader	calltoaction
0x0101009...	Home.aspx	<null>	<null>	Home	<null>	<null>	<null>	<null>	<null>	<null>
0x0101009...	ProjectHo...	<null>	<null>	Home	https://den...	Customizat...	0.0	<null>	<null>	<null>
0x0120005...	Plantillas	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>
0x0101009...	Status-rep...	<null>	<null>	Status report template	https://den...	<Project> s...	0.0	<null>	THIS PAGE...	<null>
0x0101009...	Sample-Pa...	<null>	<null>	Sample Page	https://den...	Hello! Esto ...	0.0	<null>	TEXTO SIT...	<null>
0x0101009...	Sample-Sit...	<null>	<null>	Sample Site Home	https://den...	Hello! Esto ...	0.0	<null>	TEXTO SIT...	<null>
0x0101009...	This-is-a-te...	<null>	<null>	This is a test	https://den...	Hello! Esto ...	0.0	<null>	TEXTO SIT...	<null>
0x0101009...	Sample-Co...	<null>	<null>	Sample Content Report Page	https://den...	<Project> s...	0.0	<null>	THIS PAGE...	<null>

6 REFERENCES

[How to integrate Denodo with Sharepoint Online](#)