Denodo OData2 Custom Wrapper - User Manual

Revision 20181213

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 INTRODUCTION

OData is a protocol to access data created by Microsoft. It provides CRUD operations and is similar to JDBC or ODBC but not limited to databases.

OData uses protocols ATOM and JSON and the requests use a REST model. For this reason, OData is an implementation of the RESTful API that describes the data and their model.

2 ODATA SERVICES

There are several ways to access to an OData service from Virtual DataPort:

2.1 DENODO ODATA2 CUSTOM WRAPPER

The Denodo OData2 Custom Wrapper allows you to access OData services even if they require authentication (HTTP BASIC or NTLM supported) or behind a proxy server (which might also be authenticated).

2.1.1 Import the Custom Wrapper

To import the custom wrapper, follow these steps:

1. In the VDP Administration Tool, go to:
   - `Until Denodo 6.0: File → Jar management`
   - `From Denodo 7.0: File → Extension management`
2. Click on "Create" button and select the "`denodo-odata2-wrapper-{denodo-version}-{version}-jar-with-dependencies.jar`" file downloaded from Denodo SupportSite.

2.1.2 Create the OData data source

To create a new OData custom data source:

1. In the VDP Administration Tool, go to: `File → New… → Data source → Custom`
2. In the "Create a New Custom Data Source" window, do the following:
   - Set a name for the new OData 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.odata.wrapper.ODataWrapper`
3. Click on "Ok" button.

2.1.3 Create the base view

To create a new base view using the OData data source:

1. Double-click on the OData data source and then click on “Create base view”.

2. Set the parameters as follows:

    - **Service Endpoint** (*mandatory*): is the URL to the OData service. Must be something like: `http://services.odata.org/OData/OData.svc/`

    - **Entity Collection** (*mandatory*): must be one of the collections defined into the OData service. *Note: If the OData service defines textual names (titles) for its entity collections (as seen at the Service Document), different from their “href” values, it’s the “href” value what should appear here (the fragment to be used in URLs).*

    - **Service Format** (*mandatory*): is the format used by the OData custom wrapper to access to the OData service. Must be one of these:
        i. JSON
        ii. XML-Atom

    - **Service Version**: if specified, the custom wrapper try to force the compatibility of the OData service with one of these versions:
        i. V1
        ii. V2

    - **Expand Related Entities**: if checked, the references to other entities appear directly in the main entity as arrays or registers.

    - **Use NTLM Authentication:** if checked, the fields “User” and “Password” will use NTLM Authentication instead of HTTP Basic Authentication. A NTLM domain could be used using the field “NTLM Domain”.

    - **Enable Pagination**: if checked, two parameters are added to the view to permit the pagination of the results:
        i. fetch_size
        ii. Offset_size

    - **Pass-through session credentials:** if checked, the value of the login and password fields are used for introspection. During execution, the credentials of the user authenticated in VDP are used.

    - **User**: OData service user for HTTP Basic Authentication or NTLM Authentication, this is an optional parameter.

    - **Password**: OData service password for HTTP Basic Authentication or NTLM Authentication, this is an optional parameter.

    - **Proxy Host:** host to connect to the client through a proxy, this is an optional parameter.

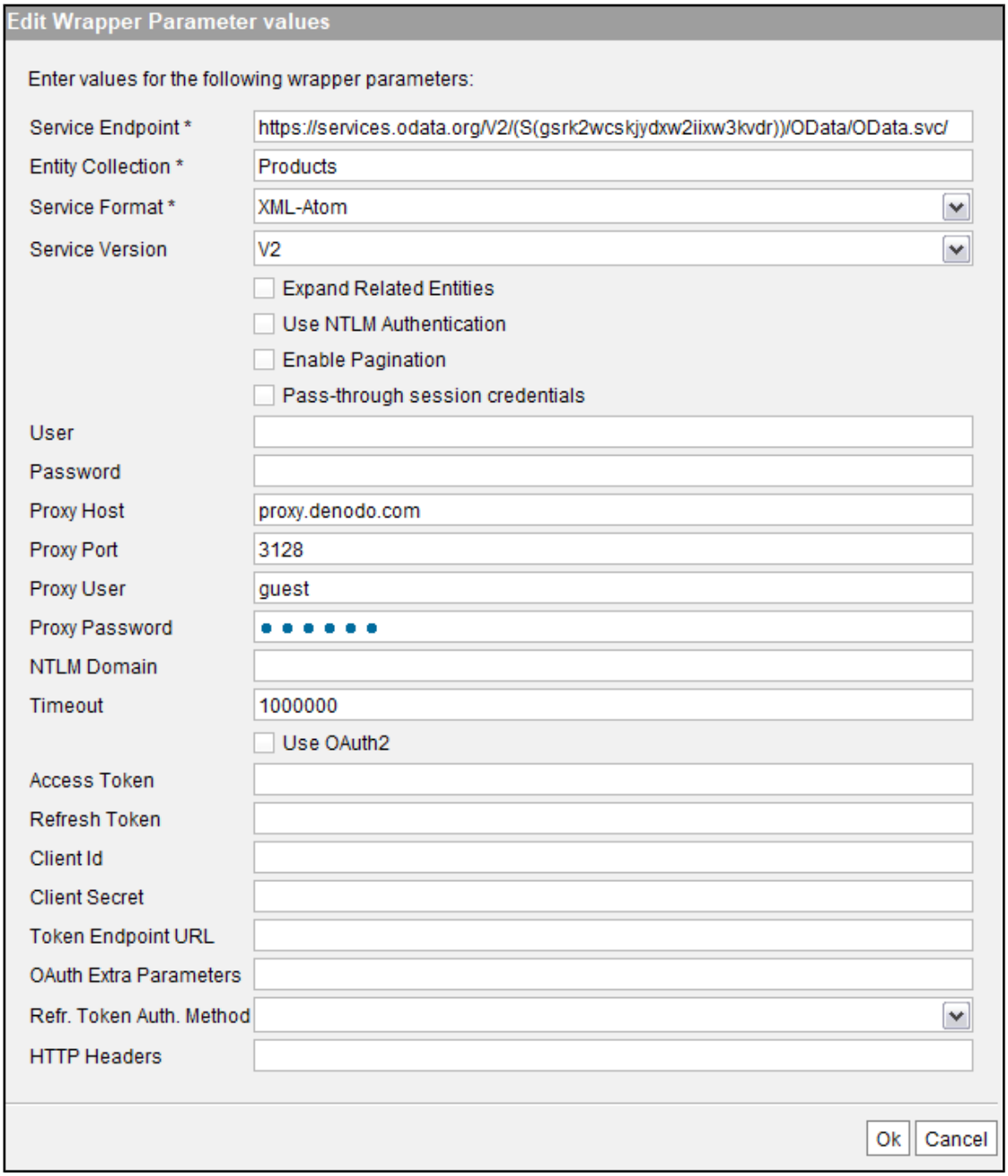
    - **Proxy Port:** port to connect to the client through a proxy, this is an optional parameter.

- **Proxy User**: user for the authentication proxy, this is an optional parameter.
- **Proxy Password**: password for the authentication proxy, this is an optional parameter.
- **NTLM Domain:** OData service domain for NTLM Authentication, this is an optional parameter.
- **Timeout:** Maximum time (in milliseconds) the custom wrapper will wait for a query to finish. If it is empty, it will wait indefinitely until the sentence ends.
- **Use OAuth2:** if checked, the protocol OAuth 2.0 will be used for authorization. With this option the fields: Access Token, Refresh Token, Client Id, Client Secret, and Token Endpoint Url are mandatory. You can use the Oauth credentials wizard to obtain the Access Token and the Refresh Token. You can read more in the VDP ADMINISTRATION GUIDE in the subsection of Oauth.
- **Access Token:** You can use the Oauth credentials wizard to get it
- **Refresh Token**: You can use the Oauth credentials wizard to get it. It is used when the Access Token has expired to get a new access token.
- **Client Id:** consumer key from the remote access application definition.
- **Client Secret:** consumer secret from the remote access application definition.
- **Token Endpoint URL:** it is URL to make OAuth refresh request when the Access Token has expired.
- **OAuth Extra Parameters:** extra parameters to be used in the the refresh token requests, this is an optional field. *Note: Multiple parameters are allowed to be added. The format must be as follows:* `field_1="value_1";field_2="value_2";...;field_n="value_n" ;`. *Being "*`field`*" the name of the parameter and "*`value`*" its value.*
- **Refr. Token Auth. Method:** controls how the credentials are sent to the service when requesting a new OAuth access token. Must be one of these:
  i. Include the client credentials in the body of the request
  ii. Send client credentials using the HTTP Basic authentication scheme
- **HTTP Headers:** custom headers to be used in the underlying HTTP client, this is an optional parameter. *Note: Multiple HTTP headers are allowed to be added. The format must be as follows:* `field_1="value_1";field_2="value_2";...;field_n="value_n" ;`. *Being "*`field`*" the name of the header and "*`value`*" its value.*

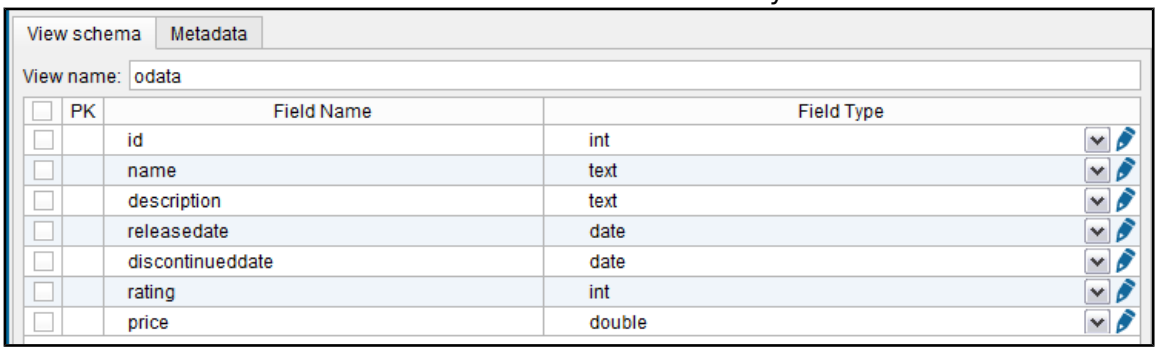
2.1.4 Example

1. Create a base view over this test service:
   a)
   - Service Endpoint = http://services.odata.org/V2/(S(gsrk2wcskjydxw2iixw3kvdr))/OData/OData.svc/
   - Entity = Products
   - Service Format = XML-Atom
   - Expand Related Entities = false
   - Enable Pagination = false

   b) This is the same option but accessing with a proxy
   - Service Endpoint = http://services.odata.org/V2/(S(gsrk2wcskjydxw2iixw3kvdr))/OData/OData.svc/
   - Entity = Products
   - Service Format = XML-Atom
   - Expand Related Entities = false
   - Use NTLM Authentication = false
   - Enable Pagination = false
   - Enable Pagination = false
   - Proxy Host=proxy.denodo.com
   - Proxy Port=3128
   - Proxy User=guest
   - Proxy Password=******
   - Timeout = 1000000


Edit Wrapper Parameter values
Enter values for the following wrapper parameters:
Service Endpoint *
https://services.odata.org/V2/(S(gsrk2wcskjydxw2iixw3kvdr))/OData/OData.svc/
Entity Collection *
Products
Service Format *
XML-Atom
Service Version
V2
Expand Related Entities
Use NTLM Authentication
Enable Pagination
Pass-through session credentials
User
Password
Proxy Host
proxy.denodo.com
Proxy Port
3128
Proxy User
guest
Proxy Password
NTLM Domain
Timeout
1000000
Use OAuth2
Access Token
Refresh Token
Client Id
Client Secret
Token Endpoint URL
OAuth Extra Parameters
Refr. Token Auth. Method
HTTP Headers
Ok
Cancel

2. The schema of the base view is shown and you can rename it.



3. After clicking on “Ok”, you can execute queries (SELECT, INSERT, UPDATE or DELETE), for example:

   - `SELECT * FROM products WHERE id = 6;`
   - ```
     INSERT INTO products
     (id,name,description,releasedate,rating,price) VALUES
     (9,'HDTV','32 inch 720p television',NOW(), 2, 600);
     ```
   - `UPDATE products SET price = 800 WHERE id = 9;`
   - `DELETE FROM products WHERE ID = 9;`

2.1.5 Known Limitations

This custom data source currently only works with OData versions 1.0 or 2.0.

OData version 3.0 is partially supported interpreting it as a lower version, but this method may not work. More information:
http://code.google.com/p/odata4j/wiki/Roadmap

You can't filter elements specified obtained through "expand" related entities. VDP must post-filter these items using ROW syntax in the query.

The insertion and the update of complex fields are not supported.

The insertion of arrays are not supported.

The wrapper doesn't allow to access to databases that contains tables without keys. In this case it throws the following exception: 'Root types must have keys'.

2.2 OTHER WAYS TO CONSUME ODATA SOURCES

2.2.1 Using ATOM/XML
     ```

Is possible to access to OData server through an URL. For example, the following URL returns all the entities of the OData server:

http://services.odata.org/OData/OData.svc/

Using this URL you can access to all entities of one type:

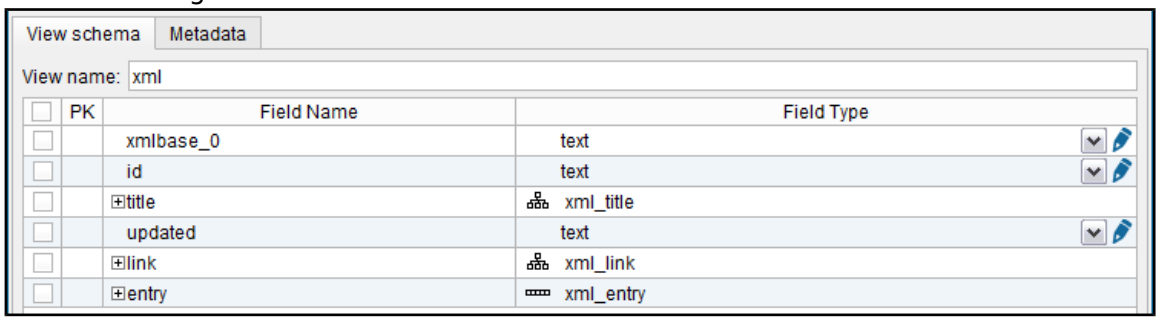
http://services.odata.org/OData/OData.svc/Products

And you can access to one entity using the identifier:

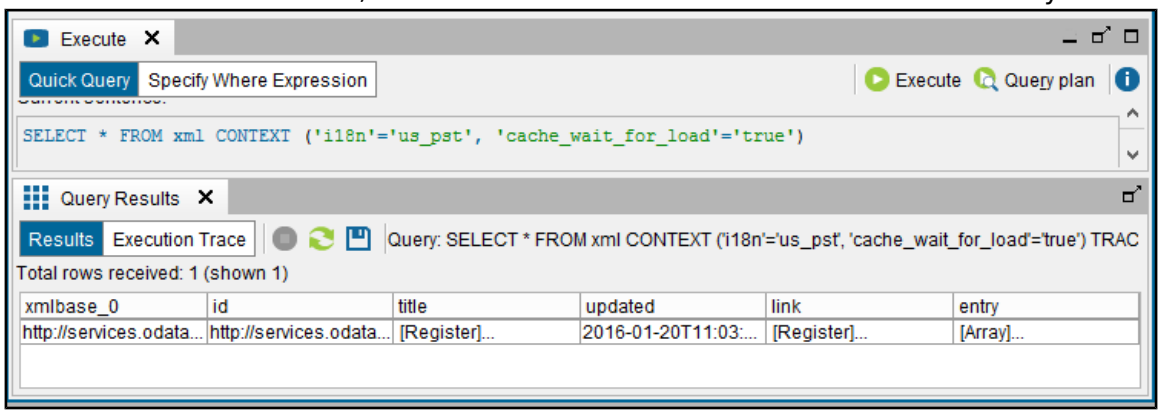
http://services.odata.org/OData/OData.svc/Products(0)

2.2.1.1 Importing ATOM/XML into VDP

1. In the VDP Administration Tool, go to: `File → New… → Data source → XML`
2. Set the parameters as follows:
   a. **Name**: the name of the new XML data source.
   b. **Data route**: "HTTP Client" configured as:
      i. HTTP method: GET
      ii. URL: URL to the entities. For example: http://services.odata.org/OData/OData.svc/Products
3. Create a new base view:
   a. We can only recover data selecting "entry" in the section "Stream output at specified level".
4. When clicking "Ok" we have a similar base view to the next one:



5. If we execute the view, the results are located into the field “entry”:



2.2.2 Using JSON

To access OData using JSON only is necessary to add the following parameter to the URL:

?$format=JSON

It’s also possible to access OData using JSON adding the following parameters to the HTTP header when doing a GET:

```
Accept: application/json
```

The following is an example of using the URL to access to OData thought JSON:

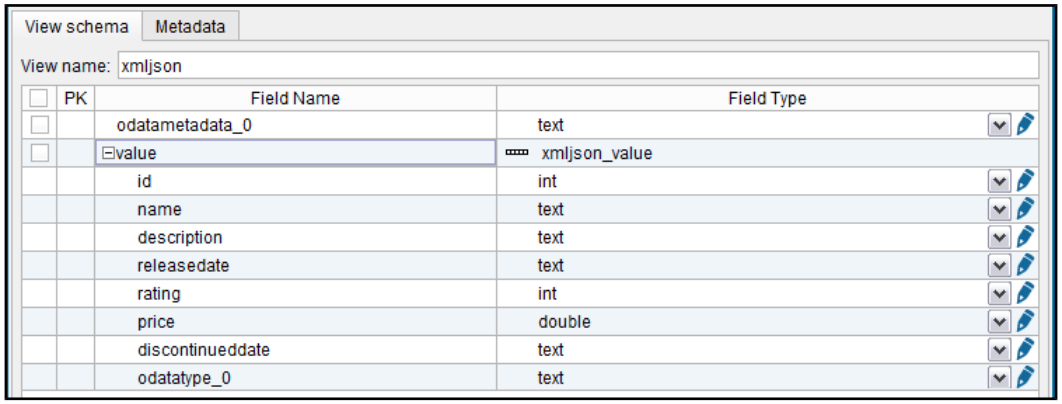
http://services.odata.org/OData/OData.svc/Products?$format=json

2.2.2.1 Importing JSON into VDP

The steps to access to OData using JSON are:

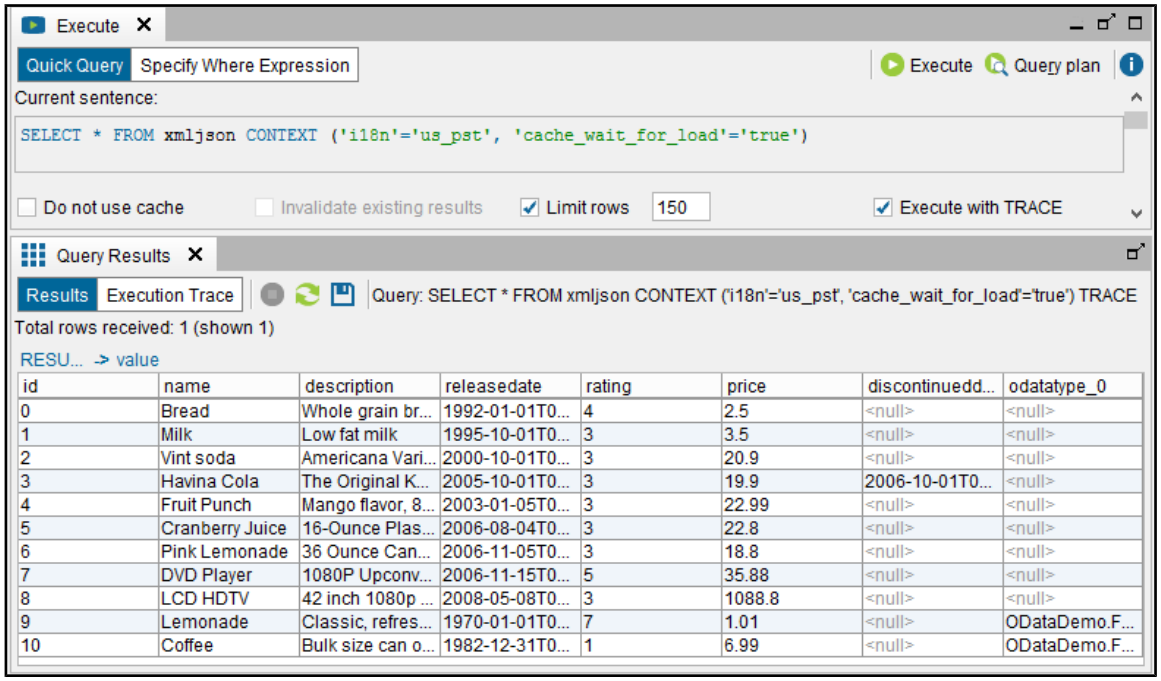
1. In the VDP Administration Tool, go to: `File → New… → Data source → JSON`
2. Set the parameters as follows:
   - **Name**: the name of the new data source.
   - **Data route**: “HTTP Client” configured as:
     - HTTP method: GET
     - URL: URL to the entities in JSON format. For example: http://services.odata.org/OData/OData.svc/Products?$format=json
3. Create a base view from the new datasource:

- We can get only the data setting the JSON root as: `/JSONFile/value`



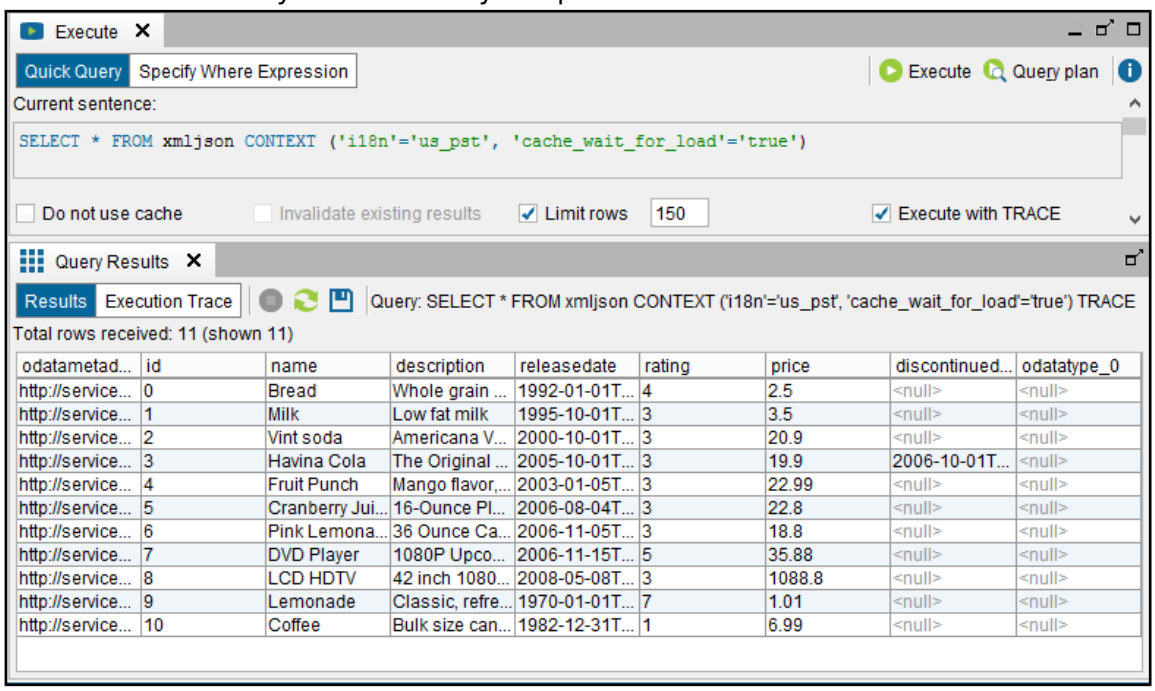
4. When clicking "Ok" we have a similar base view to the next one:

5. Data are located in the field "array_value":



id	name	description	releasedate	rating	price	discontinuedd...	odatatype_0
0	Bread	Whole grain br...	1992-01-01T0...	4	2.5	<null>	<null>
1	Milk	Low fat milk	1995-10-01T0...	3	3.5	<null>	<null>
2	Vint soda	Americana Vari...	2000-10-01T0...	3	20.9	<null>	<null>
3	Havina Cola	The Original K...	2005-10-01T0...	3	19.9	2006-10-01T0...	<null>
4	Fruit Punch	Mango flavor, 8...	2003-01-05T0...	3	22.99	<null>	<null>
5	Cranberry Juice	16-Ounce Plas...	2006-08-04T0...	3	22.8	<null>	<null>
6	Pink Lemonade	36 Ounce Can...	2006-11-05T0...	3	18.8	<null>	<null>
7	DVD Player	1080P Upconv...	2006-11-15T0...	5	35.88	<null>	<null>
8	LCD HDTV	42 inch 1080p ...	2008-05-08T0...	3	1088.8	<null>	<null>
9	Lemonade	Classic, refres...	1970-01-01T0...	7	1.01	<null>	ODataDemo.F...
10	Coffee	Bulk size can o...	1982-12-31T0...	1	6.99	<null>	ODataDemo.F...

6. Data can be directly accessed if you specified the root "`/JSONFile/value`":



odatametad...	id	name	description	releasedate	rating	price	discontinued...	odatatype_0
http://service...	0	Bread	Whole grain ...	1992-01-01T...	4	2.5	<null>	<null>
http://service...	1	Milk	Low fat milk	1995-10-01T...	3	3.5	<null>	<null>
http://service...	2	Vint soda	Americana V...	2000-10-01T...	3	20.9	<null>	<null>
http://service...	3	Havina Cola	The Original ...	2005-10-01T...	3	19.9	2006-10-01T...	<null>
http://service...	4	Fruit Punch	Mango flavor,...	2003-01-05T...	3	22.99	<null>	<null>
http://service...	5	Cranberry Jui...	16-Ounce Pl...	2006-08-04T...	3	22.8	<null>	<null>
http://service...	6	Pink Lemona...	36 Ounce Ca...	2006-11-05T...	3	18.8	<null>	<null>
http://service...	7	DVD Player	1080P Upco...	2006-11-15T...	5	35.88	<null>	<null>
http://service...	8	LCD HDTV	42 inch 1080...	2008-05-08T...	3	1088.8	<null>	<null>
http://service...	9	Lemonade	Classic, refre...	1970-01-01T...	7	1.01	<null>	<null>
http://service...	10	Coffee	Bulk size can...	1982-12-31T...	1	6.99	<null>	<null>

3 TROUBLESHOOTING

Symptom

Error message: *“Received exception with message 'com.ctc.wstx.exc.WstxEOFException: Unexpected EOF; was expecting a close tag for element <feed> at [row,col {unknown-source}]: [<row>,<col>]' ”*

Resolution

There is a problem with the implementation of the OData Service that you are trying to access to. The wrapper is attempting to project a property that do not validate the constraints of the model. You must check the values of the properties of the entry located in the row indicated in the error message taking into account the metadata.

4 REFERENCES

Odata official page

- Documentation:
  - http://www.odata.org/docs/
- Libraries:
  - http://www.odata.org/libraries/

Wikipedia article:

- http://en.wikipedia.org/wiki/Open_Data_Protocol

OData references into the Microsoft webpage:

- Create and Consume JSON-Formatted OData:
  - http://msdn.microsoft.com/es-es/magazine/jj190799.aspx
- Building Rich Internet Apps with the Open Data Protocol:
  - http://msdn.microsoft.com/es-es/magazine/ff714561.aspx
- OData Operations:
  - http://www.odata.org/documentation/odata-v2-documentation/operations/
- Examples:
  - http://msdn.microsoft.com/en-us/library/ff478141.aspx

Web pages with OData examples:

- Example read-only service in the official web site:
  - http://services.odata.org/OData/OData.svc/
  - http://services.odata.org/OData/OData.svc/$metadata
  - http://services.odata.org/OData/OData.svc/Products
  - http://services.odata.org/OData/OData.svc/Products(0)
  - http://services.odata.org/OData/OData.svc/Products?$format=json
  - http://services.odata.org/OData/OData.svc/Products(0)?$format=json

- Example read-write service in the official web site:
  - http://services.odata.org/V2/(S(gsrk2wcskjydxw2iixw3kvdr))/OData/OData.svc/
  - http://services.odata.org/V2/(S(gsrk2wcskjydxw2iixw3kvdr))/OData/OData.svc/Categories
  - http://services.odata.org/V2/(S(gsrk2wcskjydxw2iixw3kvdr))/OData/OData.svc/Products
  - http://services.odata.org/V2/(S(gsrk2wcskjydxw2iixw3kvdr))/OData/OData.svc/Suppliers

- Example OData installing the module odata-server of JayData server
  - You can install locally this module to test the Basic Authentication, the intructions in this url: https://www.npmjs.org/package/odata-server
- In the tab Live Services there are more examples: http://www.odata.org/ecosystem/

5 APPENDIX

5.1 CONNECTION TO MICROSOFT SHAREPOINT ONLINE

This section explains a brief example of a connection to Microsoft SharePoint Online.

5.1.1 Register an app in SharePoint

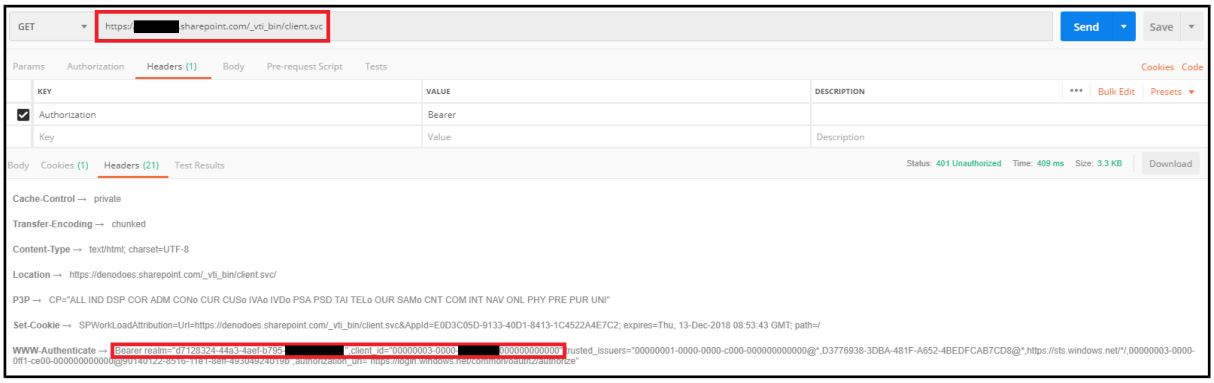
In order to register an app in SharePoint online, follow these steps:

1. Navigate to https://<your_site>.sharepoint.com/_layouts/15/appregnew.aspx
2. Press the button "*Generate*" for `Client Id` and `Client Secret`.
3. Give a name for the app,
4. Fill in the "*App Domain*" field
5. Enter `Redirect URL` value. It should be `http://localhost:9090/oauth/2.0/redirectURL.jsp`
6. Press the button *"Create"*

5.1.2 Get the Realm of the site

Realm is a constant GUID for a site. In order to retrieve it, a tool such as Postman can be used. To obtain the Realm, it is necessary to carry out the following steps:

- Make a GET request like this:
  - `https://your_site.sharepoint.com/_vti_bin/client.svc`
  - Header:
    - `Authorization: Bearer`
- Get the `Bearer` realm component from the `WWW-Authenticate` response header and save it.
- Get the `client id` component from the `WWW-Authenticate` response header and save it. This value is what later we will call Audience Principal ID.



NOTE: If you're using cookies you might not get back the `WWW-Authenticate` header with the Bearer realm. Delete the cookies in Chrome or use the Postman Interceptor to avoid this issue.

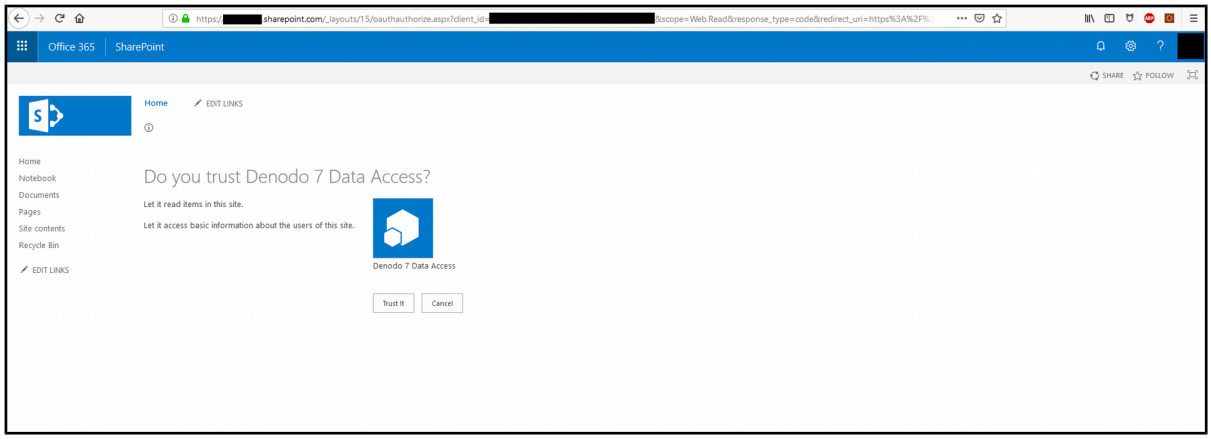
5.1.3 Get the authorization code

Construct an authorization url as follows:

```
https://<your_site>.sharepoint.com/_layouts/15/OAuthAuthorize.aspx?
client_id=<client_GUID>&scope=Web.read&response_type=code&redirect_uri=ht
tp://localhost:9090/oauth/2.0/redirectURL.jsp
```

Change the parameters of the URL to fit your connection data.

Navigate to the URL from your browser. Login to the site if you have not logged in already. This opens a consent page prompts the user to grant (or deny) the app the permissions that the app requests. In this case, the user would be granting the app read access to the current site (Web).



Once you grant the permission (by clicking trust), SharePoint Online site asks Access Control Service (ACS) to create a short-lived (approximately 5 minutes) authorization code unique to this combination of user and app. ACS sends the authorization code to the SharePoint site.

SharePoint Online site redirects the browser back to the redirect URI that was specified when the app was registered. It also includes the authorization code as a query string. The redirect URL is structured like the following:
`http://localhost:9090/oauth/2.0/redirectURL.jsp/?`**`code=<authcode>`**

5.1.4 Get the access token and refresh token

Construct the below POST request:

- URL:
  - `https://accounts.accesscontrol.windows.net/<site_realm>/tokens/OAuth/2`
- Header:
  - `Content-Type = "application/x-www-form-urlencoded"`
- Post parameters (in the body of the request):
  - `grant_type=authorization_code`
  - `client_id=<client_id>@<site_realm>`
  - `client_secret=<client_secret>`
  - `code=<auth_code>`
  - `redirect_uri=http://localhost:9090/oauth/2.0/redirectURL.jsp`
  - `resource=<audience_principal_ID>/<site_host>@<site_realm>`
  - Where:
    - `<site_realm>` is the Bearer realm obtained in the step “Get the Realm of the site”.
    - `<client_id>` is <client id when registering the app>@<site realm from the step “Get the Realm of the site>.
    - `<client_secret>` is the client_secret obtained when registering the app.
    - `<auth_code>` is the auth code obtained in step “Get the authorization code”.
    - `<resource>` is <audience principal ID>/<sharepoint domain>@<site realm>.

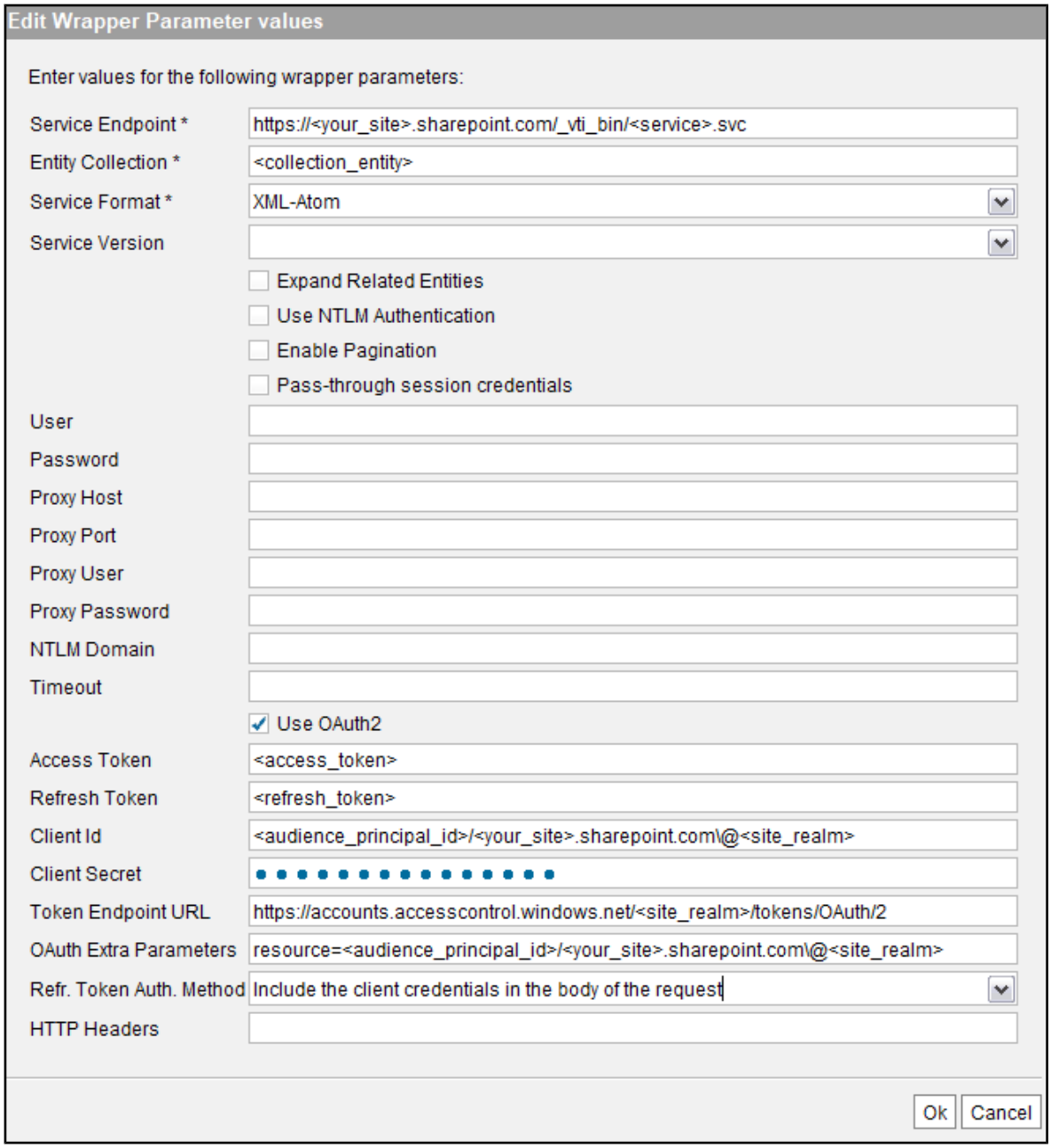
- <audience_principal_ID> is a permanent security principal ID for SharePoint. <audience_principal_ID> is obtained in step “Get the Realm of the site” (the value “client_id” in the response header WWW-Authenticate).

NOTE: all values need to be URL encoded (including the client_secret)

5.1.5 Create a base view over the service

Once the necessary data to make a connection to Sharepoint Online has been obtained, the last step is to configure the custom wrapper.

The following image shows a template of how the fields of the custom wrapper should be filled
with the previously obtained data.


Edit Wrapper Parameter values
Enter values for the following wrapper parameters:
Service Endpoint *
https://<your_site>.sharepoint.com/_vti_bin/<service>.svc
Entity Collection *
<collection_entity>
Service Format *
XML-Atom
Service Version
Expand Related Entities
Use NTLM Authentication
Enable Pagination
Pass-through session credentials
User
Password
Proxy Host
Proxy Port
Proxy User
Proxy Password
NTLM Domain
Timeout
Use OAuth2
Access Token
<access_token>
Refresh Token
<refresh_token>
Client Id
<audience_principal_id>/<your_site>.sharepoint.com\@<site_realm>
Client Secret
Token Endpoint URL
https://accounts.accesscontrol.windows.net/<site_realm>/tokens/OAuth/2
OAuth Extra Parameters
resource=<audience_principal_id>/<your_site>.sharepoint.com\@<site_realm>
Refr. Token Auth. Method
Include the client credentials in the body of the request
HTTP Headers
Ok
Cancel