

OData Access to Data Sets

Estimated time to read: 5 minutes

OData (Open Data Protocol) is an open standard protocol that enables simple and standardized access to data. **Data Platform** allows accessing the data in all data sets using this protocol.

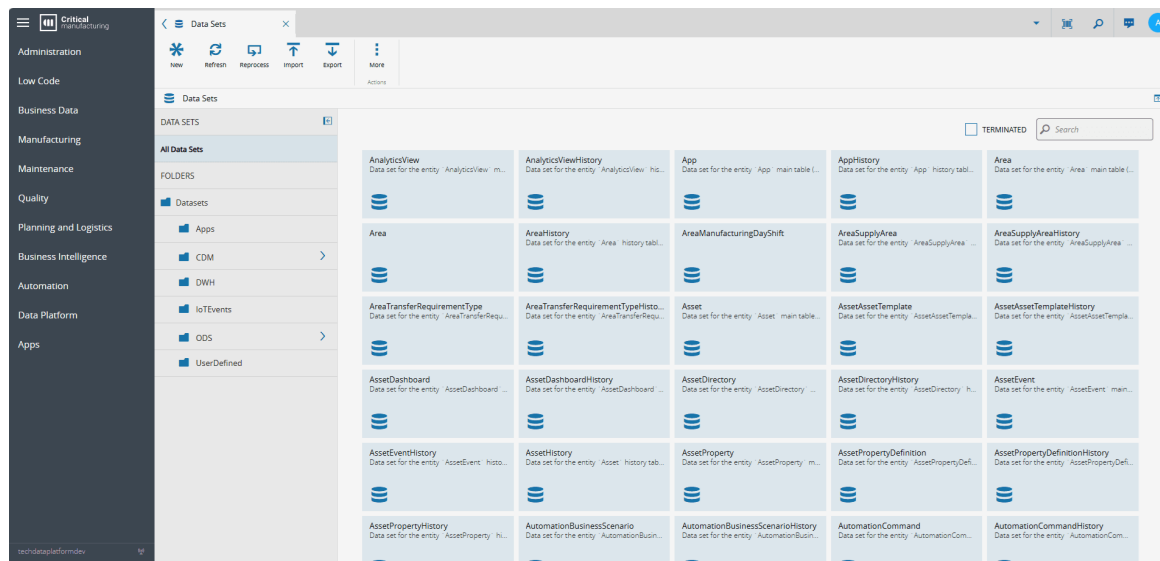
Overview

In this tutorial we describe how **OData** clients can easily access data in CM MES data sets stored in **Data Platform**.

Endpoints

Through **Data Platform**, all data sets can be accessed using the **OData V4** protocol.

If you navigate to the **Data Sets** page in the **Data Platform** section, you can see that the data sets are organized in 6 root folders:



Each folder contains different types of data sets:

- **Apps** - folder for data sets created and used by MES Apps.
- **CDM** - folder for system data sets containing the **Canonical Data Model** (CDM) data.
- **DWH** - folder for data warehouse data sets containing aggregated data.
- **IoTEvents** - folder for data sets associated with IoT events.
- **ODS** - folder for data sets containing MES ODS data.
- **UserDefined** - folder for data sets manually created by MES users.

Each of these folders can be accessed independently via **OData**. For example, to list all available data sets in the **CDM** root folder, the OData client should use one of these endpoints:

- Service Document - <https://<hostname>/datamanager/odata/CDM>
- Metadata - [https://<hostname>/datamanager/odata/CDM/\\$metadata](https://<hostname>/datamanager/odata/CDM/$metadata)

To access the data of data set `MaterialMovement` in the **CDM** root folder:

```
https://<hostname>/datamanager/odata/CDM/MaterialMovement
```

To access the data of data set `ResourceLayout` in the **Resource** folder (which is inside the **CDM** root folder):

```
https://<hostname>/datamanager/odata/CDM/Resource.ResourceLayout
```

Thus, to access data sets that are not in root folders, the path to the data set should be built using the dot separator.

Query Options

The standard OData query options are supported, plus a few extensions:

Query Option	Description	Example
\$select	Pick the columns to get.	<code>https://<hostname>/datamanager/odata/CDM/MaterialMovement?\$select=Enterprise_Name, Facility_Name, Area_Name, Step_Name, Material_Name, InProcessPrimaryQty</code>
\$filter	Filter the data to get.	<code>https://<hostname>/datamanager/odata/CDM/MaterialMovement?\$filter=contains(Material_Name, 'Cookie') and Step_Name eq 'Mixing'</code>
\$orderby	Columns to order the data.	<code>https://<hostname>/datamanager/odata/CDM/MaterialMovement?\$orderby=Material_Name asc, Step_Name desc</code>
\$top	Number of rows to get. If top is not set, Data Platform will return a maximum of 1000 rows.	<code>https://<hostname>/datamanager/odata/CDM/MaterialMovement?\$top=50</code>
\$skip	Number of rows to skip before returning the result.	<code>https://<hostname>/datamanager/odata/CDM/MaterialMovement?\$skip=50</code>
\$count	Add the total number of rows in the unfiltered data set to the reply.	<code>https://<hostname>/datamanager/odata/CDM/MaterialMovement?\$count=true</code>
\$apply	Group and/or aggregate data.	<code>https://<hostname>/datamanager/odata/CDM/MaterialMovement?\$select=Area_Name, Material_Name, TotalTrackIns, TotalTrackOuts & \$apply=groupby((Area_Name, Material_Name), aggregate(TrackInCount with sum as TotalTrackIns, TrackOutCount with sum as TotalTrackOuts))</code>

Query Option	Description	Example
\$query	If the query needs to be sent via POST instead of a GET request (example: the query is too long to be sent with a GET request), the \$query option should be used, and the actual query sent as plain text in the body of the POST request.	<code>https://<hostname>/datamanager/odata/CDM/MaterialMovement?\$query , POST request body in plain text: \$select=Step_Name,Material_Name&\$filter=Material_Name eq 'Cookie01'</code>

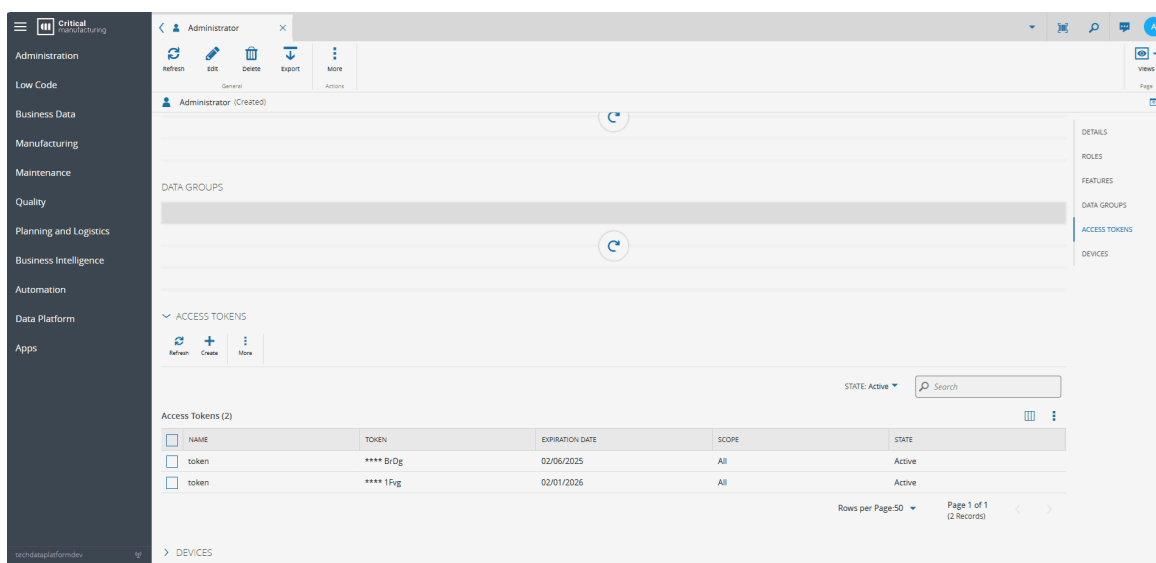
Note

All the options run server-side, so all **OData** queries should be as specific as possible to obtain the best performance.

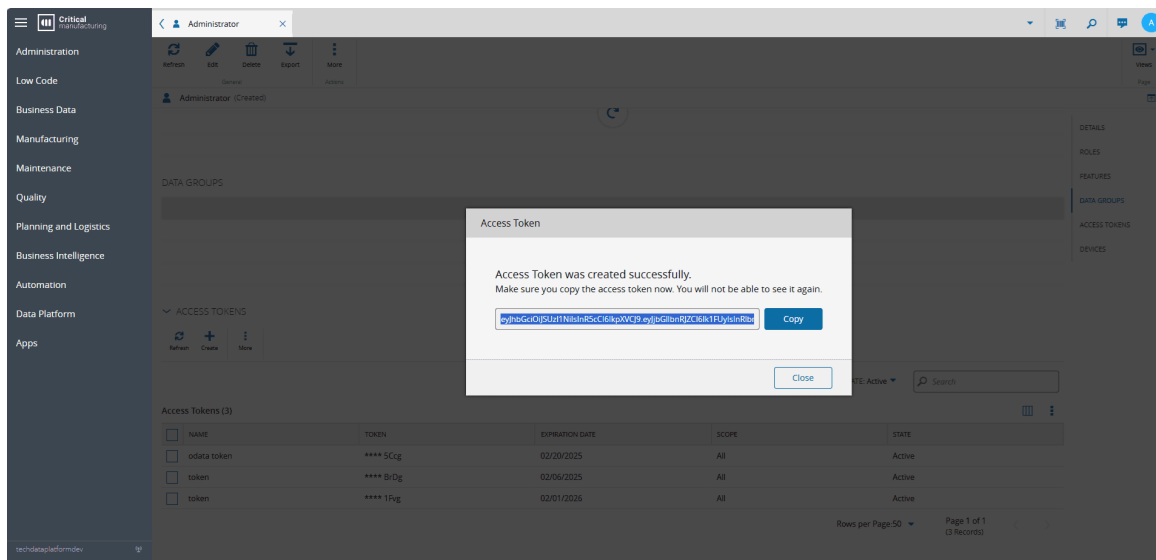
Accessing Data Sets With an OData Client

Any OData client can connect to **Data Platform**. However, the client will need an **Access Token** to retrieve the data.

To create an **Access Token** go to your user page, scroll down to the **Access Tokens** section and **Create** a new **Access Token**:

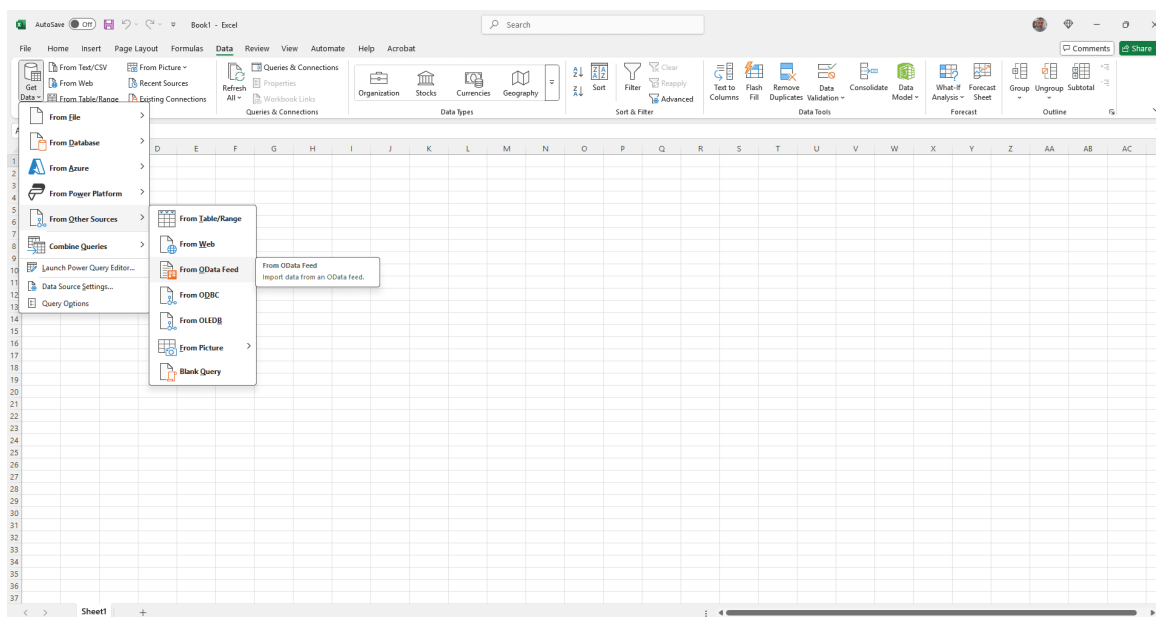


Save the token, so that it can be used in the OData Client to access the data:

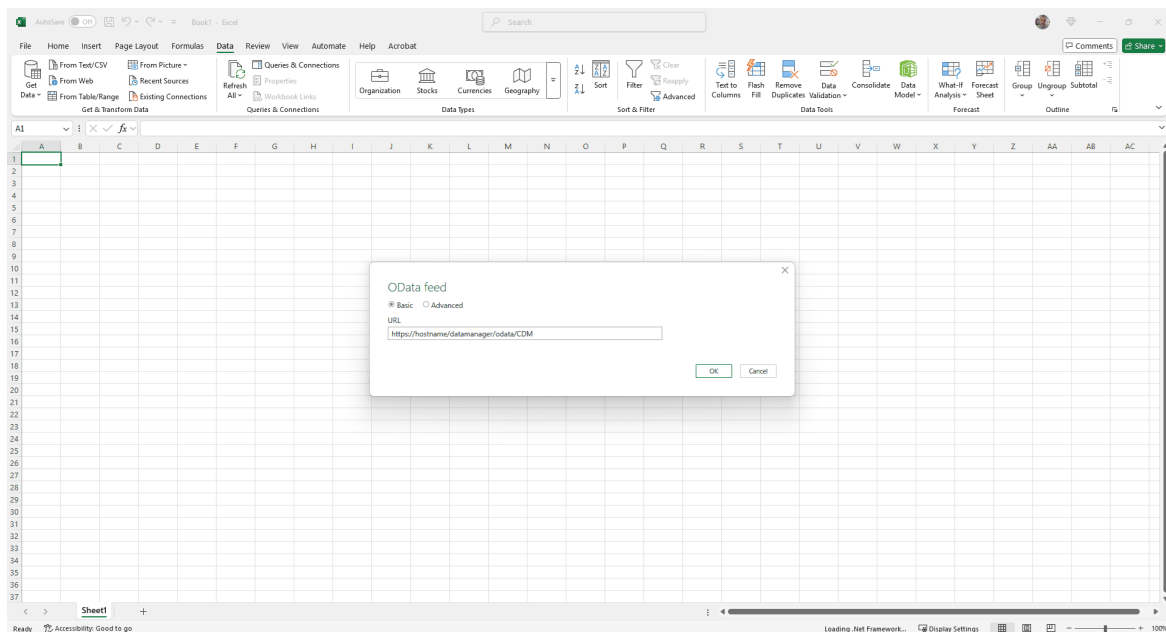


Data Sets in Excel

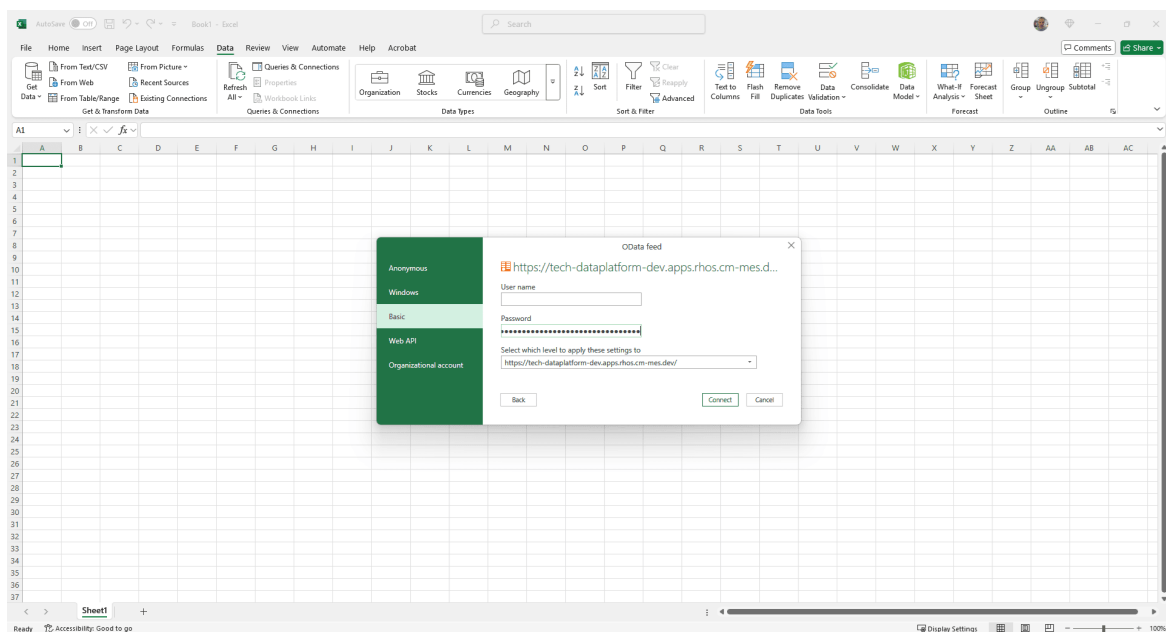
To retrieve data sets in **Excel** go to **Data -> Get Data -> From Other Sources -> From OData Feed:**



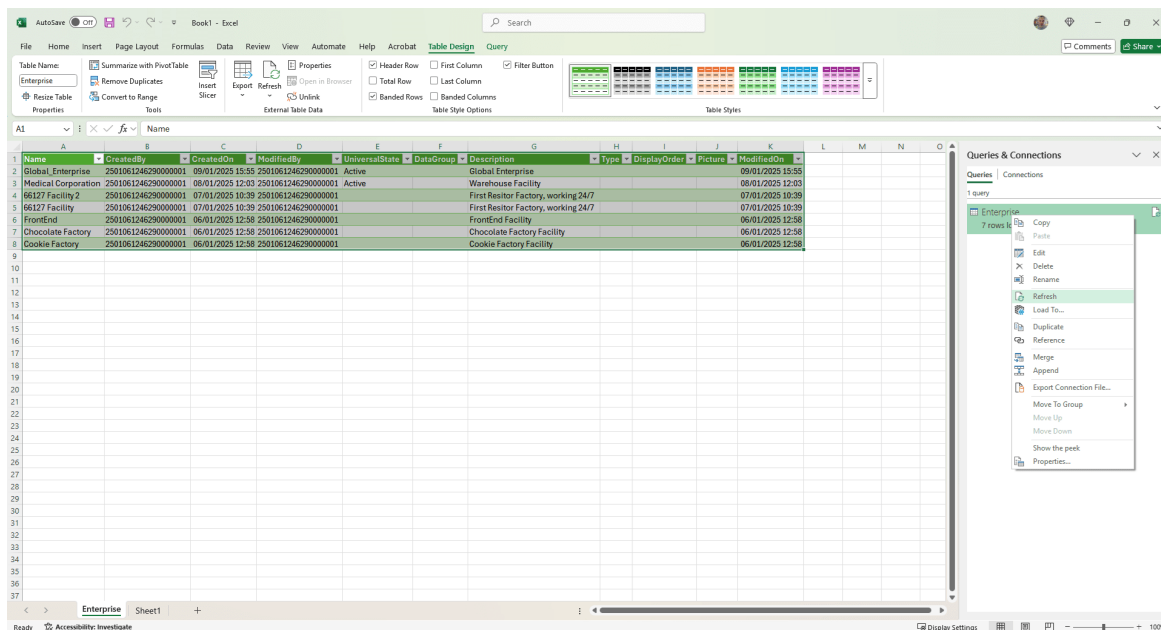
Type the URL for the root folder you want to access, or an OData query to access a specific data set with query options, and select **OK**:



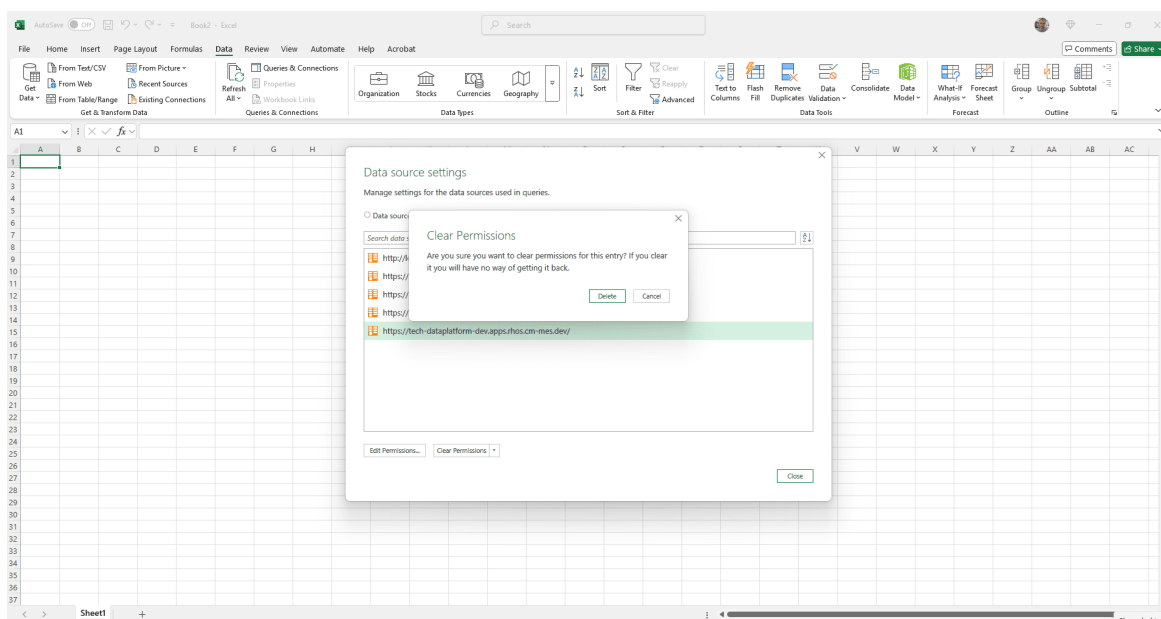
An authentication window will pop-up. Select **Basic**, paste your **Access Token** in the **Password** field and select **Connect** (you can leave the **User name** field empty):



You should now have access to the data in the data set, or to the list of data sets if you gave it an URL to a root folder (in which case you can pick the data set from the list). The data is now available to be processed and refreshed in **Excel**:



Excel caches the access tokens. Thus, if you are having trouble accessing the data (example: your token expired) you should go to **Get Data -> Data Source Settings** and select **Clear Permissions**, select the permissions you want to clear and **Delete** them:



Next time you try to connect to the data set **Excel** will ask for the token again.

Data Sets in Jupyter Notebook

There are several ways to access **OData** data sets in **Jupyter Notebook**. For example, with a regular HTTP request with an OData query:

File

Edit

View

Run

Kernel

Settings

Help

Trust

Python 3 (ipykernel)

```
[18]: import pandas as pd
import requests

# OData URL
service_url = "https://tech-dataplatform-dev.apps.rhos.cm-mes.dev/datamanager/odata/CDM/MaterialMovement?$select=Area_Name,Material_Name,TrackInCount&$top=5"

# Credentials for Basic Authentication
username = ""
password = "eyJhbGciOiJhbnR5b250IiwiaWF0IjOiMjE1OTQ1MjE1InR5cCI6IkpXVCJ9eyJjbGllghfghdghbnRJCi61k1FyUisInRlbmFudE5hbWUvcvcvbnRJCi61NRVhLlCjZzdWl0IjhzG61pb1sInNjb381IjpuZm9sL3Cj"

# Send a GET request with Basic Authentication
response = requests.get(service_url, auth=(username, password))

# Check if the request was successful
if response.status_code == 200:
    data = response.json() # Parse the JSON response
    # If the data has a "value" field, it's often the case in OData responses
    records = data.get("value", [])

    # Convert to a Pandas DataFrame
    df = pd.DataFrame(records)

    # Display the table
    print(df)
else:
    print(f"Failed to retrieve data. Status Code: {response.status_code}")
    print("Response:", response.text)
```

	TrackInCount	Material_Name	Area_Name
0	1	L11194123	66127 Area
1	0	L11194123.P01	66127 Area
2	0	L11194123.P01.B01	66127 Area
3	0	L11194123.P01.B02	66127 Area
4	0	L11194123.P01.B03	66127 Area

Just make sure to use **Basic Authentication** in the request, and set your **Access Token** as the password.