Layer2 Cloud Connector User Documentation

15th September 2023 - Version 11.3

Contents

Overview

The Layer2 Cloud Connector provides an easy and powerful way to synchronize or replicate content from many different data sources. Originally designed for Microsoft SharePoint and Office 365 integration, the Layer2 Cloud Connector has now become an all-purpose synchronization tool, allowing files and records to be synchronized between a virtually unlimited number of different data sources, such as Microsoft SharePoint, Office 365, Exchange, Dynamics, SQL servers, local files, and more. By integrating with Microsoft's ActiveX Data Objects (ADO.NET) platform, the Layer2 Cloud Connector can connect to a vast number of third-party data sources.

Getting Started

System Requirements

Minimum Requirements

The system requirements for a machine to run the Layer2 Cloud Connector are highly dependent on the amount of data that needs to be synchronized. The Layer2 Cloud Connector can run on any machine which has a supported Windows operating system installed. The Layer2 Cloud Connector can function with as little as 2GB of RAM, but we recommend that you have at least 8GB available for the most common scenarios. If you are syncing large amounts of data or have more than 20 connections running regularly, we would recommend that you have even more memory.

Note: Installing on to a 32-bit OS or using 32-bit providers limits the Layer2 Cloud Connector to only 4GB of memory. If your scenario involves large files (2GB+) or many records (100K+), it is recommended that you install the 64-bit version, if possible, so that you do not run into memory issues.

Supported Operating Systems

The Layer2 Cloud Connector is supported on the following operating systems:

- Microsoft Windows 10
- Microsoft Windows 11
- Microsoft Windows Server 2012 R2
- Microsoft Windows Server 2016
- Microsoft Windows Server 2019
- Microsoft Windows Server 2022

Virtual Machines

Running the Layer2 Cloud Connector on Virtual Machines (VM) locally or online, such as in the Microsoft Azure Cloud, is fully supported if the VM is running one of the above operating systems. It is important that the VM can reach your data source and destination, directly or via VPN.

Installing on SharePoint or Database Servers

While possible, it is not a best-practice to install the Layer2 Cloud Connector on SharePoint, Database, or other Application servers. It is advised that you install on a separate Windows server or client in the network, where it can reach all the systems it needs to connect with.

Dependencies

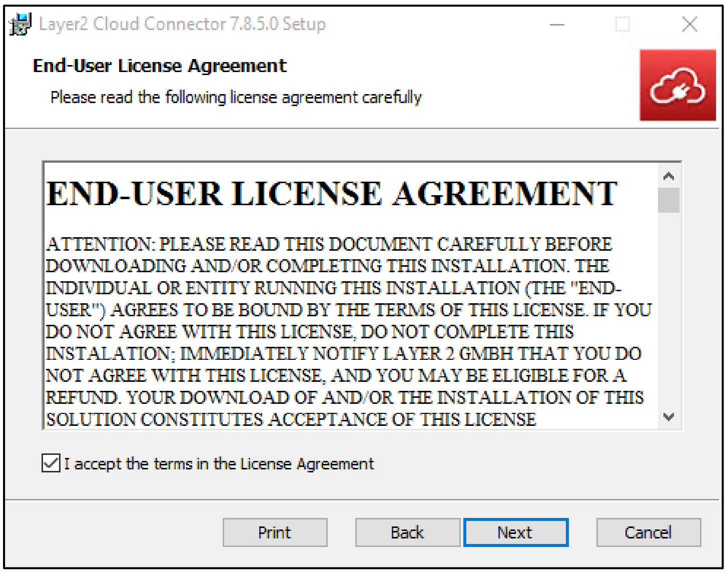
The Layer2 Cloud Connector depends on the following components:

- Microsoft .NET Framework 4.6.2
- Microsoft Management Console 3.0

If the proper .NET Framework is not present on the host machine, the Layer2 Cloud Connector will give an error during installation that it is missing. Please see the MSDN documentation for how to install the .NET Framework.

Setup

1. Extract all files from the provided .ZIP file into a folder on the host-machine.
2. Run the **Setup.msi** file in the extracted folder.
3. Read the license agreement carefully and accept it by clicking the box. If you have any questions concerning licensing, please do not hesitate to contact sales@layer2solutions.com. Otherwise, click **Next**.



Layer2 Cloud Connector 7.8.5.0 Setup

End-User License Agreement

Please read the following license agreement carefully

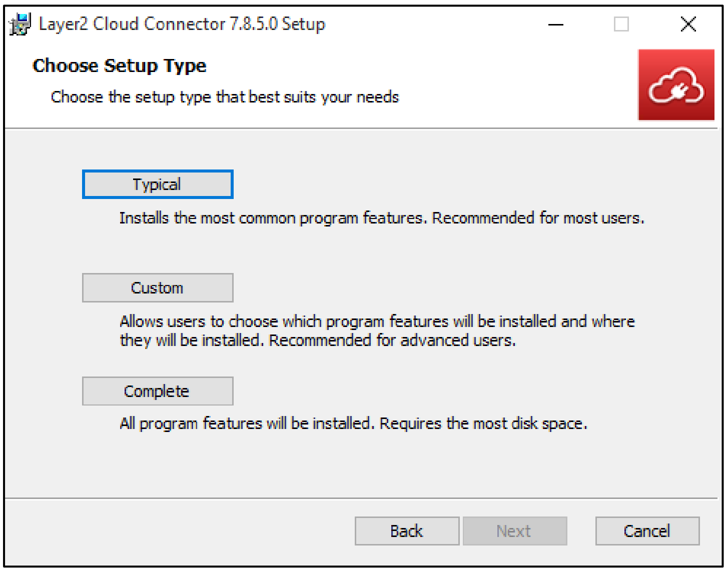
END-USER LICENSE AGREEMENT

ATTENTION: PLEASE READ THIS DOCUMENT CAREFULLY BEFORE DOWNLOADING AND/OR COMPLETING THIS INSTALLATION. THE INDIVIDUAL OR ENTITY RUNNING THIS INSTALLATION (THE "END-USER") AGREES TO BE BOUND BY THE TERMS OF THIS LICENSE. IF YOU DO NOT AGREE WITH THIS LICENSE, DO NOT COMPLETE THIS INSTALATION; IMMEDIATELY NOTIFY LAYER 2 GMBH THAT YOU DO NOT AGREE WITH THIS LICENSE, AND YOU MAY BE ELIGIBLE FOR A REFUND. YOUR DOWNLOAD OF AND/OR THE INSTALLATION OF THIS SOLUTION CONSTITUTES ACCEPTANCE OF THIS LICENSE

☑ I accept the terms in the License Agreement

Figure 1 - Installer license agreement

4. Select appropriate install type. If unsure, select **Typical**.



Layer2 Cloud Connector 7.8.5.0 Setup

Choose Setup Type

Choose the setup type that best suits your needs

Typical: Installs the most common program features. Recommended for most users.

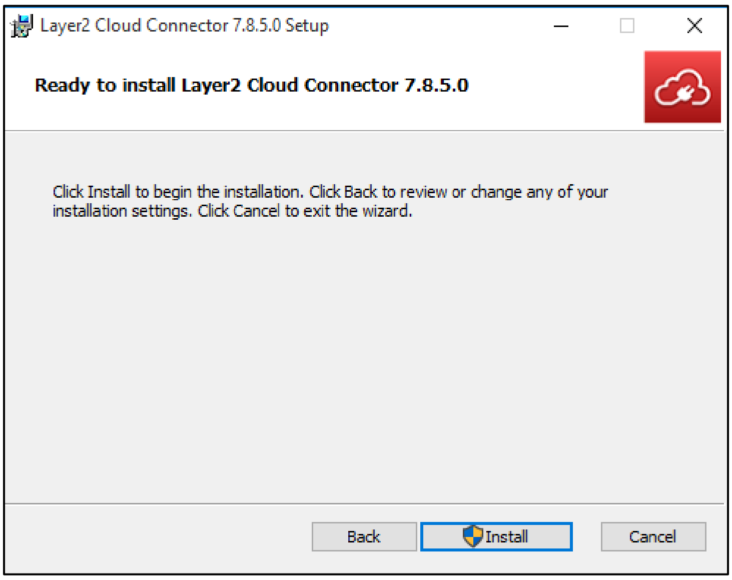
Custom: Allows users to choose which program features will be installed and where they will be installed. Recommended for advanced users.

Complete: All program features will be installed. Requires the most disk space.

Figure 2 - Installer setup type selection

Important - If doing a **Custom** install, you must install the Server, Client, and Data Providers, else the application will not function.

5. Click **Install** to start.



Layer2 Cloud Connector 7.8.5.0 Setup

Ready to install Layer2 Cloud Connector 7.8.5.0

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

Figure 3 - Installation ready to start

6. Once the installation is done, you will have the option to launch the Connection Manager right away by checking the box (recommended). Click **Finish** to complete the installation process.
7. Install the license key file. See the Installing a License section for more details.

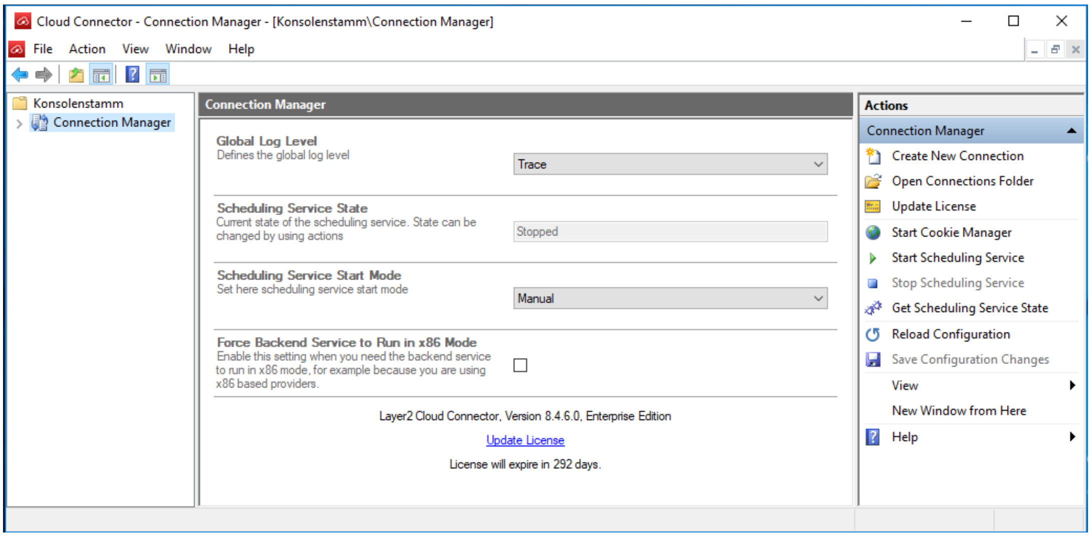
Congratulations! You have successfully installed the Layer2 Cloud Connector and you are now ready to get started with using the Connector to connect and synchronize data between different data sources.

Configuring and Executing Connections

Below are the basic steps necessary to set up a connection between two data sources (referred to as data entities henceforth), along with an example. For more detailed instructions on how to set up a connection to a specific source, please see the online examples here.

1. Open the Layer2 Cloud Connector Connection Manager application (this is listed as **Start Connection Manager** in the Start page/menu). You'll see that the Connection Manager UI is divided into three parts:

- The Connection list pane (left-hand side)
- The Properties pane (middle)
- The Actions pane (right-hand side)



Connection Manager

Global Log Level
Defines the global log level
Trace

Scheduling Service State
Current state of the scheduling service. State can be changed by using actions
Stopped

Scheduling Service Start Mode
Set here scheduling service start mode
Manual

Force Backend Service to Run in x86 Mode
Enable this setting when you need the backend service to run in x86 mode, for example because you are using x86 based providers.
□

Layer2 Cloud Connector, Version 8.4.6.0, Enterprise Edition

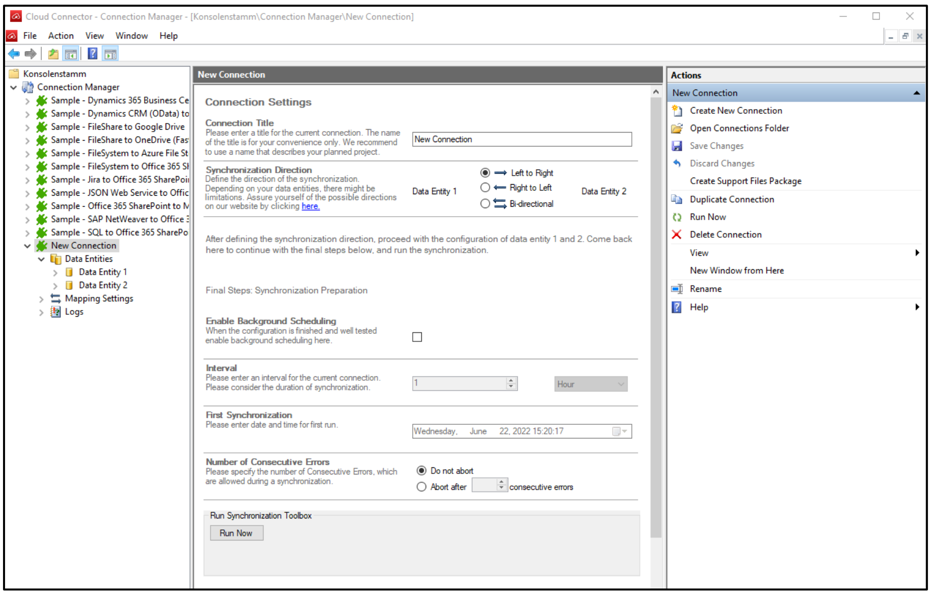
Update License

License will expire in 292 days.

Figure 4 - Layer2 Cloud Connector Connection Manager Application

These panes and their attributes are covered in more detail in the Configuration section.

2. In the right-hand pane, click **Create New Connection**. The new connection will appear at the bottom of the Connection Manager list.
   a. You will also see there are several example connections that you can use as a template to start your own connection. Right-click an example connection that fits your requirements and select **Duplicate Connection** to make a new copy to edit.


Cloud Connector - Connection Manager - [Konsolenstamm\Connection Manager\New Connection]
File
Action
View
Window
Help
Konsolenstamm
Connection Manager
Sample - Dynamics 365 Business Ce
Sample - Dynamics CRM (OData) to
Sample - FileShare to Google Drive
Sample - FileShare to OneDrive (Fas
Sample - FileSystem to Azure File St
Sample - FileSystem to Office 365 Sh
Sample - Jira to Office 365 SharePoi
Sample - JSON Web Service to Offic
Sample - Office 365 SharePoint to M
Sample - SAP NetWeaver to Office 3
Sample - SQL to Office 365 SharePo
New Connection
Data Entities
Data Entity 1
Data Entity 2
Mapping Settings
Logs
Actions
New Connection
Create New Connection
Open Connections Folder
Save Changes
Discard Changes
Create Support Files Package
Duplicate Connection
Run Now
Delete Connection
View
New Window from Here
Rename
Help

New Connection

Connection Settings

Connection Title
Please enter a title for the current connection. The name of the title is for your convenience only. We recommend to use a name that describes your planned project.
New Connection

Synchronization Direction
Define the direction of the synchronization. Depending on your data entities, there might be limitations. Assure yourself of the possible directions on our website by clicking here.
Data Entity 1 ● Left to Right ○ Right to Left ○ Bi-directional Data Entity 2

After defining the synchronization direction, proceed with the configuration of data entity 1 and 2. Come back here to continue with the final steps below, and run the synchronization.

Final Steps: Synchronization Preparation

Enable Background Scheduling
When the configuration is finished and well tested enable background scheduling here.
□

Interval
Please enter an interval for the current connection. Please consider the duration of synchronization.
1 Hour

First Synchronization
Please enter date and time for first run.
Wednesday, June 22, 2022 15:20:17

Number of Consecutive Errors
Please specify the number of Consecutive Errors, which are allowed during a synchronization.
● Do not abort
○ Abort after consecutive errors

Run Synchronization Toolbox
Run Now

Figure 5 - Creating new connection

3. Click on the **New Connection** in the left-hand pane to see its properties.
    - **Connection Title:** Set a meaningful title for your connection.
    - **Direction:** Specify uni- or bi-directional sync of data.
    - **Scheduling enabled:** [Advanced] For enabling automatic background synchronization, see the Schedule Synchronization section. It is **recommended** that you do not enable this until you have a fully functional connection configured.
        - **Interval**
        - **First Synchronization**
    - **Number of Consecutive Errors:** [Advanced] Set the number of errors permitted before a sync is aborted.

      See the Configuration section for a detailed description of each property.
4. Once you have set the **Connection Title**, the **Direction**, and any additional properties, click **Save Changes** in the right-hand pane to save your settings.
5. Expand your new connection and then **Data Entities** to see the objects for the Source and Target data entities for the connection.

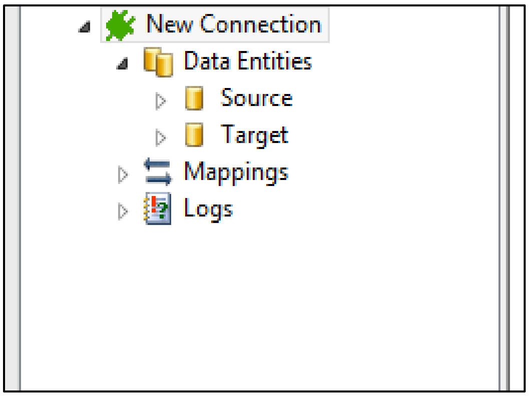
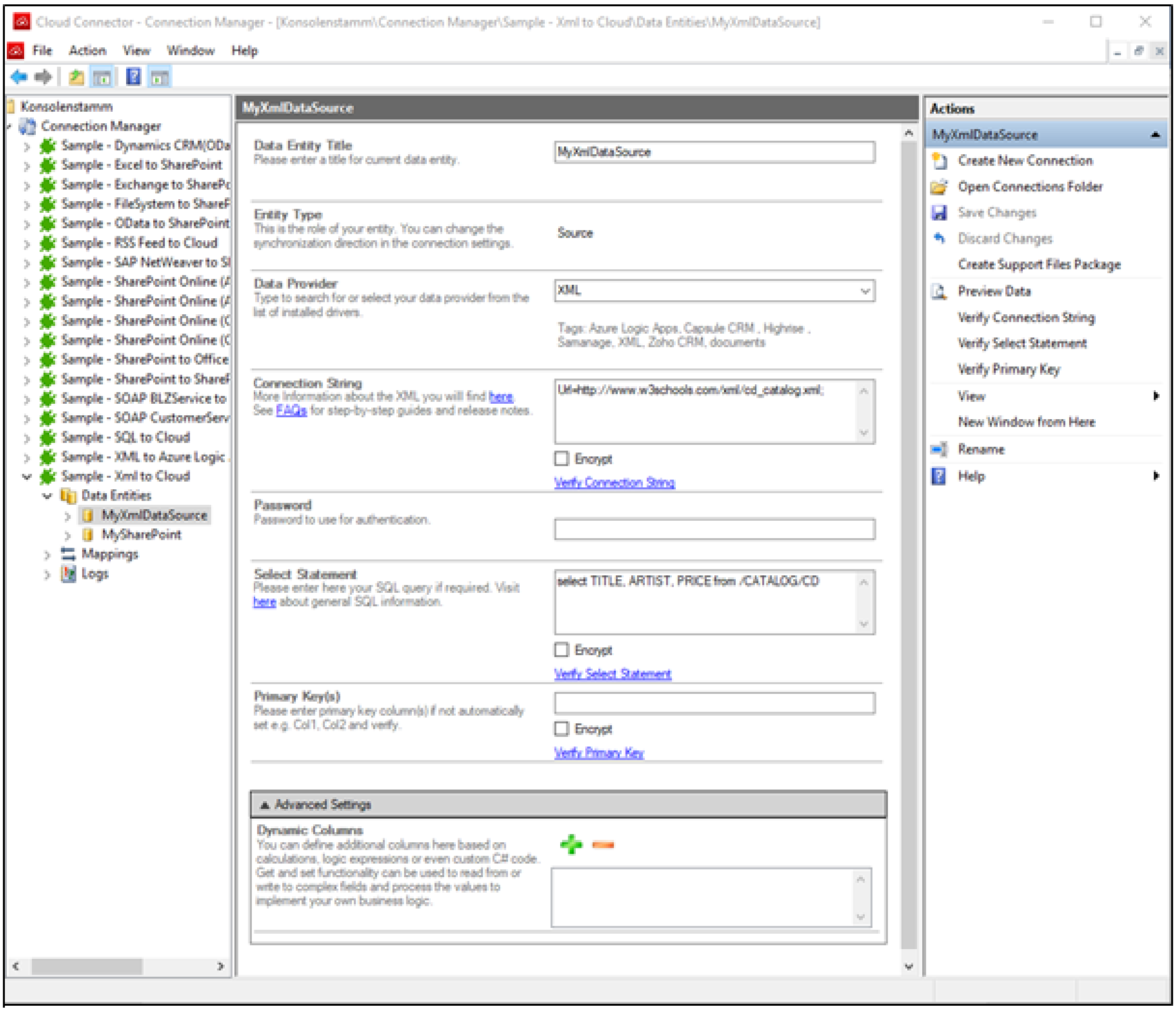

New Connection
Data Entities
Source
Target
Mappings
Logs

Figure 6 - New connection Data Entities

6. Click **Source** to see its properties (these may change based on the selected provider and connection type):
    a. **Data Entity Title:** Set a meaningful title for your Source data entity.
    b. **Data Provider:** Select the correct provider for the data entity. See the Layer2 Data Providers section for specific information.
    c. **Connection String:** Set the necessary parameters to connect to the data entity. See the Layer2 Data Providers section for specific information.
    d. **Select Statement:** Set a provider-specific query, if required, to retrieve the right data.
    e. **Password:** This field can be used to define the password parameter for the provider.
    f. **Primary Key(s):** Usually the data entity will provide this automatically, but if not, one can be defined here.
    g. **Dynamic Columns:** [Advanced Settings] You can define additional columns here based on calculations, logic expressions, or even custom C# code. For more information, see the Dynamic Columns.
    h. **Deletion Protection Threshold:** [Advanced Settings] You can specify how many records can be deleted without asking for confirmation. For more information, see Deletion Protection Threshold.

i. The **Advanced Settings** section can be expanded by clicking the header row.


Cloud Connector - Connection Manager - [Konsolenstamm\Connection Manager\Sample - Xml to Cloud\Data Entities\MyXmlDataSource]
File
Action
View
Window
Help
Konsolenstamm
Connection Manager
Sample - Dynamics CRM(ODa
Sample - Excel to SharePoint
Sample - Exchange to SharePo
Sample - FileSystem to ShareP
Sample - OData to SharePoint
Sample - RSS Feed to Cloud
Sample - SAP NetWeaver to S
Sample - SharePoint Online (A
Sample - SharePoint Online (A
Sample - SharePoint Online (C
Sample - SharePoint Online (C
Sample - SharePoint to Office
Sample - SharePoint to ShareP
Sample - SOAP BLZService to
Sample - SOAP CustomerServ
Sample - SQL to Cloud
Sample - XML to Azure Logic
Sample - Xml to Cloud
Data Entities
MyXmlDataSource
MySharePoint
Mappings
Logs
Actions
MyXmlDataSource
Create New Connection
Open Connections Folder
Save Changes
Discard Changes
Create Support Files Package
Preview Data
Verify Connection String
Verify Select Statement
Verify Primary Key
View
New Window from Here
Rename
Help

MyXmlDataSource

Data Entity Title
Please enter a title for current data entity.
MyXmlDataSource

Entity Type
This is the role of your entity. You can change the synchronization direction in the connection settings.
Source

Data Provider
Type to search for or select your data provider from the list of installed drivers.
XML
Tags: Azure Logic Apps, Capsule CRM , Highrise , Samanage, XML, Zoho CRM, documents

Connection String
More Information about the XML you will find here.
See FAQs for step-by-step guides and release notes.
Url=http://www.w3schools.com/xml/cd_catalog.xml;
☐ Encrypt
Verify Connection String

Password
Password to use for authentication.

Select Statement
Please enter here your SQL query if required. Visit here about general SQL information.
select TITLE, ARTIST, PRICE from /CATALOG/CD
☐ Encrypt
Verify Select Statement

Primary Key(s)
Please enter primary key column(s) if not automatically set e.g. Col1, Col2 and verify.
☐ Encrypt
Verify Primary Key

▲ Advanced Settings

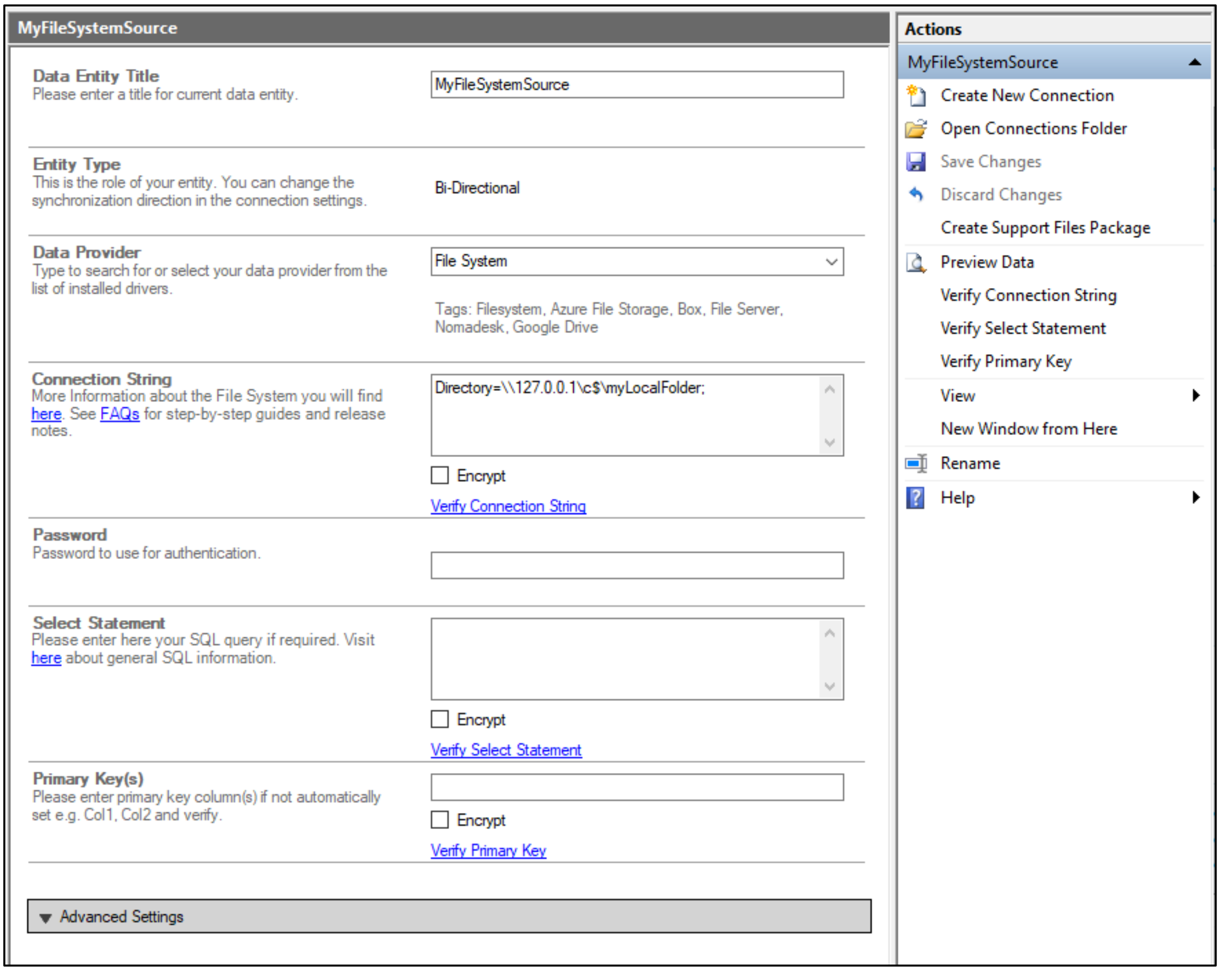
Dynamic Columns
You can define additional columns here based on calculations, logic expressions or even custom C# code. Get and set functionality can be used to read from or write to complex fields and process the values to implement your own business logic.

Figure 7 - Blank source Data Entity

See the Configuration section for a detailed description of each property.

Example:

For a connection that goes from a File System source to a SharePoint document library target, these are the resulting settings:



MyFileSystemSource

Data Entity Title
Please enter a title for current data entity.
MyFileSystemSource

Entity Type
This is the role of your entity. You can change the synchronization direction in the connection settings.
Bi-Directional

Data Provider
Type to search for or select your data provider from the list of installed drivers.
File System
Tags: Filesystem, Azure File Storage, Box, File Server, Nomadesk, Google Drive

Connection String
More Information about the File System you will find here. See FAQs for step-by-step guides and release notes.
Directory=\\127.0.0.1\c$\myLocalFolder;
□ Encrypt
Verify Connection String

Password
Password to use for authentication.

Select Statement
Please enter here your SQL query if required. Visit here about general SQL information.
□ Encrypt
Verify Select Statement

Primary Key(s)
Please enter primary key column(s) if not automatically set e.g. Col1, Col2 and verify.
□ Encrypt
Verify Primary Key

Advanced Settings

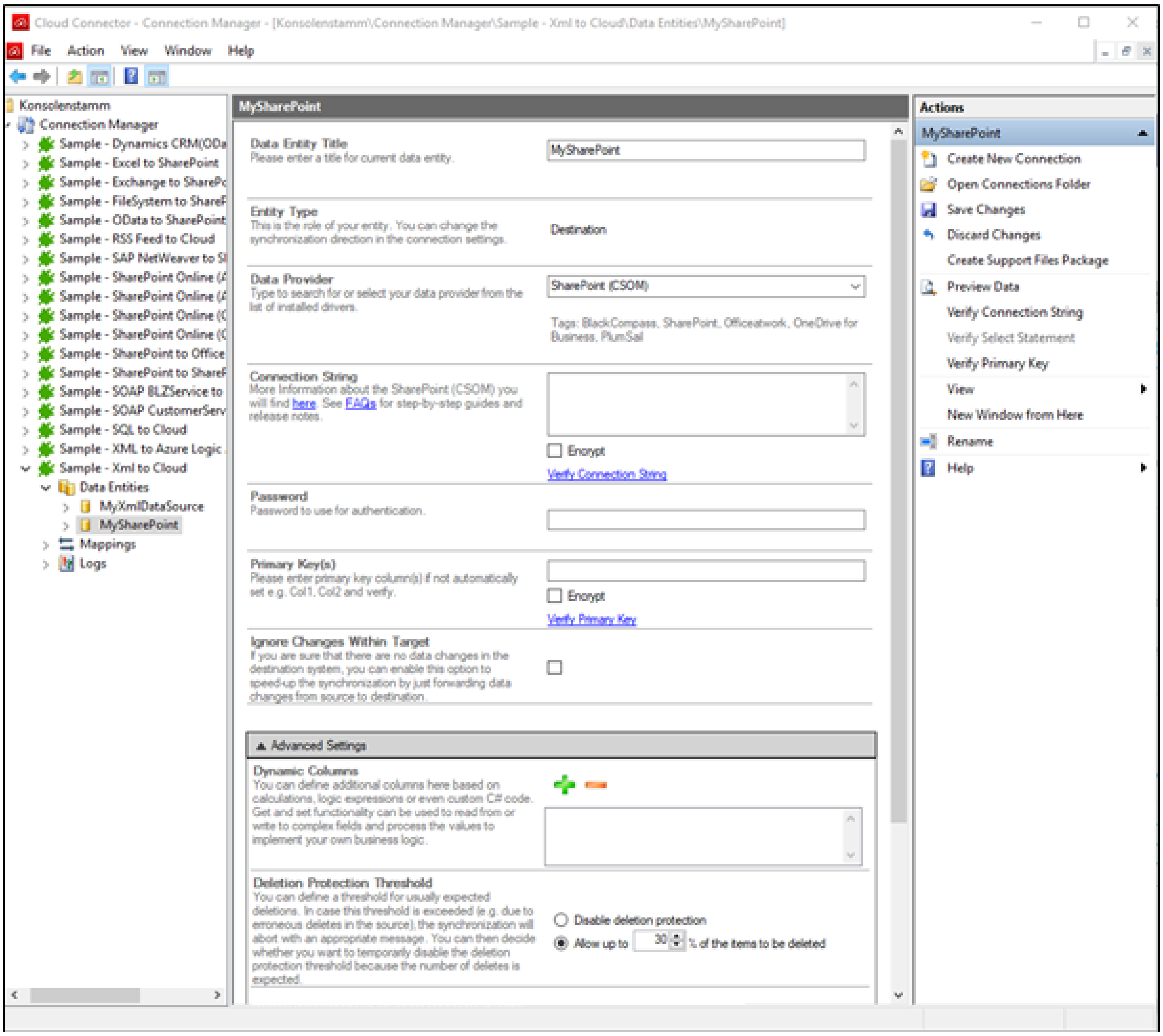
Figure 8 - Example connection for File System

7. Once you have set the **Data Entity Title**, the **Data Provider**, and other required properties, verify the fields using the **Verify *** link, and then click **Preview Data** in the right-hand pane to make sure you have a working connection. If everything is working, click **Save Changes** in the right-hand pane to save your settings.
8. Click **Target** to see its properties (these may change based on the selected provider):
   a. **Data Entity Title:** Set a meaningful title for your Target data entity.
   b. **Data Provider:** Select the correct provider for the data entity. See the Layer2 Data Providers section for specific information.
   c. **Connection String:** Set the necessary parameters to connect to the data entity. See the Layer2 Data Providers section for specific information.
   d. **Password**: This field can be used to define the password parameter for the provider.
   e. **Primary Key(s):** Usually the data entity will provide this automatically, but if not, one can be defined here.
   f. **Ignore Changes Within Target**: [Advanced] If you are sure that there are no data changes in the target system at all, you can speed up the synchronization by enabling this option.

g. **Dynamic Columns:** [Advanced Settings] You can define additional columns here based on calculations, logic expressions, or even custom C# code. For more information, see the Dynamic Columns section.

h. **Deletion Protection Threshold:** [Advanced Settings] See Deletion Protection Threshold for more information.

i. **Disable Operations:** [Advanced Settings] Used to disable certain transactions during synchronization.


Cloud Connector - Connection Manager - [Konsolenstamm\Connection Manager\Sample - Xml to Cloud\Data Entities\MySharePoint]
File
Action
View
Window
Help
Konsolenstamm
Connection Manager
Sample - Dynamics CRM(ODa
Sample - Excel to SharePoint
Sample - Exchange to SharePc
Sample - FileSystem to ShareP
Sample - OData to SharePoint
Sample - RSS Feed to Cloud
Sample - SAP NetWeaver to SI
Sample - SharePoint Online (A
Sample - SharePoint Online (A
Sample - SharePoint Online (C
Sample - SharePoint Online (C
Sample - SharePoint to Office
Sample - SharePoint to ShareP
Sample - SOAP BLZService to
Sample - SOAP CustomerServ
Sample - SQL to Cloud
Sample - XML to Azure Logic
Sample - Xml to Cloud
Data Entities
MyXmlDataSource
MySharePoint
Mappings
Logs
Actions
MySharePoint
Create New Connection
Open Connections Folder
Save Changes
Discard Changes
Create Support Files Package
Preview Data
Verify Connection String
Verify Select Statement
Verify Primary Key
View
New Window from Here
Rename
Help

MySharePoint

Data Entity Title
Please enter a title for current data entity.
MySharePoint

Entity Type
This is the role of your entity. You can change the synchronization direction in the connection settings.
Destination

Data Provider
Type to search for or select your data provider from the list of installed drivers.
SharePoint (CSOM)
Tags: BlackCompass, SharePoint, Officeatwork, OneDrive for Business, PlumSail

Connection String
More Information about the SharePoint (CSOM) you will find here. See FAQs for step-by-step guides and release notes.
☐ Encrypt
Verify Connection String

Password
Password to use for authentication.

Primary Key(s)
Please enter primary key column(s) if not automatically set e.g. Col1, Col2 and verify.
☐ Encrypt
Verify Primary Key

Ignore Changes Within Target
If you are sure that there are no data changes in the destination system, you can enable this option to speed-up the synchronization by just forwarding data changes from source to destination.
☐

▲ Advanced Settings

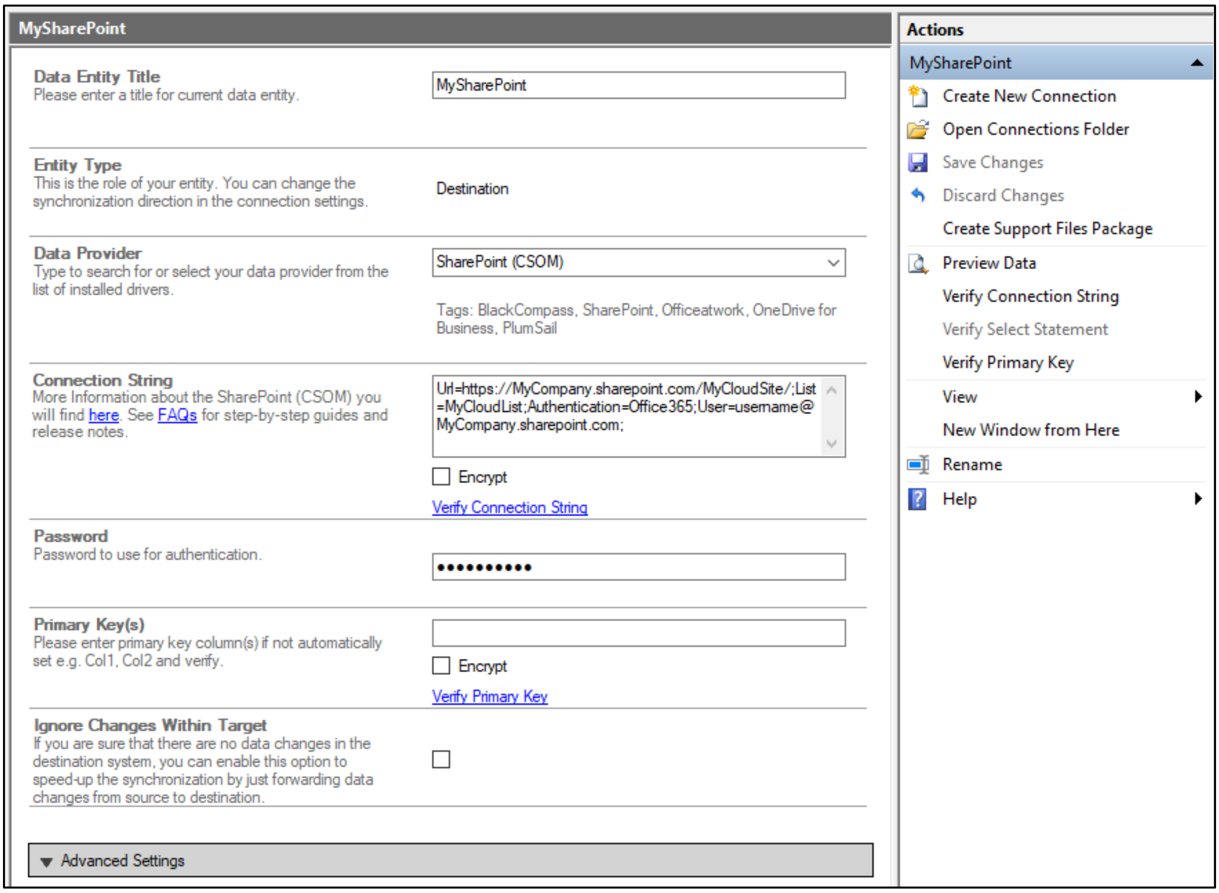
Dynamic Columns
You can define additional columns here based on calculations, logic expressions or even custom C# code. Get and set functionality can be used to read from or write to complex fields and process the values to implement your own business logic.

Deletion Protection Threshold
You can define a threshold for usually expected deletions. In case this threshold is exceeded (e.g. due to erroneous deletes in the source), the synchronization will abort with an appropriate message. You can then decide whether you want to temporarily disable the deletion protection threshold because the number of deletes is expected.
○ Disable deletion protection
● Allow up to 30 % of the items to be deleted

Figure 9 - Blank target Data Entity

See the Configuration section for a detailed description of each property.

Example For a connection that goes from a File System source to a SharePoint document library target, these are the resulting settings:



MySharePoint

Data Entity Title
Please enter a title for current data entity.
MySharePoint

Entity Type
This is the role of your entity. You can change the synchronization direction in the connection settings.
Destination

Data Provider
Type to search for or select your data provider from the list of installed drivers.
SharePoint (CSOM)
Tags: BlackCompass, SharePoint, Officeatwork, OneDrive for Business, PlumSail

Connection String
More Information about the SharePoint (CSOM) you will find here. See FAQs for step-by-step guides and release notes.
Url=https://MyCompany.sharepoint.com/MyCloudSite/;List=MyCloudList;Authentication=Office365;User=username@MyCompany.sharepoint.com;
□ Encrypt
Verify Connection String

Password
Password to use for authentication.
••••••••••

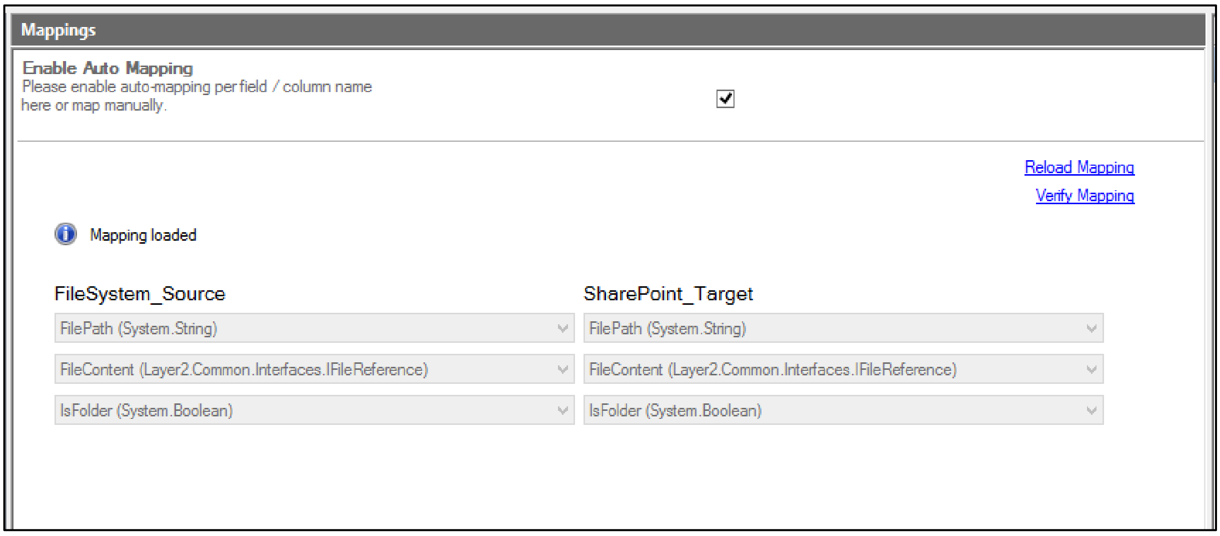
Primary Key(s)
Please enter primary key column(s) if not automatically set e.g. Col1, Col2 and verify.
□ Encrypt
Verify Primary Key

Ignore Changes Within Target
If you are sure that there are no data changes in the destination system, you can enable this option to speed-up the synchronization by just forwarding data changes from source to destination.
□

▼ Advanced Settings

Figure 10 - Example connection for SharePoint

9. Once you have set the **Data Entity Title**, the **Data Provider**, and other required properties, verify the fields using the **Verify *** link, and then click **Preview Data** in the right-hand pane to make sure you have a working connection. If everything is working, click **Save Changes** in the right-hand pane to save your settings.
10. Click **Mappings** in the left-hand Connections pane to set the mapping of fields from the source to the target. Click **Enable Auto Mapping** to have the Connection Manager perform this for you based on identical column/field names from the data entities (otherwise, you will need to set them manually).


Mappings

Enable Auto Mapping
Please enable auto-mapping per field / column name here or map manually. ☑

Reload Mapping
Verify Mapping

Mapping loaded

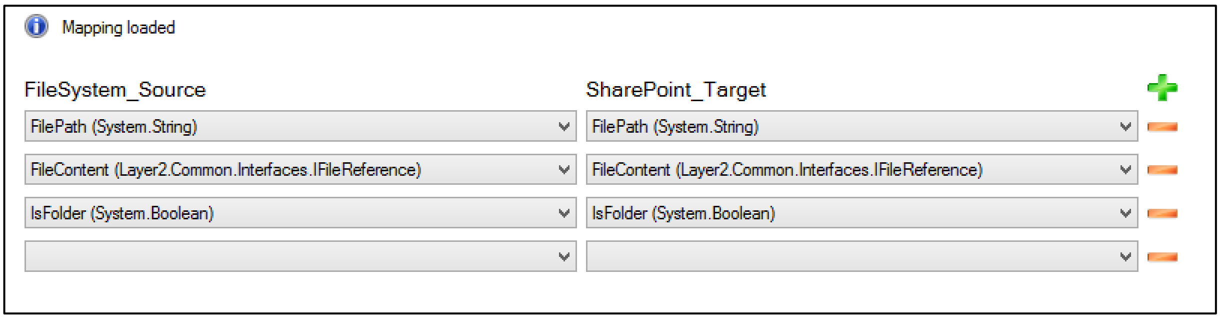
FileSystem_Source	SharePoint_Target
FilePath (System.String)	FilePath (System.String)
FileContent (Layer2.Common.Interfaces.IFileReference)	FileContent (Layer2.Common.Interfaces.IFileReference)
IsFolder (System.Boolean)	IsFolder (System.Boolean)

Figure 11 - Example with Auto Mapping for a file system source and SharePoint document library target

See the Configuration section for a detailed description of the mapping functionality.

Example:
Setting up manual mapping for the File System to SharePoint connection.

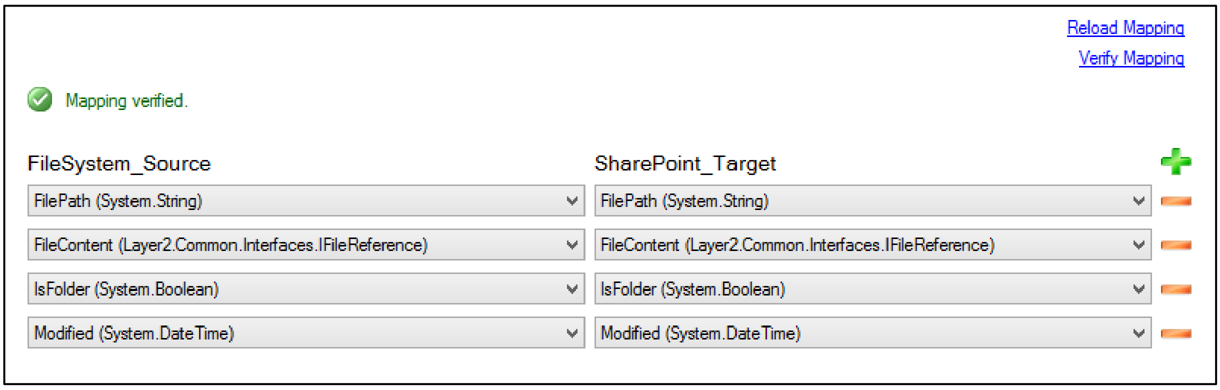
To add a new mapping pair, click the **green '+'**.


Mapping loaded

FileSystem_Source	SharePoint_Target
FilePath (System.String)	FilePath (System.String)
FileContent (Layer2.Common.Interfaces.IFileReference)	FileContent (Layer2.Common.Interfaces.IFileReference)
IsFolder (System.Boolean)	IsFolder (System.Boolean)

Figure 12 - Adding a new mapping pair

From the dropdowns, select the fields that are a matching pair for synchronization.



✓ Mapping verified.

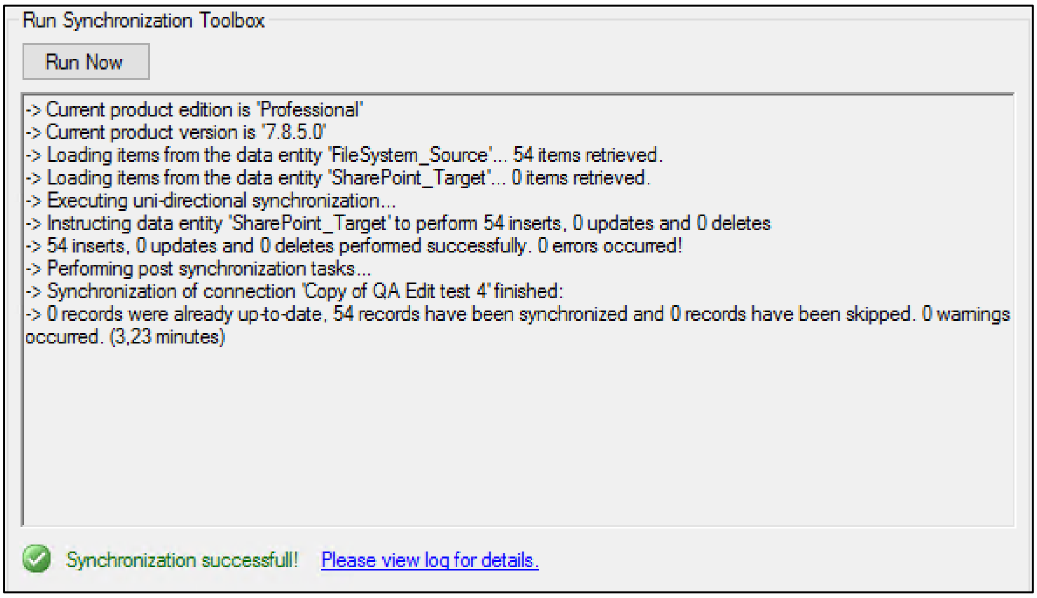
FileSystem_Source	SharePoint_Target
FilePath (System.String)	FilePath (System.String)
FileContent (Layer2.Common.Interfaces.IFileReference)	FileContent (Layer2.Common.Interfaces.IFileReference)
IsFolder (System.Boolean)	IsFolder (System.Boolean)
Modified (System.DateTime)	Modified (System.DateTime)

Figure 13 - Setting new mapped pair for Modified column

Add more fields as necessary by clicking the **green '+'**.

11. When done, click **Save Changes** and then click **Verify Mapping** to test.
12. Now you're ready to run the synchronization! Select your connection in the left-hand pane, and then click **Run now** in the central pane. Check the **Logs** for any warnings or errors and verify that the content from the Source was pushed to the Target (or to each other in the case of bi-directional).

Example:



```
-> Current product edition is 'Professional'
-> Current product version is '7.8.5.0'
-> Loading items from the data entity 'FileSystem_Source'... 54 items retrieved.
-> Loading items from the data entity 'SharePoint_Target'... 0 items retrieved.
-> Executing uni-directional synchronization...
-> Instructing data entity 'SharePoint_Target' to perform 54 inserts, 0 updates and 0 deletes
-> 54 inserts, 0 updates and 0 deletes performed successfully. 0 errors occurred!
-> Performing post synchronization tasks...
-> Synchronization of connection 'Copy of QA Edit test 4' finished:
-> 0 records were already up-to-date, 54 records have been synchronized and 0 records have been skipped. 0 warnings
occurred. (3,23 minutes)
```

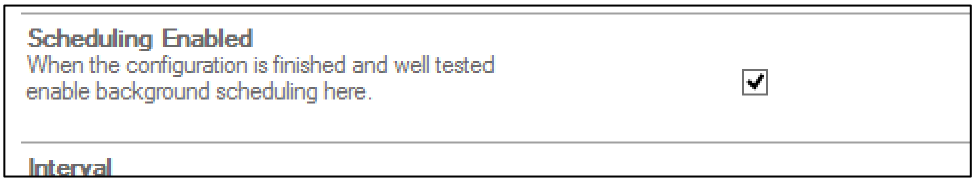
✓ Synchronization successfull! Please view log for details.

Figure 14 - Running the File System to SharePoint connection first time with sync summary data

Schedule Synchronization

Once you have your connection running as expected and without errors when it is run manually, you can now set a schedule to execute automatically in background (without needing to run the Connection Manager). Below are the basics steps necessary to set up automated synchronization schedule for your connection. For more information on the scheduling properties, see the Configuration section.

1. In the Connection Manager, select the connection from the list that you wish to set up scheduling for.
2. Check the box for **Scheduling enabled** in the properties pane.

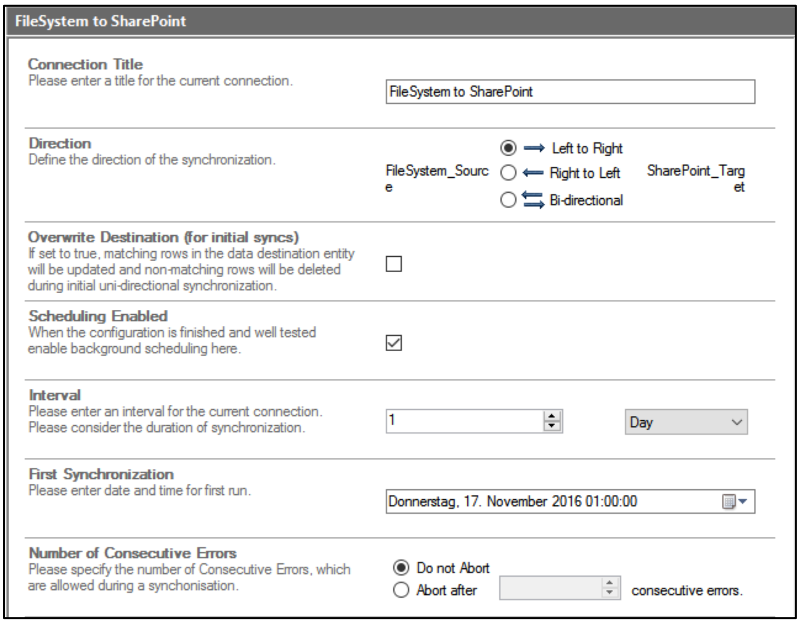


Scheduling Enabled
When the configuration is finished and well tested
enable background scheduling here. ☑

Interval

Figure 15 - Scheduling Enabled property

3. Determine the **Interval** you wish the synchronization to occur at. You can specify by “Minute”, “Hour”, and “Day”, as well as a number value for **Interval**.
   Warning! Make sure to set an appropriate interval based on how long the synchronization of your specific connection usually takes.
4. Set the **First Synchronization** property. This will determine when the automatic synchronization starts, as well as what time subsequent runs will be performed.



FileSystem to SharePoint

Connection Title
Please enter a title for the current connection.
FileSystem to SharePoint

Direction
Define the direction of the synchronization.
FileSystem_Source ● Left to Right ○ Right to Left ○ Bi-directional SharePoint_Target

Overwrite Destination (for initial syncs)
If set to true, matching rows in the data destination entity will be updated and non-matching rows will be deleted during initial uni-directional synchronization.
☐

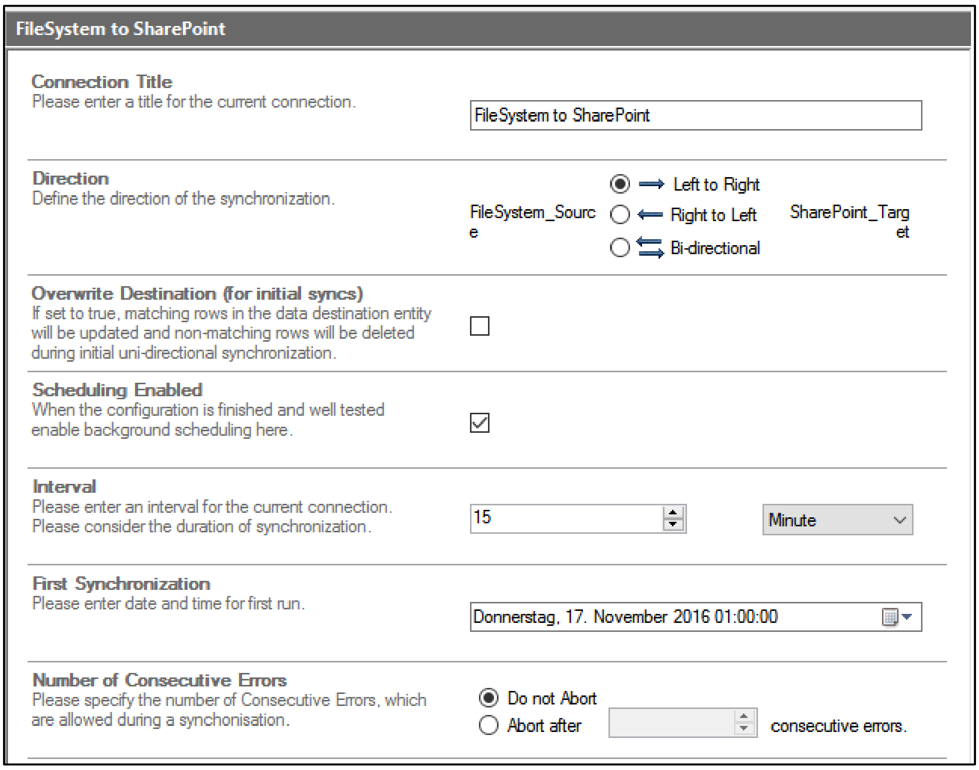
Scheduling Enabled
When the configuration is finished and well tested enable background scheduling here.
☑

Interval
Please enter an interval for the current connection.
Please consider the duration of synchronization.
1 Day

First Synchronization
Please enter date and time for first run.
Donnerstag, 17. November 2016 01:00:00

Number of Consecutive Errors
Please specify the number of Consecutive Errors, which are allowed during a synchonisation.
● Do not Abort
○ Abort after consecutive errors.

Figure 16 – Example: File System to SharePoint set to run once a day at 01:00 AM



FileSystem to SharePoint

Connection Title
Please enter a title for the current connection.
FileSystem to SharePoint

Direction
Define the direction of the synchronization.
FileSystem_Source ● Left to Right ○ Right to Left ○ Bi-directional SharePoint_Target

Overwrite Destination (for initial syncs)
If set to true, matching rows in the data destination entity will be updated and non-matching rows will be deleted during initial uni-directional synchronization.
□

Scheduling Enabled
When the configuration is finished and well tested enable background scheduling here.
☑

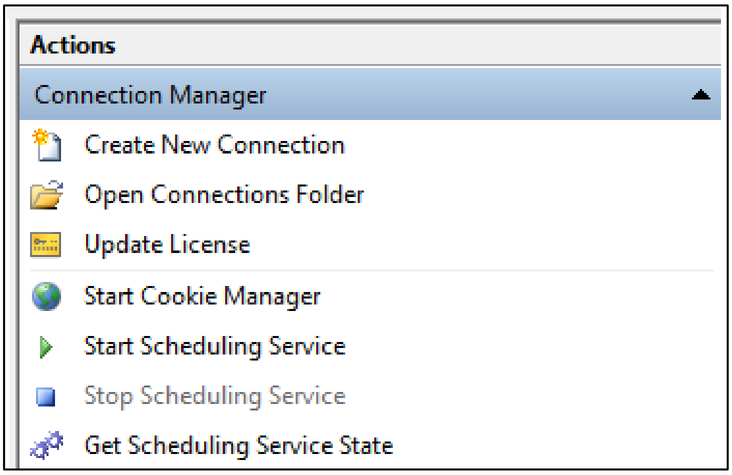
Interval
Please enter an interval for the current connection. Please consider the duration of synchronization.
15 Minute

First Synchronization
Please enter date and time for first run.
Donnerstag, 17. November 2016 01:00:00

Number of Consecutive Errors
Please specify the number of Consecutive Errors, which are allowed during a synchonisation.
● Do not Abort
○ Abort after consecutive errors.

Figure 17 – Example: File System to SharePoint set to run every 15 minutes

5. Click **Save Changes** in the right-hand pane.
6. Start the “Layer2CloudConnectorScheduler” Windows service and make sure it is **running** (the scheduled synchronization will not work without it). See the Scheduling Windows Service section for additional assistance.
    a. You can start the “Layer2CloudConnectorScheduler” from the Start menu/page option **Start Scheduling Service,** or from the Connection Manager itself in the right-hand Action pane.



Actions

Connection Manager

Create New Connection

Open Connections Folder

Update License

Start Cookie Manager

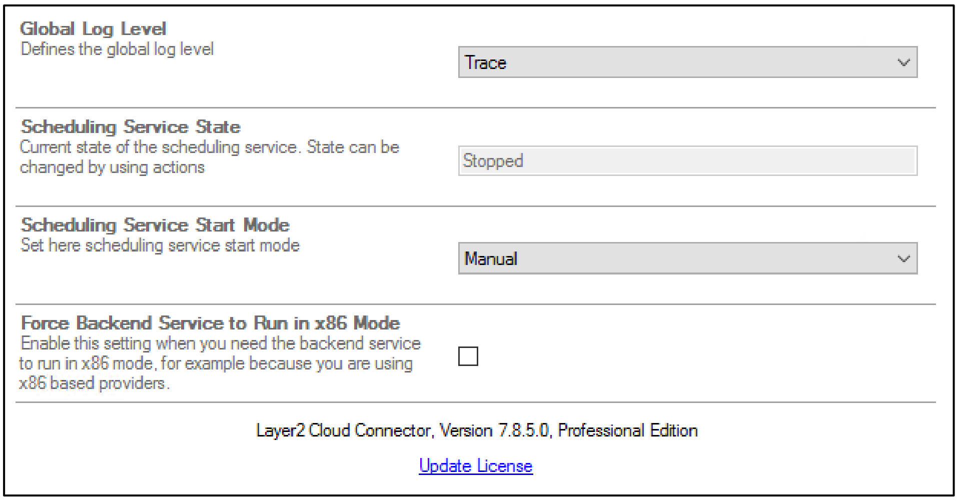
Start Scheduling Service

Stop Scheduling Service

Get Scheduling Service State

Figure 18 - Starting service in the Actions pane

b. From the root node of the Connection Manager, you can see the state of the service under the **Scheduling Service State** property.



Global Log Level
Defines the global log level
Trace

Scheduling Service State
Current state of the scheduling service. State can be changed by using actions
Stopped

Scheduling Service Start Mode
Set here scheduling service start mode
Manual

Force Backend Service to Run in x86 Mode
Enable this setting when you need the backend service to run in x86 mode, for example because you are using x86 based providers.
□

Layer2 Cloud Connector, Version 7.8.5.0, Professional Edition

Update License

Figure 19 - Starting service from the Connection Manager Properties pane

If you have issues with getting the scheduling to run, see [Troubleshooting Scheduled Sync Issues in our FAQ.]()

Logging Basics

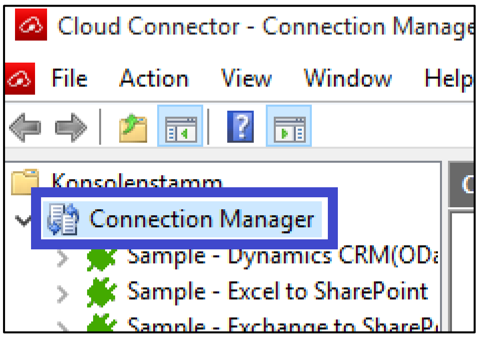
The Layer2 Cloud Connector contains a robust and customizable logging system based on NLog. Here are the basics to getting started with the logging system to help you find and troubleshoot issues with your synchronizations.

Setting the Log Detail Level

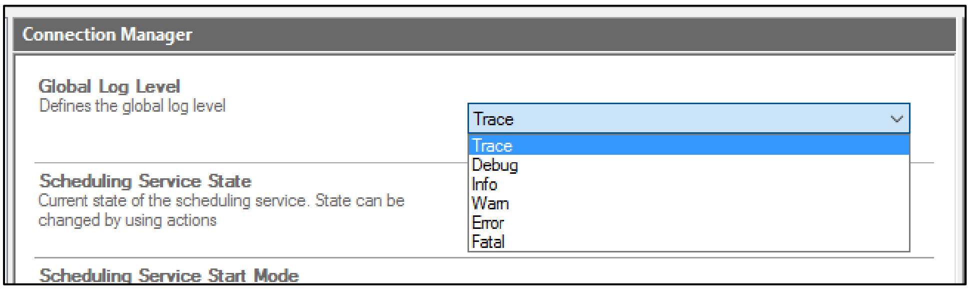
By default, the logging is set to record a “Warn” level of detail, which will only display warnings and critical failures during syncs. The lowest setting is “Fatal” with the least detail and the highest setting is “Trace” with the most detail. For more information, see the Global Log Level section.

To change the log details:

1. Click on the **Connection Manager** node at the top of the left-hand column, above the connections.


Cloud Connector - Connection Manage
File
Action
View
Window
Help
Konsolenstamm
Connection Manager
Sample - Dynamics CRM(ODa
Sample - Excel to SharePoint
Sample - Exchange to ShareP

2. In the middle pane are the global properties, including the **Global Log Level**.
3. In the drop-down, select the level of detail you want for the logging. “Fatal” is least detailed, “Trace” is most detailed.



Connection Manager

Global Log Level
Defines the global log level

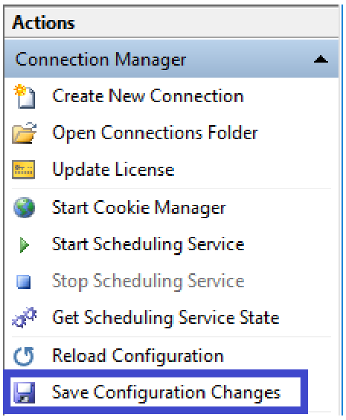
Trace

- Trace
- Debug
- Info
- Warn
- Error
- Fatal

Scheduling Service State
Current state of the scheduling service. State can be changed by using actions

Scheduling Service Start Mode

4. Click **Save Configuration Changes** in the right-hand Action pane to save your log settings.



Viewing the Logs

You can view the latest logs in the Logs nod within the connection. This view has the newest logs at the top and it only shows the most recent information. If the detail is high enough, the earlier logs will be cut off as it is too long. If that is the case, you should click **Open Log File** in the right-hand action pane to see the full log file.

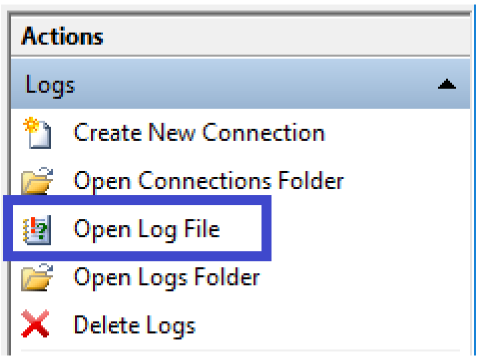


Figure 20 - How to open the full log file for a connection

Customized Logging/Email Alerts

With NLog, you can enable other kinds of logging, including having alerts emailed to you. For more information, see the Logging and Alerting section which describes how to set up some of these alternative logging types.

For more detailed information about the logging system and settings, see these sections in this document:

- Global Log Level
- Logging and Alerting
- Logs

Advanced User's Guide/Technical Information

How the Layer2 Cloud Connector Works

This section will go a little deeper into the inner workings of the Layer2 Cloud Connector to provide a glance behind the scenes of the synchronization mechanism leading to a better understanding of the Layer2 Cloud Connector system overall.

The Metabase

The Layer2 Cloud Connector stores information about the synchronized data on the local disk of the machine where it is installed. This data store is called the Metabase, and it is essentially the internal change record for the Layer2 Cloud Connector. With the help of the Metabase, the system can figure out which data source items have been modified since the last sync.

The Metabase is updated by the Layer2 Cloud Connector with every synchronization run. If configuration settings like the type of connection or field mappings have been changed, the Metabase is also adapted.

The Metabase is stored in the Layer2 Cloud Connector Data Directory as a binary file (see The Layer2 Cloud Connector Data Directory section). If a Metabase file is deleted, it will be re-created automatically by the Layer2 Cloud Connector during the next synchronization run, but the links between source and destination records will be lost, and it will be assumed that all records are new. That's why deleting the Metabase file for a connection with two synchronous data sources may lead to duplicate the data with the next synchronization run.

Under normal circumstances, the **Metabase file should not be deleted**.

However, deleting the Metabase file is sometimes necessary. For example, if the type of the ID field has changed from GUID to integer in one of the data sources, it is not possible for the Layer2 Cloud Connector to convert the IDs inside of the Metabase. There are also situations where the Metabase may be corrupted, such as having the host machine crash during a write to the Metabase. In these kinds of situations, deleting the Metabase will be required to clear the error.

The effect of deleting the Metabase will remove any associations between the records of the data entities. Depending on the configuration, this can lead to:

- **Duplication of the records:**
  On a bi-directional synchronization, records will be duplicated since the Layer2 Cloud Connector does not know they are associated.

- **Deletion and re-adding all records:**
  This will happen on uni-directional connections where the primary keys on both sides are

not associated and if you are not using deletion protection. Otherwise, the synchronization will either just synchronize normally (associated primary keys) or abort and ask for confirmation (deletion protection).

- **Updating all records:**
  There is a special case that comes into play when there is a bi-directional synchronization and the primary key of one side has been mapped to a writable field on the other side, which is usually the case with file-synchronizations. In this case, the Layer2 Cloud Connector can determine the record-associations and rebuilds the Metabase automatically. This can also occur with uni-directional syncs if the primary key field is the same values on both sides and the key fields are mapped.

Metabase Auto-Backup

The Metabase is saved periodically (every hour by default) during a synchronization, so that in case of an unexpected application or service termination, already performed operations will not entirely be repeated on the next synchronization run. The interval of the auto-backup, expressed in hours, is specified by a registry value *AutoBackupInterval* which is described in more detail under the Registry Keys section.

You can disable this feature by using a negative value for the interval.

Field-Mappings and Type-Conversions

The Layer2 Cloud Connector system features a smart, tolerant conversion and comparison engine to provide field mappings between various field types. The following table shows a digest of the most common data types and visualizes which kinds of conversions are supported and which are not. Note that for bi-directional synchronization it is necessary that conversion is supported in both directions for a mapped field pair.

Layer2 Cloud Connector Type Conversions		To				
		String	DateTime	Integer	Float	Boolean
From	String	✓	✓	✓	✓	✓
	DateTime	✓	✓	✗	✗	✗
	Integer	✓	✗	✓	✓	✓
	Float	✓	✗	✓	✓	✗
	Boolean	✓	✗	✓	✗	✓

There are two different modes to define mapping between the fields of both data sources: manual and automatic. Manual mapping is performed by choosing specific fields from both data sources and associating them as a mapped pair. If automatic mapping is activated, the Layer2 Cloud Connector will consider all fields that have the same name (ignoring the case) as mapped.

Primary Keys

The Layer2 Cloud Connector requires a primary key from both data sources to identify the records. Normally, the data providers will provide a primary key, but there are cases where the primary key is not returned by the provider or where there is no primary key. If, for example, an Excel-sheet is used as a data source, there will be no primary key available. It is then necessary to manually define one or more fields as a primary key while configuring the connection.

If the data source provides a primary key, which is the most common case, it is not necessary to manually define a key. All the Layer2 data providers included in the product will return a primary key.

If the data source does not return a primary key, one of three things has happened:

- **The ADO.NET provider does not populate primary keys:**
  Some ADO.NET providers do not populate the primary keys, even though the data source has one.
- **The source has no primary key defined**:
  This case happens when the provider supports primary keys, but one is not passed to the Layer2 Cloud Connector, such as when a SQL database does not have a primary key defined for the table being accessed.
- **The data source does not have a primary key at all:**
  This happens when the data source itself does not support primary keys.

In all these cases, a primary key must be defined in the connection configuration. Any existing field can be used, if it is unique throughout all the records (for example: an ID number, customer number, or account name). If no such field is available, it is also possible to define multiple fields as a combined primary key, with a comma separating each field (for example, column1, column2). The field content must be unique when combined for each item if you are using multiple fields.

Data Providers

The Layer2 Cloud Connector synchronization process goes through three distinct phases:

1. First is the data read phase, where it connects to both data sources that are configured to be synchronized and retrieves their data.
2. Next is the synchronization phase, where the Layer2 Cloud Connector inspects record by record, field by field, comparing and updating all three data sources (the third ‘data source’ is the Metabase) in memory.
3. Lastly is the data writing phase, after the synchronization is complete, the modified data will be written back to the data sources.

The data-read-phase and the data-write-phase involve the data providers, which are plugins that implement a certain kind of data access. They are based on a Microsoft framework called ActiveX Data Objects (ADO). Accessing the data sources through this framework makes the Layer2 Cloud Connector independent from how the data is retrieved and enables it to work with many existing ADO.NET providers.

Common ways to acquire data providers:

- Layer2 data providers are included in the distribution package and licensed with the Layer2 Cloud Connector - see the Layer2 Data Providers section for the list of providers. For a complete list of providers per license, please refer to the online Compare Sheet.
- Some data providers are installed by default on Windows: ODBC, OLEDB, Text/CSV.
- Some are freely available directly from system or application vendors, like for Microsoft SQL Server, Oracle, or IBM iSeries.
- Some are available from 3rd-party vendors that are specialized in that business (these may need to be purchased).

See the Solutions page online to find out which data provider is necessary for a particular application.

Important – While the Layer2 Cloud Connector can utilize both 32- and 64-bit providers, by default it uses 64-bit ones if installed on a 64-bit OS. If you need to use a 32-bit provider, you must enable **x86 Mode**. This is explained in more detail in the Force Backend Service to Run in x86 Mode section.

Uni-directional Synchronization

Uni-directional synchronization uses one data source as the “Source” and one data source as the “Target”. The data from the Source will be accessed as read-only and all detected changes to the Source will be written to the Target. Any changes in the Target that do not match the Source will be changed to match the Source.

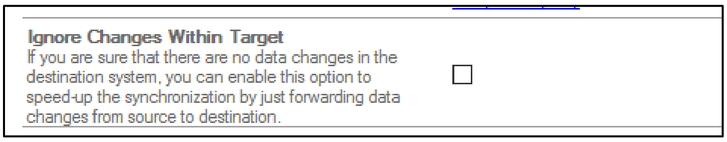
In this mode, the Layer2 Cloud Connector will perform the synchronization in the following way:

- **If two mapped fields have different values:** the Target will get the value from the Source.
- **If a record only exists in the Source:** the record will be added to the Target.
- **If a record only exists in the Target:** the record will be deleted in the Target.

With the above process, a configuration mistake, like setting synchronization from A to B instead of from B to A, would be devastating since all the data in the Target would be deleted. Therefore, the Layer2 Cloud Connector implements following security mechanism: If a synchronization would delete 30% or more of the records on either side, it will abort and ask for confirmation. You can read more about it in the Deletion Protection Threshold section.

The option **Ignore Changes Within Target** can be enabled on the target data entity to speed up the uni-directional synchronization. This setting causes it to not read the data from the target for the comparisons as described above but will rely on the data in the Metabase. For more details, see the Ignore Changes Within Target section in the Data Entity description.

Important – Enable this option with care! If changes are made to the Target outside of the sync, it will cause synchronization errors.



Ignore Changes Within Target
If you are sure that there are no data changes in the destination system, you can enable this option to speed-up the synchronization by just forwarding data changes from source to destination. □

Figure 21 – The option to ignore changes within the target data source

Bi-directional Synchronization

The Bi-directional synchronization process reads from and writes to both data entities of the connection, using the Metabase to decide which side and records need to be updated.

In this mode, the synchronization will be performed in the following way:

If two mapped fields have different values:

In this case, the Layer2 Cloud Connector checks the Metabase and determines which value has changed. If both values are changed since the last run, the Layer2 Cloud Connector detects that there is a data conflict and will use the user-defined conflict resolution strategy to determine what action should be taken. See the Conflict Resolution section for the options available. By default, the synchronization will abort when it runs into this case.

If a record only exists in one data source:

- If a record is found in the Metabase but not in another data source, it would have existed in all data sources (including the Metabase) after the last synchronization but must have been deleted in a data source between now and then. Thus, the record will be deleted from the Metabase and from the other data source.

- If the record cannot be found in the Metabase, it did not exist after the last synchronization and must have been added to a data source, which causes the Layer2 Cloud Connector to insert the record in the Metabase and into the other data source.

Conflict Resolution

When the Layer2 Cloud Connector encounters a conflict, meaning a field has been changed on both sides of the synchronization (Target and Source) since the last synchronization, it will check the conflict resolution strategy of the connection set by the user, which can be one of the following:

FailAndAbort

With this strategy, the Layer2 Cloud Connector will stop the synchronization process and report an error containing information about which fields are in conflict and their values. It would then be necessary to check the data sources directly and resolve the conflict by updating the data manually. This is the least elegant of the strategies as it halts the sync process, but it's also the safest in terms of data consistency.

This strategy is recommended for scenarios where conflicts are very unlikely but would have a big impact if they were to occur.

This conflict resolution strategy is the **default setting**.

WarnAndContinue

This strategy causes the Layer2 Cloud Connector to put out a warning into the log and continue with the synchronization process. The fields in question **will not be updated**, which means that the conflict remains in the system and the warning will occur with every synchronization run until it is manually resolved.

This strategy is recommended for scenarios where conflicts would have a big impact, but the synchronization needs to continue for the other objects. With this it is recommended to have someone check the logs regularly to manually resolve the conflicts.

WinnerLoser

With this strategy, the user defines which of the data entities is the winner (making the other one the loser) in the Mappings properties (see the Configuration section for more details on this property). When a conflict occurs, the Layer2 Cloud Connector will synchronize the winning source's value to the loser and to the Metabase. This is one of the strategies where the resolution is executed automatically by the Layer2 Cloud Connector, requiring no user intervention.

This strategy is recommended for scenarios where one of the data entities can be defined as a master, which will always provide the determining value whenever a conflict occurs. It is also useful that the strategy is hands-off, meaning no user needs to check for and resolve the conflicts.

Important – It is possible with this scenario that the 'Loser' entity had the most recent copy of the change but is overwritten because another user updated the 'Winner' since the last data sync, causing potential data loss. Make sure your users understand this to limit this from occurring.

KeepBoth

This conflict resolution strategy is currently available for file synchronizations with the Layer2 File System Provider or the Layer2 SharePoint Provider. With this strategy, one data entity will be chosen as the 'master entity'. Whenever a conflict is detected, the master entity will keep its changes while the other, non-master entity will copy its changes to a new file which will have "(L2CC-CONFLICT)" appended to the file name (for example: “myFile (L2CC-CONFLICT).txt”). The changes from the master entity will then be written to the original non-master entity file (in this case, “myFile.txt”).

The original non-master file will be kept up to date with the master entity, while the file marked with “(L2CC-CONFLICT)” will no longer be part of the synchronization process (effectively ignored). Note that the original non-master file cannot be changed as long as the “(L2CC-CONFLICT)” file exists. This will cause an error to be thrown on the next synchronization run.

To resolve the conflict, the two files on the non-master side (the original and “(L2CC-CONFLICT)”) need to be merged manually into the original file and the “(L2CC-CONFLICT)” file needs to be removed. On the next synchronization run, the changes from the merge to the original file will be updated on the master entity and the conflict will be considered resolved.

It is recommended to only use KeepBoth if only very few conflicts are expected as each conflict needs to be resolved manually.

Layer2 Cloud Connector Components

The Layer2 Cloud Connector system consists of multiple components which are installed during setup and are described in more detail below.

The Connection Manager

The Connection Manager is a user interface for managing Layer2 Cloud Connector connections. It is integrated into the Windows operating system as a snap-in for the Microsoft Management Console (MMC) 3.0, allowing the administration of the Layer2 Cloud Connector connections by using a familiar and established interface. The Connection Manager is discussed in more detail in the Configuration section.

The Backend Windows Service

This is the core of the Layer2 Cloud Connector, responsible for synchronization logic, connection management, and most other Layer2 Cloud Connector features. It is installed in the system as **Layer2.Data.Synchronization.BackendService / Layer2CloudConnectorBackendService** and can be accessed via an API or by using a client, such as the Connection Manager. For additional information about the Backend Service and its API, see the Backend Service section.

The Scheduling Windows Service

Layer2 Cloud Connector connections can be enabled to synchronize automatically in the background on a pre-defined schedule. This is facilitated by a Windows Service which is installed in the system as part of the Layer2 Cloud Connector: **Layer2.Data.Synchronization.Service** / **Layer2CloudConnectorScheduler**. The service is disabled by default and can be started using either the Connection Manager or the Windows Service Management. For additional information, see the Scheduling Service section.

The Layer2 ADO.NET Providers

The Layer2 Cloud Connector comes with several ready-to-use ADO.NET providers to access a big variety of systems, such as SharePoint lists (2013, 2016, 2019, Subscription Edition, and SharePoint Online), OData services, XML files, RSS feeds, Microsoft Exchange Server (on-premises or cloud-based), and SOAP web services which are part of Layer2 Cloud Connector Personal and Standard. For a complete list of providers available with Layer2 Cloud Connector Enterprise, please refer to the following online Comparison Sheet on our internet site.

These providers are currently bound to be used with the Layer2 Cloud Connector or other Layer2 products. This means that they will be limited in functionality if used in absence of a Layer2 Cloud Connector license, and even with a valid license, the usage of the providers in other contexts than the Layer2 products is, while technically maybe possible, not supported. See the Layer2 Data Providers section for details on the providers which are included in the Layer2 Cloud Connector.

The Layer2 Cloud Connector as Console Application

In addition to using the Connection Manager or the Windows Scheduling Service to start a synchronization, the Layer2 Cloud Connector provides another way to do that. The console application is available in the Start menu as **Start synchronization manually**. For client operating systems since version 8 and server operating systems since 2012, the 'Start synchronization manually' application is available in the **Start** screen under Apps. When it is used, it will then synchronize all enabled connections or a specified one and put out real-time messages about its activity. The console application is discussed in greater detail in the Console Mode section.

The Cookie Manager [Deprecated]

The Cookie Manager offers an additional option to authenticate to Microsoft Office 365, CRM Online, or SharePoint FBA. It uses the technique of adding federation cookies to the target server connection request object. The Cookie Manager can be started from Connection Manager by clicking **Start Cookie Manager** in the right-hand pane.

This is a deprecated feature and is normally not required to authenticate with supported systems.

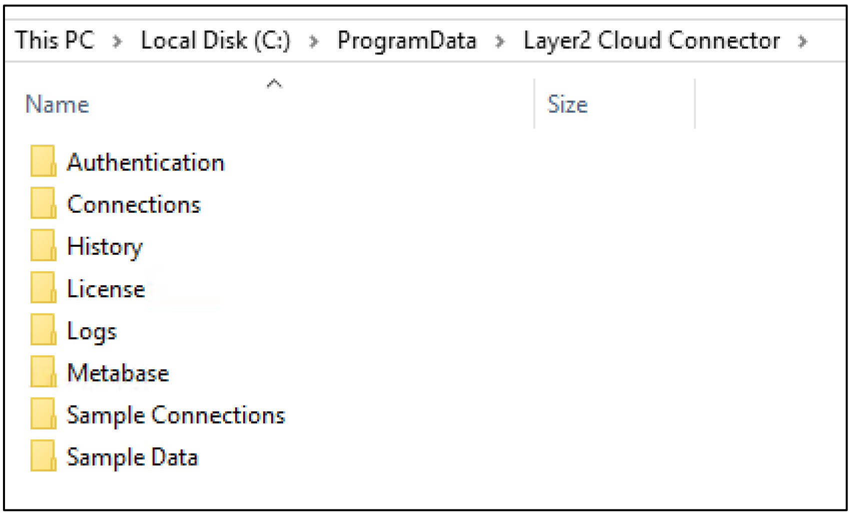
The Layer2 Cloud Connector Data Directory

The Layer2 Cloud Connector data directory is where the system stores all files that it needs to function properly except for the binary files (libraries and executable files). This includes connection definitions, Metabase-files, logs, and the license file.

It is in the Windows application data directory:

C:\ProgramData\Layer2 Cloud Connector

The path can be read from the environment variable **%PROGRAMDATA%**.



This PC > Local Disk (C:) > ProgramData > Layer2 Cloud Connector >

Name	Size
Authentication	
Connections	
History	
License	
Logs	
Metabase	
Sample Connections	
Sample Data	

Figure 22 - Items within in the Data Directory

Authentication

This is used to store definitions for custom authentication methods. See the Authentication section for more information.

Connections

The connection definitions are stored in this folder as XML files. They're named after the connection, so for example, a definition for a connection named MyConnection will be saved in a file named "MyConnection.xml". The XML format itself is be described in detail in the Connection Definition Files section.

History

The folder contains information about the last synchronizations stored as XML files for each connection. It is used for reporting in the Connection Manager.

License

The Layer2 Cloud Connector will check this directory for a file named "productkey.xml", which will be used to license the product. If no such file is found, the Layer2 Cloud Connector will run in shareware mode. See the Licensing section for details.

Logs

This folder contains the log files that will be written during synchronization. Each connection has its own log file which is named after the connection plus the .log extension. There may be four additional files: Server.log, MMC.log, Scheduler.log and NLog.config. Server.log, MMC.log, and Scheduler.log contain all activities for the part of the system they apply to. NLog.config contains the logging configuration itself. For some of the providers there also is another log file which includes the data entity name. This log file contains provider internal logs and is only relevant for support.

These files are discussed further in the Logging and Alerting section.

Metabase

This folder contains all the Metabase files of the connection. Each file is named after the connection that it is storing the Metabase for. So, if there is a connection named MyConnection there will be a corresponding Metabase file called "MyConnection.metabase". The contents of these files are in binary format, and it is not recommended to change them. See the Metabase section for more details.

Metadata [Deprecated]

This folder contains metadata that is cached by the provider **Layer2 Data Provider for OData (Deprecated)**. This folder is no longer used after 7.8.

Sample Connections

This folder contains a backup copy of the sample connections.

Sample Data

This folder contains an XML file with sample data used in Appendix A – Examples.

Configuration

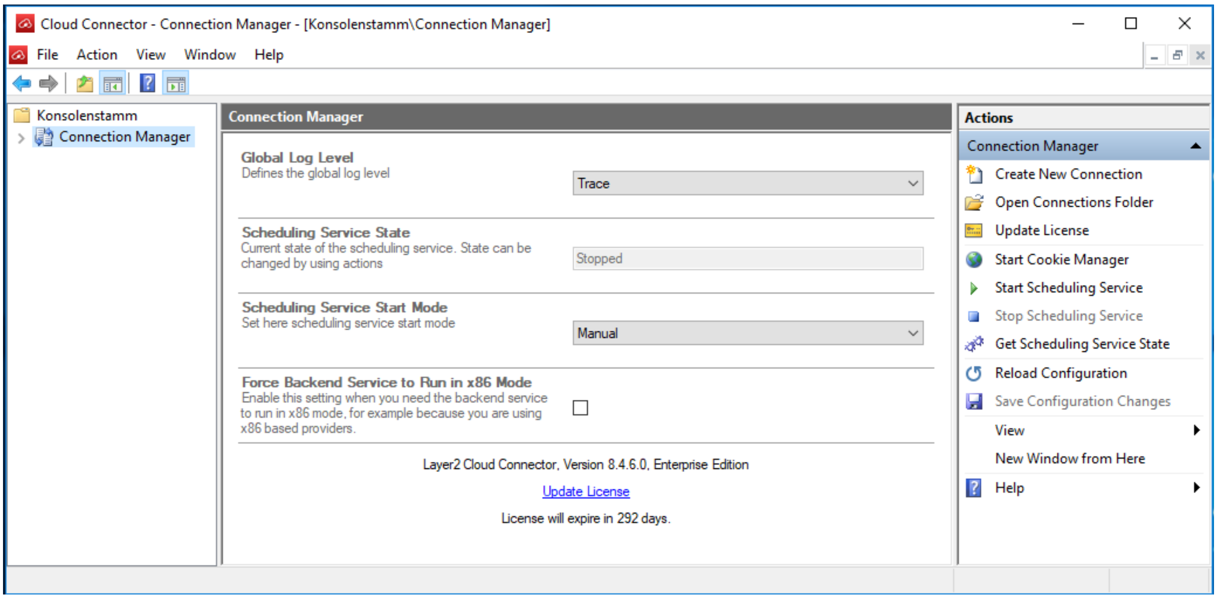
The configuration and administration of synchronization jobs for the Layer2 Cloud Connector is primarily done by using the Connection Manager. It is a snap-in for the Microsoft Management Console (MMC) and can be started by using the shortcut “Start Connection Manager” in the start menu or by adding a new snap-in to an existing MMC console configuration.

The general interface layout of the Microsoft Management Console consists of three panes:

The left-hand Connection pane provides the navigation through all the connections. For every connection, there is a node which contains sub-nodes for data entities, mapping, and logs.

The middle Properties pane is the main area and contains the configuration properties which change depending on which node has been selected in the navigation pane.

The right-hand Action pane lists all the actions that are available in the context of the currently selected navigation node. These actions are also available through the Action menu at the top of the window and in the right-click context menu of the currently selected navigation node.



Connection Manager

Global Log Level
Defines the global log level: Trace

Scheduling Service State
Current state of the scheduling service. State can be changed by using actions: Stopped

Scheduling Service Start Mode
Set here scheduling service start mode: Manual

Force Backend Service to Run in x86 Mode
Enable this setting when you need the backend service to run in x86 mode, for example because you are using x86 based providers. □

Layer2 Cloud Connector, Version 8.4.6.0, Enterprise Edition

Update License

License will expire in 292 days.

Figure 23 - View of Connection Manager properties

Connection Manager Global Settings

The root node is labeled **Connection Manager**. Activating this node in the navigation pane will show settings that impact the entire system.

Global Log Level

The setting defines the log threshold for the activity logging of all connections. The value configured here is the minimum severity that a log message must have to come through to the logging system. After changing the setting, changes must be saved using the **Save Configuration Changes** action to take effect.

There are six different severities, from most detailed to least detailed:

Trace

On this level, the most detailed log will be created, but this will also have the greatest impact on the performance of the synchronization. Messages, for example, informing about which values are compared and why they are considered equal or not will be put out to the log.

Debug

This level will not record the more detailed, technical information but it will still be verbose about what the Layer2 Cloud Connector is doing during synchronization. For example, messages about which records are modified will be written to the log.

Info

This level will restrict the log to only putting out the key data about the synchronization process, such as start of the synchronization, connection attempts to the data sources, and information about how many changes have been made.

Warn

[Default] On this level there will be warnings about suspicious system conditions.

Error

All messages on this level refer to a serious problem during the synchronization and will usually cause the synchronization to stop.

Fatal

Messages on this level will be generated when unexpected errors occur, such as hardware-originated issues. These will cause the synchronization to stop.

Setting the **Global Log Level** to a specific value will always include all severities below that. If, for example, the log level is set to “Info”, all messages with the severities “Info”, “Warning”, “Error”, and “Fatal” will get through to the log.

When the Layer2 Cloud Connector system is installed, the **Global Log Level** is by default set to “Warn”. This will provide basic issues and errors when testing and connections. It is recommended that for regular production use, the log level is set no higher “Info”. “Debug” and “Trace” should only be enabled for troubleshooting issues with the synchronization and then reverted to a lower logging level once the issue has been addressed.

Scheduling Service State

This setting shows the current state of the Layer2 Cloud Connector Scheduling Service. If there are connections scheduled for automatic background synchronization, these will only be synchronized periodically if the Scheduling Service is in the “Running” state. When the Layer2 Cloud Connector system is installed, the service is not started by default and the starting type is “Manual”. Background synchronization can be enabled by starting the Windows service using the **Start Scheduling Service** action, which is available from the right-hand Action pane.

Scheduling Service Start Mode

This setting defines what the start mode of the scheduling service is. The options are “Automatic”, “Manual” (default), and “Disabled”. To make sure that the service will be started automatically at operating system startup, the option “Automatic” needs to be chosen in the **Scheduling Service Start Mode** option box. Changes must be saved after making changes on global configuration section by clicking **Save Configuration Changes** in the right-hand Actions pane.

Force Backend Service to Run in x86 Mode

Some data providers only have a 32-bit version. To be able to use those on a x64 system (64-bit), the Backend Service can be forced to run in x86 mode (32-bit). Please note that enabling this option restarts the Backend Service and thus will abort all currently running connections. Also note that all 64-bit providers are not usable in this mode.

Note: For 32-bit ADO.NET providers, you will need to restart the Connection Manager after you enable x86 mode to have those providers available for selection in the drop-down. For ODBC and OLE DB drivers, those should function as soon as x86 mode is enabled.

Show Enterprise Edition Providers

By default, the list of available providers includes those that are part of the Enterprise edition. Without an Enterprise license, they will be limited to 25 items for trying out Enterprise scenarios. If you do not want to see those providers, you can hide them by unchecking this checkbox.

Version

This lists the current installed version of the Layer2 Cloud Connector, as well as the currently installed license type and the number of remaining days until expiration. If it is less than 14 days, the message will be shown in red.

Update License

This link will allow you to install a new license file. It is the same as the **Update License** action in the Actions pane described below.

Actions

Following actions are available while the **Connection Manager** node is selected:

Create New Connection

This creates a new, empty Connection node at the bottom on the connections list in the left-hand pane.

Open Connections Folder

Opens a Windows Explorer window for the connection configuration XML files. This is a quick way to access the configuration files in case a manual update needs to be done. See The Layer2 Cloud Connector Data Directory section for more details.

Update License

Opens a Windows Explorer window to locate a productkey.xml file to update the current license state. This option is also available from the central pane of the global settings. See the Licensing section for more details.

Start Cookie Manager

Starts the Cookie Manager application. See the Cookie Manager section for more details. [Deprecated]

Start Scheduling Service

Starts the Scheduling synchronization service. See the Scheduling Service section for more details.

Stop Scheduling Service

Stops the Scheduling synchronization service.

Get Scheduling Service State

See Scheduling Service State above.

Reload Configuration

This forces the Layer2 Cloud Connector to reload all connection files. This is useful if the connection configuration XML files are changed outside of the Connection Manager so that the changes can be loaded into the client. This may also refresh some UI elements.

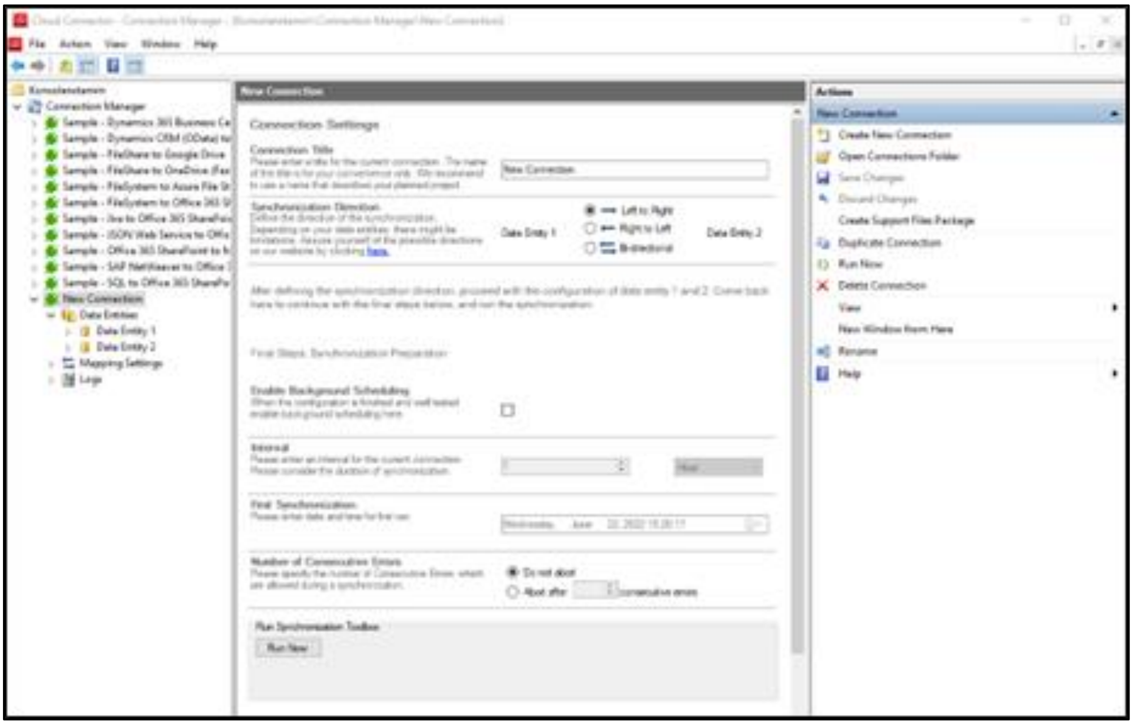
Save Configuration Changes

Saves all changes made to the global settings.

Connection Definition

Below the root node there is one sub-node for every connection. Layer2 Cloud Connector ships with a set of sample connections as examples to help the user get started in creating their own connections. These sample connections are disabled for background synchronization.

For the selected connection, the Connection Manager will display general configuration options in the central panel.



File Action View Window Help
Konsolenstamm
Connection Manager
Sample - Dynamics 365 Business Ce
Sample - Dynamics CRM (OData) to
Sample - FileShare to Google Drive
Sample - FileShare to OneDrive (Fa
Sample - FileSystem to Azure File St
Sample - FileSystem to Office 365 S
Sample - Jira to Office 365 SharePoi
Sample - JSON Web Service to Offic
Sample - Office 365 SharePoint to Is
Sample - SAP NetWeaver to Office 3
Sample - SQL to Office 365 SharePo
New Connection
Data Entities
Data Entity 1
Data Entity 2
Mapping Settings
Logs
Actions
New Connection
Create New Connection
Open Connections Folder
Save Changes
Discard Changes
Create Support Files Package
Duplicate Connection
Run Now
Delete Connection
View
New Window from Here
Rename
Help

New Connection

Connection Settings

Connection Title: New Connection

Synchronization Direction: Data Entity 1 ● Left to Right ○ Right to Left ○ Bi-directional Data Entity 2

Final Steps: Synchronization Preparation

Enable Background Scheduling □

Interval: Hour

First Synchronization: Wednesday, June 22, 2022 15:20:17

Number of Consecutive Errors: ● Do not abort ○ Abort after consecutive errors

Run Synchronization Toolbox: Run Now

Figure 24 - View of Connection properties

Connection Title

This setting is for specifying a title for the connection. Each connection must have a unique name.

Important – Be careful when using special characters. Period "." Is not supported. Brackets and parenthesis must be closed.

Direction

The setting defines the direction in which the synchronization will be processed. Uni-directional synchronizations can be defined as **Left to Right** or **Right to Left**, depending on which data entity should be the source and which the target. Bi-directional synchronizations will synchronize changes from both data entities to each other.

Scheduling Enabled

If this option is enabled, the connection will be included in the automatic background synchronization performed by the Scheduling Service. The background synchronization will synchronize the connection without needing to start the Connection Manager. This requires that the Scheduling Service is running.

Interval

If background synchronization is enabled for the connection, this interval will define how often a synchronization run will be executed. This can range from a few minutes to multiple days.

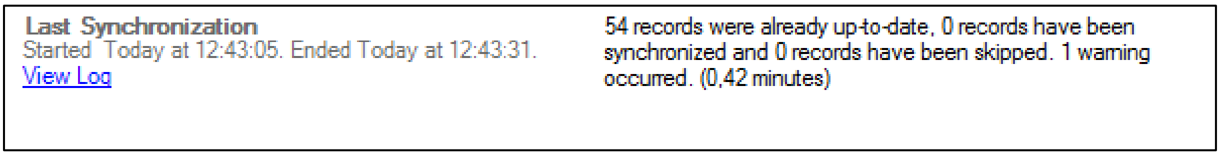
First Synchronization

This defines the exact date and time when the automatic background synchronization for this connection will run for the first time (or, if it has been running before, for the next time). This is based on the local machine's time.

For example, if the connection should be configured to run every night, **Interval** would be set to 1 day whereas the next run will be set to 12:00 am. This way the synchronization will be started every night at 12:00 am.

Last Synchronization

If a connection has already been executed at least once, a summary of the last synchronization run will be displayed just below the **Number of Consecutive Errors** section. Synchronization runs performed by the Scheduling Service will be displayed here as well.



Last Synchronization
Started Today at 12:43:05. Ended Today at 12:43:31.
View Log

54 records were already up-to-date, 0 records have been synchronized and 0 records have been skipped. 1 warning occurred. (0,42 minutes)

Figure 25 – Example of a Last Synchronization summary

Number of Consecutive Errors

This defines the number of errors that occur consecutively, before the synchronization is aborted. Options are **Do Not Abort** and **Abort After** which includes user defined field. By default, it is set to abort after 10 consecutive errors.

Run Synchronization Toolbox (Run Now)

This action starts the synchronization of the currently selected connection. It can be used to test the connection or to perform one-time replications, such as migration tasks. This toolbox window also shows the current state of the sync, like total items updated, as well as fatal errors.

Abort

This action button only appears here while a sync is actively running. Clicking this button will cause the sync to stop where it is at and record any changes it had made at that point to the Metabase.

Actions

Create New Connection

This action creates a new empty connection under the root node.

Delete Connection

This will delete the currently selected connection entirely. All associated data (connection file, history, logs) will be deleted.

Save Changes

As soon as a setting of the connection is changed, it will be active to allow saving the changes.

Discard Changes

Will discard any changes not yet saved and revert the information to the last saved state.

Note, if there are unsaved changes on the current connection and another node is selected in the navigation pane, the Connection Manager will show a dialog demanding to save or discard the changes.

Create Support Files Package

This will ask for a location and create a ZIP archive with all support-related files (connection file, logs, and history file) associated with the current connection. Sensitive information (username/password) is automatically masked.

Duplicate Connection

With this action, the connection can be duplicated to use it as a template for a new connection.

Run Now

This action starts the synchronization of the currently selected connection. It can be used to test the connection or to perform one-time replications, such as migration tasks.

Delete the Connection

This will delete the currently selected connection entirely. All associated data (connection file, history, logs, Metabase) will be deleted.

Rename

With this action, the connection will be renamed as will with all related files like log, history, etc.

Data Entity

Both data sources that are participating in the synchronization, as source or target, will be referred to as a data entity. Every connection contains two sub-nodes for these data entities, which can be configured to access various data sources.

Data Entity Title

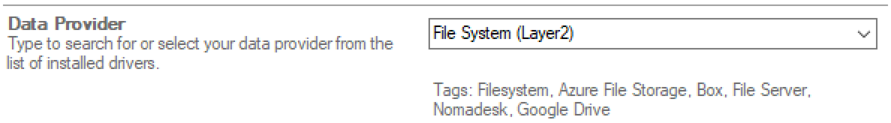
This is the name of the data entity used to identify the entity in the navigation pane and in log messages.

Entity Type

This is showing the role that this entity owns in the synchronization process and can be **Target**, **Source**, or **Bi-directional**. This value cannot directly be changed in the data entity screen and is defined by specifying a value for the synchronization direction in the main connection properties.

Data Provider

Provides a selection of the ADO.NET providers which will be used to retrieve data and write changes for the current data entity. All installed providers that are compatible with the current processor architecture mode will be shown in the drop-down.



Data Provider
Type to search for or select your data provider from the list of installed drivers.

File System (Layer2)

Tags: Filesystem, Azure File Storage, Box, File Server, Nomadesk, Google Drive

Figure 26 - Example of selected provider showing the associated tags

You can just type into the provider drop-down box to search for a provider. Providers can be found by name as well as by tag. The tags assigned to the currently selected provider are shown below the drop-down.

By default, the Layer2 Cloud Connector starts in 64-bit mode if installed on a 64-bit machine. If you need to use 32-bit providers, you need to switch to x86 mode described in the Force Backend Service to Run in x86 Mode section.

To see the pre-installed providers provided with Layer2 Cloud Connector Personal and Standard, go to the Layer2 Data Providers section. For additional 3rd-party data providers, one can usually find downloads for these on the 3rd-party's website. They usually come as an executable setup file which can be run and then the provider (will appear in the list of options here. If you have the Layer2 Cloud Connector open while installing a new provider, you must restart it before you can see the new provider in the list. If the provider does not appear in the drop-down after installation, please make sure the correct mode is set and then close and reopen the application. The provider should then appear.

Note: With some 3rd-party providers you may need to reboot the host machine before they work properly. This is not required for the included Layer2 data providers and for most 3rd-party providers.

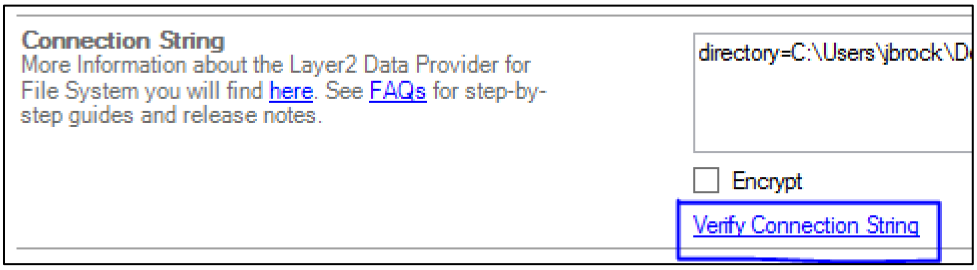
Connection String

This is the ADO connection string for the selected provider. These connection strings consist of key-value pairs separated by semicolons.

Example `Url=http://MySharePointServer/MySharePointSite/; List=Links; Authentication=Windows; User Id=MyDomain\MyUserName;`

Connection strings are highly specific to the provider that is used. There are many examples and documentation at www.connectionstrings.com and on the Layer2 Solutions page. The **Encrypt** option can be used to hide security relevant information in the connection files.

There is a **Verify Connection String** link below the connection string text box that is useful to quickly evaluate if the provided connection string is valid. This has the same behavior as the Verify Connection String action listed below.



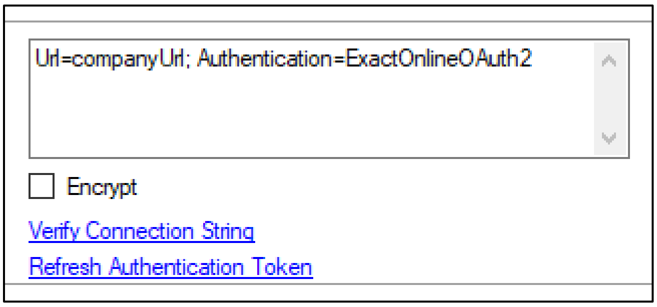
Connection String
More Information about the Layer2 Data Provider for File System you will find here. See FAQs for step-by-step guides and release notes.

directory=C:\Users\jbrock\D

□ Encrypt

Figure 27 – "Verify Connection String" action highlighted in the connection string details

Some authentications require user interaction. For those, a new action will be displayed.



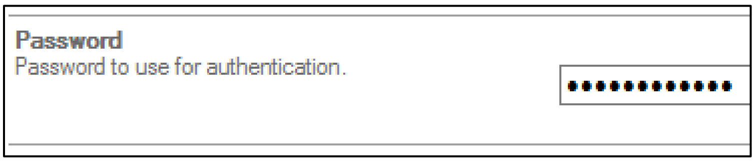
Url=companyUrl; Authentication=ExactOnlineOAuth2

□ Encrypt

Figure 28 – "Refresh Authentication Token" action visible due to using an authentication that requires user interaction

Password

Here you can enter the password connection string parameter, if required by the data provider. This field masks the value for better security.



Password
Password to use for authentication.

●●●●●●●●●●●●●

Figure 29 – Password field with masked value

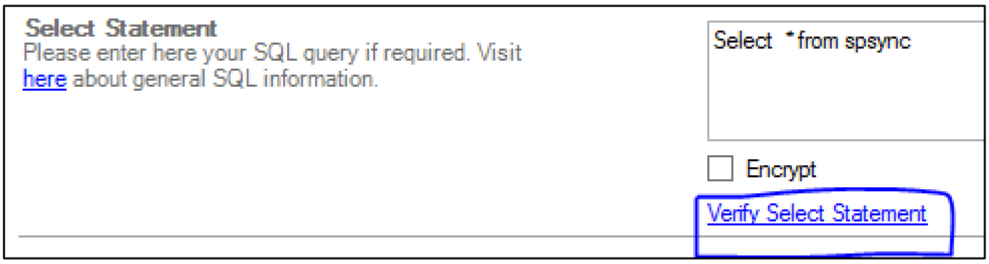
You can still use the connection string to enter your password to prevent typos and specify the password parameter name (Pwd, Pass, Password, etc.), but the next time the Layer2 Cloud Connector is started, the password is automatically moved into the masked password field.

Note: If both fields contain a password and they do not match, an error will be shown.

Select Statement

In this text box a data query statement can be defined. The format to be used for the query is specific to the provider selected (see the specific provider documentation to find out what is required) and is not necessary for all providers. If the provider is known by the Connection Manager to not support a select statement, the **Select Statement** setting will not be available. The **Encrypt** option can be used to hide security relevant information in the connection files.

The select statement can be validated like the connection string with the **Verify Select Statement** link. This has the same behavior as the Verify Select Statement action listed below.



Select Statement
Please enter here your SQL query if required. Visit here about general SQL information.

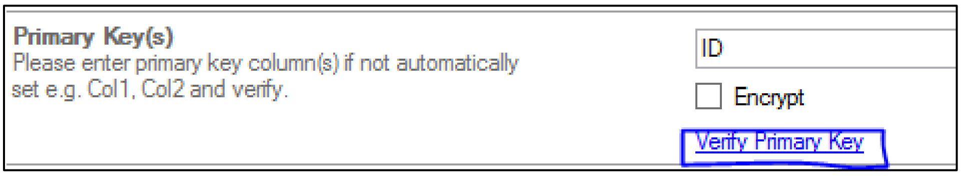
Select * from spsync

Figure 30 - "Verify Select Statement" action highlighted in the select statement details

Primary Key(s)

This setting is optional. If the data source provides a primary key, there is no need to define one explicitly. If this is not the case, the key can be defined here. Any field that is provided by the data source can be defined as a primary key. Furthermore, multiple fields are defined, separated by a comma, to define a composite primary key. The **Encrypt** option can be used to hide security relevant information in the connection files.

The primary key can be validated like the connection string with the **Verify Primary Key** link. This has the same behavior as the Verify Primary Key action listed below.



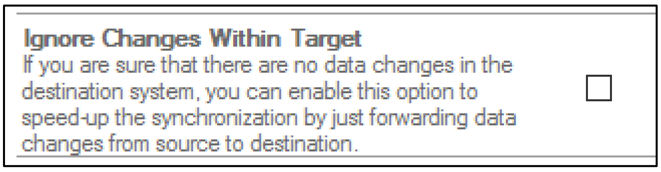
Primary Key(s)
Please enter primary key column(s) if not automatically set e.g. Col1, Col2 and verify.

ID

Figure 31 - "Verify Primary Key" action highlighted in the primary key details

Ignore Changes Within Target

This setting is only available for the target data entity in a uni-directional synchronization and is optional. If enabled, it speeds up the uni-directional synchronization because the Layer2 Cloud Connector does not read the data from the target for the data comparisons but will rely on the data in the Metabase. So, for example, a field value is changed in the target only, it will not be overwritten with the source value (as it normally would), because this change is not recognized. Therefore, use this option with care and only if you are sure that there are no changes made at all in the target system.

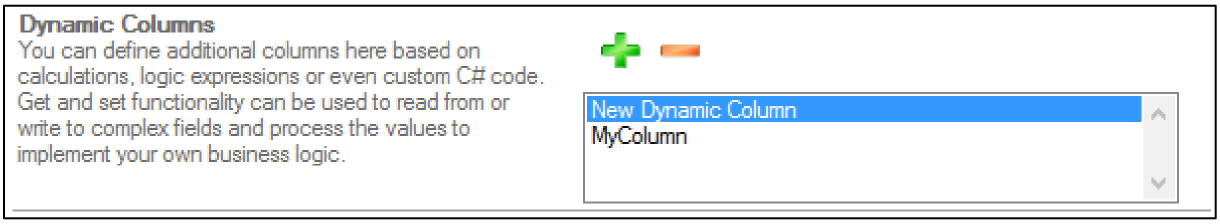


Ignore Changes Within Target
If you are sure that there are no data changes in the destination system, you can enable this option to speed-up the synchronization by just forwarding data changes from source to destination. □

Figure 32 - Options to Ignore Changes Within Target

Dynamic Columns

This setting in the **Advanced Settings** section allows users to create new custom fields, which are added to the fields from the data source. For complete details on Dynamic Columns, see the [Dynamic Columns]() section.



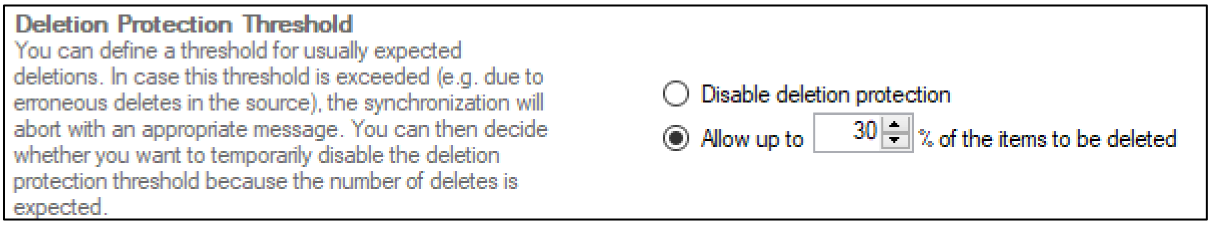
Dynamic Columns
You can define additional columns here based on calculations, logic expressions or even custom C# code. Get and set functionality can be used to read from or write to complex fields and process the values to implement your own business logic.

New Dynamic Column
MyColumn

Figure 33 - "Dynamic Columns" list in the data entity settings

Deletion Protection Threshold

This optional setting in the **Advanced Settings** section will prevent accidental data loss due to unforeseen circumstances such as hardware failure. It is only visible for target data entities.



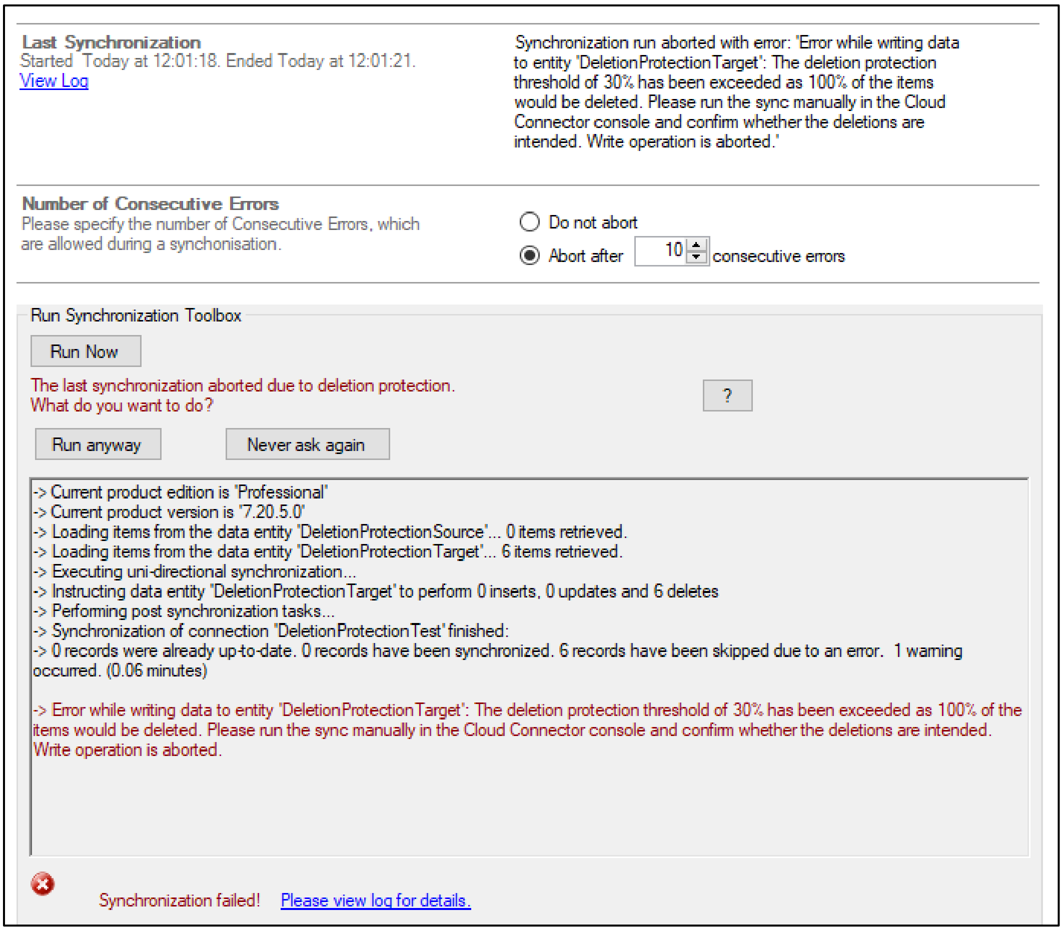
Deletion Protection Threshold
You can define a threshold for usually expected deletions. In case this threshold is exceeded (e.g. due to erroneous deletes in the source), the synchronization will abort with an appropriate message. You can then decide whether you want to temporarily disable the deletion protection threshold because the number of deletes is expected.

○ Disable deletion protection
● Allow up to 30 % of the items to be deleted

Figure 34 - "Deletion Protection Threshold" in the data entity settings

When a synchronization hits the threshold, it aborts with an error. Then you can solve this in multiple ways:

- If this was an unexpected but temporary issue, just wait for the next scheduled sync or run the connection again manually
- If this was expected (e.g., you deleted all files in the source), you can disable this once by clicking on **Run anyway** in the connection overview
- If you don't want to use this feature, either disable it in the data entity settings or click on **Never ask again** in the connection overview



Last Synchronization
Started Today at 12:01:18. Ended Today at 12:01:21.
View Log

Synchronization run aborted with error: 'Error while writing data to entity 'DeletionProtectionTarget': The deletion protection threshold of 30% has been exceeded as 100% of the items would be deleted. Please run the sync manually in the Cloud Connector console and confirm whether the deletions are intended. Write operation is aborted.'

Number of Consecutive Errors
Please specify the number of Consecutive Errors, which are allowed during a synchronisation.

○ Do not abort
● Abort after 10 consecutive errors

Run Synchronization Toolbox

The last synchronization aborted due to deletion protection.
What do you want to do?

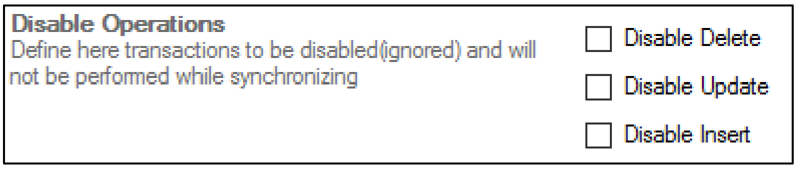
```
-> Current product edition is 'Professional'
-> Current product version is '7.20.5.0'
-> Loading items from the data entity 'DeletionProtectionSource'... 0 items retrieved.
-> Loading items from the data entity 'DeletionProtectionTarget'... 6 items retrieved.
-> Executing uni-directional synchronization...
-> Instructing data entity 'DeletionProtectionTarget' to perform 0 inserts, 0 updates and 6 deletes
-> Performing post synchronization tasks...
-> Synchronization of connection 'DeletionProtectionTest' finished:
-> 0 records were already up-to-date. 0 records have been synchronized. 6 records have been skipped due to an error.  1 warning
occurred. (0.06 minutes)

-> Error while writing data to entity 'DeletionProtectionTarget': The deletion protection threshold of 30% has been exceeded as 100% of the
items would be deleted. Please run the sync manually in the Cloud Connector console and confirm whether the deletions are intended.
Write operation is aborted.
```

Synchronization failed! Please view log for details.

Figure 35 – Connection overview after a synchronization was aborted due to deletion protection

Disable Operations

This setting in the **Advanced Settings** section allows users to define which operations/transactions are omitted during execution of a synchronization. For example, when **Disable Delete** is selected, all delete requests to this entity during synchronization will be omitted. Possible values are Insert, Update and Delete. Individual or multiple selections are possible. The setting is optional and only available on entities that are being written to.



Disable Operations
Define here transactions to be disabled(ignored) and will not be performed while synchronizing

☐ Disable Delete
☐ Disable Update
☐ Disable Insert

Figure 36 - Options to disable certain operations form affecting the data entity

Actions

Create New Connection

This action creates a new empty connection under the root node.

Open Connections Folder

Opens a Windows Explorer window for the connection configuration XML files. A quick way to access the configuration files in case a manual update needs to be done. See The Layer2 Cloud Connector Data Directory.

Save Changes

As soon as a setting of the data entity is changed, it will be active to allow saving the changes.

Discard Changes

Will discard any changes not yet saved and revert the information to the last saved state.

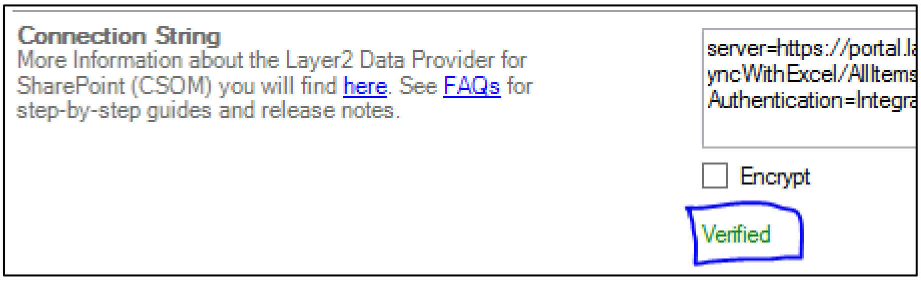
Note, if there are unsaved changes on the current connection and another node is selected in the navigation pane, the Connection Manager will show a dialog demanding to save or discard the changes.

Preview Data

This action is used when the data entity has been set up and there is a need to check if all the necessary fields will come back as expected. It will show a simple view of all the columns selected by the query for the first ten records by default. It is possible, however, to load more items using the "Load more" link. The preview is also helpful to identify a primary key, if required.

Verify Connection String

The Connection Manager will try to create a connection using the specified connection string and will show any errors that occur while doing that. If no errors occur, it will show a green "Verified" message.



Connection String
More Information about the Layer2 Data Provider for SharePoint (CSOM) you will find here. See FAQs for step-by-step guides and release notes.

server=https://portal.la
yncWithExcel/AllItems
Authentication=Integra

☐ Encrypt

Verified

Figure 37 - Example of verifying a connection string

Verify Select Statement

Like the **Verify Connection String** action, the Connection Manager will try to create a connection using the specified connection string and select statement and shows any errors that occur while doing that. If no errors occur, it will show a green “Verified” message.

Verify Primary Key

Like the **Verify Connection String** action, the Connection Manager will try to create a connection using the specified connection string and primary key and shows any errors that occur while doing that. If no errors occur, it will show a green “Verified” message.

Rename

With this action, the data entity name will be editable in the tree view to the left-hand side.

Mapping

The mapping defines which fields will be compared during the synchronization. It is based on field names. In case the data entity provides different values for an internal field name and the related display name (for example, in a SharePoint list), both are shown to make it easier to identify the correct fields to be mapped. The internal name is shown first, with the display name shown in single quotes after it, followed by the data type in parenthesis.

Example:

Author ‘Created By’ (String)

Enable Auto Mapping

By checking **Enable Auto Mapping**, fields with the same name will be mapped to each other. Fields of the first data entity are shown on the left side, fields of the second data entity are shown on the right side.

You can also do manual mapping. For that, additional entries can be added by clicking the **green plus** button and existing ones can be deleted by clicking **red minus** button. If any of the data entities are not valid (e.g., invalid connection string), the mapping screen will show the error message which occurred during the connection attempt.

Conflict Resolution Mode

This option only appears when the connection is set as bi-directional. It is the method that will be used to resolve synchronization conflicts. The options are “FailAndAbort”, “WarnAndContinue”, “WinnerLoser” and only available for file synchronizations “KeepBoth. The methods and behavior are described in greater detail in the Conflict Resolution section.

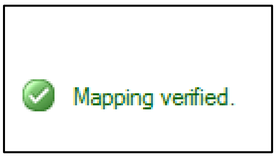
By default, the option is set to “FailAndAbort”.

Master Entity/Winning Entity

This option only appears when either “WinnerLoser” or “KeepBoth” are selected as the Conflict Resolution Mode. The options are the two entities, so you can define which is the Master/Winning data source.

Verify Mapping

Using the **Verify Mapping** action or link will have compared fields checked to make sure their types match and that there are no other issues (such as read-only fields that might be written to). If no errors occur, it will show a green “Mapping verified” message.



✓ Mapping verified.

Figure 38 - Confirmation shown for verified mapping

Reload Mapping

Using the **Reload Mapping** action or link with reload all the available fields from the data entities as well as the names/types of currently mapped items.

Run Now

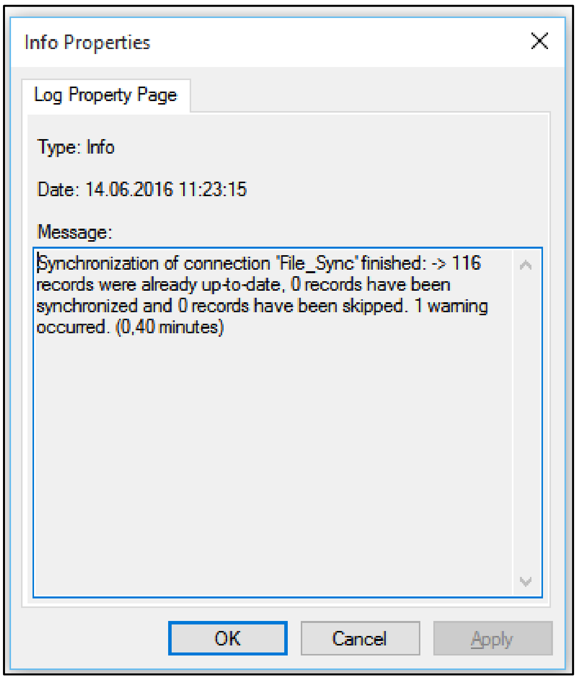
This action starts the synchronization of the currently selected connection. It can be used to test the connection or to perform one-time replications, such as migration tasks.

Logs

This shows the most recently logged messages for the selected connection. You can double-click on any entry in the log to see the full details in a dialog box. Note that the Log UI shows the items from newest to oldest, with the newest at the top of the pane. Also, be aware that the Log UI does not show all logged items from a sync, nor some detailed information (such as stack traces). You will need to get the log file which will have the full record, which can be done via the Open Log File action mentioned below.

By default, the logging level is set to “Warn”, which only shows warnings and fatal errors. If you want to increase the detail of the logging, please see the Global Log Level section for how to adjust that.

For more information about logging and configuring other types of alerts (like email), see the Logging and Alerting section.



Type: Info

Date: 14.06.2016 11:23:15

Message:

```
Synchronization of connection 'File_Sync' finished: -> 116 records were already up-to-date, 0 records have been synchronized and 0 records have been skipped. 1 warning occurred. (0,40 minutes)
```

Figure 39 - Example of full text for a log entry

Open Log File

Opens the full log file for the current connection in the default program for files with the ".log"-extension.

Open Logs Folder

Opens a Windows Explorer window for the Logs folder. This is a quick way to access all log files for a connection (if some have been set as archived). See The Layer2 Cloud Connector Data Directory for more information.

Delete Logs

Deletes the log file for the current connection.

View

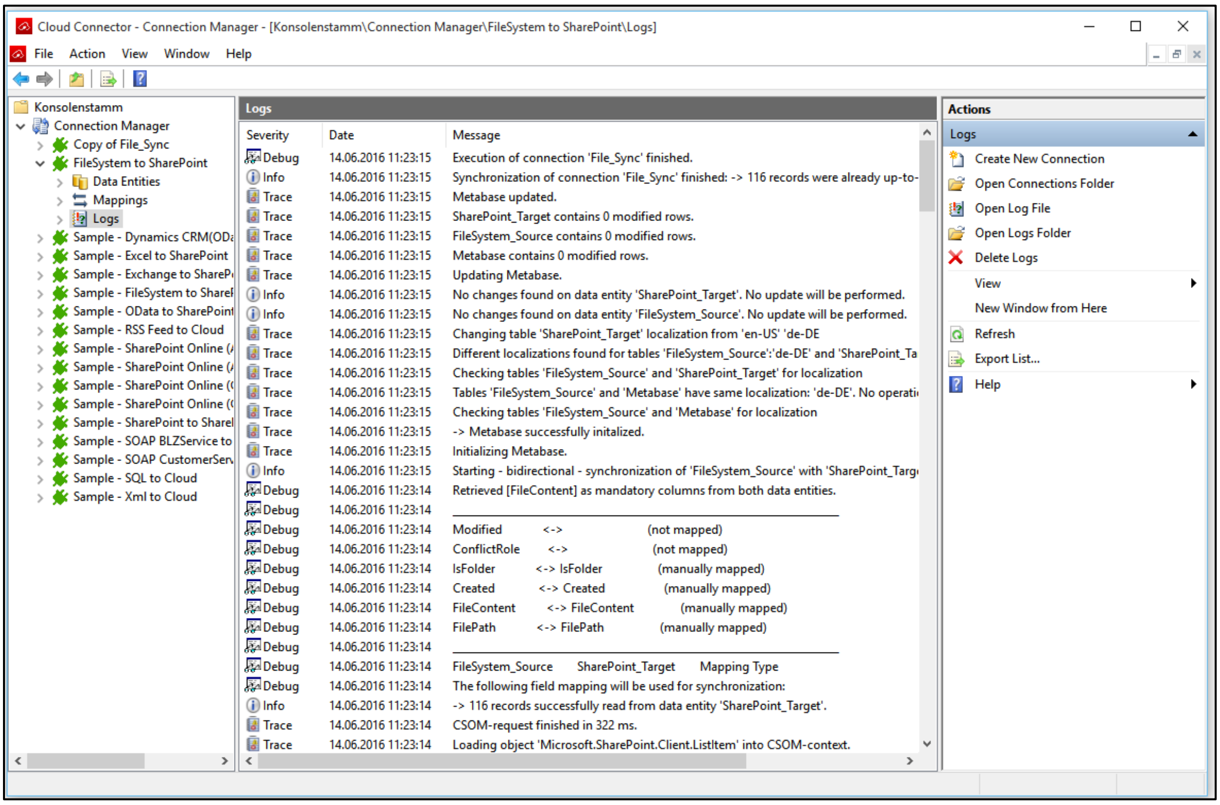
Filter the logs by selecting **View** in the right action pane and choose the required filter option.

Refresh

Refreshes the data in the logs UI to the most current information from the log file.

Export List

Allows you to export the current information in the logs UI to a .txt file. Note that this is **not** an export of the full log file, but of the limited data shown in the Logs UI central pane. Use **Open Log File** instead to get the full log details.


Cloud Connector - Connection Manager - [Konsolenstamm\Connection Manager\FileSystem to SharePoint\Logs]
File
Action
View
Window
Help
Konsolenstamm
Connection Manager
Copy of File_Sync
FileSystem to SharePoint
Data Entities
Mappings
Logs
Sample - Dynamics CRM(OD
Sample - Excel to SharePoint
Sample - Exchange to ShareP
Sample - FileSystem to Share
Sample - OData to SharePoint
Sample - RSS Feed to Cloud
Sample - SharePoint Online (
Sample - SharePoint Online (
Sample - SharePoint Online (
Sample - SharePoint Online (
Sample - SharePoint to Share
Sample - SOAP BLZService to
Sample - SOAP CustomerServ
Sample - SQL to Cloud
Sample - Xml to Cloud
Logs
Actions
Logs
Create New Connection
Open Connections Folder
Open Log File
Open Logs Folder
Delete Logs
View
New Window from Here
Refresh
Export List...
Help

```
Severity   Date                 Message
Debug      14.06.2016 11:23:15  Execution of connection 'File_Sync' finished.
Info       14.06.2016 11:23:15  Synchronization of connection 'File_Sync' finished: -> 116 records were already up-to-
Trace      14.06.2016 11:23:15  Metabase updated.
Trace      14.06.2016 11:23:15  SharePoint_Target contains 0 modified rows.
Trace      14.06.2016 11:23:15  FileSystem_Source contains 0 modified rows.
Trace      14.06.2016 11:23:15  Metabase contains 0 modified rows.
Trace      14.06.2016 11:23:15  Updating Metabase.
Info       14.06.2016 11:23:15  No changes found on data entity 'SharePoint_Target'. No update will be performed.
Info       14.06.2016 11:23:15  No changes found on data entity 'FileSystem_Source'. No update will be performed.
Trace      14.06.2016 11:23:15  Changing table 'SharePoint_Target' localization from 'en-US' 'de-DE
Trace      14.06.2016 11:23:15  Different localizations found for tables 'FileSystem_Source':'de-DE' and 'SharePoint_Ta
Trace      14.06.2016 11:23:15  Checking tables 'FileSystem_Source' and 'SharePoint_Target' for localization
Trace      14.06.2016 11:23:15  Tables 'FileSystem_Source' and 'Metabase' have same localization: 'de-DE'. No operati
Trace      14.06.2016 11:23:15  Checking tables 'FileSystem_Source' and 'Metabase' for localization
Trace      14.06.2016 11:23:15  -> Metabase successfully initalized.
Trace      14.06.2016 11:23:15  Initializing Metabase.
Info       14.06.2016 11:23:15  Starting - bidirectional - synchronization of 'FileSystem_Source' with 'SharePoint_Targ
Debug      14.06.2016 11:23:14  Retrieved [FileContent] as mandatory columns from both data entities.
Debug      14.06.2016 11:23:14  ____________________________________________________________
Debug      14.06.2016 11:23:14  Modified        <->                    (not mapped)
Debug      14.06.2016 11:23:14  ConflictRole      <->                    (not mapped)
Debug      14.06.2016 11:23:14  IsFolder        <-> IsFolder            (manually mapped)
Debug      14.06.2016 11:23:14  Created         <-> Created             (manually mapped)
Debug      14.06.2016 11:23:14  FileContent       <-> FileContent           (manually mapped)
Debug      14.06.2016 11:23:14  FilePath        <-> FilePath            (manually mapped)
Debug      14.06.2016 11:23:14  ____________________________________________________________
Debug      14.06.2016 11:23:14  FileSystem_Source     SharePoint_Target     Mapping Type
Debug      14.06.2016 11:23:14  The following field mapping will be used for synchronization:
Info       14.06.2016 11:23:14  -> 116 records successfully read from data entity 'SharePoint_Target'.
Trace      14.06.2016 11:23:14  CSOM-request finished in 322 ms.
Trace      14.06.2016 11:23:14  Loading object 'Microsoft.SharePoint.Client.ListItem' into CSOM-context.
```

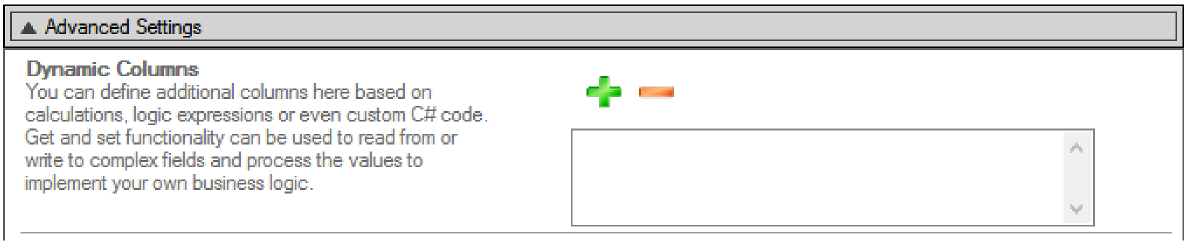
Figure 40 - Example of Logs properties

Dynamic Columns

This feature allows users to define new data columns in addition to the ones that are returned by the data entity. The values of these columns are calculated at run-time and can be based on the content of the other columns originating from the data entity. This allows for customized conversions, translations, and formatting of the data to be synchronized.

The definition of Dynamic Columns is based on the programming language C# in version 6. Either a valid C# expression or a valid method-body returning a value on all possible code-paths are acceptable. The code used includes common parts of the .NET Class-Library, but currently does not allow loading additional assemblies or namespaces. It is, however, possible to load additional assemblies through reflection.

The Dynamic Column feature is under the **Advanced Settings** for each Data Entity.



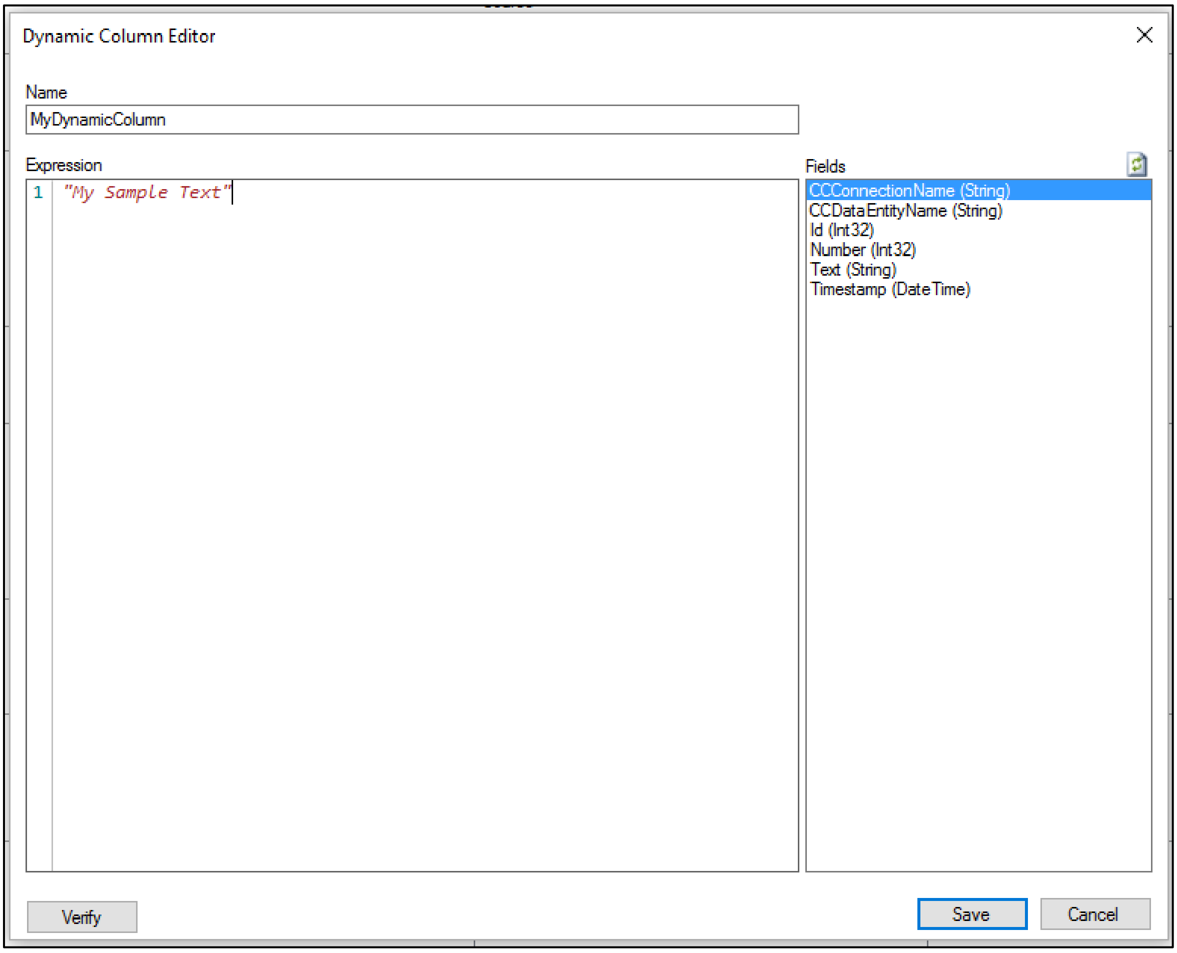
Advanced Settings

Dynamic Columns

You can define additional columns here based on calculations, logic expressions or even custom C# code. Get and set functionality can be used to read from or write to complex fields and process the values to implement your own business logic.

Figure 41 - Dynamic Column location in UI

Click the red '+' to create a new Dynamic Column or double-clicking an existing one will open an editor for the column, where the name and code of the dynamic column can be edited.



Dynamic Column Editor

Name: MyDynamicColumn

Expression

```
"My Sample Text"
```

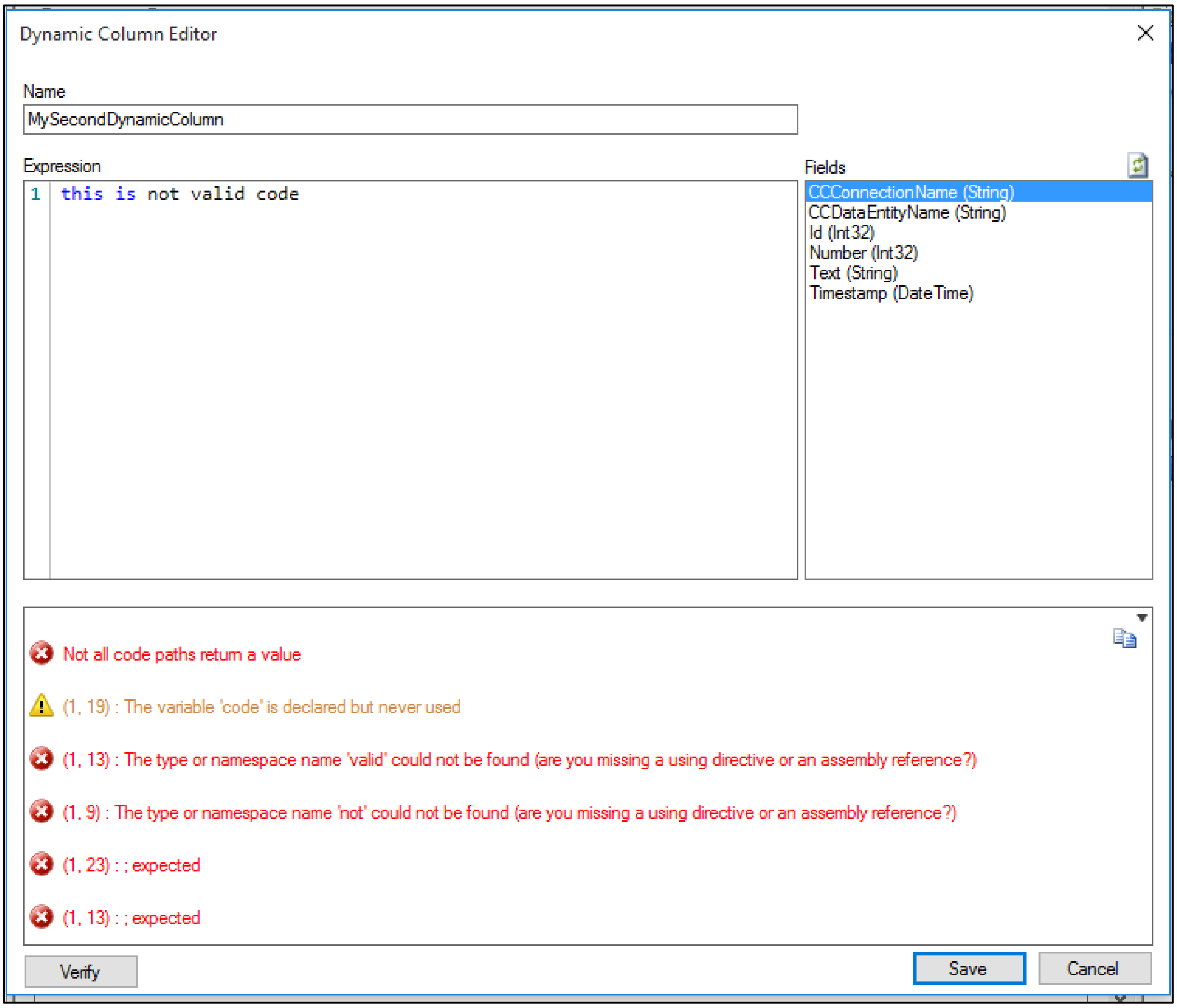
Fields

- CCConnectionName (String)
- CCDataEntityName (String)
- Id (Int32)
- Number (Int32)
- Text (String)
- Timestamp (DateTime)

Figure 42 – The Dynamic Columns editor

A list of fields provided by the data entity is displayed on the right that can be used to add references to the fields in the Dynamic Column code by either drag-and-drop or double-clicking the entry. In the case that fields have been changed while the dialog was open, the list can be refreshed by clicking the refresh icon above it.

If the code is an expression that can be resolved into a value, no return statement is necessary. The data type of the column is automatically detected by the returned values. All returns need to have the same data type, or the verification will report an error.



MySecondDynamicColumn

```
this is not valid code
```

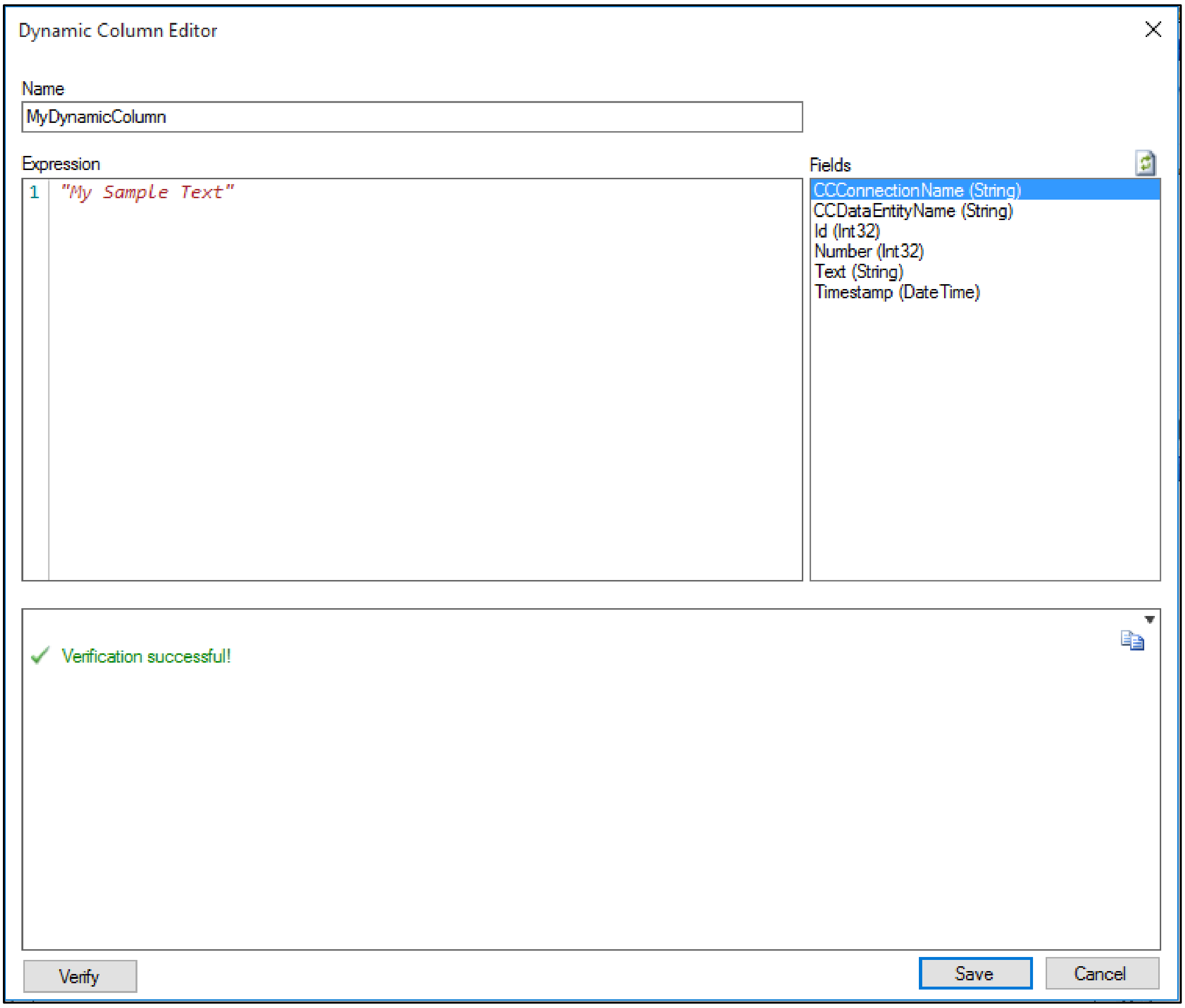
CCConnectionName (String)
CCDataEntityName (String)
Id (Int32)
Number (Int32)
Text (String)
Timestamp (DateTime)

```
Not all code paths return a value
(1, 19) : The variable 'code' is declared but never used
(1, 13) : The type or namespace name 'valid' could not be found (are you missing a using directive or an assembly reference?)
(1, 9) : The type or namespace name 'not' could not be found (are you missing a using directive or an assembly reference?)
(1, 23) : ; expected
(1, 13) : ; expected
```

Figure 43 – A failed verification due to invalid code

Red circles with a white X are errors that need to be fixed before the Dynamic Column can be used. Yellow triangles with a black exclamation mark are warnings which do not necessarily have to be fixed but should not be ignored in most cases as they usually note an oversight or mistake.

With the copy symbol at the upper right of the verification window, you can copy the error messages and some additional information to the clipboard.



Dynamic Column Editor

Name

MyDynamicColumn

Expression

```
"My Sample Text"
```

Fields

- CCConnectionName (String)
- CCDataEntityName (String)
- Id (Int32)
- Number (Int32)
- Text (String)
- Timestamp (DateTime)

✓ Verification successful!

Figure 44 – The successful verification of a very simple Dynamic Column

Note: If you change the data type of the Dynamic Column after it has been mapped in the **Mappings** node, make sure that you reload and verify the mapping. This helps to avoid errors due to mismatching column types in the mapping.

Note: By default, Dynamic Columns are read-only. To use them with bi-directional syncs you need to define get{} and set{} parameters in the Dynamic Column code. See the Lookup Translation section below for an example.

Code Examples for Dynamic Columns

This section describes various possible uses for Dynamic Columns with example code.

Conditional (True or False)

This column will return `true` or `false` as boolean values based on a user-defined condition, such as "The field 'FilePath' starts with '/myFolder'".

Note: Please be careful with StartsWith, as in this case, files under "/myFolder2" would also be true.

Fields	`FilePath = "/myFolder/myFile.txt"`
Code	`FilePath.StartsWith("/myFolder")`
Result	`true : bool`

Field Combination

This column will return a combination of other fields based on the user-defined format. The data type will be `string`.

Fields	`MyFirstField  = "Test1"`
	`MySecondField = 125`
	`MyThirdField  = false`

Code	`string.Format("First: {0}, Second {1}, Third: {2}", MyFirstField, MySecondField, MyThirdField)`
Result	`"First: Test1, Second 125, Third: false" : string`

Current Date and Time

This example shows how to provide the target system with the current date and time of the host system.

Fields	–
Code	`DateTime.Now`
Result	`(Current Date and Time) : DateTime`

Field Transformation

This example shows a simple number conversion for a target system that cannot handle the source system value without adjustments.

Fields	YearlyRevenue = 1500000
Code	YearlyRevenue / 12
Result	125000 : float

Conditional Field Selection

In this example, the source system contains products with a regular price, a discount price, and a flag that defines if the product should use the discount. The target system should only receive a price and no information about discounts.

Fields	RegularPrice = 20.00 DiscountPrice = 16.99 UseDiscount = true

Code	if (UseDiscount)
	{
	  return DiscountPrice;
	}
	return RegularPrice;

Result	16.99 : float

Note: If you use a string field to hold boolean values (in SQL for example), you need to parse the string value into a boolean variable first, like:

```
bool flag;
bool.TryParse(UseDiscount, out flag);
```

Then use the flag in the if-statement instead of UseDiscount.

Lookup Translation

In the case of a migration sync from one SharePoint to another, a source list with a lookup column must be sent to the new list in the new SharePoint. There is a similar lookup set up regarding the labels, but the IDs do not match those of the old SharePoint, leading to data issues. With a simple dynamic column that only provides and sets the lookup label without the ID, this example can solve that problem. The dynamic column must be mapped to the Lookup column of the other data entity.

Note: This sample does not work if your lookup values contain the substring ";#" as this is a special character combination that SharePoint uses internally for the lookup field representation.

<table>
<tr><th>Fields</th><td><pre><code>LookupColumn = "3;#Value"</code></pre></td></tr>
<tr><th>Code</th><td><pre><code>get
{
  if (string.IsNullOrEmpty(LookupColumn))
  {
    return string.Empty;
  }

  return LookupColumn.Split(new[] { ";#" },
                StringSplitOptions.RemoveEmptyEntries).Last();
}

set
{
  if (string.IsNullOrEmpty(value))
  {
    LookupColumn = string.Empty;
    return;
  }

  LookupColumn = value.Split(new[] { ";#" },
                StringSplitOptions.RemoveEmptyEntries).Last();
}</code></pre></td></tr>
<tr><th>Get-Result</th><td><pre><code>"Value" : string</code></pre></td></tr>
<tr><th>Set-Result</th><td><pre><code>LookupColumn set to "Value"</code></pre></td></tr>
</table>

Multi-Lookup Translation

In the case of a migration sync from one SharePoint to another, a source list with a lookup column that allows multiple values must be sent to the new list in the new SharePoint. There is a similar lookup set up regarding the labels, but the IDs do not match those of the old SharePoint, leading to data issues. With a simple dynamic column that only provides and sets the lookup labels without the

ID, this example can solve that problem. The dynamic column must be mapped to the Multi-Lookup column of the other data entity.

Note: This sample does not work if your lookup values contain the substring ";#" or "],".

Fields

```
MyMultiLookup = "[3;#Value1],[4;#Value2]"
```

Code

```
get
{
  if (string.IsNullOrEmpty(MyMultiLookup))
  {
    return string.Empty;
  }

  // append a comma so that the split works correctly
  var multiLookupValue = MyMultiLookup + ",";

  var valueArray = multiLookupValue.Split(new [] {"],"},
                     StringSplitOptions.RemoveEmptyEntries);

  if (valueArray.Length == 0)
  {
    return string.Empty;
  }

  List<string> results = new List<string>();
  foreach (var value in valueArray)
  {
    var lookupLabel = value.Split(new[] {";#"},
                StringSplitOptions.RemoveEmptyEntries).Last();
    results.Add(string.Format("[{0}]", lookupLabel));
  }

  return string.Join(",", results);
}
```

```
set
{
  if (string.IsNullOrEmpty(value))
  {
    MyMultiLookup = string.Empty;
    return;
  }

  // append a comma so that the split works correctly
  var multiLookupValue = value + ",";

  var valueArray = multiLookupValue.Split(new [] {"],"},
                     StringSplitOptions.RemoveEmptyEntries);

  if (valueArray.Length == 0)
  {
    MyMultiLookup = string.Empty;
    return;
  }

  List<string> results = new List<string>();
  foreach (var lookupValue in valueArray)
  {
    var lookupLabel = lookupValue.Split(new[] {";#"},
               StringSplitOptions.RemoveEmptyEntries).Last();
    results.Add(string.Format("[{0}]", lookupLabel));
  }

  MyMultiLookup = string.Join(",", results);
}
```

Get-Result	`"[Value1],[Value2]" : string`
Set-Result	`MyMultiColumn set to "[Value1],[Value2]” that results in both values being set in SharePoint`

Identifier Transformation

In the case of a migration sync from one SharePoint to another, a source list with a lookup column must be sent to the new list in the new SharePoint. There is a slightly different lookup set up because the labels changed to another language and the IDs do not match those of the old SharePoint, leading to data issues. Because of the different labels, the former sample will not work. With a simple resolution mapping (old ID = new ID), this example can solve that problem.

Fields	`Old SharePoint (German) : LookupColumn = "3;#Wert"`
	`New SharePoint (English): LookupColumn = "144;#Value"`

Code

```
var mapping = new Dictionary<int, int>
{
  { 1, 14 },
  { 2, 18 },
  { 3, 144 },
  { 4, 26 }
};

if(string.IsNullOrEmpty(LookupColumn))
{
 return -1;
}

var id = int.Parse(LookupColumn.Split(new[] { ";#" },

               StringSplitOptions.RemoveEmptyEntries)[0]);

if (!mapping.ContainsKey(id))
{
  var errorMessage = string.Format("Unknown id: {0}", id);
  throw new Exception(errorMessage);
}

return mapping[id];
```

Result

```
144 : int (which in this sample is the Id for lookup label
"Value" on the new SharePoint (English))
```

Identifier Translation via CSV file

This example is a variation of the previous ID resolution via dictionary, but instead uses a CSV file to define the mapping, making it easier to maintain ID-mappings, for example with Excel.

Note: The CSV file is accessed for every record, so this method can have some performance impact.

Fields

```
Old SharePoint (German) : LookupColumn = "3;#Wert"
New SharePoint (English): LookupColumn = "14;#Value"

CSV file content:
2;13
3;14
4;15
5;16
```

Code

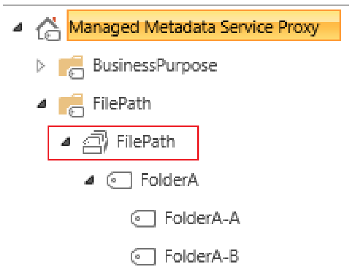
```
if(string.IsNullOrEmpty(LookupColumn))
{
 return -1;
}

var id = int.Parse(LookupColumn.Split(new[] { ";#" },

             StringSplitOptions.RemoveEmptyEntries)[0]);

var csvLines = File.ReadAllLines(@"C:\mapping.csv");

foreach(var line in csvLines)
{
 var entry = line.Split(';');
 if(entry[0] == id)
 {
  return entry[1];
 }
}

var errorMessage = string.Format("Unknown id: {0}", id);
throw new Exception(errorMessage);
```

Result

```
"14" : string
```

Setting Managed Metadata

This example shows how to set a managed metadata column in a SharePoint library to the folder structure that the file resides in. It is assumed that the metadata column is bound to a term set that reflects the folder structure, for example:



Managed Metadata Service Proxy
BusinessPurpose
FilePath
FilePath
FolderA
FolderA-A
FolderA-B

The following Dynamic Column is created on the file system-side:

Fields

```
FilePath = "/FolderA/FolderA-A/myFile.txt"
```

Code

```
get
{
  var convertedFilePath =
                  FilePath.Substring(1).Replace("/", ";");

  if (IsFolder == true)
  {
    return convertedFilePath;
  }

  var delimiterIndex = convertedFilePath.IndexOf(";");
  if (delimiterIndex > 0)
  {
    convertedFilePath = convertedFilePath.Substring(0,
                         convertedFilePath.LastIndexOf(";"));
    return convertedFilePath;
  }
  else
  {
    return string.Empty;
  }
}
```

Result

```
"FolderA;FolderA-A" : string, which will be resolved to the
managed metadata term "FolderA-A" on the SharePoint side
```

Note: In this example the FilePath value is not written back to the filesystem (because there is no set method), so in case someone changes the value of the managed metadata column on the SharePoint side, the change will be ignored in the next synchronization on the file system side (although it notifies about an update). The next synchronization will then automatically correct the value on the SharePoint side back to the actual path as found in the file system.

Synchronizing Documents into a Flat Library

When synchronizing files and documents from a file share into SharePoint, it is sometimes necessary to ignore the existing folder structure on the file system and migrate only the given files. In this case, the “FilePath” property needs to have the folder information stripped.

The Dynamic Column serves as a custom replacement for the original FilePath field and must be created on the file system-side:

Fields	`FilePath = "/FolderA/FolderA-A/myFile.txt"`
Code	`FilePath.Substring(FilePath.LastIndexOf('/'))`
Result	`"/myFile.txt" : string, containing only the filename itself`

In the Mappings section of the connection, map this Dynamic Column to the FilePath property of the SharePoint side.

Note: The example above is for use with uni-directional connections. If you want this to work with a bi-directional connection, you have to define `get{}` and `set{}` and use a placeholder field in the SharePoint library to store the original folder path for later use.

Accessing Fields with Special Characters

This example shows an alternative way to access fields in case of code-unfriendly names. For the return value, it turns all characters into upper case. More text manipulation methods like ToUpper() can be found here.

Fields	`First&LastName = "Jack J. Jackson"`

Code	`var firstNameAndLastName = (string)fields["First&LastName"];`
	`return firstNameAndLastName.ToUpper();`

Result	`"JACK J. JACKSON" : string`

Updating Multiple Columns

This example shows how to update multiple columns through one dynamic column. In this case, it is assumed that the data source has separate fields for the first name and last name, but the other side combines both into one field. So, this will split the incoming value at the space character and provide the first/last name fields with the appropriate parts.

Note: This will not work if the full name contains a middle name.

<table>
<tr><td><b>Fields</b></td><td>

```
(other side) FullName = "Barry Benson"
```

</td></tr>
<tr><td><b>Code</b></td><td>

```
get
{
  return string.Format("{0} {1}", FirstName, LastName);
}

set
{
  if(string.IsNullOrEmpty(value))
 {
    FirstName = string.Empty;
    LastName = string.Empty;
  return;
 }

 var split = value.Split(' ');
  FirstName = split[0];
  LastName = split[1];
}
```

</td></tr>
<tr><td><b>Result</b></td><td>

```
FirstName: "Barry" : string

LastName: "Benson" : string
```

</td></tr>
</table>

Splitting SharePoint-Specific Fields

SharePoint has a field for links that contain an actual URL and a description. This example shows you how to create two dynamic columns that each return either only the URL or only the description.

Fields	MyLink = "www.mysite.com;#My Homepage"

Code
(LinkUrl)
	if (string.IsNullOrEmpty(MyLink))
	{
	  return string.Empty;
	}
	
	var delimiterIndex = MyLink.IndexOf(";#");
	
	if (delimiterIndex < 0)
	{
	  return MyLink;
	}
	
	return MyLink.Substring(0, delimiterIndex);

Code
(LinkDescription)
	if (string.IsNullOrEmpty(MyLink))
	{
	  return string.Empty;
	}
	
	var delimiterIndex = MyLink.IndexOf(";#");
	
	if (delimiterIndex < 0)
	{
	  return MyLink;
	}
	
	return MyLink.Substring(delimiterIndex + 2);

Result	LinkUrl: "www.mysite.com" : string LinkDescription: "My Homepage" : string

Reading Parameters from XML Document

When you have XML Documents with data and parameters, the Layer2 Cloud Connector can help you not only retrieve and integrate the XML data, but also the parameters can be read and synchronized to any other data target.

Fields	`FileName = "myXmlDocument"`

Code (Title)	`var assemblyStrongName = "System.Xml, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089";`
	`var filePath = @"C:\temp\Sync\" + FileName + ".xml";`
	`var xmlAssembly = Assembly.Load(assemblyStrongName);`
	`var xmlDocType = xmlAssembly.GetType("System.Xml.XmlDocument");`
	`var loadMethod = xmlDocType.GetMethod("Load", new[] { typeof(string) });`
	`var selectNodeMethod = xmlDocType.GetMethod("SelectSingleNode", new[] { typeof(string) });`
	`var nodeType = xmlAssembly.GetType("System.Xml.XmlNode");`
	`var innerTextProperty = nodeType.GetProperty("InnerText");`
	`var xmlDocInstance = System.Activator.CreateInstance(xmlDocType);`
	`loadMethod.Invoke(xmlDocInstance, new[] {filePath});`
	`var node = selectNodeMethod.Invoke(xmlDocInstance, new[] {"/item/Title"});`
	`return innerTextProperty.GetValue(node);`

Result	`Title: "sampleTitle" : string`

Xml File	`<?xml version="1.0" encoding="utf-8"?>`
	`<item>`
	`  <Title>sampleTitle</Title>`
	`  <Data>sampleData</Data>`
	`</item>`

Reading Parameters from JSON Web Service

When you have a web service with JSON data, the Layer2 Cloud Connector can retrieve that data and integrate it with any other system.

Fields

```
DataUrl = "https://sampleurl.com/api/users?page=2"
```

Code (FirstUserId)

```
var request = HttpWebRequest.Create(DataUrl);
using (var response = request.GetResponse())
using (var streamReader = new
StreamReader(response.GetResponseStream()))
{
    var responseContent = streamReader.ReadToEnd();
    var jsonObject =
JsonConvert.DeserializeObject<Dictionary<string,
object>>(responseContent);
    var dataString =
JsonConvert.SerializeObject(jsonObject["data"]);
    var data =
JsonConvert.DeserializeObject<List<Dictionary<string,
object>>>(dataString);
    return data.First()["id"].ToString();
}
```

Result

```
FirstUserId: "4" : string
```

JSON Data

```
{
        "page": 2,
        "per_page": 3,
        "total": 12,
        "total_pages": 4,
        "data": [
                {
                        "id": 4,
                        "first_name": "Eve",
                        "last_name": "Holt",
                        "avatar": "http://sampleurl.com/4.jpg"
                },
                {
                        "id": 5,
                        "first_name": "Charles",
                        "last_name": "Morris",
                        "avatar": "http://sampleurl.com/5.jpg"
                },
                {
                        "id": 6,
                        "first_name": "Tracey",
                        "last_name": "Ramos",
                        "avatar": "http://sampleurl.com/6.jpg"
                }
        ]
}
```

Read Extended NTFS File Attributes

The File System Provider provides all extended NTFS attributes in one column named “ExtendedAttributes” in JSON format. This sample shows how to synchronize an extended attribute named “Title” to another system.

Fields	`ExtendedAttributes = (contains all extended attributes for a file or directory)`

Code (Title)	`get`
	`{`
	`    return`
	`JsonConvert.DeserializeObject<List<Dictionary<string, object>>>(ExtendedAttributes)`
	`        .Where(x => x["CanonicalName"]?.ToString() == "Title")`
	`        .Select(x => x["Value"]?.ToString())`
	`        .FirstOrDefault();`
	`}`

Result	`Title: "Document Title Sample" : string`

Licensing

The licensing model of the Layer2 Cloud Connector provides different product licenses, which are explained in more detail online. The product is licensed per local installation; each server that is running the Layer2 Cloud Connector requires its own license key.

Installing a License

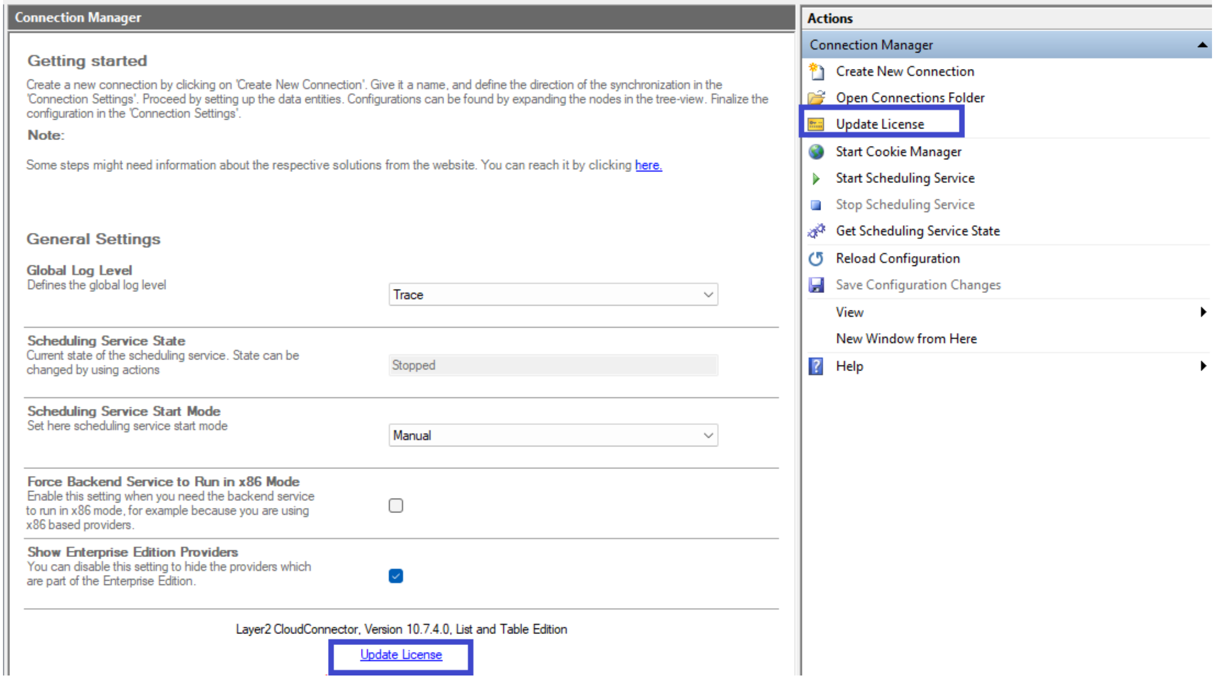
There are two ways to install a license key for the Layer2 Cloud Connector: manually placing the file into the License folder or using the **Update License** action in the Connection Manager UI. If you have any issues with installing the license, contact sales@layer2solutions.com for assistance.

Method One – Manual

1. The license XML file (productkey.xml) will be provided by the Layer2 Sales team by email. **Note:** Do not modify the signed file in any way. It will invalidate it.
2. Copy (do NOT move) the attached file into the License folder in the Layer2 Cloud Connector data directory.
   - C:\Documents and Settings\All Users\Layer2 Cloud Connector\License\   OR
   - C:\ProgramData\Layer2 Cloud Connector\License\   (Vista or higher)
3. Restart the Layer2 Cloud Connector Connection Manager and verify the version has been updated to the correct license that was purchased. If it still says “Shareware”, then the license was not recognized. Some possible reasons why the license was not recognized are that the license does not match the installed software version, that the file was modified, or file corruption.

Method Two – Using the Update License Action

1. The license XML file (productkey.xml) will be provided by the Layer2 Sales team by email. **Note:** Do not modify the signed file in any way. It will invalidate it.
2. Copy the attached file to the local machine.
3. Open the Layer2 Cloud Connector Connection Manager.
4. Select the **Update License** action in the right-hand pane or click the link in the Connection Manager node.



Connection Manager

Getting started

Create a new connection by clicking on 'Create New Connection'. Give it a name, and define the direction of the synchronization in the 'Connection Settings'. Proceed by setting up the data entities. Configurations can be found by expanding the nodes in the tree-view. Finalize the configuration in the 'Connection Settings'.

Note:

Some steps might need information about the respective solutions from the website. You can reach it by clicking here.

General Settings

Global Log Level
Defines the global log level: Trace

Scheduling Service State
Current state of the scheduling service. State can be changed by using actions: Stopped

Scheduling Service Start Mode
Set here scheduling service start mode: Manual

Force Backend Service to Run in x86 Mode
Enable this setting when you need the backend service to run in x86 mode, for example because you are using x86 based providers. ☐

Show Enterprise Edition Providers
You can disable this setting to hide the providers which are part of the Enterprise Edition. ☑

Layer2 CloudConnector, Version 10.7.4.0, List and Table Edition

Update License

Figure 45 - Location of the Update License action

5. Select the productkey.xml file and click **Open**.
6. Verify the version has been updated to the correct license that was purchased.

Connection Definition Files

All connections will be saved as an XML file (connection name plus .xml extension) on the local machine hosting the Layer2 Cloud Connector. These files can be found in The Layer2 Cloud Connector Data Directory under Connections.

These files are created automatically by the Connection Manager when you save a new connection. However, as they are simple XML, they can be programmatically generated if you require multiples of files.

<connection>

The root node of the XML file is the `<connection>` node. This node can have the following connection attributes:

firstRun

This is date and time information, defining when the automatic background synchronization will be run for the first time. It is tied to the UI setting First Synchronization. This setting is only influential if it is set to a future date. The Layer2 Cloud Connector will wait until this specific date and time to start the synchronization for the first time allowing exact timing of the synchronization runs. Different date and time formats can be used here, even localized to the system that runs the Layer2 Cloud Connector. One format which is always accepted is mm/dd/yyyy hh:MM.

maximumConsecutiveErrors

Tied to the UI setting Number of Consecutive Errors. By default, this attribute is not present in the XML configuration which means the number before aborting is set to 10. If the sync should never abort due to consecutive errors, the value is 0.

interval

This defines the synchronization interval in minutes. This attribute is mandatory although it is ignored if **enabled** is set to “false”. Is tied to the UI setting Interval.

enabled

This can be either true or false. It specifies if the connection is enabled for automatic background scheduling, and it is mandatory. Is tied to the UI setting Scheduling Enabled.

version

This is an internally used attribute which is used by the Layer2 Cloud Connector to differentiate between versions of the connection definition file format. Currently, this should always be set to 1.4.

Two sub-nodes need to be enclosed in the `<connection>` root node: `<dataEntities>` and `<fieldMappings>`.

<dataEntities>

The `<dataEntities>` node must contain exactly two `<dataEntity>` sub-nodes. A `<dataEntity>` is defined using the following attributes:

type

The type of entity can have one of three distinct values: source, destination, or bi-directional. The types of both entities are defining the synchronization direction. If one entity is “bidirectional”, the other one must be defined as “bidirectional” too. If one of the entities is typed as being “destination”, the other entity must have the “source” type.
This attribute is optional. If omitted, both entities will be considered “bidirectional”. Is tied to the UI setting Direction.

name

This sets a name for the data entity. It is mandatory and both entities need to have different names. Is tied to the UI setting Data Entity Title.

provider

This identifies the ADO provider that is used for this data entity. This attribute is expecting the invariant name of the provider, which is a unique name to identify a specific ADO provider. This attribute is mandatory. Is tied to the UI setting Data Provider.

connectionString

The connection string is the provider-specific configuration information to connect to that data source. For more information on connection strings, see the Data Entity: Connection String section. This attribute is mandatory.

selectStatement

This attribute is optional and is necessary for some providers. It defines a query for selecting the data. For more information on select statements, see the Data Entity: Select Statement section.

primaryKey

This attribute is optional and can be used to define a primary key, if it is not automatically returned by the provider. For more information on primary keys, see the Data Entity: Primary Key(s) section.

deletionProtectionThreshold

This attribute is optional and defines how many records can be deleted without asking for confirmation. This feature should prevent unexpected data loss due to unforeseen circumstances such as hardware failure in the source data entity. For more information, see Deletion Protection Threshold.

disabledOperations

The attribute allows users to define which operations/transactions are skipped for the entity during execution of a synchronization. The setting is optional and can contain or more of the values, comma separated: Update, Insert, Delete. For more information on disabled operations, see the Data Entity: Disable Operations section.

writeOnly

This is an optional attribute that can only be used on the “destination” of a uni-directional sync. It causes the destination to not be read during the data read step of the sync, and thus should only be used if the data in the destination is never changed outside of the data synchronizations. Is tied to the UI setting Ignore Changes Within Target.

Optional Encryption Flags

isConnectionStringEncrypted
isSelectStatementEncrypted
isPrimaryKeyEncrypted

These flags are used internally to specify that the contents are encrypted. Is tied to the UI setting **Encrypt** which is a check box under each item that can be encrypted. See the Data Entity section for more details.

The <dataEntity> nodes may also contain optional <dynamicColumns> nodes, which are explained below in their own section.

<dynamicColumns>

The <dynamicColumns> node is optional and contains one or more <dynamicColumn> sub-nodes. The content of the <dynamicColumn> node is an expression (the code executed when the dynamic column is read) encased in a CDATA block. A <dynamicColumn> is defined using the following attributes:

name

This mandatory attribute defines the name of the Dynamic Column. It must be unique in the context of the data entity (for example, two Dynamic Columns under two data entities can share the same name).

<fieldMappings>

The second node that is always contained inside the connection node is the <fieldMappings> node. It contains the definition of mappings between the fields of both data entities. The <fieldMappings> node can only have one attribute which is optional: <autoMapping>. This attribute can be either true or false and defines if fields in both data entities that have the same name will automatically be considered as mapped. The default value, if the attribute is omitted, will be false.

Inside of the <fieldMappings> node there might be an arbitrary number of <fieldMapping> (singular) sub-nodes defining a single mapping. Every <fieldMapping> node has exactly two sub-nodes both named <field>. These two subnodes define the fields that should be mapped. Each field node must have two attributes: name and entity.

entity

Entity is the name of the data entity the field belongs to. This information is not case sensitive.

name

Name specifies the field name. This information is not case sensitive.

If the connection is set to bi-directional, there is also an optional <conflictResolution> node that will be present inside the <fieldMapping> node.

conflictResolution

This defines which conflict resolution mode will be used for a bi-directional connection. If it is absent, it will default to the conflict resolution mode “FailAndAbort”. The other options are: WarnAndContinue, WinnerLoser, or KeepBoth. Is tied to the UI setting Conflict Resolution Mode.

focusedEntity

This is only available as a sub node to <conflictResolution> if “WinnerLoser” or “KeepBoth” are defined. This is mandatory of one of those resolution methods is being used. The option will be the name of the Entity that is set as the Master or Winning entity. Is tied to the UI setting Master Entity/Winning Entity.

In addition, there might be an optional attribute on the `<fieldMapping>` node called isMapped. It can be true or false, and it can be used in case of an automatic mapping to explicitly un-map two fields that have the same name but should be excluded from the synchronization.

Logging and Alerting

The Layer2 Cloud Connector creates a variety of messages about its activities during synchronization. At first, these messages will be filtered by the current global log level. Any message that gets through this filter will be handed to the logging subsystem NLog. It is a free logging framework used by the Layer2 Cloud Connector system to manage its logging activities.

By default, there will be one log file created for each connection, as well as files for Server.log, MMC.log, and Scheduler.log file These files will be saved under the Logs subdirectory under the Layer2 Cloud Connecter Data Directory. You can also see some logging details in the Layer2 Cloud Connector UI. See the Logs section for more information.

If a particular log reaches a certain size, it will be renamed as an Archive log and the newest information will be logged into the file with the non-archive marked name. Archive logs have the following naming convention, where <logname> is the name of the connection or the log type, and the #s will be the number of the archive starting with 00000:

<logname>Archive#####.log

Typically, up to 10 archive files will be kept for each log before the oldest ones are overwritten. This setting can be changed in the NLog.config file as well.

Connection Logs

This log type will record the synchronization events of the connection. They are named after the connection they belong to plus the .log extension. The level of details in the connection log will depend on the current Global Log Level setting, which you can read about in the section Setting the Log Detail Level.

Server.log

This log type records the actions of the core Layer2 Cloud Connector server (aka the Backend Service). It can be used for identifying issues with main functions and features of the application.

MMC.log

This log records the actions and events of the Layer2 Cloud Connector UI, which is an MMC snap-in. It can used for identifying issues with the UI and the MMC snap-in for the application.

Scheduler.log

This log records the actions of the Scheduling Service, such as when the next connections are scheduled to sync and issues with the scheduling.

System.log

This log records the actions of the system and the Scheduling Service in Layer2 Cloud Connector versions prior to 7.8. It is no longer in use after that version as the functionality has been replaced by the logs above.

NLog.config

Also located in the Logs folder is the logging configuration file, NLog.config. This file contains all logging specific configuration settings and defines different log formats for the Connection Manager and the background Windows Services log files. In the log configuration, writing the log to a .log file (which is viewable with a simple text editor) is enabled by default. There are also sample configurations for using the Windows Event Log or databases. You can uncomment the desired configuration to log to different targets. To use a database target, a connection string must be edited, and a target table must be created. The required SQL script to create an appropriate table is also placed in the configuration file as a comment. Since some log entries contain prohibited characters for SQL, the insertion of some entries may fail.

More detailed information about how to configure NLog can be found in the NLog documentation at https://github.com/nlog/nlog/wiki.

Windows Event Log Configuration

To configure the Layer2 Cloud Connector for logging into the Windows Event Log, a target section in the log configuration file needs to be created, like the following:

```
<target xsi:type="EventLog"
      name="EventLogTarget"
      layout="${message}"
      source="Layer2 Cloud Connector"
      log="Application" />
```

xsi:type

Defines the type of target to log to. For the Windows Event Log this needs to be “EventLog”, but there is a variety of different target-types to use. Please see the NLog documentation for a complete reference.

name

This attribute contains the name of the target to reference it in the rules configuration.

layout

This defines the layout of the log message to be written to the event log. It can contain several placeholders of the format ${placeholderName}. Please see the NLog documentation for a complete reference.

source

The name of the source to appear in the Windows Event Log.

log

The name of the event log to be written to.

Additionally, to make the new target work, it is necessary to define a logging rule in the configuration like the following.

```
<logger minlevel="Warn"
      name="*"
      writeTo="EventLogTarget" />
```

Email Alert Configuration

To configure the Layer2 Cloud Connector for sending emails for specific log messages, a target section in the log-configuration file needs to be created, like the following.

```
<target xsi:type="Mail"
     name="MailTarget"
     subject="Synchronization Error"
     body="${message}"
     to="me@myCompany.com"
     smtpUserName="cloudConnector@myCompany.com"
     smtpPassword="myPassword"
     smtpAuthentication="Basic"
     smtpServer="mail.myCompany.com" />
```

xsi:type

Defines the type of target to log to. For mail alerts this needs to be “Mail”, but there is a variety of different target-types to use. Please see the NLog documentation for a complete reference.

name

This attribute contains the name of the target to reference it in the rules configuration.

subject

This defines the subject of the mail alert. It can contain several placeholders of the format ${placeholderName}. Please see the NLog documentation for a complete reference.

body

This defines the body-text of the mail alert. It can contain several placeholders of the format ${placeholderName}. Please see the NLog documentation for a complete reference.

This attribute contains the email addresses of the recipients, separated by semicolons.

smtpUserName and smtpPassword

These settings specify the account on the mail server to be used to send the email alert.

smtpAuthentication

Defines the authentication mode to use when authenticating against the SMTP mail server. It can be one of the following.

- **None:** Anonymous access to the mail server. Username and password are not necessary with this setting.
- **Basic:** Basic username and password authentication.
- **Ntlm:** NTLM-Authentication. Username and password are not necessary with this setting.

Additionally, to make the new target work, it is necessary to define a logging rule in the configuration like the following.

```
<logger minlevel="Error"
name="*"
writeTo="MailTarget" />
```

Service Management

As of version 7.8, the Layer2 Cloud Connector installs two Windows services – The Backend Service, which is the core responsible for synchronization logic, connection management, and most other Layer2 Cloud Connector features; and the Scheduling Service which is responsible to running the connections configured to run automatically.

While the Scheduling Service is optional, the Backend service must be running for the Layer2 Cloud Connector to function properly. Both services are explained in greater detail below.

Note: If running a version prior to 7.8, you only have what is now referred to as the Scheduling Service and can ignore the information about the Backend Service.

Backend Service

This is the core of the Layer2 Cloud Connector. This service is responsible for synchronization logic, connection management, and most other Layer2 Cloud Connector features.

This Windows service is registered in the Windows system during installation and will by default be in the “Stopped” state. It will start automatically the first time the Connection manager is opened.

Being the core of the product, the Backend Service must be running for the Layer2 Cloud Connector to function. The service can be administrated like any other Windows service through the "**Services" MMC-snap-in**.

By default, the Backend Service is set up to use the "local system account" on the host machine as the logon account. However, there can be synchronization issues if this is left as the default account, as the SYSTEM account may not have access to all data entities. It is recommended that **Layer2CloudConnectorBackendService** logon account should be changed to a service account that does have access to the data entities. You can change this in the properties of the service.

All synchronizations are run through this service. There are no more "manual" and "service" synchronizations as there had been earlier, so the opportunities to accidentally run two versions of the same connection are removed. The Backend Service will continue to run after the Connection Manager is closed. Syncs will continue if they were in progress when the application was closed and be triggered by the Scheduling Service if you are using that.

The Backend Service also includes a REST API you can access for greater usability of the Layer2 Cloud Connector functionality. For more information and examples, see the REST API section.

Scheduling Service

The Scheduling Service enables the Layer2 Cloud Connector system to synchronize according to the configured schedule, even if the actual Connection Manager is not running.

This Windows service is registered in the Windows system during installation and will by default be in the "Stopped" state. It is possible to start and stop the service, as well as changing the startup type, from the Connection Manager (see the Configuration section for details), but also the service can be administrated like any other Windows service through the "**Services" MMC-snap-in**.

REST API

Since version 7.8.0.0, the Layer2 Cloud Connector can be accessed via REST API. The auto-generated documentation of the actual methods can be retrieved via web browser by opening http://localhost:20537/swagger/ui/index on the machine where the Layer2 Cloud Connector is running.

Configuration File

This file can be found under the installation directory (usually *C:\Program Files (x86)\Layer2 Cloud Connector*) and is called *Layer2.Data.Synchronization.BackendService.exe.config*. It can be opened with a text editor to change settings, but changes won't be used until the service is restarted.

WebApiUrl

This is the access point of the Backend Service REST API and should only be changed if the port (20537) is already in use. If the port is changed, the registry key ApiUrl needs to be adjusted accordingly.

PowerShell Samples for API usage

Here are some samples for accessing the REST API via PowerShell for scripted usage of the Layer2 Cloud Connector. The swagger link points to the automatically generated documentation of the API function used in the sample.

Note: These scripts do not function with v2 of PowerShell, so if you get errors about missing “Invoke-RestMethod”, please update your installed version of PowerShell to the latest.

Get Server Version

Script	`$result = Invoke-RestMethod -Uri "http://localhost:20537/v1/server"`
	`$result.version`

Description	This will ask the server for general information and then returns the version.
Swagger	http://localhost:20537/swagger/ui/index#!/Server/Server_GetServerInformation

Get All Connections

Script	`Invoke-RestMethod -Uri "http://localhost:20537/v1/connections"`
Description	This will return all connections currently known by the server.
Swagger	http://localhost:20537/swagger/ui/index#!/Connection/Connection_GetConnections

Start Connection

Script	`$result = Invoke-RestMethod -Uri "http://localhost:20537//v1/connections/Sample - Excel to SharePoint/startSync" -Method Post`
Description	This will request the start of a synchronization for the connection “Sample – Excel to SharePoint” and return the server response.
Swagger	http://localhost:20537/swagger/ui/index#!/Connection/Connection_StartSync

Get Connection Status

Script	`Invoke-RestMethod -Uri "http://localhost:20537/v1/connections/Sample - Excel to SharePoint/status"`

Description	This returns the current status (running or not) and progress of the connection “Sample – Excel to SharePoint”.
Swagger	http://localhost:20537/swagger/ui/index#!/Connection/Connection_GetConnectionStatus

Create Connection

Script

```
$body = @{
    name = "TestConnection1"
    leftEntity = @{
        name = "FileSystem Source"
        providerInvariantName = "Layer2.FileSystem.Provider"
        connectionString = "Directory=C:\myFolder"
    }
    rightEntity = @{
        name = "SharePoint Target"
        providerInvariantName = "Layer2.SharePoint.Provider"
        connectionString = 
"Url=https://myCompany.sharepoint.com/sites/mySite;List=MyList;Authentication=Office365;User=myUser@myCompany.onmicrosoft.com;Password=myPassword"
        operations = ("Delete", "Insert", "Update")
    }
}

$jsonBody = $body | ConvertTo-Json

Invoke-RestMethod -Uri "http://localhost:20537/v1/connections" -Method Post -Body $jsonBody -ContentType "application/json"
```

Description	This creates a new connection named “TestConnection1” with a file system source and a SharePoint target where reading is disabled (also known as write-only entity) and then returns the connection with additional default values, such as a left-to-right direction and auto-mapping.
Swagger	http://localhost:20537/swagger/ui/index#!/Connection/Connection_CreateConnection

Change Connection Settings

Script

```
$body = @{
    name = "TestConnection1"
    direction = "RightToLeft"
}

$jsonBody = $body | ConvertTo-Json

Invoke-RestMethod -Uri "http://localhost:20537/v1/connections" -Method Patch -Body $jsonBody -ContentType "application/json"
```

Description	This sets the direction of the connection “TestConnection1” to right-to-left and then returns all connection information of the updated connection.

Swagger	http://localhost:20537/swagger/ui/index#!/Connection/Connection_UpdateConnection

Delete Connection

Script	`Invoke-RestMethod -Uri "http://localhost:20537/v1/connections/TestConnection1" -Method Delete`
Description	This deletes the connection “TestConnection1” and returns the result of the operation, such as an error message if the connection was not found on the server.
Swagger	http://localhost:20537/swagger/ui/index#!/Connection/Connection_DeleteConnection

Console Mode

The Layer2 Cloud Connector Scheduling Service can be started in console mode. This can be used to:

- Execute connections on-demand.
- Execute several connections with dependencies, one after another.

During installation, the setup will create a Start menu shortcut that reads **Start synchronization manually,** which starts the Layer2 Cloud Connector in console mode. The console application is identical to the Scheduler Windows service binary. It is in the program directory as Layer2.Data.Synchronization.Service.exe.

Starting this executable on a command prompt will result in an error message stating that a service cannot be started directly. To make the service run as a console application, it needs command line arguments.

To synchronize all existing connections that are scheduled to run at this time, exactly as the Scheduling synchronization service would do it, the command is:

```
Layer2.Data.Synchronization.Service.exe -console
```

To force all connections that are enabled for scheduling to run, even if they are not supposed to run at this time, the command is:

```
Layer2.Data.Synchronization.Service.exe -console -force
```

If you want to synchronize a specific connection in console mode, the command is, where <connection name> is the name of the connection you wish to synchronize:

```
    Layer2.Data.Synchronization.Service.exe –execute "<connection name>"
```

Please note that, if the Backend Service is not yet running, the console mode should be run as administrator so that the Backend Service can be started successfully.

Please also note that you might need to use the absolute path to the executable, which is by default is here:

C:\Program Files\Layer2 Cloud Connector\Layer2.Data.Synchronization.Scheduler.exe

For other options to programmatically start connections, see the REST API section.

Registry Keys

Some global Layer2 Cloud Connector settings can be changed by creating or modifying a registry key. All keys need to be created under the following node:

HKEY_LOCAL_MACHINE\SOFTWARE\Layer2 GmbH\CloudConnector

AutoBackupInterval

This defines how long the Layer2 Cloud Connector waits between two Metabase backups (in hours). You can disable this feature by using a negative value for the interval.

Default Value: *1*

ApiUrl

Can be created to change the URL where the backend-service is running. If this is specified, the backend service configuration needs to be adjusted accordingly. (See Configuration File) It is possible to host the backend-service on a server and access it through the MMC on a client machine. However, this is currently not officially supported.

MMCLog_Directory

This defines the location of the MMC log file (client-side of the Layer2 Cloud Connector).

Default Value: *${environment:ALLUSERSPROFILE_APPDATA}\Layer2 Cloud Connector\Logs*

MMCLog_ArchiveAboveSize

This defines when the MMC log should create an archive file to keep log files below a certain file size (in bytes).

Default Value: *10000000*

MMCLog_MaxArchiveFiles

This defines how many archive files for the MMC log should be kept.

Default Value: *10*

MemoryWatchdogThresholdInMb

This defines the threshold of available memory that is considered to be a state of “low memory” for the host machine. When the value is reached, it causes all running connections to be automatically aborted. This value is given in megabytes.

Default Value: *100*

MemoryWatchdogPollingIntervalInMilliseconds

This defines how often the available memory is checked against the above-mentioned low memory threshold. This value is given in milliseconds.

Default Value: *500*

ApiCalls_Timeout

This defines the time in milliseconds until a frontend (MMC/Scheduler) API call times out.

Default Value: *0*

ApiCalls_MaxRetries

This defines how often a failed API call will be retried before being handled as an error.

Default Value: *5*

ApiCalls_WaitBetweenRetries

This defines the milliseconds to wait between two retry attempts.

Default Value: *1000*

Automatic Fields

The Layer2 Cloud Connector adds some fields to every data-entity result, regardless of the provider in use. These fields are referred to as automatic fields. The name of an automatic field always starts with the prefix CC. Currently the Layer2 Cloud Connector adds the following automatic fields:

CCConnectionName

This field contains the name of the Layer2 Cloud Connector Connection.

CCEntityName

This field contains the name of the Layer2 Cloud Connector data entity.

These can be used in the case where one would execute several different synchronizations with the same target to differentiate between data from a specific connection or data entity.

Layer2 Data Providers

Along with the Layer2 Cloud Connector system, several ADO providers are included in the installation which enable the Layer2 Cloud Connector to connect to many different data entities.

Layer2 Data Provider for Exchange

The Layer2 Exchange Provider connects to Microsoft Exchange servers, on-premises or cloud-based (Exchange Online, Outlook.com). It can read and write contacts, appointments, emails, notes, and tasks. This provider is often used to mobilize line-of-business data from SQL or ERP/CRM and make it available for traveling users. It can also be used for aggregating calendars or tasks from several systems and keep it sync.

See also the Exchange Provider Specification on the web.

Connection String

A typical connection string for the Exchange provider looks like this:

User=<UserName>;

Additionally, the password needs to be typed into the Connection String or Password field.

With this connection string, the provider will automatically discover the exchange-server URL by the given username and connect with the given credentials.

If you are trying to connect to Exchange Online, the default connection string should look like the following. You won't have to type in a password since that will be handled by an authentication popup:

Url=https://outlook.office365.com; Authentication=Microsoft_Modern_Exchange;
User=<UserName>;

Here is the full list of connection string parameters for this provider:

URL

This is the URL of the Exchange server. This setting is optional for Exchange on-premises, but mandatory for Exchange Online, where it should be https://outlook.office365.com. If it is not specified, the provider will attempt auto-discovery by using the specified username or mailbox-user. Auto-discovery usually takes an additional 4-5 seconds.

Version

This setting specifies the Exchange version which will be assumed while processing requests on the Exchange server. If the version configured by this setting is lower than the actual Exchange version, some properties that have been introduced in a newer Exchange version may not be available. If it is higher than the actual version, errors might be produced if properties are accessed which are not available in the actual Exchange server version.

This setting is optional and, in most cases, does not need to be specified at all. If it is not specified, the provider will attempt to detect the server-version automatically.

Possible values are:

- Exchange2007_SP1
- Exchange2010
- Exchange2010_SP1
- Exchange2010_SP2
- Exchange2013
- Exchange2013_SP1

Authentication

This parameter is **mandatory** for Exchange Online tenants created after October 2020. The only supported authentications currently are Microsoft_Modern_Exchange and Microsoft_Modern_Exchange_Admin.

User

This parameter is **mandatory**. The name of the user to authenticate with against the Exchange server. If the user is not associated with a mailbox on the Exchange server, the **Email** parameter will be mandatory.

Password

This parameter is **mandatory** for Exchange on-premises and older Exchange Online tenants. Exchange Online tenants created after October 2020 need to specify the authentication parameter instead. The password of the user to authenticate with against the Exchange server. It needs to be typed into the Connection String or Password field.

Email

This setting defines the mailbox which should be accessed. It is optional and if not specified, the provider will access the mailbox of the user specified with the **User** parameter. If the **Email** parameter is specified, the user which is used for authentication (specified with the **User** parameter) must have permission to access the mailbox account. Based on the **Impersonation** parameter, this either means delegated access or permission to impersonate.

Impersonation

When using the **Email** parameter to access a different mailbox, you can either do so via Impersonation (Impersonation=true) or Delegation (Impersonation=false). This setting is optional and defaults to “false”.

Recursive

This setting is used to define whether the provider should traverse through all subfolders to gather the items (Recursive=true) or only return items which are direct children of the specified folder (Recursive=false). This setting is optional and defaults to “false”.

NamingScheme

When synchronizing specific item-types, such as contacts or appointments, to another system, fields with the same meaning may have different names in both systems. To minimize the effort for setting up synchronizations, the Exchange provider supports naming-schemes which effectively rename the resulting columns to match the names of the target system, which enables automapping-functionality for these scenarios. Currently, there is one naming-scheme “SharePoint”, which will rename the Exchange-columns for contacts and appointments to match the corresponding SharePoint fields. This setting is optional.

BatchReadItems

With this parameter, it is possible to define the maximum number of items that will be read in one request from the Exchange server. This setting is optional, defaults to 1000, and normally does not need to be adjusted. When working with large amounts of data, performance may be enhanced by tweaking this setting.

TrustAllSSL

This parameter can be used to make the provider bypass the validation of certificates when connecting through HTTPS. This setting is only for testing purposes; it is NOT recommended that this be used in a production environment due to the security risk. The setting is optional and defaults to “false”.

Queries

The Exchange provider supports a SQL-like syntax to query the items from the Exchange server as described below.

```
SELECT [Fields] FROM [Folder] WHERE [AQS-filter]
```

The list of fields can either contain a wildcard (*) or a comma-separated list of Exchange-property names. Furthermore, it is possible to rename the fields by using the SQL alias-syntax, i.e., SELECT Companyname AS cmp, would select the contents of the property Companyname and populate it as a field named cmp in the result.

The folder part specifies the Exchange folder to select the items from. It is defined as a folder path separated by slashes (/) while the first folder must always be one of the Exchange root folders as seen below:

Calendar	- Root folder for calendars.
Contacts	- Root folder for contacts.
DeletedItems	- Deleted items folder.
Drafts	- Drafts folder.
Inbox	- Mail inbox.
Journal	- Journal items.
Notes	- Notes folder.
Outbox	- Mail outbox.
SentItems	- Mail sent items.
Tasks	- Tasks folder.
MsgFolderRoot	- Root folder for public folders.
PublicFoldersRoot	- Root folder for public search folders.
SearchFolders	- Root folder for search folders.

The filter part, which starts with the keyword WHERE, is optional and contains a filter-query in the Advanced Query Syntax. For more information about AQS, please refer to the Microsoft documentation.

Layer2 Data Providers for File System, FTP and DropBox

There are 5 Layer 2 Data Providers that can be used to synchronize files:

- File System Provider
- FTP Provider
- DropBox Provider
- GoogleDrive Provider (please refer to Layer2 Data Provider for GoogleDrive)
- Azure Blob Storage Provider (please refer to Layer2 Data Provider for Azure Blob Storage)
-

Note: Files can also be synchronized from/to SQL database blobs using the .Net Framework Data Provider for SqlServer.

The Layer2 File System provider, which is available with all licenses, connects to local or remote file system resources to retrieve files. Server file shares are also supported.

The Layer2 FTP provider is only available in Layer2 Cloud Connector Enterprise and connects to an FTP Server via FTP(S) to retrieve files. The provider supports a variety of FTP servers, among which are FileZilla, PureFTPd and XLight and many others.

The Layer2 DropBox provider is only available in Layer2 Cloud Connector Enterprise and connects to a DropBox account to retrieve files.

All providers are typically used to synchronize files with SharePoint or Office 365 / OneDrive for Business document libraries (including the files) or lists (metadata only). The providers support both reading and writing.

Connection String

If a complete select statement is used, the connection string can be left empty, otherwise it usually looks like this:

File System Provider:	*directory=<folderPath>;*
FTP Provider:	*directory=<ftpPath>;*
DropBox Provider:	no connection string required

Here is the full list of connection string parameters for the providers:

Directory

This parameter is not used for the DropBox provider, as the file location is determined by the user account.
For File System and FTP provider, this parameter is optional if the select statement states a directory in it’s FROM part, otherwise it is mandatory. It is the path for the files to be retrieved. The value can be a local path or a network share in UNC format for the File System provider and an FTP(S) path for the FTP provider. If both this and the select statement FROM part are used, the select statement has a higher priority and a warning will be logged.
Note: To avoid errors, it is recommended for network shares that you use the device name (UNC) or its IP address instead of a mapped drive letter. **Authentication**
For the DropBox provider please refer to Layer2 Dropbox Provider Specifics.
It specifies how the providers authenticates against the target server. See the Authentication Methods section for details.

User

This part of the connection string specifies the username for the account which is used to authenticate. It needs to be specified for several authentication methods. See the Authentication Methods section for details.
For DropBox this setting is not required. However, if you have multiple connections targeting different accounts, you need to specify the user in order to differentiate between them.

Password

This setting is not used with the DropBox provider.
It defines the password for the account which is used to authenticate. It needs to be specified for several authentication methods by typing it into the Connection String or Password field. See the Authentication Methods section for details.

FilenameFilter

This is an optional search pattern for filtering files by name or extension. The parameter accepts standard windows file search patterns like these: * ?

Note that only one filter can be set with this parameter.

FileReadMode

This is an optional setting for specifying how to read files. Options are:

- **Recursive (Default):** Gets the content of all folder and subfolder recursively.
- **Flat:** Gets the files only in the specified folder excluding all subfolder contents.

FilesToInclude

This is an optional setting for reading hidden or protected system files which is only used by the Files System provider. The parameter has three possible values:

- **Visible (Default):** Hidden or protected system files will be excluded.
- **Hidden:** Only protected system files will be excluded.
- **All:** All files regardless from its attribute values will be included.

IncludeFolders

This setting is optional. If it is not specified, the provider will read all folders as records for synchronization by default. If this is false, folders will not be treated as records, but will only be created implicitly through the file-paths.

IncludeDirectorySize

This setting is optional. If it is not specified, the provider will return 0 for all directory-sizes. This is due to optimization: To determine the size of a folder, it is necessary to traverse through the whole directory-tree and sum up the file-sizes. This can be very slow on a large number of files and is in many cases unnecessary. To retrieve the actual sizes, set this to “true”.

IncludeExtendedAttributes

This setting is optional and only used by the File System Provider. If it is set to “true”, the provider will return a field named “ExtendedAttributes”, which contains a JSON-based representation of the extended file attributes. This can be used with dynamic columns to include extended attributes in the synchronization. If it is omitted or set to “false” the field is not returned. Note that including the extended attributes has impact on reading performance. See also: Read Extended NTFS File Attributes

IncludeOwner

This setting is optional and only used by the File System Provider. If it is set to “true”, the provider will return the owner information of files and folders in the owner field. The owner information field is read-only. If it is omitted or set to “false” the field is not returned. Note that including the owner might cause a lot of warnings in case that owners no longer exist in the Active Directory.

Select Statement

The providers support a SQL-like select statement which is described in detail here: SQL-Like Select Statement

Layer2 Dropbox Provider Specifics

The creation and modification date of Dropbox are read-only as they cannot be changed via the Dropbox API.

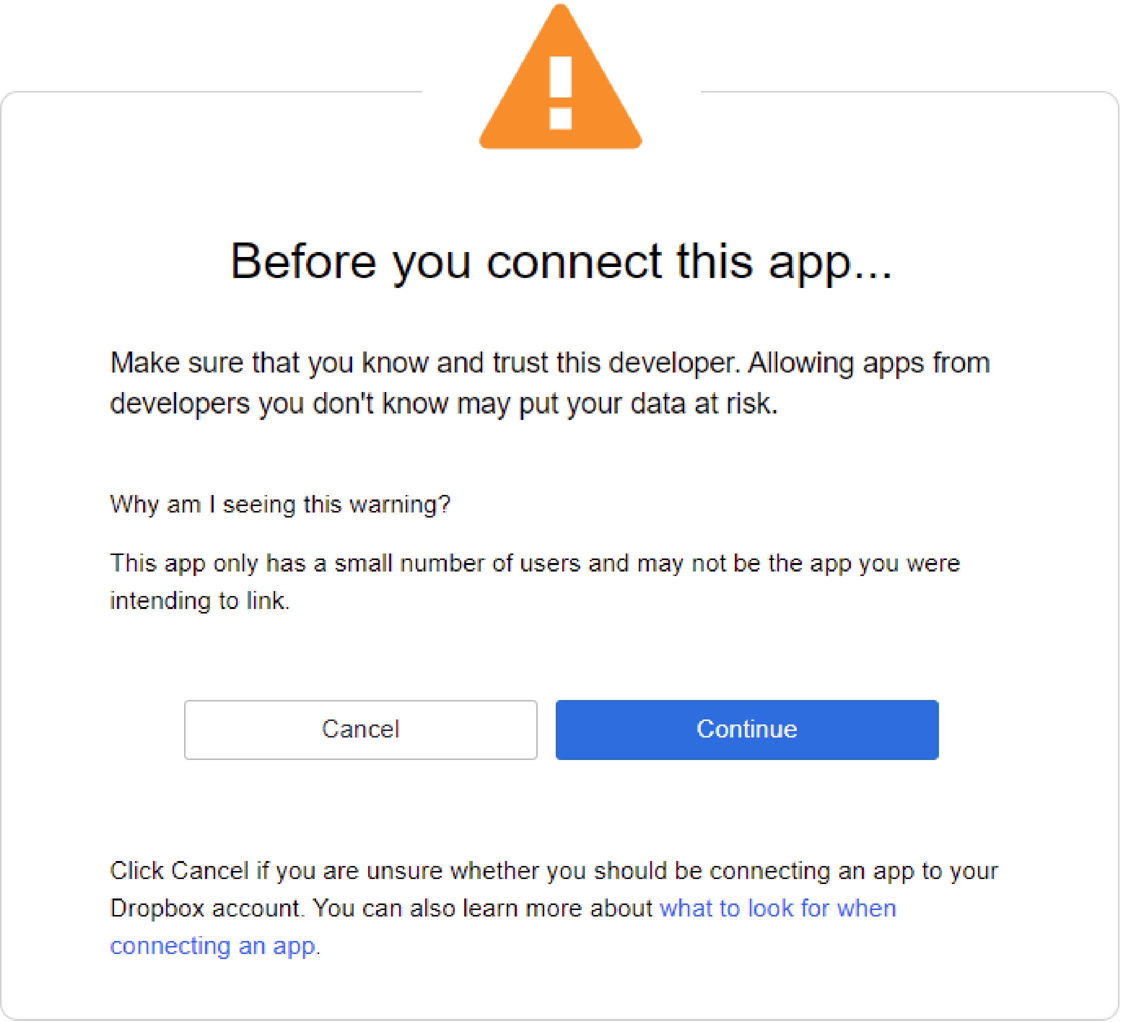
In case you have more than one Dropbox connection and they are targeting different accounts, you need to differentiate between the connections by adding the *User* parameter with different values to the connection strings.

There are two options for authentication:

- Authentication using the *Layer2 Cloud Connector* app
- Authentication using your own custom app

Authentication using the Layer2 Cloud Connector app

In case you do not add any authentication parameters to the connection string, a Dropbox app named *Layer2 Cloud Connector* which has been created by the Layer 2 GmbH is used to access the files in your Dropbox. As this app is currently not verified by Dropbox, you will get the following warning when trying to refresh the authentication token:



Before you connect this app...

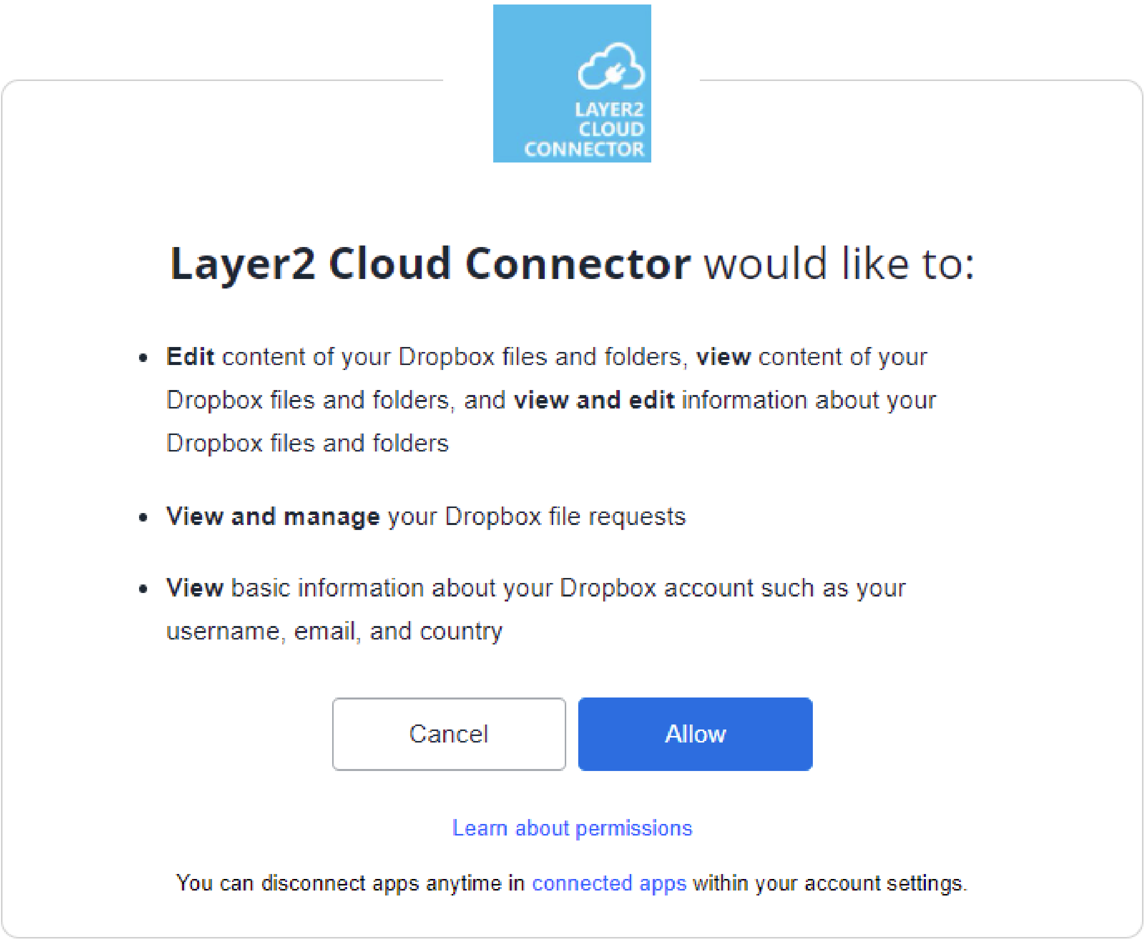
Make sure that you know and trust this developer. Allowing apps from developers you don't know may put your data at risk.

Why am I seeing this warning?

This app only has a small number of users and may not be the app you were intending to link.

Click Cancel if you are unsure whether you should be connecting an app to your Dropbox account. You can also learn more about what to look for when connecting an app.

If you click "*Continue*", you will get the consent dialog where you give the app the permission to access your files:



LAYER2 CLOUD CONNECTOR

Layer2 Cloud Connector would like to:

- **Edit** content of your Dropbox files and folders, **view** content of your Dropbox files and folders, and **view and edit** information about your Dropbox files and folders
- **View and manage** your Dropbox file requests
- **View** basic information about your Dropbox account such as your username, email, and country

You can disconnect apps anytime in connected apps within your account settings.

Authentication using your own custom Dropbox app

You can use your own Dropbox application for authentication:

- Go to the [Dropbox App Console]()
- Click the **Create app** button
- Choose the **Scoped access** API
- Choose the type of access: **Full Dropbox – Access to all files and folder in a user's Dropbox**
- Name your app and click **Create app**
- Go to the **Permissions** tab, select all **Files and folder** permissions and click **Submit**
- Note down **App key** and **App secret** from the **Settings** tab

Add App key as *ClientID*- and App secret as *ClientSecret*-parameters to the connection string.

Layer2 Data Provider for GoogleDrive

The Layer2 GoogleDrive provider is available in Layer2 Cloud Connector Enterprise and connects to a Google account to retrieve files from the related GoogleDrive.
The provider is typically used to synchronize files with SharePoint or Office 365 / OneDrive for Business document libraries (including the files) or lists (metadata only). It supports both read and write.

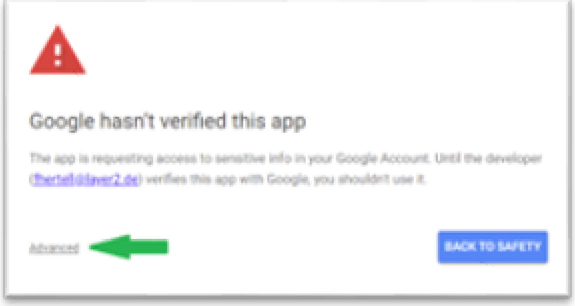
Authentication Options

There are two options for authentication which will be described in the following sections:

- Authentication using the *Layer2 CloudConnector* app
- Authentication using your own custom app

Authentication using the Layer2 Cloud Connector app

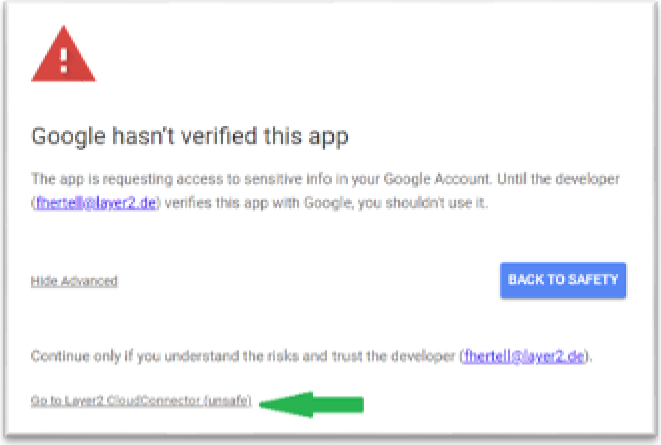
In case you do not add any authentication parameters to the connection string, a Google app named *Layer2 CloudConnector* which has been created by the Layer 2 GmbH is used to access the files in your GoogleDrive. As this app is currently not verified by Google, you will get the following warning when trying to refresh the authentication token:



Google hasn't verified this app

The app is requesting access to sensitive info in your Google Account. Until the developer (fhertell@layer2.de) verifies this app with Google, you shouldn't use it.

You must click “Advanced” and will get another warning:

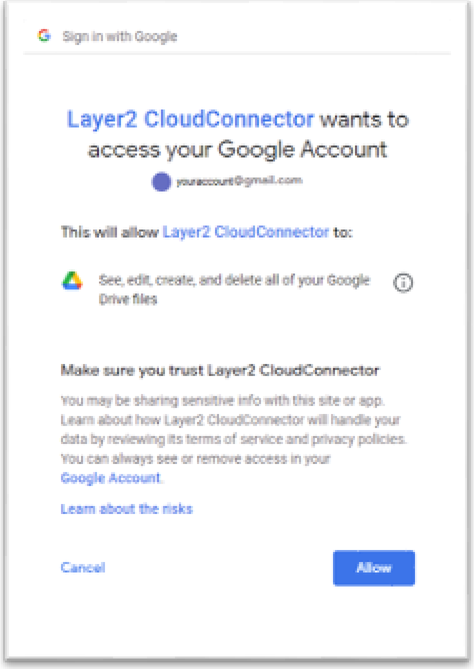


Google hasn't verified this app

The app is requesting access to sensitive info in your Google Account. Until the developer (fhertell@layer2.de) verifies this app with Google, you shouldn't use it.

Continue only if you understand the risks and trust the developer (fhertell@layer2.de).

If you click “*Go to Layer2 CloudConnector (unsafe)*”, you will get the consent dialog where you give the app the permission to access your files:



Sign in with Google

Layer2 CloudConnector wants to access your Google Account

youraccount@gmail.com

This will allow Layer2 CloudConnector to:

See, edit, create, and delete all of your Google Drive files

Make sure you trust Layer2 CloudConnector

You may be sharing sensitive info with this site or app. Learn about how Layer2 CloudConnector will handle your data by reviewing its terms of service and privacy policies. You can always see or remove access in your Google Account.

Learn about the risks

If you do not get the "*Advanced*" option to allow the unsafe app, please check your Google Account and allow unsafe apps or create your own custom Google app for authentication as described in the next section.

Authentication using your own custom Google app

You can use your own Google application for authentication:

- Go to the Google API Console
- If you have not yet created any API project, click *Select Project* to create one. Otherwise, select an existing project.
    - If you are creating a new project, enter the Project Name and click *CREATE*.
- After you created or selected a project, you will be redirected to the project dashboard.
- In the left-hand menu select *Credentials* under *the APIs & Services* section.
- Select the *OAuth Consent Screen* tab:
    - Specify the name of your application
    - Specify an email address as support email
    - Click *Next* which redirects you to the scopes page
    - Click *ADD OR REMOVE SCOPES* and add the Google Drive API scope *../auth/drive*
    - Click *SAVE AND CONTINUE* till the end and then go back to the Dashboard (you do not need to click PREPARE FOR VERIFICATION because you will be the only one using this app)
- Switch to the *Credentials* tab, click *the Create Credentials* drop-down and select *OAuth client ID*
- Choose *Desktop app* as *Application Type*, give it a name and click *CREATE*
- Click on the created ClientId and copy *Client ID* and *Client Secret*

Add Client ID and Client Secret of your Google application to the connection string (see below).

Connection String

Here is the full list of connection string parameters for the provider:

ClientId

By default, this parameter is not required and the Layer2 Cloud Connector app is used for authentication (see above). In case you prefer to use your own Google application, use this parameter to specify your application's Client Id.

ClientSecret

By default, this parameter is not required and the Layer2 Cloud Connector app is used for authentication (see above). In case you prefer to use your own Google application, use this parameter to specify your application's Client Secret.

User

This setting is not required. However, if you have multiple connections targeting different accounts, you need to specify the user in order to differentiate between them.

FilenameFilter

This is an optional search pattern for filtering files by name or extension. The parameter accepts standard windows file search patterns like these: * ?
Note that only one filter can be set with this parameter.

FileReadMode

This is an optional setting for specifying how to read files. Options are:

- **Recursive (Default):** Gets the content of all folder and subfolder recursively.
- **Flat:** Gets the files only in the specified folder excluding all subfolder contents.

IncludeFolders

This setting is optional. If it is not specified, the provider will read all folders as records for synchronization by default. If this is false, folders will not be treated as records, but will only be created implicitly through the file-paths.

IncludeDirectorySize

This setting is optional. If it is not specified, the provider will return 0 for all directory-sizes. This is due to optimization: To determine the size of a folder, it is necessary to traverse through the whole directory-tree and sum up the file-sizes. This can be very slow on a large number of files and is in many cases unnecessary. To retrieve the actual sizes, set this to "true".

Select Statement

The providers support a SQL-like select statement which is described in detail here: SQL-Like Select Statement

Hints and Limitations

GoogleDrive allows to have several files or folder with the same path and name in parallel and it even creates such duplicates in case of synchronization conflicts that occur in the Google Client. The Layer2 Cloud Connector does not support duplicate files and folders as these are not allowed by the other systems like the Windows file system or Dropbox.
Therefore, the Layer2 GoogleDrive Provider always synchronizes from/to the newest file or folder (determined by creation date) in case there are duplicates. The other versions are skipped which is logged in a warning.
In case a file or folder is deleted by the GoogleDrive provider during the synchronization, it is not permanently deleted but just moved to the trash.
The creation date of GoogleDrive is read-only as it cannot be changed via the GoogleDrive API.
In case you have more than one GoogleDrive connection and they are targeting different accounts, you need to differentiate between the connections by adding the *User* parameter with different values to the connection strings.

Layer2 Data Provider for Azure Blob Storage

The Layer2 Azure Blob Storage provider is available in Layer2 Cloud Connector Enterprise and connects to an Azure Storage account to retrieve files from within a blob storage container.
The provider is typically used to synchronize files with SharePoint or Office 365 / OneDrive for Business document libraries (including the files) or lists (metadata only). It supports both read and write.

Connection String

Here is the full list of connection string parameters for the provider:

DefaultEndpointsProtocol

This parameter is taken from the connection string which can be found under *Settings -> Access keys* in the Azure Storage Account.

AccountName

This parameter is taken from the connection string which can be found under *Settings -> Access keys* in the Azure Storage Account.

EndpointSuffix

This parameter is taken from the connection string which can be found under *Settings -> Access keys* in the Azure Storage Account.

AccountKey

This parameter is taken from the connection string which can be found under *Settings -> Access keys* in the Azure Storage Account. You can enter it in the connection string or alternatively in the password field below the connection string. If you entered it in the connection string, it will be moved to the password field on save.

FilenameFilter

This is an optional search pattern for filtering files by name or extension. The parameter accepts standard windows file search patterns like these: * ?

Note that only one filter can be set with this parameter.

FileReadMode

This is an optional setting for specifying how to read files. See hints and limitations for details. Options are:

- **Recursive (Default):** Gets the content of all folder and subfolder recursively.
- **Flat:** Gets the files only in the specified folder excluding all subfolder contents.

IncludeFolders

This setting is optional. If it is not specified, the provider will read all folders as records for synchronization by default. If this is false, folders will not be treated as records, but will only be created implicitly through the file-paths.

IncludeDirectorySize

This setting is optional. If it is not specified, the provider will return 0 for all directory-sizes. This is due to optimization: To determine the size of a folder, it is necessary to traverse through the whole directory-tree and sum up the file-sizes. This can be very slow on a large number of files and is in many cases unnecessary. To retrieve the actual sizes, set this to “true”.

Select Statement

The providers support a SQL-like select statement which is described in detail here: SQL-Like Select Statement

Hints and Limitations

Azure Blob Storage is a flat virtual file system without folder hierarchy. The folder hierarchy that is built in the UI is virtual and based on blob name prefixes. So, in case a blob is named “myPrefix/mySuffix”, it will be represented by a folder named “myPrefix” and a blob inside this folder with name “mySuffix”. Therefore, it is not possible to create empty folders in Azure Blob Storage.

The Layer2 Cloud Connector handles this as if the empty folder would exist in the blob storage. However, if you check your Azure Blob Storage container, it will not contain empty folders even if the Layer2 Cloud Connector reports and treats them as being created.

In case the connection string parameter *FileReadMode* is set to *Flat* after the initial sync, the full folder hierarchy in the Azure Blob container will still be provided (but without the files in sub folders).

The connection string parameter *IncludeFolders* should only be set to *false* before the initial sync. Otherwise, the Layer2 Cloud Connector will always provide all folders from the Azure Blob container.

Layer2 Data Provider for Microsoft Flow and Logic Apps

The Layer2 Microsoft Flow and Logic Apps provider connects to the Request trigger API of a Flow or an Azure Logic App. It is used to write inserted, changed, and deleted data (change notifications) to the Request trigger of the Flow or the Logic App after each synchronization. The change notifications contain information about changes in the source data entity that were made since the last synchronization run. The information is sent as a JSON payload and if the related JSON schema has been pasted to the Request trigger, it can be used in further actions or conditions of the flow.

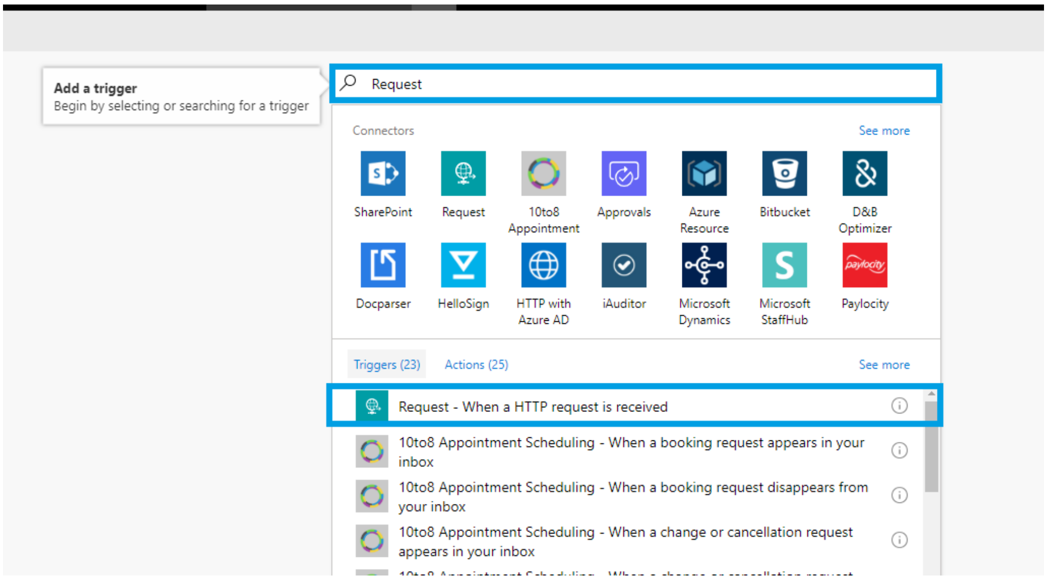
Change notifications can only be written to but not received from Flow or Logic Apps using the Layer2 Cloud Connector.

The Layer2 Microsoft Flow and Logic Apps provider does not manipulate any data but just writes change notifications to a Flow or Logic App to trigger the flow. Therefore, it is best practice to define the connection as uni-directional (although bi-directional does work but does not change anything).

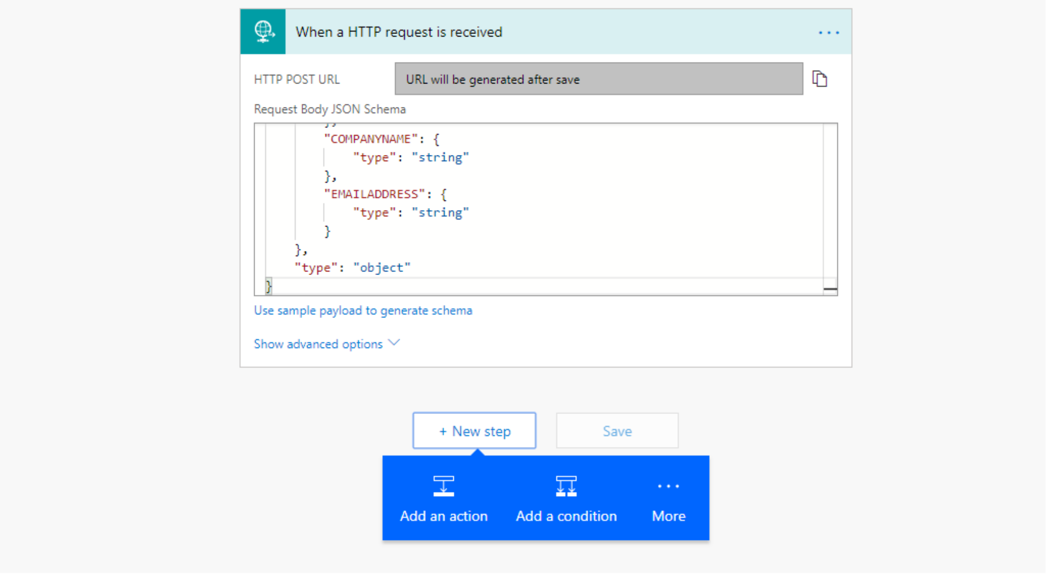
How to Setup a Microsoft Flow that is Triggered by the Layer2 Cloud Connector

To use this provider to trigger a Flow, you just need to sign up with an email address. To create a Flow that receives the Change Notifications from the Layer2 Cloud Connector, follow these steps:

1. On the Flow landing page, sign in (or sign up, if not yet done).
2. Click "My flows" on the top menu bar.
3. On the "My flows" page, click "Create from blank"
4. The first item you need to create is the trigger. This is the event that will start your Flow. Click on "Search hundreds of connectors and triggers", search for and select the "Request" trigger:


Add a trigger
Begin by selecting or searching for a trigger
Request
Connectors
See more
SharePoint
Request
10to8 Appointment
Approvals
Azure Resource
Bitbucket
D&B Optimizer
Docparser
HelloSign
HTTP with Azure AD
iAuditor
Microsoft Dynamics
Microsoft StaffHub
Paylocity
Triggers (23)
Actions (25)
See more
Request - When a HTTP request is received
10to8 Appointment Scheduling - When a booking request appears in your inbox
10to8 Appointment Scheduling - When a booking request disappears from your inbox
10to8 Appointment Scheduling - When a change or cancellation request appears in your inbox

5. Paste the JSON schema from the Layer2 Cloud Connector (explained in section [Create and Get the JSON Schema]() below) into the **Request Body** field:



When a HTTP request is received

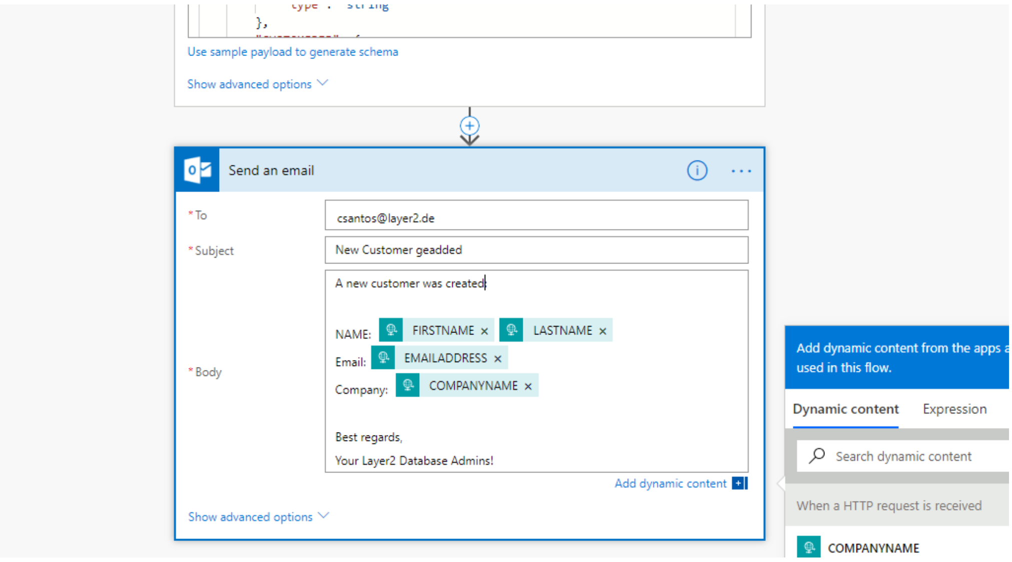
HTTP POST URL: URL will be generated after save

Request Body JSON Schema

```
        "COMPANYNAME": {
            "type": "string"
        },
        "EMAILADDRESS": {
            "type": "string"
        }
    },
    "type": "object"
}
```

Figure 46- Example request body JSON Schema in the API UI

6. Create a new step and set the information for your notification email :



Send an email

* To: csantos@layer2.de

* Subject: New Customer geadded

* Body:

A new customer was created

NAME: FIRSTNAME × LASTNAME ×

Email: EMAILADDRESS ×

Company: COMPANYNAME ×

Best regards,

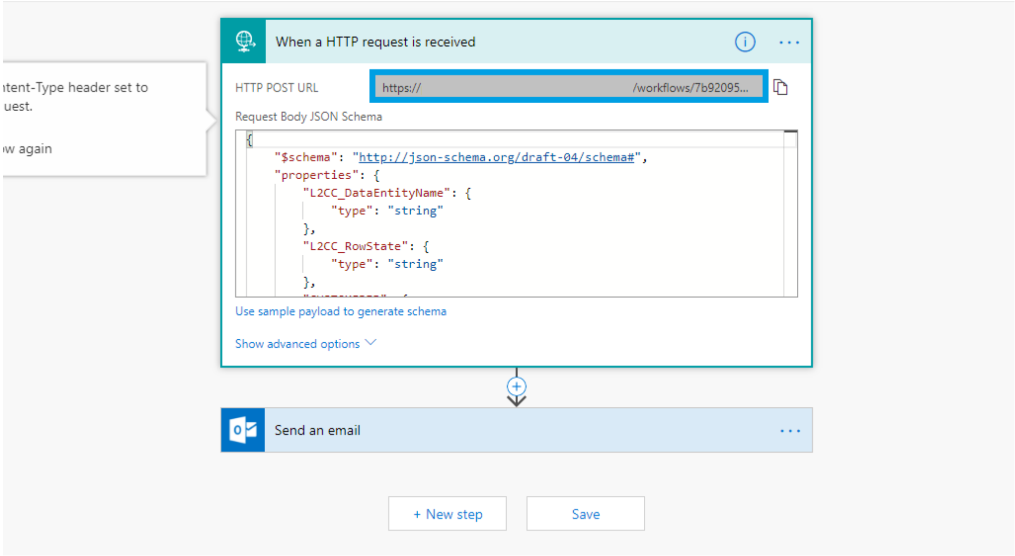
Your Layer2 Database Admins!

Use sample payload to generate schema
Show advanced options
Add dynamic content
Show advanced options
Add dynamic content from the apps a used in this flow.
Dynamic content
Expression
Search dynamic content
When a HTTP request is received
COMPANYNAME

Figure 47 - UI for adding Send an Email action

As you can see, the fields from the JSON schema are shown to be used (shown at the right-hand side).

7. Add a name for the Flow (at the top).
8. Save the Flow and copy the URL that has been created for the trigger



```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "properties": {
        "L2CC_DataEntityName": {
            "type": "string"
        },
        "L2CC_RowState": {
            "type": "string"
        },
```

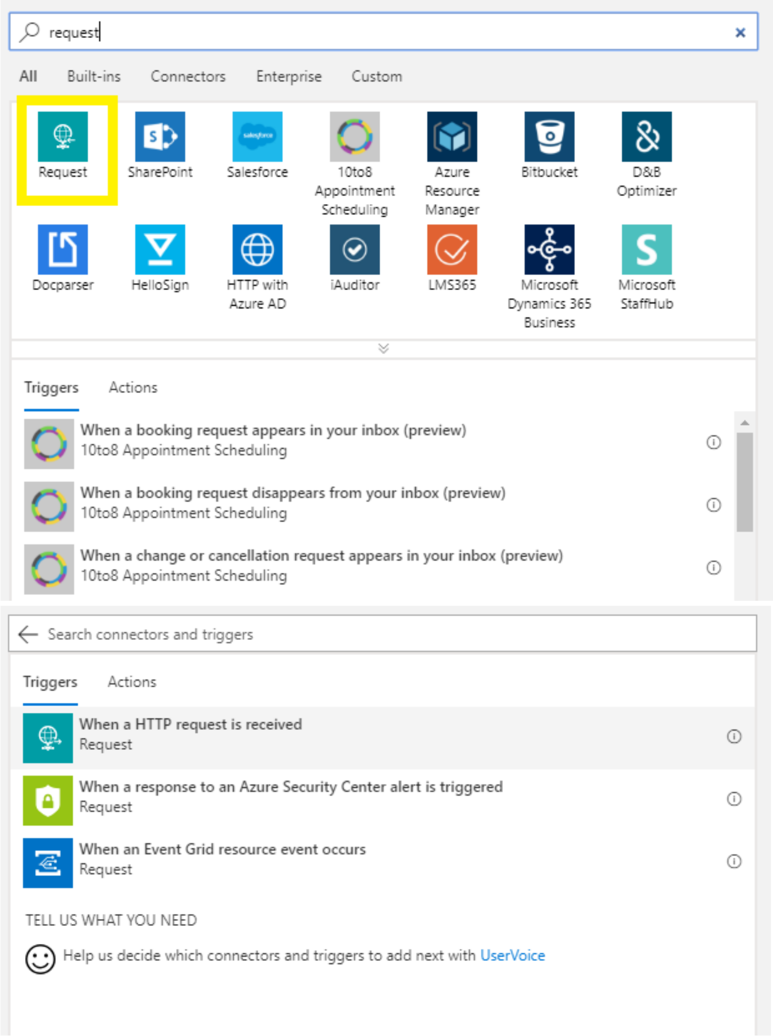
Figure 48 - Example Flow Trigger URL

How to Setup an Azure Logic App that is Triggered by the Layer2 Cloud Connector

To use this provider to trigger a Logic App, you need an Azure subscription. To create a Logic App that receives the Change Notifications from the Layer2 Cloud Connector, follow these steps:

1. On the Azure portal dashboard, select **New**.
2. In the search bar, search for 'logic app', and then select **Logic App**. You can also select **New, Web + Mobile**, and select **Logic App**.
3. Enter a name for your Logic App, select a **Location**, **Resource Group**, and then click **Create**. If you select Pin to Dashboard the Logic App will automatically open once deployed.
4. After opening your Logic App for the first time you can select from a template to start. For now, click **Blank Logic App** to build a new Logic App.

5. The first item you need to create is the trigger. This is the event that will start your Logic App. Select the “Request” trigger:



request

When a booking request appears in your inbox (preview)
10to8 Appointment Scheduling

When a booking request disappears from your inbox (preview)
10to8 Appointment Scheduling

When a change or cancellation request appears in your inbox (preview)
10to8 Appointment Scheduling

Search connectors and triggers

When a HTTP request is received
Request

When a response to an Azure Security Center alert is triggered
Request

When an Event Grid resource event occurs
Request

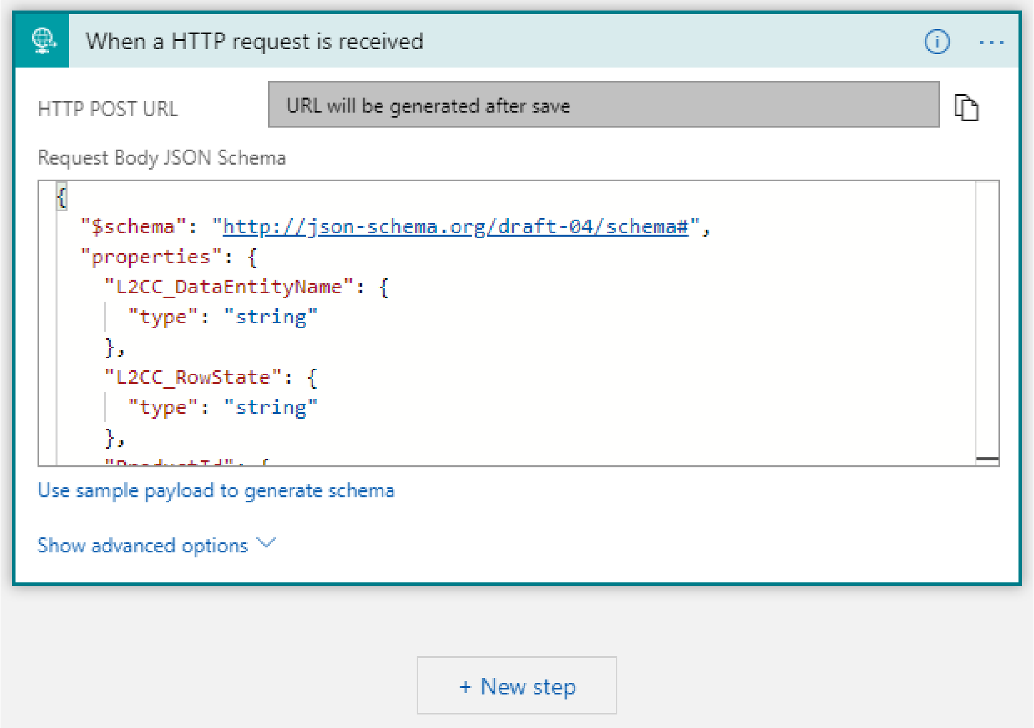
TELL US WHAT YOU NEED

Help us decide which connectors and triggers to add next with UserVoice

6.

Figure 49- Logic API triggers, with 'Request' highlighted

7. Paste the JSON schema from the Layer2 Cloud Connector (explained in section Create and Get the JSON Schema below) into the **Request Body** field:



When a HTTP request is received

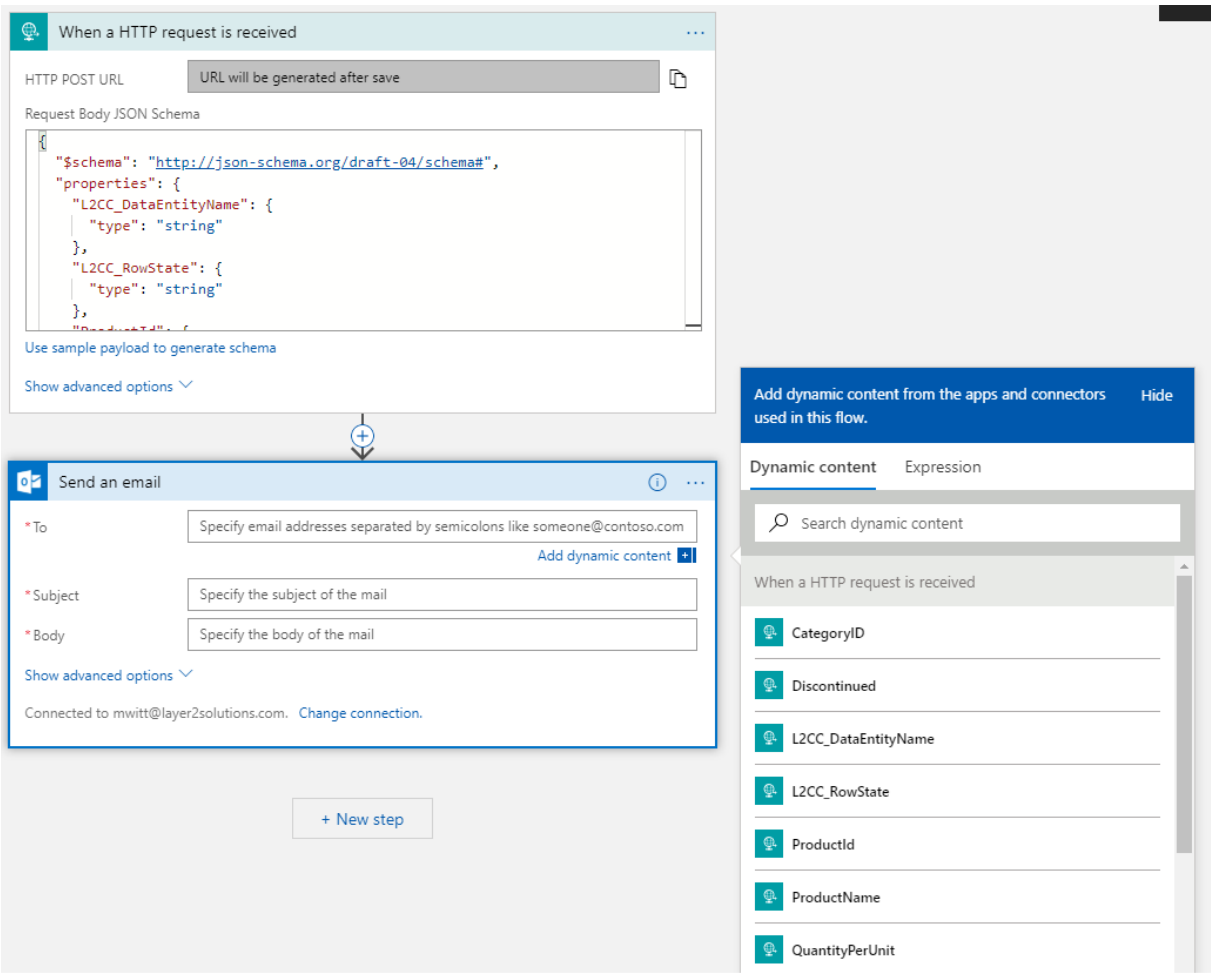
HTTP POST URL: URL will be generated after save

Request Body JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "properties": {
    "L2CC_DataEntityName": {
      "type": "string"
    },
    "L2CC_RowState": {
      "type": "string"
    },
```

Figure 50- Example request body JSON Schema in the API UI

8. Add an action (for example "Send an email") and fill in the required fields:



When a HTTP request is received

HTTP POST URL: URL will be generated after save

Request Body JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "properties": {
    "L2CC_DataEntityName": {
      "type": "string"
    },
    "L2CC_RowState": {
      "type": "string"
    },
```

Send an email

*To: Specify email addresses separated by semicolons like someone@contoso.com

*Subject: Specify the subject of the mail

*Body: Specify the body of the mail

Connected to mwitt@layer2solutions.com. Change connection.

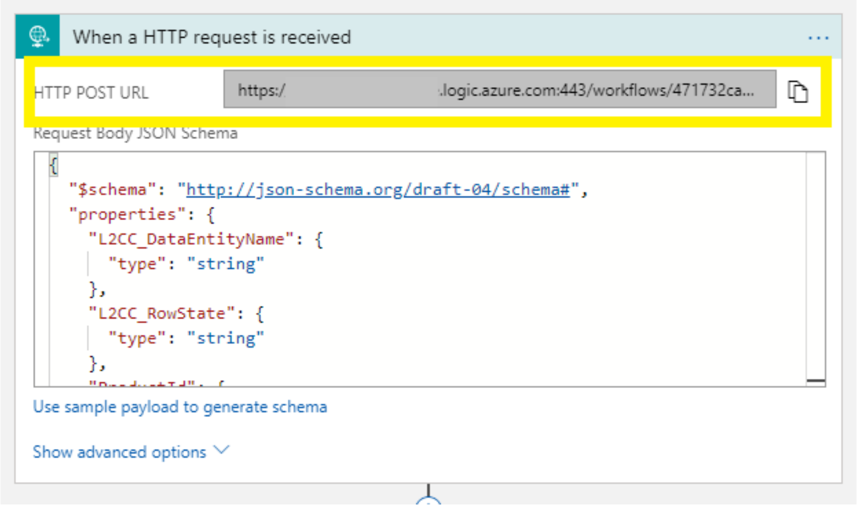
Add dynamic content from the apps and connectors used in this flow.

When a HTTP request is received: CategoryID, Discontinued, L2CC_DataEntityName, L2CC_RowState, ProductId, ProductName, QuantityPerUnit

Figure 51 - UI for adding Send an Email action

As you can see, the fields from the JSON schema are shown to be used (shown at the right-hand side).

9. Save the Logic App and copy the URL that has been created for the trigger:



When a HTTP request is received

HTTP POST URL https:/ .logic.azure.com:443/workflows/471732ca...

Request Body JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "properties": {
    "L2CC_DataEntityName": {
      "type": "string"
    },
    "L2CC_RowState": {
      "type": "string"
    },
    "ProductId": {
```

Use sample payload to generate schema

Show advanced options

Figure 52 - Example Logic App Trigger URL

Change-Notification Payload

Each of the notifications that will be sent contain information about a changed record in the source system. The information is sent as JSON payload and the Layer2 Cloud Connector provides a feature to get and copy the JSON schema of the notification to paste it into the Flow's or Logic App's **Request** trigger.

Connection String

A typical connection string for the Flow and Logic Apps provider looks like this:

WebHookUrl=<url of the flow or logic app trigger>

Here is the full list of connection string parameters for this provider:

WebHookUrl

The WebHookUrl is created in the Flow or Logic App you want the Layer2 Cloud Connector to send notifications to.
After saving the Flow or Logic App, the URL of the Request trigger action will be displayed (see above). This URL is the Webhook, and you need to use this URL in the connection string of the Layer2 Cloud Connector.

Note: The Webhook URL will validate even if it is incorrect as it cannot be checked without sending a notification to post.

SkipNotifications

In case of an initial sync or in case of any data updates that should not be sent to the Flow or Logic App, you can specify the parameter *SkipNotifications=true*. The default value is “false”, so that change notifications for inserted, deleted, and updated records are sent. However, you can suppress them using this connection string parameter, if desired.

Primary Key(s)

You must specify the field(s) that are used to identify the records on the target (Flow or Logic App) side.

Select Statement

The select statement defines which fields are offered for mapping and of what data type they are.

The format for the select statement is a comma-separated list of the fields and optionally their data types (shown in brackets, because of being optional):

```
SELECT FIELDNAME1[:DATATYPE], FIELDNAME2[:DATATYPE]
```

Datatypes that can be specified are the following:

- integer
- long
- float
- decimal
- double

The above datatypes will be sent as JSON type *number* in the notification.

- string
- byte
- datetime

The above datatypes will be sent as JSON type *string* in the notification.

- boolean

The above datatype will be sent as JSON type b*oolean* in the notification.

If no datatype is specified, the default type *string* will be used.

Example Select Statement:

Select Id:integer, CreditLimit:float, Name, ExistingCustomer:boolean

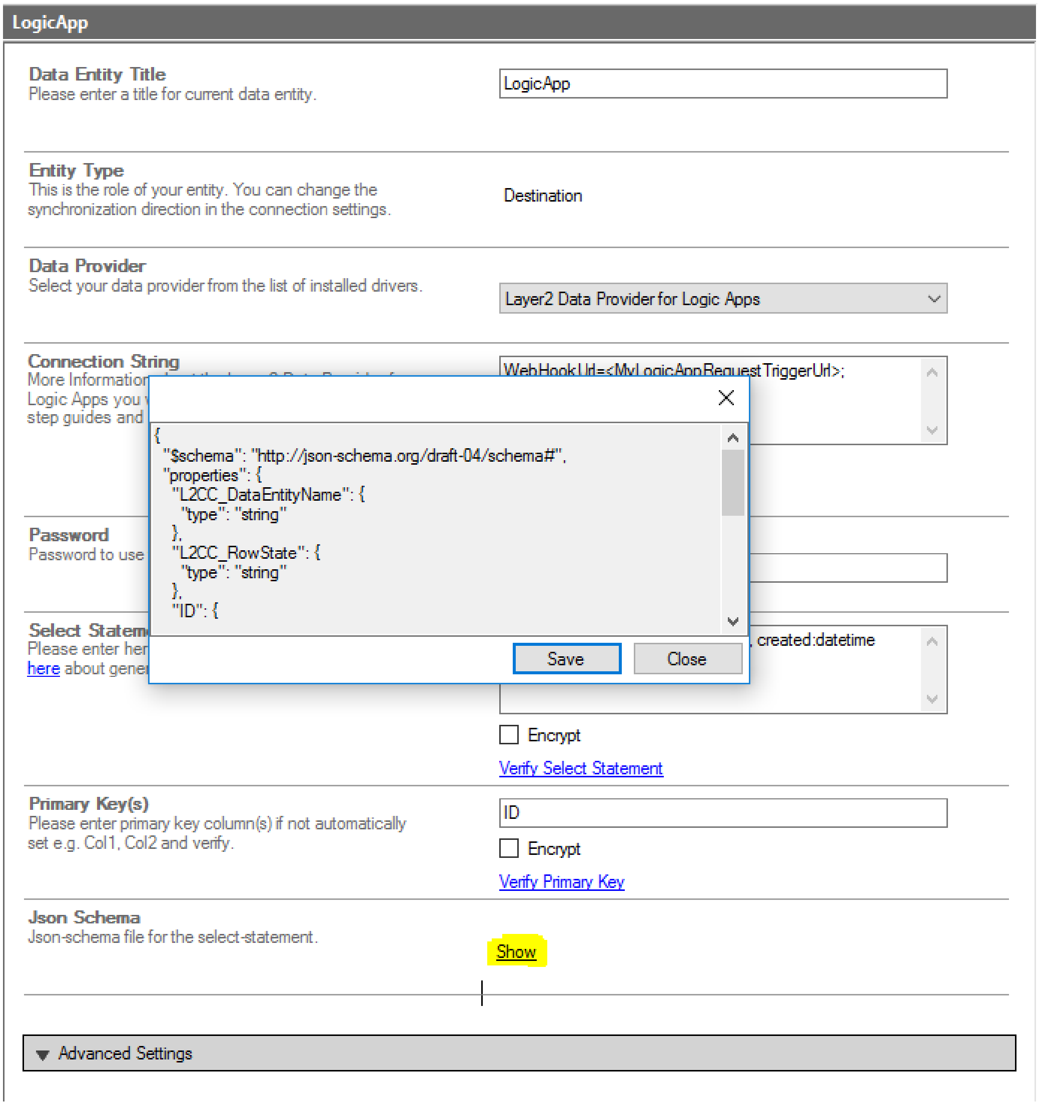
In this example, the field “Name” will be of type string, as no explicit datatype has been specified.

Fields must be explicitly defined; the wildcard * is not supported. In case no data type has been defined, the default of *string* is used.

Note: Only changes for fields that are mapped will be sent with the change notification. So, in case a field of a record in your business system has changed and this field is not mapped (and therefore not part of the synchronization), the change will not trigger a change notification.

Create and get the JSON Schema

In the data entity with the Flow and Logic Apps provider, there is a section **Json Schema**. After clicking “Show” (marked yellow in the image below), the schema is created based on the select statement and is shown in a popup window. You can now copy the schema or save it to disk. This is the schema you need to paste into the JSON Schema Body of the Flow or Logic Apps Request Trigger (see How to setup an Azure Logic App that is Triggered by the Layer2 Cloud Connector).



LogicApp

Data Entity Title
Please enter a title for current data entity.

LogicApp

Entity Type
This is the role of your entity. You can change the synchronization direction in the connection settings.

Destination

Data Provider
Select your data provider from the list of installed drivers.

Layer2 Data Provider for Logic Apps

Connection String
More Information
Logic Apps you
step guides and

WebHookUrl=<MyLogicAppRequestTriggerUrl>;

```
{
 "$schema": "http://json-schema.org/draft-04/schema#",
 "properties": {
  "L2CC_DataEntityName": {
   "type": "string"
  },
  "L2CC_RowState": {
   "type": "string"
  },
  "ID": {
```

Password
Password to use

Select Statem
Please enter her
here about gener

created:datetime

☐ Encrypt

Verify Select Statement

Primary Key(s)
Please enter primary key column(s) if not automatically set e.g. Col1, Col2 and verify.

ID

☐ Encrypt

Verify Primary Key

Json Schema
Json-schema file for the select-statement.

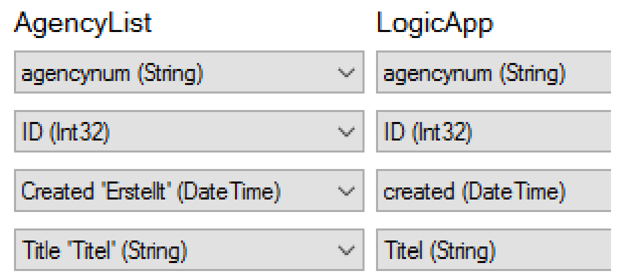
Show

Figure 53 - Show link for the Json Schema option for the Flow and Logic App Provider

Example:

Synchronization from an Agency list in SharePoint to a Logic App.

Connection string	WebHookUrl=<MyLogicAppRequestTriggerUrl>;
Select statement	select ID:integer, agencynum, Titel, created:datetime

Mapping	

AgencyList	LogicApp
agencynum (String)	agencynum (String)
ID (Int32)	ID (Int32)
Created 'Erstellt' (DateTime)	created (DateTime)
Title 'Titel' (String)	Titel (String)

Example of the resulting change notification that is sent to the Logic App for a changed record – L2CC_DataEntityName and L2CC_RowState are always sent:

```
{
  "headers": {
    "Connection": "Close",
    "Expect": "100-continue",
    "Host": "prod-05.westeurope.logic.azure.com",
    "Content-Length": "307",
    "Content-Type": "application/json"
  },
  "body": {
    "L2CC_DataEntityName": "AgencyList",
    "L2CC_RowState": "added",
    "ID": 225,
    "agencynum": "00000001",
    "Titel": "Nummer1",
    "created": "13.06.2016 13:46:58"
  }
}
```

Possible values for L2CC_RowState are “added”, “deleted”, and “modified” – the row state can be used to determine which step to execute next in the Logic App.

You can find a detailed example for the whole setup process in Appendix A. See Start an Azure Logic Apps Workflow on Local XML Data Changes section.
Note: The configuration and process to trigger a Flow is quite similar, just the UI is a little bit simpler for Flow.

Layer2 Data Provider for Microsoft Teams Channel Webhook

The Layer2 Microsoft Teams Channel Webhook provider connects to Webhooks of Microsoft Teams channels so that it can write a "Chat entry" to it after each synchronization. The chat entry contains information about changes in the source data entity that were made since the last synchronization run. It can, for example, be used to regularly notify team members about changes that have been made in a certain local or cloud-based business system like SQL, ERP/CRM, etc., as supported by the Layer2 Cloud Connector. As the chat entries are written into specific channels, it is possible to target change notifications to special interest groups that watch specific channels.

The Layer2 Data Provider for Microsoft Teams Channel Webhook cannot be used to sync documents in a Microsoft Teams document library. However, you can view the Team files in SharePoint and use of the Layer2 Data Provider for SharePoint (Libraries and Lists) to synchronize files to this library.

Chat entries can only be written to but not read from Microsoft Teams Channels using the Layer2 Cloud Connector.

The Layer2 Microsoft Teams Channel Webhook provider does not manipulate any data but just writes chat entries to a Microsoft Teams channel. Therefore, it is best practice to define the connection as uni-directional (although bi-directional also works).

Chat Entry Reference

The chat entry that will post the changes contains the following elements:

Summary	The summary contains information about the changes that were made since the last synchronization run. The format is: *x items inserted, y items modified, and z items removed in <Office 365 data entity name[1]>.*
Section	A section is created per item that has been changed. The title gives information about the data record and the kind of change. The format will be similar to the following: *Item '<record-key>' was added to '<Office 365 data entity name>'.*

[1] No data is changed in the Microsoft Teams (just a chat entry sent), however the team is treated as target data entity and "as-if" data had been changed there during the synchronization.

Section-Details	Each section can contain detailed information about the changes made to the data record. The information is shown as table of name/value pairs, showing the mapped fields (see below) and their new values.

Connection String

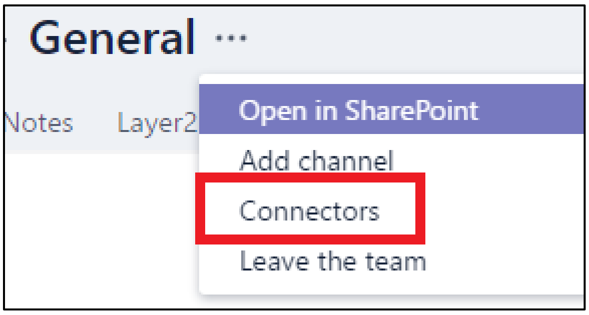
A typical connection string for the Microsoft Teams Channel Webhook provider looks like this:

WebHookUrl=<url of the webhook to the Microsoft Team>;

Here is the full list of connection string parameters for this provider:

WebHookUrl

The WebHookUrl is created in the Microsoft Teams channel you want the Layer2 Cloud Connector to send chat entries to.
When opening the Microsoft Teams channel, you can find an option at the top menu bar called **Connectors**. Click on that to start configuring your Webhook Connector.



You need to create a connector by choosing the "Incoming Webhook" Connector and then click **Add**.

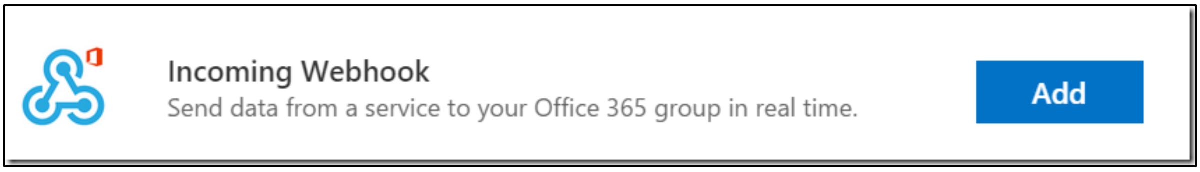

Incoming Webhook
Send data from a service to your Office 365 group in real time.
Add

Figure 54 – "Incoming Webhook" Connector in Office 365 Groups

To configure the Incoming Webhook, you need to give it a name and optionally add an image for the Connector. Once that is done you need to click on **Create** to create the Webhook. When it is created, you will see an input box with a URL in it. This is the Webhook and you need this URL in the connection string of the Layer2 Cloud Connector. The URL is unique to this Team and Webhook. If you remove the Webhook, you will disable it.

Note: The Webhook URL will validate even if it is incorrect as it cannot be checked without sending a card to post.

SkipDetails

When this parameter is set to “true”, no sections are created, but just the Summary information. This is useful if many data records have been changed (for example, due to a mass update in your business system) and you do not want to get every change detail but just the summary of the synchronization This parameter is optional and defaults to “false”.

BatchSize

This parameter is optional and defaults to 10. This means that up to 10 changes are listed in one single chat entry. So, in case you have 32 changes, the Layer2 Cloud Connector would send 4 change chat entries (3 entries with 10 items listed each and another one with 2 items listed).

> **Note**:
> In case you receive an error like this: “*Error while writing data to entity 'Target': Could not write Office 365 card. Details: Target responded with 'Microsoft.Griffin.Connectors.Providers.Common.MessageDelivery.MessageDeliveryException : RequestEntityTooLarge*” during the sync, try to reduce the batch size.

Select Statement

The select statement defines which fields are offered for mapping and, if mapped, what will be shown in the section details.

The format for the select statement is this, where `[FIELDS]` is a comma-separated list of the fields:

```
SELECT [FIELDS]
```

Fields must be explicitly defined; the wildcard * is not supported.

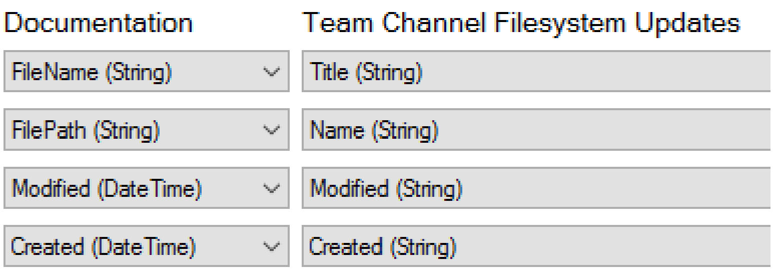
Note: Only changes for fields that are mapped will be tracked. So, in case a field of a record in your business system has changed and this field is not mapped (and therefore not part of the synchronization), the change will not be tracked in the chat entry at all.

Primary Key(s)

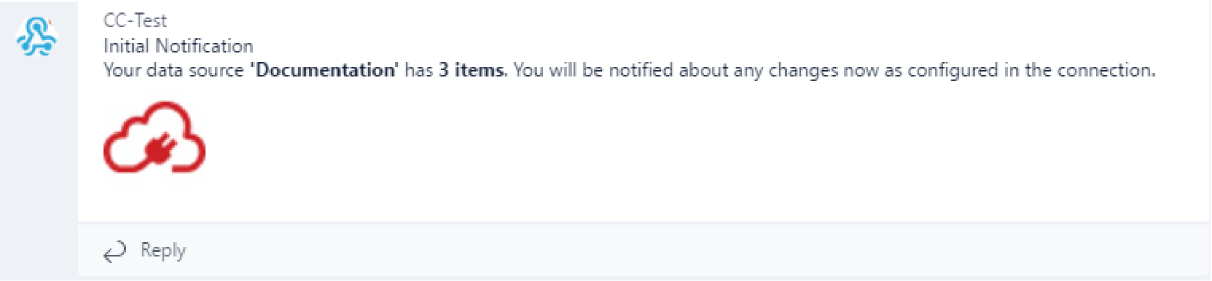
You need to specify the field(s) that are used to create the Section title. If using a GUID- or counter-type of primary key, it might be a good idea to add another detailed column (for example, “Agencynum, Title”) to have human-readable information in the card.

Example:

Synchronization from a file system to Microsoft Teams Channel

Connection string	WebHookUrl=<MyWebHookUrl>;
Select statement	select Name, Title, Created, Modified
Primary Key(s)	Name
Mapping	 Documentation → Team Channel Filesystem Updates FileName (String) → Title (String) FilePath (String) → Name (String) Modified (DateTime) → Modified (String) Created (DateTime) → Created (String)

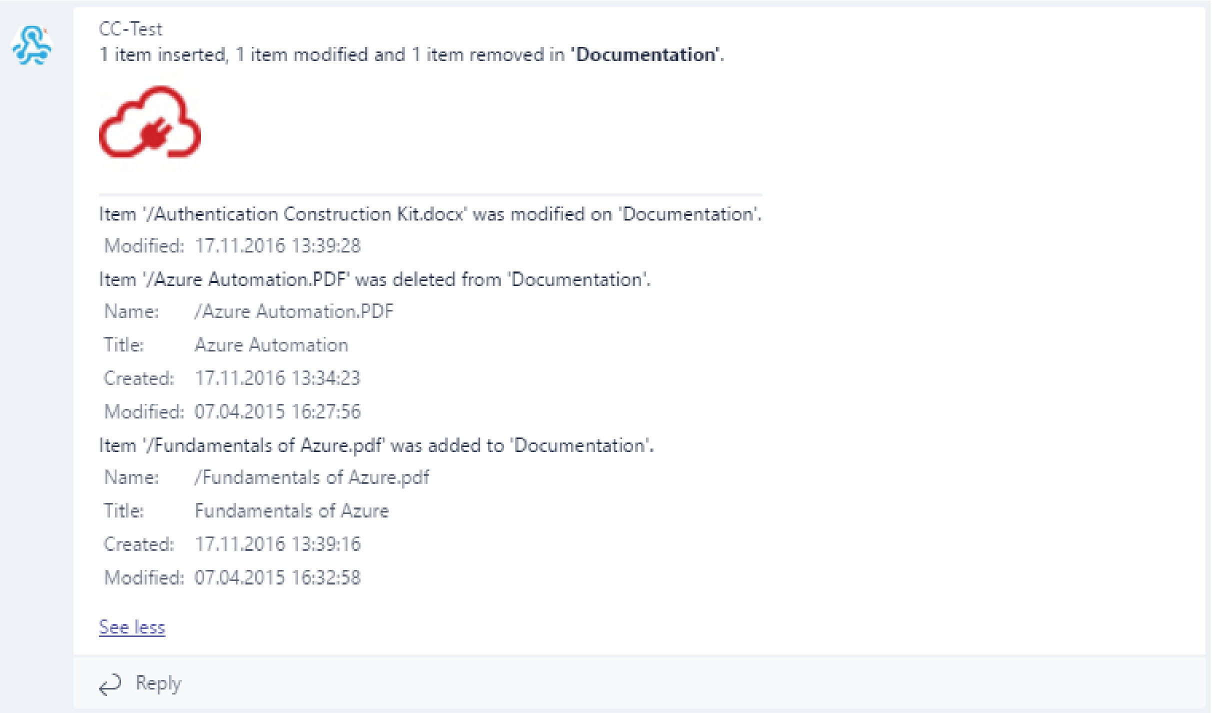
Example resulting Chat Entry for the initial synchronization:



CC-Test
Initial Notification
Your data source '**Documentation**' has **3 items**. You will be notified about any changes now as configured in the connection.

Figure 55 - Example chat entry for initial synchronization

Example resulting Chat Entry for subsequent synchronization:



CC-Test

1 item inserted, 1 item modified and 1 item removed in '**Documentation**'.

Item '/Authentication Construction Kit.docx' was modified on 'Documentation'.

Modified: 17.11.2016 13:39:28

Item '/Azure Automation.PDF' was deleted from 'Documentation'.

Name: /Azure Automation.PDF
Title: Azure Automation
Created: 17.11.2016 13:34:23
Modified: 07.04.2015 16:27:56

Item '/Fundamentals of Azure.pdf' was added to 'Documentation'.

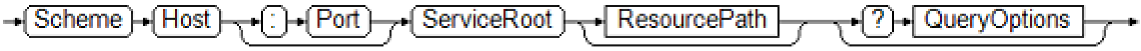
Name: /Fundamentals of Azure.pdf
Title: Fundamentals of Azure
Created: 17.11.2016 13:39:16
Modified: 07.04.2015 16:32:58

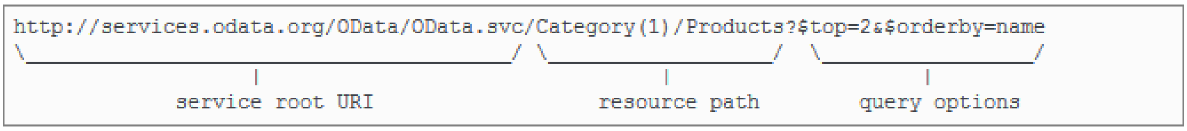
Figure 56 - Example chat entry after a synchronization with changes

Layer2 Data Provider for OData

The Layer2 OData provider retrieves data from OData sources. The Open Data Protocol (OData) is a Web protocol for querying and managing data. OData is being used to expose and access information from a variety of sources including, but not limited to, relational databases, file systems, content management systems, and traditional web sites. Examples of popular applications supporting OData are Microsoft Dynamics, SAP via NetWeaver, Microsoft SharePoint, and Visual Studio TFS. Also, MS-SQL databases might be populated through this protocol.

OData requests are presented as a single URL. The parts of an OData URI are separated to different fields, e.g., the select statement of the OData provider refers to the Query Options section of an OData URL.





```
http://services.odata.org/OData/OData.svc/Category(1)/Products?$top=2&$orderby=name
\______________________________________/ \__________________/  \_________________/
                  |                               |                    |
          service root URI                  resource path        query options
```

Figure 57 - OData URI construction

OData Specifications V1-V4

There are currently four versions of the OData-specification. The full specifications can be found at http://www.odata.org.

Formats – ATOM and JSON

The OData-protocol can provide the data in either the *Atom Syndication Format (ATOM)* or *JSON*. Both formats are supported. If a server also supports both formats, JSON will be given preference.

Specifying the URL

The URL which is specified for the provider can point to the root of an OData-service or to an OData-collection directly. In the latter case, the provider will determine the root URL automatically from the collection-response.

Selecting Data

Select statements can be defined by using either the OData query syntax or a SQL-style select statement.

The OData query syntax is the style that uses query options normally added as part of an OData URL, which start after the question mark as defined in the specification. The keywords `$value`, `$inlinecount`, `$count`, and `$format` are not supported.

The SQL-style query is a query using the SQL-like-syntax:

```
SELECT [fields] FROM [source] WHERE [filter]
```

Note that the fragments of a complex filter expression must be bracketed, so for example instead of:

```
$filter=CustomerTypeCode/Value ne 9 and CustomerTypeCode/Value ne null and
StateCode/Value eq 0
```

you need to use:

```
$filter=(CustomerTypeCode/Value ne 9) and (CustomerTypeCode/Value ne null)
and (StateCode/Value eq 0)
```

The provider will translate the query to a corresponding OData query. The source is OData collection parameter that was used in the connection string. If you use the SQL-style query, you do not need to define the "Collection" parameter in the connection string (as it will be in the select statement).

The filter expression supports all functions (like `endswith()`, `startswith()`, `indexof()`, etc.) that can be used in an OData query. Also, the SQL TOP keyword is supported. ORDER BY clauses are not supported, but also not necessary since the order of records does not affect the synchronization.

Note that the systems being queried may not support all functions in the OData standard; check their OData query documentation for more information.

Batching and Paging

The OData provider supports the batching of operations sent to an OData-service as well as the paging of results returned. Paging is always driven by the server, which decides how many records it will return in one page and the provider will read all the returned pages accordingly.
When the provider sends operations (such as inserts, updates, and deletes) to the OData-service, it will, by default, attempt to send the operations as a batch to the OData batch-endpoint `$batch`. If the request fails because the server does not provide a batch endpoint, all operations will be sent one-by-one. By default, the provider will create batches of 500 items. The batch-size can be configured by specifying the "*BatchSize=;*" parameter in the connection string.

Entity-Inheritance

The provider supports inheritance of entity- and complex-types defined in the metadata-description of the OData-service.

Specifying the Entity-Type

In most cases, the provider can automatically detect the entity type. There are some rare cases when this is not possible:

- The URL points to a collection (i.e., has not been specified explicitly with the "*Collection=;*"property)
- The format is ATOM
- The data source is empty

In this case, the provider will return an error stating that the entity-type needs to be specified explicitly. Specifying the entity-type can also be useful, especially if the collection does contain different types of entities.

The Update HTTP-Verb

There are different HTTP-verbs which both are frequently used to indicate an update to an OData-service: MERGE and PATCH. The provider supports both verbs and tries to detect automatically which one is expected by the server. By default, it tries to use the MERGE-verb for services with OData versions V1-V3 and the PATCH-verb for services with version V4. If the default verb is rejected, the provider will use the alternative. To prevent this additional step, the verb can also be defined explicitly as part of the connection-string.

The Entity Tag (etag)

The entity tag is a HTTP-header which specifies the version of a HTTP-resource. It is commonly used to manage caching of web-resources. Some OData-services provide an etag-header and expect that the provided tag is sent on subsequent update-requests, which enables the server to make sure the entity has not been modified since the last read-request.
The entity-header is populated as a read-only field in the result table.

Reading SharePoint Lists

Microsoft SharePoint also provides access to all content (Webs, Libraries, Lists, etc.) via OData. The SharePoint OData endpoint can be found at siteCollection/_api. If, for example, the site collection is at https://mySharepoint.com/siteCollection, the OData-endpoint would be at https://mySharepoint.com/siteCollection/_api. To access list or library contents the URL would be: https://mySharepoint.com/siteCollection/_api/web/lists('8E6C08E4-AD36-470D-AB8B-276306D88A10')/items, for a list with the given ID.

Reading from SAP NetWeaver

SAP NetWeaver requires a CSRF token header to be initially fetched and sent with all subsequent data modification requests (insert, update, delete). To enable support for the CSRF token, the keyword *SAPCsrfHeader* must be added as part of the authentication type in the connection string (for example:

```
Authentication=Windows,SAPCsrfHeader;
```

This means that Windows authentication is used and the CSRF token is supported).

Navigation Properties

OData entities can have properties referencing other OData entities. Those are called navigation properties. Creating a new entity for them is called deep insert and is currently not supported.

Specifying an existing entity as a navigation property during an insert operation or changing the referenced entity during an update operation can be done with reference fields. These are automatically generated by the Layer2 Cloud Connector and follow this pattern: Reference_*PROPERTYNAME* (e.g., `Reference_Supplier` for a product entity with the navigation property `Supplier`)

The value of these fields will be the key values for the referenced entity. For example, `Reference_Supplier` could be `1` if suppliers have an integer key property. If an entity has multiple key properties, the values are comma-separated with fieldname=value.

Example:

```
Area=North,Identifier=12
```

Please note: This feature only works with single value navigation properties and services following the V4 OData-specification. It's also currently not possible to set no reference (e.g., `Reference_Supplier=NULL`).

Connection String

A typical connection string for the OData provider looks like this:

Url=<DataServiceRootUri>; Collection=<ResourcePath>; Authentication=<AuthenticationType>;

or

Url=<FullCollectionUri>; Authentication=<AuthenticationType>;

Here is the full list of connection string parameters for this provider:

URL

This can be either the service root URL for the OData source or the full URL to an ATOM or JSON representation of an OData-collection. This parameter is **mandatory**.

Collection

This optional parameter defines the OData-collection that is to be accessed as a URL relative to the root of the OData-service, which in many cases is just the name of the collection.
If the URL defined with the URL connection-string parameter already points to a collection or if the collection is specified in the FROM-part of a SQL-style select, this parameter should not be specified.

Authentication

This parameter specifies how the OData provider authenticates against the target server. See the Authentication section for details. In case of an SAP Netweaver connection, the authentication parameter needs to also contain the *SAPCsrfHeader* keyword.
This parameter is optional.

User

This part of the connection string specifies the username for the account which is used to authenticate against the server. It needs to be specified for several authentication methods. See the Authentication Methods section for details.

Password

This defines the password for the account which is used to authenticate. It needs to be specified for several authentication methods by typing it into the Connection String or Password field. See the Authentication Methods section for details.

EntityType

This parameter can be used to explicitly define the type of the OData-entity, either if it could not be determined automatically or if the collection can contain multiple different entity-types.
This parameter is optional.

BatchSize

Specifies the number of operations which will be sent in a single batch. The default is 500.
This parameter is optional.

Batching

Can be used to explicitly turn batching of operations sent to the OData-server off or on. It accepts the values “Auto”, “True”, or “False”. If it is not specified, it defaults to auto. In auto mode, operations will be sent as a batch to the OData *$batch*-endpoint and, if that fails, all operations will be sent one-by-one in single requests.
If the server does not support batching, it can be explicitly turned off with *Batching=False*, to prevent the provider from executing the failing batch-request.
On the other hand, the provider might try to commit operations in single requests, even if batching is supported, if the batch-request fails due to an unrelated error.
In this case, batching can be explicitly turned on with *Batching=True*, to prevent the provider from unnecessarily retrying with single-requests.
This parameter is optional.

UpdateVerb

Defines the verb, which should be used to indicate update-operations to the OData-server. This can be any text, but typically only *MERGE*, *PATCH* or *Auto* would be used.
The default is *Auto*, which will first try the HTTP/MERGE verb for OData V1-V3 and fallback to HTTP/PATCH if that fails. For OData V4 it will first try HTTP/PATCH and then fallback to HTTP/MERGE.
To prevent the provider from performing unnecessary failing requests, the verb can be defined explicitly with this parameter.
This parameter is optional.

IEEE754Compatible

This parameter specifies if the JSON format that is sent to the service should be IEEE754 compatible. The default value is “auto” which will first try “ExplicitOn” and then “ImplicitOn”. If the service accessed is not IEEE754-compatible, this parameter should be set to “ExplicitOff”. If the service accessed cannot handle the IEEE754 content type argument, but still expects IEEE754-compatible content, this parameter should be set to “ImplicitOn”. If the service accessed is not IEEE754-compatible and cannot handle the content type argument, this parameter should be set to “ImplicitOff”.
This parameter is optional.

DateTimeFormat

This parameter specifies the JSON format that is used to send dates. The default-value is *yyyy-MM-ddTHH:mm:ssZ* (for example, *2016-01-01T10:10:10Z*). For connections using the SAPCsrfHeader authentication, the default-value is *yyyy-MM-ddTHH:mm:ss*.
This parameter is optional.

Timeout

This parameter allows for timeout values to be specified for situations where the default timeout is not long enough for the data pull. It is defined in seconds. This parameter is optional.

Layer2 Data Provider for Office 365 Fast File Sync

This provider connects to Microsoft SharePoint Online (Office 365 or OneDrive for Business) or SharePoint 2016 On-Premises to access files and folders on SharePoint document libraries in a way which is optimized for fast file uploading. The provider does not support a select statement, although querying can be done by setting up an appropriate SharePoint library view.

Connection String

A typical connection string for this provider looks like this:
Url=<SharePointSiteUrl>; Library=<LibraryName>; Authentication=<auth>
Here is the full list of connection string parameters for this provider:

Url

This is the URL of the SharePoint site-collection where the document library is located or the full URL to the view in the SharePoint web-site. When a full URL is used then the library, folder and view information is automatically retrieved from it. Otherwise, the library name must be specified separately. The URL is mandatory.

Authentication

This setting specifies how the provider authenticates against the target server. See the Authentication Methods section for details.
This setting is optional. If not provided, it will be Anonymous. The most commonly used method is Office365.

User

This part of the connection string specifies the username for the account which is used to authenticate against SharePoint. It needs to be specified for several authentication methods. See the Authentication Methods section for details.

Password

This defines the password for the account which is used to authenticate. It needs to be specified for several authentication methods by typing it into the Connection String or Password field. See the Authentication Methods section for details.

Library

The Library setting specifies the SharePoint document library to be used as the data entity. This could be the internal name, the display name, or the ID of the document library. This setting is mandatory when the full URL to the library is not used in the connection string.
Note: Custom lists, Calendar lists, etc., cannot be used here. Only document libraries.

View

The View setting is optional and can be used to define a specific set of sub-data of a document library to be synchronized. The view can be created and configured on the SharePoint portal as usual and then the view can be specified in the connection string. The View setting will accept the name of the view as well as the URL of the view’s .aspx-site or just the name of the view’s .aspx-site.

Optional Parameters

The following parameters are optional and should be changed with care as misconfigurations can lead to serious performance issues (CPU as well as memory) or errors during the synchronization (for example, in case the batch sizes are too large and cannot be processed).
On the other hand, environment-dependent, optimized settings for these parameters can greatly improve synchronization throughput by utilizing the host machine’s resources.

ReadingBatchSize

This setting specifies how many SharePoint items will be retrieved together from the server in one read request. By default, SharePoint allows reading up to 5000 items.

WritingBatchSize

This setting specifies how many SharePoint items will be committed together to the server in one update request. By default, the provider commits up to 200 items together during one update request. In case this causes errors, the value can be reduced.

RequestTimeout

This setting specifies the time in milliseconds before a client-side timeout exception is raised. By default, this is set to 300000ms (5 minutes). This value may be increased in case of frequent timeouts.

ChunkSize

Large files are uploaded in chunks, as recommended by Microsoft. This setting specifies the chunk size in MB. The default value is set to 250MB.

MaxFileSize

SharePoint has limitations about the maximum size of files. However, as this limit is not consistent among all SharePoint tenants, it is possible to change the maximum file size in MB that the Layer2 Cloud Connector tries to upload with this setting. The default value is set to 2000MB (2GB) because that is the official limit for file uploads to SharePoint Online.

SmallFilesMaxSize

Small files are uploaded in batches. This setting specifies the maximum size in MB that will be sent in one batch. The default value is set to 200MB. As this upload method is the fastest, increasing this value still can speed-up the upload, but increases the risk of request size errors.

MediumFilesMaxSize

Files of medium size are uploaded in one single request. This setting specifies the maximum file size in MB that is uploaded using this method. The default value is set to 250MB, as recommended by Microsoft. As this upload method is faster than the chunked upload, increasing this value can speed-up the upload, but increases the risk of request size errors.

SmallFilesThreadCount

For performance reasons, multiple files are uploaded simultaneously. This setting specifies how many small file uploads will be running at once. The default value is set to 5. Increasing this value can speed-up the upload, but leads to an increased memory consumption as each of these uploads may consume up to SmallFilesMaxSize MBs of memory.

MediumFilesThreadCount

For performance reasons, multiple files are uploaded simultaneously. This setting specifies how many medium file uploads will be running at once. The default value is set to 5. Increasing this value can speed-up the upload, but leads to an increased memory consumption as each of these uploads may consume up to MediumFilesMaxSize MBs of memory.

LargeFilesThreadCount

For performance reasons, multiple files are uploaded simultaneously. This setting specifies how many large file uploads will be running at once. The default value is set to 3. Increasing this value can speed-up the upload, but leads to an increased memory consumption as each of these uploads may consume up to ChunkSize MBs of memory.

RetryCount

In case of a transient error during a file upload, the provider will wait a few seconds and retry the upload. This parameter specifies how often an upload attempt is retried. The default value is 5.

RetryPeriod

In case of a transient error during a file upload, the provider will wait a few seconds and retry the upload. This parameter specifies how long the provider waits (in milliseconds) before attempting to retry. The default value is set to 5000ms (5 seconds).

Optimizing Configuration for Fast Uploads

To get the maximum upload performance, the configuration can be adjusted to the resources provided by the machine which is running the Layer2 Cloud Connector.
The decisive resource is the memory (RAM) of the machine. If the Layer2 Cloud Connector is the main application running on the system, about 2GB should be reserved for the operating system and other processes. If there are other applications with significant memory demands, these need to be taken into account as well.

The remaining memory can be configured to be used by the file upload. The consumed memory is calculated by the number of threads times the maximum upload size of the upload-method. With the default-configuration, the Provider will consume up to 2GB during the upload:

Memory for small file uploads:	5 x 200 MB =	1	GB
Memory for medium file uploads	5 x 250 MB =	1.25	GB
Memory for large file uploads	3 x 250 MB =	0.75	GB

Of course, memory will only be consumed as long as there are files of the specific size. So, if there are no files smaller than 200MB, all 5 small file threads will terminate right from the start and the maximum memory consumption will drop to 1GB.

On a machine with 8GB RAM, assuming 2GB for other processes, there will be 4 GB left to tweak the configuration and get a faster upload (8GB – 2GB reserved – 2GB for default configuration = 4GB). Depending on the spread of different sizes of files to upload, the thread count should be increased. For example, if there are a lot of files smaller than 200MB, the thread count for small files could be set to 25 to utilize the existing RAM and speed up the synchronization significantly.

Limitations Compared to the Generic SharePoint Provider

No Automatic Renaming

Files and folders will not be automatically renamed to work around SharePoint limitations. They will instead be rejected.

No Automatic Zipping

File extensions forbidden by SharePoint will not re-uploaded as a ZIP archive. They will be rejected.

Only SharePoint Online or SharePoint 2016

This provider only works with SharePoint Online or SharePoint 2016 on-premises.

Only Document Libraries

SharePoint lists (custom lists, calendar lists, etc.) are not supported.

No Folder as Target

Using the **Folder** connection string parameter or using an URL with a folder reference will return an error as this is not supported by the provider.

No Check-in / Check-out

The provider does not check out the files to update them. If the document library is configured to require check-out, the provider will return an error.

Conflict Resolution Type KeepBoth is not supported

The conflict resolution type KeepBoth cannot be used with the Office 365 Fast File Sync Provider. KeepBoth is currently only available for file synchronizations with the Layer2 File System Provider or the Layer2 SharePoint Provider

Switching Between Providers

Switching a synchronization from using the fast file upload provider to using the generic SharePoint provider is supported. Switching the other way around, however, is not recommended.

Layer2 Data Provider for Office 365 Groups

The Layer2 Office 365 Groups provider connects to Webhooks of Microsoft Office 365 Groups so that it can write a “Card” to it after each synchronization. The card contains information about changes in the source data entity that were made since the last synchronization run. It can, for example, be used to regularly notify Office 365 Group members about changes that have been made in a certain local or cloud-based business system like SQL, ERP/CRM, etc., as supported by the Layer2 Cloud Connector.

The Layer2 Data Provider for Office 365 Groups cannot be used to sync documents in an Office 365 Groups document library. You can use of the Layer2 Data Provider for SharePoint (Libraries and Lists) to do this.

Group Cards can only be written to but not read from Office 365 Groups using the Layer2 Cloud Connector.

The Layer2 Office 365 Groups provider does not manipulate any data but just writes cards to an Office 365 Group. Therefore, it is best practice to define the connection as uni-directional (although bi-directional also works).

Card Reference

The card that will post the changes contains the following elements:

CardTitle	The title of the card. The content of this field needs to be specified in the connection string as the “CardTitle=;” parameter.
Summary	The summary contains information about the changes that were made since the last synchronization run. The format is: *x items inserted, y items modified, and z items removed in <Office 365 data entity name*[2]*>.*
Section	A section is created per item that has been changed. The title gives information about the data record and the kind of change. The format will be like the following: *Item ‘<record-key>’ was added to ‘<Office 365 data entity name>’.*

[2] No data is changed in the Office 365 Group (just a card sent), however the group is treated as target data entity and “as-if” data had been changed there during the synchronization.

Section-Details	Each section can contain detailed information about the changes made to the data record. The information is shown as table of name/value pairs, showing the mapped fields (see below) and their new values.

Connection String

A typical connection string for the Office 365 Groups provider looks like this:

WebHookUrl=<url of the webhook to the Office 365 group>; CardTitle=MyCardTitle

Here is the full list of connection string parameters for this provider:

WebHookUrl

The WebHookUrl is created in the Office 365 Group you want the Layer2 Cloud Connector to send cards to.
When opening the Office 365 Office group, you can find an option at the top menu bar called **Connectors**. Click on that to start configuring your Webhook Connector.

You need to create a connector by choosing the "Incoming Webhook" Connector and then click **Add**.

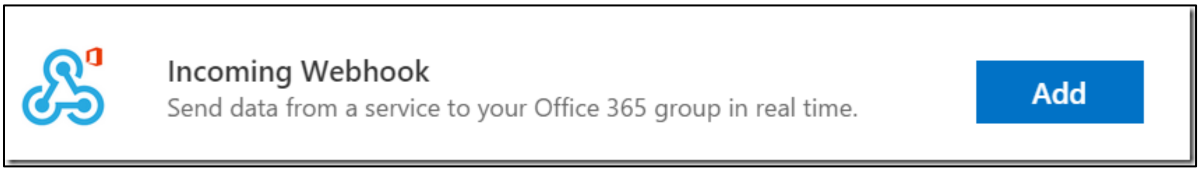

Incoming Webhook
Send data from a service to your Office 365 group in real time.

Figure 60 –"Incoming Webhook" Connector in Office 365 Groups

To configure the Incoming Webhook, you need to give it a name and optionally add an image for the Connector. Once that is done you need to click on **Create** to create the Webhook. When it is created you will see an input box with a URL in it. This is the Webhook, and you need this URL in the connection string of the Layer2 Cloud Connector. The URL is unique to this Group and Webhook. If you remove the Webhook, you will disable it.

Note: The Webhook URL will validate even if it is incorrect as it cannot be checked without sending a card to post.

CardTitle

This parameter sets the text that is written as the title of the card. It is recommended to use the connection name so that you can immediately see which connection sent this card.

SkipDetails

When this parameter is set to “true”, no sections are created, but just the CardTitle and the Summary information. This is useful if many data records have been changed (for example, due to a mass update in your business system) and you do not want to get every change detail but just the summary of the synchronization This parameter is optional and defaults to “false”.

Select Statement

The select statement defines which fields are offered for mapping and, if mapped, what will be shown in the section details.

The format for the select statement is this, where `[FIELDS]` is a comma-separated list of the fields:

```
    SELECT [FIELDS]
```

Fields must be explicitly defined; the wildcard * is not supported.

Note: Only changes for fields that are mapped will be tracked. So, in case a field of a record in your business system has changed and this field is not mapped (and therefore not part of the synchronization), the change will not be tracked in the card at all.

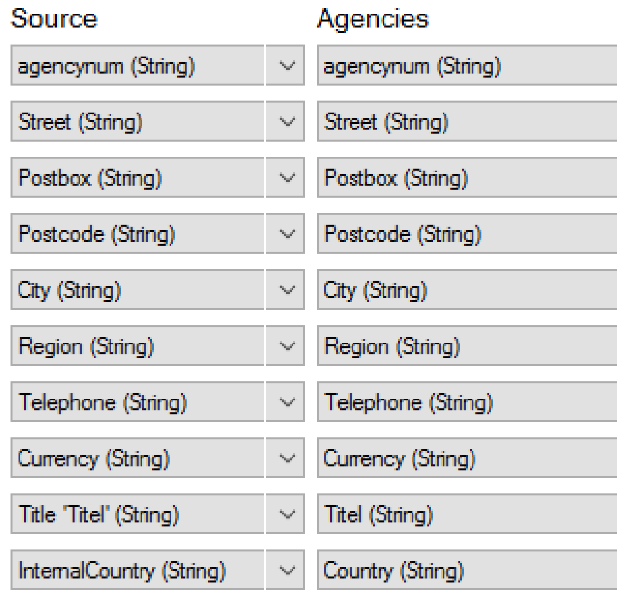
Primary Key(s)

You need to specify the field(s) that are used to create the Section title. If using a GUID- or counter-type of primary key, it might be a good idea to add another detailed column (for example, “Agencynum, Title”) to have human-readable information in the card.

Example:

Synchronization from an Agency list in SharePoint to Office 365 Group

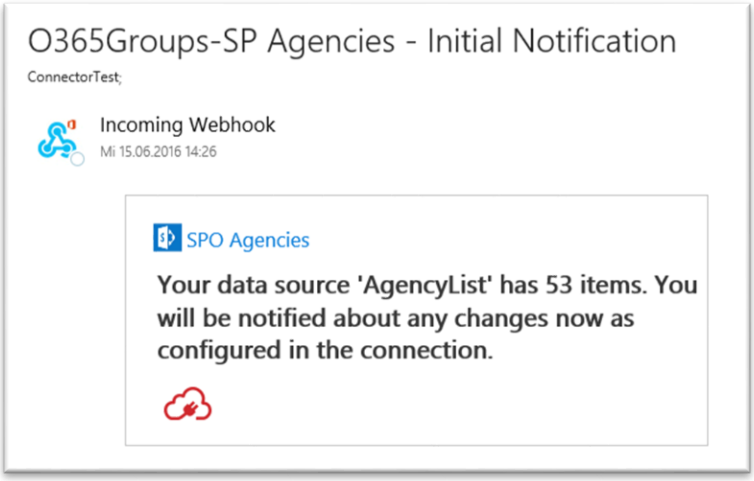
Connection string	WebHookUrl=<MyWebHookUrl>;CardTitle=O365Groups-SP Agencies;
Select statement	select agencynum, Titel, Street, Postbox, Postcode, City, Region, Telephone, Language, Currency, Country
Primary Key(s)	Agencynum, Title



Mapping	Source	Agencies
	agencynum (String)	agencynum (String)
	Street (String)	Street (String)
	Postbox (String)	Postbox (String)
	Postcode (String)	Postcode (String)
	City (String)	City (String)
	Region (String)	Region (String)
	Telephone (String)	Telephone (String)
	Currency (String)	Currency (String)
	Title 'Titel' (String)	Titel (String)
	InternalCountry (String)	Country (String)

Example resulting Group Card for the initial synchronization:

- **O365Groups-SP Agencies** has been set via CardTitle parameter in the connection string
- **SPO Agencies** is the name of the created Webhook
- **SharePoint logo** has been set when the Webhook was created
- **AgencyList** is the name of the source data entity



O365Groups-SP Agencies - Initial Notification

ConnectorTest;

Incoming Webhook

Mi 15.06.2016 14:26

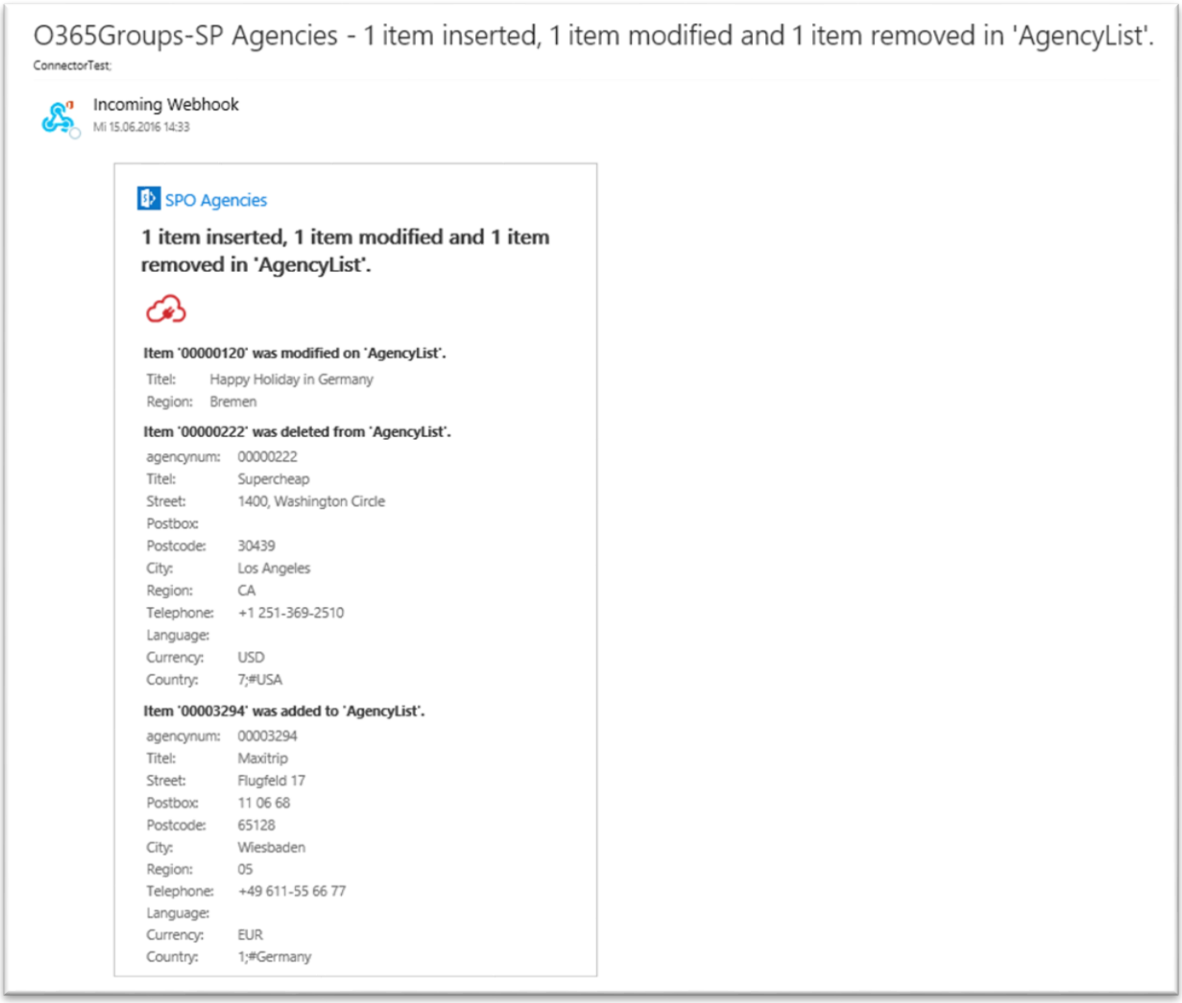
SPO Agencies

Your data source 'AgencyList' has 53 items. You will be notified about any changes now as configured in the connection.

Figure 58 - Example card output for initial synchronization

Example resulting Group Card for subsequent synchronization:

- The detail fields are shown in the order as specified in the select statement
- Detail fields are only shown, if changed



O365Groups-SP Agencies - 1 item inserted, 1 item modified and 1 item removed in 'AgencyList'.

ConnectorTest;

Incoming Webhook
Mi 15.06.2016 14:33

SPO Agencies

1 item inserted, 1 item modified and 1 item removed in 'AgencyList'.

Item '00000120' was modified on 'AgencyList'.

Titel:	Happy Holiday in Germany
Region:	Bremen

Item '00000222' was deleted from 'AgencyList'.

agencynum:	00000222
Titel:	Supercheap
Street:	1400, Washington Circle
Postbox:	
Postcode:	30439
City:	Los Angeles
Region:	CA
Telephone:	+1 251-369-2510
Language:	
Currency:	USD
Country:	7;#USA

Item '00003294' was added to 'AgencyList'.

agencynum:	00003294
Titel:	Maxitrip
Street:	Flugfeld 17
Postbox:	11 06 68
Postcode:	65128
City:	Wiesbaden
Region:	05
Telephone:	+49 611-55 66 77
Language:	
Currency:	EUR
Country:	1;#Germany

Figure 59 - Example card after a synchronization with changes

Layer2 Data Provider for RSS

The Layer2 RSS Provider retrieves data from RSS (known as Really Simple Syndication) feeds. As RSS is a very common format for data exchange, this can be used to connect many systems. For example, it is used for news feeds. The connector is read-only.

The provider does not support select statements. It gets all mandatory and additional (non-standard) columns from the feed.

Connection String

A typical connection string for the RSS provider looks like this:

Url=<FeedUrl>; FeedType=<FeedFormat>;

Here is the full list of connection string parameters for this provider:

URL

This is the URL of the RSS feed. This information is mandatory.

FeedType

This is the format information of the RSS feed. Possible values are RSS, ATOM or RDF. The content will be based on this parameter.

Source

Creates an additional column named “Source”, if it does not already exist, and fills the content with the given value. If the column is present, the values will be overwritten by the given value. This is useful when gathering RSS feeds from different sources.

Authentication

This setting specifies how the RSS provider authenticates against the target server. See the Authentication Methods section for details.

User

This part of the connection string specifies the username for the account which is used to authenticate against the server. It needs to be specified for several authentication methods. See the Authentication Methods section for details.

Password

This defines the password for the account which is used to authenticate. It needs to be specified for several authentication methods by typing it into the Connection String or Password field. See the Authentication Methods section for details.

Layer2 Data Provider for SharePoint

The Layer2 SharePoint Provider connects to SharePoint Foundation, SharePoint 2013, SharePoint 2016, SharePoint 2019, SharePoint Server Subscription Edition, and SharePoint Online (Office 365 or OneDrive for Business) to retrieve data from and write data to SharePoint lists, calendars, contacts, tasks, and document libraries. The provider supports the SQL-Like Select Statement. However, this should only be used if querying cannot be done by setting up an appropriate SharePoint list view. See also the SharePoint (CSOM) Provider Specifications site on the web.

Connection String

A typical connection string for the SharePoint provider looks like this:
Url=<SharePointSiteUrl>;List=<ListName>;Authentication=<auth>
Here is the full list of connection string parameters for this provider:

URL

This is the URL of the SharePoint site-collection where the list or document library is located or the full URL to the view in the SharePoint web-site. When a full URL is used then the list, folder and view information is automatically retrieved from it. Otherwise the list-name must be specified separately. The URL is mandatory.

Authentication

This setting specifies how the SharePoint-Provider authenticates against the target server. See the Authentication Methods section for details.
This setting is optional. If not provided, it will be Anonymous.

User

This part of the connection string specifies the username for the account which is used to authenticate against SharePoint. It needs to be specified for several authentication methods. See the Authentication Methods section for details.

Password

This defines the password for the account which is used to authenticate. It needs to be specified for several authentication methods by typing it into the Connection String or Password field. See the Authentication Methods section for details.

List

The List setting specifies the SharePoint list, calendar, or document library to be used as the data entity. This could be the internal name, the display name or the ID of the list or document library. This setting is mandatory when the full URL to the list is not used in the connection string.

View

The View setting is optional and can be used to define a specific set of sub-data of a list or document library to be synchronized. The view can be created and configured on the SharePoint portal as usual and then the view can be specified in the connection string. The View setting will accept the name of the view as well as the URL of the view's .aspx-site or just the name of the view's .aspx-site.

Folder

With this optional setting, it is possible to define a folder-path which should be used as the root for the synchronization.

BatchReadItems

This setting is optional and specifies how many SharePoint items will be retrieved together from the server in one read request. By default, SharePoint allows reading up to 5000 items.

BatchWriteItems

This setting is optional and specifies how many SharePoint items will be committed together to the server in one update request. By default, SharePoint allows committing up to 50 items together during one update request.

Scope

This setting is optional and specifies how the items on a list with folders will be read. The value "Recursive" tells that all items will be read including subfolders, and that folders are not listed in results. The value "RecursiveAll" gets all items plus folders. Default value is "RecursiveAll".

AppWebUrl

Parameter is mandatory for and only used when **License** is set to "SPAppStore". The parameter value must be set to the unique URL of the Layer2 Cloud Connector application web on destination SharePoint instance.

AutoRenaming

This setting is optional and can either be set to "true" to enable it or "false" to disable it. If it is not specified, the provider will consider it enabled (default value is "true") except for newly created connections to SharePoint Online (version >= 16.0.8405.1236). Various restrictions concerning item names (forbidden characters, prefixes, etc.) will be avoided by escaping them if enabled.
For a detailed description of the renaming logic, see the AutoRenaming section.

AutoZipping

This setting is optional and can either be set to "true" to enable it or "false" to disable it. If it is not specified, the provider will consider it enabled (default value is "true"). If the SharePoint blocks a file upload due to forbidden file extensions, this feature will try to upload a ZIP archive of the file in question. Its name will be appended by ".l2cc.zip" to distinguish it from regular archives.

AutoZippingExceptions

This setting is optional and expects a comma-separated list of file extensions which should be excluded from the auto-zipping feature. If these extensions are blocked by the SharePoint, uploading a file with one of them will throw an error. By default, the following extensions are excluded: pdf, xlsx, doc, docx, jpeg, and jpg.

Managed Metadata Handling

Managed Metadata can be synchronized between mapped fields in SharePoint lists and libraries, if one of the following two prerequisites is matched:

- The fields are bound to the same term set (same SharePoint farm).
- The fields are bound to different term sets (possibly different SharePoint farms) but the terms and the hierarchy in the two term sets are identical.

In the case that Managed Metadata from a list or library is synchronized to non-SharePoint data sources, the values are exported as described in the following examples:

__TermSet__ Regions

```
EMEA
    Europe
        Germany
        Italy
        Spain
    Middle-East
USA
    North America
        Washington
        Massachusetts
    South America
```

Managed Metadata field is bound to "Regions".

Example 1:

Field has value "Washington"
Value is exported as "USA;Nort America;Washington"

Example 2:

Multiple values are allowed and the field has two values set: "Italy;South America"
Value is exported as "EMEA;Europe;Italy|USA;South America"

To import Managed Metadata values from a non-SharePoint data source into a SharePoint list or library, the values must be in the exact format as described in Examples 1 and 2 above and must match the bound term set hierarchy and terms of the Managed Metadata field that is being synchronized to. Note that the Layer2 Cloud Connector does not currently support synchronization of Managed Metadata between external sources and the SharePoint Term Store.

Writing Modified and Created timestamps

Usually, SharePoint maintains the fields *Created* and *Modified.* When a list-item is created in a SharePoint-list or a file is added to a document-library, SharePoint automatically assigns the current date and time for the *Created* field. Likewise, whenever an entry changes SharePoint will update the *Modified* field.

Due to its intended internal usage, there are some things to consider when writing to those fields:

- The user, which is used to synchronize, needs to have Site-collection administrator permissions; otherwise, *Created* and *Modified* are read-only.
- If checkout/checkin is enforced on a document-library the modified field will always be overwritten with the current date on check-in.

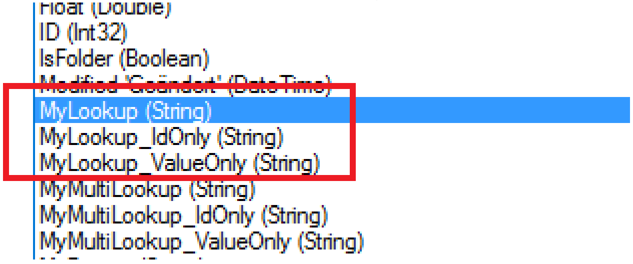
User/Lookup Handling

For both user and lookup fields, the SharePoint provider can handle "only ID" (e.g., 12), "only value" (e.g., myLookupValue), and full values (e.g., 12;#myLookupValue). Please note that giving only the value will require it to be unique in the target system and comes with a slight impact on synchronization performance.

For lookup fields, you can explicitly map the part of the lookup you wish to use:

- The original lookup field (delivers the full value, e.g. *1;#myLookupValue*)
- The *<lookpField>_idOnly* field (delivers only the id, e.g. *1*)
- The *<lookupField>_valueOnly* field (delivers only the value, e.g. *myLookupValue*).

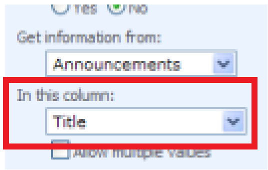
The latter *_valueOnly* field is required if the values contain numeric values as it's otherwise not possible to determine whether the passed value is an id or a value.



Float (Double)
ID (Int32)
IsFolder (Boolean)
Modified 'Geändert' (DateTime)
MyLookup (String)
MyLookup_IdOnly (String)
MyLookup_ValueOnly (String)
MyMultiLookup (String)
MyMultiLookup_IdOnly (String)
MyMultiLookup_ValueOnly (String)

Figure 61 - Example of how the lookup fields are shown in the mapping

Please make sure that when you synchronize from SharePoint to SharePoint that you specify the same source column in the lookup field definition for both sides. So, for example, "Title" should be used on both sides or "ID" on both sides.



○ Yes ● No

Get information from:
Announcements

In this column:
Title

□ Allow multiple values

Layer2 Data Provider for SOAP Web Services

The Layer2 Data Provider for SOAP Web Services connects to SOAP web services based on their WSDL. It only supports read operations.

Connection String

A typical connection string for the SOAP provider looks like this:

Url=<Url of the web service >;User=<UserName>; Protocol=<SOAP protocol version>;WsdlQuery=<wsdl | singlewsdl>

Additionally, the password needs to be typed into the Connection String or Password field.

With this connection-string, the provider will automatically discover the WSDL by the given URL and connect with the given credentials.

Full list of connection string parameters for the provider:

URL

This parameter is **mandatory**. It is the URL of the web service.

User

The name of the user to authenticate with against the web service. This setting is optional.

Password

The password of the user to authenticate with against the web service. It needs to be typed into the Connection String or Password field. This setting is optional.

Protocol

This setting specifies the SOAP protocol version to use. It is optional. If it isn’t set, the protocol SOAP1.2 is used as default. You can determine the protocol from the namespace definitions in the web service’s WSDL.

WsdlQuery

This setting specifies how the Wsdl of the web service is queried. It is optional. If it isn’t set, the query “?wsdl” is used as default. If “singlewsdl” is set, the query “?singlewsdl” is used to query for the full single-file Wsdl without any external Xsd-references.

Queries

The Layer2 Data Provider for SOAP Web Services supports a slightly different syntax than SQL to query the items from the web service as described below.

```
    SELECT [Fields] FROM [Operation]([Parameters])
```

The list of fields can either contain a wildcard (*) or a comma-separated list of property names returned by the web service method. Furthermore, it is possible to rename the fields by using the SQL alias-syntax, i.e., SELECT Companyname AS cmp, would select the contents of property Companyname and populate it as a field named cmp in the result.

The operation part specifies the operation to use for item retrieval.

The parameters part, which is enclosed in parentheses, is optional and contains a comma-separated list of the parameters the operation expects.

Restrictions

The Layer2 Data Provider for SOAP Web Services supports web services only under the following conditions:

- The WSDL must be compliant to WS-I Basic Profile.
- The web services require either no authentication or windows authentication (username and password). Custom authentication mechanisms (i.e., session-id or token based) are not supported.
- Operation parameters are of simple type (i.e., string, integer, etc.). Parameters of complex types defined in the WSDL or of type “any” are not supported.
- The result returned from the operation does not contain multi-occurrence properties. If it does, these properties will be skipped (omitted) and a warning will be raised.

Layer2 Data Provider for XML

The Layer2 Xml Provider retrieves data from XML sources. It is typically used to connect to any XML-based data sources, files, or web requests. It could also be used to integrate systems or applications that do not offer a specific data provider. Data can be exported as an XML file or exposed as XML via HTTP to connect to those kinds of systems. The provider is read-only.

The data is fetched using XPath expressions. See the XML Provider Specifications on the web for more information.

Reading multiple XML files

If the target URL is a directory path, all XML files in that directory and any sub-directories are read for the resulting data set.

The primary keys are merged into a new L2CC_ID column, and the file paths are added as a new FilePath column. Columns that are not present in all files will have empty values for files without them.

Connection String

A typical connection string for the XML provider looks like this:

Url=<XmlPath>;

Here is the full list of connection string parameters for this provider:

URL

The path to the XML file which can be a URL, a local file path or a local directory path. This field is mandatory. Aliases are: Url, DataSource, Data source, Web, Host, Server, Address.

Authentication

This setting specifies how the XML-Provider authenticates against the target server. See the Authentication Methods section for details.

User

This part of the connection string specifies the username for the account which is used to authenticate against the server. It needs to be specified for several authentication methods. See the Authentication Methods section for details.

Password

This defines the password for the account which is used to authenticate. It needs to be specified for several authentication methods by typing it into the Connection String or Password field. See the Authentication Methods section for details.

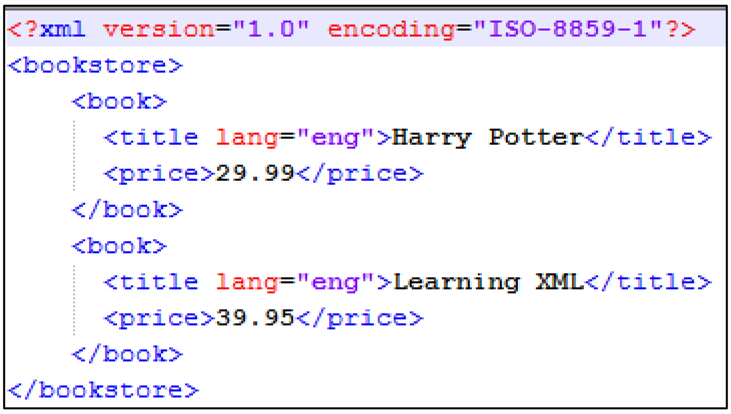
Select Statement

The select statement could look like the following example, which requests the title, language, and price of each book in the bookstore XML structure:

```
    Select title, title/@lang, price from /bookstore/book
```

The “select” and “from” are fixed keywords (just like SQL select statements). The other parts are XPath expressions which extract the relevant portions out of the XML. Referring to the example above, `/bookstore/book` is the start node of the select, `title` gets the content of the title node (sub-node of book), and `title/@lang` selects the lang attribute of the title node (see figure 35).

Title	Title/@lang	Price
Harry Potter	eng	29.99
Learning Xml	eng	39.95



```
<?xml version="1.0" encoding="ISO-8859-1"?>
<bookstore>
    <book>
      <title lang="eng">Harry Potter</title>
      <price>29.99</price>
    </book>
    <book>
      <title lang="eng">Learning XML</title>
      <price>39.95</price>
    </book>
</bookstore>
```

Figure 60 - Example XML file

SQL-Like Select Statement

The following providers support a SQL-like syntax to filter the files and folders found in the given directory:

- Layer2 Data Provider for File System
- Layer2 Data Provider for FTP
- Layer2 Data Provider for Dropbox
- Layer2 Data Provider for Google Drive

The select statement can be specified as described below.

```
SELECT [Fields] FROM [Directory] WHERE [Filter]
```

The list of [Fields] can either contain a wildcard (*) or a comma-separated list of field names. Furthermore, it is possible to rename the fields by using the SQL alias-syntax. For example, `SELECT FileExtension AS ext` would select the contents of the file extension field and populate it as a field named 'ext' in the result.

The FROM part of the select statement currently supports exactly one directory or FTP(S) path. This is optional if the connection string contains the 'Directory' parameter, otherwise it is mandatory. If both are given, the select statement has a higher priority and a warning will be logged.

The WHERE clause supports various elements known from SQL to build a complex conditional filter:

- Conditional operators: <, >, <=, >=, =, <>, LIKE, IN
- Logical operators: NOT, AND, OR
- Brackets can be used to change precedence

Various wildcards can be used with conditional operators. LIKE supports "%" (any characters) and all other conditional operators support "?" (exactly one character) and "*" (like %).

Example filter only includes files/folders in specific folders:

```
WHERE FilePath IN ("/myFolderA/*", "/myFolderC/myFolderD/*")
```

Example filter excludes files/folders in specific folders:

```
WHERE NOT FilePath IN ("/myFolderA/*", "/myFolderC/myFolderD/*")
```

Example for a simple filter to get all txt files:

```
WHERE FileExtension = ".txt"
```

Example for a complex filter to get all files created before the 1st of February 2016 with 'customer' in their file names:

```
WHERE Filename = "*customer* " AND Created < "2/1/2016"
```

Example filter selects all files starting with the letter ‘A’:

```
WHERE Filename LIKE "A% "
```

Example filter selects all files starting with the letter ‘A’ and ending with ‘CDE’:

```
WHERE Filename = "A*CDE"
```

Example filter selects all files bigger than 1GB:

```
WHERE Size > 1 GB
```

Example filter excludes all files with extension .aspx:

```
WHERE NOT FileExtension LIKE ".aspx "
```

Supported units for file-sizes are:

- KB = Kilobytes
- MB = Megabytes
- GB = Gigabytes

If the “IncludeFolders” parameter is “true”, which is the default, all folders that contain files matching the filter, will implicitly be included in the result, even if they do not match the filter. This is to make sure files will never be returned without the corresponding folders. If only files should be returned, the “IncludeFolders” parameter needs to be set to “false”.

Synchronization of Binary Content

This section describes how to synchronize binary content (i.e., files) between two systems. Unless explicitly denoted, the data is streamed to reduce memory consumption.

Local or Remote File System and Server File Shares

Local or remote files and folders can be synchronized using the Layer2 Data Provider for File System.

Whether a file has been changed is determined based on the modified date.

SharePoint and Teams Document Libraries

Files and folders on SharePoint and Teams can be synchronized using the Layer2 Data Provider for SharePoint or the Layer2 Data Provider for Office 365 Fast File Sync.

Whether a file has been changed is determined based on the modified date.

Microsoft SQL Binary Data

Binary data stored in a SQL BLOB can be synchronized using the .Net Framework Data Provider for SqlServer.

Whether the data in the BLOBs has been changed is determined based on the checksum of these columns in the record. The BLOB content is shown as “Binary” in the Data Preview and can be mapped and synchronized to file system or SharePoint FileContent column for example.

Note: The SQL field type Image is not supported by the Checksum function. Therefore, changes in Image fields are not detected out-of-the box.

Note: MS SQL supports a maximum of 2 GB for a single BLOB field.

Other Provider Binary Data

String and ByteArray fields can also be mapped and synchronized to binary data (i.e., FileContent).

When string content is written to a binary field, it is encoded using UTF-8.

In case a provider returns a string field, the value of this column can be mapped and synchronized to FileContent of a file on the file system for example.

In case a provider returns a ByteArray field, this will be treated as binary field which means:

- It will be shown with data type “Binary” in the mapping dialog
- Its value will be represented as “Binary” in the data preview
- Its value can be synchronized to any other binary data column (i.e., FileContent)

In the case of synchronizing ByteArray values to binary data, it is recommended to specify a ChangeDetectionField in the connection string to avoid repeated updates. The following sample shows how to define the ChangeDetectionField:

ChangeDetectionField=”MyChangeIndicationField”

Note: The data in the String and ByteArray fields is not streamed but for all records entirely loaded into memory. Therefore, large content synchronizations require at least as much free memory as the sum of the binary data columns.

Authentication

This section describes the details regarding the authentication used by our data providers against various target systems.

Authentication Sequence

Instead of just one authentication method, multiple methods can be used in a sequence. Later methods will use all authentication data provided by former methods. Currently, this feature is only in use with the SapHeader method. The following sample shows how to define a sequence:

Authentication=ADFS,SapHeader

Authentication Construction Kit

The Layer2 Cloud Connector comes with many pre-built authentication methods that cover most user cases (see the Authentication Methods section for details). The Authentication Construction Kit is intended for advanced users that have authentication needs not covered under the standard pre-built methods.

With the Authentication Construction Kit feature, any browser-based authentication can be added to the Layer2 Cloud Connector. All you need is an analysis of the authentication process used by your data source (such as with a Fiddler trace) and the guidance given in this section. Based on the analysis, you can write an XML-based file with a series of actions to mimic the process.

Authentication Directory

This folder can be found under The Layer2 Cloud Connector Data Directory. It contains authentication files, a folder for regular expression files, and a template folder for reusable XML-fragments. If access to the file system is not possible, the default authentication configurations will be used, but the authentications cannot be customized.

Authentication File

These files describe the actions taken to mimic the process necessary to authenticate against a target system.

```xml
<?xml version="1.0" encoding="utf-8" ?>
<Authentication>
  <Actions>
    <Store />
    <Request />
    <Log />
    ...
  </Actions>
  <RegularExpressions>
    <RegularExpression />
    ...
  </RegularExpressions>
</Authentication>
```

Figure 61 – Example authentication file

Regular expressions can either be defined inline, inside of an authentication file, or globally, in regular expression files in the “RegularExpressions” directory. Using an Inline definition of regular expressions increases the portability because the authentication files can be distributed without any regular expression files. However, defining regular expressions globally enhances reuse. For global regular expressions, see the Regular Expression File section below.

If your authentication requires user input (for example via Browse action) then you need to create a second file that follows this pattern: *<AuthName>_client.xml* (e.g., *DynamicsCrmViaOAuth_client.xml*)
These will only be executed in a client context (**Run Now** in the MMC, **Data Preview** in the MMC, and so on).

In addition, if you create an authentication file that is named after the invariant name of any provider, this file is also only executed in a client context. This is useful for scenarios where you want to execute some authentication logic for third-party providers.

Regular Expression File

These files are used to store necessary global regular expression definitions.

```
<?xml version="1.0" encoding="utf-8" ?>
<RegularExpressions>
  <RegularExpression />
  ...
</RegularExpressions>
```

Figure 62 - Example regular expression file

Regular expression nodes need a name and a regular expression, which contains at least one match-group. If an expression contains XML-specific characters, it must be encased in a CDATA block.

```
<RegularExpression Name="Url">
  <Expression>
<![CDATA[^(?<ProtocolHost>(?<Protocol>https?:\/\/)?(?<Host>[\da-z\.-
]+)?)?(?<Path>\/[\/\w \.-]*)*]]>
  </Expression>
</RegularExpression>
```

Figure 63 - Example regular expression definition

Actions

All attributes and sub-nodes of actions support placeholders. If a value contains XML-specific characters, it must be encased in a CDATA block.

Condition Attribute

All actions have a common attribute called “Condition”. Depending on the given value, the action is executed or not.

No condition	`Action is executed`
Empty	`Action is NOT executed`

Non-empty	Action is executed
Empty with exclamation mark	Action is executed
Non-empty with exclamation mark	Action is NOT executed

```
<Log Message="This message will be logged." />
```

Figure 64 – Example for no condition

```
<Log Condition="[NonEmptyVariable]" Message="This message will be
logged." />
```

Figure 65 – Example for non-empty condition

```
<Log Condition="[NonExistingVariable|]" Message="This message will NOT be
logged." />
```

Figure 66 – Example for empty condition

```
<Log Condition="![NonEmptyVariable]" Message="This message will NOT be
logged." />
```

Figure 67 – Example for non-empty condition with exclamation mark

```
<Log Condition="![NonExistingVariable|]" Message="This message will be
logged." />
```

Figure 68 – Example for empty condition with exclamation mark

<Request>

This sends out a web request and stores the response as “LastResponse”. It also adds all received cookies to “Cookies”. All requests need a URL as a target. By default, the request will be executed using the HTTP/POST verb if a body is set and HTTP/GET if no <Body> is specified. The verb can also explicitly be specified using the “Method”-property. It is also possible to define headers for the request as shown in the example.

```
<Request>
  <Headers>
    <Header Name="Content-Type" Value="application/soap+xml; charset=utf-
8" />
  </Headers>
  <Url>[Settings.Url]</Url>
  <Body>
<![CDATA[UserName=[Settings.User]&Password=[Settings.Password]&AuthMethod
=FormsAuthentication]]>
  </Body>
</Request>
```

Figure 69 - Example <Request> action

AutoRedirect

The request automatically tries to follow redirects, but this behavior can be disabled by setting “AutoRedirect” to “false” (as seen in the sample below).

```
<Request AutoRedirect="false">
  <Url>[Settings.Url]</Url>
</Request>
```

Figure 70 - Example of a <Request> without auto-direct

NTLM

The request automatically tries to handle NTLM authentication with default credentials. You can specify the credentials (User, Password, and Domain [optional]) via sub-nodes (as shown in the sample below) or disable this feature completely by setting “HandleNTLM” to “false”.

```
<Request User=”[Settings.User]” Password=”[Settings.Password]”
Domain=”myDomain”>
  <Url>[Settings.Url]</Url>
</Request>
```

Figure 71 - Example <Request> with custom credentials for NTLM authentication

<Store>

This action needs a name and a value. The value is stored as a named variable for later usage.

```
<Store Name="Url_ProtocolHost" Value="[Settings.Url:Url.ProtocolHost]" />
```

Figure 72 - Example <Store> action

Some special variables are automatically created for specific uses:

- “Settings” holds all connection string parameters
- “Output” holds “Output.Headers” and “Output.Cookies” which are used to determine the headers and cookies used after the authentication process is done
- “CrmIdentifier” helps identifying the CRM realm for some authentications
- “AuthenticationData” holds data from previous authentications in the current sequence
- “Persisted” holds values that should be saved longer than one authentication execution

<CreateFile>

This action needs the path to a file and its content. If the file already exists, the content will be overwritten and not appended.

```
<CreateFile FileName="C:\MyFolder\MyFile.txt">
  <Content><![CDATA[This is the first line.
This is the second line.
This is the third line. And now we will add an empty line.
]]></Content>
</CreateFile>
```

Figure 73 - Example <CreateFile> with multiple lines of text as content

<Log>

This action writes to the Layer2 Cloud Connector log and is mostly for debugging purposes. It requires a message and optionally a log level, which can have one of the following values: Trace, Debug, Info, Warn, Error, or Fatal.

```
<Log LogLevel="Trace" Message="Step 3 - MS authentication via SAML token
from [LastResponse.Url]" />
```

Figure 74 - Example <Log> action

<Browse>

This action was primarily implemented for authentication processes that require input via Browser and then send a response to a callback (such as OAuth). It opens the given URL in a browser-like popup and starts a HTTP listener on the given port.

```
<Browse Url="http://mysite.mydomain" Port="8910" />
```

Figure 75 - Example <Browse> action

Event Handlers

Event handlers are an optional part of the authentication file. They will execute their actions whenever the associated event is triggered.

```
<?xml version="1.0" encoding="utf-8" ?>
<Authentication>
  <EventHandlers>
    <EventHandler Event="BeforeAction" Action="Request">
      <Log Message="Next action is a request." />
    </EventHandler>
  </EventHandlers>
  <Actions>
    <Store />
    <Request />
    <Log />
    ...
  </Actions>
  <RegularExpressions>
    <RegularExpression />
    ...
  </RegularExpressions>
</Authentication>
```

Figure 76 – Example file with an event handler which is logging a message before each request action

Scope Identification

Each authentication needs a scope identifier for internal purposes such as caching. If there is no scope identification block in an authentication file, the default scope identifier is used.

```
<?xml version="1.0" encoding="utf-8" ?>
<Authentication>
  <ScopeIdentification>
   <Store Name="Output.ScopeIdentifier" Value="[Settings.User];Custom" />
  </ScopeIdentification>
  <Actions>
    <Store />
    <Request />
    <Log />
    ...
  </Actions>
  <RegularExpressions>
    <RegularExpression />
    ...
  </RegularExpressions>
</Authentication>
```

Figure 77 – Example file with a custom scope identifier

Placeholders

These follow a simple pattern: [Name-Transformation-Default] or [Name-Default-Transformation]

The placeholder always starts with the opening placeholder token '[', then always expects a name of a variable. After that a default value (indicated by '|'), one of several transformations can be specified and the placeholder ends with ']'. Default value and transformations are both optional.

```
 [LastResponse.Body:myRegularExpression|http://www.sample.com]
```

Figure 78 - Example Placeholder. Returns the action of the first form in the body of the last response or 'http://www.sample.com' if that cannot be done

Name

This can either be something previously stored with the store action or one of these default variables: "Settings" (connection string components, e.g., Settings.User), "LastResponse" (e.g., LastResponse.Headers.Location) or "Cookies". LastResponse and Cookies are only available after the first request action.

Default Value

This is indicated by '|' and specifies a value that is used if the variable or transformation before it returns nothing.

Transformations

These are indicated by various characters, but they all perform a change the value.

Trim

This is indicated by ';Trim' and removes any unimportant leading or trailing whitespace characters.

Example:

```
[Settings.Url;Trim]
```

Url Decode

This is indicated by ';UrlDecode' and will decode URL-encoded strings.

Example:

```
[LastResponse.Headers.Custom;UrlDecode]
```

Url Encode

This is indicated by ‘;UrlEncode‘ and will URL-encode values which might be required by some systems.

Example:

```
[Settings.User;UrlEncode]
```

Regular Expression

This is indicated by ‘:’ followed by the name of a regular expression and optionally the name of a named group inside that expression. These regular expressions are executed with the SingleLine option.

Example of a regular expression transformation with named group:

```
[Settings.Url:Url.ProtocolHost]
```

Example of a regular expression transformation *without* named group:

```
[LastResponse.Body:MyRegularExpression]
```

Extract Input Value

This is indicated by ‘;extractInputValue‘ followed by the name or id of the desired input element. As the name suggests, this will return the value of said input.

Example:

```
[LastResponse.Body;extractInputValue.Wresult]
```

Extract Form Action

This is indicated by ‘;extractFormAction ‘ followed by the name or id of the desired form element. As the name suggests, this will return the action of said form.

Example:

```
[LastResponse.Body;extractFormAction.Login]
```

Authentication Methods

The Layer2 ADO Providers provide several different methods to authenticate against the system which they access. This section describes the different provided authentication methods. Only the Layer2 Exchange Provider has its own authentication mechanisms (See the Layer2 Data Provider for Exchange section for details).

All authentications implemented with the authentication construction kit have a fallback for compatibility reasons. To use the fallback, you add "_legacy" to the method name. For example: ADFS_legacy

ADFS

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

It is the authentication method for accessing Office365 using ADFS. ADFS is the authentication method to be used when your organization redirects you to a custom login page in a browser when accessing Office 365.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This part of the connection string specifies the local Windows domain username in the format: DOMAIN\Username.

Password

This defines the password of the Windows domain account, which is used to authenticate. It needs to be typed into the Connection String or Password field.

OnlineUser

Specifies the users online ID in the format: userid@company.com.

Office365UserRealm

It is used to query online user ID information. Parameter is optional and should not be specified in most cases. The default value is:

https://login.microsoftonline.com/pp910/GetUserRealm.srf

WsTrustVersion

This setting is optional and used to set the WS-Trust version which defines the message format of the Secure Token Server authentication token. The possible values are "WSTrustFeb2005" or "WSTrust13". The default value is "WSTrustFeb2005".

AdfsEndpointUrl

This is used to define the local ADFS server endpoint URL for issuing an ADFS token. By default, it is queried from the service at the Office365UserRealm by the online user ID. This parameter is optional.

ADFSIntegratedWindows

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

It is the authentication method for accessing Office365 using ADFS with current user credentials. Before version 7.8.0.0, connections run via MMC were using the currently logged on user and scheduled connections were using the Scheduling Service account. Now all connections are using the Backend Service account instead.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

OnlineUser

Specifies the user's online ID in the format: userid@company.com.

Office365UserRealm

It is used to query online user id information. Parameter is optional and should not be specified in most cases. Default value is:

https://login.microsoftonline.com/GetUserRealm.srf

WSTRUSTVERSION

This setting is optional and used to set the WS-Trust version which defines the message format of the Secure Token Server authentication token. The possible values are "WSTrustFeb2005" or "WSTrust13". The default value is "WSTrustFeb2005".

ADFSENDPOINTURL

This is used to define the local ADFS server endpoint URL for issuing an ADFS token. By default, it is queried from the service at the Office365UserRealm by the online user id. This parameter is optional.

ADFSOnPremise

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is the authentication method for accessing On-Premises-Sharepoint servers using ADFS.

Related connection-string settings:

User

This part of the connection string specifies the username.

Password

This defines the password of the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

WsTrustVersion

This setting is optional and used to set the WS-Trust version which defines the message format of the Secure Token Server authentication token. The possible values are "WSTrustFeb2005" or "WSTrust13". The default value is "WSTrustFeb2005".

AdfsEndpointUrl

This is used to define the ADFS server endpoint URL for issuing an ADFS token. This setting is mandatory for On-Premises ADFS.

AdfsOnPremiseNtlm

(OData-Provider, XML-Provider, RSS-Provider)

This is for ADFS On-Premises authentications where an NTLM handshake happens, but instead of being integrated, it does require user/password credentials.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This part of the connection string specifies the username which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

OnPremiseCrmAdfs

(OData-Provider, XML-Provider, RSS-Provider)

This is a variation of **AdfsOnPremise** for an on-premises CRM system with an on-premises ADFS authentication.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This part of the connection string specifies the username which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

Anonymous

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider, SOAP-Provider)

If the target server is configured to support it, this method can be used to connect to the server without any authentication.

AzureSP

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is an authentication for Azure-hosted On-Premises SharePoint using the Azure Active Directory for credentials management.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User
This part of the connection string specifies the username for the account which is used to authenticate.

Password
This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

DynamicsCrmAdfs

(OData-Provider, XML-Provider, RSS-Provider)

This is the authentication method for accessing data on a Microsoft Dynamics CRM Online Instance using ADFS authentication.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User
This part of the connection string specifies the Windows Live ID which is used to authenticate.

Password
This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

OnlineUser
Specifies the users online ID in the format: userid@company.com.

Office365UserRealm
It is used to query online user id information. Parameter is optional and should not be specified in most cases. The default value is:

https://login.microsoftonline.com/GetUserRealm.srf

DynamicsCrmOnline

(OData-Provider, XML-Provider, RSS-Provider)

This is the authentication method for accessing data on a Microsoft Dynamics CRM Online Instance, preferably by using the OData provider, but it can also be used for XML content on the CRM server or RSS feeds.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This part of the connection string specifies the Office365 username for the account which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

DynamicsCrmOnlineLiveId

(OData-Provider, XML-Provider, RSS-Provider)

This is the authentication method for accessing data on a Microsoft Dynamics CRM Online Instance using Windows Live ID authentication.

Related connection-string settings:

User

This part of the connection string specifies the Windows Live ID which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the separate Password field.

ExactOnline

(OData-Provider, XML-Provider, RSS-Provider)

DEPRECATED!

This is a custom authentication to log into the ERP system Exact Online (exactonline.com). Since the service introduced mandatory two-way authentication, this method is marked as deprecated. Please try ExactOnlineOAuth2 instead.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This part of the connection string specifies the username which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

Important – Exact Online gives different login portals for customers in different regions. The authentication method is set to the USA by default (start.exactonline.com). If you are using Exact Online for another region (for example, start.exactonline.nl), you will need to modify the two login site URLs in the authentication file to the correct URL (see the Authentication Directory section above for more information on where this file is).

ExactOnlineOAuth2

(OData-Provider, XML-Provider, RSS-Provider)

This is a custom authentication to log into the ERP system Exact Online (exactonline.com). You need to set up an API key in the Exact Online App Development Portal:
https://apps.exactonline.com/us/en-US/Manage
Please note that you need to adjust the link if you are not using the US region of Exact Online.

The redirect URL needs to look like this: http://localhost:8910
You can use any available port, but then make sure to set the **Port** connection string parameter.

Afterwards, edit the newly created API key and you will see all the info needed for this authentication.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

ClientId

This is the client id of the API key in the Exact Online App Development Portal.

ClientSecret

This is the client secret of the API key in the Exact Online App Development Portal.

AuthorizeEndpoint

You only need to provide this setting if you are not using the US region of Exact Online (e.g., start.exactonline.nl). The default value is this: *https://start.exactonline.com/api/oauth2/auth*

TokenEndpoint

You only need to provide this setting if you are not using the US region of Exact Online (e.g., start.exactonline.de). The default value is this: https://start.exactonline.com/api/oauth2/token

Port

This is the port number you provided in the redirect URL earlier. It defaults to 8910.

IECookie

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

If an authentication cookie has been created using the Cookie Manager, this authentication method can be configured to authenticate by using the cookies. (See the Cookie Manager section for details)

IntegratedWindows

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

In versions before 7.8.0.0, this authentication method would use the currently logged on user to authenticate against the target server. Now it will use the Backend Service account to do so, which usually is the local system account or more specific accounts such as NetworkService. If this account has no access to the target system and the Backend Service account cannot be changed, it is recommended to use the Windows authentication instead to specify credentials.

This method has “WindowsIntegrated” as an alias, so either name can be used in a connection string.

MSOPartner

(SharePoint-Provider)

This is a custom authentication for Microsoft Online partner systems, where the login is handled by a custom login page (e.g., ‘login.partner.microsoftonline.cn’) instead of login.microsoftonline.com.

Related connection-string settings:

User

This part of the connection string specifies the username which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

Realm

This parameter is optional and should not be specified in most cases. It is the authentication interface of the SharePoint, where the Microsoft Online security token is converted into rtFa and FedAuth cookies. Default value is:

(Url-Authority) + /_forms/default.aspx?wa=wsignin1.0

NextCRMOnline

(OData-Provider, XML-Provider, RSS-Provider)

This is a custom authentication for the NextCRM system, but it also works with various other CRM providers (online, as well as on-premises).

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This part of the connection string specifies the username which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

NextCRMOnlineIntegrated

(OData-Provider, XML-Provider, RSS-Provider)

This is the integrated version of the NextCRMOnline authentication. Before version 7.8.0.0, connections run via MMC were using the currently logged on user and scheduled connections were using the Scheduling Service account. Now all connections are using the Backend Service account instead.

This authentication is implemented with the authentication construction kit.

OAuth2

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is a generic authentication for the OAuth2 standard. It requires a registered app in the target system. The details may vary, but when asked for a redirect URL, it always needs to be http://localhost:myPort (for example: http://localhost:8910), where myPort is an available port on the system that is executing this authentication method.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

ClientId

This parameter is mandatory and is used to identify the registered app in the target system.

ClientSecret

If the registered app provides a client secret, this is mandatory. Otherwise, this is irrelevant.

Scope

Depending on the target system, this is mandatory or irrelevant. It is used to state the requested permissions.

Example:

Scope=https://www.googleapis.com/auth/drive https://www.googleapis.com/auth/calendar;

AuthorizeEndpoint

This mandatory parameter must be set to the target system URL that is handling the first part of an OAuth2 token request.

Example: https://login.microsoftonline.com/common/oauth2/authorize

TokenEndpoint

This mandatory parameter must be set to the target system URL that is handling the second part and refreshing of an OAuth2 token request.

Example: https://login.microsoftonline.com/common/oauth2/token

Port

If the app registration asked for a redirect URL, this needs to be set to the same port. Otherwise, this is optional and will be set to an available port by default.

OAuth2AzureAD

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is a variation of the OAuth2 method, optimized for the Azure Active Directory.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

ClientId

This parameter is mandatory and is used to identify the registered app in the Azure Active Directory.

Port

As the Azure Active Directory app registration requires a redirect URL, this parameter is mandatory and must be set to the same port as in the redirect URL. The default value is 50287.

ResourceUri

The Azure Active Directory uses this parameter to identify the target application of the access token.

Example (OneDrive for Business): https://layer2-my.sharepoint.com/

AuthorizeEndpoint

This parameter is optional and only must be given if the Azure Active Directory does not use the default endpoints. This endpoint handles the first part of an access token request.

Example: https://login.microsoftonline.com/common/oauth2/authorize

TokenEndpoint

This parameter is optional and only must be given if the Azure Active Directory does not use the default endpoints. This endpoint handles the second and refreshing parts of an access token request.

Example: https://login.microsoftonline.com/common/oauth2/token

ModernCrmCustomApp

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is a variation of the Microsoft_Modern_CRM method, where the user must provide their own app registration.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

ClientId

This parameter is mandatory and is used to identify the registered app in the Azure Active Directory.

ClientSecret

This parameter is mandatory and comes from the Azure Active Directory app registration.

Port

As the Azure Active Directory app registration requires a redirect URL, this parameter is mandatory and must be set to the same port as in the redirect URL.

MSGraphViaAAD

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is a variation of the OAuth2 method, optimized for Microsoft Graph via Azure Active Directory.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

ClientId

This parameter is mandatory and is used to identify the registered app in the Azure Active Directory.

Port

As the Azure Active Directory app registration requires a redirect URL, this parameter is mandatory and must be set to the same port as in the redirect URL. The default value is 50287.

Scope

The scope defines which access rights you want to request. The scope offline_access is automatically added to any given scope as it is required to receive a refresh token.

Example (Groups): group.read.all

AuthorizeEndpoint

This parameter is optional and only must be given if the Azure Active Directory does not use the default endpoints. This endpoint handles the first part of an access token request.

Example: https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize

TokenEndpoint

This parameter is optional and only must be given if the Azure Active Directory does not use the default endpoints. This endpoint handles the second and refreshing parts of an access token request.

Example: https://login.microsoftonline.com/organizations/oauth2/v2.0/token

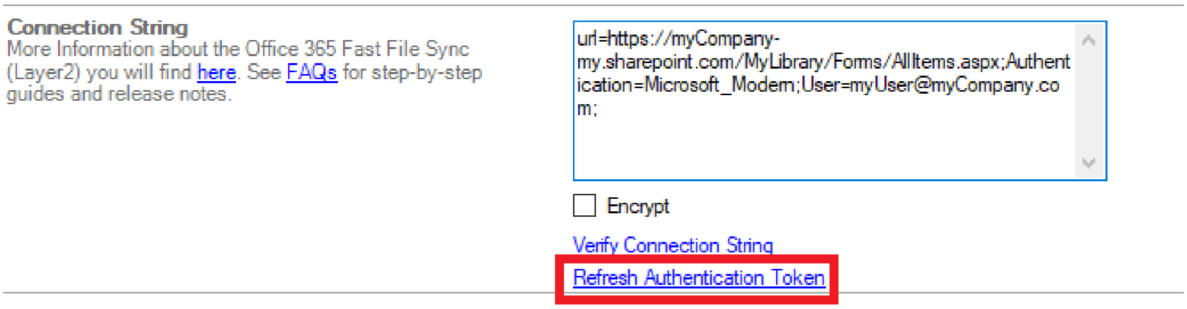
Microsoft_Modern

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is the default authentication method to access Microsoft Office 365 instances and should work in most cases, even if the SharePoint site is connected to an ADFS.

This authentication supports Multifactor Authentication and does not need any further connection string settings, other than the URL of the connected system.

The authentication needs to be setup in the UI.



Connection String
More Information about the Office 365 Fast File Sync (Layer2) you will find here. See FAQs for step-by-step guides and release notes.

```
url=https://myCompany-my.sharepoint.com/MyLibrary/Forms/AllItems.aspx;Authentication=Microsoft_Modern;User=myUser@myCompany.com;
```

☐ Encrypt

Figure 79 – Link in the UI to refresh the authentication token for Microsoft_Modern authentication

It will show a popup from Microsoft prompting to login. Then the popup will list requested permissions to be accepted by the user. The Layer2 Cloud Connector will retrieve a token which will be saved to disk and used for subsequent synchronizations, which now can be performed without interaction by the scheduling service.

After some time, the token will expire and needs to be refreshed through the UI.

To create multiple connections to the same URL, but with different users, it is possible to specify the User setting in the connection string, so that the token will be saved for that specific user.

Microsoft_Modern_Admin

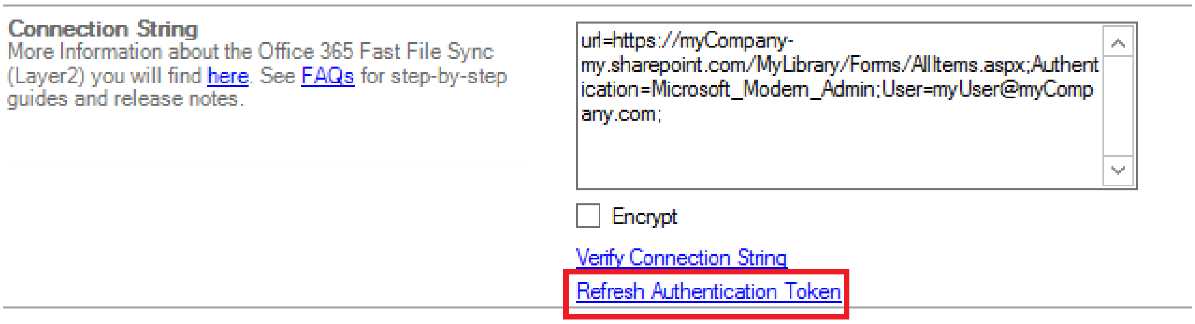
(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This authentication method can be used by administrator accounts to perform certain actions that require administrative permissions, e. g.: the **modified** field in a SharePoint list can only be changed with admin permissions.

Microsoft_Modern_Admin accesses Microsoft Office 365 instances and should work in most cases, even if the SharePoint site is connected to an ADFS.

The authentication supports Multifactor Authentication and does not need any further connection string settings, other than the URL of the connected system.

The authentication needs to be setup in the UI.



Connection String
More Information about the Office 365 Fast File Sync (Layer2) you will find here. See FAQs for step-by-step guides and release notes.

url=https://myCompany-my.sharepoint.com/MyLibrary/Forms/AllItems.aspx;Authentication=Microsoft_Modern_Admin;User=myUser@myCompany.com;

Figure 80 – Link in the UI to refresh the authentication token for Microsoft_Modern_Admin authentication

It will show a popup from Microsoft prompting to log in. Then the popup will list requested permissions to be accepted by the user. The Layer2 Cloud Connector will retrieve a token which will be saved to disk and used for subsequent synchronizations, which now can be performed without interaction by the scheduling service.

After some time, the token will expire and needs to be refreshed through the UI.

To create multiple connections to the same URL, but with different users, it is possible to specify the User setting in the connection string, so that the token will be saved for that specific user.

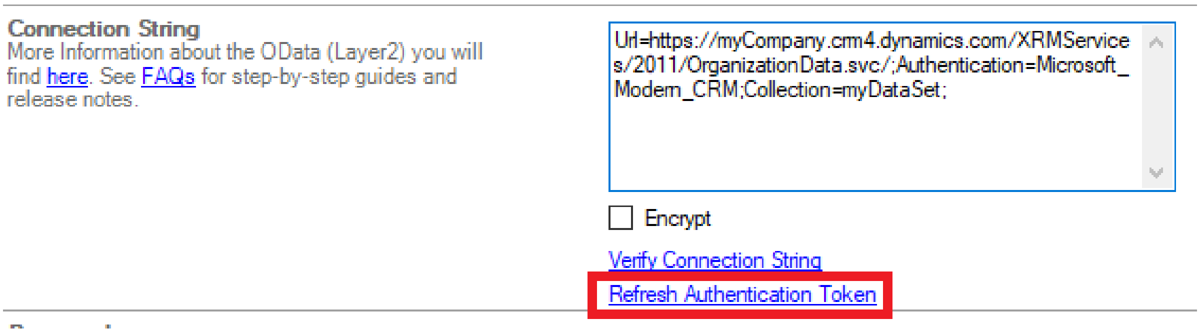
Microsoft_Modern_CRM

(OData-Provider, XML-Provider, RSS-Provider)

This is the default authentication method to access Microsoft Dynamics CRM Online instances and works in most cases.

This authentication supports Multifactor Authentication and does not need any further connection string settings, other than the URL of the connected system.

The authentication needs to be setup in the UI.



Connection String
More Information about the OData (Layer2) you will find here. See FAQs for step-by-step guides and release notes.

```
Url=https://myCompany.crm4.dynamics.com/XRMServices/2011/OrganizationData.svc/;Authentication=Microsoft_Modern_CRM;Collection=myDataSet;
```

Figure 81 – Link in the UI to refresh the authentication token for Microsoft_Modern_CRM authentication

It will show a popup from Microsoft prompting to login. Then the popup will list requested permissions to be accepted by the user. The Layer2 Cloud Connector will retrieve a token, which will be saved to disk and used for subsequent synchronizations, which now can be performed without interaction by the scheduling service.

After some time, the token will expire and needs to be refreshed through the UI.

To create multiple connections to the same URL, but with different users, it is possible to specify the User setting in the connection string, so that the token will be saved for that specific user.

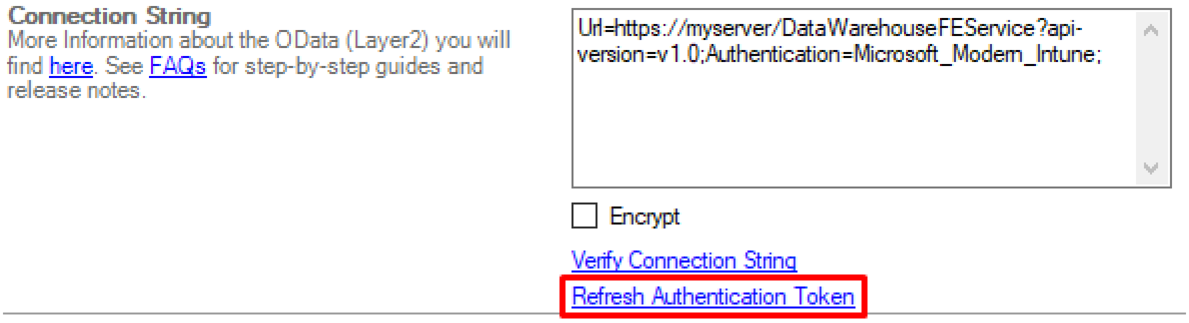
Microsoft_Modern_Intune

(OData-Provider, XML-Provider, RSS-Provider)

This is the default authentication method to access Azure Intune instances and works in most cases.

This authentication supports Multifactor Authentication and does not need any further connection string settings, other than the URL of the connected system.

The authentication needs to be setup in the UI.



Connection String
More Information about the OData (Layer2) you will find here. See FAQs for step-by-step guides and release notes.

Url=https://myserver/DataWarehouseFEService?api-version=v1.0;Authentication=Microsoft_Modern_Intune;

□ Encrypt

Verify Connection String
Refresh Authentication Token

Figure 82 – Link in the UI to refresh the authentication token for Microsoft_Modern_Intune authentication

It will show a popup from Microsoft prompting to login. Then the popup will list requested permissions to be accepted by the user. The Layer2 Cloud Connector will retrieve a token, which will be saved to disk and used for subsequent synchronizations, which now can be performed without interaction by the scheduling service.

After some time, the token will expire and needs to be refreshed through the UI.

To create multiple connections to the same URL, but with different users, it is possible to specify the User setting in the connection string, so that the token will be saved for that specific user.

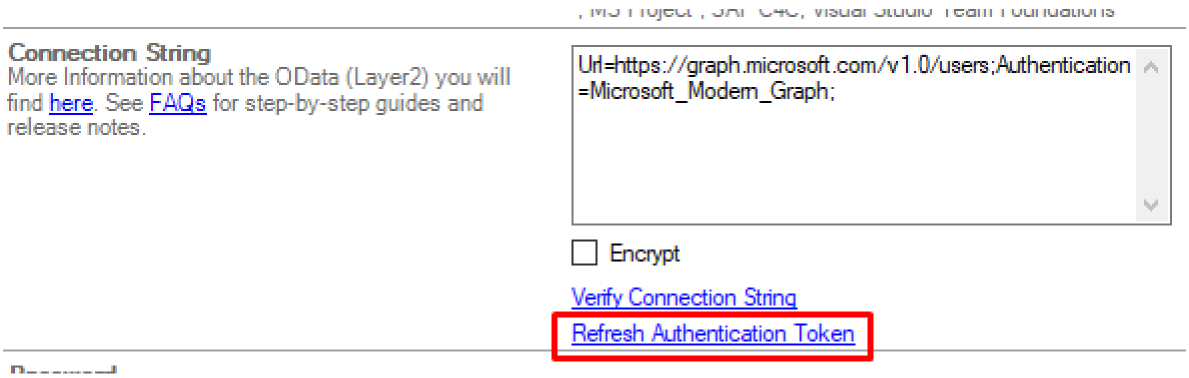
Microsoft_Modern_Graph

(OData-Provider, XML-Provider, RSS-Provider)

This is the default authentication method to access Microsoft Graph instances and works in most cases.

This authentication supports Multifactor Authentication and does not need any further connection string settings, other than the URL of the connected system.

The authentication needs to be setup in the UI.



Connection String
More Information about the OData (Layer2) you will find here. See FAQs for step-by-step guides and release notes.

Url=https://graph.microsoft.com/v1.0/users;Authentication=Microsoft_Modern_Graph;

□ Encrypt

Verify Connection String
Refresh Authentication Token

Figure 83 – Link in the UI to refresh the authentication token for Microsoft_Modern_Graph authentication

It will show a popup from Microsoft prompting to login. Then the popup will list requested permissions to be accepted by the user. The Layer2 Cloud Connector will retrieve a token, which will be saved to disk and used for subsequent synchronizations, which now can be performed without interaction by the scheduling service.

After some time, the token will expire and needs to be refreshed through the UI.

To create multiple connections to the same URL, but with different users, it is possible to specify the User setting in the connection string, so that the token will be saved for that specific user.

You can use Microsoft_Modern_Graph_Admin instead if your scenario requires Graph permissions with administrator privileges.

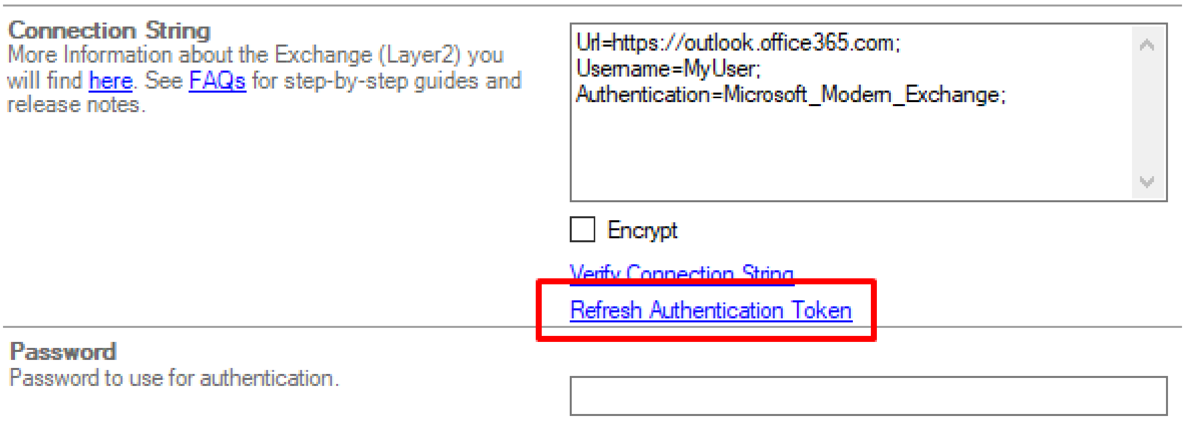
Microsoft_Modern_Exchange

(Exchange Provider)

This is the default authentication method to access Exchange Online tenants created after October 2020.

This authentication supports Multifactor Authentication and needs the username parameter. The server url is optional, but auto discovery is slow.

The authentication needs to be setup in the UI.



Connection String
More Information about the Exchange (Layer2) you will find here. See FAQs for step-by-step guides and release notes.

```
Url=https://outlook.office365.com;
Username=MyUser;
Authentication=Microsoft_Modern_Exchange;
```

□ Encrypt

Password
Password to use for authentication.

Figure 84 – Link in the UI to refresh the authentication token for Microsoft_Modern_Exchange authentication

It will show a popup from Microsoft prompting to login. Then the popup will list requested permissions to be accepted by the user. The Layer2 Cloud Connector will retrieve a token, which will be saved to disk and used for subsequent synchronizations, which now can be performed without interaction by the scheduling service.

After some time, the token will expire and needs to be refreshed through the UI.

To create multiple connections to the same URL, but with different users, it is possible to specify the User setting in the connection string, so that the token will be saved for that specific user.

If you want to access the “places” resources or the profiles of other users, you will have to use Microsoft_Modern_Exchange_Admin instead.

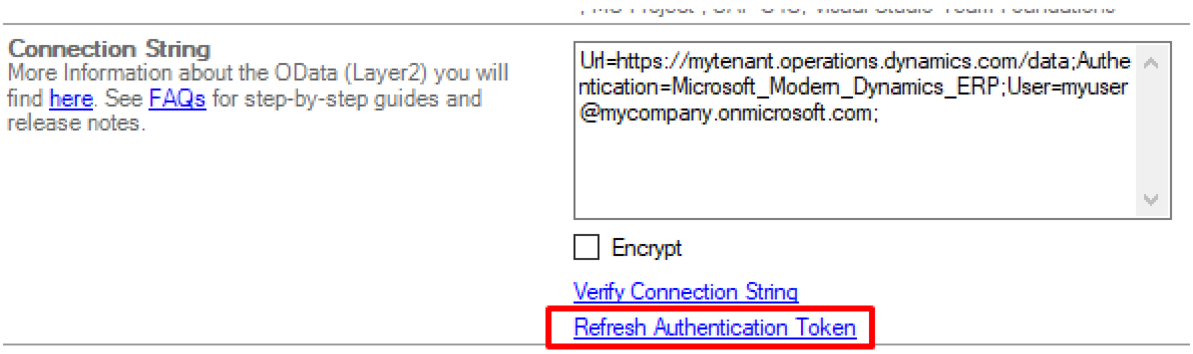
Microsoft_Modern_Dynamics_ERP

(OData-Provider, XML-Provider, RSS-Provider)

This is the default authentication method to access Microsoft Dynamics AX or Dynamics ERP instances and works in most cases.

This authentication supports Multifactor Authentication and does not need any further connection string settings, other than the URL of the connected system.

The authentication needs to be setup in the UI.



Connection String
More Information about the OData (Layer2) you will find here. See FAQs for step-by-step guides and release notes.

```
Url=https://mytenant.operations.dynamics.com/data;Authentication=Microsoft_Modern_Dynamics_ERP;User=myuser@mycompany.onmicrosoft.com;
```

□ Encrypt

Figure 85 – Link in the UI to refresh the authentication token for Microsoft_Modern_Dynamics_ERP authentication

It will show a popup from Microsoft prompting to login. Then the popup will list requested permissions to be accepted by the user. The Layer2 Cloud Connector will retrieve a token, which will be saved to disk and used for subsequent synchronizations, which now can be performed without interaction by the scheduling service.

After some time, the token will expire and needs to be refreshed through the UI.

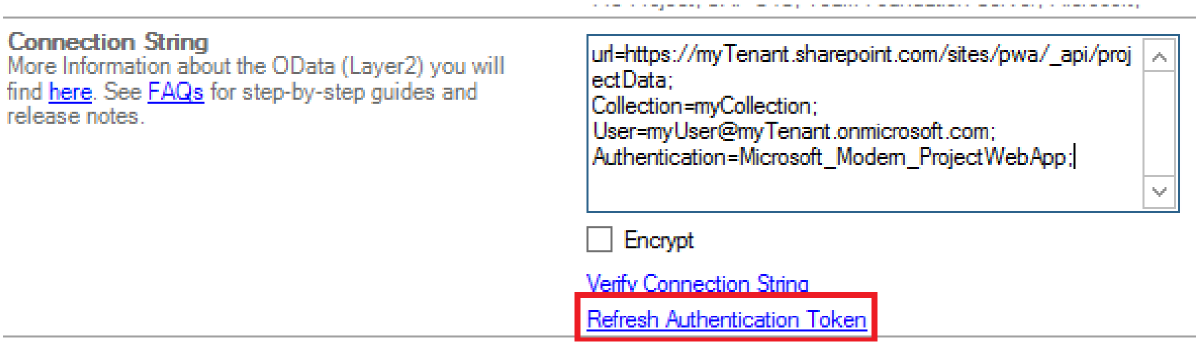
To create multiple connections to the same URL, but with different users, it is possible to specify the User setting in the connection string, so that the token will be saved for that specific user.

Microsoft_Modern_ProjectWebApp

(OData-Provider)

This is the default authentication method to access your Microsoft Project. This authentication supports Multifactor Authentication.

The authentication must be setup in the UI as shown below.



Connection String
More Information about the OData (Layer2) you will find here. See FAQs for step-by-step guides and release notes.

```
url=https://myTenant.sharepoint.com/sites/pwa/_api/projectData;
Collection=myCollection;
User=myUser@myTenant.onmicrosoft.com;
Authentication=Microsoft_Modern_ProjectWebApp;
```

□ Encrypt

Figure 86 – Link in the UI to refresh the authentication token for Microsoft_Modern_Dynamics_ERP authentication

It will show a popup from Microsoft prompting to login. Then the popup will list requested permissions to be accepted by the user. The Layer2 Cloud Connector will retrieve a token, which will be saved to disk and used for subsequent synchronizations, which now can be performed without interaction by the scheduling service.

After some time, the token will expire and needs to be refreshed through the UI.

To create multiple connections to the same URL, but with different users, it is possible to specify the User setting in the connection string, so that the token will be saved for that specific user.

Office365

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is another authentication method to access Microsoft Office 365 instances and should work in most cases, even if the SharePoint site is connected to an ADFS. This authentication **DOES NOT** support Multifactor Authentication:

Related connection-string settings:

User

This part of the connection string specifies the username for the account which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

SecureTokenService

(Only used in the legacy version)

This setting is optional and should not be specified in most cases. It defines the URL of the secure token service which is used for authentication. In most cases, this should be https://login.microsoftonline.com/extSTS.srf, which is the default.

SignInUrl

(Only used in the legacy version)

This setting is optional and should not be specified in most cases. It is the site collection relative URL which is used to sign in after the authentication token has been retrieved from the secure token service. If omitted, it will by default be /_forms/default.aspx?wa=wsignin1.0.

Realm

(Only used in the legacy version)

This setting needs to be specified, if the URL which is used to access the SharePoint Online instance is not the default URL. SharePoint Online default URLs have the format https://myCompany.sharepoint.com. This URL is used in two different contexts:

First it is used to identify the SharePoint instance to the secure token server (STS), in this context, the URL is called a Realm.

Second, it is used to locate and access the SharePoint instance, for example in a browser, as a normal URL. If a different URL, then the SharePoint Online default URL has been established to access the SharePoint Online instance, the URL will be, for example, https://mySharepoint.myCompany.com, but the realm will still be https://myCompany.sharepoint.com. In this case, the Layer2 Cloud Connector will no longer be able to infer the realm from the URL and the realm would need to be defined explicitly through this setting.

Office365viaGoDaddy

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is a custom variation of the Office365 authentication for those who are using godaddy.com to authenticate against their SharePoint Online.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

OnlineUser

This part of the connection string specifies the username for the account which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

LandingPage

This setting is optional and should not be specified in most cases. It defines the URL to the login form or landing page for non-authenticated users if it is not possible to detect that automatically.

Office365viaOkta

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is a custom variation of the Office365 authentication for those who are using okta.com to authenticate against their SharePoint Online.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

OnlineUser

This part of the connection string specifies the username for the account which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

LandingPage

This setting is optional and should not be specified in most cases. It defines the URL to the login form or landing page for non-authenticated users if it is not possible to detect that automatically.

Office365viaPing

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is a custom variation of the Office365 authentication for those who are using the SSO service “Ping” to authenticate against their SharePoint Online.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

OnlineUser

This part of the connection string specifies the username that helps SharePoint Online identifying you as a Ping user. You would use this username on the Microsoft Online login page to be redirected to Ping.

User

This defines the username of the account used to login via Ping.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

LandingPage

This setting is optional and should not be specified in most cases. It defines the URL to the login form or landing page for non-authenticated users if it is not possible to detect that automatically.

Office365viaOfficeOnline

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is a custom authentication for those who are logging into their SharePoint via “OfficeOnline”.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This defines the username of the account used to login just like in your browser.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

LandingPage

This setting is optional and should not be specified in most cases. It defines the URL to the login form or landing page for non-authenticated users if it is not possible to detect that automatically.

Office365withSPRedirect

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider)

This is a custom variation of the Office365 authentication for cases where the authentication process is redirected to a second SharePoint before receiving authentication cookies from the target SharePoint. A good indication of this scenario is that the login page (login.microsoftonline.com) has a different SharePoint as “wreply” argument.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This part of the connection string specifies the username for the account which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the separate Password field.

LandingPage

This setting is optional and should not be specified in most cases. It defines the URL to the login form or landing page for non-authenticated users if it is not possible to detect that automatically.

RMUnify

(SharePoint-Provider)

This is a custom authentication for RM Unify, which uses a combination of STS-provided SAML-Bearer-Token and other cookie-based tokens.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This part of the connection string specifies the RM Unify username which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

PortalUrl

This is a customer-specific URL used to access the RM Unify portal.

Example Value:

http://ccs.rmunify.com

SignInUrl

This is used to provide Microsoft Online with the required information for association with the used SharePoint instance. In most cases it can be acquired by manually logging into the RM Unify portal and searching for a SharePoint tile. (“MySite” for example).

Example Value:
https://login.microsoftonline.com/login.srf?wa=wsignin1.0&wp=MBI_KEY&wreply=https:%2F%2Fcastlecourtcom-my.sharepoint.com%2F_layouts%2FMySite.aspx&whr=castlecourt.com&CBCXT=out

SAPCsrfHeader

(OData-Provider, XML-Provider, RSS-Provider)

This is an authentication that re-enables writing access to SAP systems after the addition of the cross-site request forgery token. It must be attached to an existing authentication as it won't work by itself.

This authentication is implemented with the authentication construction kit.

Example:

Authentication=Windows,SAPHeader

Consistency

(OData-Provider, XML-Provider)

This is an authentication that adds a ConsistencyLevel header to set the consistency level to "eventual". This is required by some Graph data queries and must be attached to an existing authentication as it won't work by itself.

This authentication is implemented with the authentication construction kit.

Example:

Authentication=Microsoft_Graph,Consistency

SharePoint_FBA

(SharePoint-Provider)

This uses the forms-based authentication provided by SharePoint but won't work if this feature is not enabled on the target system.

This authentication is implemented with the authentication construction kit.

Related connection-string settings:

User

This part of the connection string specifies the username which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

Windows

(OData-Provider, SharePoint-Provider, XML-Provider, RSS-Provider, FTP-Provider, SOAP-Provider)

This authentication method is using a Windows domain-account to authenticate against the target server. This covers Basic, NTLM, Negotiate, Kerberos and Digest.

We recently changed the process, so if this authentication fails, there is Windows_legacy as a fallback.

Related connection-string settings:

User

This part of the connection string specifies the username for the account which is used to authenticate in the format: DOMAIN\username.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

xRMLive

(OData-Provider, XML-Provider, RSS-Provider)

It is the authentication method for accessing data on a Microsoft Dynamics CRM Online Instance using xRMlive.com as a CRM provider.

Related connection-string settings:

User

This part of the connection string specifies the Windows Live ID which is used to authenticate.

Password

This defines the password for the account which is used to authenticate. It needs to be typed into the Connection String or Password field.

SignInUrl

This is used to request xRM authentication cookies based on the SAML token acquired from the STS. This parameter is optional and should not be specified in most cases. The default value is:

https://auth-2013.xrmlive.com/

Realm

This is the host address used to connect to xRMLive. This parameter is optional and should not be specified in most cases. The default value is acquired automatically.

SecureTokenService

This is used to exchange authentication information with the xRMLiveSecure Token Service (STS). This parameter is optional and should not be specified in most cases. The default value is acquired automatically.

SPOnlineAppOnly

(SharePoint-Provider)

This the authentication method for accessing data on a Microsoft SharePoint Online Instance using SharePoint add-in credentials. Microsoft documentation about this authentication method can be found here: https://docs.microsoft.com/en-us/sharepoint/dev/solution-guidance/security-apponly-azureacs

Related connection-string settings:

ClientId

This is the client id of the SharePoint add-in/app. It is a **mandatory** Setting.

TenantId

This is the tenant id of the accessed SharePoint instance. It can be determined by logging into the SharePoint instance, then opening https://login.microsoftonline.com/contoso.onmicrosoft.com/.well-known/openid-configuration (while "contoso" is replaced with the actual tenant name) and searching for "token_endpoint". The tenant id will be part of the URL found after that: https://login.microsoftonline.com/YourTenantId/oauth2/token. It is a **mandatory** Setting.

SPDomain

This is the tenant's name of the accessed SharePoint instance, e.g., "contoso" for https://contoso.sharepoint.com. By default, the authentication will automatically take the SharePoint tenant name from the URL connection string parameter. If the value must be different, this **optional** Setting can be used.

AutoRenaming

This is a feature of the Layer2 Data Provider for SharePoint that is enabled by default except for newly created connections to SharePoint Online (version >= 16.0.8405.1236) and can optionally be disabled in the connection string. If enabled, file and folder names that violate SharePoint naming restrictions are automatically renamed so that they can be uploaded.

It is not recommended to change the AutoRenaming settings for a connection between synchronizations as this can lead to duplication of files and folders that have been renamed previously and will now be handled differently.

Renaming includes:

- Escaping forbidden characters, character sequences, prefixes and suffixes for files and folders
- Shortening file names that violate the length restrictions
- Ensuring that the renaming result is still a unique file path

Folders are escaped in a way that allows un-escaping the name back to the original name on the source system. This is required to avoid files being copied to wrong folders during synchronization.

Files are just escaped and cannot be un-escaped to the original name. For the synchronization process, the information about the mapping “source-system file name” ⇔ “auto-renamed SharePoint file name” is maintained in the Metabase.

Note: This means losing the Metabase (for example by manually deleting it) will have the following effects:

- Files will be duplicated by the uniqueness check due to the association between un-escaped source and escaped target being lost (the “new” escaped file is not associated with the previously created escaped file)
- Folders will not be duplicated by the uniqueness check, but will throw errors instead as they are not renamed (to prevent folder structure changes)

Escaping of File and Folder Names

Escaping is done for files as well as folders. Which characters and sequences are escaped depends on the SharePoint version.

All SharePoint Versions except SharePoint Online

File and folder names may not end with any of the following strings:

- .files
- _files
- -Dateien
- -fichiers
- _bestanden
- -filer
- _file
- _archivos
- _arquivos
- _tiedostot
- _pliki
- _soubory
- _elemei
- _ficheiros
- _dosyalar
- _datoteke
- _fitxers

- _failid
- _fails
- _bylos
- _fajlovi
- _fitxategiak

The escaping process for those suffixes differs between folders and files.

Folders

The first two entries are escaped as follows:

-Dateien	+Dateien+
.files	+f+files+
_files	+g+files+

All other suffixes in the list are escaped by omitting the leading character and then adding a '+' at the beginning and the end.

Examples:
A folder named "myFolder.files" will be renamed to "myFolder+f+files+" in SharePoint.
A folder named "myFolder-fichiers" will be renamed to "myFolder+fichiers+" in SharePoint.

Files

For files only the first character of a forbidden suffix is replaced with a + character.

Example: A file "myFile.files" will be renamed to "myFile+files" in SharePoint.

SharePoint Online and SharePoint 2016

Folders

For SharePoint Online the following characters and sequences are escaped in folder names:

Character or Sequence	Escaped to	Sample
#	+1+	"my # folder" ⇔ "my +1+ folder"
%	+2+	"my % folder" ⇔ "my +2+ folder"
*****	+4+	"my * folder" ⇔ "my +4+ folder"
\	+7+	"my \ folder" ⇔ "my +7+ folder"

:	+8+	“my : folder” ⇔ “my +8+ folder”
<	+9+	“my < folder” ⇔ “my +9+ folder”
>	+a+	“my > folder” ⇔ “my +a+ folder”
?	+b+	“my ? folder” ⇔ “my +b+ folder”
|	+c+	“my | folder” ⇔ “my +c+ folder”
“	+d+	“my “ folder” ⇔ “my +d+ folder”
..	+.+	“my .. folder” ⇔ “my +.+ folder”
+	++	“my + folder” ⇔ “my ++ folder”

So, for example, if a folder named “my # folder” is uploaded from a file system to SharePoint, it is named “my +1+ folder” on SharePoint as # is a forbidden character. When reading from SharePoint, the folder name is un-escaped to “my # folder” again.

Files

For files, the forbidden characters and sequences are replaced with a single + character.
Example: A file “my # File.txt” will be renamed to “my + File.txt” in SharePoint.

These are the currently forbidden characters and sequences for files in SharePoint Online separated by a whitespace:
* \ : < > ? | “ .. +

The only forbidden file prefix (whitespace) is also replaced with a single + character.

As mentioned before, the relation between auto-renamed file name and original file name is maintained in the Metabase as renamed file names cannot be un-escaped.

SharePoint 2013

Folders

Character or Sequence	Escaped to	Sample
~	+0+	“my ~ folder” ⇔ “my +0+ folder”
#	+1+	“my # folder” ⇔ “my +1+ folder”

%	+2+	“my % folder” ⇔ “my +2+ folder”
&	+3+	“my & folder” ⇔ “my +3+ folder”
*****	+4+	“my * folder” ⇔ “my +4+ folder”
{	+5+	“my { folder” ⇔ “my +5+ folder”
}	+6+	“my } folder” ⇔ “my +6+ folder”
\	+7+	“my \ folder” ⇔ “my +7+ folder”
:	+8+	“my : folder” ⇔ “my +8+ folder”
<	+9+	“my < folder” ⇔ “my +9+ folder”
>	+a+	“my > folder” ⇔ “my +a+ folder”
?	+b+	“my ? folder” ⇔ “my +b+ folder”
|	+c+	“my | folder” ⇔ “my +c+ folder”
“	+d+	“my “ folder” ⇔ “my +d+ folder”
..	+.+	“my .. folder” ⇔ “my +.+ folder”
+	++	“my + folder” ⇔ “my ++ folder”

Prefix	Escaped to	Sample
.	**+e+**	**“.my folder” ⇔ “+e+my folder”**
<whitespace>	**+h+**	**“ my folder” ⇔ “+h+my folder”**

Files

For files the forbidden characters, sequences and prefixes are replaced with a single + character.

Example: A file “my # File.txt” will be renamed to “my + File.txt” in SharePoint.

As mentioned before, the relation between auto-renamed file name and original file name is maintained in the Metabase as renamed file names cannot be un-escaped.

Shortening of File Names

SharePoint restricts the length of these items:

- Full URL
- File name
- Folder name

These restrictions are version specific.

In case a folder violates these restrictions, an error is raised so that you can identify the conflicting folder and rename it to a shorter name.

In case a file violates these restrictions, the Layer2 Cloud Connector shortens the file name step-by-step until it conforms to the restrictions, if possible. If this is not possible, an error is raised so that you can identify the conflicting file and rename it.

The relation between the shortened and the original file name is maintained in the Metabase.

Ensuring a Unique File Name

If a file name needs to be escaped or shortened, the Layer2 Cloud Connector ensures that the resulting new file name is unique. In case a file with the new file name already exists, it appends a number.

The relation between the modified and the original file name is maintained in the Metabase.

Logging

When a file or folder name has been renamed during the synchronization process, this is logged in the log files with log level *Debug*. Here is an example of what the messages look like:

NamingEnforcer: The folder path '/one +9+ two' was unescaped to '/one < two'.

Support

Online FAQs

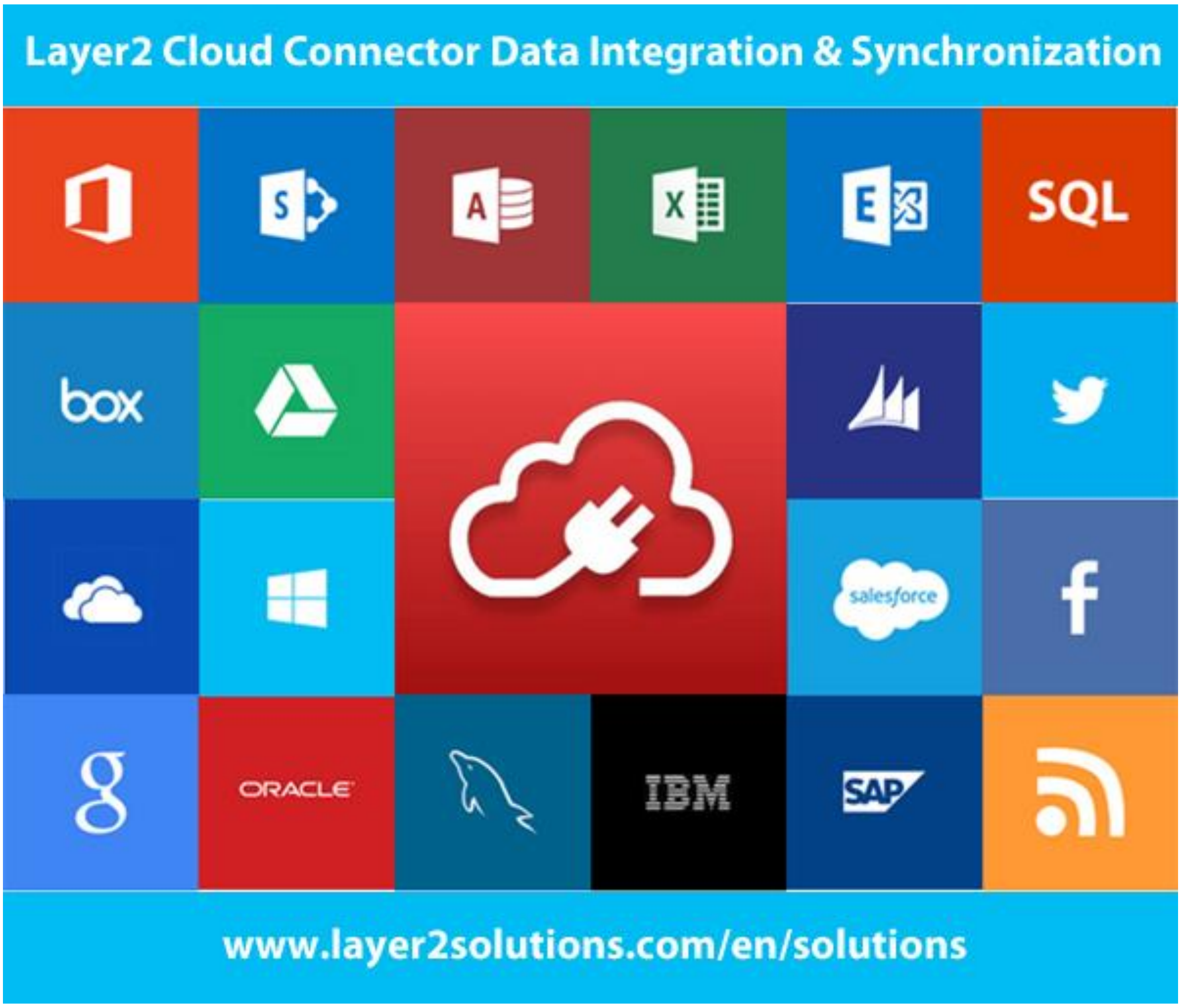
Please find answers to frequently asked questions online:

https://www.layer2solutions.com/en/community/FAQs/general/Pages/default.aspx

https://www.layer2solutions.com/en/community/FAQs/cloud-connector/Pages/default.aspx

Common Scenarios

You will find common scenarios and supported systems/applications online here:
https://www.layer2solutions.com/en/solutions/Pages/default.aspx


Layer2 Cloud Connector Data Integration & Synchronization
SQL
box
salesforce
ORACLE
IBM
SAP
www.layer2solutions.com/en/solutions

Trial

Please visit the product home page to register for trial. You will then receive instructions on how to download and install the Layer2 Cloud Connector Shareware. If you are interested in evaluating the application with full features, please contact sales@layer2solutions.com to receive a time-limited license key.

Ordering

Please visit the product home page to order online. For specialty orders (such as volume packages) please contact sales@layer2solutions.com for a detailed quote.

Upgrade

When a new version of Layer2 Cloud Connector is released, we will announce the changes in the change log on the version history page. Please look at the release notes online before installation. Contact sales@layer2solutions.com to request a license upgrade, if required.

Before running the installer, make sure that the Layer2 Cloud Connector Connection Manager is closed. The installer will update the existing components. During the upgrade, all existing configuration-settings and connected data sets will be preserved. However, it will reset the properties of the Backend Service as well as the logging, so please change those back to your preferred settings once the upgrade is complete.

Migration

In the case that you need to migrate your existing Layer2 Cloud Connector installation to a new server or newer operating system, please see the instructions below to execute the migration successfully.

Part 1 - Installation and Configuration

Download and install the latest Layer2 Cloud Connector version onto your new server. If you were using any 3rd-party providers or ODBC configurations, install and configure the necessary drivers. Make sure the provider architectures (32- or 64-Bit) match your installed version of the Layer2 Cloud Connector, as they may have changed with the new installation (see the Data Providers section for more information).

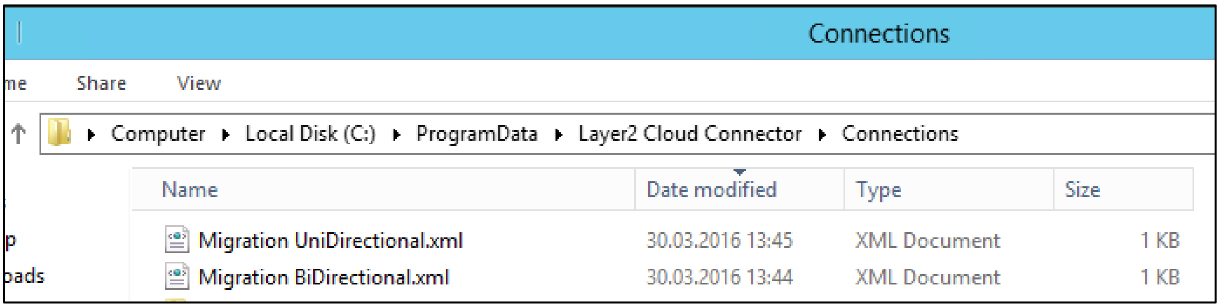
Then contact sales@layer2solutions.com to request a new license file with the new server's name and correct version (if upgrading from an older version). Your old license will become invalid when the new one is generated. Install the new license file per the instructions provided by Sales or see Installing a License.

Part 2 - Migrating Connections

For each connection, you will need to migrate the associated connection definition file and the Metabase files to the new server. Those can be found in the "Connections" and "Metabase" directories under one of these paths:

- C:\ProgramData\Layer2 Cloud Connector (for Windows Vista or higher)
- C:\Documents and Settings\All Users\Layer2 Cloud Connector

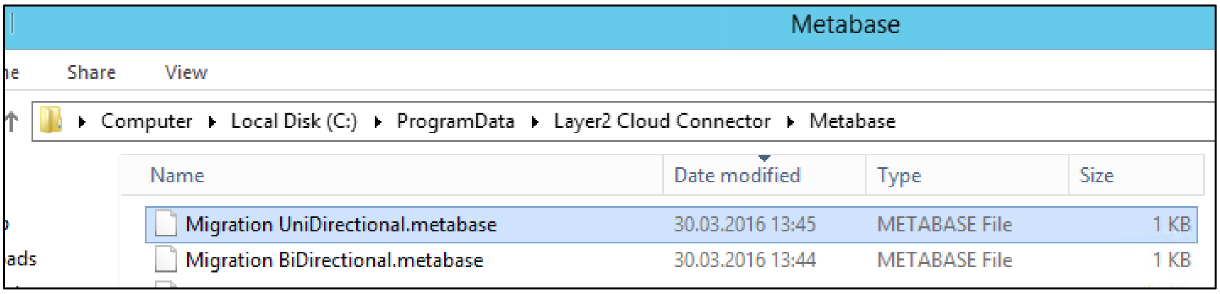
The path can be read from the environment variable %ALLUSERSPROFILE% (or %PROGRAMDATA% for Windows Vista and higher).



Computer ▸ Local Disk (C:) ▸ ProgramData ▸ Layer2 Cloud Connector ▸ Connections

Name	Date modified	Type	Size
Migration UniDirectional.xml	30.03.2016 13:45	XML Document	1 KB
Migration BiDirectional.xml	30.03.2016 13:44	XML Document	1 KB

Figure 87 - Example location of connection definitions



Computer ▸ Local Disk (C:) ▸ ProgramData ▸ Layer2 Cloud Connector ▸ Metabase

Name	Date modified	Type	Size
Migration UniDirectional.metabase	30.03.2016 13:45	METABASE File	1 KB
Migration BiDirectional.metabase	30.03.2016 13:44	METABASE File	1 KB

Figure 88 - Example location of Metabase files

You can also move the log and history files, but those are not mandatory for a successful migration.

IMPORTANT - If you had any custom alerts/notifications set up in the NLog.config file, then that file will also need to be migrated to the Logs folder on the new server to preserve your settings.

Migrating the connection content itself is as simple as copying and pasting the mentioned files to the same folders on your new system.

Part 3 – Validating the Connections

As a last check, go through and make sure that all your connections are still working as intended by launching a manual synchronization run for each migrated connection. Once this has been completed, then you can enable scheduling/start the Layer2 Cloud Connector Service to have the jobs run automatically (if appropriate).

After performing all three parts, your migration is done. If you encounter issues along the way or errors during the connection validation step, contact support@layer2solutions.com for assistance.

Contact

In case of general questions about the Layer2 Cloud Connector, contact sales@layer2solutions.com. If you have technical issues or errors with a connection, please contact support@layer2solutions.com.

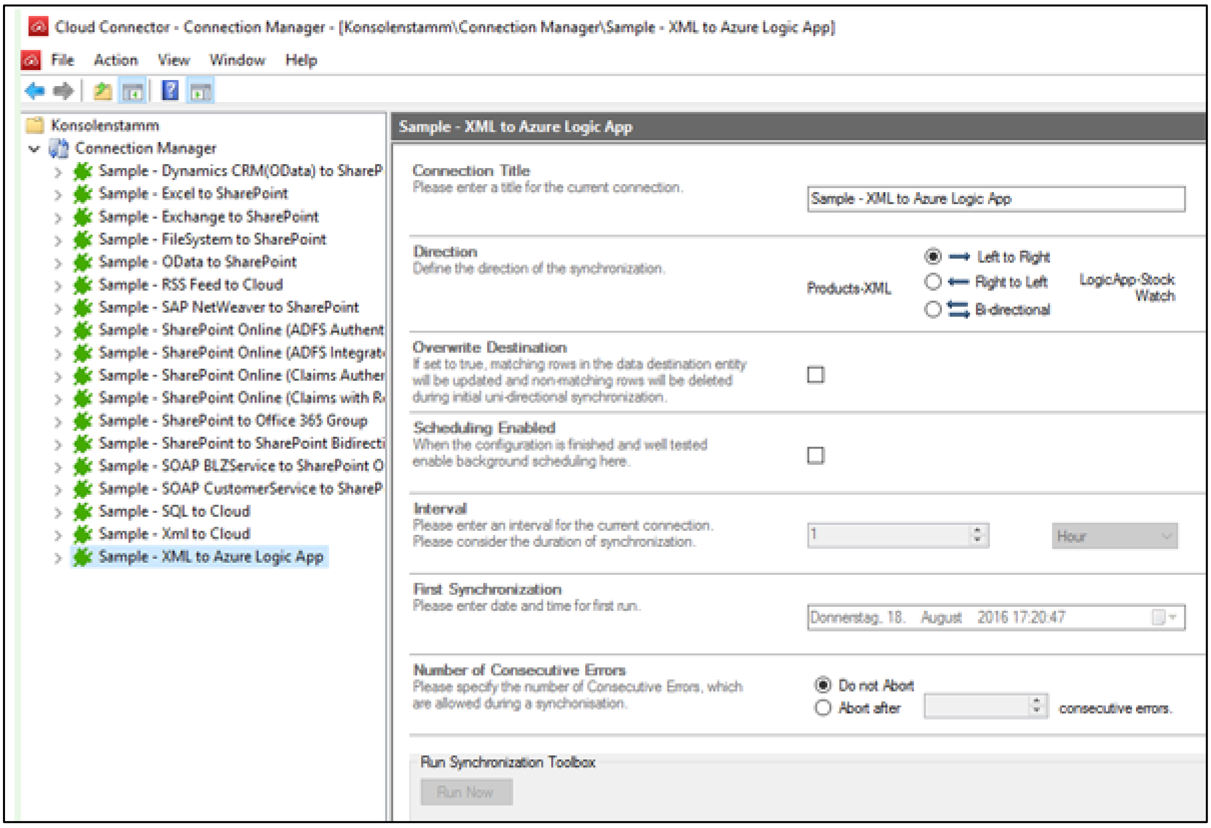
Appendix A – Examples

Start an Azure Logic Apps Workflow on Local XML Data Changes

In this example, we want to start an Azure Logic Apps workflow on local XML data changes.

Note: This sample works similar to a Microsoft Flow instead of an Azure Logic App.

Please copy your Layer2 Cloud Connector sample connection and adapt it as required. The sample XML file that is used here can be found folder *C:\ProgramData\Layer2 Cloud Connector\Sample Data* so that you can easily replay this example.


Cloud Connector - Connection Manager - [Konsolenstamm\Connection Manager\Sample - XML to Azure Logic App]
File Action View Window Help
Konsolenstamm
Connection Manager
Sample - Dynamics CRM(OData) to ShareP
Sample - Excel to SharePoint
Sample - Exchange to SharePoint
Sample - FileSystem to SharePoint
Sample - OData to SharePoint
Sample - RSS Feed to Cloud
Sample - SAP NetWeaver to SharePoint
Sample - SharePoint Online (ADFS Authent
Sample - SharePoint Online (ADFS Integrat
Sample - SharePoint Online (Claims Authen
Sample - SharePoint Online (Claims with R
Sample - SharePoint to Office 365 Group
Sample - SharePoint to SharePoint Bidirecti
Sample - SOAP BLZService to SharePoint O
Sample - SOAP CustomerService to ShareP
Sample - SQL to Cloud
Sample - Xml to Cloud
Sample - XML to Azure Logic App

Sample - XML to Azure Logic App

Connection Title
Please enter a title for the current connection.
Sample - XML to Azure Logic App

Direction
Define the direction of the synchronization.
Products-XML ● Left to Right ○ Right to Left ○ Bi-directional LogicApp-Stock Watch

Overwrite Destination
If set to true, matching rows in the data destination entity will be updated and non-matching rows will be deleted during initial uni-directional synchronization.
□

Scheduling Enabled
When the configuration is finished and well tested enable background scheduling here.
□

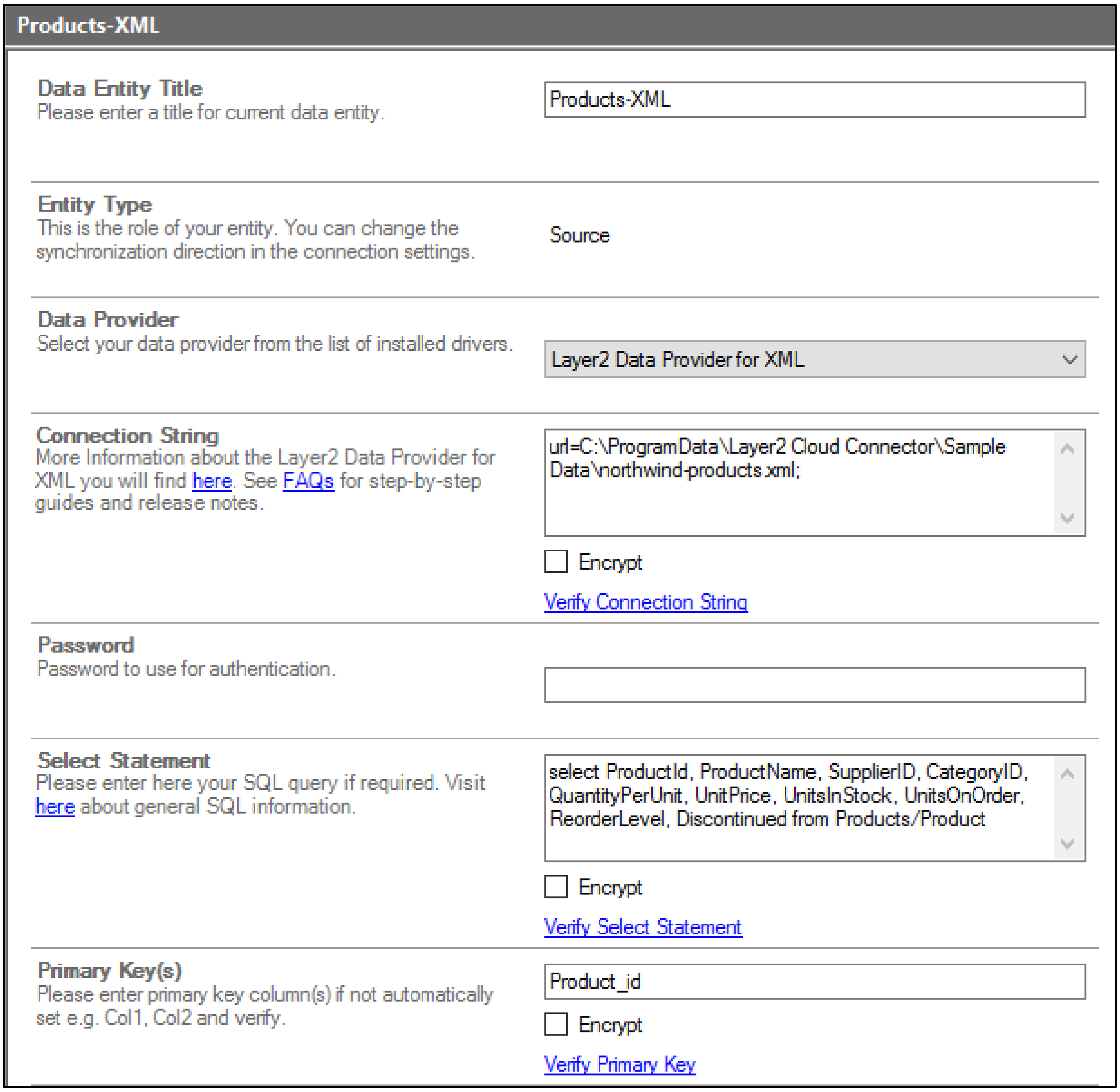
Interval
Please enter an interval for the current connection. Please consider the duration of synchronization.
1 Hour

First Synchronization
Please enter date and time for first run.
Donnerstag, 18. August 2016 17:20:47

Number of Consecutive Errors
Please specify the number of Consecutive Errors, which are allowed during a synchronisation.
● Do not Abort
○ Abort after consecutive errors.

Figure 89- Example connection to adapt for evaluation

Please configure your data source first. In this case, the example data source is a local XML file, but can be any supported data source. The Layer2 Data Provider for XML is used to query an XML file via XPath. Please note the primary key (column with unique values) in the result set. It is required for the data sync. You can also make use of several columns that are unique together, e.g., Col1, Col2, Col3, to make a combination primary key.



Products-XML

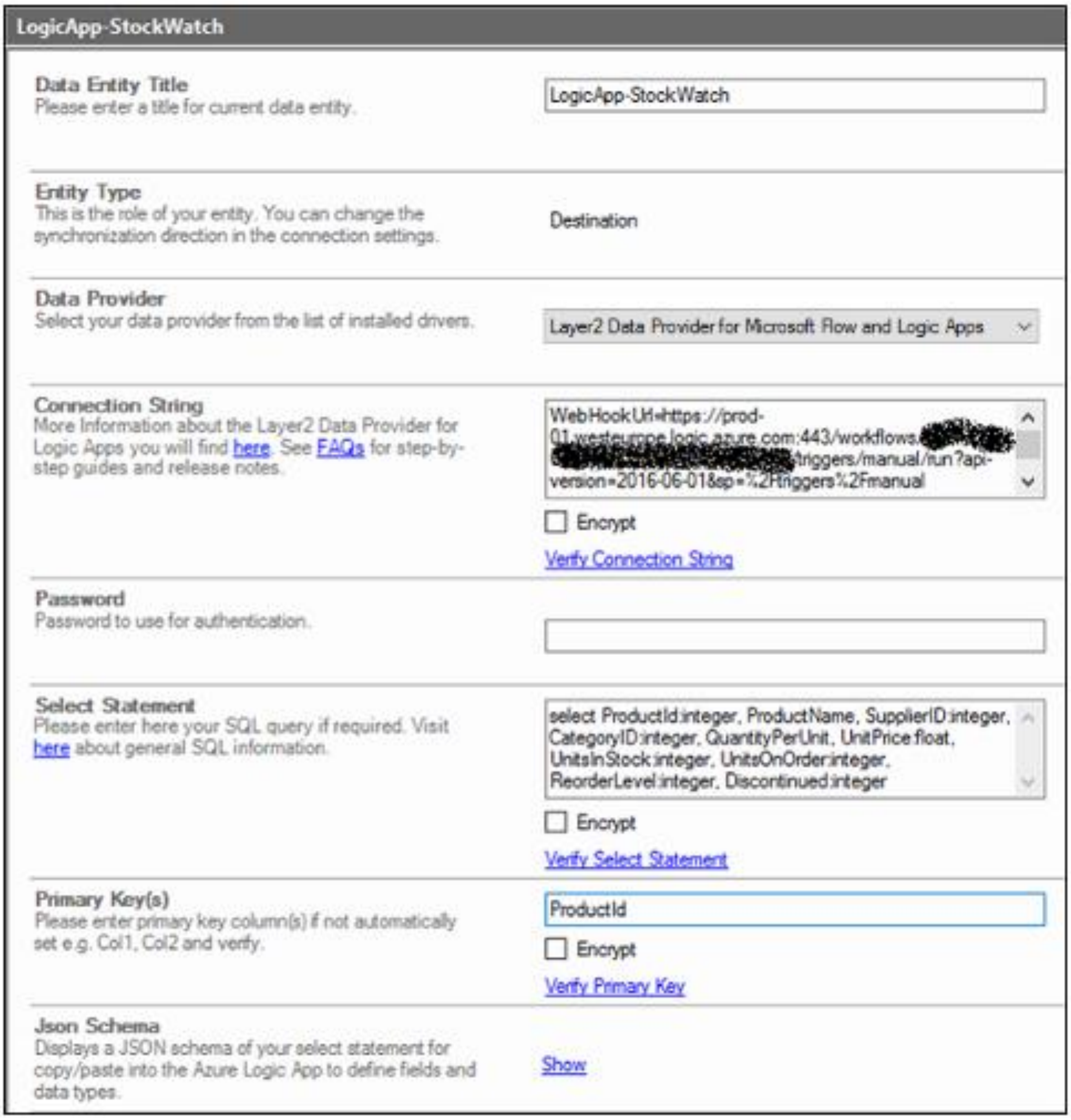
Data Entity Title Please enter a title for current data entity.	Products-XML
Entity Type This is the role of your entity. You can change the synchronization direction in the connection settings.	Source
Data Provider Select your data provider from the list of installed drivers.	Layer2 Data Provider for XML
Connection String More Information about the Layer2 Data Provider for XML you will find here. See FAQs for step-by-step guides and release notes.	url=C:\ProgramData\Layer2 Cloud Connector\Sample Data\northwind-products.xml; □ Encrypt Verify Connection String
Password Password to use for authentication.	
Select Statement Please enter here your SQL query if required. Visit here about general SQL information.	select ProductId, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued from Products/Product □ Encrypt Verify Select Statement
Primary Key(s) Please enter primary key column(s) if not automatically set e.g. Col1, Col2 and verify.	Product_id □ Encrypt Verify Primary Key

Figure 90 - Example data source using an XML file.

Next configure the Azure Logic Apps data destination. Please select the Layer2 Data Provider for Azure Logic Apps for this. You will need a Web Hook URL for this (see below how to get it). Please also provide a symbolic SQL-like data query for data mapping and to declare data types, and a primary key. The SQL query is also used to generate and display a JSON schema of your data items required for the Azure Logic App.

You can use a notation similar to this:
Select ProductID: integer, ProductName, SupplierID:integer, UnitPrice:float

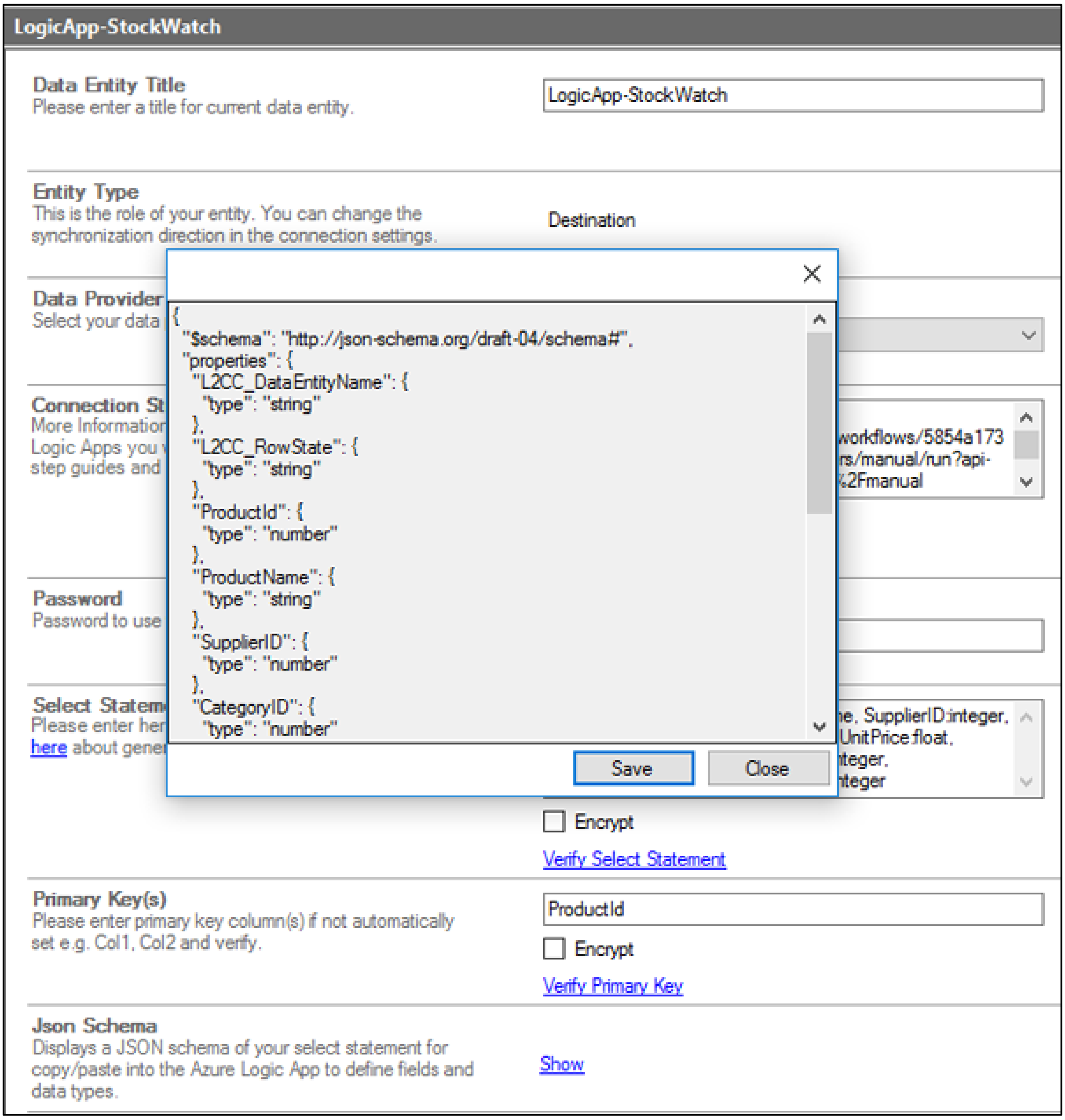
If you don't add any data type, “string” is assumed by default. Please look into the User's Guide for more data types.



LogicApp-StockWatch

Data Entity Title Please enter a title for current data entity.	LogicApp-StockWatch
Entity Type This is the role of your entity. You can change the synchronization direction in the connection settings.	Destination
Data Provider Select your data provider from the list of installed drivers.	Layer2 Data Provider for Microsoft Flow and Logic Apps
Connection String More Information about the Layer2 Data Provider for Logic Apps you will find here. See FAQs for step-by-step guides and release notes.	WebHookUrl=https://prod-01.westeurope.logic.azure.com:443/workflows/[redacted]/triggers/manual/run?api-version=2016-06-01&sp=%2Ftriggers%2Fmanual ☐ Encrypt Verify Connection String
Password Password to use for authentication.	
Select Statement Please enter here your SQL query if required. Visit here about general SQL information.	select ProductId:integer, ProductName, SupplierID:integer, CategoryID:integer, QuantityPerUnit, UnitPrice:float, UnitsInStock:integer, UnitsOnOrder:integer, ReorderLevel:integer, Discontinued:integer ☐ Encrypt Verify Select Statement
Primary Key(s) Please enter primary key column(s) if not automatically set e.g. Col1, Col2 and verify.	ProductId ☐ Encrypt Verify Primary Key
Json Schema Displays a JSON schema of your select statement for copy/paste into the Azure Logic App to define fields and data types.	Show

Figure 91 - Example configuration of a Azure Logic App data destination



LogicApp-StockWatch

Data Entity Title
Please enter a title for current data entity.

LogicApp-StockWatch

Entity Type
This is the role of your entity. You can change the synchronization direction in the connection settings.

Destination

Data Provider
Select your data

Connection St
More Informatior
Logic Apps you
step guides and

workflows/5854a173
rs/manual/run?api-
%2Fmanual

Password
Password to use

Select Statem
Please enter her
here about gener

ie, SupplierID:integer,
UnitPrice:float,
nteger,
nteger

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "properties": {
    "L2CC_DataEntityName": {
      "type": "string"
    },
    "L2CC_RowState": {
      "type": "string"
    },
    "ProductId": {
      "type": "number"
    },
    "ProductName": {
      "type": "string"
    },
    "SupplierID": {
      "type": "number"
    },
    "CategoryID": {
      "type": "number"
```

□ Encrypt

Verify Select Statement

Primary Key(s)
Please enter primary key column(s) if not automatically set e.g. Col1, Col2 and verify.

ProductId

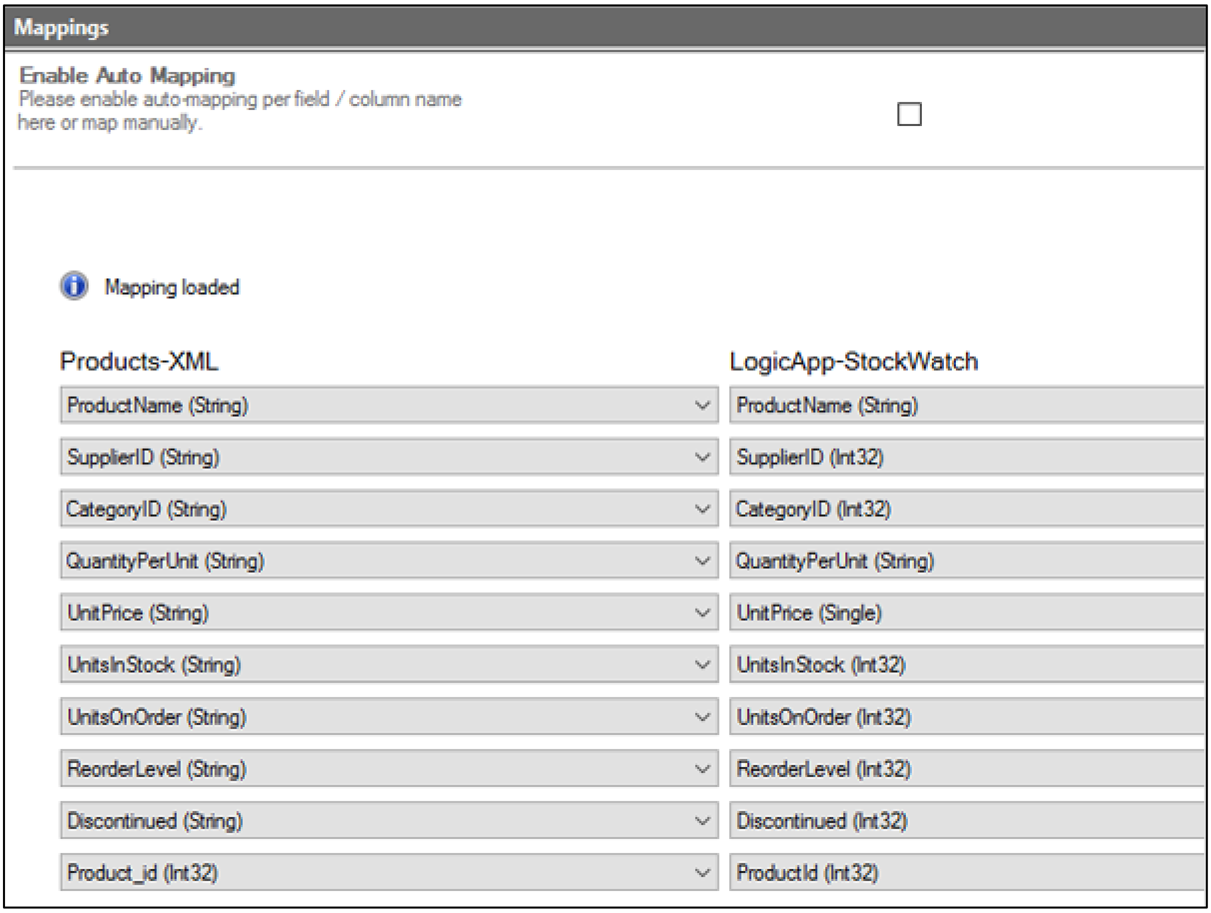
□ Encrypt

Verify Primary Key

Json Schema
Displays a JSON schema of your select statement for copy/paste into the Azure Logic App to define fields and data types.

Show

Figure 92 - The Layer2 Cloud Connector provides the JSON schema of the data source as required in Azure Logic Apps to connect to the data source and map fields



Mappings

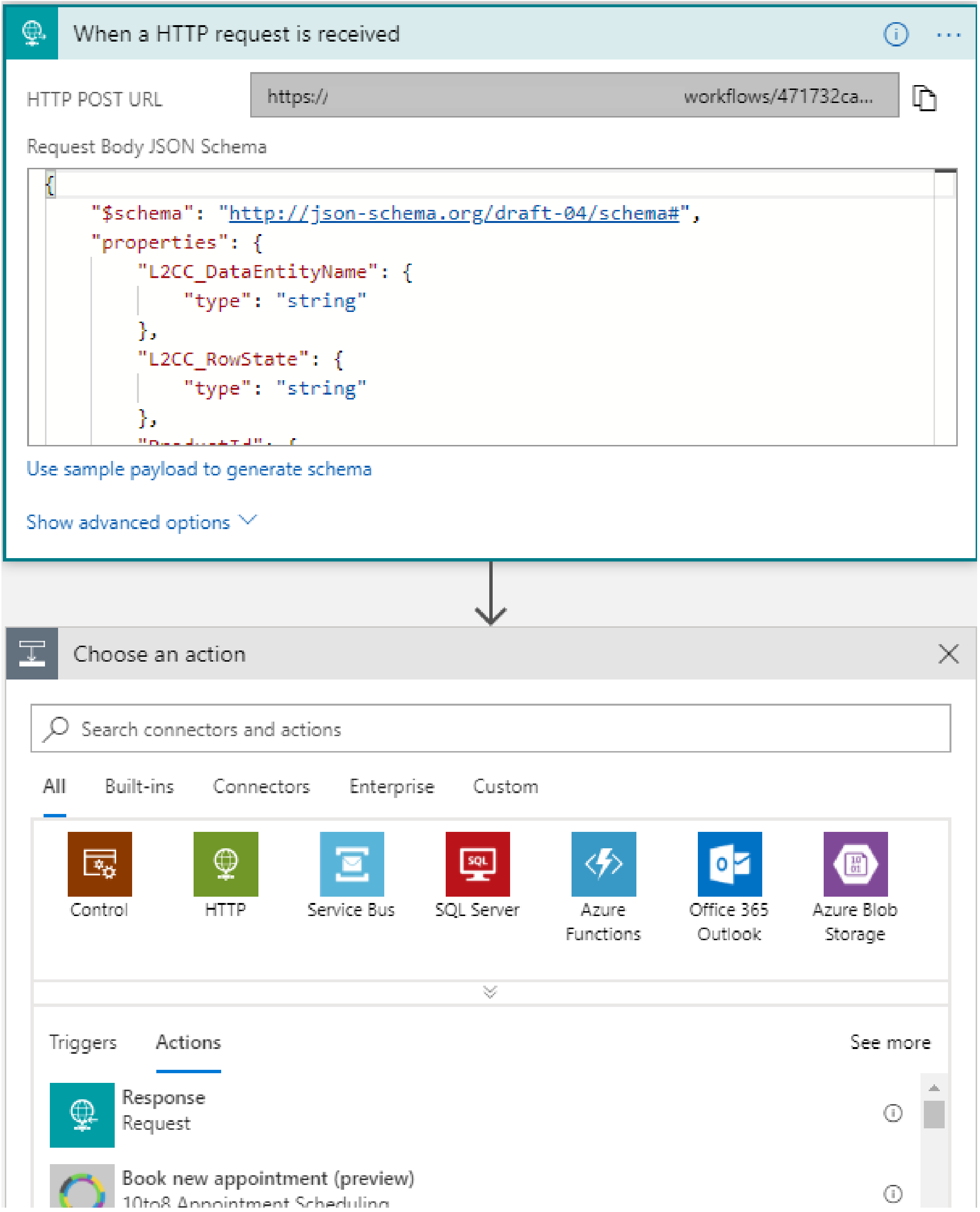
Enable Auto Mapping
Please enable auto-mapping per field / column name here or map manually. □

Mapping loaded

Products-XML	LogicApp-StockWatch
ProductName (String)	ProductName (String)
SupplierID (String)	SupplierID (Int32)
CategoryID (String)	CategoryID (Int32)
QuantityPerUnit (String)	QuantityPerUnit (String)
UnitPrice (String)	UnitPrice (Single)
UnitsInStock (String)	UnitsInStock (Int32)
UnitsOnOrder (String)	UnitsOnOrder (Int32)
ReorderLevel (String)	ReorderLevel (Int32)
Discontinued (String)	Discontinued (Int32)
Product_id (Int32)	ProductId (Int32)

Figure 93 - Example mapping between XML and Logic App data entities

Next create your Azure Logic App as follows:



HTTP POST URL: https:// workflows/471732ca...

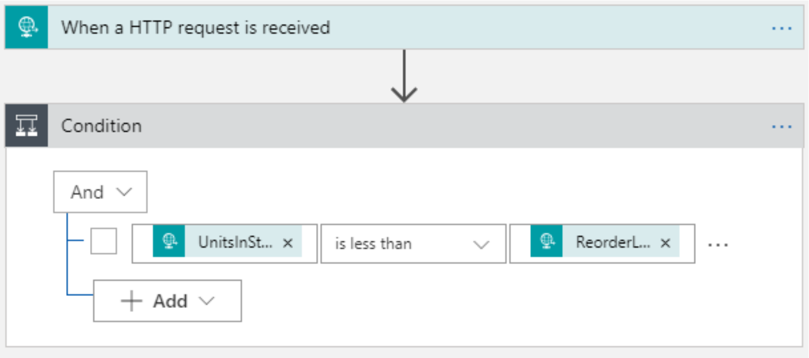
Request Body JSON Schema

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "properties": {
        "L2CC_DataEntityName": {
            "type": "string"
        },
        "L2CC_RowState": {
            "type": "string"
        },
        "ProductId": {
```

Figure 94 - Configure the Azure Logic Apps HTTP POST end-point

You will find the Webhook URL in the definition of the HTTP Post. Also paste your JSON schema to the Request Body section.

Now you are ready to add some specific logic. This example will send an email notification via Azure Logic App if the number of units in stock is below the reorder level in the local data source.



When a HTTP request is received

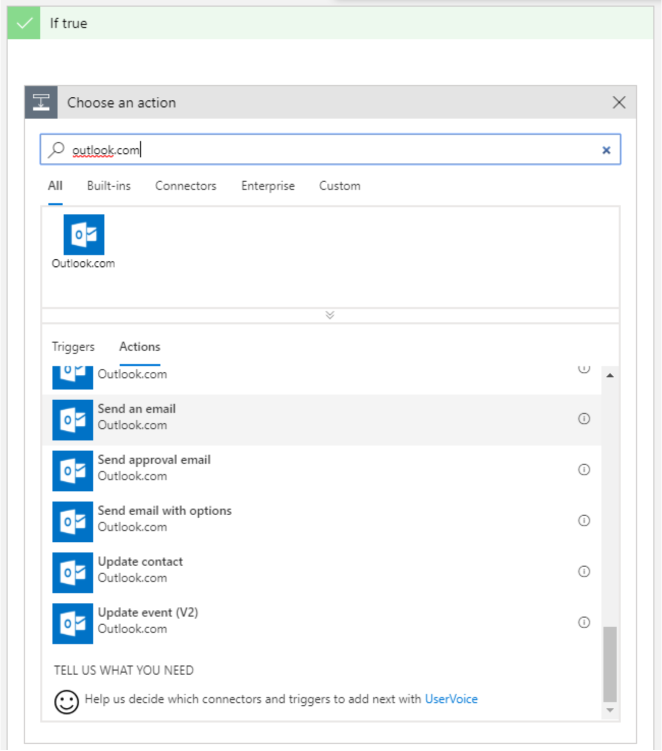
Condition

And

□ UnitsInSt... × is less than ReorderL... ×

+ Add

Figure 95 - Apply a logical expression in Azure Logic Apps based on the changed local data record



If true

Choose an action

outlook.com

All Built-ins Connectors Enterprise Custom

Outlook.com

Triggers Actions

Outlook.com

Send an email
Outlook.com

Send approval email
Outlook.com

Send email with options
Outlook.com

Update contact
Outlook.com

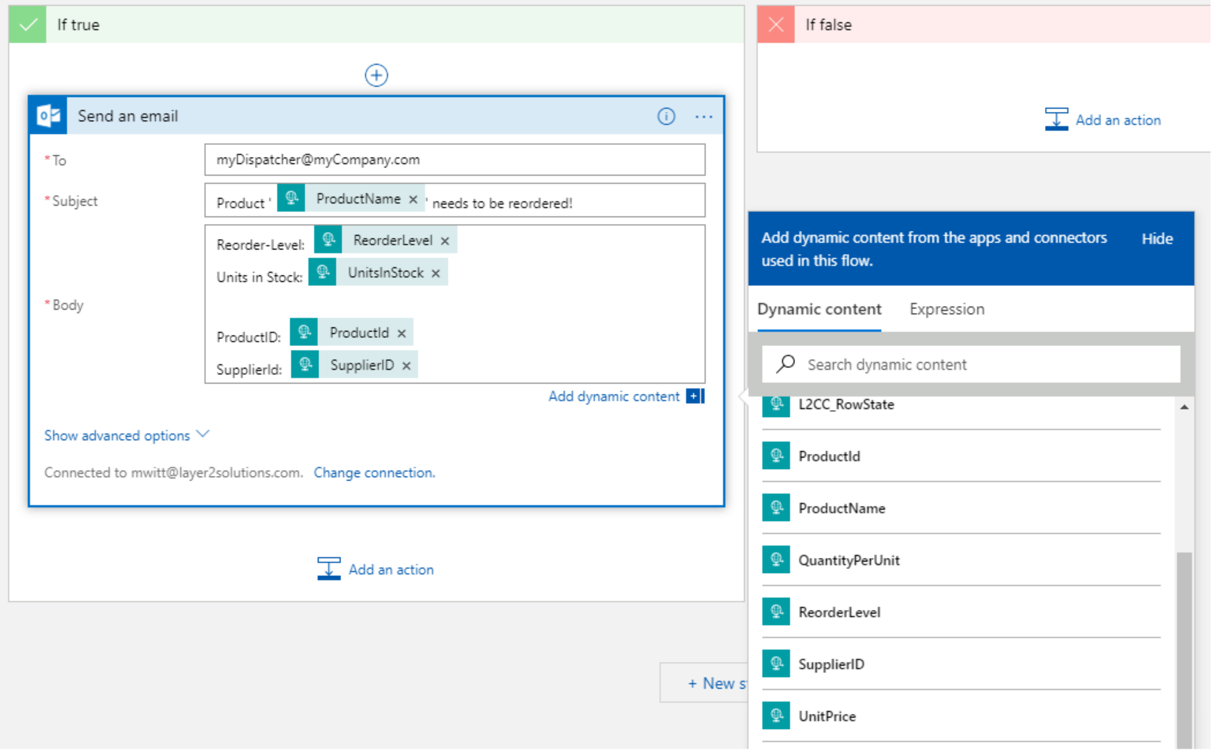
Update event (V2)
Outlook.com

TELL US WHAT YOU NEED

Help us decide which connectors and triggers to add next with UserVoice

Figure 96 - Select the "Send an email" action from the actions offered by Azure Logic Apps

You can configure the email based on the local data source record as provided by the Layer2 Cloud Connector.



*To: myDispatcher@myCompany.com

*Subject: Product ' ProductName ' needs to be reordered!

*Body:
Reorder-Level: ReorderLevel
Units in Stock: UnitsInStock

ProductID: ProductId
SupplierId: SupplierID

Add dynamic content

Show advanced options

Connected to mwitt@layer2solutions.com. Change connection.

Add dynamic content from the apps and connectors used in this flow. Hide

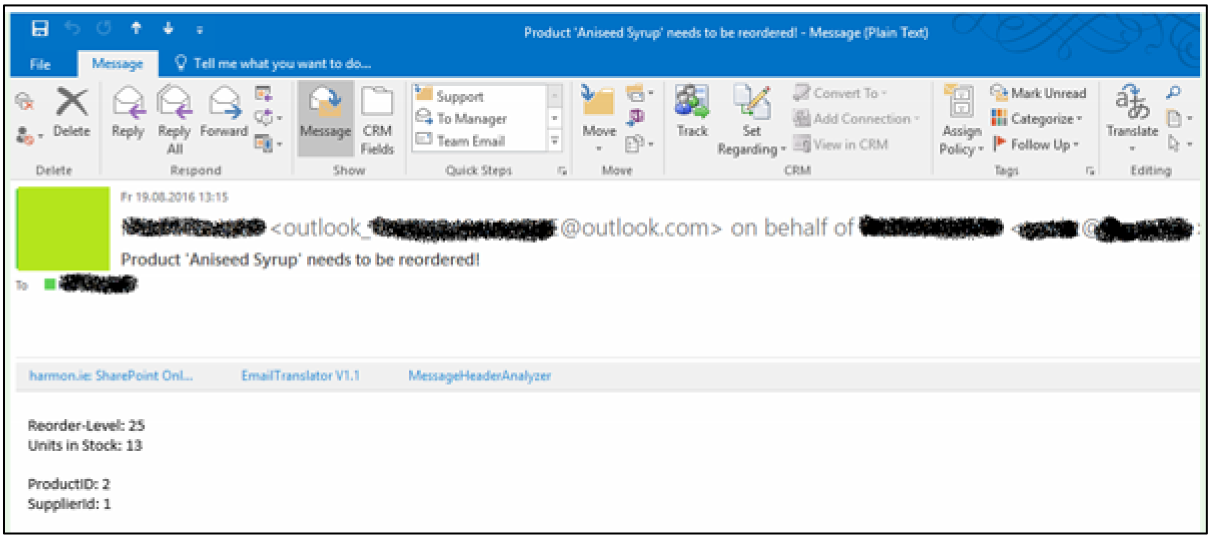
Dynamic content | Expression

Search dynamic content

- L2CC_RowState
- ProductId
- ProductName
- QuantityPerUnit
- ReorderLevel
- SupplierID
- UnitPrice

Figure 97 - Example email configuration to send notifications in Azure based on local data changes

You are now ready to run your Layer2 Cloud Connector connection: manually using the **Run Now** button, on-demand using the command line, or scheduled using the Windows service. If your condition is true, you will receive the following email from your connected Azure Logic App:



Fr 19.08.2016 13:15

[redacted] <outlook_[redacted]@outlook.com> on behalf of [redacted] <[redacted]@[redacted]>

Product 'Aniseed Syrup' needs to be reordered!

To [redacted]

Reorder-Level: 25
Units in Stock: 13

ProductID: 2
SupplierId: 1

Figure 98 - Example mail received when the condition is "true" for the changes

If you did not receive emails as expected from Azure, check your SPAM folder.