Syniti Solutions

Metadata Discovery 2.1

BackOffice Associates, LLC DBA Syniti
info@Syniti.com
www.Syniti.com

Table of Contents

Contents

Overview **1**

Metadata Scanner Capability Matrix 1

Configure Metadata Discovery **5**

Configure Knowledge Tier Download 5

Select Internal Job Queue for Background Jobs 6

Configure Object Dependencies 6

Configure Templates for Auto Generation 7

Generate Dependencies 8

Configure Metadata Exchange **8**

Metadata Exchange for SAP Information Steward 9

Metadata Exchange for Collibra 15

Metadata Exchange for ASG Data Intelligence 22

Metadata Exchange for the Syniti Stewardship Tier 25

Sending Rules to the Knowledge Tier 32

Retrieve Metadata Using the Open API Specification Parser 34

Retrieve Metadata Using the SAP HANA Scanner 35

Retrieve Metadata Using the Salesforce Scanner 35

Retrieve Metadata Using the XML Schema Reader 35

Additional Metadata Discovery Configuration Options 36

Copy a System Type Model 37

Configure Settings for Creating Application Module Hierarchy 37

Use Metadata Discovery **39**

Create Application Module Taxonomies 39

Access the Application Hierarchy Visualization 40

Locate Modules in the Application Hierarchy Visualization Using a Table Name or Transaction 42

Analyze Modules 42

Export Metadata 49

Create System with Technical Metadata from a System Type in the Knowledge Tier 49

Create Business Concepts 51
Send the Business Concept to the Syniti Catalog as a Term 53
Use Inherit Tables to Jump Start a Migration Project 54
Additional Configuration Menu Pages 54
Appendix 55
Use the Hierarchy Maintenance Plugin 55
Version History 56
Installation 57
Adding to a WebApp 57
Plugin Data Row Contract 58
Hierarchy Source Dataset 59
Hierarchy Target Table 60
Select Target Table 60
Use the Knowledge Tier API 3.0 Plugin 61
Version History 61
Installation 65
Technical Documentation 67
Extract Plugin Data Row Contracts 67
ExtractID 67
Username 67
Password 67
BasePath 67
ProxyAddress 67
ExtractID 68
Username 68
Password 68
BasePath 68
ProxyAddress 68
SystemID 68
ExtractID 68
Username 68

Password 68
BasePath 68
ProxyAddress 68
DatasetID 68
Inbound Plugins 68
RequestID 69
Username 69
Password 69
BasePath 69
ProxyAddress 69
HTTPVerb 69
Endpoint 69
RequestID 71
LinkID 71
id 71
http_code 71
api_response 71
Using the SharePoint Plugin 71
Version History 77
Installation 77
SharePoint:GroupDrives 77
Plugin Data Row Contract 78
TokenURL 78
ClientID 78
ClientSecret 78
Output Tables 78
SharePoint:FileUpload 79
Plugin Data Row Contract 80
RequestID 80
TokenURL 81
ClientID 81

ClientSecret 81
DriveID 81
LocalFileRef 81
TargetFileRef 81
Output table 82
SharePoint:ListDriveItems 83
Plugin Data Row Contract 83
RequestID 83
TokenURL 83
ClientID 83
ClientSecret 83
DriveID 83
Output table 83
SharePoint:DriveItemContents 84
Plugin Data Row Contract 84
RequestID 84
TokenURL 85
ClientID 85
ClientSecret 85
DriveID 85
ItemID 85
TargetDirectory 85
Configuring Office 365 Access 85
Missing Documentation or Documentation Questions 86

Overview

Syniti Metadata Discovery is an extension to System Types functionality in the Common WebApp of the Stewardship Platform.

The Metadata Discovery application can perform many tasks, such as:

- Generating Application Module Hierarchy Taxonomies
- Generating flat taxonomies based on tables
- Assigning tables to the application modules
- Discovering where the application modules exist
- Visualizing relationships among application modules, tables, and hierarchies
- Displaying where all transactions exist and which module they belong to

Specifically, using Metadata Discovery, users can view the following for a System Type:

- Schemas, tables (objects) and fields (attributes)
- Custom Objects
- Object and Attribute level Relationships from
    - Constraints (Key structure)
    - Object Dependencies
    - Data Dictionary information
- Multi Language where supported
- SAP ABAP pool and cluster tables
- SAP HANA model views
- Object Row Count and Size
- Transaction codes by Application Modules
- Objects within a transaction
- Interfaces in use and direction (Outbound vs Inbound)

Metadata Discovery provides ubiquitous connectivity to:

- On-prem databases and data warehouses.
    - OLE, ODBC, JDBC
- Cloud databases and SaaS-based applications
    - ODBC, OData, HTTPS, REST, SOAP

It can also leverage a customer's iPaaS or API management solution based on the use case and capabilities of that solution.

Metadata Discovery provides conceptual grouping of data objects, allowing users to:

- Navigate through application components and discover how data objects and tables are categorized within the application module hierarchy
- Automatically create logical business-friendly names grouping of objects
- Jump start a Catalog by integrating that business language into one or many Metadata Catalogs and Business Glossaries

The product also provides conceptual Grouping of Application Transactions | API paths, allowing users to:

- Navigate through application components and discover which transactions exist and how they are related to the application or modules within the application
- Relate publicly available APIs and their coverage within the application where relevant.
- Relate these transactions to business terminology and jump start a governance initiative by discovering new concepts that can then be defined as a Business Term and related to the technical metadata

With the creation of Application Module Hierarchies, users can:

- Decipher application-specific metadata and structure to automate business user friendly Application Module Hierarchies.
    - Extend those hierarchies to be cross application and cross functional, supporting the unique way applications interact in your application ecosystem
- Automate the creation of Conceptual Groupings of data elements from discovered Application Module Hierarchies

After the Metadata Discovery extension has been installed, System Types can store additional information used for data lineage:

- At the System Type level, the Instance and Database fields have been added
- At the table level, users can assign different schemas to different tables to facilitate sending data to third party systems
- At the table level, the size, rows and comment fields have been added
- For relationships, an indicator for transformations has been added, as well as a text fields to describe the transformation

.

Metadata Scanner Capability Matrix

The following table lists systems for which System Type Models are available and the types of objects that can be discovered as a result of the scan. Some of these generic models are reusable. When purchased, these connectors are added as System Type Models to the Stewardship Platform. These Models are the basis for System Types, and System Types are automatically added to the Metadata Discovery extension.

NOTE: The Stewardship Tier is delivered with a set of scanner technologies, but custom technologies can also be created as part of a services engagement.

Application & Product Name	Objects & Attributes	Relationships - Constraints	Additional Relationships Dependencies	Transformations	Reference Data	Custom Objects	Module Hierarchy	Interfaces
SAP ECC-SAP ERP	√	√	√		√	√	√	√
SAP CRM-SAP Customer Relationship Management	√	√	√		√	√	√	√
SAP SCM-SAP Supply Chain Management	√	√	√		√	√	√	√
SAP SRM-SAP Supplier Relationship Management	√	√	√		√	√	√	√

Application & Product Name	Objects & Attributes	Relationships - Constraints	Additional Relationships Dependencies	Transformations	Reference Data	Custom Objects	Module Hierarchy	Interfaces
SAP HCM-SAP Human Capital Management	√	√	√		√	√	√	√
SAP PLM-SAP Product Lifecycle Management	√	√	√		√	√	√	√
SAP APO-SAP Advanced Planning and Optimization	√	√	√		√	√	√	√
SAP EWM-SAP Extended Warehouse Management	√	√	√		√	√	√	√
SAP S/4HANA-SAP S/4HANA	√	√	√		√	√	√	√
SAP C/4HANA-SAP C/4HANA	√	√	√				√	
SAP HANA-SAP HANA	√	√	√		NA	NA	NA	

Application & Product Name	Objects & Attributes	Relationships - Constraints	Additional Relationships Dependencies	Transformations	Reference Data	Custom Objects	Module Hierarchy	Interfaces
SAP Hybris Marketing- SAP Hybris Marketing Cloud	√	√	√					
SAP Hybris Commerce-SAP Hybris Commerce Cloud	√	√	√					
SAP SuccessFactors-SAP SuccessFactors	√	√	√		√			
SAP Ariba-SAP Ariba	√	√	√				√	
Workday-Workday	√	√	√		√		√	
Salesforce-Salesforce	√	√	√		√	√		
Snowflake-Snowflake	√	√						

Application & Product Name	Objects & Attributes	Relationships - Constraints	Additional Relationships Dependencies	Transformations	Reference Data	Custom Objects	Module Hierarchy	Interfaces
Cloud Data Platform								

Configure Metadata Discovery

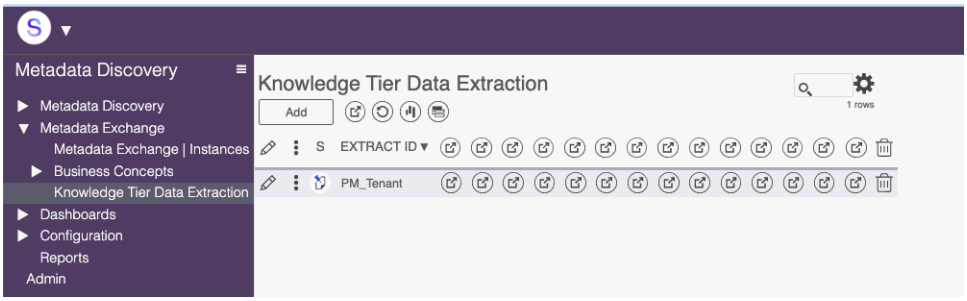
Consult the *Syniti Metadata Discovery Installation Guide* for instructions for installing Metadata Discovery. The document includes scripts to create sample data and post-installation configuration steps.

Configure Knowledge Tier Download

This must be done to use the Metadata Discovery features and this application was built to integrate with the Syniti Knowledge Tier and Syniti Catalog.

A Syniti API user ID must be requested via Support.Syniti.com in order to configure it properly.

A single record with a unique Extract ID (string value) can be entered without the UserID and Password information. This record is necessary to use the Metadata Discovery section of the application. Without the user name and password the below features will not work as expected.



Each Button or Toolbar event on the page above supports a particular set of functionalities.

- **Extract All –** will call all of "all assets" GET endpoints in a particular synchronous order. Each of the Extracts can be run on its own individually ad hoc. There is also a service page that runs every 4 hours that runs these extract jobs.
    - Extract Terms
    - Extract Rules
    - Extract Policies
    - Extract Vision
    - Extract Mission
    - Extract Goals
    - Extract Initiatives
    - Extract Categories
    - Extract Users
    - Extract Programs
    - Extract Systems
    - Extract Datasets

    - Extract System Components
    - Extract Enforcements
- **Reset** - This will reset the status of the ExtractID record so that the events on the page can be run again, if it has failed previously.
- **Datasets –** Once the Extract Datasets event has been successfully completed this toolbar button will allow a user to extract the field information that is associated with every dataset or just a single dataset. This toolbar button navigates to a different dataset summary page.
    - Knowledge Tier Dataset page has the following Toolbar events:
        - **Extract Datasets** – this allows a user to extract all datasets manually if there is one or more, they believe exists in the SKP tenant but isn't on this page.
        - **Extract Dataset Fields –** this allows a user to select a dataset record on the page and extract the field associated with it.
- **Systems** - Once the Extract Systems event has been successfully completed. This toolbar button will allow a user to extract the component information that is associated with every System or just a single System. This toolbar button navigates to a different Systems summary page.
    - Knowledge Tier Extract System page has the following Toolbar events:
        - **Extract Systems**– this allows a user to extract all systems manually if there is one or more, they believe exists in the SKP tenant but isn't on this page. This will not pull back System Components or Fields.
        - **Extract Components –** this allows a user to select a system record on the page and extract the components associated with it.

Select Internal Job Queue for Background Jobs

- Navigate to the Configuration Menu
    - Parameters Page
        - Internal Job Queue
            - Select Value for Service Queue
                - The Product team recommends a Background Events service Queue.

Configure Object Dependencies

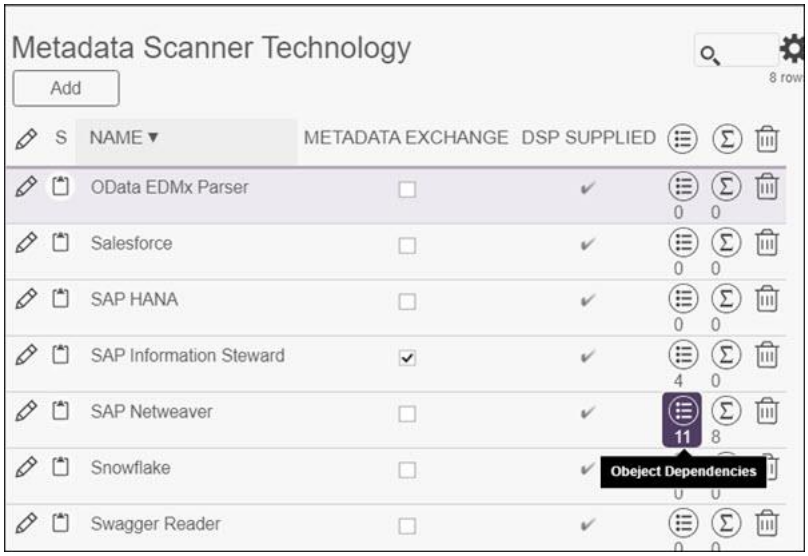
Dependencies must be set up for the Metadata Discovery extension so that it can perform automated tasks, such as creating Application Module Hierarchies or analyzing modules for "in use" studies.

The *Metadata Scanner Technology | Object Dependencies* page provides the details, including the tables that need to be downloaded in Collect or some other system, and why they need to be

downloaded to execute that module. Check boxes indicate whether the object is needed for creating application modules or for creating other objects, like System Types.

To set up object dependencies for the scanner technology in Metadata Discovery:

1. Select **Configuration > Metadata Scanner Technology** in the *Navigation* pane.
2. Click the **Object Dependencies** icon.



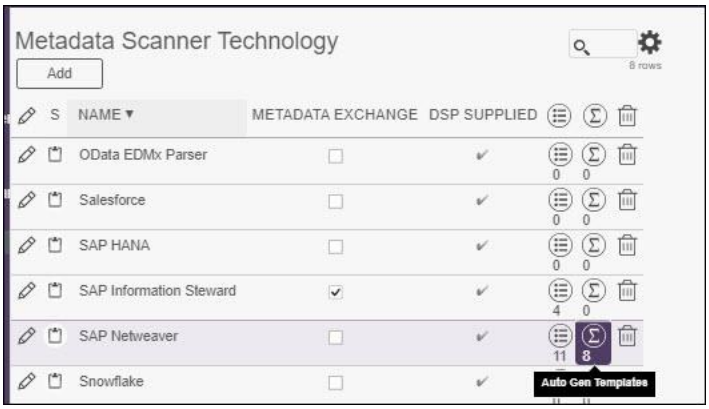
1. Click Add.
2. Enter the name of the object in the OBJECT NAME field.

NOTE: This could be the table name for a table that must be downloaded in Collect or some other source.

3. Enter a description of the object in the OBJECT DESCRIPTION field.
4. Enter a description of the type of data the object stores that explains why it must be included in the Application Module Hierarchy scan in the PURPOSE field.
5. Check the "FOR APPLICATION MODULES" check box if the object is required to create the Application Module.
6. Check the "FOR OBJECTS" check box if the object is required to build other objects, like System Types.
7. Check the "FOR INTERFACES" if the object is required to scan interface metadata.
8. Check the "FOR METADATA EXCHANGE" if the scanner is against another metadata solution where we are sending or receiving metadata scanner results to and/or from.
9. Click Save.

Configure Templates for Auto Generation

On the *Metadata Scanner Technology* page **(Configuration > Metadata Scanner)** click the **Auto Gen Template** icon to access the *Metadata Scanner | AutoGen Templates* page.



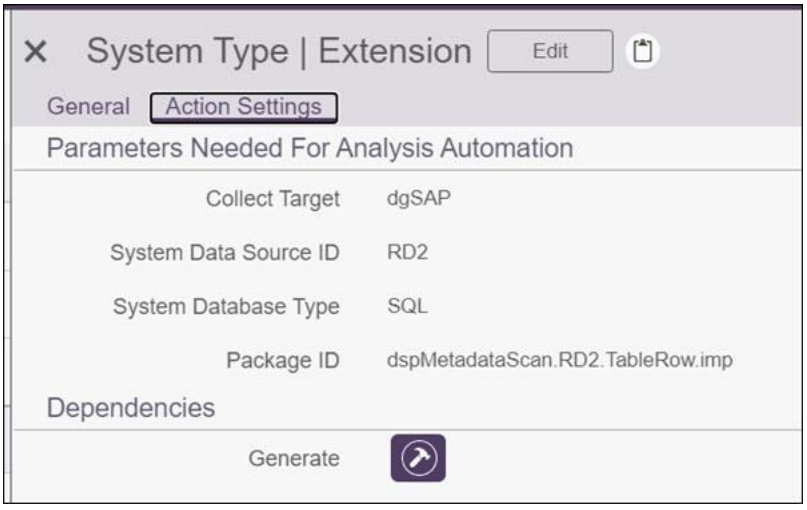
This page lists the SQL Templates used by the scanning process. Views are created in the local Stewardship Tier Metadata Discovery database that are leveraged by stored procedures to execute the logic that creates the Application Module Hierarchy and the Application Hierarchy Visualizations. Users can add additional levels to the template as needed.

These templates are also used to capture the table assignment to application modules and which T codes are assigned to which application modules.

After the dependencies have been configured and the templates created, the dependencies must be generated for the System Type.

Generate Dependencies

To generate the dependencies, access the *Vertical* View of the *System Type | Extension* page (**Configuration > System Type | Extension)**. On the Action Settings tab, under Dependencies, click **Generate**. NOTE: not all scanners have dependencies or need to generate. This option will only appear if needed.



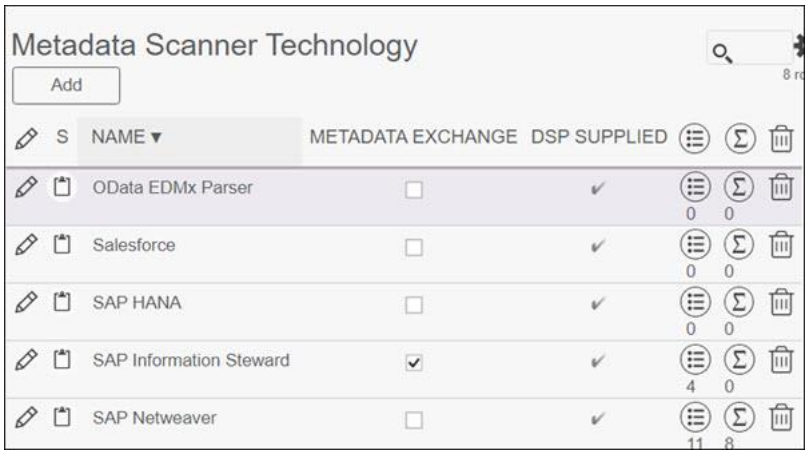
Configure Metadata Exchange

A scanning technology called a metadata exchange is used to send to or receive metadata from a third-party metadata repository or data catalog. Results from Syniti Metadata Scanners can be sent

to a third-party metadata repository or catalog. In addition to sending, Metadata exchange can also receive results of metadata scanners or integrators. This can be used to catalog this information in the Knowledge Tier, or to drive automation in Syniti ADM, DQ, or MDM.

These types of scanner technologies are identified on the *Metadata Scanner Technology* page (**Configuration > Metadata Scanner Technology**). New exchange technologies can be created to fit your requirements as this feature is extensible. Please check with Syniti Customer Success to see what other metadata exchange technologies are on the roadmap.

In this version example, **Syniti Stewardship Tier**, **SAP Information Steward** and **ASG Data Intelligence** uses a metadata exchange scanner technology. Each Scanner technology has a unique details page that is dynamic based on the scanner technology as different features and functionality are supported for each solution.



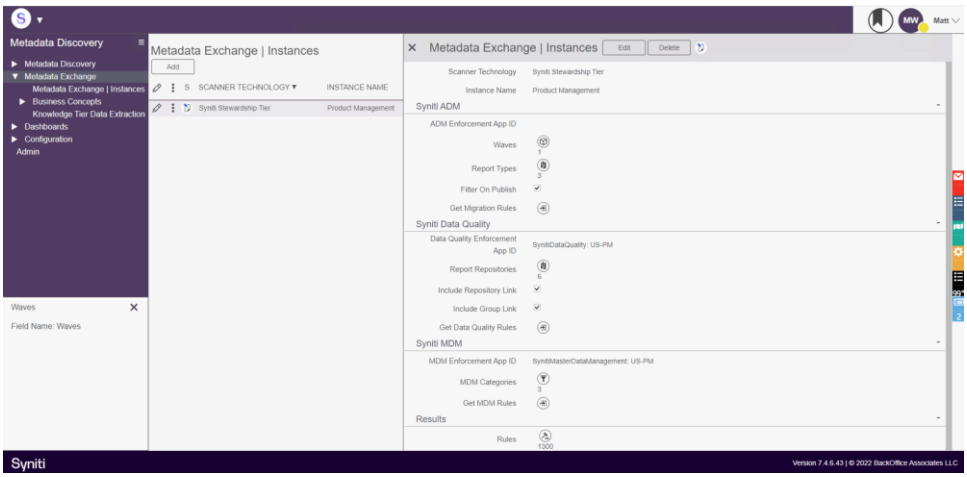
Metadata Exchange for SAP Information Steward

For SAP Information Steward the following is supported:

On the *Metadata Exchange | Instance's* page (**Configuration > Metadata Exchange | Instances)**, additional details display. This is where auser can define physical instances of the third-party metadata exchange and the necessary details to enable the exchange functionality. The *Horizontal* View of this page is the same regardless of the scanner technology used, but the fields and actions available on the *Vertical* View vary depending on the scanner technology.

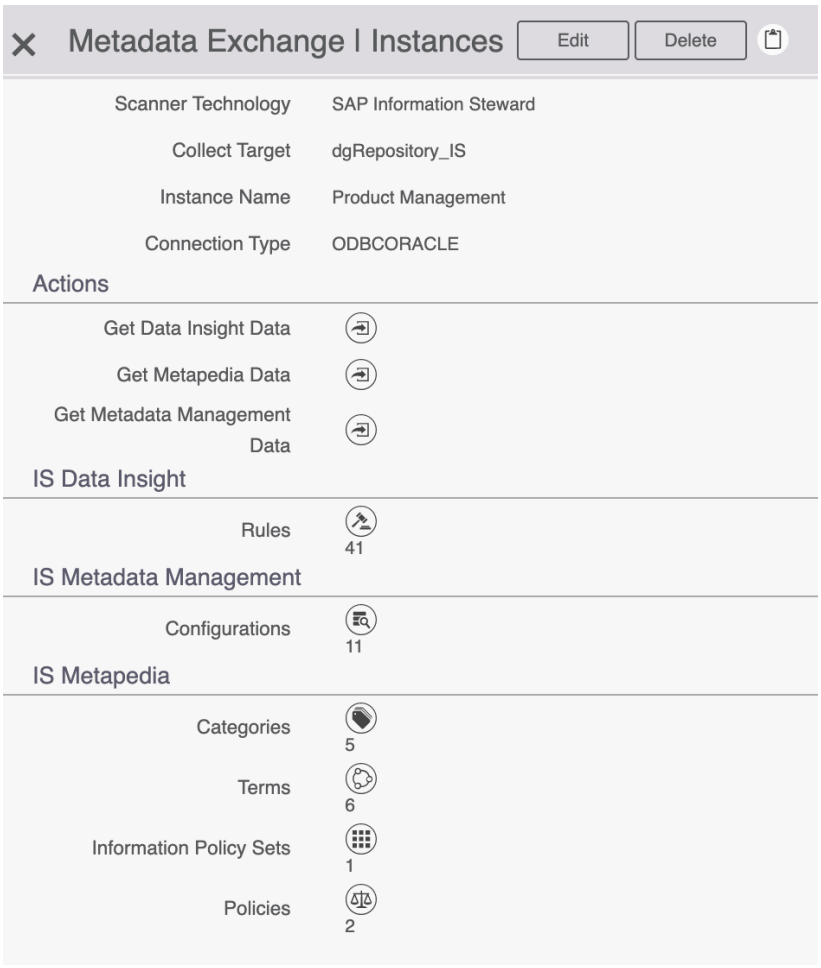
Adding the SAP Information Steward Instance:

- Select the "SAP Information Steward" Scanner Technology
- Name the Instance (I.e., Business Unit for Widgets)
- Select the collect target that was used to download the dependent data
    - See Scanner Technology Dependencies for a list of required tables for SAP Information Steward



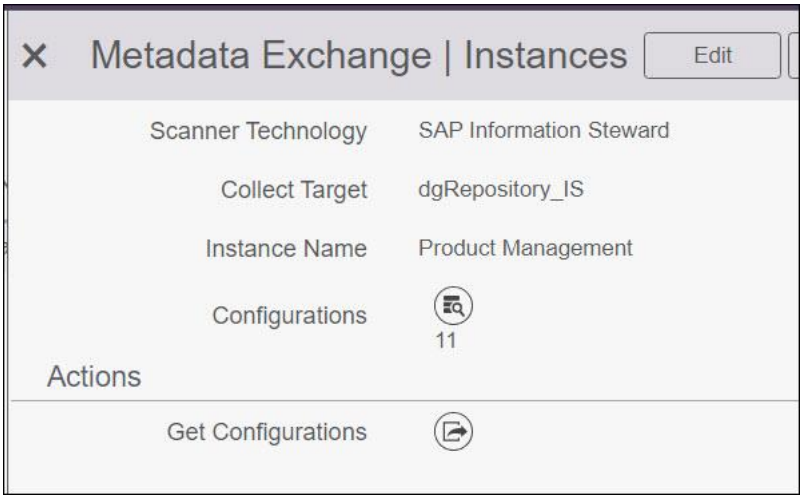
By default, all the metadata available is brought in to the Stewardship Tier via Collect. It is unlikely that all of it will be useful or will fit into the format of a System Type Model and System Type, therefore, it is possible to exclude specific types of metadata once imported, so that they are not cataloged within the Knowledge Tier.

Review the *Vertical* View for the instance to remove unnecessary data from the scan. This cuts down on processing time and allows a user to review information more easily.

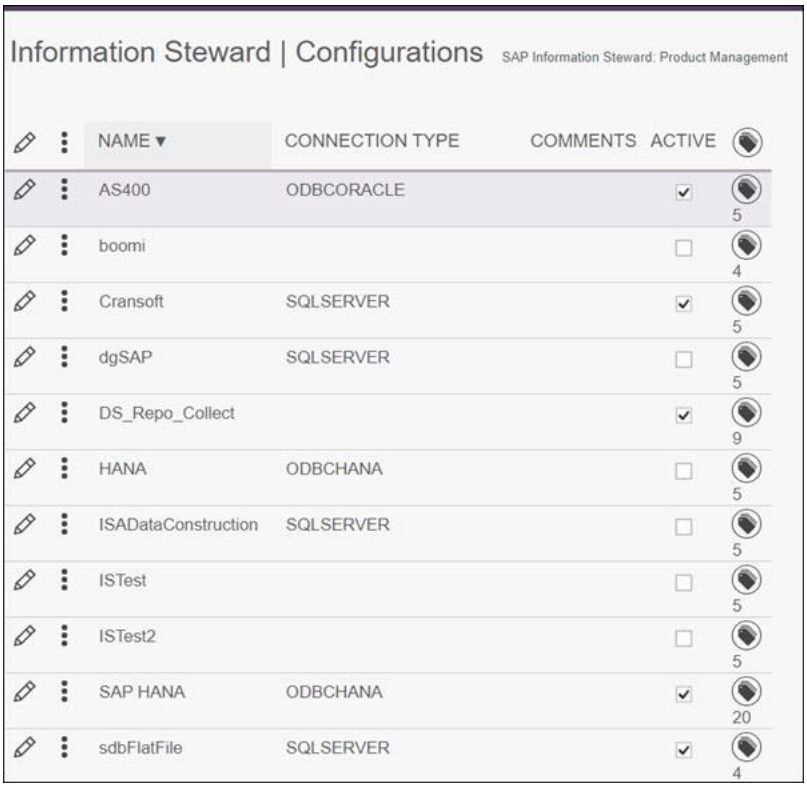


- Actions
    - Get Metadata Management Data
        - Receiving Metadata Integrator results from IS: Metadata Management via Configurations page
            - Inclusion/ Exclusion of discovered Type codes to send to Syniti Stewardship Tier Metadata Repositories. (I.e. I want to include database objects and not technical packages, keys, defaults, system views, etc...)

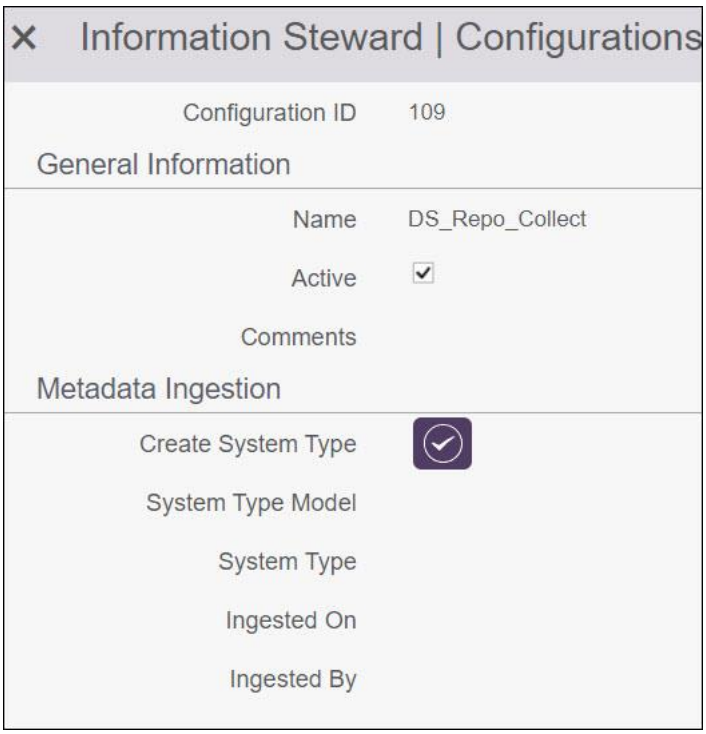
In this example, to indicate which data sources to exchange metadata with, click the **Configurations** icon.



On the Configuration page, select the **ACTIVE** check box for the data source to exchange metadata with.

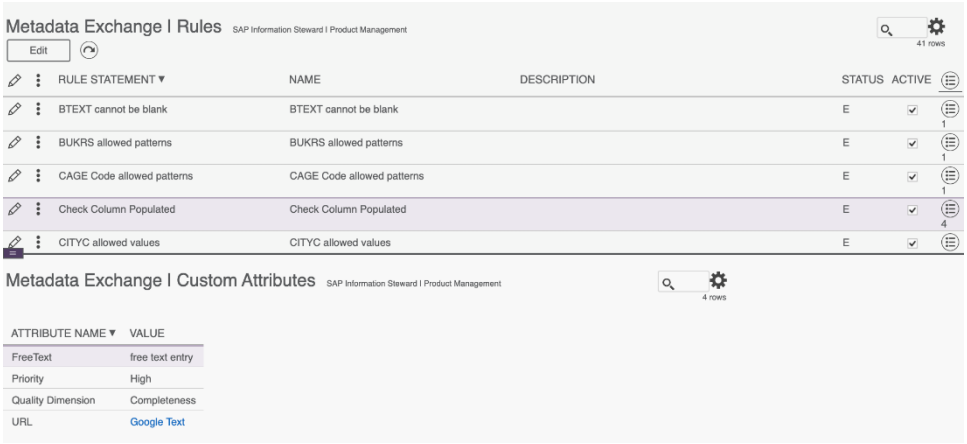


To create a System Type from a data source, click the **Vertical View** icon for a data source, then click the **Create System Type** icon. This will create a unique system type model and system type for each configuration. The system type can then be automatically populated or imported from the System Type Model generated.

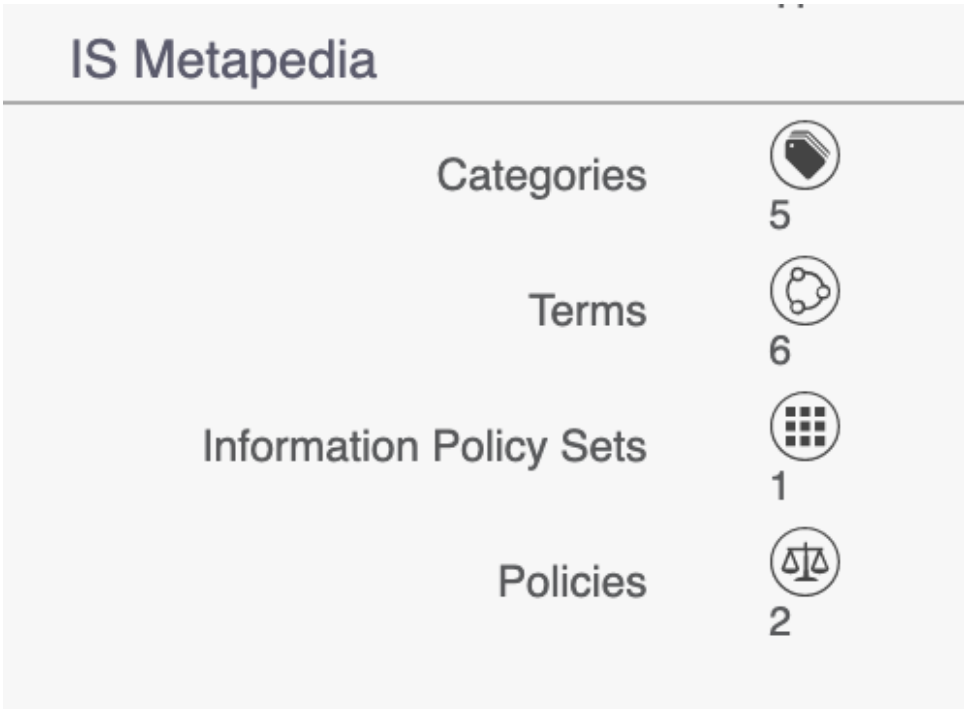


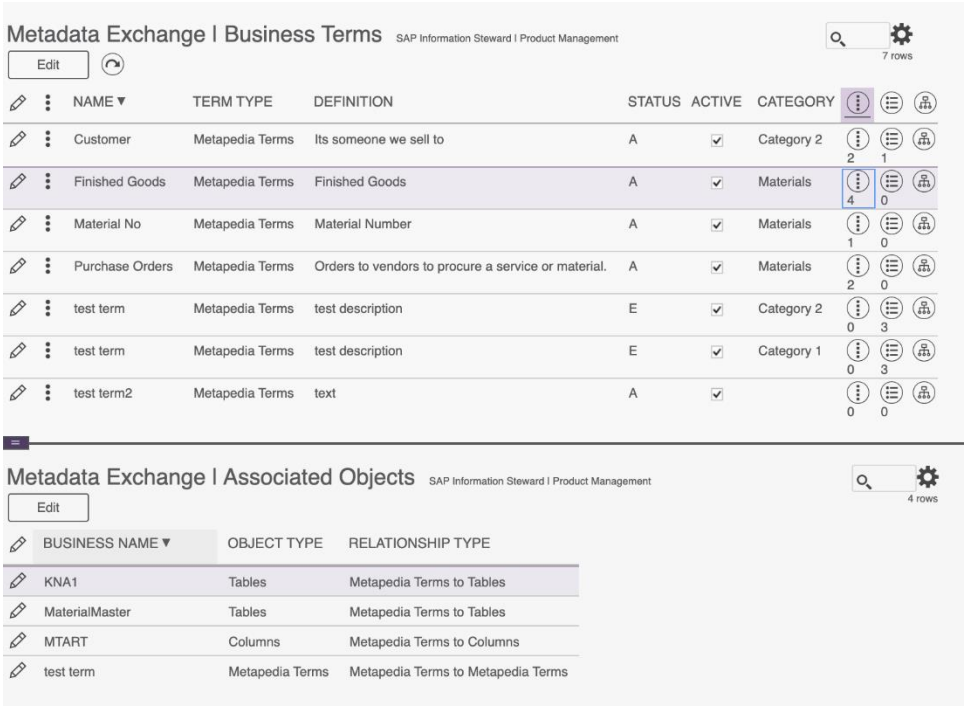
Once a System Type is created, it is automatically included in the Metadata Discovery application and can be discovered in the System Type Extension page

- Actions
  - Get Data Insight Data
    - This will retrieve Rules in a Rule Statement like format along with custom attributes and their values
    - This can be used to stage, transform or enrich then send Rule Asset to the Syniti Knowledge Tier. For more information on how to send discovered rules to the Knowledge Tier, refer to Sending Rules to the Knowledge Tier.
  - Mapping from Information Steward Rules to Knowledge Tier Rule Assets
    - SourceID = MMT_Rule.rule_id
    - RuleStatement = MMT_Rule.business_name
    - Implication = MMT_Rule.description



- Actions
    - Get Metapedia Data
        - Retrieves Categories, Terms, Policy Sets, and Policies; including relationships between these objects and metadata within Metadata Management
        - Custom Attributes and their values.
        - Visual Hierarchy of Categories





Metadata Exchange for Collibra

For Collibra Data Intelligence, the following is supported via Collibra's 2.0 REST API layer:

- Extracting Metadata from Collibra Data Intelligence and storing in Syniti Metadata Discovery
- Configuring Attributes and Relations by Asset Type
- Sending Metadata from Syniti Metadata Discovery to Collibra Data Intelligence

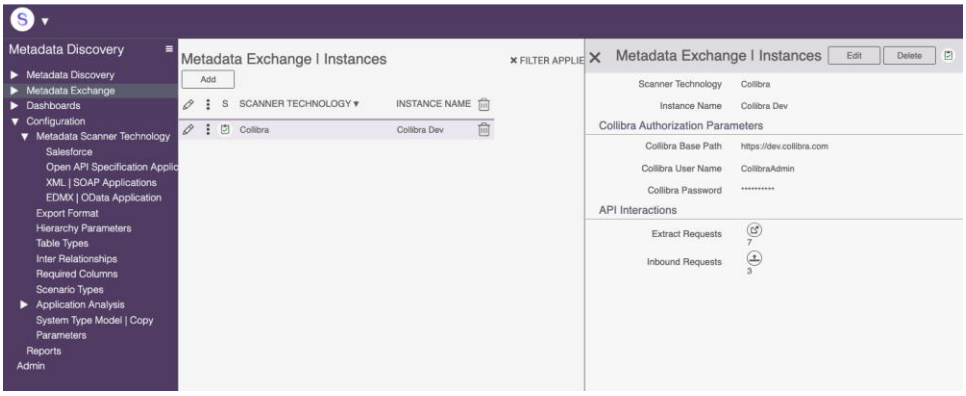
To configure Collibra Authentication for Metadata Exchange prepare the following connection information:

1. Create a Metadata Exchange Instance by adding a record and selecting Collibra as the scanner technology.
2. Enter the name of the instance
    a. Press SAVE
3. Navigate to the Vertical View of the Collibra Instance record
    a. Enter the following:
        i. Collibra Base Path

1. This is used for building the REST API endpoints and is often the base path of the Collibra instance you are accessing in the browser.

ii. Collibra User Name

iii. Collibra Password



Once Authentication is configured, API interactions can be used

1. Extract Requests will list all the GET HTTP Verbs that are supported by Metadata Exchange.
    a. The Collibra Integration | Extraction page will list all GET Endpoints supported and provide the following functionality
    b. EXTRACT – button to call the endpoint manually
    c. VIEW DATA – a button to view the data that was returned by that endpoint.
2. Inbound Requests will provide a User Interface for creating inbound requests to send metadata to Collibra.
    a. Supported HTTP VERBs are POST, PATCH, and DELETE for BULK Assets, Attributes, and Relations.
    b. Requests will have a scenario type that will determine which endpoints will be called once all the data is collected and validated.
        i. CREATE will call
            1. POST /assets/bulk
            2. POST /attributes/bulk
            3. POST /relations/bulk
        ii. CHANGE will call
            1. PATCH /assets/bulk
            2. PATCH /attributes/bulk
            3. PATCH /relations/bulk

iii. DELETE will call

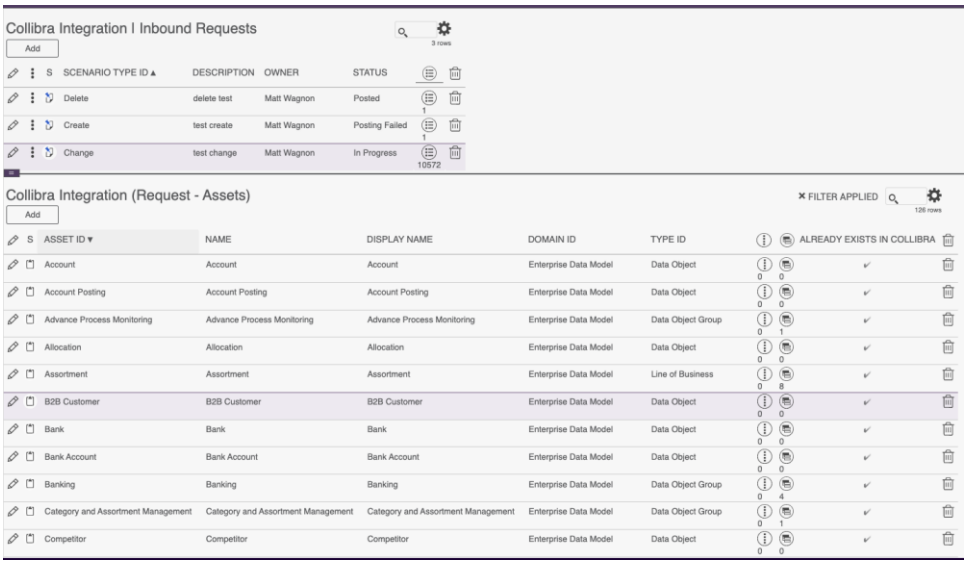
1. DELETE /assets/bulk

Collibra Inbound Requests

To send data to Collibra an inbound request will need to be created to group together assets, their attributes and values for those attributes, and relations to other assets in BULK.

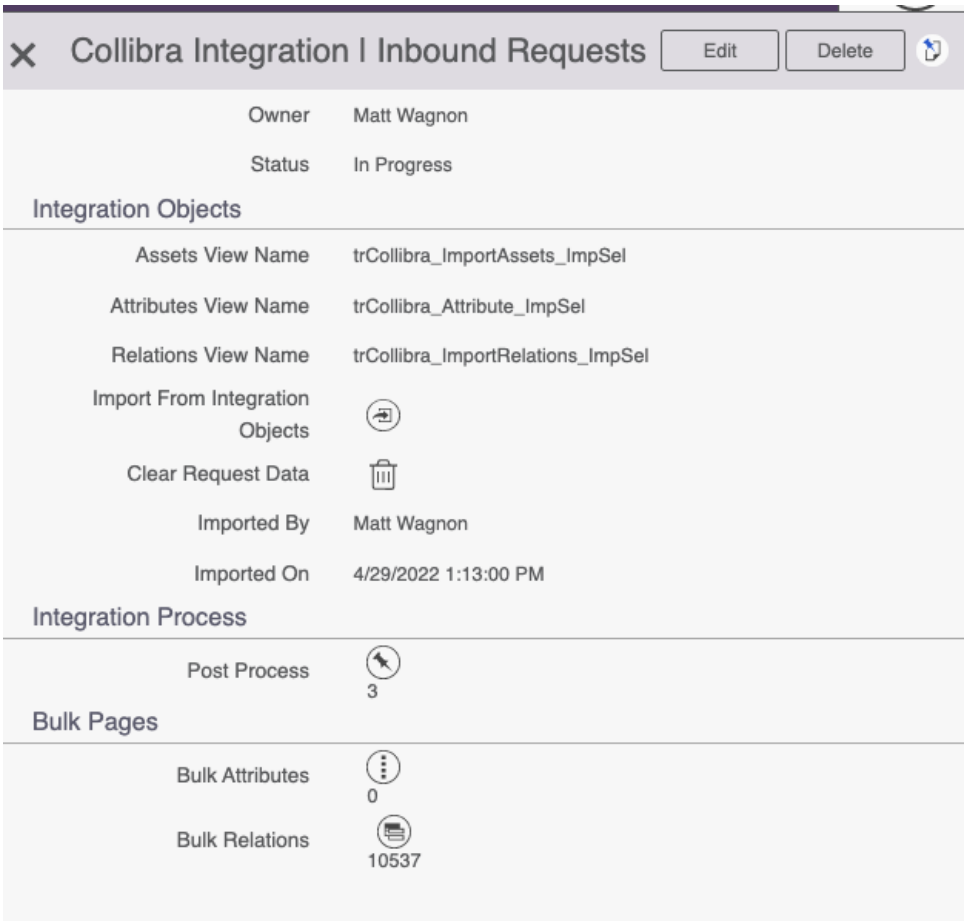
Requests can be created but can only perform one scenario, that is CREATE, CHANGE, or DELETE. This Scenario Type must be selected as part of the request.

A description can be added optionally and a Metadata Discovery User must be selected as the owner of this request. This is used to signify who is responsible for curating the data within and eventually posting the validated information into Collibra.



Importing Collibra data from a SQL server view

Each Request can have data sent to it via SQL server view registration. Once the request has been created, navigate to the vertical view of the request.



Under the Integration Objects label, a user can seed the data within the request from the following:

- Assets View Name
    - Required columns in the view

Column Name	Description and Use
[id]	Collibra ID; guid
[name]	Collibra Name; string
[displayName]	Collibra Display Name (this is not a description of the Asset, it's the name in the UI); string
[domainid]	Collibra ID of the domain the assets exist in or are intended to exist in; guid
[typeid]	Collibra Asset Type id; guid

[AlreadyExistsinCollibra]	Used to mark if the data already exists in the Collibra system or not. This is needed to drive request and posting functionality; bit

- Attributes View Name
  - Required columns in the view

Column Name	Description and Use
[assetid]	Collibra ID; guid
[value]	Collibra Name; string
[typeid]	Collibra AttributeType id; guid
[AlreadyExistsinCollibra]	Used to mark if the data already exists in the Collibra system or not. This is needed to drive request and posting functionality; bit

- Relations View Name
  - Required columns in the view

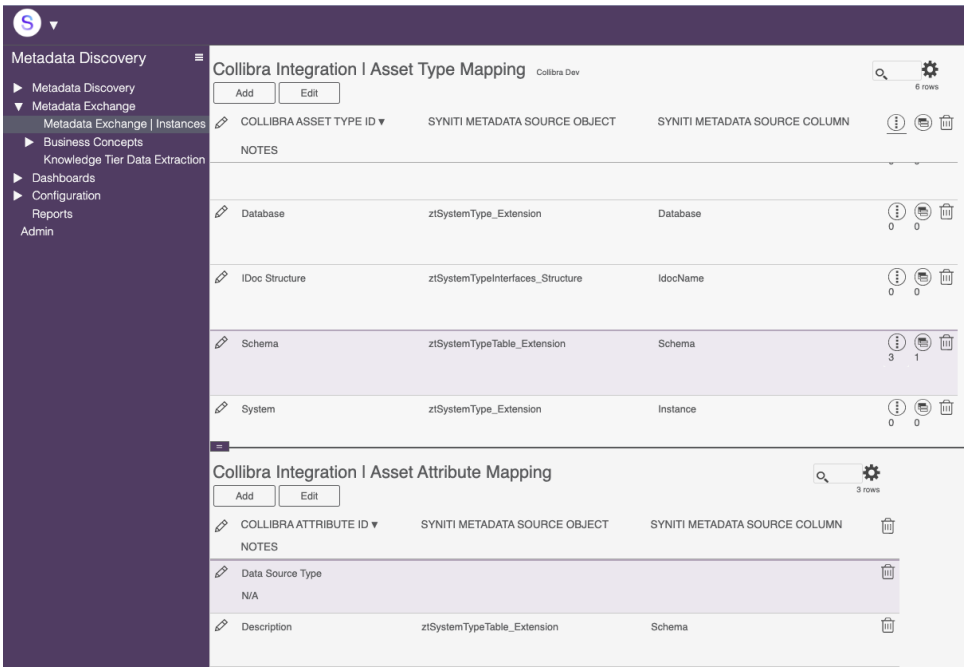
Column Name	Description and Use
[sourceid]	Source Asset Collibra ID; guid
[targetid]	Target Asset Collibra ID; guid
[typeid]	Collibra Relation Type id; guid
[AlreadyExistsinCollibra]	Used to mark if the data already exists in the Collibra system or not. This is needed to drive request and posting functionality; bit

Once the views are selected and meet the schema structure documented above, you can Import the data that returns in the views into the Request table structure.

Note that if the views do not have the required columns a validation error message will appear. Click on this message to see which fields you are missing.

If the data is not imported in the manner you want, you have the option to Clear the request Data

On the Collibra Inbound Mapping area or the vertical view of the Collibra Metadata Exchange Instance, a user can add asset types and document how they are to be mapped from the Syniti Metadata Repository to Collibra.



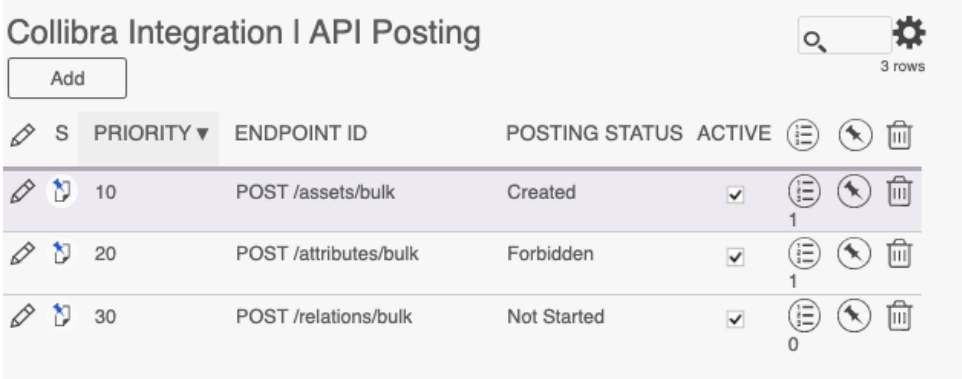
NOTE: This mapping does not automate ingestion into Collibra but rather can be used to do the following:

- Document Mapping Specification from Syniti to Collibra
- Capture static configuration when creating asset types
    - Relationships between asset types (I.e., Database to Schema, Schema to Table, Table to Column)
    - Standard Attributes Types and their desired assignment to Asset Types along with their values, if available, to be seen on the Asset Once Posted to Collibra.

Posting data to Collibra | Integration Process

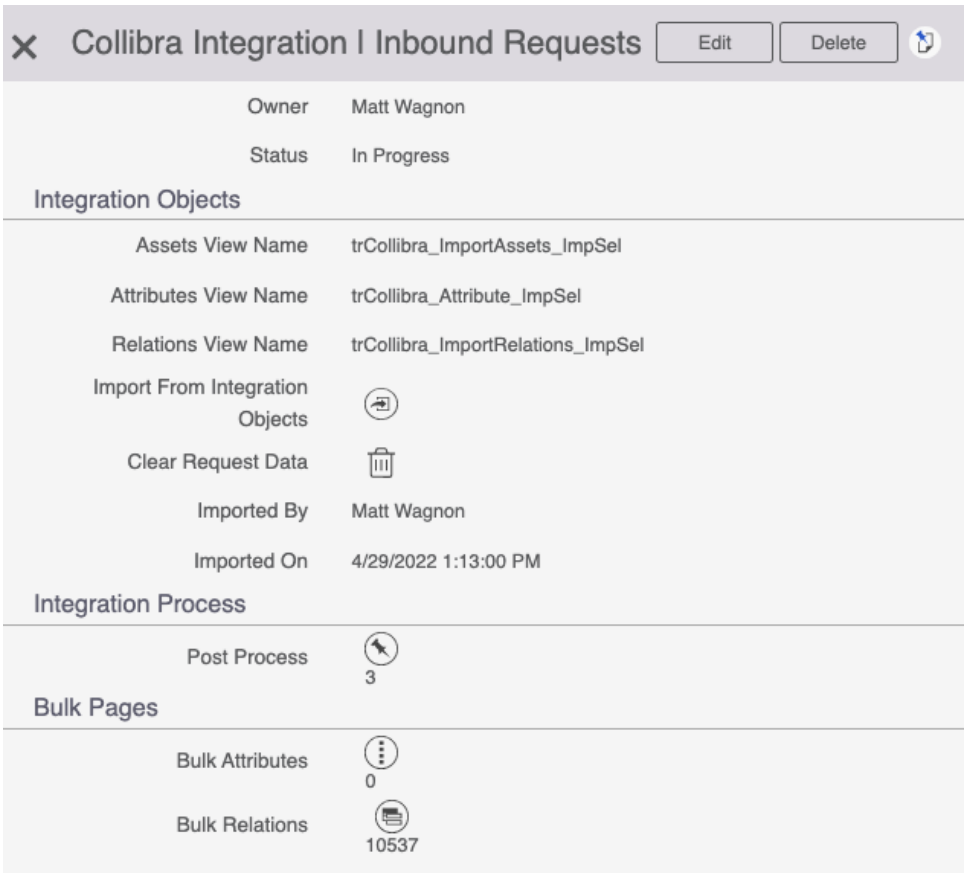
The integration process with Collibra is driven at the Request level. On the vertical of the request, Post Processes are seeded to the request in order based on the scenario type of the request.

In the case of Collibra, posting includes calling a specific REST API endpoint using data that is stored in the Request staging tables. Each endpoint needs to be called in priority order and returns a response and status so that the end-user knows the efficacy of the process.



Bulk Pages

In the main request UI, you can only add data attributes and relations per asset. Under BULK pages a user can upload via excel integration or via the UI, many attributes and/ or relations for many assets.

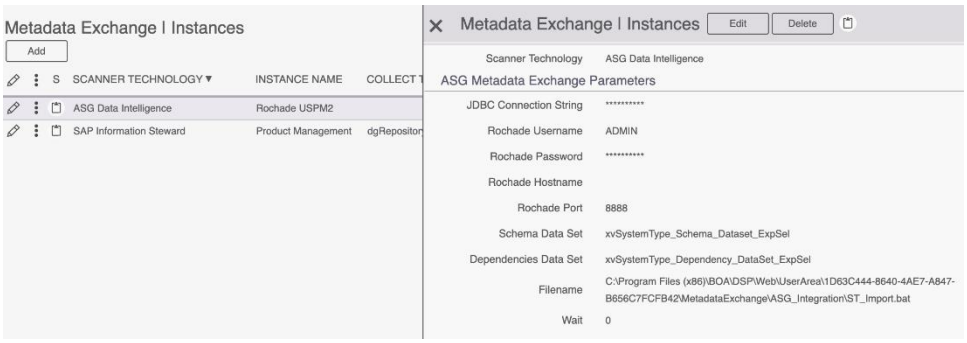


These pages will also provide the end-user with a count of the total amount of relations and attributes that are being prepared to be sent to the Collibra REST API layer.

Metadata Exchange for ASG Data Intelligence

For ASG Data Intelligence, the following is supported:

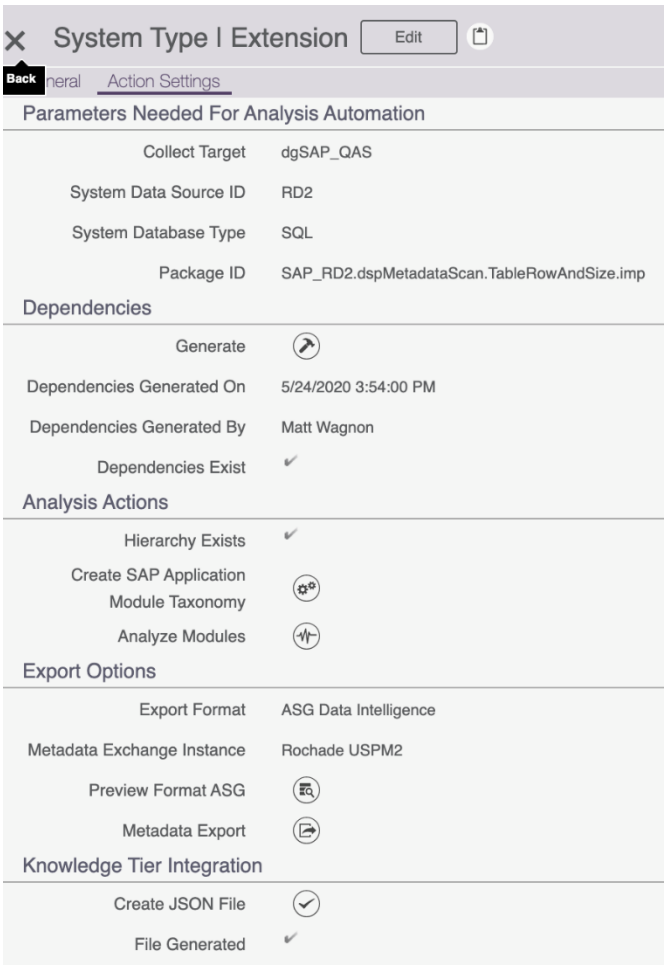
- Sending Metadata from Syniti Metadata Discovery to ASG Data Intelligence
    - A connection per ASG Data Intelligence instance much be created



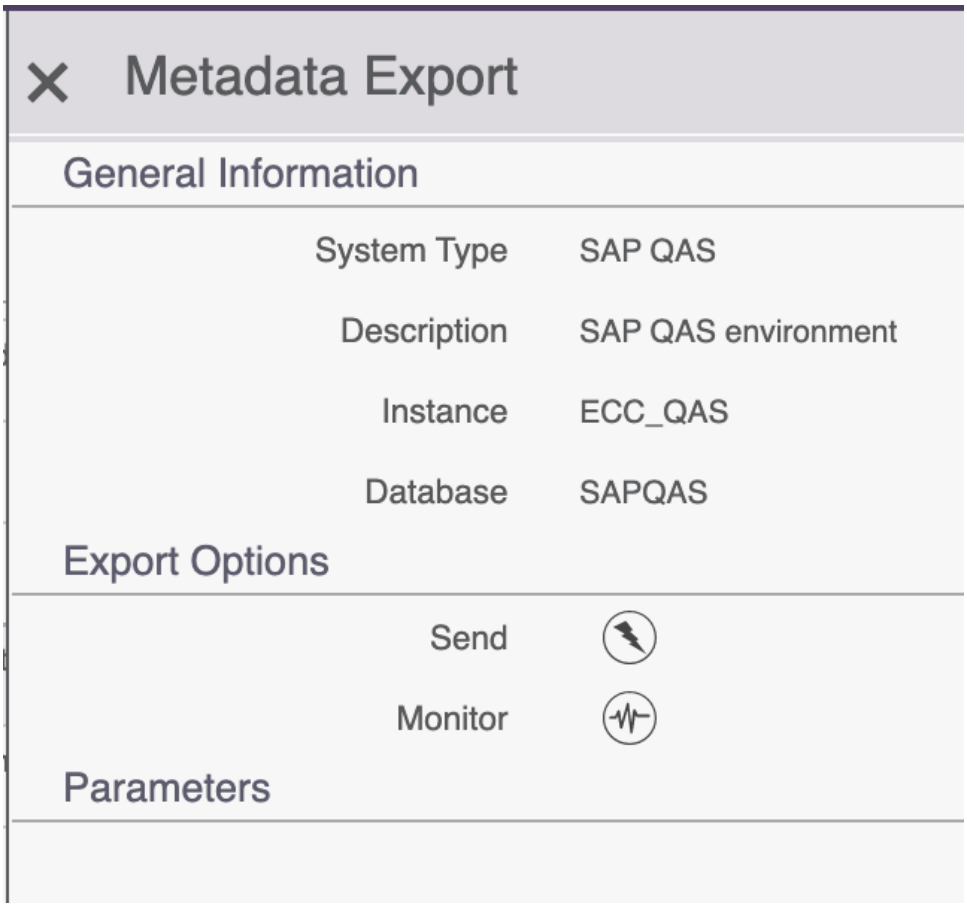
Once the connection is created, Metadata can be sent to ASG Data Intelligence in batch or scheduled via a service.

On the vertical of a System Type on the System Type | Extension page, on the Action Settings tab, within the Export Options label perform the following to send metadata to ASG:

- Export Format = ASG Data Intelligence
- Metadata Exchange Instance = “Name of the instance” created on the Metadata Exchange Instances page



- Click Metadata Export
    - Ensure Instance, Database and Schema fields are populated for all objects in the system type. Metadata Exchange will fail if not
- Click Send (Monitor will allow the user to monitor the status and tasks of the integration and see error or success messages)



The System, datatypes, and all relevant metadata will now be available in ASG Data Intelligence.

Metadata Exchange for the Syniti Stewardship Tier

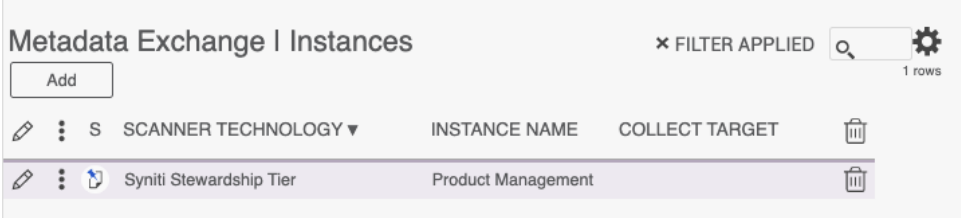
For the Syniti Stewardship Tier, the following is supported:

NOTE: the stewardship tier instance that can is to be scanned must be the instance that this application (Metadata Discovery) is installed on.

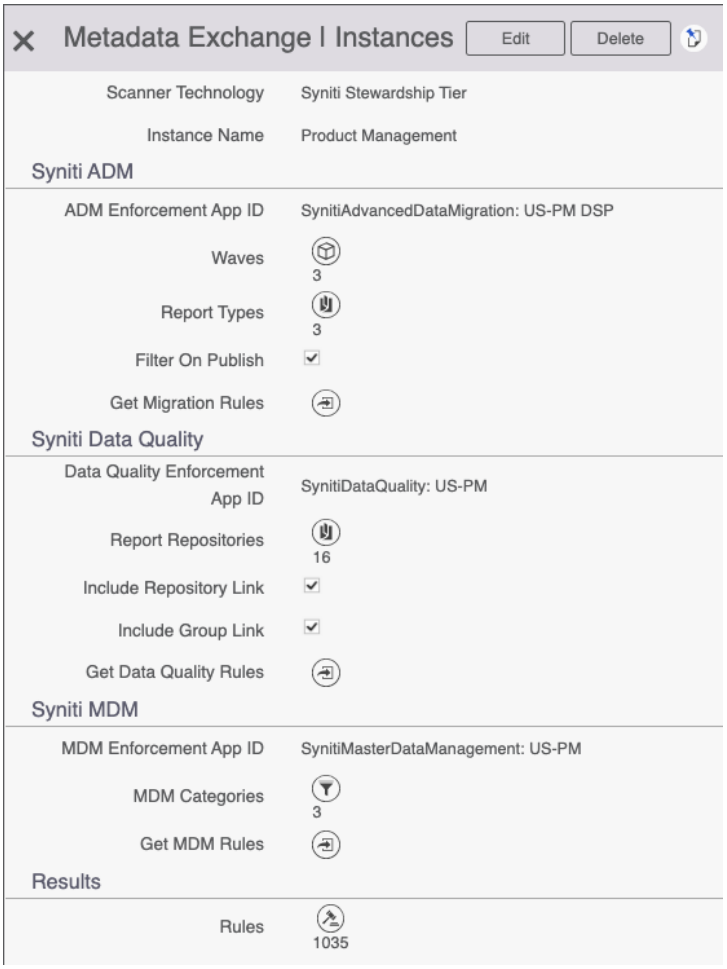
For Metadata Exchange, the "Syniti Stewardship Tier" Metadata Scanner Technology must be installed (acquire via support).

Once installed, The Syniti Stewardship Tier scanner technology will appear in the scanner technology dropdown when adding a new reocrd on the Metadata Exchange | Instances page on the Configuration menu of the application. Only the Instance name is required once selecting the

scanner technology. There is not functionality tied to the Collect Target field for this Metadata Exchange Instance.

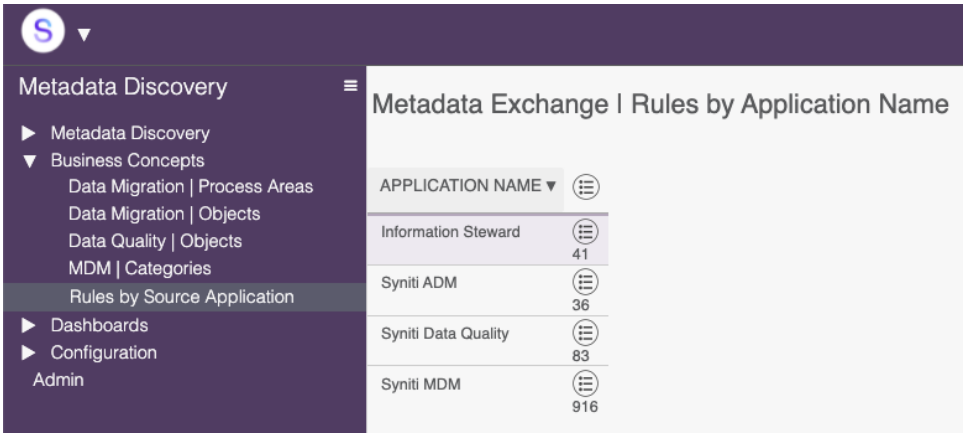


The Syniti Stewardship Tier Metadata Exchange can be used to mine Rules and metadata associated from Syntii Advanced Data Migration, Syniti Data Quality and Syniti MDM and send them to the Syniti Knowledge Tier with enforcement profile links programmatically.



Results of the GET Rules events can be seen at the end of the vertical view shown in the screenshot above.

Results can also be reviewed by Application in the Business Concepts menu ; Rules by Source Application page



There are some slight functional differences by Stewardship Tier application as you can see in the screenshot of the vertical view.

NOTE: the following post steps should be completed before sending rules to the knowledge tier from Syniti Metadata Discovery.

1. Create Rule with an enforcement profile and enforcement app location in the UI of the Knowledge Tier.
    a. Create at least 1 Application for each Method you intend to create rules for (I.e. Data Migration, Data Quality, MDM).
    b. Once created the enforcement application will need to be extracted to support the population of the enforcement app dropdowns in the vertical view. This is done in step 2. If already done, then skip the next step.
2. Configure "Knowledge Tier Data Extraction"
    a. Navigate to Configuration --> Knowledge Tier Data Extraction
        i. Add new Row. Give the row an Extact ID (often this is just the name of the tenant or name of your company).
        ii. Enter the API user credentials for the Knowledge Tier. If you do not have credentials and base path please request them a support.syniti.com
        iii. Need help configuring or questions on API coverage refer to the Knowledge Tier API documentation.
        iv. Run the GET operations for all endpoints.

v. The Enforcement app dropdowns on the vertical view of the metadata exchange for the stewardship tier should now be populated.

1. If not, STEP 1 above was not done or the API call failed.

3. For discovering Rules out of Syniti ADM

a. The project has the ability to filter rules and send them ad hoc, one by one or in bulk. The filtering capabilities consist of a combination of the following:

i. Include one or more waves

ii. Includle one or more report types (I.e. Business Readiness, Error, Target Readiness only)

iii. Filter on Published rules only (target reports that have the published flag on).

b. Once your desired filtering options are set, click "Get Migration Rules"

i. The results number should increase in the amount of target reports that were in scope based on that filtering.

c. Documented Mapping from ADM to Knowledge Tier Required Fields

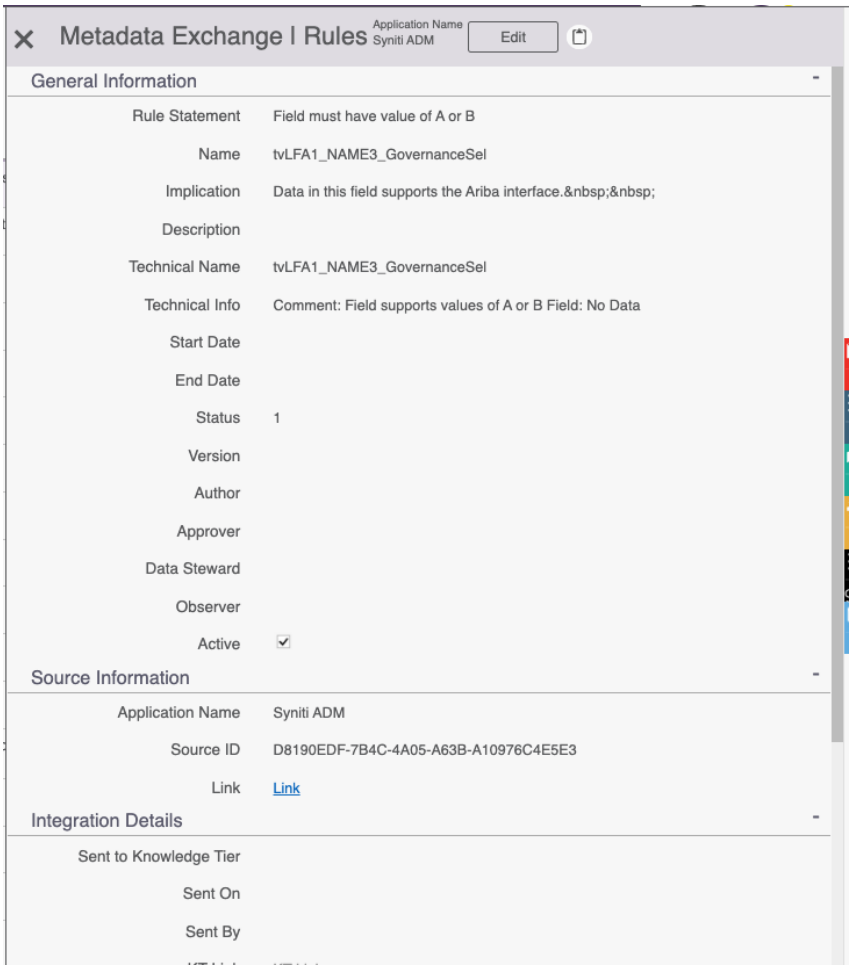
i. SourceID = DSW.dbo.ttWaveProcessAreaObjectTargetReport.WaveProcessAreaObjectTargetReportID

ii. RuleStatement = DSW.dbo.ttWaveProcessAreaObjectTargetReport.Description

iii. Implilcation = DSW.dbo.ttWaveProcessAreaObjectTargetReport.Implication

NOTE: The desired values for the knowledge tier should be filled out at the source application level to support API calls and usability of the data once in the Knowledge Tier. A source link exists to help navigate directly to the Target Report in ADM to make the desired changes. If changes are made the "Get Migration Rules" will need to executed to reflect the changes. Any rules already sent to the Knowledge Tier will not be affected by the re execution.

Below is an example of a rule that was discovered by Metadata Discovery. Sending the rules to the Knowledge Tier will be covered later in the help.



4. For discovering Rules out of Syniti Data Quality
    a. The project has the ability to filter rules and send them ad hoc, one by one or in bulk. The filtering capabilities consist of a combination of the following:
        i. Include one or more report repositories
    b. For the enforcement profiles, you have the option of automatically creating one or many links
        i. Check the Include Repository link to include a link to the registration of the DQ rule at the repository level. This link will filter to the registration of the views.
        ii. Check the Include group link to include a generic link to any group that the DQ rule is registered to. If the report is registered to five groups, then five links in the enforcemetn profile will exists, with a description of the group to differentiate the links.
    c. Once your desired filtering options are set, click “Get Data Quality Rules”
        i. The results number should increase in the amount of target reports that were in scope based on that filtering.

d. Documented Mapping from SDQ to Knowledge Tier Required Fields

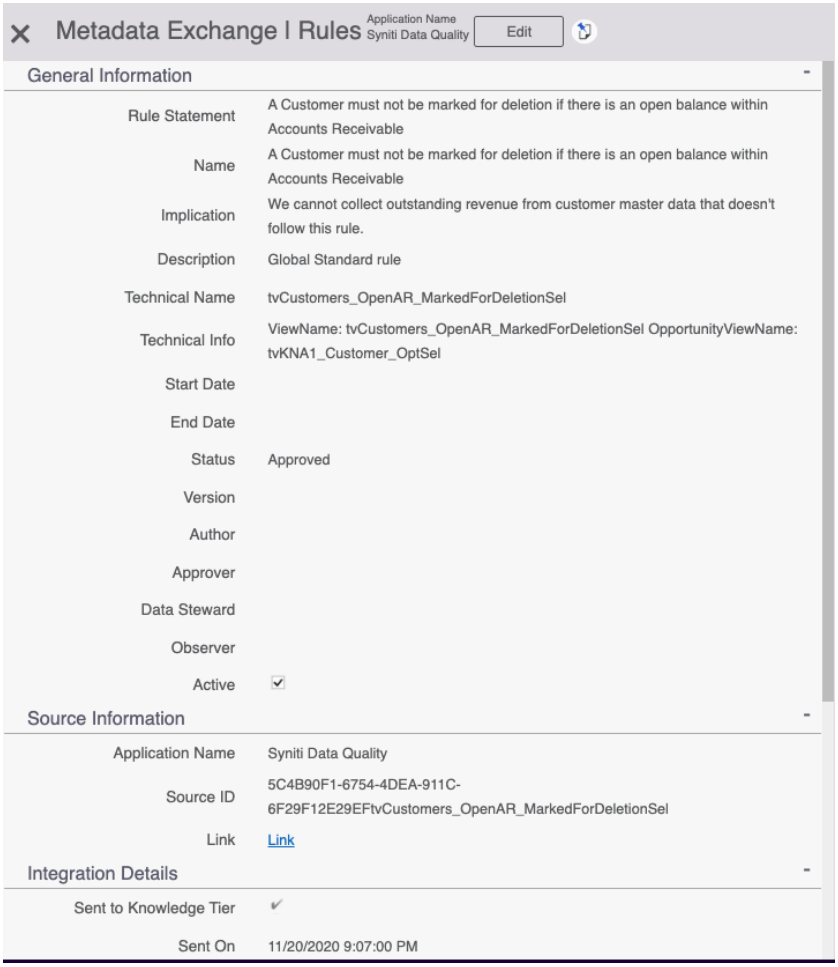
   i. SourceID = DataDialysis.dbo.ddReport.DataSourceID (NOTE: combine with technical name for report view name).

   ii. RuleStatement = DataDialysis.dbo.ddReport.Title

   iii. Implication = DataDialysis.dbo.ddReport.Implication (NOTE: if there is a cost per failure configured in ddReport this will be added dynamically to the implication in the Knowledge Tier.

NOTE: The desired values for the knowledge tier should be filled out at the source application level to support API calls and usability of the data once in the Knowledge Tier. A source link exists to help navigate directly to the Report Repository in SDQ to make the desired changes. If changes are made the "Get Data Quality Rules" will need to executed to reflect the changes. Any rules already sent to the Knowledge Tier will not be affected by the re execution.

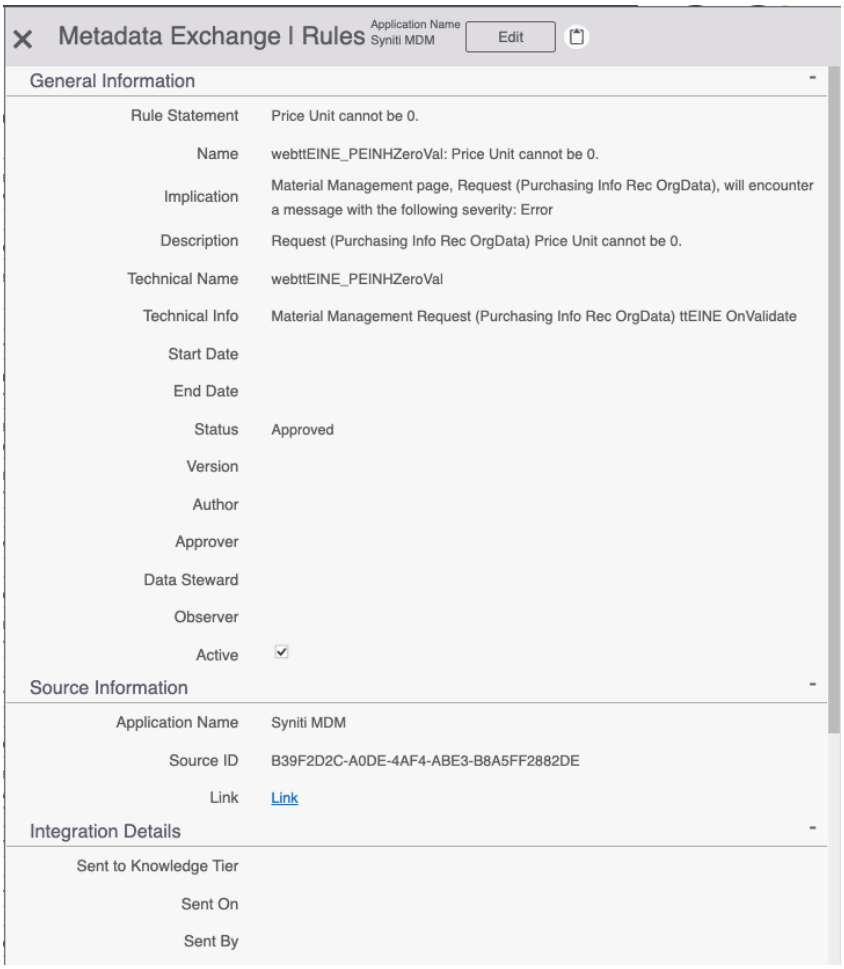
Below is an example of a rule that was discovered by Metadata Discovery. Sending the rules to the Knowledge Tier will be covered later in the help.



5. For discovering Rules out of Syniti Master Data Management
    a. The project has the ability to filter rules and send them ad hoc, one by one or in bulk. The filtering capabilities consist of a combination of the following:
        i. Include one or more MDM categories. NOTE: this will limit to page validations rules related to the webapp related to the MDM category. If a customer webapp needs to be mined for rules, it can be made a MDM category and the custom webappID linked to the category in order to discover rules.
    b. For the enforcement profiles, the link will direct the user to the page valiudation registration filtered to the rule
    c. Once your desired filtering options are set, click “Get MDM Rules”
        i. The results number should increase in the amount of target reports that were in scope based on that filtering.
    d. Documented Mapping from SMDM to Knowledge Tier Required Fields
        i. SourceID = CranSoft.dbo.PageEventValidation.PageValidationID
        ii. RuleStatement = CranSoft.dbo.PageEventValidation.Comment
        iii. Implication = This value is a combination of WebbAppName and Page Description, “will encoutner a message with the following severity: “and the severity level of the validation registration.

NOTE: The desired values for the knowledge tier should be filled out at the source application level to support API calls and usability of the data once in the Knowledge Tier. A source link exists to help navigate directly to the Page Validation Registration in the stewardship tier framework to make the desired changes. If changes are made the “Get MDM Rules” will need to executed to reflect the changes. Any rules already sent to the Knowledge Tier will not be affected by the re execution.

Below is an example of a rule that was discovered by Metadata Discovery. Sending the rules to the Knowledge Tier will be covered later in the help.



Sending Rules to the Knowledge Tier

All scanners that discover rules from an application in the stewardship tier or outside of the stewardship tier will send the metadata about that rule to the Metadata Exchange Rule Repository for staging and preparation before sending the rule to the Knowledge Tier view REST API.

Once the rule discovery is done via Metadata Exchange AND the Knowledge Tier Extract API credentials are configured and tested successfully, rules can be sent one by one or in bulk to the Knowledge Tier. The required fields Rule Statement and Implication MUST be populated to send a rule to the Knowledge Tier. If not, the data MUST be cleansed at the source application level and re-discovered.

From either the Results button on the vertical view of the Syniti Stewardship Tier Metadata Exchange Instance or from Business Concepts --> Rules by Application Menu, one can access a list of rules discovered by the scanning technology.

Once there on the page toolbar exists the following actions/ events:

- Sent to KT – this will send the highlighted rows to the Knowledge Tier. This button will be dithered if any of the following are true:
    - Rule Statement has no value
    - Implication has no value
    - The rule was already sent to the knowledge tier
    - Rule is not active
- Reset All – will reset any rules that for some communication reasons, think was sent to the knowledge tier but it wasn't received.
    - Use this event to prepare rules to be sent to the knowledge tier that were not already sent but the application thinks it was.
- Activate – mark as in scope to be sent to the Knowledge Tier
- Inactivate – mark the rule as not to be sent to the Knowledge Tier

On Vertical View of each rule, you may find the Integration Details

Column Name	**Description**
Sent to Knowledge Tier	This will indicate if the rule has or has not been sent to the Knowledge Tier using the REST API – Send to KT button on the toolbar of the page. If checked the rule has been sent.
Sent On	Date and Time the rule was sent to the Knowledge Tier
Sent By	User name of the Stewardship Tier user that sent the Rule to the Knowledge Tier
KT Link	Link link to the Rule asset in the Knowledge Tier that is auto created via the Send to KT event. This will be blank if the Rule has not been sent yet.
KT Asset ID	The Rule Asset ID (RU*****) number of the Rule Asset created when sent to the

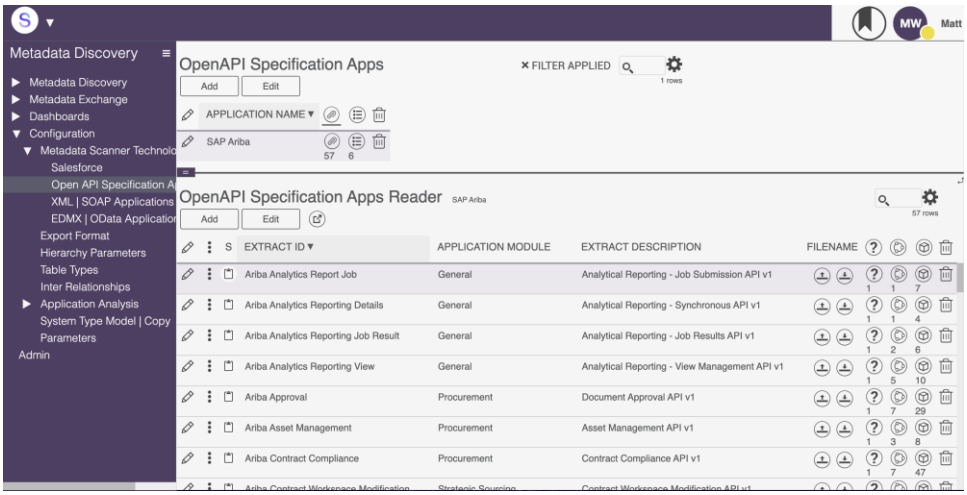
	Knowledge Tier. This will be blank if the rule has not been sent yet.
KT Asset Version	The Rule version number once sent to the Asset ID. This version number is needed to create the enforcement profile and associated links.
Responses	This is a link to the API response page. If there is an error in the API call the error code and message can be found on this subpage.
Reset	Event that is used to reset the Sent to Knowledge Tier metadata so that the Send to Knowledge Tier event may be called again for the Rule. This is helpful if there was an error on the API call on the prior attempt to send the Rule to the Knowledge Tier.
Reset On	The date and time the rule was reset
Reset By	The Stewardship Tier username that reset the rule
Enforcements	Link to one or many enforcement links that will be in the enforcement profile once it is sent to the Knowledge Tier. These links are sent as part of the Send to KT event, but can be sent individually from this subpage if needed.

Retrieve Metadata Using the Open API Specification Parser

The parser needs to be purchased separately and is delivered with application headers, APIs, scanners and metadata.

Support is up to Open API version 3.0.0

Click the Read icon for an extract ID to parse the information and map the Swagger output to the metadata model to create a System Type.



Once a System Type has been created, it is automatically included in the System Type Extension and can be exported to a third-party format, browsed in an application module hierarchy and used to jump start data migration and data quality projects.

Detailed information about using the Swagger Applications parser is distributed on purchase.

Retrieve Metadata Using the SAP HANA Scanner

The SAP HANA scanner technology is supplied at installation and is based on a supplied database. The table dependencies are delivered with the System Type Mode as part of the installation package.

Detailed information about using this scanner is distributed on purchase.

Retrieve Metadata Using the Salesforce Scanner

The Salesforce application-specific scanner must be purchased separately. It uses a plugin to extract metadata. It will automatically create the System Type.

Detailed information about using this scanner is distributed on purchase.

Retrieve Metadata Using the XML Schema Reader

The Schema Reader must be purchased separately. This scanning technology is used to extract metadata from Workday. The generic schema reader will generate objects, attributes and relationships between them used for data lineage.

Detailed information about using this reader is distributed on purchase.

Retrieve Metadata Using the SharePoint Scanner

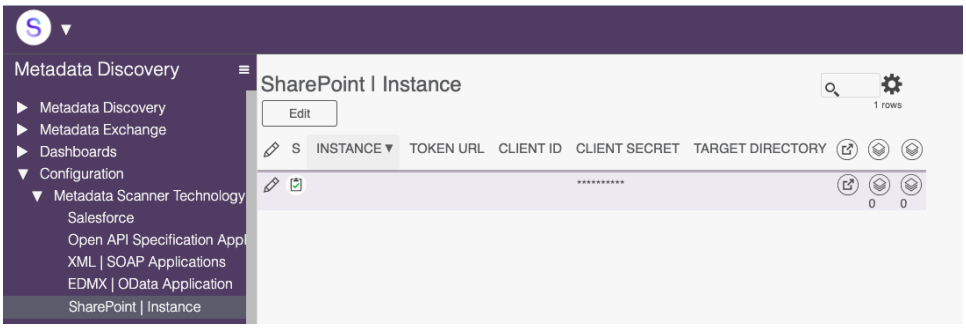
The SharePoint scanner allows a user to not only retrieve SharePoint Groups and the Drives within those groups, but also can get all the folders and items within those folders in a Drive to be used as metadata objects for downstream consumption.

Only one SharePoint Instance is supported at a time.

Please refer to Using the SharePoint plugin appendix at the end of this document for more information on how to use cases, like Downloading a file from or uploading a file to SharePoint.

Information on getting authorizations is in the appendix as well. This requires a specific Token URL, Client ID and Client Secrect.

Navigate to Metadata DIscovery --> Configuration --> Metadata Scanner Technolgoy --> SharePoint | Instance to configure and use the SharePoint Scanner.



Give the Instance a name and enter the parameters following the instructions in the appendix.

Target Directory must be configured to integrate with the Stewardship Tier framework. This folder must contain read and write access to the Windows Accounts:

- The exact service account that Runs the Stewardship Tier / DSP service(s)
- IIS_IUSRS
- IUSR

Additional Metadata Discovery Configuration Options

The following configuration pages are under the **Configuration** menu item in the *Navigation* pane:

- **Export Format**—Lists valid export formats for metadata to send to third party products. A user selects from these items on the *System Type | Extension* page's *Vertical* View on the Action Settings tab. Custom formats can be added.
- **Hierarchy Parameters**—Lists the parameters used when generating the Application Module Hierarchies. These parameters can be edited, and custom parameters can be added.
- **Table Types**—List of tables that is generated as part of the creation of the Application Module Hierarchy. Tables can be added.
- **Inner Relationships**—Lists options used when setting up relationships for Business Concepts. These options populate the VERB CONCEPT list box on the *Concept Relationships* page. These relationships can be edited, and custom ones can be added.

- **Parameters**—The following parameters can be set:
    - **Top Level Name**—Default name used for the top level when the Application Module Hierarchy is created, which can be overwritten.
    - **Top Level Description**— Default description of top-level name used when the Application Module Hierarchy is created, which can be overwritten.
    - **System JSON Batch Size**—displays the number of fields included in a JSON file before a new file is created.
    - **JSON Base File Path**—Displays the location on the application server where JSON files are generated. These files can then be sent to Knowledge Tier to be uploaded.

Copy a System Type Model

In cases where metadata is stored across multiple instances and must be scanned, use the copy feature, accessible under **Configuration > System Type Model | Copy**.

NOTE: The SAP System Type Model is delivered with Stewardship Tier.

To copy a System Type Model:

1. Click **Add** on the *System Type Model | Copy* page.
2. Select System Type Model to use as the basis for the copy from the **COPY FROM MODEL** list box.
3. Enter the name of the new model in the **NEW MODEL NAME** field.
4. Select the data source that stores views in the **LOCAL DATA SOURCE FOR VIEWS** list box.
5. Select the data source from the DATA SOURCE VIEW POINT TO list box.
   NOTE: This is the source for the metadata, and can be the same as, or different from, the data source selected as the LOCAL DATA SOURCE FOR VIEWS list box.
6. Click **Save**.
7. Click the **Copy** icon in the Page toolbar.

The new registration and the new views pointing to the new source are created.

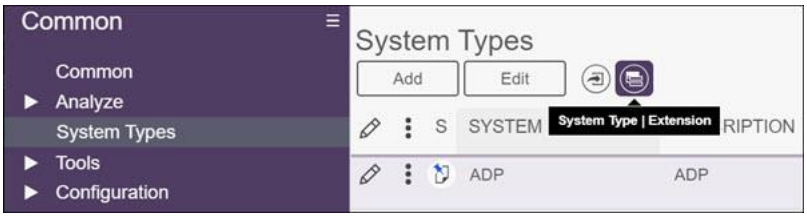
Configure Settings for Creating Application Module Hierarchy

Before analysis, discovery, and other tasks can be performed, certain items must be configured, depending on the technology used. This section describes those configuration tasks, though your installation will probably not require that you perform every one of these tasks.

To access the extension, the user must have Power User access.

The Metadata Discovery pages are automatically populated with System Types from Common. Access System Type Extensions from the *System Types* page in Common.

After the Metadata Discovery application is installed, an icon called System Type | Extension displays in the Page toolbar, which allows the user to access the Metadata Discovery pages for the selected System Type.



The following items must be configured on the *Vertical* View of the *System Type | Extension* page **(Metadata Discovery >System Type | Extension)** before Metadata Discovery tasks can be performed.

Field	Description
Scanner Technology	Select from the list of scanners that are licensed. These scanners could be generic scanners that work for many applications, or application-specific scanners.
Application Module Hierarchy Settings	
Top Level Name	Enter the top level name of the hierarchy, or keep the default based on the System Type.
Top Level Description	Enter a description for the top level or retain the default based on the System Type.
Hierarchy Parameters	
Mode	Controls the behavior of the Application Hierarchy Visualizer. Select from these options: • **None**- • **Drag and Drop Editing**—Allows the user to edit the hierarchy by dragging and dropping nodes. • **Read Only**—The user can view the Application Hierarchy visualization. • **Select NodeID to pass value to page**—Allows the user to select a node for a subsequent task, which then enables a *Select* button which when clicked writes the NodeID of the selected node to the table specified by SelectTarget.
Hierarchy Source	Select the name of the table or view containing the parent/child hierarchy data. See here for the specification of the dataset.
Direction	Select an option for the layout of the OrgChart on the Application Hierarchy Visualizer. Options are: • Bottom to Top • Left to Right • Right to Left • Top to Bottom
On Demand Loading	If checked, the visualization oads 2 levels initially and provides on-demand loading of nodes as the user navigates the hierarchy. This setting is helpful

	for large, unwieldy hierarchies which can be slow to load. It is also useful if you want to start the visualization at any node below the root node and allow upward navigation. If unchecked, the full hierarchy is displayed from the StartNode.

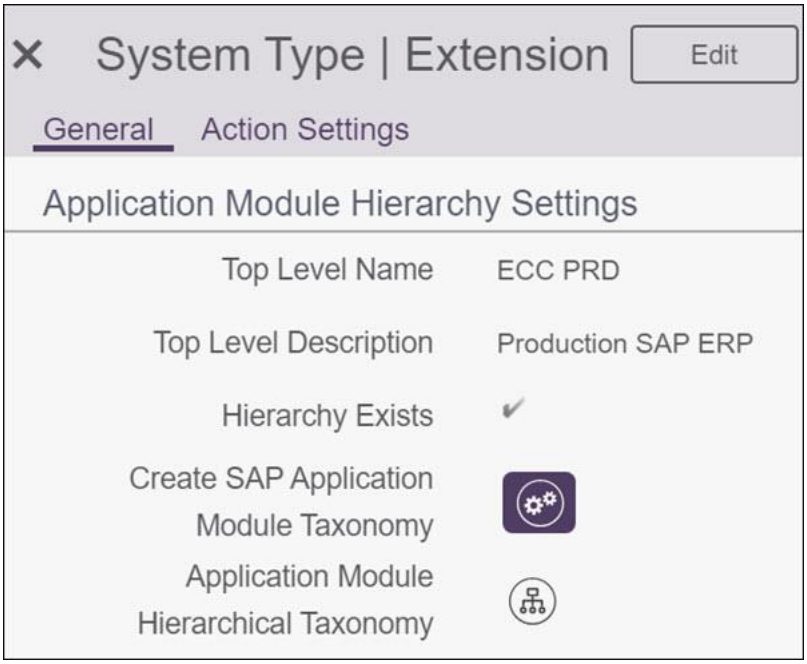
Use Metadata Discovery

- This section contains the following topics:
- Create Application Module Taxonomies
- Access the Application Hierarchy Visualization
- Locate Modules in the Application Hierarchy Visualization Using a Table Name or Transaction
- Analyze Modules
- Export Metadata
- Create Business Concepts
- Send the Business Concept to the Knowledge Tier as a Term
- Use Inherit Tables to Jump Start a Migration Project

Create Application Module Taxonomies

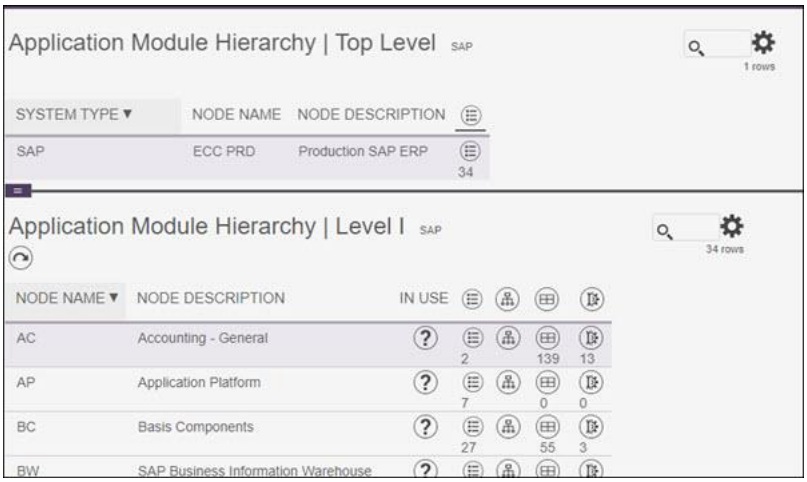
After the settings are configured and saved, icons display on the *System Type* | *Extension* page's *Vertical* View based on the selected scanner technology.

To create the hierarchy, click the **Create {Scanner Technology Name} Application Module Taxonomy** icon.



After the process is complete, the Hierarchy Exists check box is checked.

Click the A**pplication Module Hierarchical Taxonomy** icon to view the results and to access the Application Hierarchy Visualizer.



The Top Level as configured on the *Vertical* View of the *System Type | Extension* page displays in the parent pane, and the child pane displays the secondary level in the hierarchy.

All nodes and submodules display.

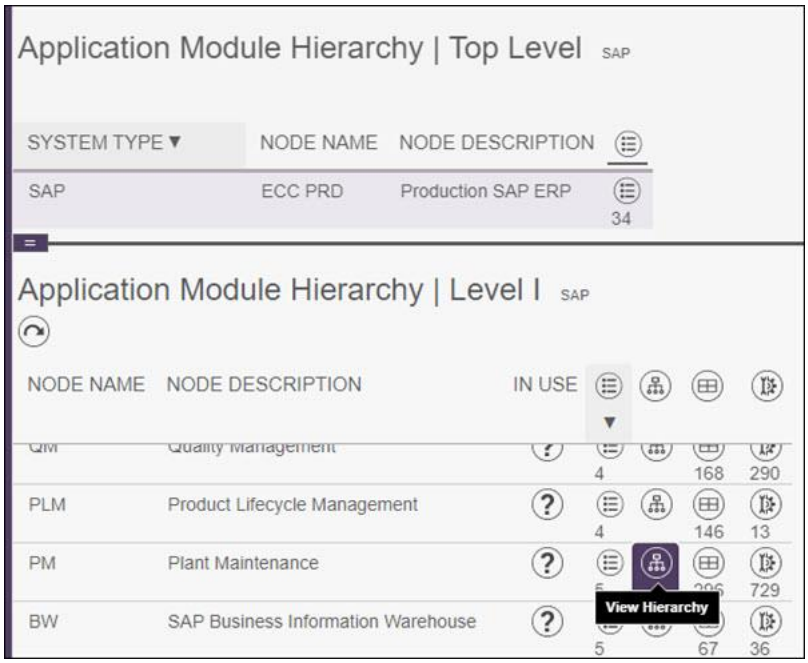
Access the Application Hierarchy Visualization

The Application Module Hierarchy Taxonomy contains these features:

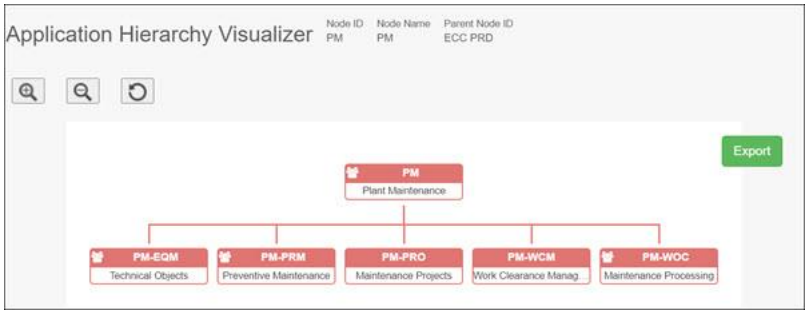
- Graphically display parent-child hierarchies
- Read hierarchical data from any table or view

- Filter hierarchy data prior to loading
- On-demand loading of nodes to handle large hierarchies
- Start visualization on any node in the hierarchy
- Capture selected node for downstream processing
- Drag and drop editing
- Display in 1 of 4 orientations

Click the View Hierarchy icon to view a visualization of the node.



The technical objects display. This allows the user to visualize what has been installed and what is in use in the environment.



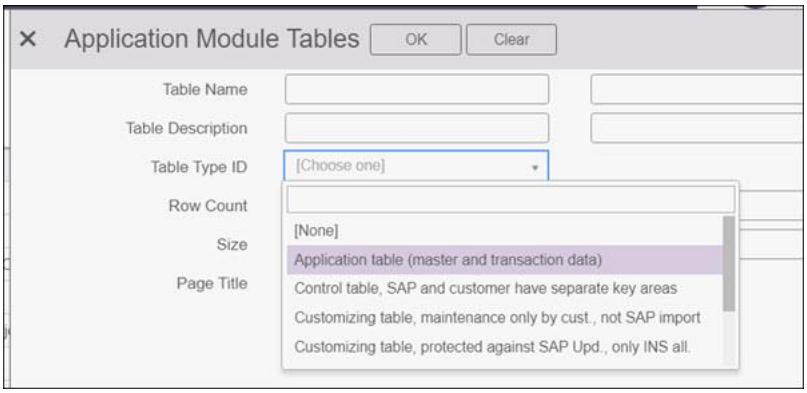
Users can discover relationships among modules by clicking the borders of each node. To share an image of the org chart, click the Export button.

Locate Modules in the Application Hierarchy Visualization Using a Table Name or Transaction

Modules can be located in a hierarchy using a table name search.

On the *System Type | Extension* page, click the **Table Flat Taxonomy** icon.

This taxonomy was generated when the Application Module Hierarchy was created. Use a filter on table types, which was also automatically generated, to narrow search criteria.



Finally, search on the table name.

Similarly, users can also search for modules using transaction codes on the *Application Modules | Transactions* page, accessed via the Transaction Flat Taxonomy icon on the *System Type| Extension* page.

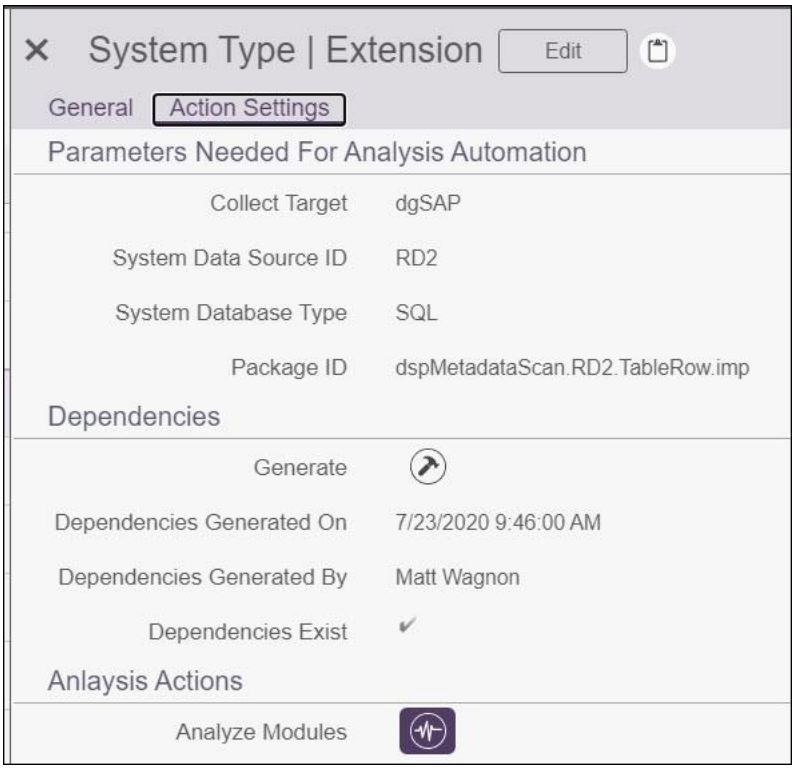
Analyze Modules

The Metadata Discovery application can scan all application modules to determine which ones are in use, out of use, or inconclusive.

To analyze modules:

1. Select **Metadata Discovery > System Type | Extension** in the *Navigation* pane.
2. Click **Vertical View** for the System Type to analyze.

3. Click the **Action Settings** tab.
4. Click the **Analyze Modules** icon.

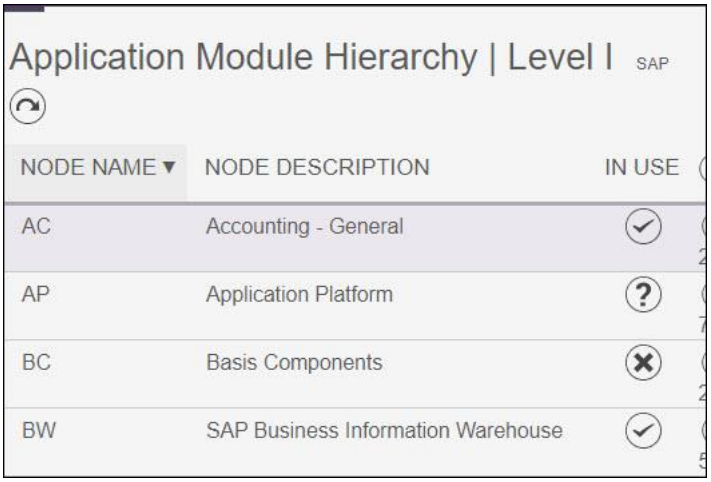


A process runs that analyzes which applications have data and which don't based on the table relationships in the application modules.

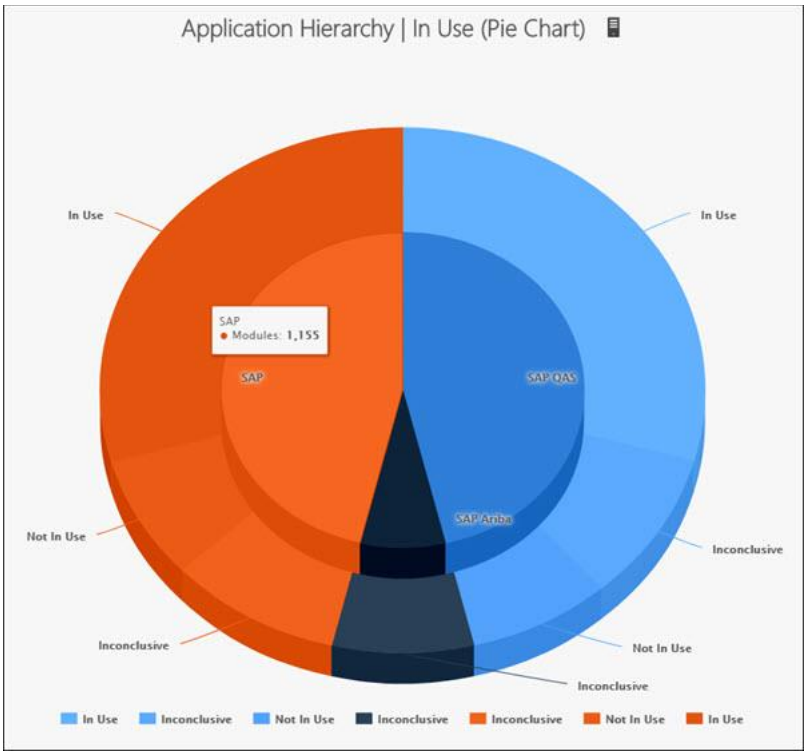
After the process is complete, view the results of the analysis:

1. Click the General tab of the System Type | Extension page's Vertical View.
2. Click the Application Module Hierarchical Taxonomy icon.

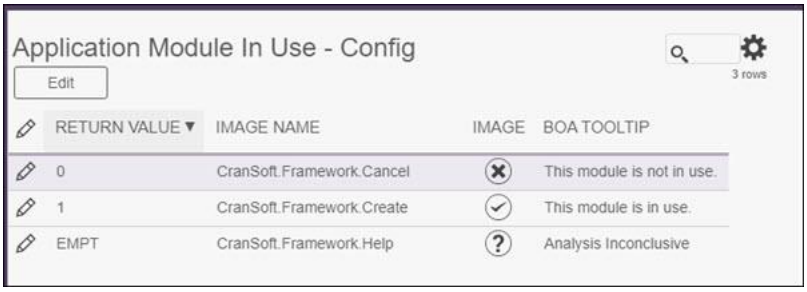
The results display on the *Application Module Hierarchy | Level 1* page.



The results also populate the Application Module Hierarchy | In Use dashboard **(Dashboards > Application Hierarchy In Use).** For each system that has been analyzed, the dashboard displays how many modules are in use, which are not in use and which are inconclusive. Click the chart to drill down to the *Application Modules | Flat Taxonomy* and the *Application Modules Tables* page for more detail.



The *Application Module In Use – Config* page (**Configuration > Application Analysis > Application Module in Use)**, lists the values used to indicate whether a module is being used.



A node is defined as not in use if no rows return data, in use if rows do return data, and inconclusive if the rows are empty. View the query used for analysis on the *Database Queries* page **(Configuration > Application Analysis > Database Queries)**.

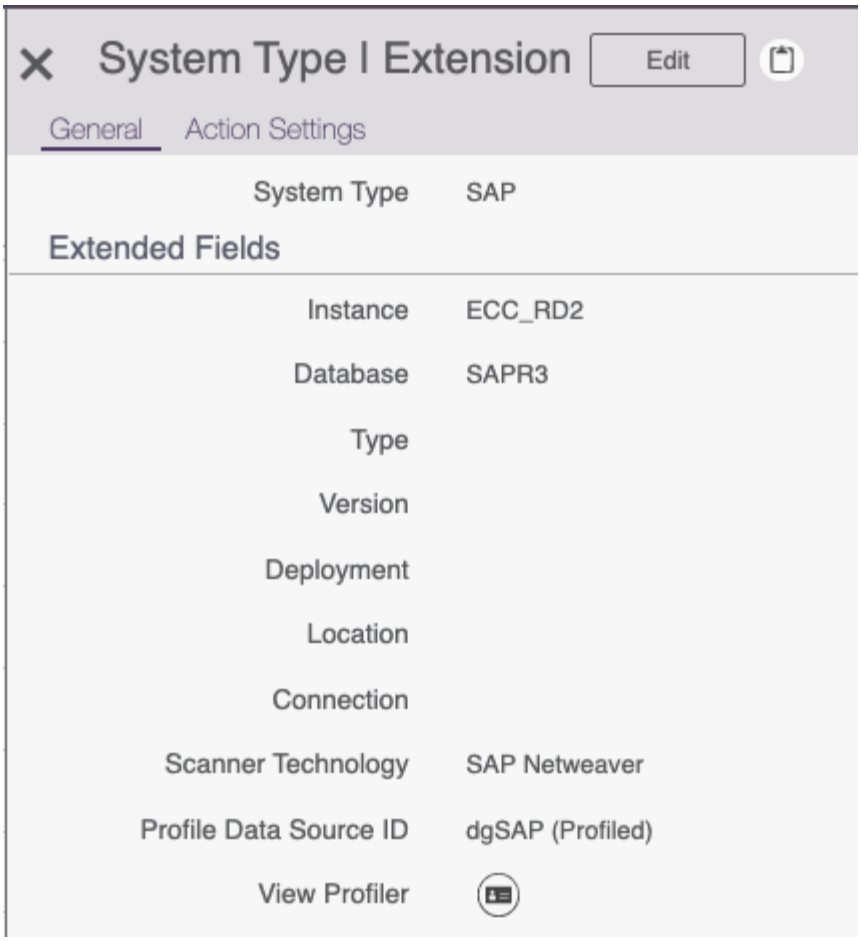
The queries can be updated, and new queries added as needed to expand the In Use Analysis.

A table in the dspMetadataScan database, ttTableRow be populated with the row and table size to support the Analyze modules event. NOTE: This needs to manually happen in the SQL server database level for this version.

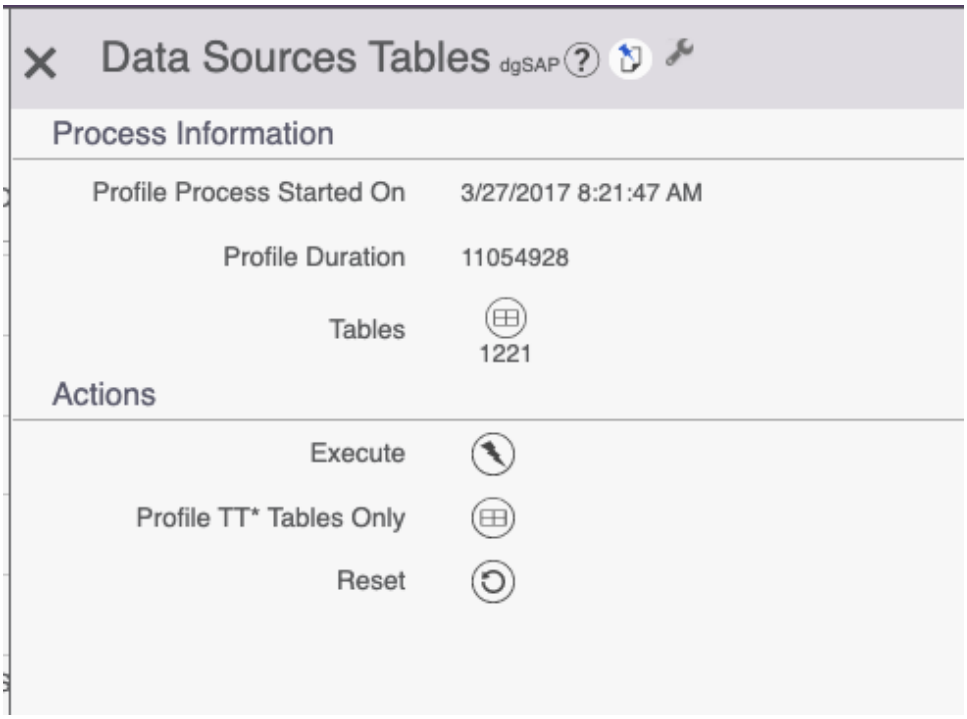
Another valuable piece of the analyze modules is to combine the results of Data Profiling to suggest which data object (I.e. tables) and attributes are being used by the system. In order to do so please connect the system data to the data profiler in the Syniti Stewardship Tier. For documentation on using the Data Profiler please use the online help here.

Once the DataSource for the System Type is profiled you can link the System Type to the Data Source in Profile on the vertical of Metadata Discovery > System Type | Extensions > Vertical View of the desired System type to link the profile data source > Profile Data Source ID dropdown.

Select the name of the DataSource that you wish to leverage the results to enhance the analyzation of the application metadata. The Dropdown of the field will indicate if profile results exist yet or not.



Once linked, the View Profiler button will appear and lead any user to the profiled results in an overlayed page. From there a user can execute data profiling activities, and view the results.



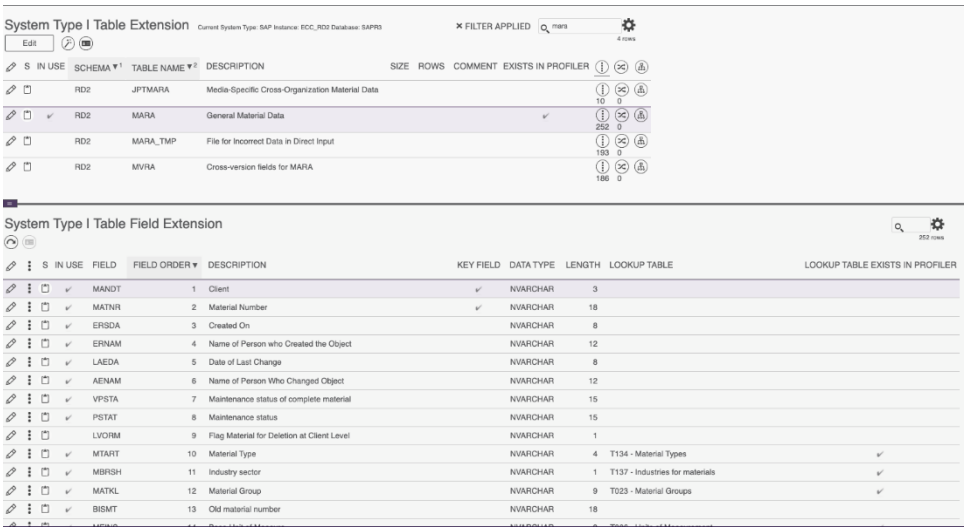
This link will also enable more functionality at the System Type | Table Extension and Table Fields Extension.

If the table is part of the profiled results a user will be able to link from the System Type Table extension record to the Profile results of that same table. This will also allow users to see data from that table as well. Proper security access must be taken into account for giving access to the data profiler because of this.

Once the application is analyzed with the profile results a user ca nsee if a table has been considered “In Use” or not

Additional functionality exists also at the System Type Table Fields Extension level. Each Field is analyzed in the profiler to determine if the field in “In Use” and the lookup table, containing reference data for the field are also indicated if they have been profiled. If so, a user can view the profiled results of the reference data and view the values of the reference data for that field.

Please see all of this in the screenshots below.



To view the System Type Table | Extension profiler results or actual data, look for a button on the toolbar called “View Profiler”

To view the Reference Data | Lookup Table profiler results or actual data, look for a button on the toolbar of the System Type | Table Field Extension called View Profiler for lookup Table.

Export Metadata

Once the metadata has been generated, it can be exported to be consumed by another technology. Export Formats are listed on the *Export Format* page (**Configuration > Export Format**).

To export metadata:

1. Select **Metadata Discovery > System Type | Extension i**n the *Navigation* pane.
2. Click **Vertical View** for the System Type to export.
3. Click the **Action Setting**s tab.
4. Click **Edit**.
5. Select the export format from the **Export Format** list box.
6. Click Save.
7. Click the **Preview Format** icon.

A page displays the data, which can then be downloaded and sent to a third-party application. Refer to Download Data in the Stewardship Tier in the help for more information.

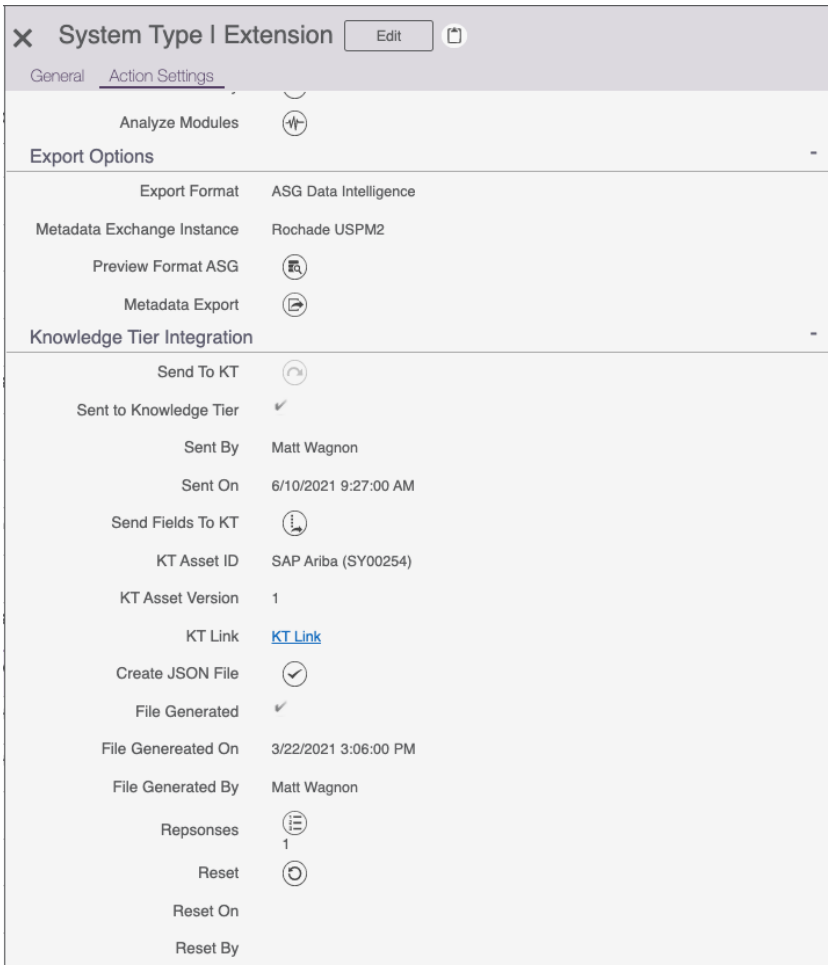
Create System with Technical Metadata from a System Type in the Knowledge Tier

Navigate to the vertical view of the Metadata Discovery > System Type Extension > Vertical View > Action Settings Tab

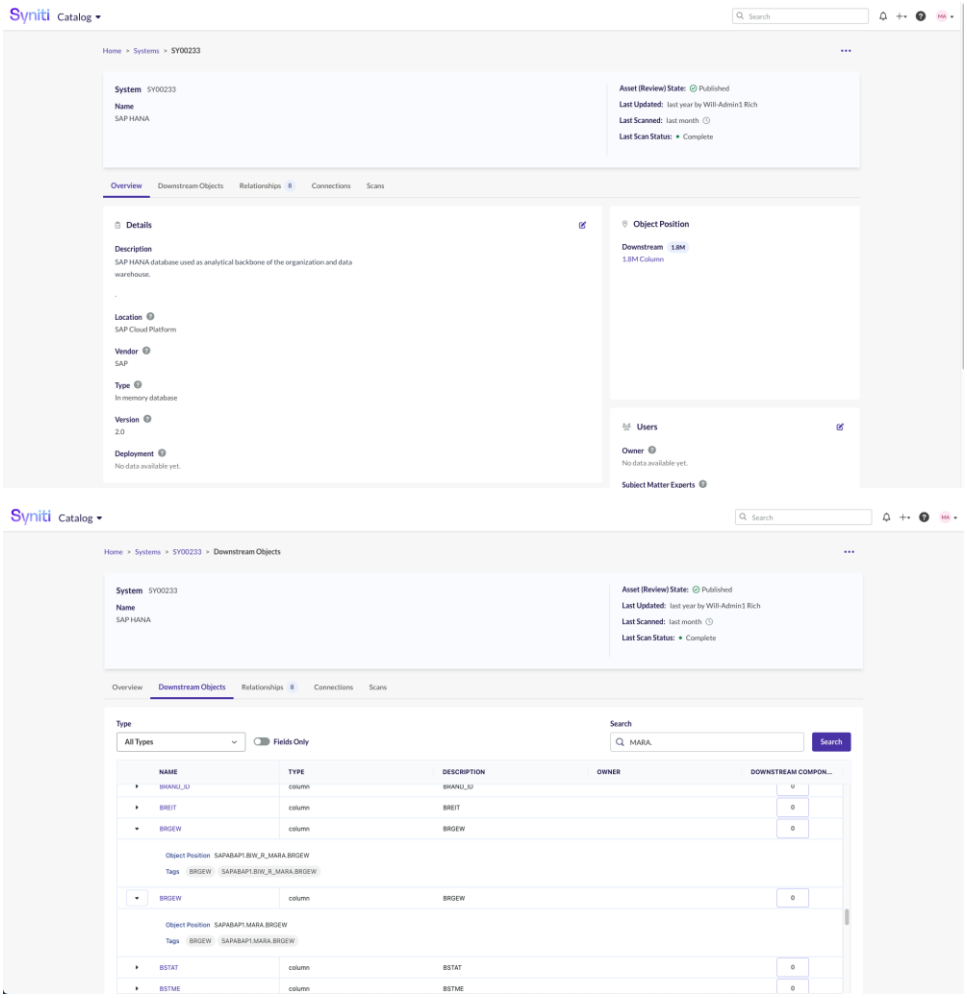
The exists a label called “Knowledge Tier Integration” with a button “Send To Knowledge Tier”

This event will call REST API to POST System and System Fields to the Knowledge Tier and return the internal ID that will populate the following:

- KT Asset ID
- KT Asset Version
- KT Link
- Sent to KT
- Sent By
- Sent On



Once sent the System will be created and all current fields in the system type will be cataloged in the Syniti Knowledge Tier for that System (see below) Those fields are now available for linking to Business Metadata (Terms and Datasets).



Once created, there may exist a scenario where new metadata is created due to new object existing in the data warehouse, upgrades to systems, etc... The process for extracting metadata and creating the system type would need to be reprocessed and then on the same vertical view and event called “Send fields to KT” that can be seen in the two screenshots above, can be leveraged to send only fields and changes that have not yet been sent.

Create Business Concepts

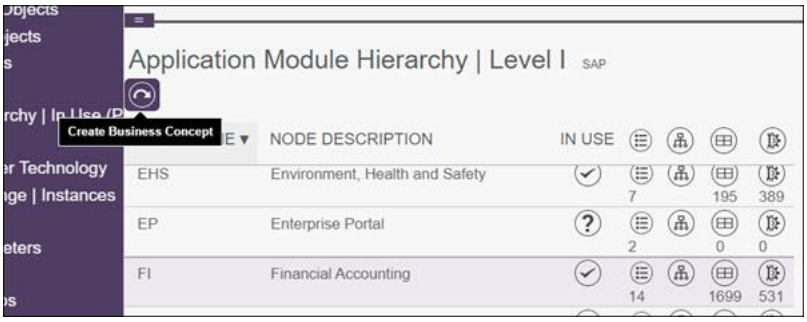
At the start of a migration project, often the first step is to analyze the legacy system to determine which industry-specific products and modules and which custom application modules are being used and not used. Users can perform this analysis automatically using the Metadata Discovery extension,

After the scan is complete, users can designate a node as a Business Concept so it can be reused throughout Syniti Knowledge Platform to jump start data migration or data quality projects.

To create a Business Concept from a node:

1. Select **Metadata Discovery > System Type | Extension i**n the *Navigation* pane.

2. Click **Vertical View** for the System Type.
3. Click the **Application Module Hierarchical Taxonomy** icon.
4. Select the node.
5. Click the **Create Business Concept** icon in the Page toolbar.



6. Click **Business Concepts** in the *Navigation* pane to work with the newly created Business Concept.
7. Click the **Related Concepts** icon for a Business Concept.
8. Click **Add**,
9. Select the Verb Concept from the list box.

Business Concepts can have the following relationships:

- Has an
- Is a synonym of
- Is an
- Is like
- Is part of an
- Is related to an

10. Select the concept for the relationship from the TO CONCEPT ID list box.

Relationships can be assigned to these concepts:

- Data Migration Process Areas
- Data Migration Objects
- Data Quality Objects
- Master Data Management Categories
- Custom Business Concepts

11. Click **Save**.

    Follow these final steps depending on the Business Concepts created.

- If the Business Concept is related to a process area, select **Business Concepts > Data Migration | Process Areas** in the *Navigation* pane, then click the **Create Process Area** icon in the Page toolbar.

The object is made available in ADM for use in a migration project.

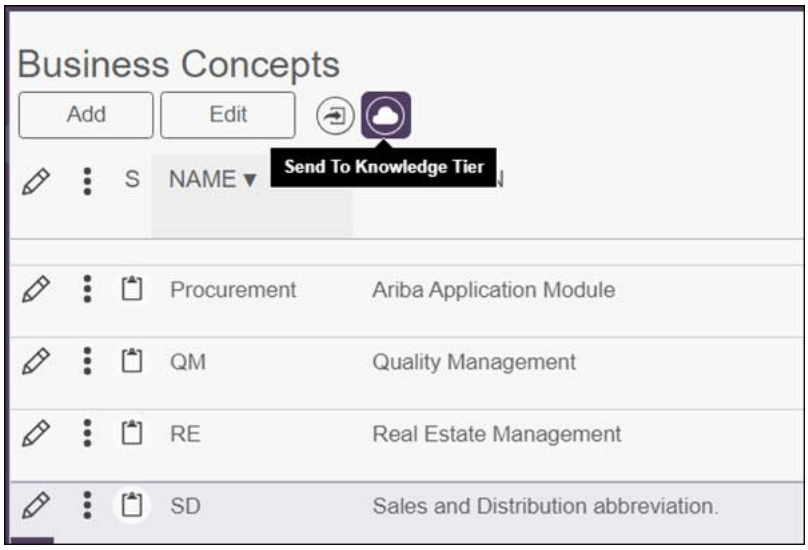
- If the Business Concept is related to a migration object, select **Business Concepts > Data Migration | Objects** in the *Navigation* pane, then click the **Create Migration Object** icon in the Page toolbar.
  The object is made available in ADM for use in a migration project.
- If the Business Concept is related to a Data Quality object, select **Business Concepts > Data Quality | Object**s in the *Navigation* pane, then click the **Create Data Quality Object** icon in the Page toolbar.
  The object is made available in Syniti's Data Quality application for use in object-level reporting for Data Quality metrics.
- If the Business Concept is related to a Master Data Management Category, select **Business Concepts** > MDM | Categories in the *Navigation* pane, then click the **Create Master Data Management Category** icon in the Page toolbar.
  The Category is made available in Syniti's MDM application.

Send the Business Concept to the Syniti Catalog as a Term

Business Concepts can also be sent to the Knowledge Tier as terms. On the *Business Concepts* page (**Navigation pane > Business Concepts**), click the **Send to Knowledge Tier** icon in the Page toolbar. The Business Concept is sent to the Knowledge Tier tenant connected to the current instance of the Stewardship Tier at the next scheduled run time..

NOTE: To create a term, this instance of the Stewardship Tier must have an Agent running that pushes data to a tenant on the Knowledge Tier.

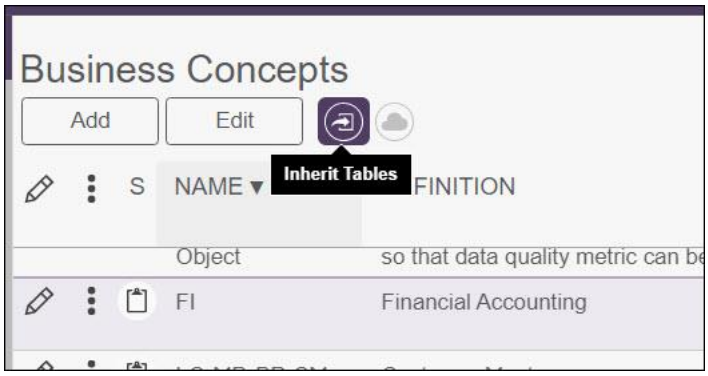
NOTE: In the coming weeks, this functionality will be updated, and Business Concepts will be sent to the Knowledge Tier via an API. The documentation will be updated when that functionality is available.



The *Vertical* View for a Business Concept provides details, such as whether the concept has been integrated into the Knowledge Tier, which System Type it belongs to, which Application module it belongs to and others.

Use Inherit Tables to Jump Start a Migration Project

On the *Business Concepts* page, the **Inherit Tables** icons in the Page toolbar can be used to jump start a migration project. When the user clicks this icon, the tables from the selected Business Concept's Application Module are sent to ADM Target Design.



This will associate the table to a business concept that can be generated as a Migration Process Area or Object and the metadata can be used to auto-populate more Data Migration Console and Target Design Hierarchy.

Additional Configuration Menu Pages

Required Columns – This is used to drive validations on the Collibra Metadata Exchange Integration objects to help guide users on how to make their import views by Asset, Attribute, and Relation. While this is currently used for Collibra only it may be expanded in the future to work with other integration points.

Scenario Types – This page is used for Collibra Metadata Exchange only. IT may be enhanced in the future to work with other integration if needed. It allows for Different endpoints to be assigned to different requests to send data to Collibra by Request and Scenario Type Combination. For example, the REST endpoints needed to Create vs Change vs Delete an asset in Collibra are all different, and therefore different posting logic is required by scenario type.

Appendix

In the Metadata Discovery Web Application, various shared plugins are delivered that can be used to enhance custom web application development or to solve requirements as needed.

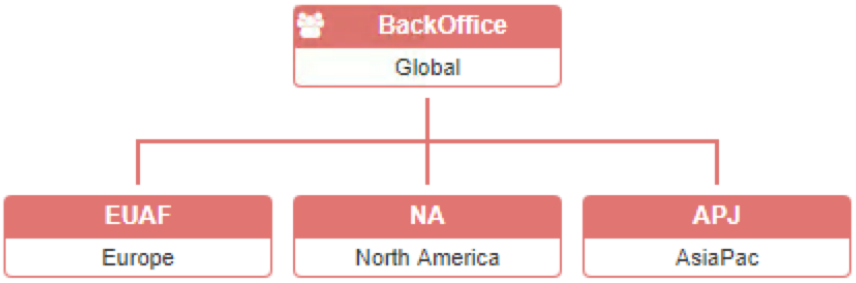
The following shared plugins documentation is part of this appendix

1. Hierarchy Maintenance Plugin
    a. Allows for interactive visualization of hierarchical data
    b. Drag and drop edit of hierarchical data
    c. Select a value from a hierarchy to be placed into a field on a framework page.
2. Knowledge Tier APIs
    a. Supports REST API layer integration for GET, PUT, PATCH endpoints
    b. Delivered SQL server staging table for extracting and sending data to and from the Knowledge Tier from local SQL server environment.
3. SharePoint Plugin
    a. Calls SharePoint APIs to get metadata out of SharePoint
    b. Upload files to SharePoint / OneDrive / Teams
    c. Download files from SharePoint/ OneDrive / Teams
4. Collibra Plugins
    a. Allows for the extraction and bulk loading of metadata to and from Collibra Data Intelligence.

Use the Hierarchy Maintenance Plugin

Data is often organized as a hierarchy to denote inclusiveness or importance of individual entities. Hierarchies are common in business systems, for example product, cost center, geographic region and employee.

A visual representation of a hierarchy can make it much easier to understand. This plugin provides the capability to visualize and edit a hierarchy from within a custom DSP WebApp.



Features

- Graphically display parent-child hierarchies
- Read hierarchical data from any table or view
- Filter hierarchy data prior to loading
- On-demand loading of nodes to handle large hierarchies
- Start visualization on any node in the hierarchy
- Capture selected node for downstream processing
- Drag and drop editing
- Display in 1 of 4 orientations

The visualization for this plugin is provided by an open-source organization chart component called OrgChart. The documentation and source code is available here - https://github.com/dabeng/OrgChart

Version History

Version	Date	Notes
1.0	20 March 2018	Initial release to obtain feedback from real world implementations.
1.1	3 September 2018	- Fixed issue when a *WHERE* clause containing single quotes caused the loader to crash - Added an image export button - Added zoom-in, zoom-out and reset buttons

1.1.1	2 April 2020	Minor documentation update. Added links to Plugin Overview and Register Plugin (not repackaged).

Installation

The deployment package contains various files and folders that must be copied to the target WebApp folder within the DSP installation.

The base folder for a web application is

[DSP Install Directory]/Web/UserArea/*[GUID of Custom WebApp]*

The deployment package contents and their target location are shown below.

Deployment file or folder	Target location
HierarchyMaintenancePlugin.dll	*[base folder]*/Processes/HierarchyMaintenancePlugin.dll
HierarchyMaintenancePlugin.pdb	*[base folder]*/Processes/HierarchyMaintenancePlugin.pdb
web	*[base folder]*/web

Adding to a WebApp

3. In the target WebApp, add `HierarchyMaintenancePlugin.dll` as a new plugin.
    b. For an Overview about Plugins in the Stewardship Tier (DSP) please refer to the online help Plugin Overview.
    c. Quick link to Register a Plugin.
4. Add a static page, for example *Hierarchy Render* and specify `web\Hierarchy.aspx` as the **Static Source**.
5. Add a page that contains the columns required for the plugin data row contract specified here. The DDL for a sample table and page view is available here.
    d. Add a button, for example *Visualise* to the page that links to the static page created in step 2.
    e. Add a page event for the *Visualise* button

f. Add a business rule to the page event and specify *External Page* as the **Procedure Type**, and *HierarchyMaintenance:PageRender* as the **Web App Plugin Type Code**.

Plugin Data Row Contract

Column	Datatype	Description
`MaintenanceID`	`string`	An identifier used when writing data back to the DSP. Only used when writing an edited hierarchy or the selected node to a DSP table, otherwise it is not used.
`Mode`	`string`	Controls the behavior of the hierarchy visualiser. Allowed values are `readonly`, `select` and `edit`. To allow the user to select a node for a subsequent task use the `select` mode, this enables a *Select* button which when clicked writes the `NodeID` of the selected node to the table specified by `SelectTarget`. To allow a user to edit the hierarchy by drag and drop of the nodes use `edit`, this enables a *Save* button which when clicked writes the complete hierarchy to the table specified by `HierarchyTarget`.
`HierarchySource`	`string`	Name of a table or view containing the parent/child hierarchy data. See here for the specification of the dataset.
`WhereClause`	`string`	A filter applied to the `HierarchySource` table/view. Useful for partitioning the data if the source contains multiple hierarchies or parts of the hierarchy should not be displayed.

`Direction`	`string`	Layout of the OrgChart. Allowed values are `t2b`, `b2t`, `l2r`, `r2l` (t = top, b = bottom, l = left, r = right).
`HierarchyTarget`	`string`	Name of the target table to be used when saving an edited hierarchy. Used when `Mode` is `edit`. See here for the specification of the table.
`SelectTarget`	`string`	Name of the table to be used to write the details of the selected node. Used when `Mode` is `select`. See here for the specification of the table.
`StartNode`	`string`	`NodeID` of the node to be used to start the visualisation. All nodes below the start node are displayed (unless `OnDemandLoading` is true). Leave blank to start at the root node. Specifying a value that is not a valid `NodeID` will cause the root node to be used to start the visualisation.
`OnDemandLoading`	`bool`	Control the on-demand loading feature. As large hierarchies can be slow to load and unwieldy, this feature loads 2 levels initially and provides on-demand loading of nodes as the user navigates the hierarchy. It is also useful if you want to start the visualisation at any node below the root node and allow upward navigation. When false the full hierarchy is displayed from the `StartNode`.

Hierarchy Source Dataset

The hierarchy source can be either a table or a view. A non-empty `WhereClause` will be applied when reading data from the hierarchy source.

The columns below must exist in the hierarchy source dataset, any other columns are ignored.

Column	Datatype	Description
NodeID	nvarchar(36)	Node identifier.
NodeName	nvarchar(50)	Node name, which is displayed in the upper section of the node.
NodeDescription	nvarchar(50)	Node description, which is displayed in the lower section of the node.
ParentNodeID	nvarchar(36)	The NodeID of the parent for this node. For the root node, this must be an empty string.

Hierarchy Target Table

When using edit mode, the user can save the current hierarchy to the target table specified using HierarchyTarget.

The columns below must exist in the table, any other columns are ignored. A MERGE statement is used when writing the hierarchy.

Column	Datatype	Description
MaintenanceID	nvarchar(36)	The MaintenanceID passed into the plugin.
NodeID	nvarchar(36)	Node identifier.
ParentNodeID	nvarchar(36)	The NodeID of the parent for this node. For the root node, this will be an empty string.

Select Target Table

When using select mode, the user can save the currently selected node to the target table specified using SelectTarget.

The columns below must exist in the table, any other columns are ignored. A MERGE statement is used when writing the selected node.

Column	Datatype	Description

MaintenanceID	nvarchar(36)	The MaintenanceID passed into the plugin.
NodeID	nvarchar(36)	Node identifier of the selected node.

This plugin is installed and working within the Metadata Discovery Application on the Configuration Menu at the Hierarchy Parameters page.

Use the Knowledge Tier API 3.0 Plugin

Syniti Knowledge Tier data can be created, updated and extracted from the Stewardship Tier (DSP) using the following DSP plugins.

1. KnowledgeTier:ExtractCategory
2. KnowledgeTier:ExtractDataset
3. KnowledgeTier:ExtractDatasetFields
4. KnowledgeTier:ExtractEnforcement
5. KnowledgeTier:ExtractGoal
6. KnowledgeTier:ExtractInitiative
7. KnowledgeTier:ExtractMission
8. KnowledgeTier:ExtractPolicy
9. KnowledgeTier:ExtractProgram
10. KnowledgeTier:ExtractRule
11. KnowledgeTier:ExtractSystem
12. KnowledgeTier:ExtractSystemComponent
13. KnowledgeTier:ExtractTerm
14. KnowledgeTier:ExtractUser
15. KnowledgeTier:ExtractVision
16. KnowledgeTier:MultiCreateUpdate

All the plugins invoke the public Knowledge Tier API v3 and write the results to various SQL Server tables. The API documentation is available here - https://skthelp.syniti.com/TechnicalDocs/SktApi.htm

Version History

Version	Date	Notes
1.0	August 2020	Tactical release to enable early adopter customers.

		Engineering will be creating their own version in the future.
1.1	October 2020	Added optional web proxy support
2.0	November 2020	Major updates to support KT API v2 • Renamed from *extract* to *API* to reflect read and write capability (although some references still exist for ease of upgrade) • Data row contract switched from *API key* to HTTP Basic authentication • Asset Category values are now stored as relationships • CategoryValue plugin removed (now covered by Category) • `KTAssetCatValue` table removed. • `Created_By`, `Created_On`, `Modified_By` and `Modified_On` added to various tables. • `KTDatasetSystem` table removed as system links are stored as relationships. • `KTAssetSupportingDoc.Type` column removed as never used.
2.1	November 2020	Added 2 very basic plugins to provide create capability for KT rules and terms.
2.2	March 2021	• Added version information into the

		plugin assembly (the DLL) • Deprecated `CreateTerm` and `CreateRule` plugins • Amended `KnowledgeTier:ExtractEnforcement` to add `enforcement_app_id` column to the output • Added new plugin `KnowledgeTier:ExtractEnforcementApp` • Added a new plugin to perform multiple update (create and replace) operations - `KnowledgeTier:MultiCreateUpdate` • Added guidance to now add the plugin manually in the Syniti Stewardship Tier
2.3	April/May 2021	• Added a new plugin to extract system fields `KnowledgeTier:ExtractSystemField` • Added a new column (`enf_app_id`) to `KTIRuleEnforcement`. The new column is mapped to `enforcement_app.id` in the request JSON. • Removed mandatory column constraints from `KTIRuleEnforcement` to match the API. • Updated system field extract plugin to allow for a non-existent

		`range` object in the response. • Added URL encoding for the GET `cursor`. • Added column `version` to `KTIResponse` • Added a new plugin to extract individual assets `KnowledgeTier:GenericExtract`
3.0	May 2022	• Removed `KnowledgeTier:GenericExtract` • Removed `KnowledgeTier:CreateTerm` • Removed `KnowledgeTier:CreateRule` V3 API Amendments • Global changes to write categories separately from relationships • Global changes to add `direction` to relationships • Added new table `KTAssetCategories` • Replaced `KnowledgeTier:ExtractSystemFields` with `KnowledgeTier:ExtractSystemComponents`

		• Replaced table `KTSystemField` with `KTSystemComponent` • Removed `KnowledgeTier:ExtractEnforcementApp` • Removed table `KTEnforcementApp` • Modified `KnowledgeTier:ExtractEnforcement` to match v3 structure • Modified `KnowledgeTier:ExtractDataset` to match v3 structure • Added `KnowledgeTier:ExtractDatasetFields` • Added new columns to `KTDatasetField` • Removed unused columns from `KTIAssetRelationship` • Removed `change_description` from rules, terms, datasets and systems update logic • Added new table `KTIAssetCategory` • Added new table `KTISystemScan` • Added new table `KTISystemComponentChildren`

Installation

The deployment package contains a plugin assembly that must be copied to the target WebApp folder within the DSP installation. It also contains some example DDL scripts for source and target tables.

The base folder for a web application is

[DSP Install Directory]/Web/UserArea/*[GUID of Custom WebApp]*

The deployment package contents and their target location are shown below.

Deployment file	Target location
KnowledgeTierAPIPlugins.dll	*[base folder]*/Processes/KnowledgeTierAPIPlugins.dll

For an Overview about Plugins in the Stewardship Tier (DSP) please refer to the online help Plugin Overview.

Quick link to Register a Plugin - DO NOT Register this plugin manually as this is handled by the SQL script that comes with this install package.

Once this plugin is in the proper user area folder and confirmed to be unblocked (check properties of the file on windows to confirm), run the accompanying SQL script. There are directions in the comments in the header of the script. Please follow the instructions now.

AssetExtract_V2wRollback.sql

This script will create all the webapp application pages, service page, buttons, views, procedures, and application events and pages within the desired custom webapp.

Post Install Verification

6. Ensure the process user has access to the service page
    g. Admin → Security → WebApp Security → [WebAppName where Script was run] → Add process user or confirm that the user is in a WebApp group that has access to the service page (FAQ answer; PowerUser will grant it).
7. Validate the WebApp plugin Assembly
    h. Admin → Webapps → [WebAppName where Script was run] → Plugin Assemblies → Validate (click the S icon at the record level) KnowledgeTierAPIPlugins.dll
    i. A Green Message box should appear with 15 plugin count at the plugin icon at the record level.

8. Register the Dynamic Page (Knowledge Tier Data Extraction) to a menu within the webapp the script was run for.
9. Navigate to the Dynamic Page (Knowledge Tier Data Extraction)
    j. Add A record
    k. Enter ExtractID value (i.e. Tenant Customer Name)
        i. SAVE navigate to Vertical view
        ii. Enter basepath
            1. https://api.syniti.com
        iii. If proxy server is required to access basepath enter the value
        iv. Enter Usernam and Password from the API key received from Syniti Customer Success or Support.
    l. Test by press 1 extract button at a time.
        i. Successful test will have data in the SQL tables and get a successful test message
        ii. Failure will get an error message. Please open a support ticket at support.syniti.com if assistance is needed

Technical Documentation

Extract Plugin Data Row Contracts

The extract plugins (except `KnowledgeTier:ExtractSystemComponent` and `KnowledgeTier:ExtractDatasetFields`) use the following contract

Column	Datatype	Description
`ExtractID`	`string`	An identifier that can be used to link the request to the response data.
`Username`	`string`	The API authentication credentials provided to your organization by Syniti Customer Success.
`Password`	`string`	
`BasePath`	`string`	Use `https://api.syniti.com`
`ProxyAddress`	`string`	The URI of an optional proxy server. For example, `http://myproxy` or `http://ourwebproxy:8080` if a specific port must be used. Use an empty string or null if a proxy is not required.

`KnowledgeTier:ExtractSystemComponent` uses the following contract

Column	Datatype	Description
ExtractID	string	An identifier that can be used to link the request to the response data.
Username	string	The API authentication credentials provided to your organization by Syniti Customer Success.
Password	string	
BasePath	string	Use https://api.syniti.com
ProxyAddress	string	The URI of an optional proxy server. For example, http://myproxy or http://ourwebproxy:8080 if a specific port must be used. Use an empty string or null if a proxy is not required.
SystemID	string	The ID of the system whose components to extract.

KnowledgeTier:ExtractDatasetFields uses the following contract

Column	Datatype	Description
ExtractID	string	An identifier that can be used to link the request to the response data.
Username	string	The API authentication credentials provided to your organization by Syniti Customer Success.
Password	string	
BasePath	string	Use https://api.syniti.com
ProxyAddress	string	The URI of an optional proxy server. For example, http://myproxy or http://ourwebproxy:8080 if a specific port must be used. Use an empty string or null if a proxy is not required.
DatasetID	string	The ID of the dataset whose fields to extract.

Inbound Plugins

KnowledgeTier:MultiCreateUpdate uses the following contract.

Column	Datatype	Description
RequestID	string	An identifier that is used to assemble the inbound request and can be used to associate the request to the response. RequestID can be used to group multiple items together for bulk operations.
Username	string	The API authentication credentials provided to your organization by Syniti Customer Success.
Password	string	
BasePath	string	Use https://api.syniti.com
ProxyAddress	string	The URI of an optional proxy server. For example, http://myproxy or http://ourwebproxy:8080 if a specific port must be used. Use an empty string or null if a proxy is not required.
HTTPVerb	string	POST or PUT. See the table below for valid combinations of HTTPVerb and EndPoint.
Endpoint	string	API resource URL. See the table below for valid combinations of HTTPVerb and EndPoint.

The values passed as HTTPVerb and Endpoint determine which API operation is invoked, only the combinations in the table below are supported.

See the API definition here for details of the operations and the values that need to be specified in each request - https://app.swaggerhub.com/apis-docs/Syniti2/syniti-knowledge_tier_api/3.0

HTTP Verb	Endpoint	Source Tables
POST	/v3/systems	KTISystem KTIAssetRelationship

		KTIAssetContact KTIAssetSupportingDoc KTIAssetCategory
PUT	/v3/systems/{id}	KTISystem KTIAssetRelationship KTIAssetContact KTIAssetSupportingDoc KTIAssetCategory
POST	/v3/systems{id}/scans	KTISystemScan KTISystemComponent KTISystemComponentChildren
POST	/v3/datasets	KTIDataset KTIAssetRelationship KTIAssetContact KTIAssetSupportingDoc
PUT	/v3/datasets/{id}	KTIDataset KTIAssetRelationship KTIAssetContact KTIAssetSupportingDoc
POST	/v3/datasets/{id}/fields	KTIDatasetField
POST	/v3/terms	KTITerm KTIAssetRelationship KTIAssetSponsor KTIAssetSupportingDoc
PUT	/v3/terms/{id}	KTITerm KTIAssetRelationship KTIAssetSponsor KTIAssetSupportingDoc
POST	/v3/rules	KTIRule KTIAssetRelationship KTIAssetSponsor KTIAssetSupportingDoc
PUT	/v3/rules/{id}	KTIRule KTIAssetRelationship KTIAssetSponsor KTIAssetSupportingDoc
POST	/v3/enforcements	KTIRuleEnforcement KTIAssetSupportingDoc

The plugin uses `RequestID` to filter the source tables, and multiple objects can be created within a specific request. Use `LinkID` in the tables to group data across the various sub-tables. When an update operation is requested, use the `id` column in the source table to determine the asset to update.

Each request will result in one or many response entries in `KTIResponse`, the structure of the table is as follows.

Column	Datatype	Description
RequestID	string	The RequestID passed in the data row contract.
LinkID	string	The LinkID from the source table(s).
id	string	The id returned in a successful API operation's response.
http_code	string	The HTTP response, for example OK, BadRequest, Unauthorized, NotFound, Created, Conflict.
api_response	string	The response body or message.

This plugin is used on the Knowledge Tier data extraction page. It can be called from a custom page to interoperate with the Knowledge Tier API endpoints as desired.

Using the SharePoint Plugin

SharePoint DSP Plugins v2.0

SharePoint, Teams and OneDrive are often used on DSP projects to store, share and collaborate with project documentation and data. Technically, files in these services are stored in a Drive.

This package provides 4 plugins that allow a DSP instance to upload and download files from a Drive.

10. SharePoint:GroupDrives - read metadata for Groups and Drives
11. SharePoint:FileUpload - upload a file to a Drive
12. SharePoint:ListDriveItems - read the list of drive items (folders and files)
13. SharePoint:DriveItemContents - download a file from a Drive

The plugins are designed to be configured by administrators for background processes and use *application permissions* to interact with the services, which will require a SharePoint administrator to configure access.

Once installed and configured, the Groups and Drives metadata is used to select a Drive. This Drive is then used as the target for an upload, or the source for listing and download.

Tip: a free Office 365 Developer account provides access to SharePoint, Teams and OneDrive - https://developer.microsoft.com/en-us/microsoft-365/dev-program.

Version History

Version	Date	Notes
1.0	19 March 2020	Initial release.
1.0.1	2 April 2020	Minor documentation update. Added links to Plugin Overview and Register Plugin (not repackaged).
2.0	2 May 2022	Added support for listing a drive's folders and files, and downloading a file.

Installation

The deployment package contains a plugin assembly that must be copied to the target WebApp folder within the DSP installation. It also contains a DDL script for source and target tables.

The base folder for a web application is

`[DSP Install Directory]/Web/UserArea/[GUID of Custom WebApp]`

The deployment package contents and their target location are shown below.

Deployment file	Target location
`SharePointPlugin.dll`	`[base folder]/Processes/SharePointPlugin.dll`

For an Overview about Plugins in the Stewardship Tier (DSP) please refer to the online help Plugin Overview.

Quick link to Register a Plugin

SharePoint:GroupDrives

This plugin extracts a list of the Office 365 Groups, and the list of Drives for each Group. Groups are automatically created in Office 365 and are associated with each SharePoint site, Team and OneDrive. Each Group can have multiple Drives which is where documents are stored (aka a document library). Groups and Drives have human readable names and descriptions, and these can be used within a WebApp to select the correct target drive for the file upload.

This plugin would not be used frequently, it is only used to select a target Drive.

The plugin writes the Groups and Drives metadata to tables in the WebApp database.

Plugin Data Row Contract

Column	Datatype	Description
`TokenURL`	`string`	The OAuth token URL for the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
`ClientID`	`string`	The Client ID of the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
`ClientSecret`	`string`	The Client Secret of the DSP's Office 365 AD app registration. See Configuring Office 365 Access.

Sample DDL for a source table (`ttSharePointUpload`) is in the deployment package, see `SharePointPluginDDL.sql`.

Output Tables

The plugin writes the Office 365 metadata to 2 tables in the web app database.

Groups

The Office 365 groups are written into a table called `ttSharePointGroup`. The table must exist in the WebApp database. Empty the table before invoking the plugin.

The columns below must exist in the table, any other columns are ignored.

Column	Datatype	Description
`GroupID`	`nvarchar(50)`	Office 365 Group identifier
`DisplayName`	`nvarchar(100)`	Group display name
`Description`	`nvarchar(500)`	Group description

Sample DDL for a table conforming to the structure is in the deployment package, see `SharePointPluginDDL.sql`.

Drives

The Office 365 group drives are written into a table called `ttSharePointDrive`. The table must exist in the WebApp database. Empty the table before invoking the plugin.

The columns below must exist in the table, any other columns are ignored.

Column	Datatype	Description
`GroupID`	`nvarchar(50)`	Office 365 Group identifier
`DriveID`	`nvarchar(100)`	Office 365 Drive identifier
`Name`	`nvarchar(100)`	Drive display name. This defaults to `Documents` in most cases, but can be changed in Office 365.
`Description`	`nvarchar(500)`	Drive description. This is empty by default, but can be changed in Office 365.
`WebURL`	`nvarchar(500)`	Office 365 Web URL of the drive. This can be used in a browser to validate the drive choice.

Sample DDL for a table conforming to the structure is in the deployment package, see `SharePointPluginDDL.sql`.

SharePoint:FileUpload

This plugin uploads a file to an Office 365 drive. A URL to the uploaded file is written to the output table.

It is currently limited to files up to 4MB in size.

Plugin Data Row Contract

Column	Datatype	Description

`RequestID`	`string`	An identifier that is used to link the request to the response data.
`TokenURL`	`string`	The OAuth token URL for the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
`ClientID`	`string`	The Client ID of the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
`ClientSecret`	`string`	The Client Secret of the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
`DriveID`	`string`	The target Office 365 drive identifier. Use the SharePoint:GroupDrives plugin to find the target drive.
`LocalFileRef`	`string`	The fully qualified location and filename of the file to upload to the Office 365 drive. For example `c:\DSP\upload\failed_data.csv`

TargetFileRef	string	The target filename and location relative to the root of the drive. If a folder structure is specified, it will be automatically created if it does not exist. For example, to write a file in the root, simply specify a file name - failed_data.csv. To write a file in a folder - phase1/customer/failed_data.csv Do not include a leading forward slash - /. If uploading to Teams, prefix the target with the channel name, for example General/failed_data.csv.

Sample DDL for a source table is in the deployment package, see SharePointPluginDDL.sql.

Output table

A URL to the uploaded file is written into a table called ttSharePointFileUpload. The table must exist in the WebApp database. The plugin uses a SQL MERGE statement to INSERT/UPDATE the data in the table using RequestID as the key.

The columns below must exist in the table, any other columns are ignored.

Column	Datatype	Description
RequestID	nvarchar(50)	RequestID from the data contract
WebURL	nvarchar(500)	Office 365 file web URL

Sample DDL for a table conforming to the structure is in the deployment package, see `SharePointPluginDDL.sql`.

SharePoint:ListDriveItems

This plugin extracts the list of folders and files from an Office 365 drive. The list of items is written to the output table. Sub-folders are automatically read.

Plugin Data Row Contract

Column	Datatype	Description
`RequestID`	`string`	An identifier that is used to link the request to the response data.
`TokenURL`	`string`	The OAuth token URL for the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
`ClientID`	`string`	The Client ID of the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
`ClientSecret`	`string`	The Client Secret of the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
`DriveID`	`string`	The Office 365 drive identifier. Use the SharePoint:GroupDrives plugin to find the target drive.

Sample DDL for a source table is in the deployment package, see `SharePointPluginDDL.sql`.

Output table

The list of items is written into a table called `ttSharePointDriveItem`. The table must exist in the WebApp database.

The columns below must exist in the table, any other columns are ignored.

Column	Datatype	Description
`RequestID`	`nvarchar(50)`	RequestID from the data contract
`itemId`	`nvarchar(100)`	Office 365 drive item identifier
`path`	`nvarchar(500)`	Office 365 full path to the item
`name`	`nvarchar(500)`	Office 365 item name (the folder or filename)
`isFolder`	`bit`	`True` if the item is a folder
`isFile`	`bit`	`True` if the item is a file

Sample DDL for a table conforming to the structure is in the deployment package, see `SharePointPluginDDL.sql`.

SharePoint:DriveItemContents

This plugin downloads a file from an Office 365 drive. The file is written to a location on the DSP server. The original filename is used when writing the file to the target directory.

Note that this plugin requires TLS1.2 support on the DSP server; older versions of Windows might need additional configuration to support this. See here for more information - https://docs.microsoft.com/en-us/graph/api/driveitem-get-content?view=graph-rest-1.0&tabs=http.

Plugin Data Row Contract

Column	Datatype	Description

RequestID	string	An identifier that is used to link the request to the response data.
TokenURL	string	The OAuth token URL for the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
ClientID	string	The Client ID of the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
ClientSecret	string	The Client Secret of the DSP's Office 365 AD app registration. See Configuring Office 365 Access.
DriveID	string	The Office 365 drive identifier. Use the SharePoint:GroupDrives plugin to find the target drive.
ItemID	string	The Office 365 drive item identifier. Only files can be downloaded. Use the SharePoint:ListDriveItems plugin to obtain a list of valid items.
TargetDirectory	string	The target directory on the DSP server, for example `d:\sharepoint\mapping\`. The value must include a trailing backslash.

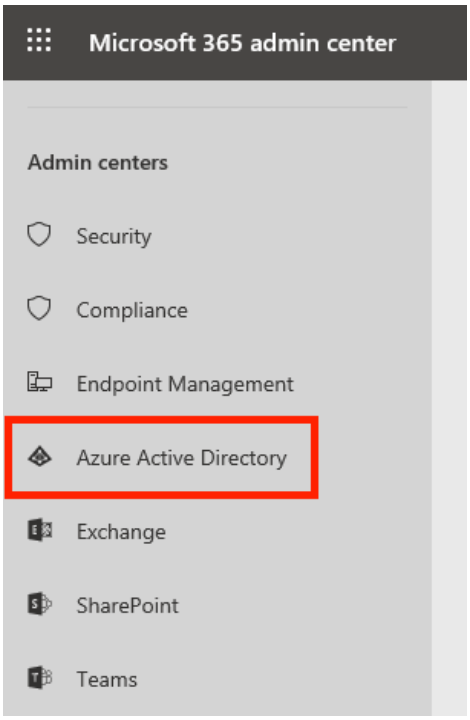
Sample DDL for a source table is in the deployment package, see `SharePointPluginDDL.sql`.

Configuring Office 365 Access

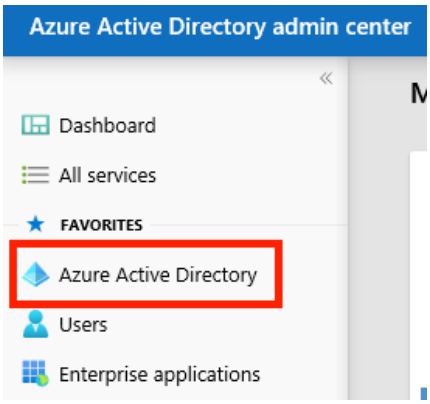
The plugins authorise against the target service without a specific user, it requires a DSP instance to be registered as an app, and the app to have application permissions assigned. The

following steps must be carried out by a SharePoint administrator to register and assign permissions.

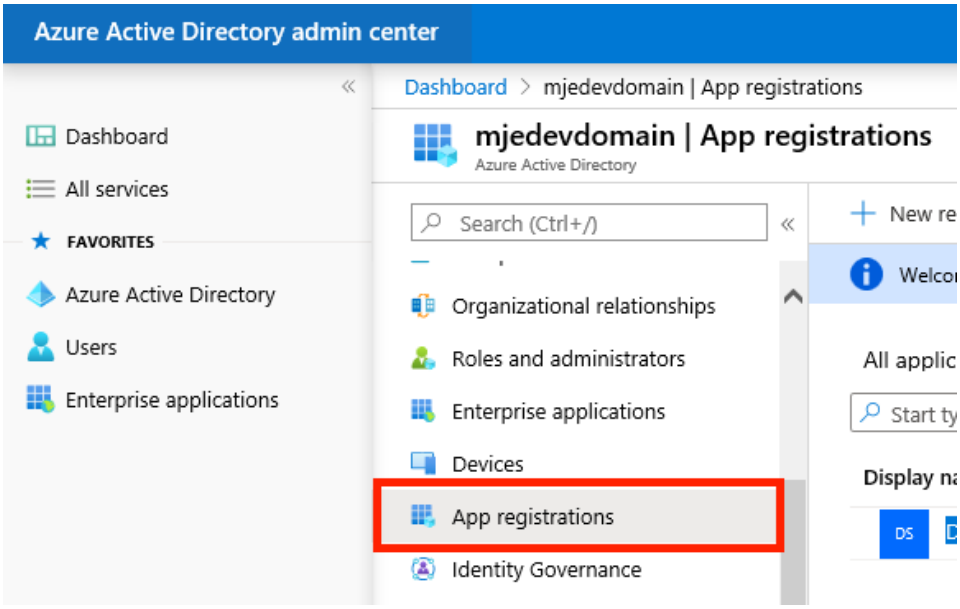
14. Open the Office 365 admin center and enter the Azure Active Directory



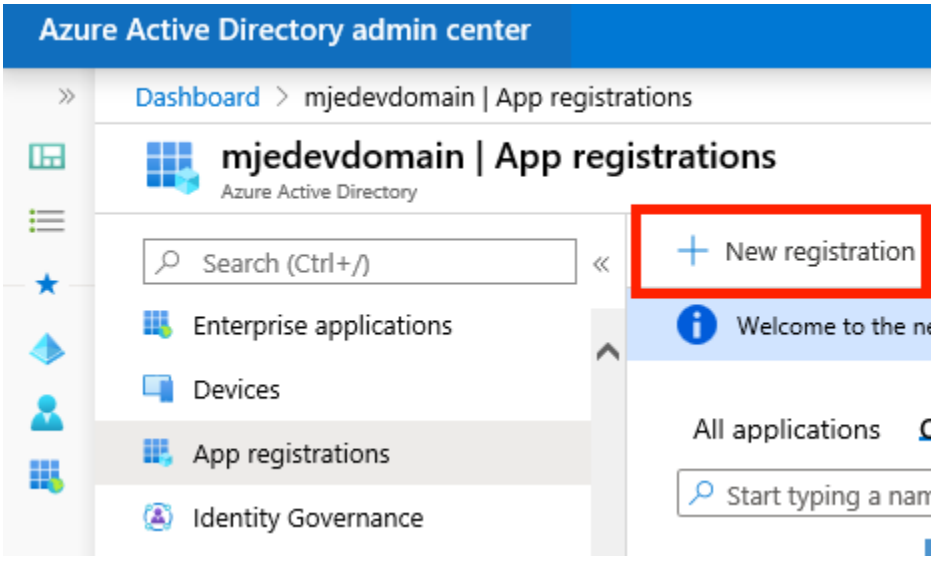
15. In Azure Active Directory admin center select Azure Active Directory



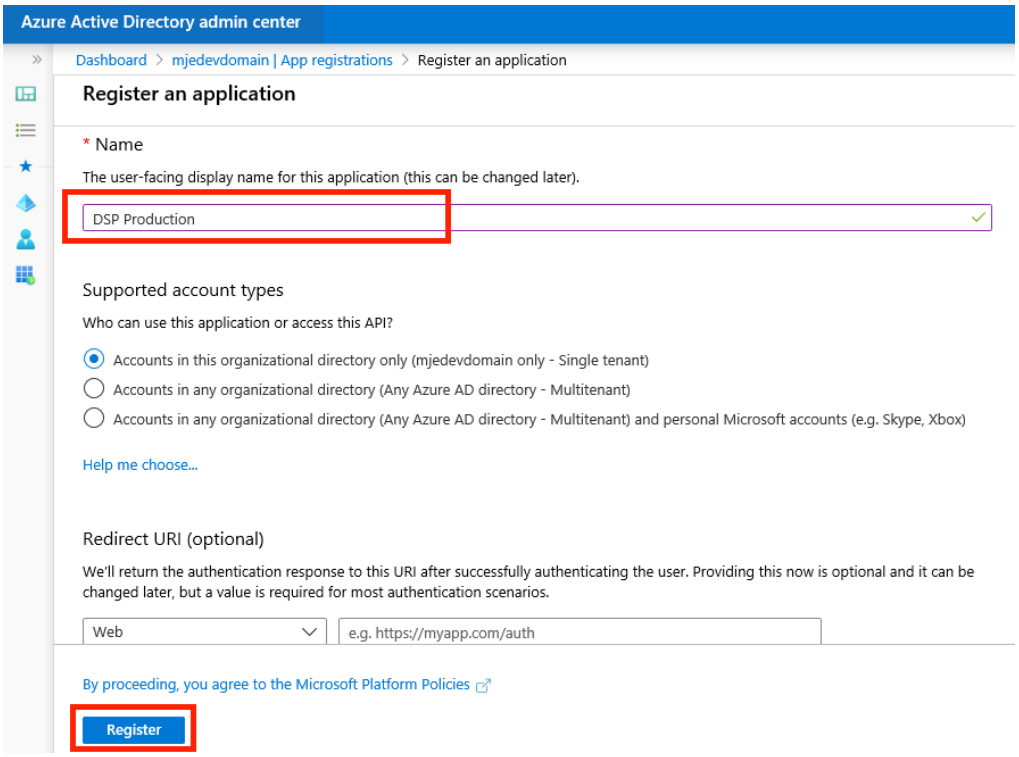
16. Select App registrations



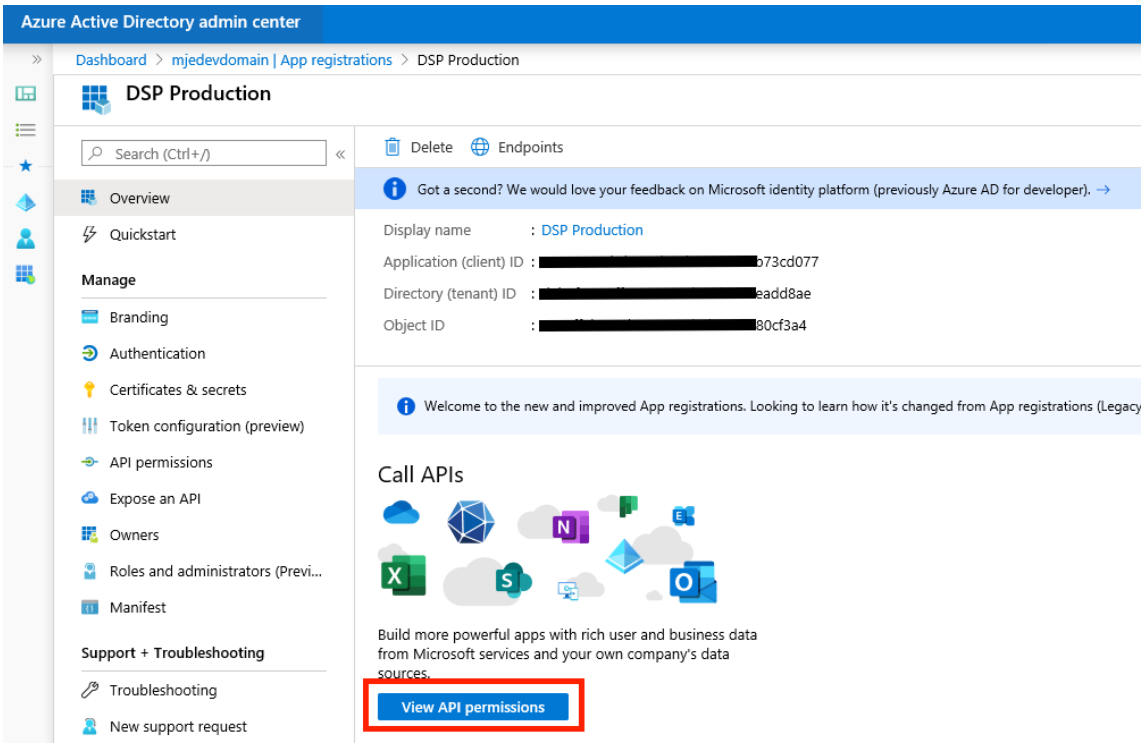
17. Select New registration



18. Enter a valid name for the application and click *Register*.

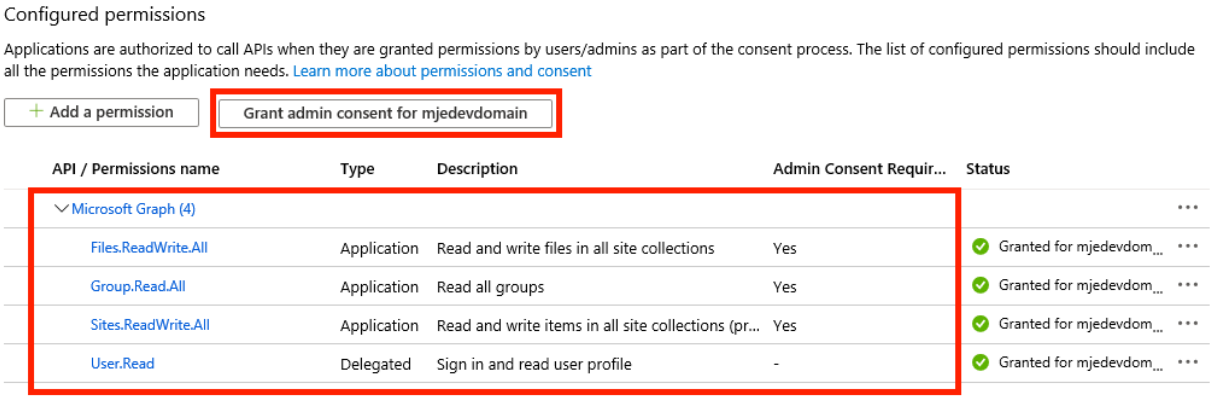


19. Click View API permissions to grant the required permissions.

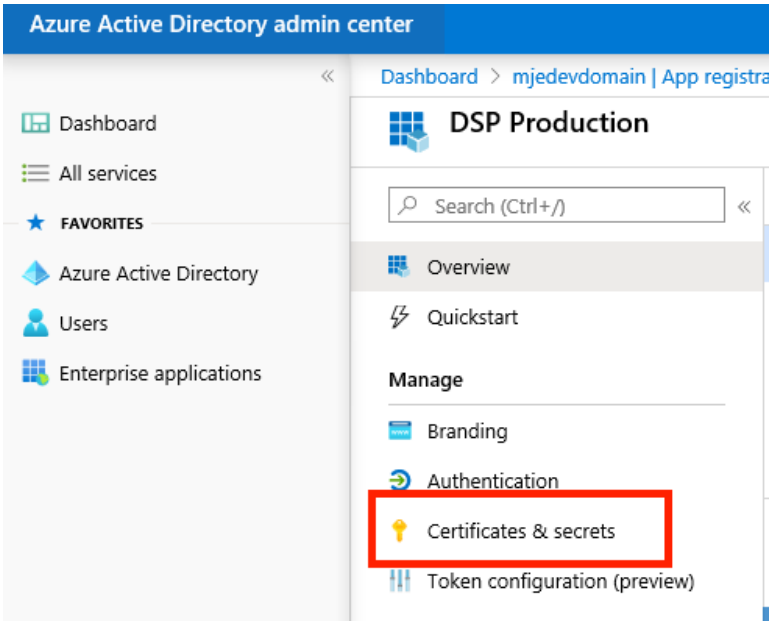


20. Add the following Microsoft Graph Application permissions, and Grant admin consent for them.

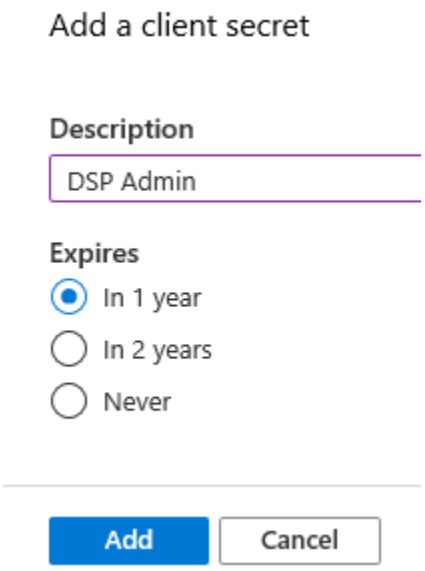
- Files.ReadWrite.All
- Group.Read.All
- Sites.ReadWrite.All



21. Navigate to Certificates and Secrets

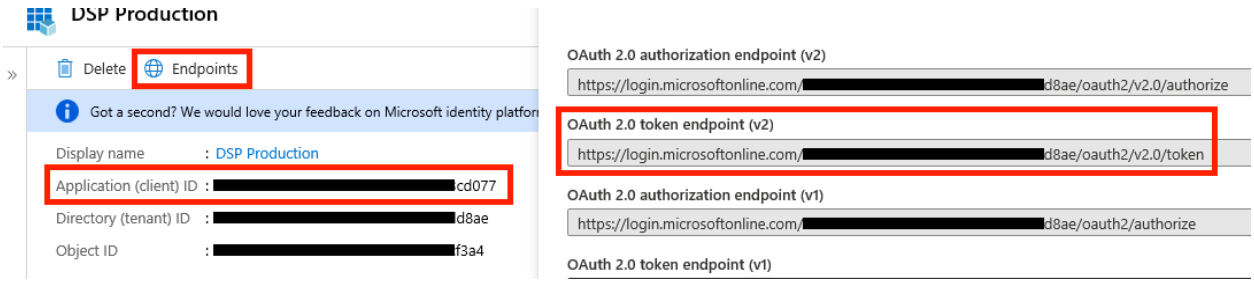


22. Create a new Client secret; specify a valid description and duration. Copy the generated value immediately, as once you move off the current blade (tab) you cannot retrieve it.



23. For the newly created app registration, copy the value for client secret (from the previous step) together with the client ID and OAuth 2 token endpoint (v2).

Securely provide these 3 values to the DSP administrator.



This page provides additional background information https://docs.microsoft.com/en-us/graph/auth-v2-service?view=graph-rest-1.0.

Missing Documentation or Documentation Questions

Resources:

Knowledge-Based Article with links to documentation

Enhancement Requests - Sign up and use this site to discuss enhancement requests with the Product team and vote on other enhancement requests within the Syniti community. I.e. New Scanner, enhancements to existing scanners.

Support Site - Report a bug. Request changes to documentation.