Quick guide for IntegrationHub ETL

By Jens Damhøj Andersen, for Adeno K/S

Introduction:

ServiceNow IntegrationHub ETL (Extract, Transform, Load) is a feature designed to simplify and automate the process of importing and integrating data from various external sources into the ServiceNow platform, particularly into the CMDB, though non-CMDB, foundational tables are also supported.

- **Data Integration:** Facilitates the seamless import of data from external sources into ServiceNow.
- **Data Transformation:** Allows data to be transformed and normalized according to ServiceNow's data model before being loaded into the CMDB or other applications.
- **Automation:** Automate the ETL process, reducing manual intervention and ensuring consistent data updates.

Requires Paid License: No. The application is free for use and will not use any transactions or have a volume limit (IH ETL is not part of Integration Hub packages)

Additional Considerations: Knowledge of ServiceNow and target datasource fields is required to map data effectively.

IntegrationHub ETL is normally handled by the Configuration Management Team in order to keep the CMDB healthy.

IntegrationHub ETL works through a guided setup and requires little to non coding skills, however a thorough knowledge of CMDB structure and how a CMDB is govern is a prerequisite.

Adeno Learning

Table of Contents

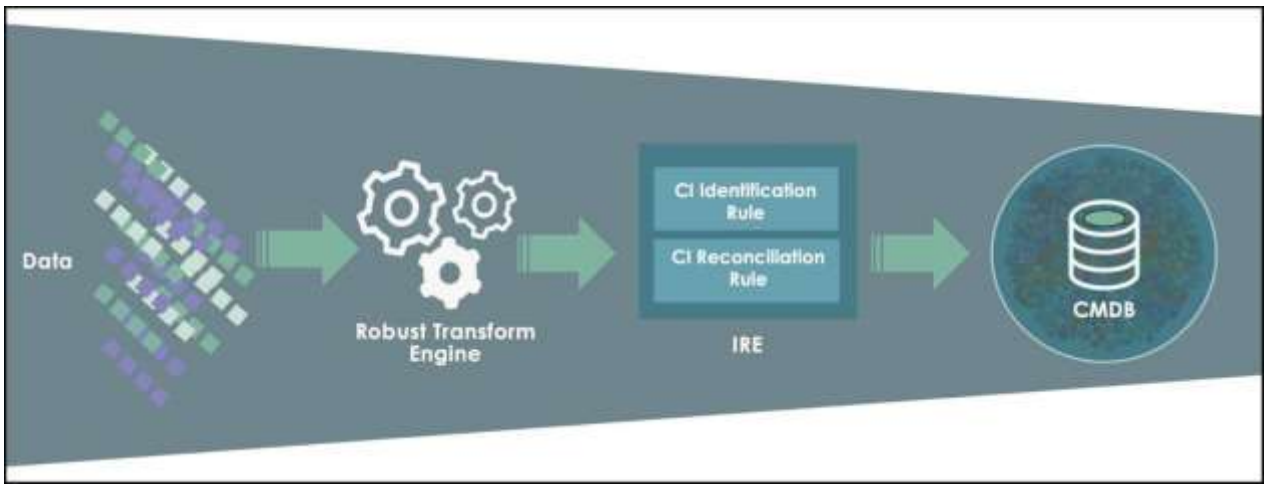
Process:

The two key components that IntegrationHub ETL (IH ETL) uses for processing are:

- Robust Transform Engine (RTE): Used to transform raw source data that is stored in staging tables, into the data that is mapped and integrated into the CMDB. RTE uses ETL transform maps that were created for the integration during data transformation.
- Identification and Reconciliation engine (IRE): Used as a centralized framework for identification and reconciliation processes across different data sources. IRE processes help maintain data integrity in the CMDB and in supported non-CMDB tables.

IntegrationHub ETL uses RTE and IRE which work together to process and integrate data. Data is first imported from a data source and is then stored in temporary staging tables in Import Sets systems. Using the data in the staging tables and the ETL transform map created by IntegrationHub ETL, RTE creates IRE payloads which are then processed by IRE. IRE applies reconciliation processes to avoid potential problems such as duplicate CIs, ensuring that the CMDB or non-CMDB tables remain healthy, and then integrates the resulting data.

When you create an integration, you import source data, transform data if needed, and select target CMDB classes (or non-CMDB tables) and attributes to map the data to. Eventually, you run an integration test of the sample data, using your settings in the IntegrationHub ETL. You can then preview the integration test results and adjust any settings before scheduling recurring integration runs for large data sets. If you develop and test the ETL transform map on a development instance, then you can test and adjust the configuration before implementation on a production instance.



IntegrationHub ETL and Import Sets

Using IntegrationHub ETL and ETL transform maps has the following advantages over using Import Sets and transform maps:

- Identification and Reconciliation Engine (IRE) processes are incorporated into the IntegrationHub ETL, so all data is automatically processed by IRE as part of the integration. Using Import Sets and transform maps does not provide a simple way to apply IRE processes.
- IntegrationHub ETL uses guided setup which provides guidance and a simple user interface for the entire process of integrating third-party data.
- IntegrationHub ETL includes an integration test for a small data set using the new ETL transform map. This test lets you review the results and adjust configuration settings before scheduling recurring integrations.

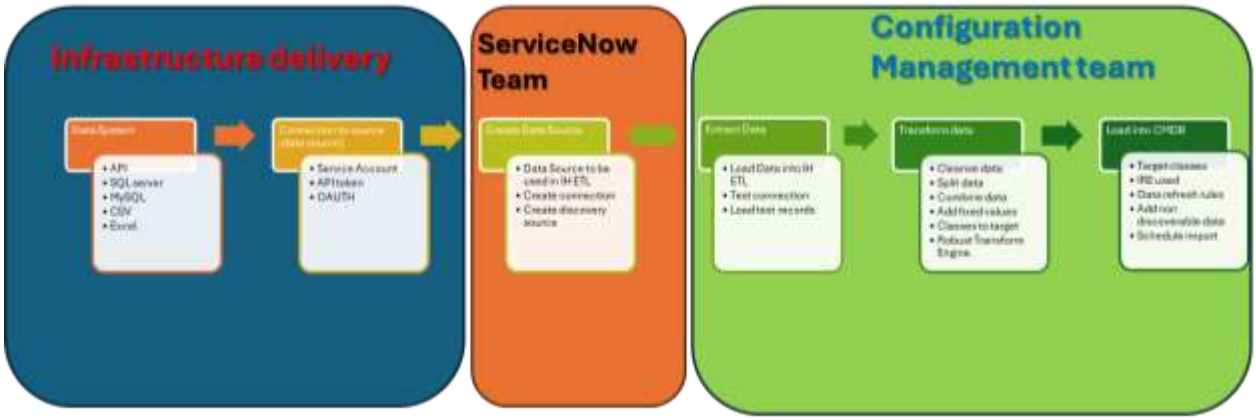
Roles:

Users with the cmdb_inst_admin role can use IntegrationHub ETL to create integrations, or customize a pre-existing integration provided by ServiceNow or a vendor at the ServiceNow Store. A vendor can create a new integration and provide it as an application for anyone to use.

Create a new integration:

When you create a new integration there is a need for assistance from the ServiceNow team in order to create the data source and create a Discovery Source value if needed.

The following graphics could be used as a guideline for working with IntegrationHub ETL:



Step 1: Access to source

The data owner gives access to the source, so a data source in ServiceNow can be built.

Step 2: Create Data Source in ServiceNow

The ServiceNow team creates the data source and if necessary, creates a new discovery source (choice)

ServiceNow team test connection

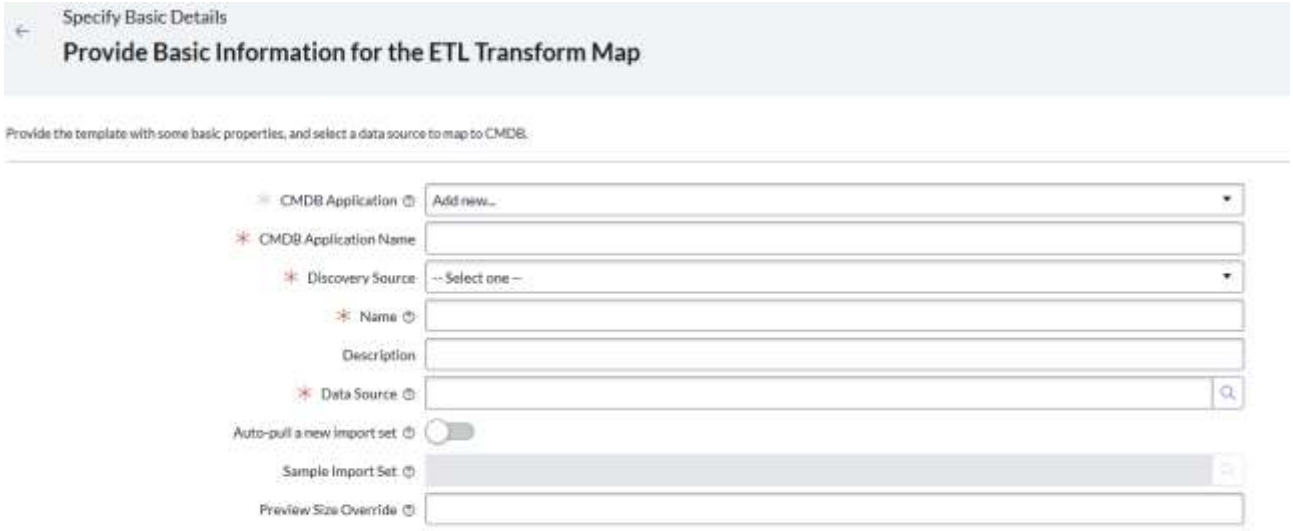
Step 3: Create integration in IntegrationHub ETL

The configuration management team creates a new integration using the guided setup.

In IH ETL click on “Create new”

Click on “import Source Data and Provide Basic Details”

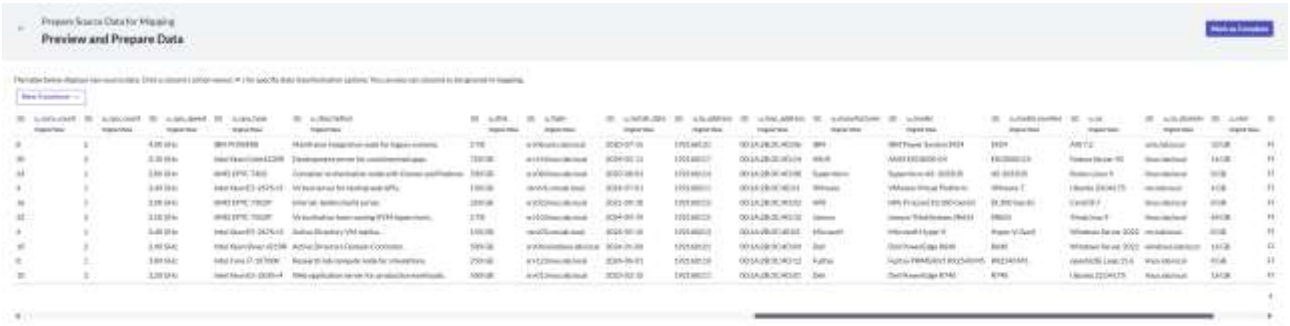
If data source is a file, you can disregard step 2 (but you might wish for a specific discovery source name which ServiceNow admin can create) and build your own data source directly from IH ETL



1. Name the CMDB application (associate name with discovery source)
2. Select discovery source (possible it was created especially for this integration)
3. Name the integration
4. Add a description for the integration (be specific)

5. Select the created data source
6. Click “save” and “Mark as complete”

The preview and prepare data:



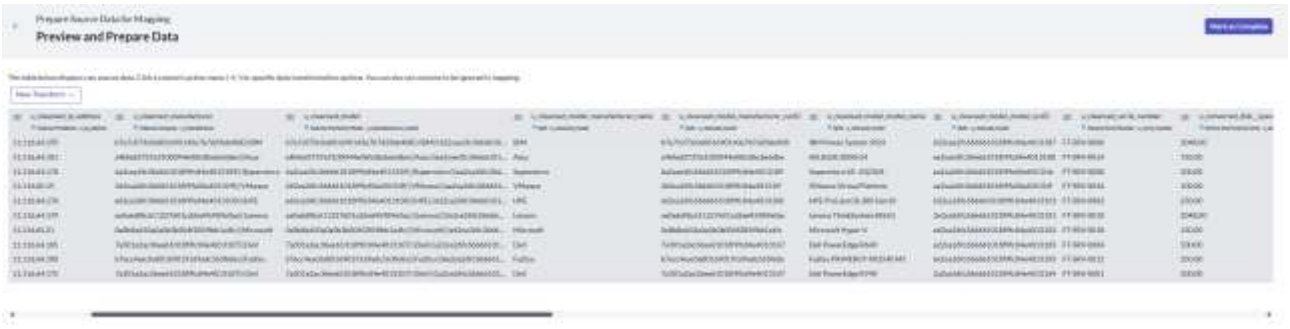
In this Step you can transform data, but it can also be done during mapping.

Go through the data and decide what transforms are needed (think twice, build once)

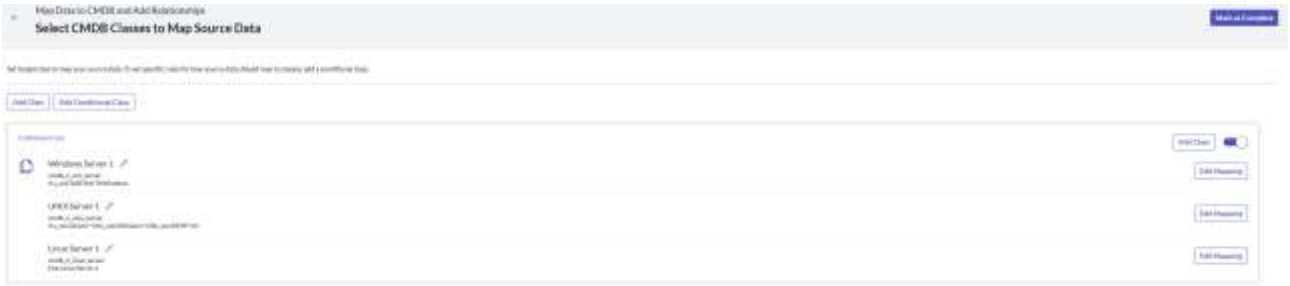
Step 4: Build the transformations

From the transformation planned data can be transformed using OOB transformations.

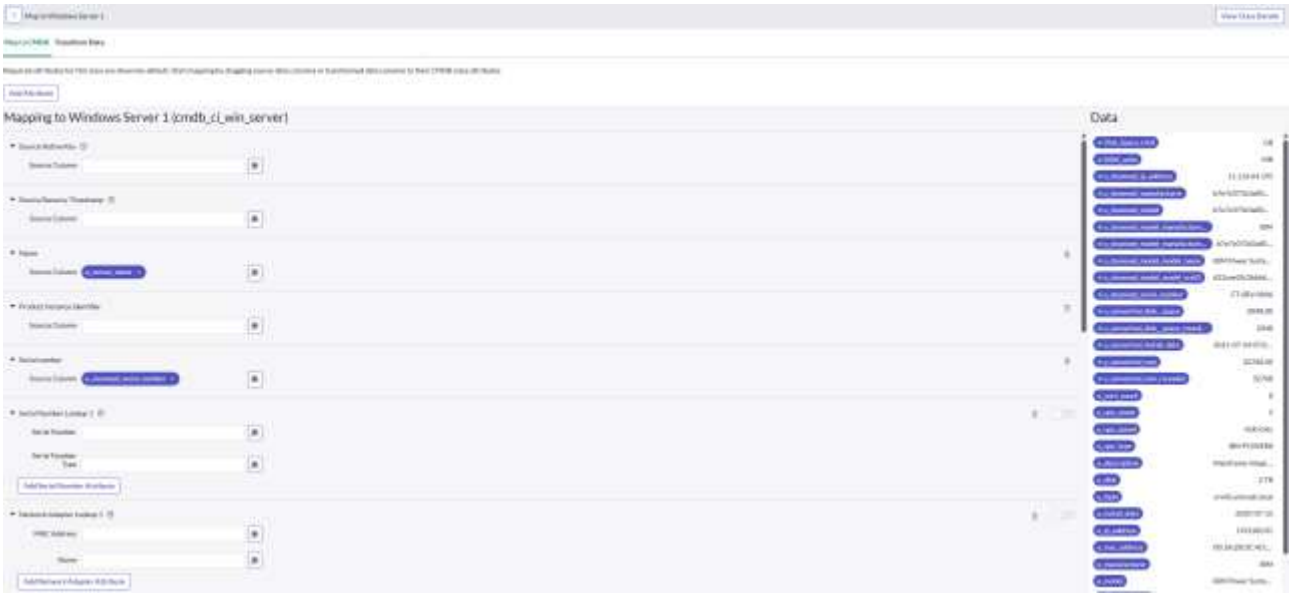
The “Cleanse” transformations will look for an existing record in the relevant class/table and create the record if a match is not found.



Step 5: Map the data to correct class



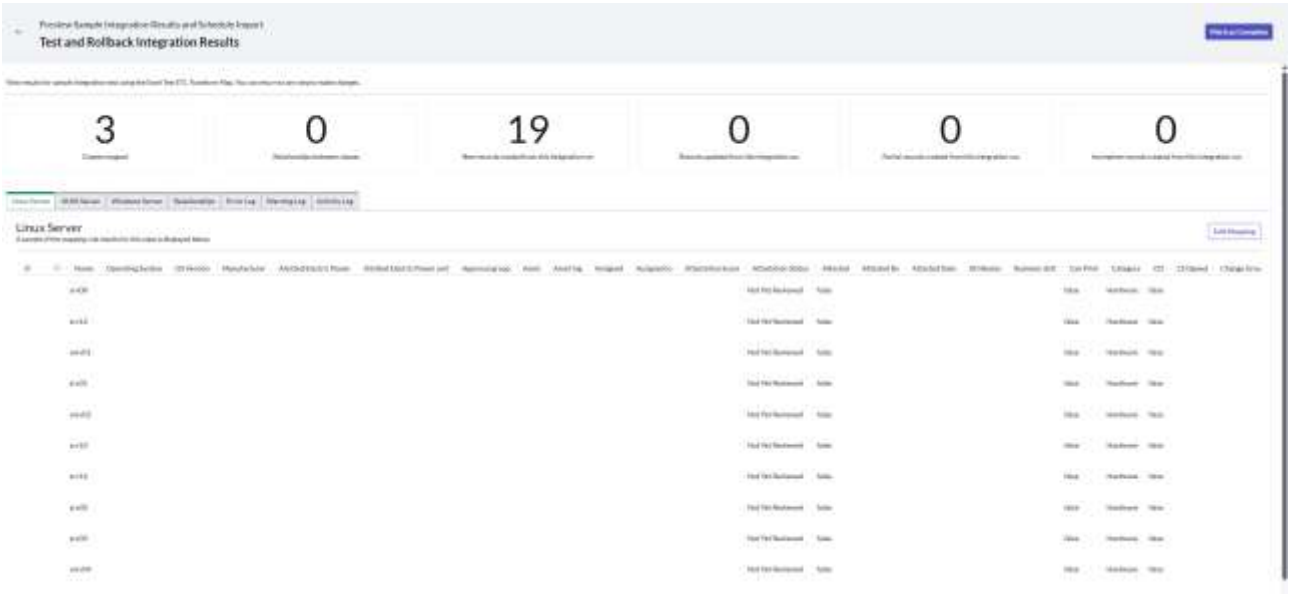
Go into each class and map relevant attributes according to data mapping plan agreed with data owner.



Transformations can also be done at this stage

Step 6: Test and review mappings

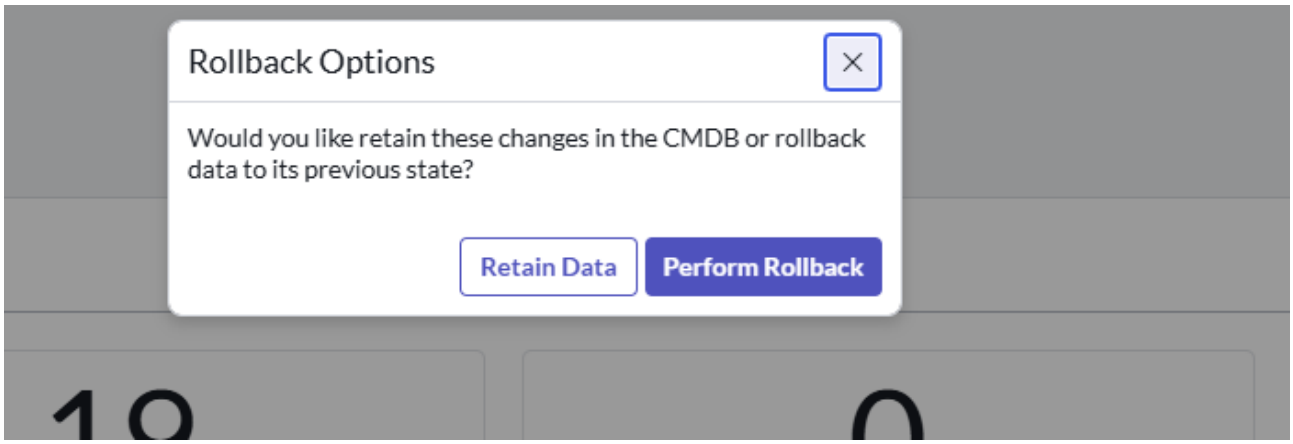
When the integration is run it will show the results:



Go through data for completeness and correct data (is everything present and is the data showing correct).

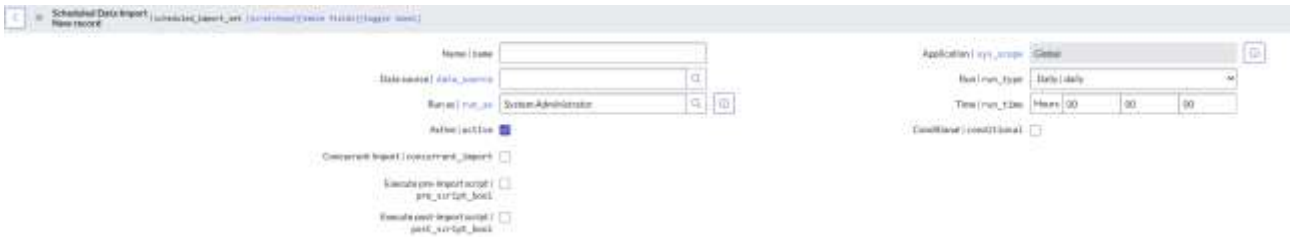
If some attributes are not filled it is due to attribute characteristics such as a reference field or concatenated field (combination of different attributes such as “Display Name”). Go through the transformation for that attribute to correct the mapping.

When finished with the review click “Mark as Complete” It will give you a choice of retaining data or roll back.
If you need to adjust the transformations, select roll back otherwise select retain data.



Step 7: Set schedule for data source

The last step is to schedule the data import and processing.



Use concurrent import with large amount of data so ServiceNow will break the data load into smaller parts for better performance.

OOB transformations:

The documentation for the different OOB transformations is scattered a bit in the manual, so find the transformations here:

- https://www.servicenow.com/docs/bundle/yokohama-servicenow-platform/page/product/configuration-management/concept/create-etl-transform-map.html

- https://www.servicenow.com/docs/bundle/yokohama-servicenow-platform/page/product/configuration-management/reference/cmdb-rte-operation-types.html

Appendix A: Transform examples

Model ID:

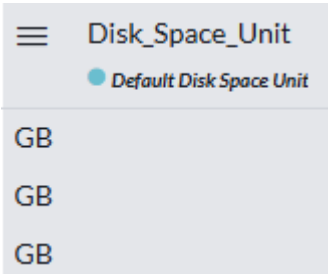
The model ID field is a display value and the model ID SYS_ID is needed to populate correctly.

1. First cleanse the model using “Cleanse Hardware model”
   The transformation will create 4 items:
   a. Sys_ID for the model manufacturer
   b. Name for manufacturer
   c. Sys_ID for model
   d. Name for model
   You will need the sys_ID for model for mapping so a split is needed
2. Split the cleansed model
   That will give you 4 separate columns (Manufacturer name, Manufacturer SYS_ID, Model name and Model SYS_ID). Name tables for easy use.
3. Map model ID using the model Sys_ID

Disk Space in GB with data in different units:

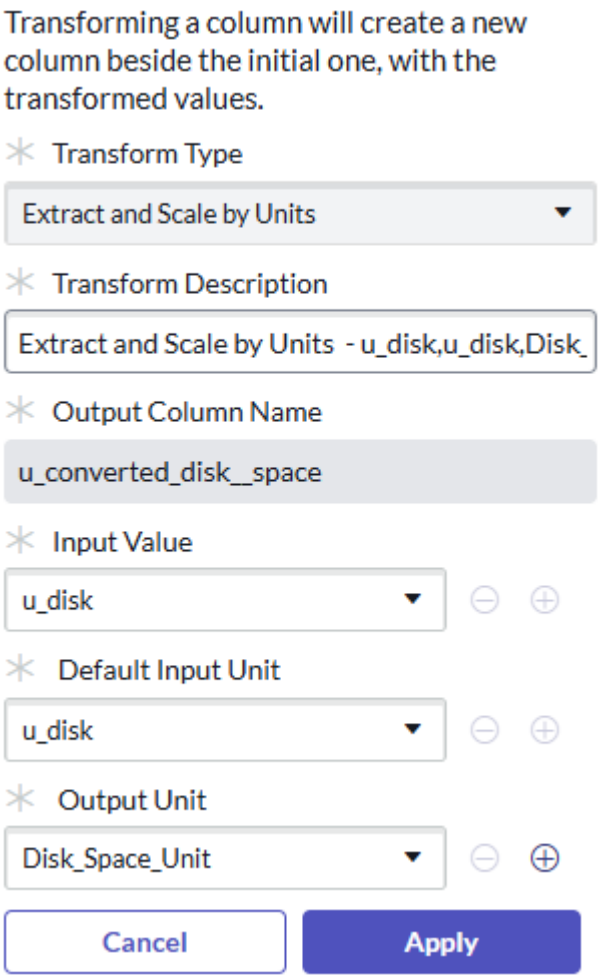
Disk Space is a field that standard is in GB, but different sources can report in other units.
So the process is to extract and convert to standard units. IH ETL can read units from the source, but if no units are given you can set the unit manually.

1. Create a “Set Fixed value Column”
2. Set column value to GB



3. If input units are not present then create the input unit (if values are present with the value IH ETL should be able to detect input unit)

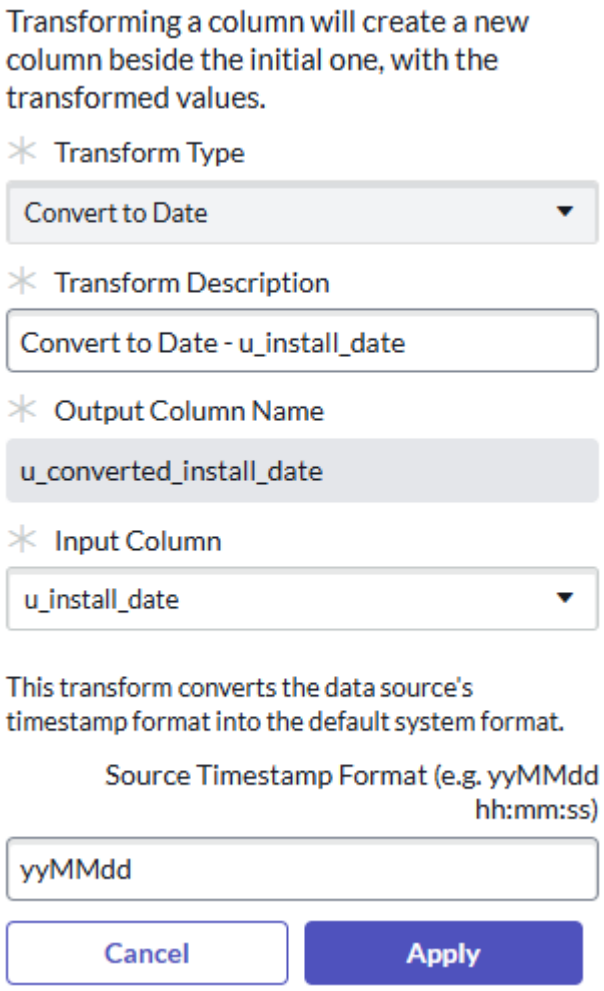
4. Use the “Extract and Scale by Units” transform



5. For better look round the conversion

Date formats:

Date formats can be converted to ServiceNow standard using “Convert to Date”



Software (cmdb_ci_spkg)

If you extract software from your source there is two tables that must be populated:

1. Software (cmdb_ci_spkg)
2. Software instance (cmdb_software_instance)

The “Software Instance” will populate the related list “Software Installed” if you have SAM installed you might need to show the correct related list.

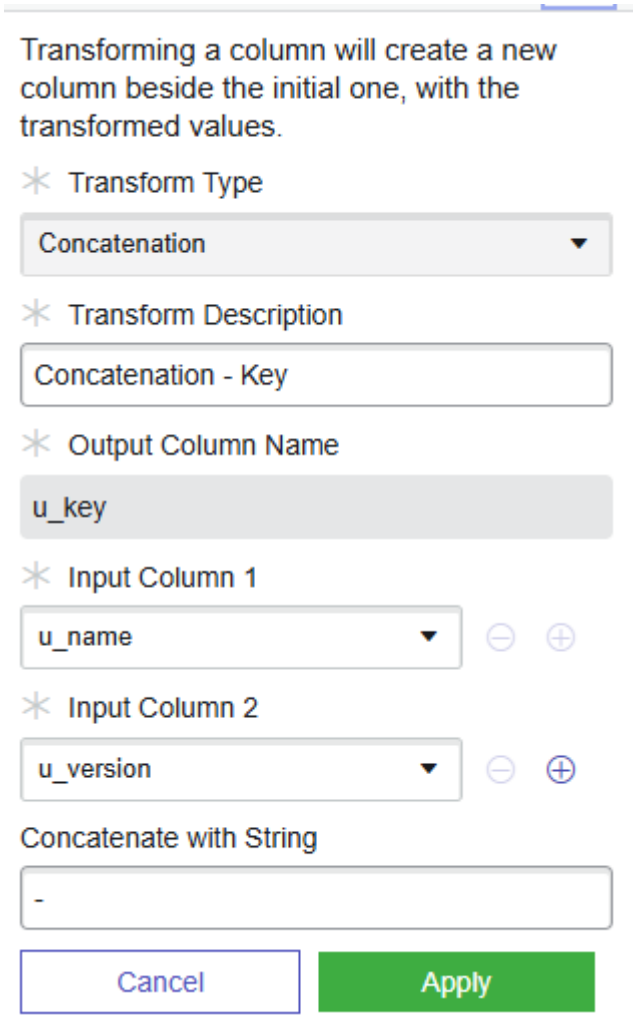
The main attribute to populate on software is the key. That is necessary for IH ETL. You need to add:

1. Name
2. Version
3. Manufacturer

Step 1:

Generating a unique key.

It is advised to use a key consisting of name and version to uniquely identify the correct software, for that use Concatenation transform.



Step 2:

Cleanse manufacturer (if not already done in other transform)

Use Cleanse Company transformation so to make sure the company (manufacturer) is in ServiceNow.

Transforming a column will create a new column beside the initial one, with the transformed values.

Transform Type

Cleanse Company

Transform Description

Cleanse Company - {1}

Output Column Name

Hide initial column used for this transform

Company Name

-- Select one --

Cancel Apply

Step 3:

Map the four attributes:

1. Key (concatenation)
2. Name
3. Version
4. Manufacturer (cleansed company)

Software Instance (cmdb_software_instance)

Software Instance is not a basic CMDB class and therefore must be associated with the software transformation (the above example).

Step 1:

Add a class: Software Instance

Add “Associated Class” Software

Transformations:

Depending on your data you can cleanse Software model, which will give you:

Transforming a column will create a new column beside the initial one, with the transformed values.

Transform Type

Cleanse Software Model

Transform Description

Cleanse Software Model - u_name,u_name,u_

Output Column Name

u_cleansed_software

Company Name

u_name

Software Name

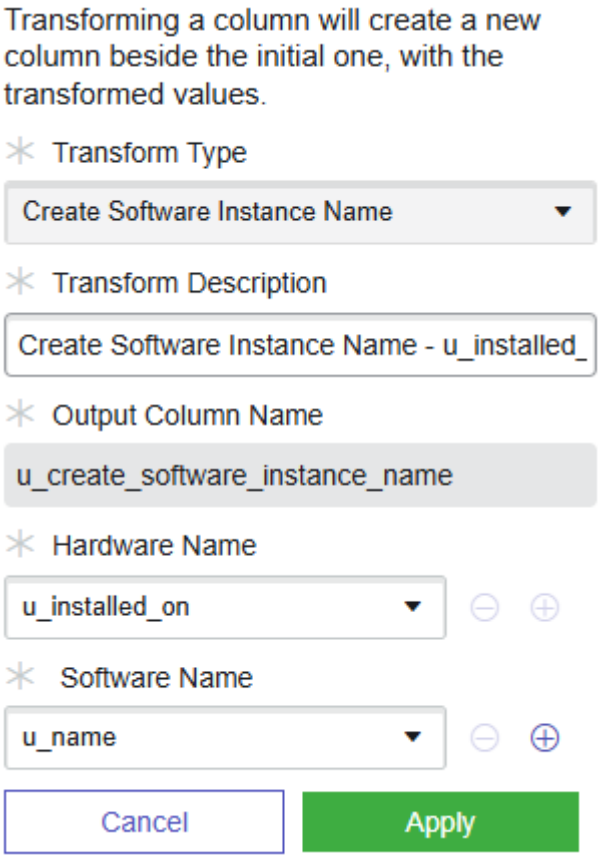
u_name

Software Version

u_version

Cancel Apply

Or you can create a Software Instance name:



Mapping Software Instance:

There are some choices you can leverage depending on quality of data and need for control.

This example uses the most secure way (using transformed values).

Mapping:

Name = Software Instance name

Installed on (transform coloumn) = “hardware/server/computer” SYS_ID is to be used

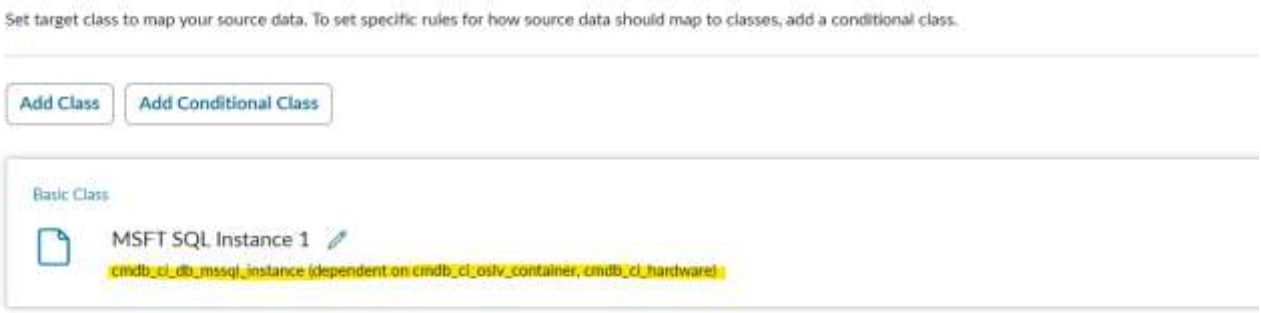
Product name (transformed coloumn) = cleansed software name

SQL Instance running on a server

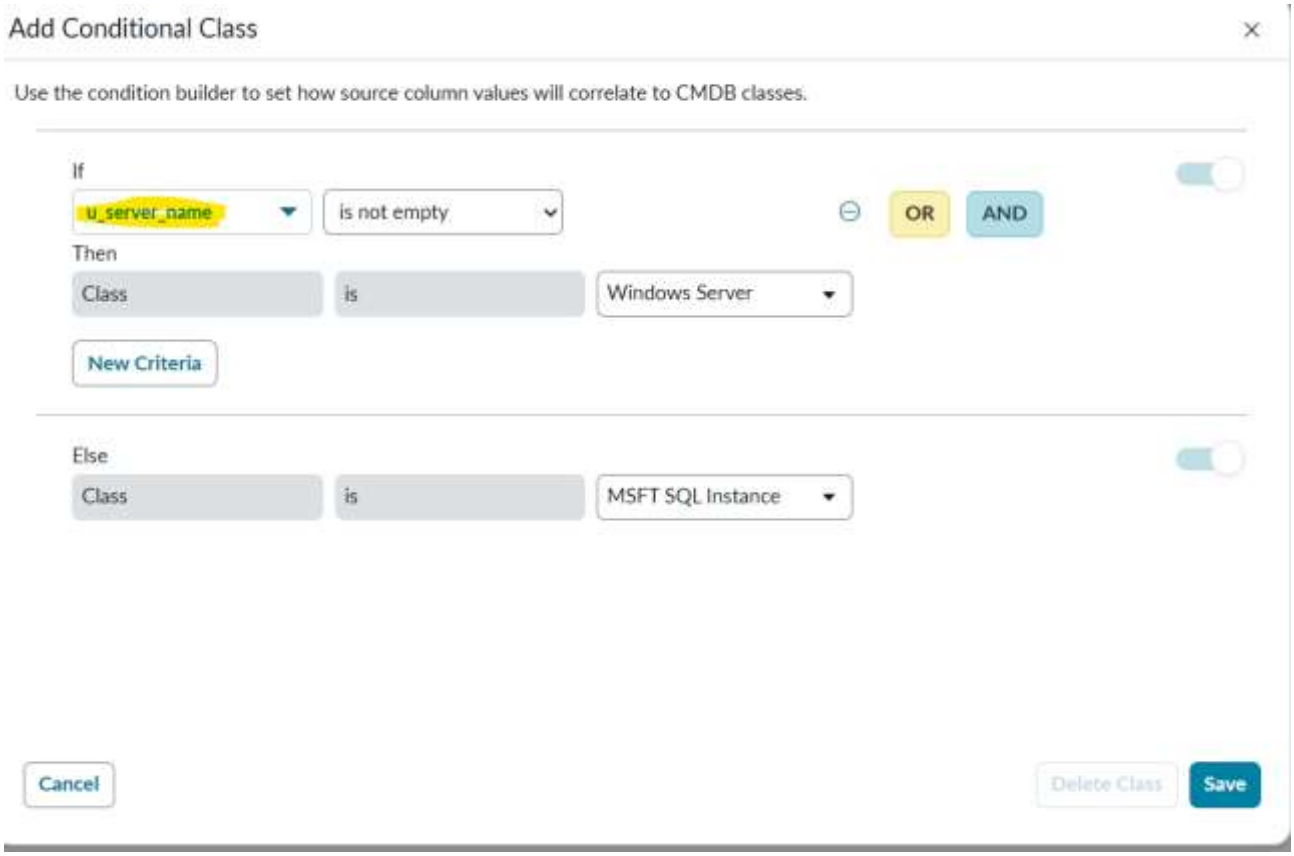
A SQL Instance runs on a computer (typically Windows Server) and need a CI Relationship to be shown correctly in CMDB.

In this case you create a record in the windows server class and a SQL Instance in the CMDB and add a relationship between the classes.

IH ETL knows there is a dependency to another class (under Hardware or Container branch) so there need to be created a conditional class to make that dependency.
A subtle indication is shown in IH ETL:



After the creation of the Basic Class for SQL Instance create a conditional class:



The “IF” is pointing towards the class for the host (in this case a windows server) but could be pointing to any class under hardware or container in the CMDB.

After the creation of conditional class the class mapping looks like this:

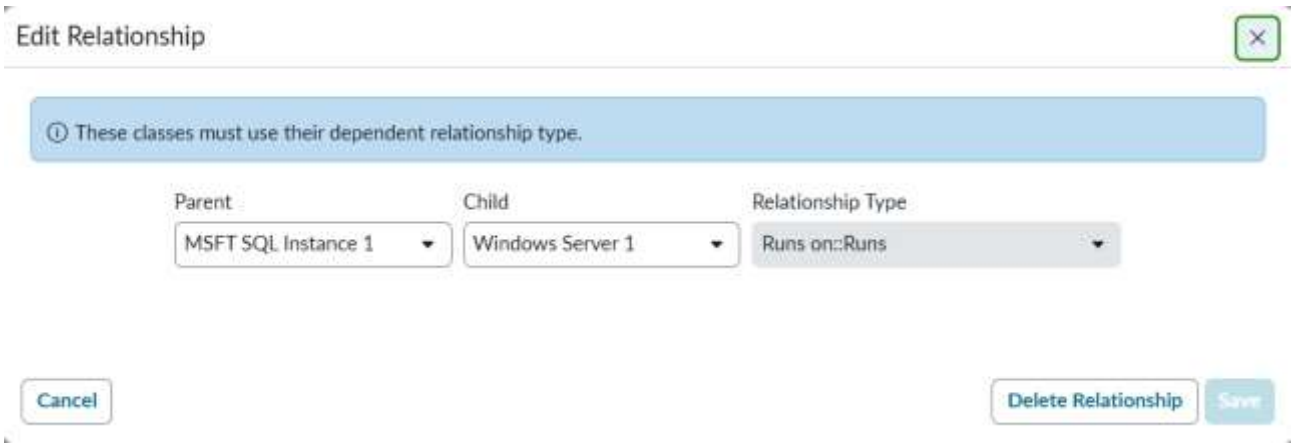
You are now ready to map the different classes.
Mapping of windows server:

In this example the server_pi (Server Product Identifier) is a concatenation of server name and serial number to uniquely identify each record.

The purpose of this mapping is to identify the host that the SQL Instance is running on, and windows server data should be present in the CMDB from other sources.

It is not needed to map “SQL Instance 2”

Add the relationship between Windows Server and SQL Instance:



Be aware to select the proper parent/child class (hint: when the correct parent and child is selected the suggested relationship will automatically be populated).

Run the integration and (with this example data set) it should look like this:

Adeno Learning

MSFT SQL Instance | Windows Server | Relationships | Error Log | Warning Log | Activity Log

MSFT SQL Instance

Edit Mapping

Name	Instance Name	Version name	Version	Service pack	Edition	TCP port (s)	Approval group	Asset	Asset tag	Assigned	Assigned to	Attestation Score	Attestation Status	Attested	Attested By	Attested Date	Attributes
SQL13			Microsoft SQL Server 2016		Standard	1433							Not Yet Reviewed	false			
SQL08			Microsoft SQL Server 2017		Enterprise	1433							Not Yet Reviewed	false			
SQL07			Microsoft SQL Server 2019		Standard	1433							Not Yet Reviewed	false			
SQL15			Microsoft SQL Server 2019		Standard	1433							Not Yet Reviewed	false			
SQL03			Microsoft SQL Server 2019		Enterprise	1433							Not Yet Reviewed	false			
SQL09			Microsoft SQL Server 2017		Standard	1433							Not Yet Reviewed	false			

MSFT SQL Instance | Windows Server | Relationships | Error Log | Warning Log | Activity Log

Windows Server

Edit Mapping

Name	Operating System	OS Version	Allotted Electric Power	Allotted Electric Power unit	Approval group	Asset	Asset tag	Assigned	Assigned to	Attestation Score	Attestation Status	Attested	Attested By	Attested Date	Attributes	Duplic
WIN-SQL06						Unknown					Not Yet Reviewed	false				
WIN-SQL12						Unknown					Not Yet Reviewed	false				
WIN-SQL04						Unknown					Not Yet Reviewed	false				
WIN-SQL09						Unknown					Not Yet Reviewed	false				
WIN-SQL13						Unknown					Not Yet Reviewed	false				
WIN-SQL01						Unknown					Not Yet Reviewed	false				

MSFT SQL Instance | Windows Server | Relationships | Error Log | Warning Log | Activity Log

Relationships

Edit Relationships

Parent	Child	Relationship Type
SQL03	WIN-SQL03	Runs on Host
SQL07	WIN-SQL07	Runs on Host
SQL13	WIN-SQL13	Runs on Host
SQL15	WIN-SQL15	Runs on Host
SQL12	WIN-SQL12	Runs on Host