USER GUIDE

Matching, Linking, and Merging

Release 10.1-MP4 (April 20, 2021)

Table of Contents

Matching, Linking, and Merging

The STEP Matching, Linking, and Merging component offers powerful functionality for identifying and handling duplicate product, entity, asset, and classification objects in STEP.

The matching, linking, and merging functionality is most commonly used for:

- Cleanup operations, such as during data migration
- Matching of the same product from multiple suppliers
- Matching of the same customer from different source systems
- Consolidation of information from different systems
- Cleansing data after migrating records from various sources

Getting Started

Before configuring the matching, linking, and merging functionality, users must define what qualifies two or more objects as duplicates and what the system should do when it encounters such duplicates.

- The **match criteria** defines what qualifies objects as duplicates
- The **match action** determines what the system should do with such duplicates

Both the match criteria and the match action are included as part of a matching algorithm.

Choice of Match Action

When setting up a matching solution, users must choose a match action. The match action defines the workflow and the data model around the objects you are matching.

Users can configure the system to only identify duplicates using the Identify Duplicates match action, or to also take action on those matches. The system supports different action strategies like merging records or generating new link golden records. For more information, see the **Identify Duplicates Match Action** topic of this documentation.

Match and Link

Match and Link creates and maintains a set of Golden Records as an aggregation of matching Source Records through an asynchronous process.

- In Product MDM, Match and Link automates the creation and maintenance of Sell-Side Products as Golden Records, based on Buy-Side Products as Source Records.
- In Customer MDM, Match and Link resolves Household Entities as Golden Records from Individual Customer Entities as Source Records.

Match and Link uses an event processor to create and update new Link Golden Records that captures the best information from each of the Source Records. The system identifies the new Link Golden Record object with a

STEP identifier and links this record to all source records contributing to it. Over time, new information may clarify that some source records that were linked together are no longer valid for linking to a specific Golden Record. The algorithm will then link these Source Records to different Link Golden Records. As a result of this automatic linking and splitting, the STEP identifier of the Link Golden Record linked to a given Source Record may change over time.

Users should **never** edit a Link Golden Record object directly. To edit a Link Golden Record object, users should add the information on a special type of source record, called a 'Silver Record,' and the information is then merged into the Link Golden Record by the matching algorithm. The promotion of information from the Silver Record to the Link Golden Record happens asynchronously through the Matching Event Processor. For more information, see the **Match and Link** topic in this documentation.

Match and Merge

The Match and Merge solution uses criteria to match entity records and merge these incoming records into Golden Records.

In Customer MDM and Supplier MDM, Match and Merge is used to consolidate, enrich, and synchronize duplicate records in surrounding systems.

Note: The Match and Merge solution only works for Entities.

Match and Merge works by combining a special importer and an event processor. When the Match and Merge Importer imports a new entity, the importer uses a matching algorithm to compare the incoming entity against an existing Golden Record. If a matching entity already exists, the system promotes the information from the incoming entity to that existing Golden Record through Survivorship Rules. As Golden Records are updated, a Matching Event Processor identifies matching Golden Records and merges information from one of the records into the other and deactivates the non-survivor. For more information, see **Match and Merge** topic in this documentation.

Match Tuning

Defining a match criteria that accurately identifies matching records is an iterative process that requires a thorough understanding of the data and collaboration between data owners and the super users defining the match criteria.

During the implementation process, you will tune the match criteria to match the correct records. You may also need to optimize your match criteria to achieve your performance goals.

For more information about the tools available and the recommended process, see the **Tuning and Monitoring a Matching Algorithm** topic in this documentation.

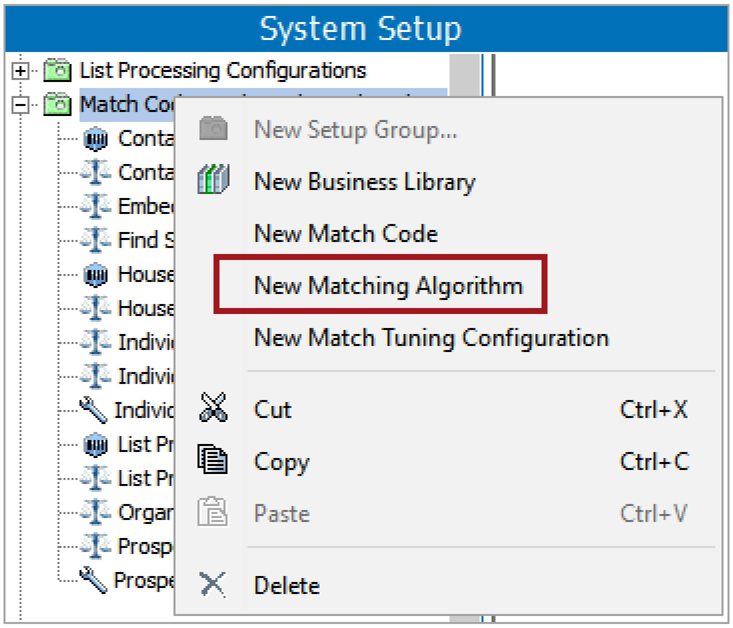
Configuring Matching Algorithms Setup Group

Matching Algorithms and External Match Codes are first-class objects in **System Setup**, living below 'Setup Groups' in the workbench. To create new Matching Algorithms and External Match, a setup group must exist to house them.

On a system that is not set up for matching, define a new setup group object type, create the needed Matching Algorithms and External Match child objects, and then create an instance of the setup group object type in System Setup. For more information, see the **Setup Group** topic of the **System Setup / Super User Guide** documentation.

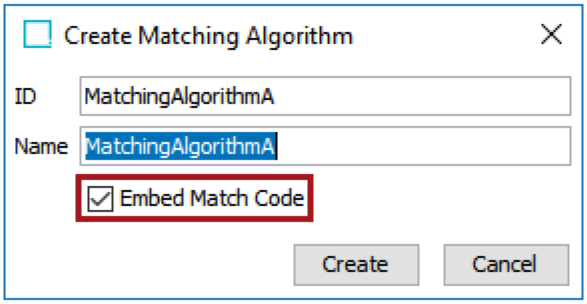
Configuring Matching Algorithms

1. In System Setup, right-click the node configured to house matching algorithms and select **New Matching Algorithm**.



2. In the Create Matching Algorithm dialog, define an **ID** and **Name** for the matching algorithm.

- Check the **Embed Match Codes** checkbox so the match codes are embedded in the algorithms.
- If the **Embed Match Codes** checkbox is not checked (legacy functionality), you must manually create a match code and link it to the matching algorithm. For more information, see the **Match Codes** topic in this documentation.

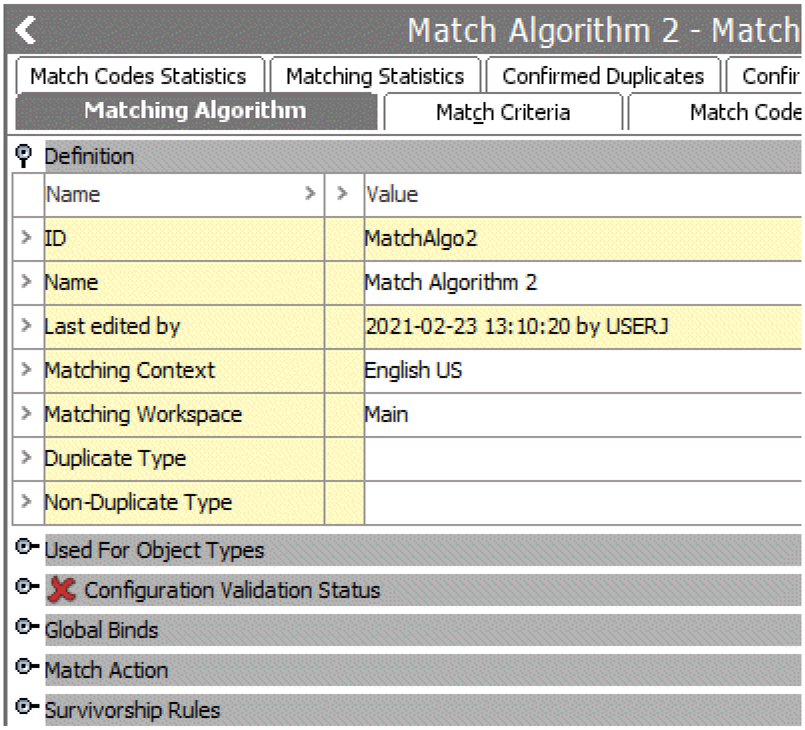


ID MatchingAlgorithmA

Name MatchingAlgorithmA

☑ Embed Match Code

- Click **Create** to display the Matching Algorithm object:



Match Algorithm 2 - Match

Definition

Name	Value
ID	MatchAlgo2
Name	Match Algorithm 2
Last edited by	2021-02-23 13:10:20 by USERJ
Matching Context	English US
Matching Workspace	Main
Duplicate Type	
Non-Duplicate Type	

Used For Object Types

Configuration Validation Status

Global Binds

Match Action

Survivorship Rules

Note: The Configuration Validation Status flipper displays a red 'X' (shown above) when the configuration is invalid. Open the flipper to view the errors that must be addressed. Correct any errors shown before running the matching algorithm. When the matching algorithm has a valid configuration, a green checkmark is displayed.

3. In the Definition flipper, for the **Matching Context** parameter, specify the context to run the matching algorithm. By default, the current context is set.

4. For the **Matching Workspace** parameter, specify the workspace to run the matching algorithm. By default, the Main workspace is selected.

5. For the **Duplicate Type** parameter, click the ellipsis button (...). In the 'Select a Duplicate Reference Type' dialog, select the appropriate reference type as defined in the component model. For more information, see the **Match Criteria Configuration** topic.

6. For the **Non-Duplicate Type** parameter, click the ellipsis button (...). In the 'Select a Duplicate Reference Type' dialog, select the appropriate reference type as defined in the component model.

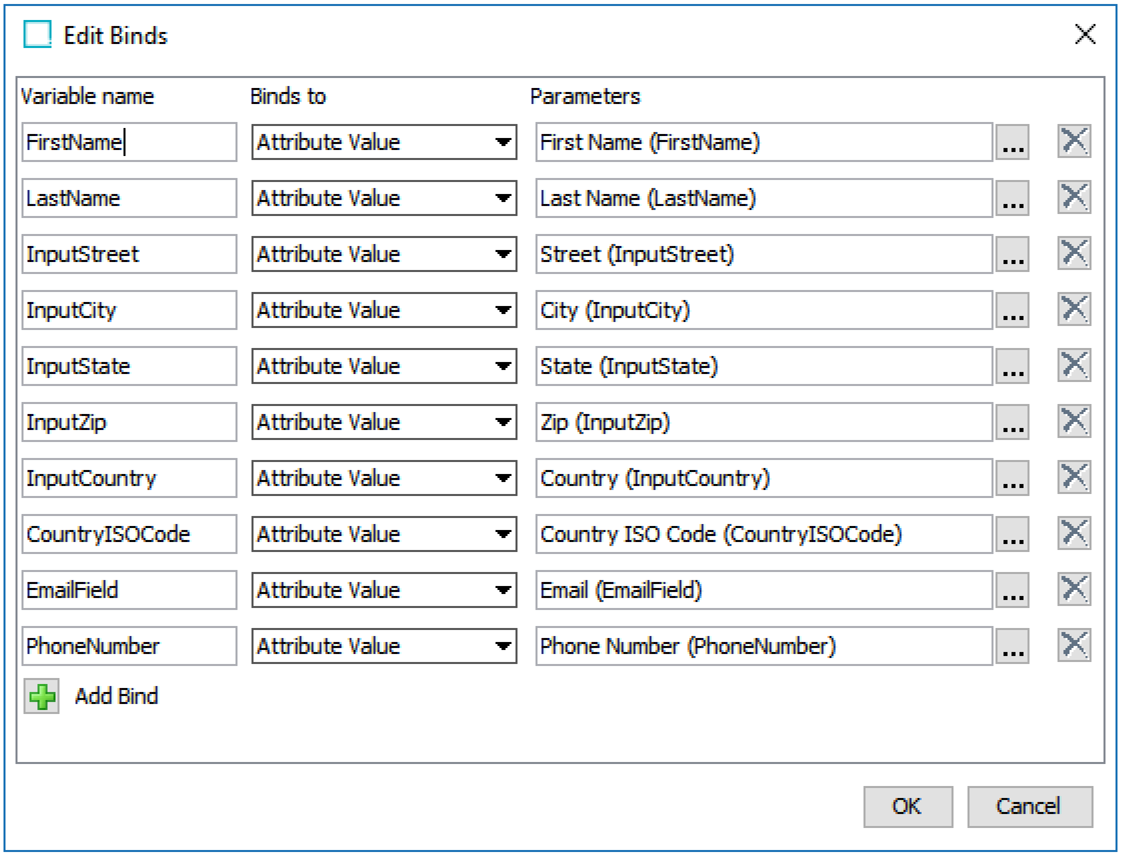
   For more information, see the **Match Criteria Configuration** topic.

7. In the Global Binds flipper, potentially improve the performance by creating global binds to obtain all attribute values used in the decision table comparison.

   The matching process can strain performance. When processing large sets of data, there is potentially a significant performance gain if the matching functionality can fetch the values for matching before the matching process begins. This fetching of data is possible via global binds configured on the matching algorithm, where the matching algorithm logic uses attributes that are bound to specific variable names. The system fetches the values for the attributes before the match criteria logic is applied and can be referenced from both JavaScript and STEP functions.

> **Important:** Global binds are not optimized for use with In-Memory.

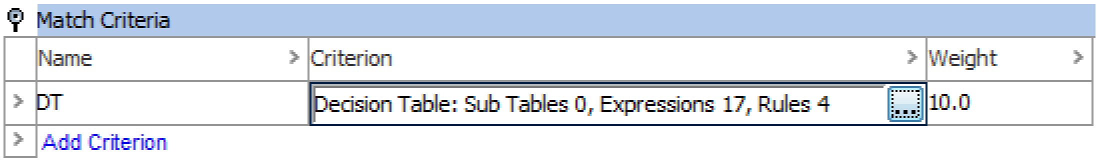
- Click the **Edit Global Binds** link to open the 'Edit Binds' dialog shown below.
- Click the **Add Bind** button to create a new bind.
- For **Variable name**, specify a variable name for the bind.
- For **Binds to**, select a bind from the dropdown (some binds are displayed within a group).
- For **Parameters**, when available, click the ellipsis button (...) to specify an object to bind.
- Click **OK** to close the dialog and return to the Matching Algorithm object.



Variable name	Binds to	Parameters
FirstName	Attribute Value	First Name (FirstName)
LastName	Attribute Value	Last Name (LastName)
InputStreet	Attribute Value	Street (InputStreet)
InputCity	Attribute Value	City (InputCity)
InputState	Attribute Value	State (InputState)
InputZip	Attribute Value	Zip (InputZip)
InputCountry	Attribute Value	Country (InputCountry)
CountryISOCode	Attribute Value	Country ISO Code (CountryISOCode)
EmailField	Attribute Value	Email (EmailField)
PhoneNumber	Attribute Value	Phone Number (PhoneNumber)

8. If the Match Criteria flipper is displayed (only for legacy algorithms where match codes are not embedded and must be created manually), configure the following. For more information, see the **Match Criteria** topic in this documentation.

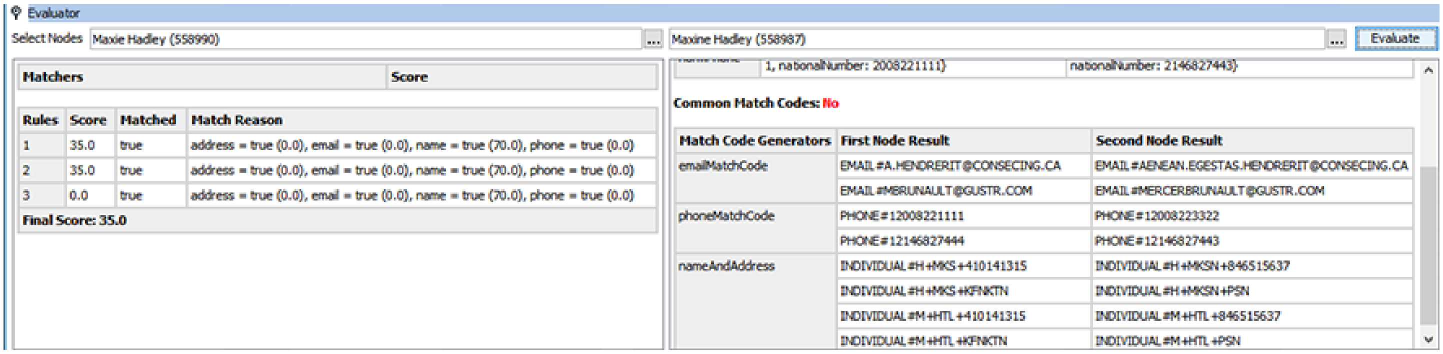
- Click the **Add Criterion** link to display the 'Select Match Criterion' dialog.
- Specify a **Name**.
- Choose a match criterion from the **Select Match Criterion** dropdown.
- Click the **Add** button.
- Click into the **Criterion** field and then click the ellipsis button (**...**) to open the editor.
- Create the matching criterion and click **OK**.
- Click into the **Weight** field and specify a weight for the criterion.



Match Criteria

Name	Criterion	Weight
DT	Decision Table: Sub Tables 0, Expressions 17, Rules 4	10.0
Add Criterion		

9. Open the **Evaluator** flipper, select two objects to test the selected criteria on a data set.



Select Nodes: Maxie Hadley (558990) ... Maxine Hadley (558987) ...

Matchers	Score

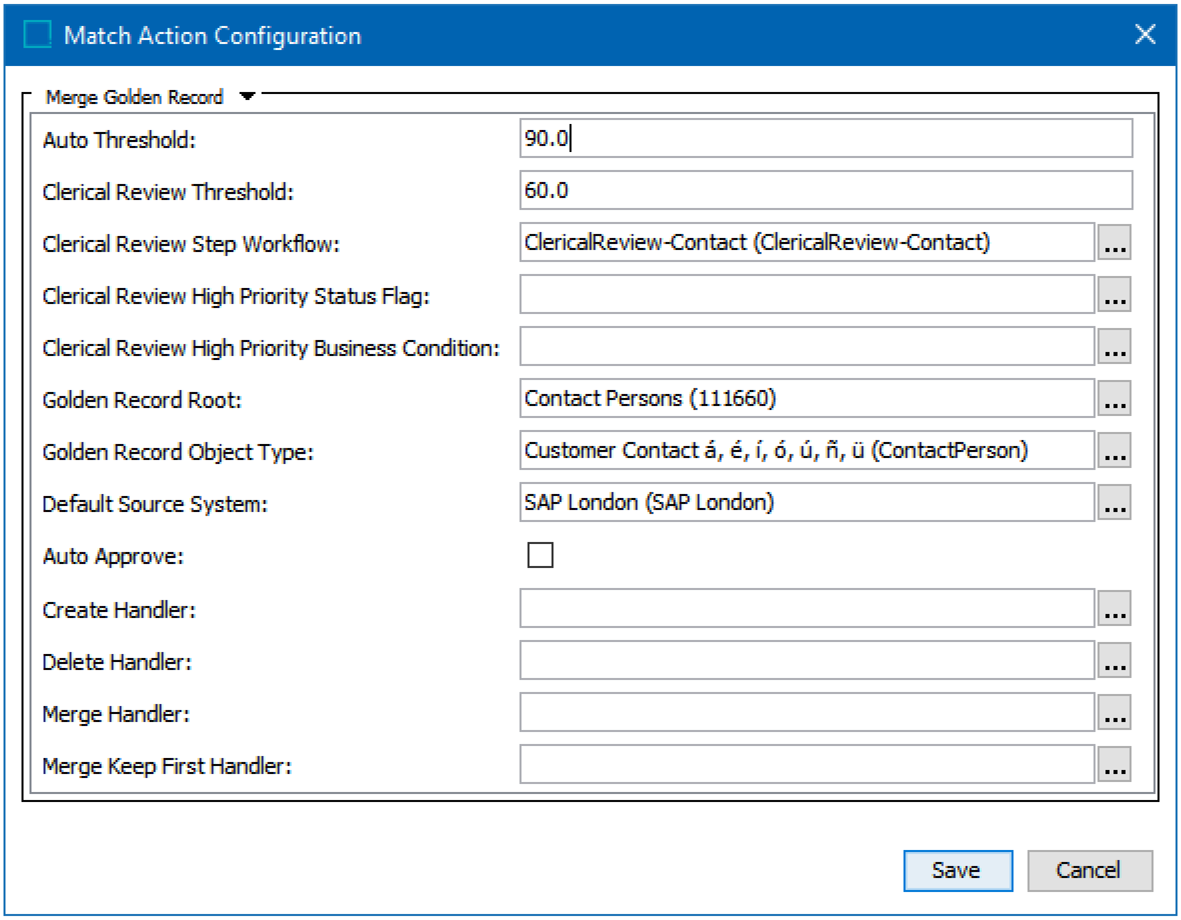
Rules	Score	Matched	Match Reason
1	35.0	true	address = true (0.0), email = true (0.0), name = true (70.0), phone = true (0.0)
2	35.0	true	address = true (0.0), email = true (0.0), name = true (70.0), phone = true (0.0)
3	0.0	true	address = true (0.0), email = true (0.0), name = true (70.0), phone = true (0.0)
Final Score: 35.0			

1, nationalNumber: 2008221111} | nationalNumber: 2146827443}

Common Match Codes: No

Match Code Generators	First Node Result	Second Node Result
emailMatchCode	EMAIL #A.HENDRERIT@CONSECING.CA	EMAIL #AENEAN.EGESTAS.HENDRERIT@CONSECING.CA
	EMAIL #MBRUNAULT@GUSTR.COM	EMAIL #MERCERBRUNAULT@GUSTR.COM
phoneMatchCode	PHONE#12008221111	PHONE#12008223322
	PHONE#12146827444	PHONE#12146827443
nameAndAddress	INDIVIDUAL #H+MKS+410141315	INDIVIDUAL #H+MKSN+846515637
	INDIVIDUAL #H+MKS+KFNKTN	INDIVIDUAL #H+MKSN+PSN
	INDIVIDUAL #M+HTL+410141315	INDIVIDUAL #M+HTL+846515637
	INDIVIDUAL #M+HTL+KFNKTN	INDIVIDUAL #M+HTL+PSN

10. Set up the match action as needed. For more information, see the **Match Actions** topic in this documentation.



Auto Threshold:	90.0
Clerical Review Threshold:	60.0
Clerical Review Step Workflow:	ClericalReview-Contact (ClericalReview-Contact)
Clerical Review High Priority Status Flag:	
Clerical Review High Priority Business Condition:	
Golden Record Root:	Contact Persons (111660)
Golden Record Object Type:	Customer Contact á, é, í, ó, ú, ñ, ü (ContactPerson)
Default Source System:	SAP London (SAP London)
Auto Approve:	□
Create Handler:	
Delete Handler:	
Merge Handler:	
Merge Keep First Handler:	

Match Criteria

The match criteria are responsible for matching records against each other to find those who match. In some cases, users may only be interested in exact matches. In these scenarios, the match criteria would be reasonably straightforward.

Suppose the social security number for two customer objects, or the EAN number for two products, are identical. In that case, these are likely duplicates, and the matching criteria should return 100%. If the social security number does not match, the match criteria should probably return 0%.

In many cases, however, you cannot work with exact matches; but instead, you will have to deal with approximate matches or a combination of exact and approximate matches. For example, you do not have a social security number available. You will have to identify duplicates based on names, mail addresses, phone numbers, and street addresses. You will need to identify the product based on the same manufacturer and manufacturer part number information for products.

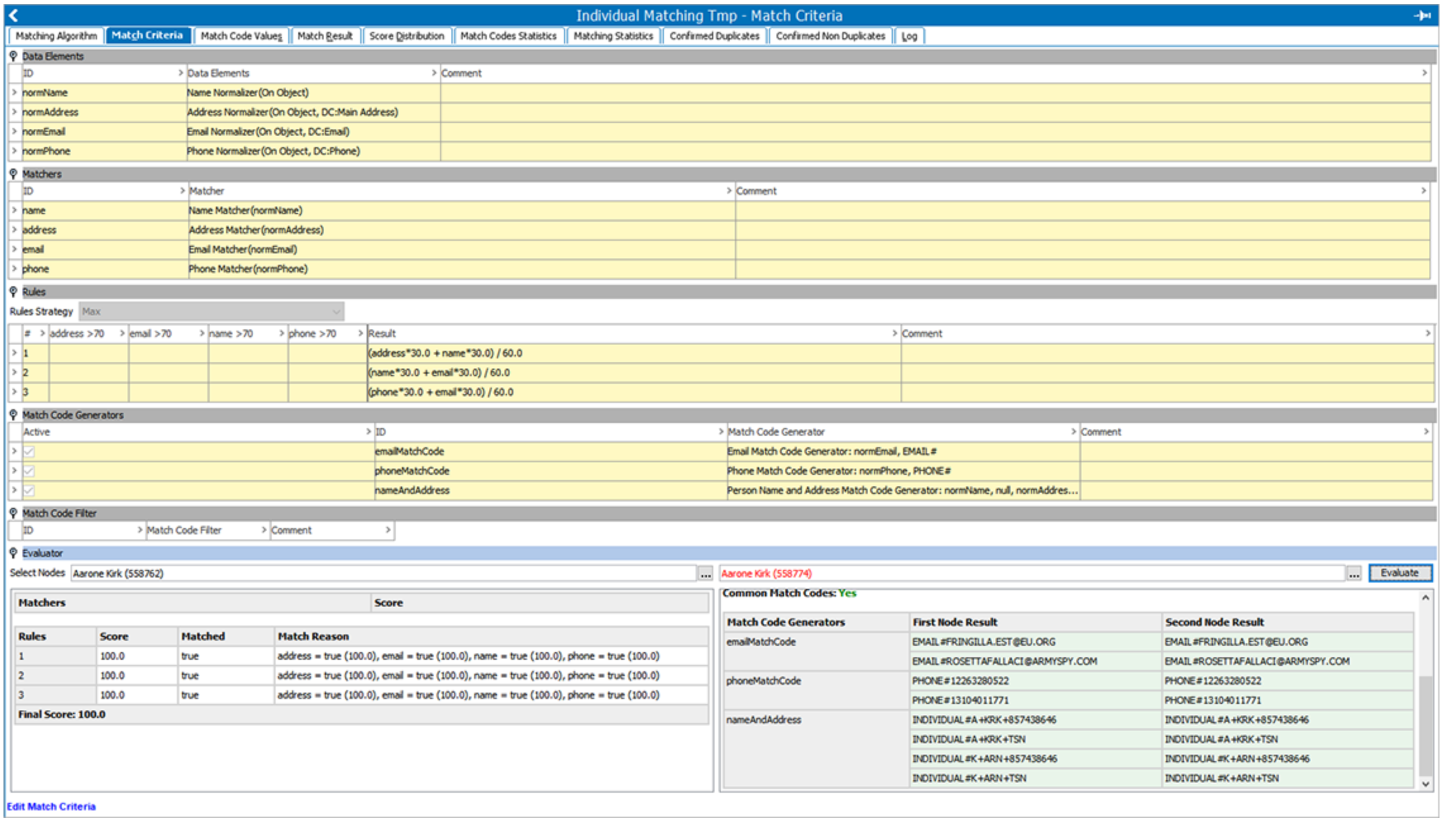
These pieces of data can have variations, even in objects that represent the same real-world entity. Names and addresses could be spelled differently, middle names could be left out, abbreviations could be used in names and addresses, the customers could be registered with different phone numbers or mail addresses, and myriad other options that introduce ambiguity to these records.

Because of this complexity, the match criteria's logic is most often represented in a decision table, which not only depends on match codes but further divides the functionality into normalizers, matchers, and rules.

Overview

The Match Criteria tab contains how to compare two objects and evaluate to what degree they are similar. It is separated into sections:

- **Data Elements**: This section contains the data elements that the matchers and match code generators evaluate. The data elements present data in a form that is easy to compare, which often includes normalization in some form.
- **Matchers**: This section contains the field-level comparison logic and is where to specify exactly how to match two company names.
- **Rules**: Rules determine how field-level matchers resolve if two objects are a match or not.
- **Match Code Generators**: When dealing with datasets of thousands or millions of objects, comparing every object with every other object is not a viable solution. To identify the records that should be compared, match codes must be generated for the applicable objects. Only records with at least one equal match code are then evaluated via rules.
- **Match Code Filter**: This component allows users to filter values that meet criteria set on a Transformation Lookup Table
- **Evaluator**: This section is used to test the comparison of two objects.



Data Elements

ID	Data Elements	Comment
normName	Name Normalizer(On Object)	
normAddress	Address Normalizer(On Object, DC:Main Address)	
normEmail	Email Normalizer(On Object, DC:Email)	
normPhone	Phone Normalizer(On Object, DC:Phone)	

Matchers

ID	Matcher	Comment
name	Name Matcher(normName)	
address	Address Matcher(normAddress)	
email	Email Matcher(normEmail)	
phone	Phone Matcher(normPhone)	

Rules

Rules Strategy: Max

#	address >70	email >70	name >70	phone >70	Result	Comment
1					(address*30.0 + name*30.0) / 60.0	
2					(name*30.0 + email*30.0) / 60.0	
3					(phone*30.0 + email*30.0) / 60.0	

Match Code Generators

Active	ID	Match Code Generator	Comment
✓	emailMatchCode	Email Match Code Generator: normEmail, EMAIL#	
✓	phoneMatchCode	Phone Match Code Generator: normPhone, PHONE#	
✓	nameAndAddress	Person Name and Address Match Code Generator: normName, null, normAddres...	

Match Code Filter

ID	Match Code Filter	Comment

Evaluator

Select Nodes: Aarone Kirk (558762) | Aarone Kirk (558774)

Matchers	Score

Rules	Score	Matched	Match Reason
1	100.0	true	address = true (100.0), email = true (100.0), name = true (100.0), phone = true (100.0)
2	100.0	true	address = true (100.0), email = true (100.0), name = true (100.0), phone = true (100.0)
3	100.0	true	address = true (100.0), email = true (100.0), name = true (100.0), phone = true (100.0)

Final Score: 100.0

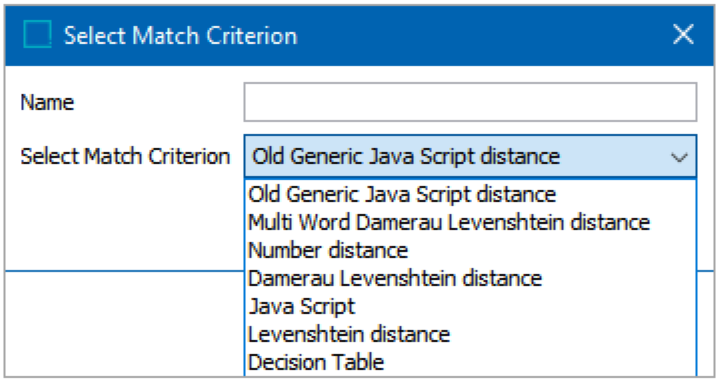
Common Match Codes: Yes

Match Code Generators	First Node Result	Second Node Result
emailMatchCode	EMAIL#FRINGILLA.EST@EU.ORG	EMAIL#FRINGILLA.EST@EU.ORG
	EMAIL#ROSETTAFALLACI@ARMYSPY.COM	EMAIL#ROSETTAFALLACI@ARMYSPY.COM
phoneMatchCode	PHONE#12263280522	PHONE#12263280522
	PHONE#13104011771	PHONE#13104011771
nameAndAddress	INDIVIDUAL#A+KRK+857438646	INDIVIDUAL#A+KRK+857438646
	INDIVIDUAL#A+KRK+TSN	INDIVIDUAL#A+KRK+TSN
	INDIVIDUAL#K+ARN+857438646	INDIVIDUAL#K+ARN+857438646
	INDIVIDUAL#K+ARN+TSN	INDIVIDUAL#K+ARN+TSN

Other Match Criteria

For match algorithms without embedded match codes, several legacy options exist for match criteria.

It is recommended that all new match algorithms are created with embedded match codes.


Select Match Criterion
Old Generic Java Script distance
Multi Word Damerau Levenshtein distance
Number distance
Damerau Levenshtein distance
Java Script
Levenshtein distance
Decision Table

Name:

Select Match Criterion: Old Generic Java Script distance

When match codes are not embedded into the matching algorithm, match code generators are not available. For more information, see the **Match Codes** topic in this documentation.

Match Criteria Configuration

To set up the match criteria for a match algorithm, the Matching Component Model must be populated. The user needs to know the data that are to be matched. One tool for such analysis is Data Profiling.

Matching Component Model

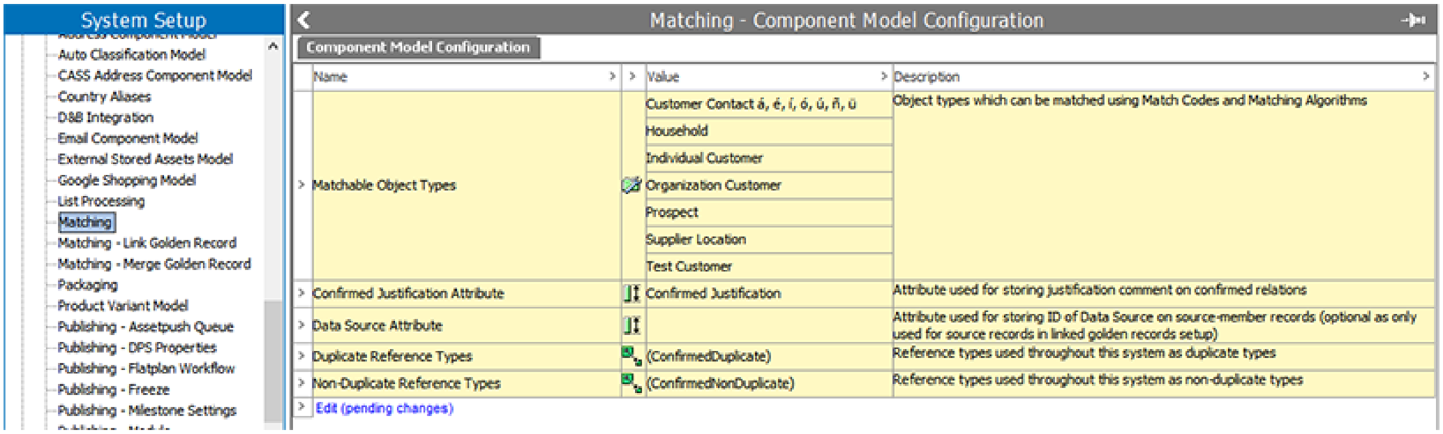
Before match codes can be generated and matching algorithms applied, the Matching Component Model must be configured. The Component Model determines which objects, attributes, and references are relevant to the configuration and how they apply.

> **Note:** Additional Component Models must be configured for certain Match Actions. For more information, see the **Match Codes** topic in this documentation.

All relevant Object Types, Attributes, and References must be created before they can be mapped to the component model.

The Matching Component Model defines all Objects Types that are allowed to be matched.

1. In System Setup, expand 'Component Models,' and click on the 'Matching' node.
2. On the 'Component Model Configuration' tab, click the Edit link.


System Setup
Auto Classification Model
CASS Address Component Model
Country Aliases
D&B Integration
Email Component Model
External Stored Assets Model
Google Shopping Model
List Processing
Matching
Matching - Link Golden Record
Matching - Merge Golden Record
Packaging
Product Variant Model
Publishing - Assetpush Queue
Publishing - DPS Properties
Publishing - Flatplan Workflow
Publishing - Freeze
Publishing - Milestone Settings
Matching - Component Model Configuration
Component Model Configuration

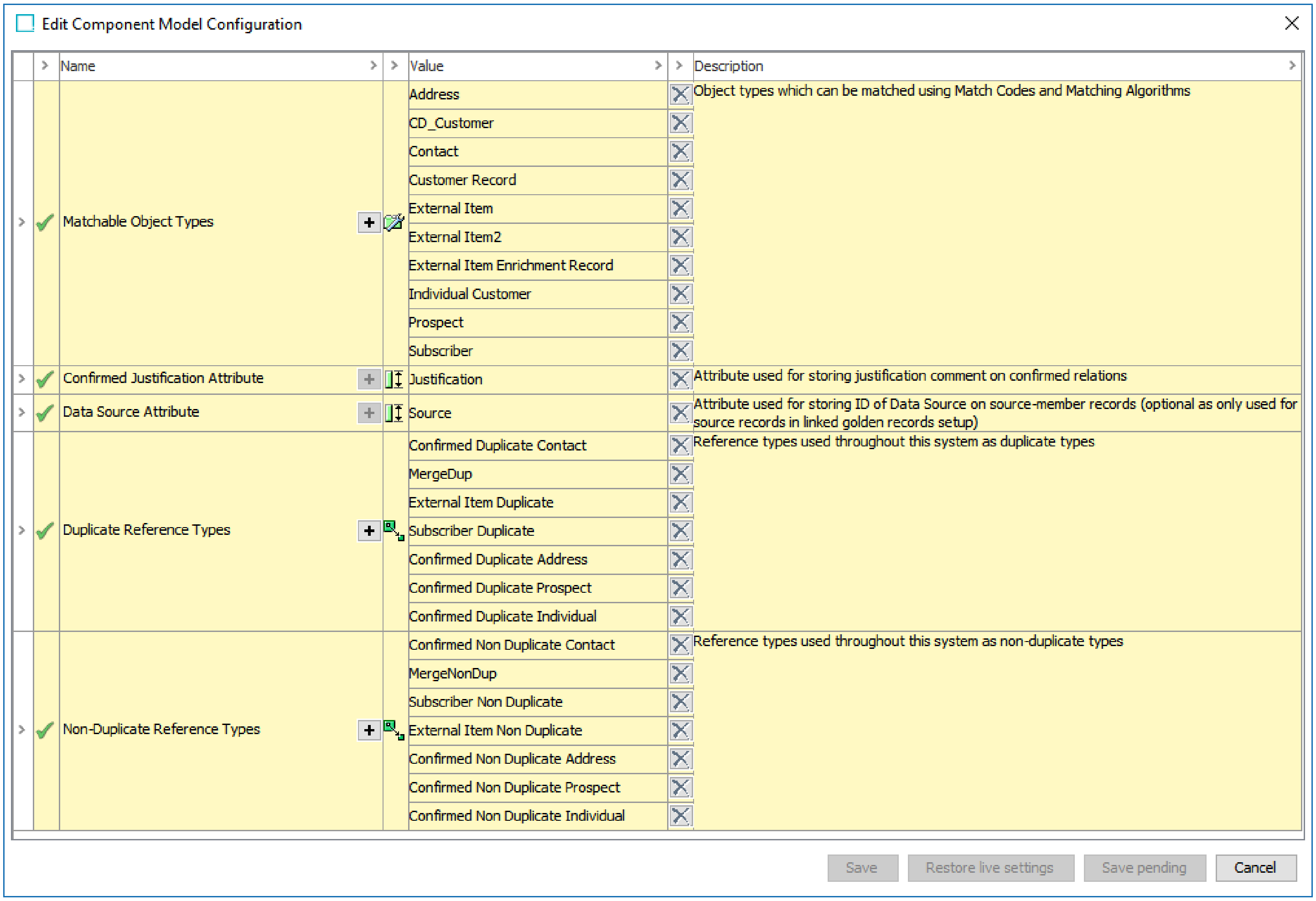
Name	Value	Description
Matchable Object Types	Customer Contact á, é, í, ó, ú, ñ, ü Household Individual Customer Organization Customer Prospect Supplier Location Test Customer	Object types which can be matched using Match Codes and Matching Algorithms
Confirmed Justification Attribute	Confirmed Justification	Attribute used for storing justification comment on confirmed relations
Data Source Attribute		Attribute used for storing ID of Data Source on source-member records (optional as only used for source records in linked golden records setup)
Duplicate Reference Types	(ConfirmedDuplicate)	Reference types used throughout this system as duplicate types
Non-Duplicate Reference Types	(ConfirmedNonDuplicate)	Reference types used throughout this system as non-duplicate types

Edit (pending changes)

3. Click the 'plus' button for the relevant component aspect to display the selection dialog, and then choose to add an object, attribute, or reference:

- **Matchable Object Types** – Select the object types that need to be matched. Only the object types configured can be used as object types for match codes. On objects of these types, the 'Matching' tab is automatically enabled. The 'Matching' tab shows match code values, potential duplicates, and confirmed relations for the selected object.

- **Confirmed Justification Attribute** – Select a valid description attribute for all reference types specified in the 'Duplicate Reference Types' and 'Non-Duplicate Reference Types' fields. This attribute stores a description explaining why two objects are marked as duplicates or non-duplicates.
- **Data Source Attribute** – Select one or more description attributes valid for all source object types specified in the 'Source Object Types' field. This attribute contains the source ID of the source objects. If you select more than one attribute in this field, then exactly one of these attributes must be valid per source object type chosen in the 'Source Object Types' field. This field is only required for Link Golden Records solutions with Trusted Source survivorship rules configured.
- **Duplicate Reference Types** – Select one or more reference types to store the manually maintained confirmed duplicate references. These references store the reason for confirming two objects as duplicates specified in the attribute selected in the 'Confirmed Justification Attribute' field. All the selected reference types must have exactly one valid attribute from the 'Confirmed Justification Attribute' field. Only the duplicate reference types you select can be used as 'Duplicate Type' on a matching algorithm. In a typical scenario, you will have different duplicate reference types for different matching algorithms. If you reuse duplicate reference type between algorithms, then the confirmed duplicates will be reused between those algorithms.
- **Non-Duplicate Reference Types** - Select one or more reference types used by the system for storing the manually maintained confirmed non-duplicate references. These references store the reason for confirming two objects as non-duplicates specified in the attribute selected in the 'Confirmed Justification Attribute' field. All the selected reference types must have exactly one valid attribute from the 'Confirmed Justification Attribute' field. Only reference types selected can be used as 'Non-Duplicate Type' on a matching algorithm. In a typical scenario, you will have different duplicate reference types for different matching algorithms. If you reuse the non-duplicate reference type between algorithms, then the confirmed non-duplicates will be reused between those algorithms as well.



Edit Component Model Configuration

	Name	Value	Description
✓	Matchable Object Types	Address CD_Customer Contact Customer Record External Item External Item2 External Item Enrichment Record Individual Customer Prospect Subscriber	Object types which can be matched using Match Codes and Matching Algorithms
✓	Confirmed Justification Attribute	Justification	Attribute used for storing justification comment on confirmed relations
✓	Data Source Attribute	Source	Attribute used for storing ID of Data Source on source-member records (optional as only used for source records in linked golden records setup)
✓	Duplicate Reference Types	Confirmed Duplicate Contact MergeDup External Item Duplicate Subscriber Duplicate Confirmed Duplicate Address Confirmed Duplicate Prospect Confirmed Duplicate Individual	Reference types used throughout this system as duplicate types
✓	Non-Duplicate Reference Types	Confirmed Non Duplicate Contact MergeNonDup Subscriber Non Duplicate External Item Non Duplicate Confirmed Non Duplicate Address Confirmed Non Duplicate Prospect Confirmed Non Duplicate Individual	Reference types used throughout this system as non-duplicate types

Click the 'X' button to remove the relevant object, attribute, or reference from the component model. A green checkmark will appear if the applicable row has a valid configuration.

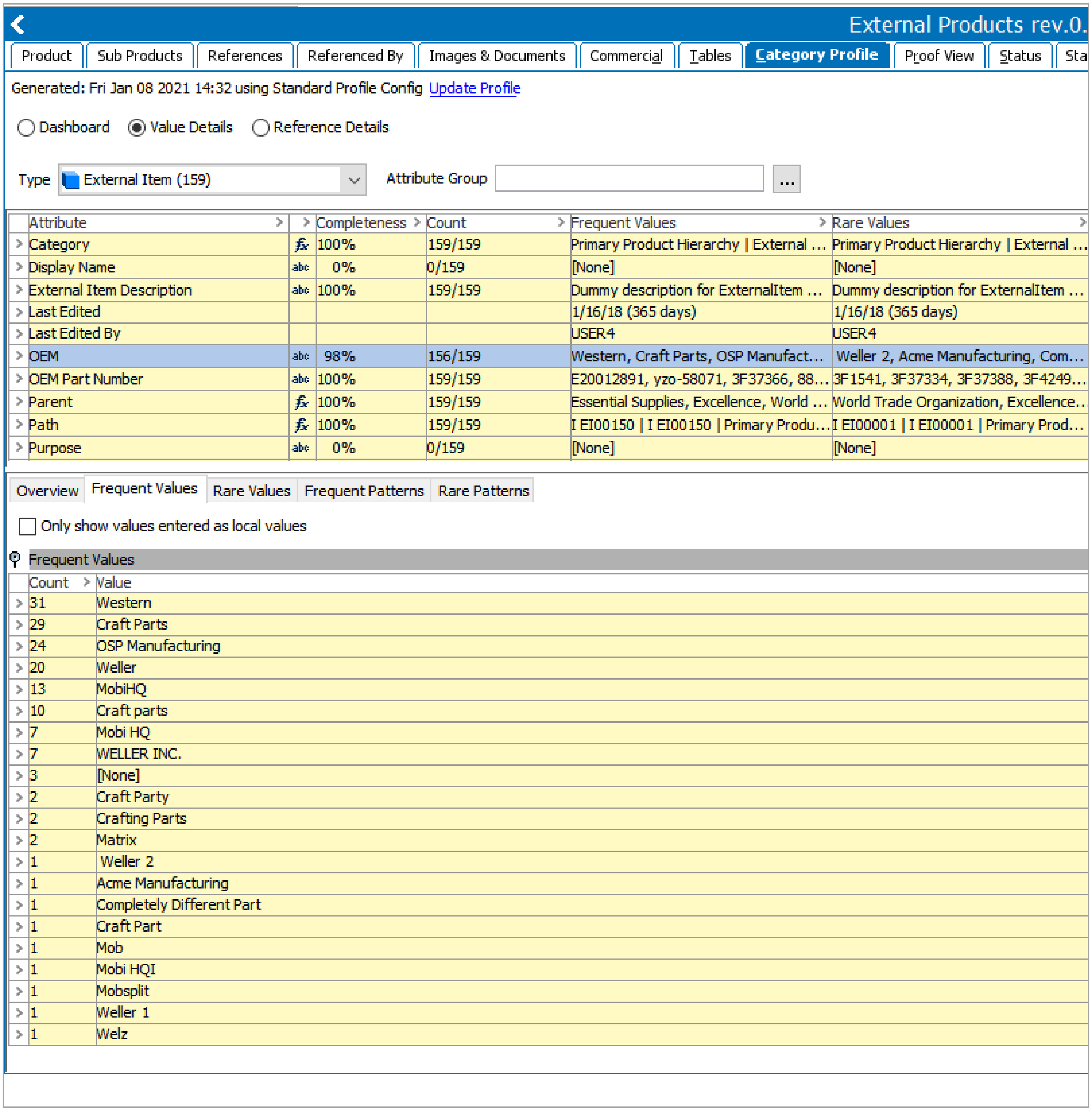
4. Click Save to save changes.

> **Note:** If you need to navigate away from the configuration dialog and some of the rows are not yet valid (they have an 'X' instead of a checkmark), click Save pending to save your work.

Data Profile Analysis as Preparation for Match Criteria

Designing a deduplication strategy requires an intimate understanding of the data, and to that end, STEP Data Profiles can be of great assistance. Data profiles show the extent to which relevant attributes are populated and highlight the most frequent and rare values and patterns. For more information, see the **Data Profiling** documentation.

If a profile is generated from the 'External Products' node, it is possible to see that there are missing values for both OEM and OEM Part Number. This ability to highlight missing values should be accounted for in the deduplication strategy. Furthermore, as illustrated below, the profile shows that the OEM values include obvious duplicates like 'Craft Parts' / 'Craft parts' and 'Weller' / 'WELLER INC,' indicating that some form of normalization is required.



Generated: Fri Jan 08 2021 14:32 using Standard Profile Config Update Profile

○ Dashboard ● Value Details ○ Reference Details

Type: External Item (159) Attribute Group:

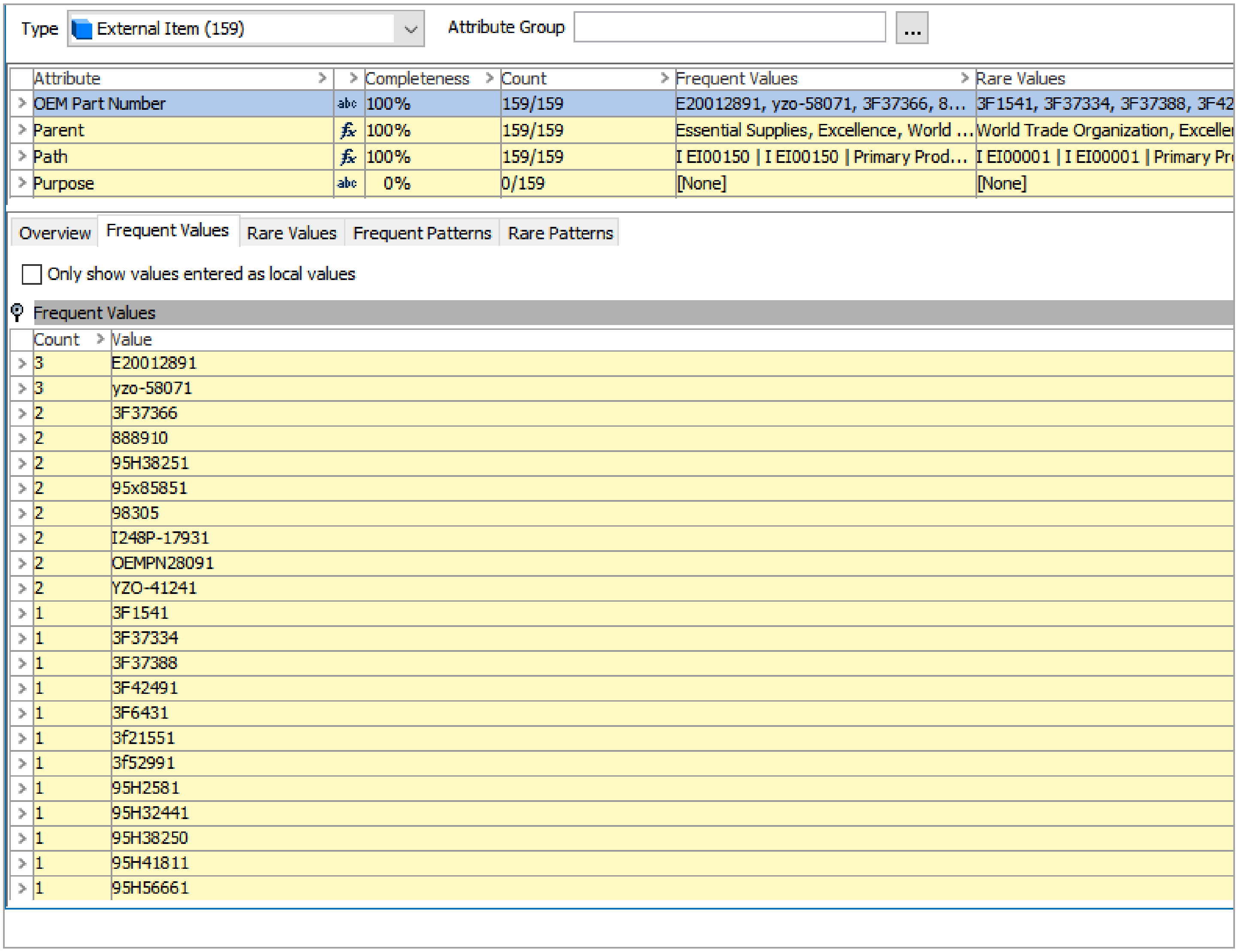
Attribute		Completeness	Count	Frequent Values	Rare Values
Category	fx	100%	159/159	Primary Product Hierarchy | External ...	Primary Product Hierarchy | External ...
Display Name	abc	0%	0/159	[None]	[None]
External Item Description	abc	100%	159/159	Dummy description for ExternalItem ...	Dummy description for ExternalItem ...
Last Edited				1/16/18 (365 days)	1/16/18 (365 days)
Last Edited By				USER4	USER4
OEM	abc	98%	156/159	Western, Craft Parts, OSP Manufact...	Weller 2, Acme Manufacturing, Com...
OEM Part Number	abc	100%	159/159	E20012891, yzo-58071, 3F37366, 88...	3F1541, 3F37334, 3F37388, 3F4249...
Parent	fx	100%	159/159	Essential Supplies, Excellence, World ...	World Trade Organization, Excellence...
Path	fx	100%	159/159	I EI00150 | I EI00150 | Primary Produ...	I EI00001 | I EI00001 | Primary Prod...
Purpose	abc	0%	0/159	[None]	[None]

□ Only show values entered as local values

Frequent Values

Count	Value
31	Western
29	Craft Parts
24	OSP Manufacturing
20	Weller
13	MobiHQ
10	Craft parts
7	Mobi HQ
7	WELLER INC.
3	[None]
2	Craft Party
2	Crafting Parts
2	Matrix
1	Weller 2
1	Acme Manufacturing
1	Completely Different Part
1	Craft Part
1	Mob
1	Mobi HQI
1	Mobsplit
1	Weller 1
1	Welz

For OEM Part Number, there are more than one hundred distinct values, and thus, the profile does not provide exact statistics with the default settings. Still, it is possible to see that both uppercase and lowercase letters are used, and that punctuation is used in some values and not in others. Again, this indicates that normalization will be required.



Type: External Item (159)   Attribute Group: ...

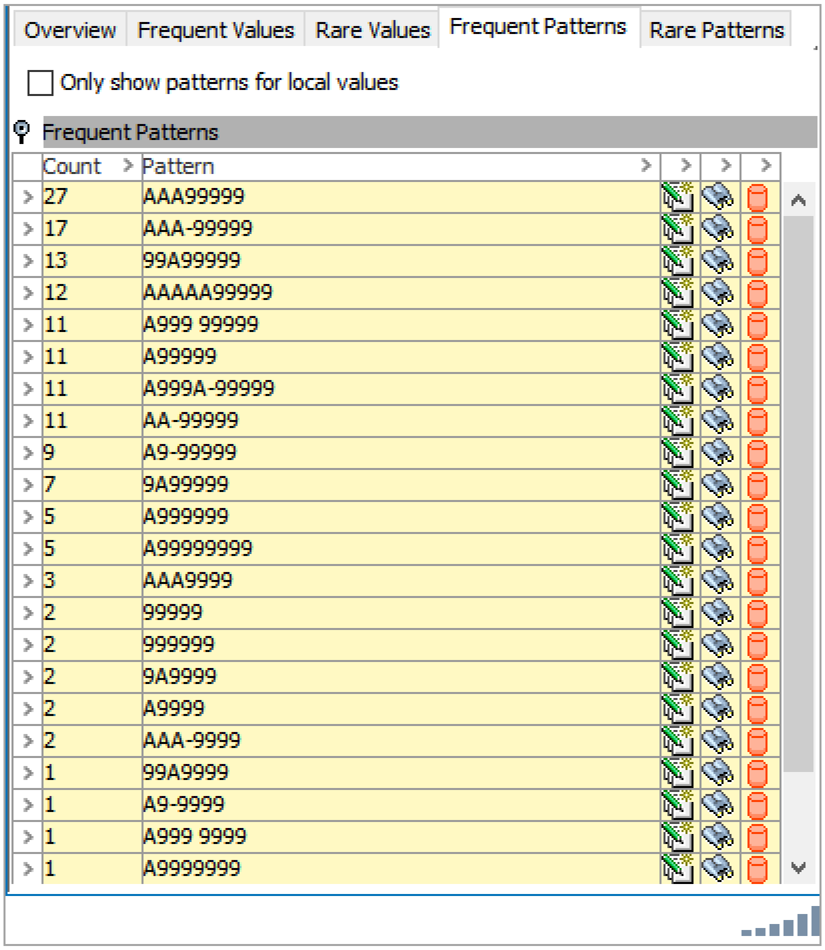
Attribute		Completeness	Count	Frequent Values	Rare Values
OEM Part Number	abc	100%	159/159	E20012891, yzo-58071, 3F37366, 8...	3F1541, 3F37334, 3F37388, 3F42
Parent	fx	100%	159/159	Essential Supplies, Excellence, World ...	World Trade Organization, Excelle
Path	fx	100%	159/159	I EI00150 | I EI00150 | Primary Prod...	I EI00001 | I EI00001 | Primary Pr
Purpose	abc	0%	0/159	[None]	[None]

□ Only show values entered as local values

Frequent Values

Count	Value
3	E20012891
3	yzo-58071
2	3F37366
2	888910
2	95H38251
2	95x85851
2	98305
2	I248P-17931
2	OEMPN28091
2	YZO-41241
1	3F1541
1	3F37334
1	3F37388
1	3F42491
1	3F6431
1	3f21551
1	3f52991
1	95H2581
1	95H32441
1	95H38250
1	95H41811
1	95H56661

Notice, that when looking at the frequent patterns info, there are no clear, distinct patterns in the values.



□ Only show patterns for local values

Frequent Patterns

Count	Pattern
27	AAA99999
17	AAA-99999
13	99A99999
12	AAAAA99999
11	A999 99999
11	A99999
11	A999A-99999
11	AA-99999
9	A9-99999
7	9A99999
5	A999999
5	A99999999
3	AAA9999
2	99999
2	999999
2	9A9999
2	A9999
2	AAA-9999
1	99A9999
1	A9-9999
1	A999 9999
1	A9999999

With two 'matching' attributes, it would be possible to generate two match codes per object, but for this case, this is likely not the best strategy because the number of different OEM values is quite low, especially if they are normalized. Further, comparing all items from the same OEM would result in too many comparisons.

As there are a significant spread in OEM Part Numr values, generating match codes based solely on these values could work. Additionally, the OEM value should be used as a basis for match since a specific OEM Part Number value pattern to an OEM cannot be be assumed. For example, a match on OEM Part Number is not necessarily a true match as these values are reused. However, this approach would require that the matching algorithm logic inspect the OEMs later to determine if there was a match or not.

A possible solution is to generate composite match codes that include information from both attributes. Suppose the values are normalized during the match code generation. In that case, it will be possible to simplify the setup so that identical match codes are automatically considered a match. This strategy can be achieved by working with a Window Size of one, which only compares objects with the same match code, and the matching algorithm logic does not check anything, but it indicates a match for each comparison.

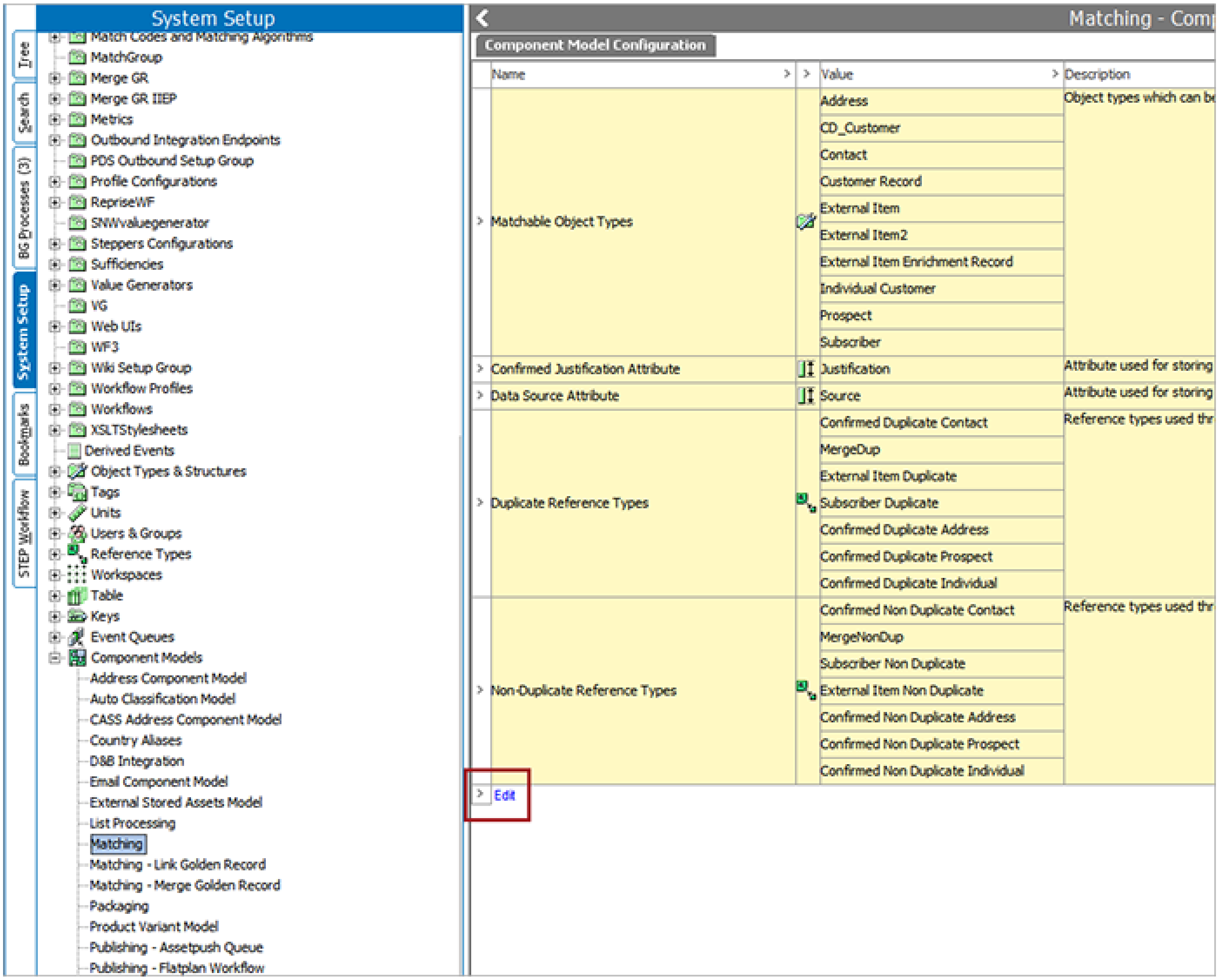
Matching Component Model

Before match codes can be generated and matching algorithms applied, the Matching Component Model must be configured. The Component Model determines which objects, attributes, and references are relevant to your configuration and how these configurations are used.

> **Note:** Additional Component Models must be configured for certain Match Actions. See Match Actions

All relevant Object Types, Attributes, and References must be created before they can be mapped to the component model. The Matching Component Model defines all Objects Types that are allowed to be matched.

1. In System Setup, expand 'Component Models,' and click on the 'Matching' node.
2. On the 'Component Model Configuration' tab, click the Edit link.

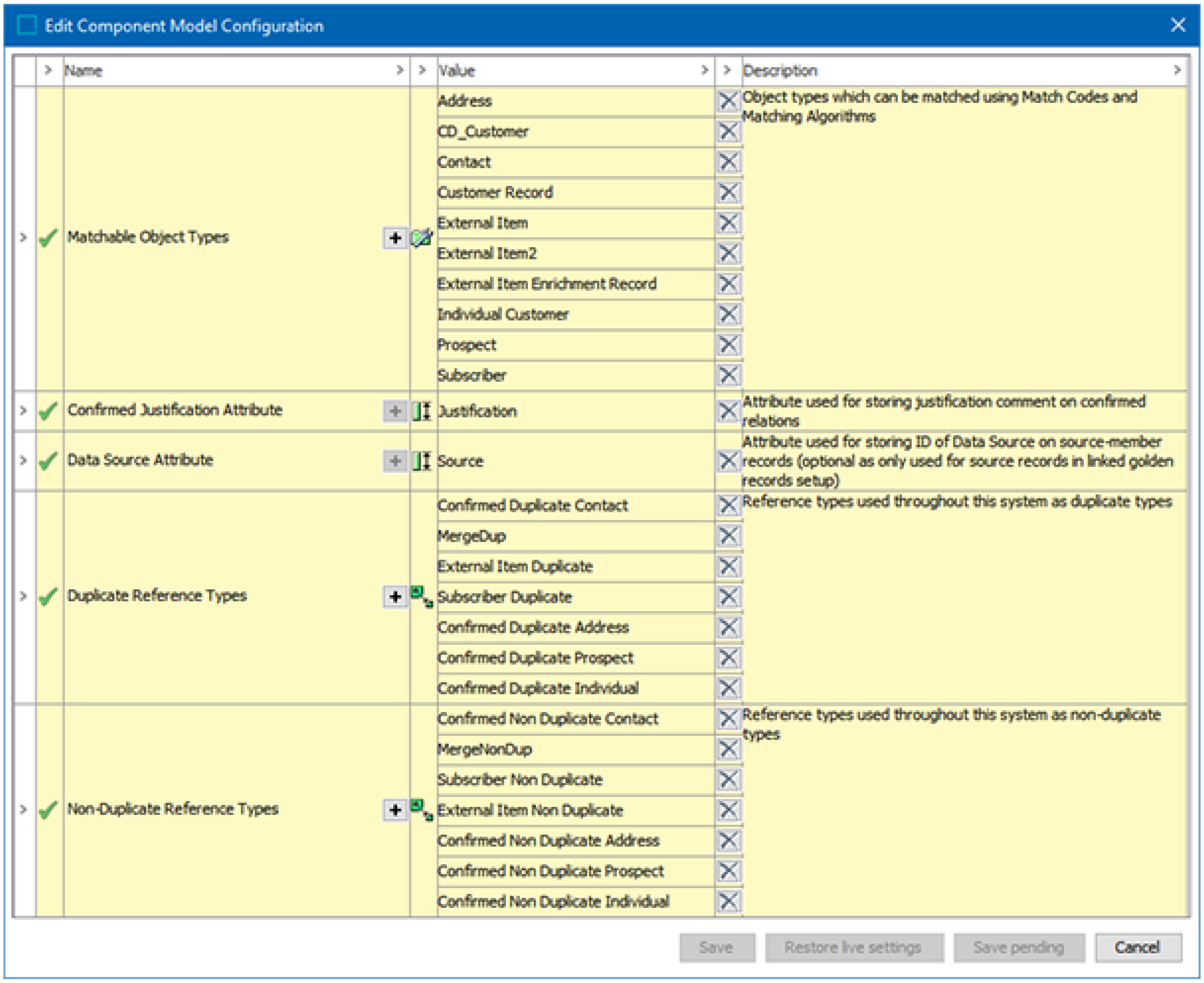


Name	Value	Description
Matchable Object Types	Address CD_Customer Contact Customer Record External Item External Item2 External Item Enrichment Record Individual Customer Prospect Subscriber	Object types which can be
Confirmed Justification Attribute	Justification	Attribute used for storing
Data Source Attribute	Source	Attribute used for storing
Duplicate Reference Types	Confirmed Duplicate Contact MergeDup External Item Duplicate Subscriber Duplicate Confirmed Duplicate Address Confirmed Duplicate Prospect Confirmed Duplicate Individual	Reference types used thr
Non-Duplicate Reference Types	Confirmed Non Duplicate Contact MergeNonDup Subscriber Non Duplicate External Item Non Duplicate Confirmed Non Duplicate Address Confirmed Non Duplicate Prospect Confirmed Non Duplicate Individual	Reference types used thr

3. Click the 'plus' button for the relevant component aspect to display the selection dialog, and then choose to add an object, attribute, or reference:

- **Matchable Object Types** – Select the object types that need to be matched. Only the object types configured can be used as object types for match codes. On objects of these types, the 'Matching' tab is automatically enabled. The 'Matching' tab shows match code values, potential duplicates, and confirmed relations for the selected object.

- **Confirmed Justification Attribute** – Select a valid description attribute for all reference types specified in the 'Duplicate Reference Types' and 'Non-Duplicate Reference Types' fields. This attribute stores a description explaining why two objects are marked as duplicates or non-duplicates.
- **Data Source Attribute** – Select one or more description attributes valid for all source object types specified in the 'Source Object Types' field. This attribute contains the source ID of the source objects. If choosing more than one attribute in this field, then exactly one of these attributes must be valid per source object type selected in the 'Source Object Types' field. This field is only required for Link Golden Records solutions with Trusted Source survivorship rules configured.
- **Duplicate Reference Types** – Select one or more reference types to store the manually maintained confirmed duplicate references. These references store the reason for confirming two objects as duplicates specified in the attribute selected in the 'Confirmed Justification Attribute' field. All the selected reference types must have exactly one valid attribute from the 'Confirmed Justification Attribute' field. Only the duplicate reference types you select can be used as 'Duplicate Type' on a matching algorithm. In a typical scenario, you will have different duplicate reference types for different matching algorithms. If you reuse duplicate reference type between algorithms, then the confirmed duplicates will be reused between those algorithms.
- **Non-Duplicate Reference Types** - Select one or more reference types used by the system for storing the manually maintained confirmed non-duplicate references. These references store the reason for confirming two objects as non-duplicates as specified in the attribute selected in the 'Confirmed Justification Attribute' field. All the selected reference types must have exactly one valid attribute from the 'Confirmed Justification Attribute' field. Only reference types selected can be used as 'Non-Duplicate Type' on a matching algorithm. In a typical scenario, you will have different duplicate reference types for different matching algorithms. If you reuse non-duplicate reference types between algorithms, then the confirmed non-duplicates will be reused between those algorithms as well.



Edit Component Model Configuration

	Name	Value		Description
✓	Matchable Object Types	Address	☒	Object types which can be matched using Match Codes and Matching Algorithms
		CD_Customer	☒	
		Contact	☒	
		Customer Record	☒	
		External Item	☒	
		External Item2	☒	
		External Item Enrichment Record	☒	
		Individual Customer	☒	
		Prospect	☒	
		Subscriber	☒	
✓	Confirmed Justification Attribute	Justification	☒	Attribute used for storing justification comment on confirmed relations
✓	Data Source Attribute	Source	☒	Attribute used for storing ID of Data Source on source-member records (optional as only used for source records in linked golden records setup)
✓	Duplicate Reference Types	Confirmed Duplicate Contact	☒	Reference types used throughout this system as duplicate types
		MergeDup	☒	
		External Item Duplicate	☒	
		Subscriber Duplicate	☒	
		Confirmed Duplicate Address	☒	
		Confirmed Duplicate Prospect	☒	
		Confirmed Duplicate Individual	☒	
✓	Non-Duplicate Reference Types	Confirmed Non Duplicate Contact	☒	Reference types used throughout this system as non-duplicate types
		MergeNonDup	☒	
		Subscriber Non Duplicate	☒	
		External Item Non Duplicate	☒	
		Confirmed Non Duplicate Address	☒	
		Confirmed Non Duplicate Prospect	☒	
		Confirmed Non Duplicate Individual	☒	

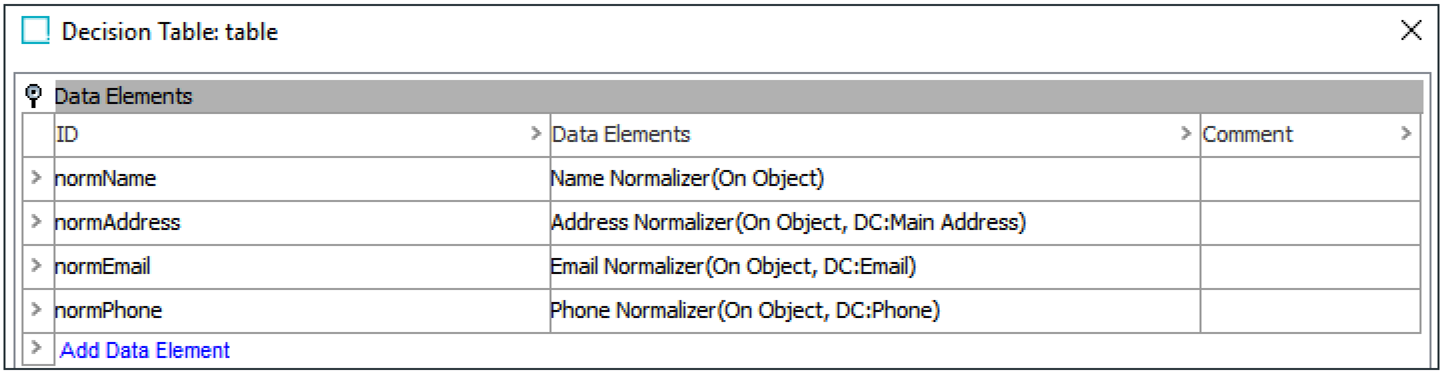
Click the 'X' button to remove the relevant object, attribute, or reference from the component model. A green checkmark will appear if the applicable row has a valid configuration.

4. Click Save to save changes.

> **Note:** If you need to navigate away from the configuration dialog and some of the rows are not yet valid (they have an 'X' instead of a checkmark), click Save pending to save your work.

Match Criteria Data Elements

The Data Elements section of a decision table defines input data for the match criteria. The data element is responsible for retrieving data, and making it easy to compare. This often involves the reduction of data to a kind of canonical form, like lowercasing letters in a text, removing spaces from phone numbers, or expanding abbreviations.


Decision Table: table

Data Elements

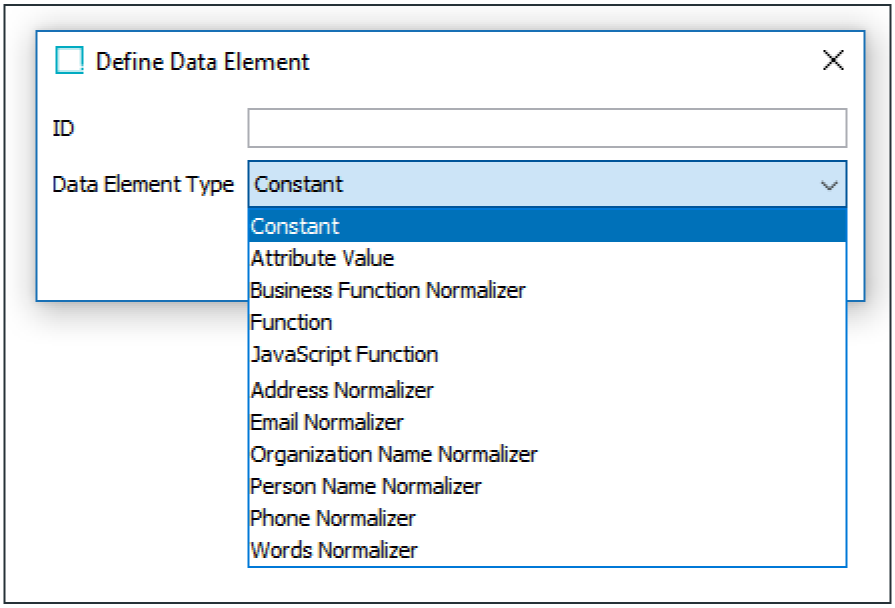
ID	Data Elements	Comment
normName	Name Normalizer(On Object)	
normAddress	Address Normalizer(On Object, DC:Main Address)	
normEmail	Email Normalizer(On Object, DC:Email)	
normPhone	Phone Normalizer(On Object, DC:Phone)	

Most data elements take data from source objects, then normalize it in some form, before providing the data to matchers and match code generators.

The following data element types are available:

- Standard Data Elements
  - Constant
  - Attribute Value
  - Business Function Normalizer
  - Function
  - JavaScript Function
- Party Data Normalizers
  - Address Normalizer
  - Email Normalizer
  - Organization Name Normalizer
  - Person Name Normalizer
  - Phone Normalizer
  - Words Normalizer

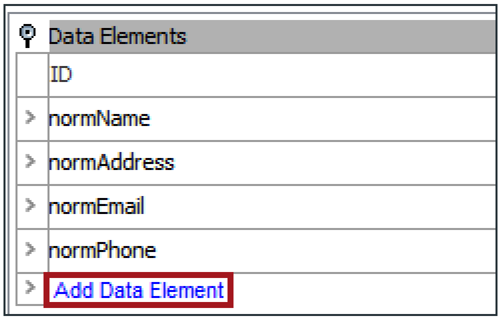
Data elements can be chained so that the output of one data element can be used as input to another data element.



ID

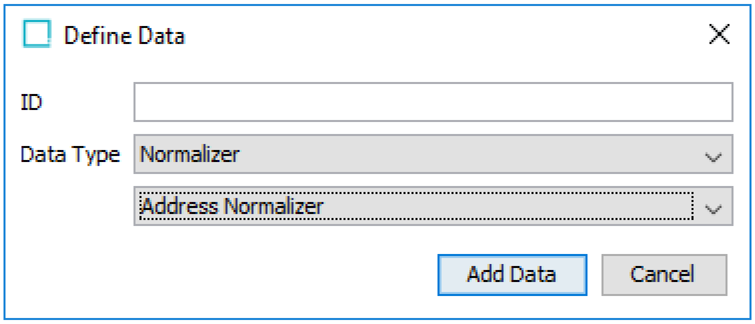
Data Element Type: Constant

Configuration



Data Elements
ID
normName
normAddress
normEmail
normPhone
Add Data Element

1. To add data to the table, click the 'Add Data' element link.
2. In the 'Define Data Element' popup dialog, enter an ID for the data element and use the Data Type dropdown (s) to define the data type, then click the Add Data button.



ID

Data Type: Normalizer

Address Normalizer

Important: No two data elements, matchers, or match code generators should have the same ID.

3. The new data element will be present in the table, but is still not configured. Click the ellipsis button (...) to populate the data element. See the section corresponding to your chosen data element under **Standard Data Elements** and **Party Data Normalizers** below for more information.

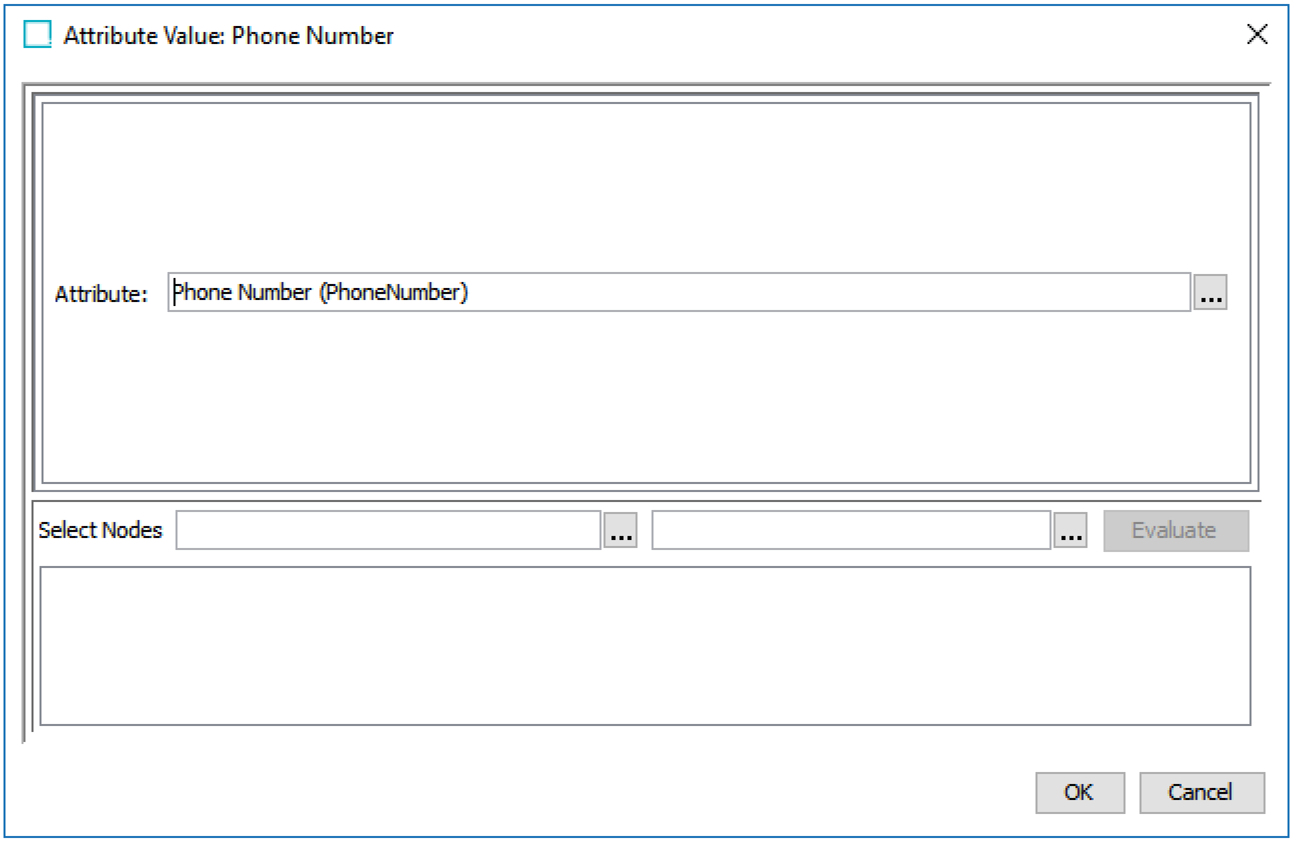
Standard Data Elements

Standard data elements include:

- Constants
- Attribute Value
- Business Function Normalizer
- STEP Functions
- JavaScript Functions

Attribute Value

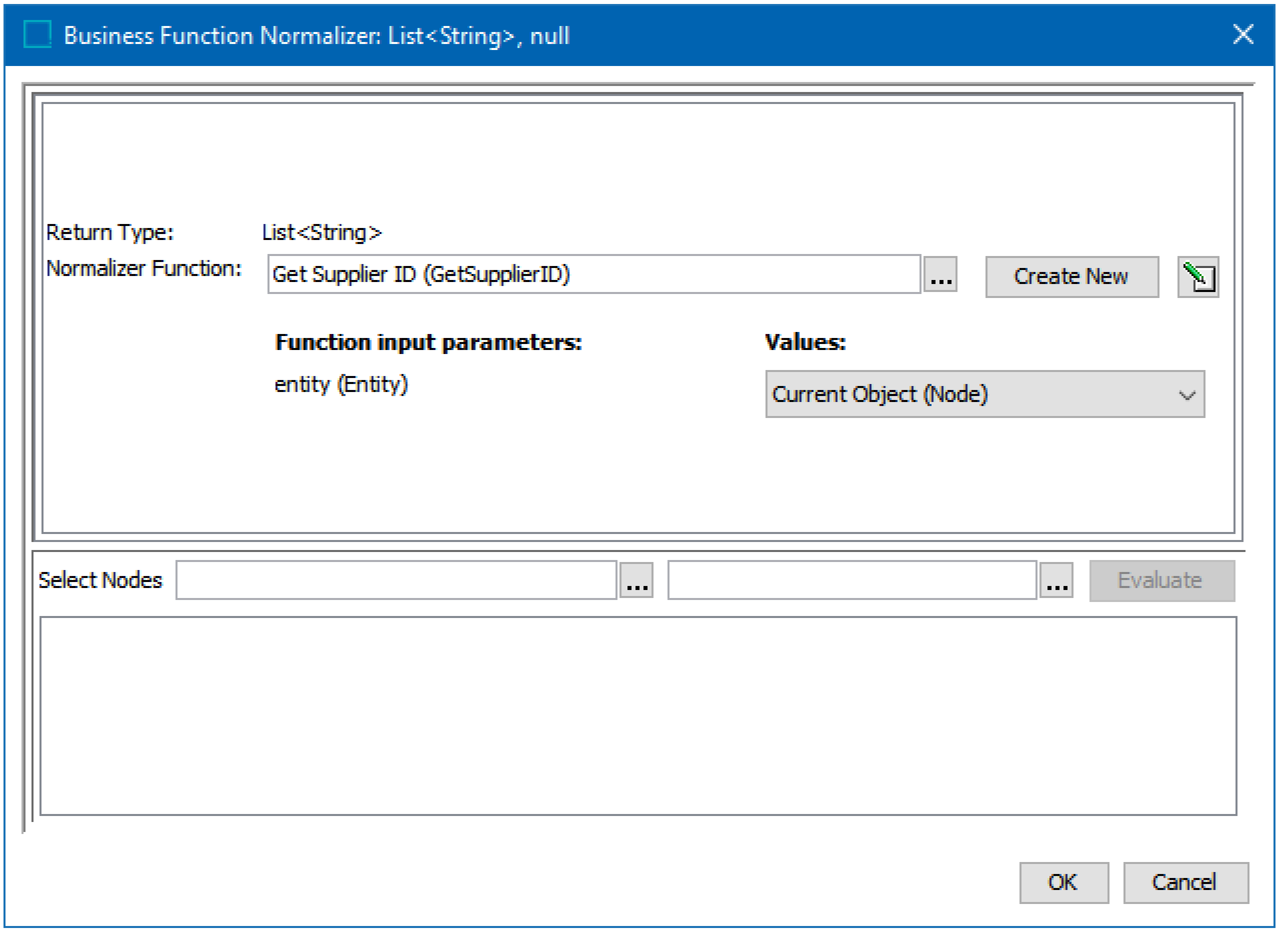
The Attribute Value allows users to specify a single attribute and output its value. To specify an attribute, click the ellipsis button (...) and browse or search for the desire attribute. This data element does not normalize its output.


Attribute Value: Phone Number

Attribute: Phone Number (PhoneNumber)

Business Function Normalizer

A Business Function Normalizer is the most versatile normalizer in the toolbox. Using a business function, it can take any number of values from the source records and produce a normalized data element. For more information, see the **Business Functions** topic of the **Business Rules** documentation.



Business Function Normalizer: List<String>, null

Return Type: List<String>

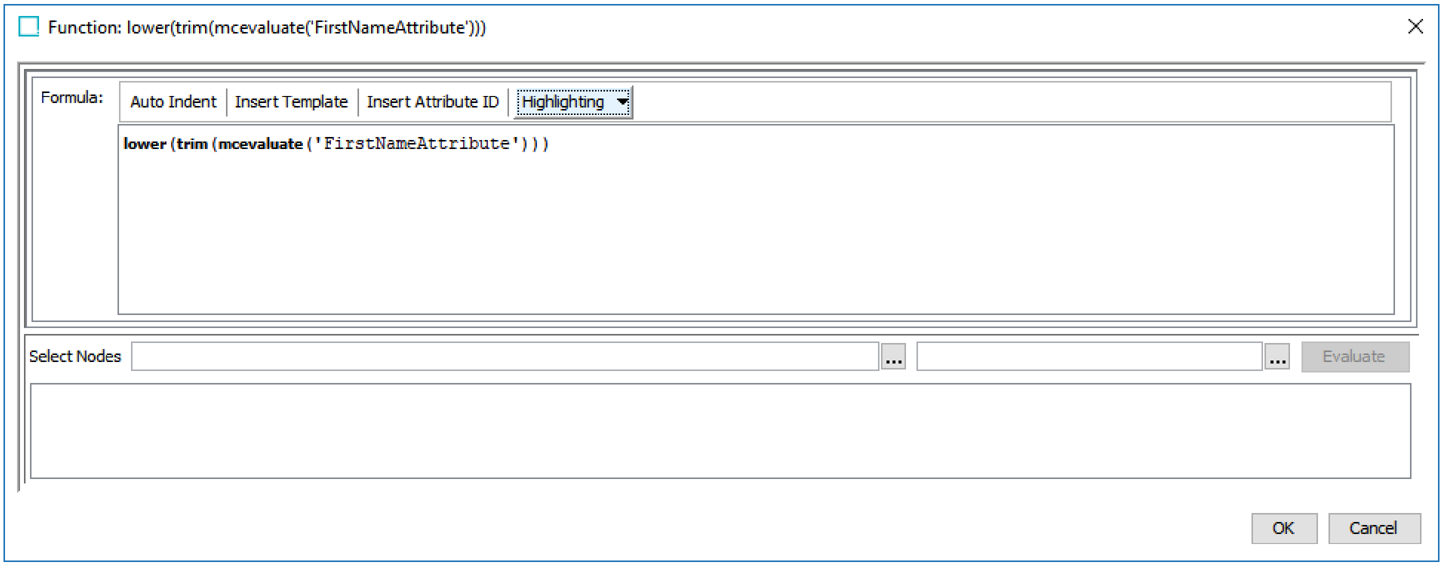
Normalizer Function: Get Supplier ID (GetSupplierID)

Function input parameters: entity (Entity)

Values: Current Object (Node)

STEP Function

The element is called 'Function' on the dropdown list, and this option will normalize values via built-in STEP functions.



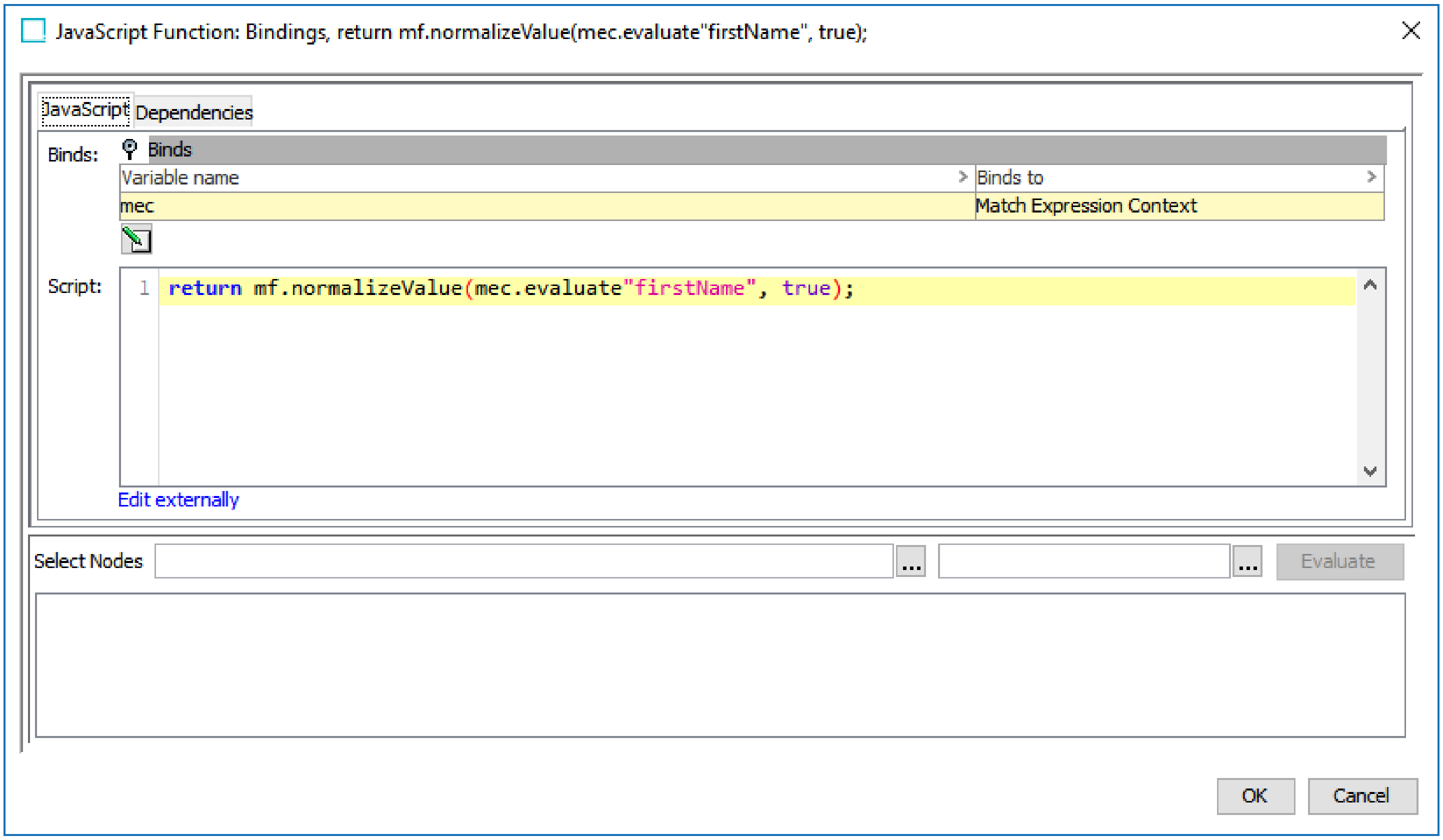
Function: lower(trim(mcevaluate('FirstNameAttribute')))

Formula:

```
lower(trim(mcevaluate('FirstNameAttribute')))
```

JavaScript Function

Normalized values produced via JavaScript functions.



JavaScript Function: Bindings, return mf.normalizeValue(mec.evaluate"firstName", true);

Binds:

Variable name	Binds to
mec	Match Expression Context

Script:

```
return mf.normalizeValue(mec.evaluate"firstName", true);
```

Party Data Normalizers

These normalizer templates are intended for matching of party data:

- Address Normalizer
- Email Normalizer
- Organization Name Normalizer
- Person Name Normalizer
- Phone Normalizer
- Words Normalizer

Address Normalizer

The Address Normalizer produces a normalized set of addresses for use in the corresponding Address Matcher.

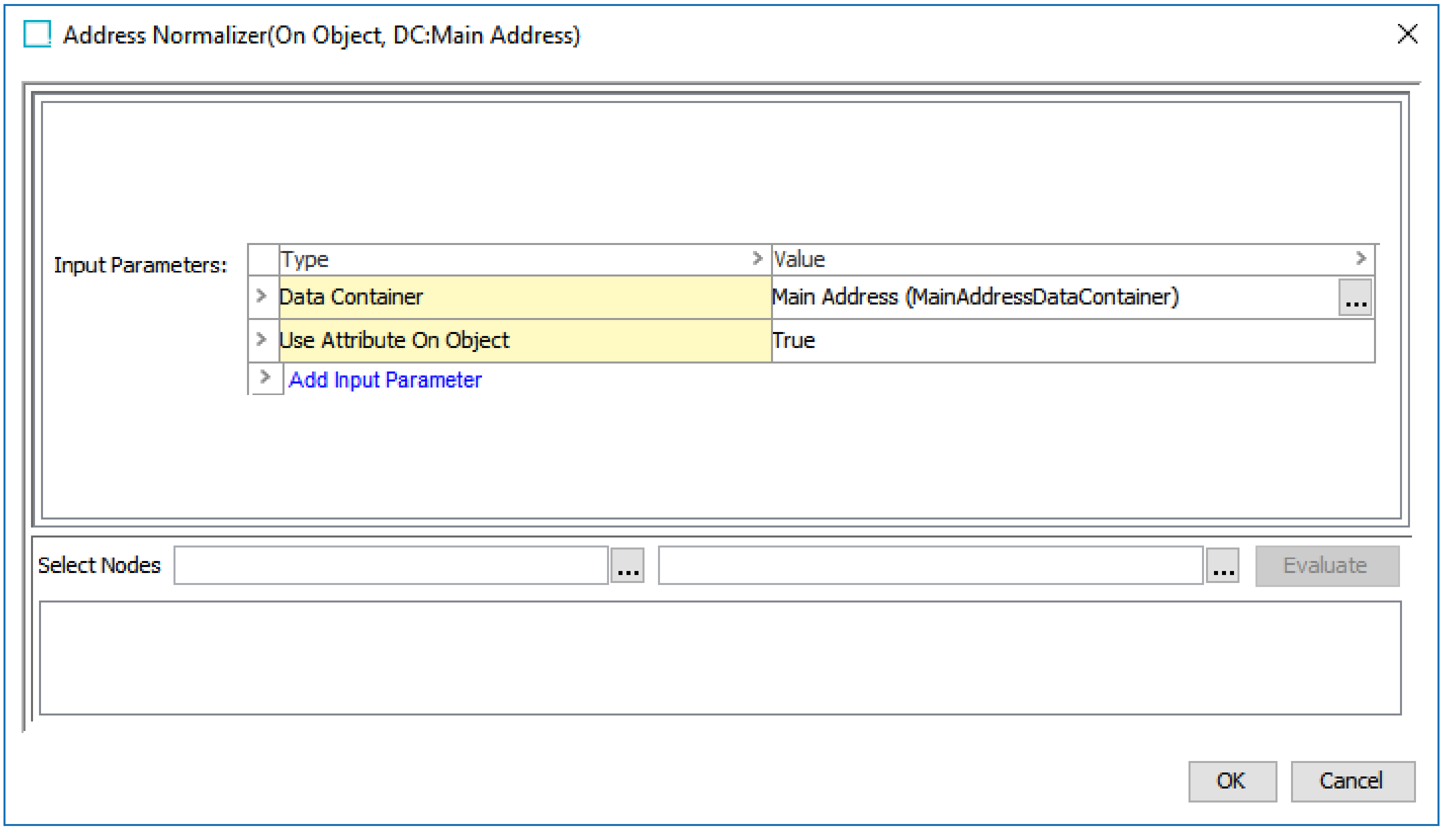
This data is provided by the input address element attributes mapped to the Address component model and include the following: Input City, Input Country, Input State, Input Street, Input Zip, Standardized City, Standardized Country, Standardized Country ISO Code, Standardized State, Standardized Street, Standardized Zip.

The output of the Address Normalizer is java.util.Set<com.stibo.partydatamatching.domain.address.Address>

Note: If the postal code mapped to the corresponding standardized component model parameter is valid for the objects being compared, the decision table will output normalized values for address attributes mapped to the standardized attribute fields on the component model.

For more information on the Address Component Model, see the **Address Component Model** section of the **Data Integration** documentation.

1. To normalize customer address data, click the ellipsis button (...) in the Data column to access the configuration.

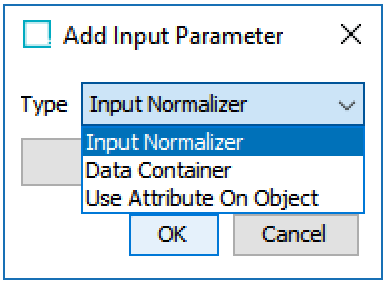


Address Normalizer(On Object, DC:Main Address)

Input Parameters:

Type	Value
Data Container	Main Address (MainAddressDataContainer)
Use Attribute On Object	True

2. For Input Parameters, click the Add Input Parameter link, and in the dialog, select the input type from the dropdown.



Add Input Parameter

Type: Input Normalizer

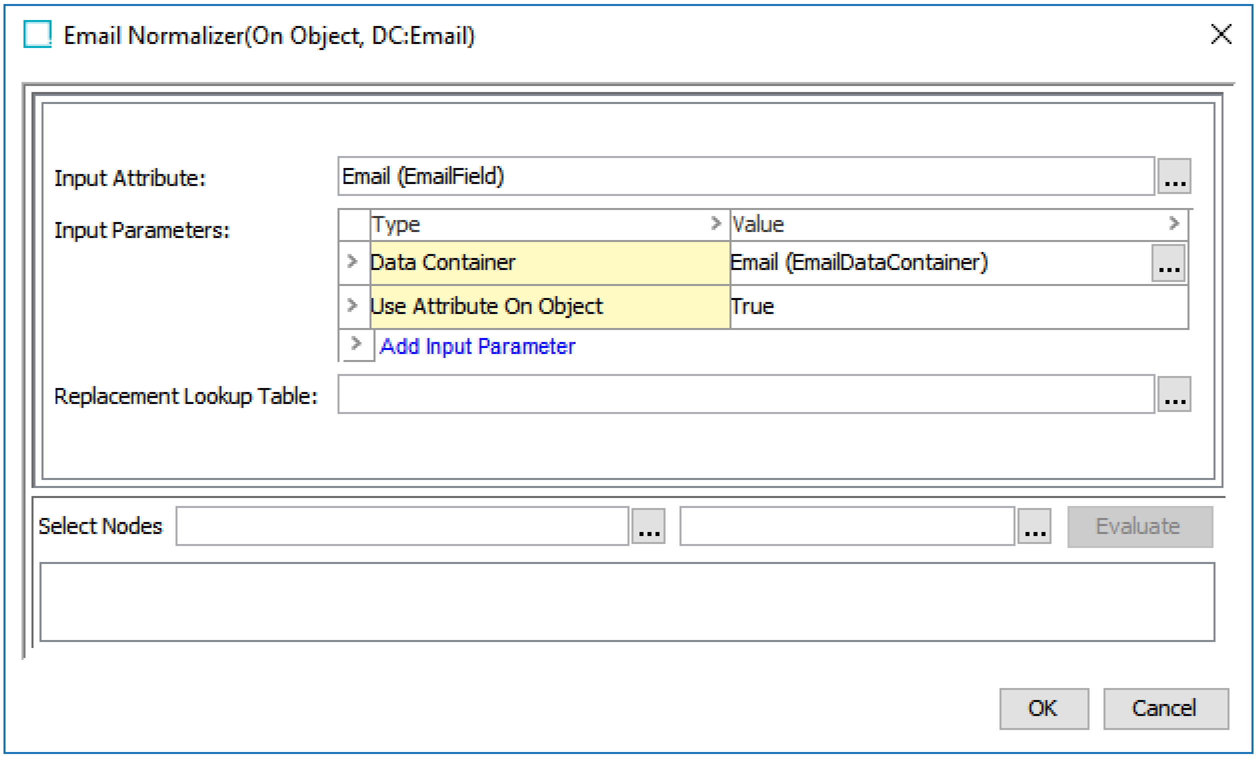
- Input Parameter: Select either a data container or the node object itself, which must hold the input attributes the Address Normalizer use, as defined by the component model. There is also the option of using an Input Normalizer, typically a business function written in JavaScript.
- Data Container: When you select 'Data Container,' click the ellipsis button (...) and browse or search for a data container type. The selected data container type will have its address data normalized when used in a matcher.
- Use Attribute on Object: When you select 'Use Attribute On Object,' click the dropdown and select 'True.' This input parameter will normalize address data that has been mapped to the address component model. This input parameter will be configured by default.

3. Click OK once the selection is made.

Email Normalizer

An email normalizer can normalize email data for use in the corresponding email matcher.

1. To normalize email data, click the ellipsis button (...) in the Data column to access the configuration.



Email Normalizer(On Object, DC:Email)

Input Attribute: Email (EmailField)

Input Parameters:

Type	Value
Data Container	Email (EmailDataContainer)
Use Attribute On Object	True

Add Input Parameter

Replacement Lookup Table:

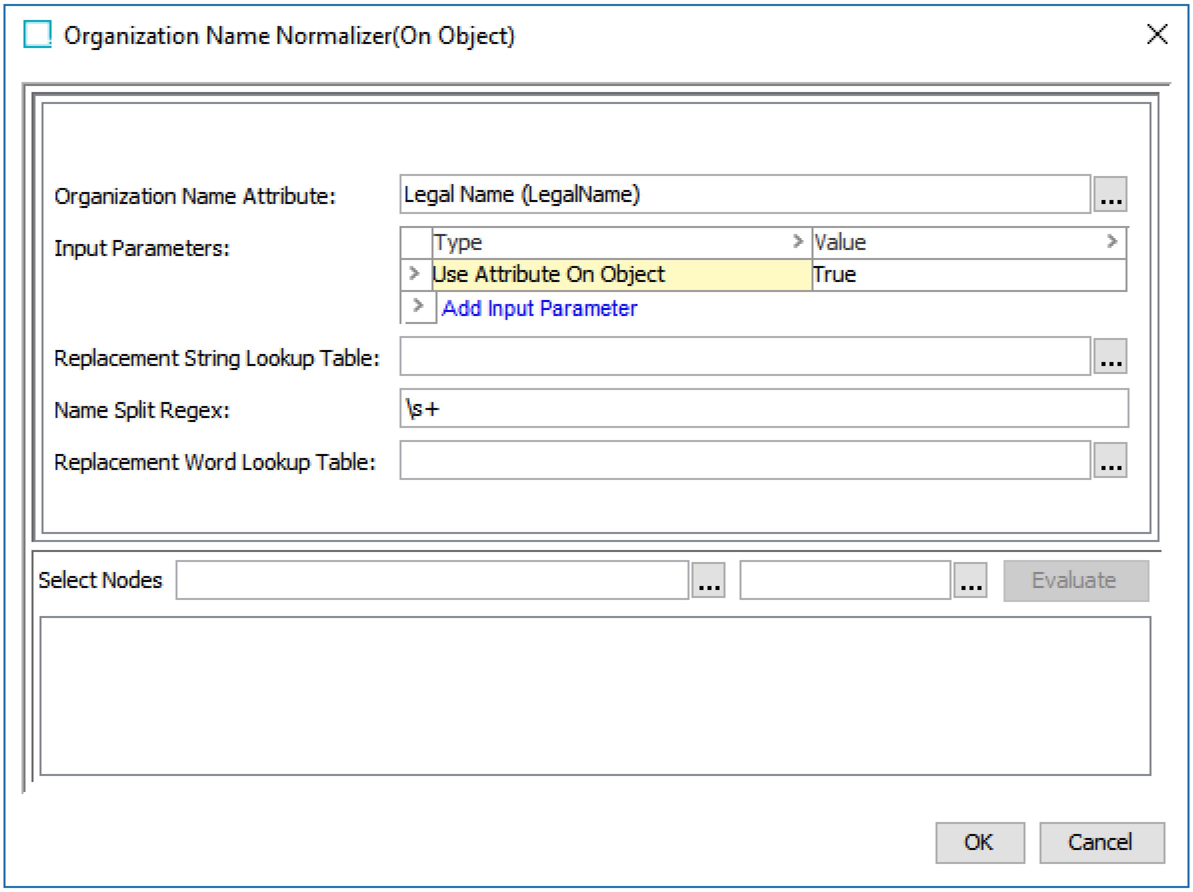
Select Nodes

2. For the Input Attribute parameter, click the ellipsis button (...) and browse or search for an email attribute to normalize.
3. For Input Parameters, click the Add Input Parameter link, and in the Add Input Parameter popup dialog, select the input type from the dropdown.
   - Input Normalizer: Enter the ID of another Email Normalizer to use its output. Typically, this normalizer is written in JavaScript.
   - Data Container: When you select 'Data Container,' click the ellipsis button (...) and browse or search for a data container. The selected data container will have its email data normalized when used in a matcher.
   - Use Attribute on Object: When you select 'Use Attribute On Object,' click the dropdown and select 'True.' This input parameter will normalize the email attribute mapped to the Input Attribute field. This input parameter will be configured by default.
4. For the Replacement Lookup Table parameter, click the ellipsis button (...) and select a lookup table. Typically, this is used to remove invalid email values.
5. Click OK when finished.

Organization Name Normalizer

An Organization Name Normalizer can normalize organization name data for use in the corresponding Organization Name Matcher.

1. To normalize organization name data, click the ellipsis button (...) in the Data column to access the configuration.



Organization Name Normalizer(On Object)

Organization Name Attribute: Legal Name (LegalName)

Input Parameters:

Type	Value
Use Attribute On Object	True

Replacement String Lookup Table:

Name Split Regex: \s+

Replacement Word Lookup Table:

2. For the Organization Name Attribute parameter, click the ellipsis button (...) and browse or search for an organization name attribute to normalize.
3. For Input Parameters, click the Add Input Parameter link, and in the Add Input Parameter popup dialog, select the input type from the dropdown.
   - Input Normalizer: Enter the ID of another Organization Name Normalizer to use its output. Typically, this normalizer is written in JavaScript.
   - Data Container: When you select 'Data Container,' click the ellipsis button (...) and browse or search for a data container. The selected data container will have its organization name data normalized when used in a matcher.

- Use Attribute on Object: When you select 'Use Attribute On Object,' click the dropdown and select 'True.' This input parameter will normalize the organization name attribute mapped to the Organization Name Attribute field. This input parameter will be configured by default.

4. For the Replacement String Lookup Table parameter, click the ellipsis button (...) and select a lookup table.

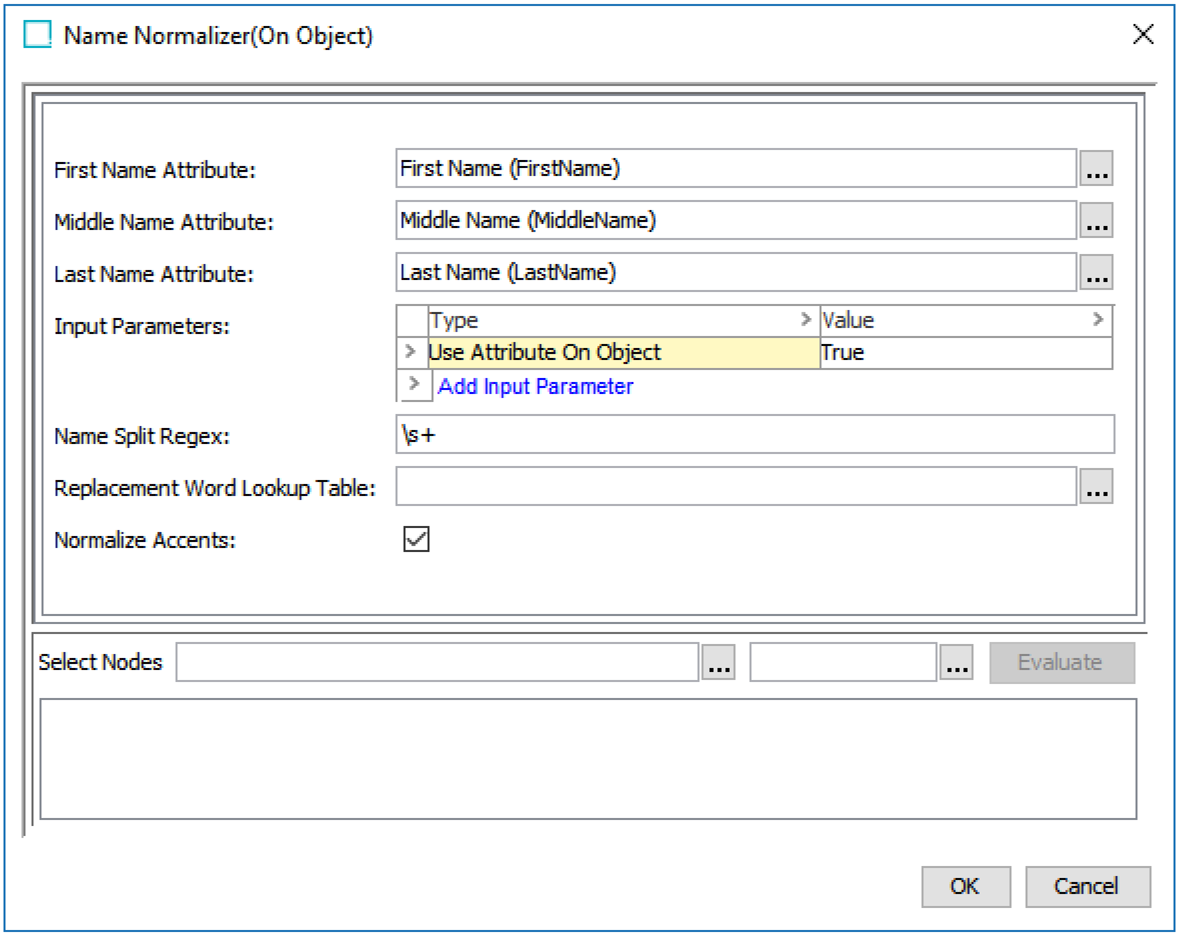
   This is used to account for inconsistencies in organization names by defining semantically equivalent strings (especially the usage of apostrophes and quotations). For example, normalizing ''s' to be 's' would change the organization name ACME's into ACMEs. Normalizing 'n' to be and would change the organization name ACME'n'SON to ACME and SON.
5. For the Name Split Regex parameter, enter the RegEx used to split the value of the Organization Name Attribute into words.
6. For the Replacement Word Lookup Table parameter, click the ellipsis button (...) and select a lookup table.

   Typically, this is used to account for the inconsistent use of common words in organization names. For example, '&' can be replaced by 'and.'
7. Click OK when finished.

Person Name Normalizer

A Person Name Normalizer can normalize customer name data for use in the corresponding Person Name Matcher.

1. To normalize customer name data, click the ellipsis button (...) in the Data column to access the configuration.



Name Normalizer(On Object)

First Name Attribute: First Name (FirstName)

Middle Name Attribute: Middle Name (MiddleName)

Last Name Attribute: Last Name (LastName)

Input Parameters:

Type	Value
Use Attribute On Object	True

Name Split Regex: \s+

Replacement Word Lookup Table:

Normalize Accents: ☑

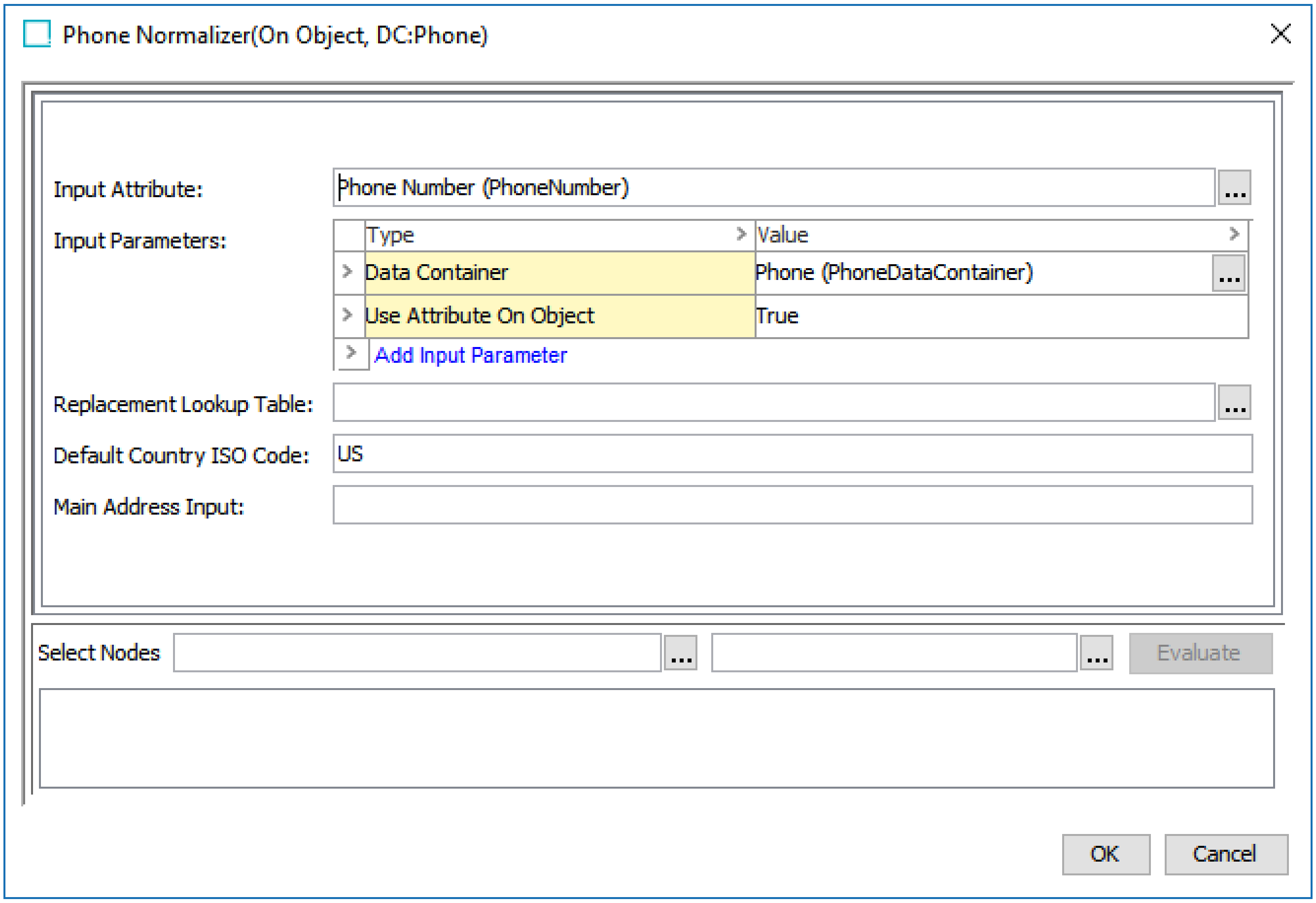
2. For the First Name Attribute parameter, click the ellipsis button (...) and browse or search for a first name attribute to normalize. Repeat this step for the Middle Name Attribute and Last Name Attribute parameters.
3. For Input Parameters, click the Add Input Parameter link, and in the Add Input Parameter popup dialog, select the input type from the dropdown.
   - Input Normalizer: Enter the ID of another Person Name Normalizer to use its output. Typically, this normalizer is written in JavaScript.
   - Data Container: When you select 'Data Container,' click the ellipsis button (...) and browse or search for a data container. The selected data container will have its name data normalized when used in a matcher.
   - Use Attribute on Object: When you select 'Use Attribute On Object,' click the dropdown and select 'True.' This input parameter will normalize the name attributes mapped to the First Name Attribute, Middle Name Attribute, and Last Name Attribute fields, and output them as one value. This input parameter will be configured by default.
4. For the Name Split Regex parameter, enter the RegEx used to split the First Name, Middle Name, and Last Name values into words.
5. For the Replacement Word Lookup Table parameter, click the ellipsis button (...) and select a lookup table. Typically, this is used to remove unwanted words from names. For example, 'Mr.,' 'Dr.,' or 'Von.'

6. Check the Normalizer Accents checkbox if accented characters should be normalized.
7. Click OK when finished.

Phone Normalizer

A Phone Normalizer can normalize phone data for use in the corresponding Phone Matcher.

1. To normalize customer phone data, click the ellipsis button (...) in the Data column to access the configuration.



Phone Normalizer(On Object, DC:Phone)

Input Attribute: Phone Number (PhoneNumber)

Input Parameters:

Type	Value
Data Container	Phone (PhoneDataContainer)
Use Attribute On Object	True

Add Input Parameter

Replacement Lookup Table:

Default Country ISO Code: US

Main Address Input:

Select Nodes

2. For the Input Attribute parameter, click the ellipsis button (...) and browse or search for a phone attribute to normalize.
3. For Input Parameters, click the Add Input Parameter link, and in the Add Input Parameter popup dialog, select the input type from the dropdown.

    - Input Normalizer: Enter the ID of another Phone Normalizer to use its output. Typically, this normalizer is written in JavaScript.
    - Data Container: When you select 'Data Container,' click the ellipsis button (...) and browse or search for a data container. The selected data container will have its phone data normalized when used in a matcher.

- Use Attribute on Object: When you select 'Use Attribute On Object,' click the dropdown and select 'True.' This input parameter will normalize the phone attribute mapped to the Input Attribute field. This input parameter will be configured by default.

4. For the Replacement Lookup Table parameter, click the ellipsis button (...) and select a lookup table. Typically, this is used to remove invalid phone values.
5. For the Default Country ISO Code parameter, enter a two-letter ISO code string.
6. For the Main Address Input parameter, enter the ID of an Address Normalizer. The Country ISO Code value of the normalizer output is used in place of the Default ISO Code, if one exists.

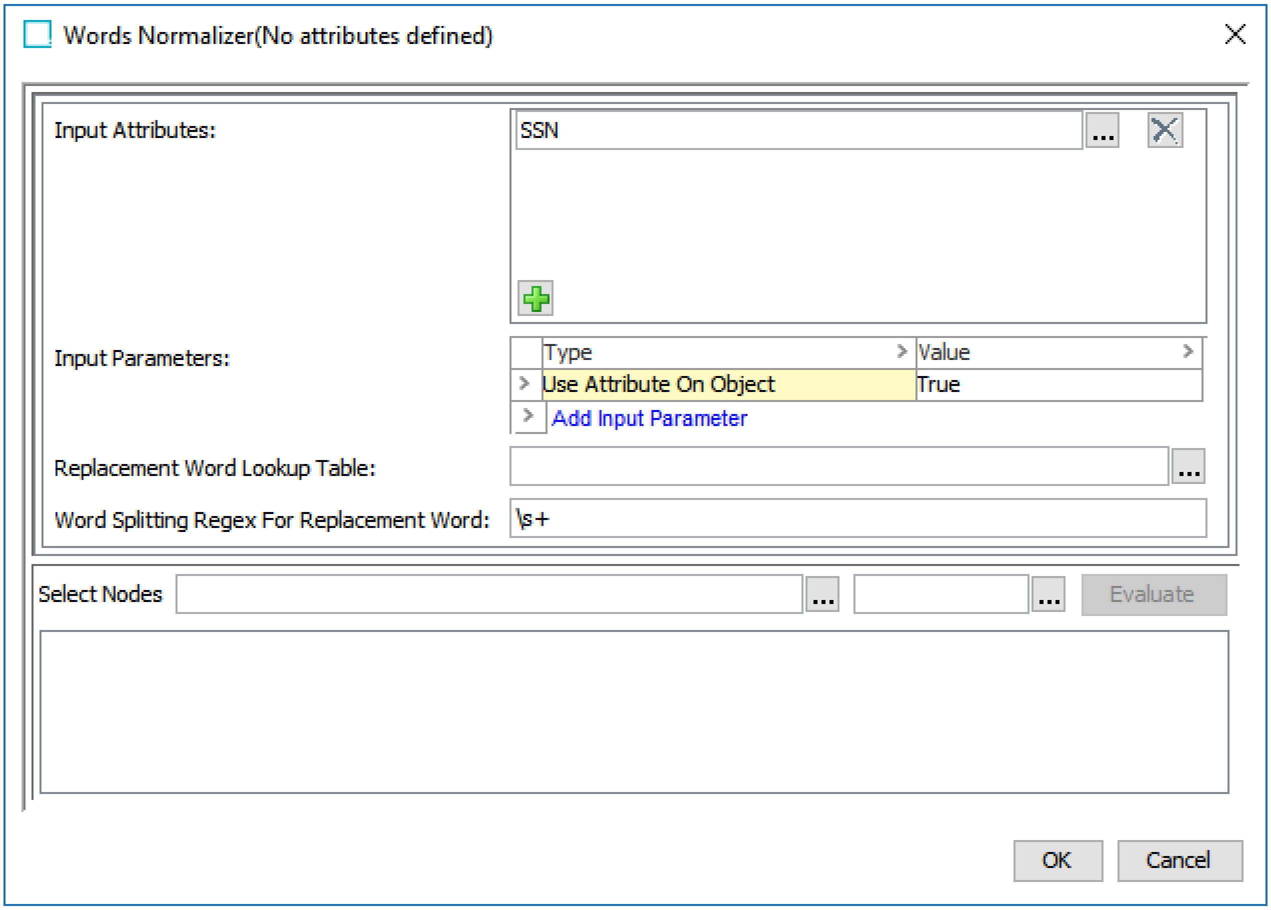
> **Note:** To use this parameter, write a JavaScript address normalizer that outputs a Country ISO code.

7. Click OK when finished.

Words Normalizer

A Words Normalizer can normalize attribute data for use in the corresponding Words Matcher. Multiple attributes can be mapped to the same normalizer. When the corresponding Words Matcher is applied, all mapped attributes will be evaluated.

1. To add a words normalizer, click the ellipsis button (...) in the Data column to access the configuration.



Words Normalizer(No attributes defined)

Input Attributes: SSN

Input Parameters:

Type	Value
Use Attribute On Object	True

Replacement Word Lookup Table:

Word Splitting Regex For Replacement Word: \s+

2. For the Input Attribute parameter, click the plus sign (➕) to create a new entry, then click the ellipsis button (...) and browse or search for attributes whose values should be normalized.
3. For Input Parameters, click the Add Input Parameter link, and in the Add Input Parameter popup dialog, select the input type from the dropdown.
   - Input Normalizer: Enter the ID of another Words Normalizer to use its output. Typically, this normalizer is written in JavaScript.
   - Data Container: When you select 'Data Container,' click the ellipsis button (...) and browse or search for a data container.
   - Use Attribute on Object: When you select 'Use Attribute On Object,' click the dropdown and select 'True.' This input parameter will normalize the attribute mapped to the Input Attribute field. This input parameter will be configured by default.
4. For the Replacement Word Lookup Table parameter, click the ellipsis button (...) and select a lookup table. Typically, this is used to remove invalid values.
5. For the Word Splitting Regex For Replacement Word parameter, enter the RegEx used to split the attribute values into individual words.
6. Click OK when finished.

Customer Data JavaScript Normalizers

For especially complicated solutions, it is possible to expand the capabilities of a customer data normalizer via JavaScript. These JavaScript normalizers can be written to input the normalized values of basic customer data normalizer(s) and manipulate the data in ways the standard normalizer could not. In other words, a JavaScript normalizer inputs a set of strings / values from a basic customer data normalizer and outputs a new set of strings / values.

Note: The JavaScript normalizer should output a completely new set of strings / values, and should not overwrite existing strings / values.

These normalizers can also be built completely from scratch rather than enhancing an existing customer data normalizer.

In many situations the more complex JavaScript normalizer would be cited by a corresponding matcher, rather than the basic customer data normalizer.

A typical customer data JavaScript normalizer complies with the following steps:

1. Uses the evaluate function on a Match Expression Context to retrieve the output of a desired normalizer.
2. Uses an iterator to access the set of values / strings.
3. Uses a builder pattern to create new values / strings from the iterated data.
4. Inserts the new values / strings into something that can be iterated, such as a set, and returns that set.

Match Criteria Matchers

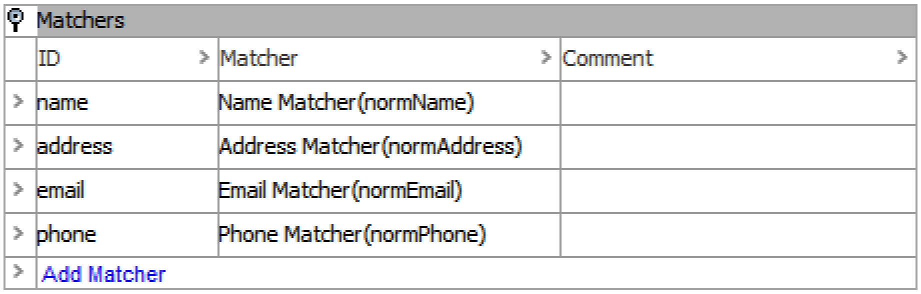
A matcher compares the values of a data element from the two records being compared by the matching algorithm and produces a match score. Many matchers allow setting a default threshold for what is considered 'True' or 'False' in a match rule condition.

Many matchers come with different weights and metrics, allowing detailed calibration of the algorithm to specific datasets. It is significant to do this calibration during match tuning.

The following matchers are available:

- Standard Matchers
  - Business Function Matcher
  - STEP Function Matcher
  - JavaScript Function Matcher
- Party Data Matchers
  - Address Matcher
  - Email Matcher
  - Organization Name Matcher
  - Person Name Matcher
  - Phone Matcher
  - Words Matcher

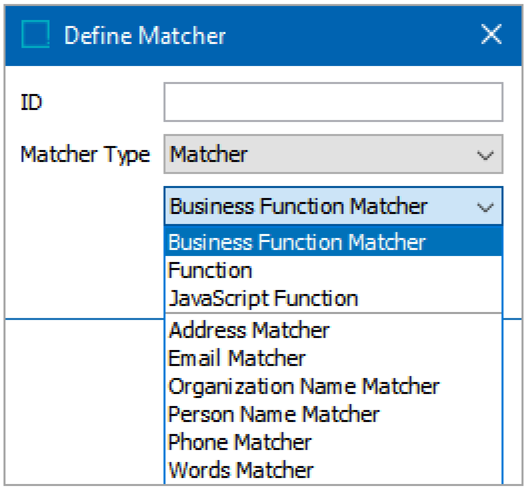
Configuration



Matchers

ID	Matcher	Comment
name	Name Matcher(normName)	
address	Address Matcher(normAddress)	
email	Email Matcher(normEmail)	
phone	Phone Matcher(normPhone)	
Add Matcher		

1. To add a matcher to the table, click the 'Add Matcher' link.
2. In the Define Matcher dialog, enter an ID for the matcher and use the Matcher Type dropdown(s) to define the match type, then click the 'Add Matcher' button.



Define Matcher

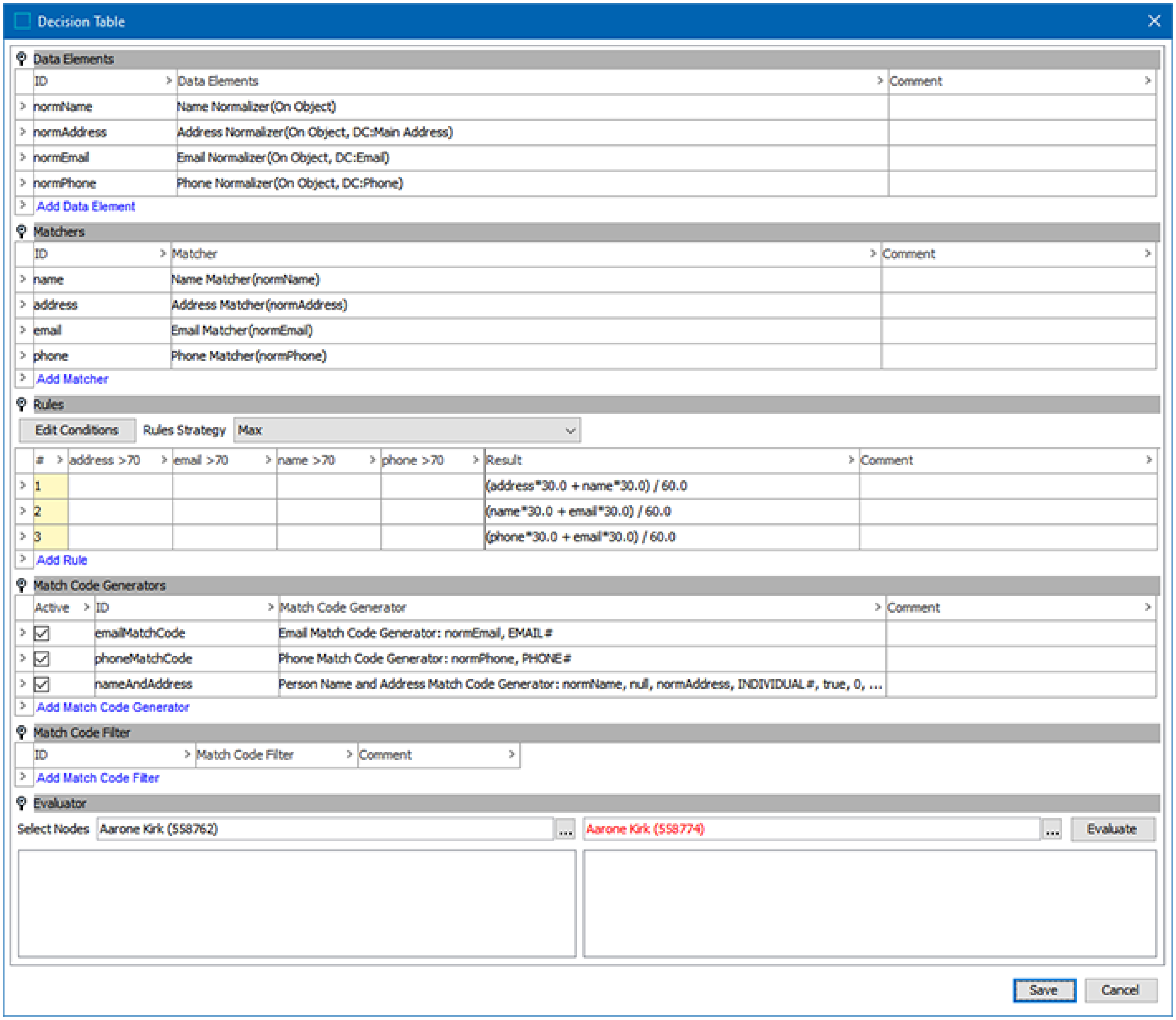
ID

Matcher Type: Matcher

Business Function Matcher

Important: No two data element, matchers, or match code generators should have the same ID.

3. Populate the Matcher column of the table:
   - For matchers, click the ellipsis button (**...**) to access the configuration. Configuration steps vary depending on the type of matcher selected. For more information on these matchers, see the Standard Matchers and Customer Data Matchers sections below.



Data Elements

ID	Data Elements	Comment
normName	Name Normalizer(On Object)	
normAddress	Address Normalizer(On Object, DC:Main Address)	
normEmail	Email Normalizer(On Object, DC:Email)	
normPhone	Phone Normalizer(On Object, DC:Phone)	

Matchers

ID	Matcher	Comment
name	Name Matcher(normName)	
address	Address Matcher(normAddress)	
email	Email Matcher(normEmail)	
phone	Phone Matcher(normPhone)	

Rules Strategy: Max

#	address >70	email >70	name >70	phone >70	Result	Comment
1					(address*30.0 + name*30.0) / 60.0	
2					(name*30.0 + email*30.0) / 60.0	
3					(phone*30.0 + email*30.0) / 60.0	

Match Code Generators

Active	ID	Match Code Generator	Comment
☑	emailMatchCode	Email Match Code Generator: normEmail, EMAIL#	
☑	phoneMatchCode	Phone Match Code Generator: normPhone, PHONE#	
☑	nameAndAddress	Person Name and Address Match Code Generator: normName, null, normAddress, INDIVIDUAL#, true, 0, ...	

Match Code Filter

ID	Match Code Filter	Comment

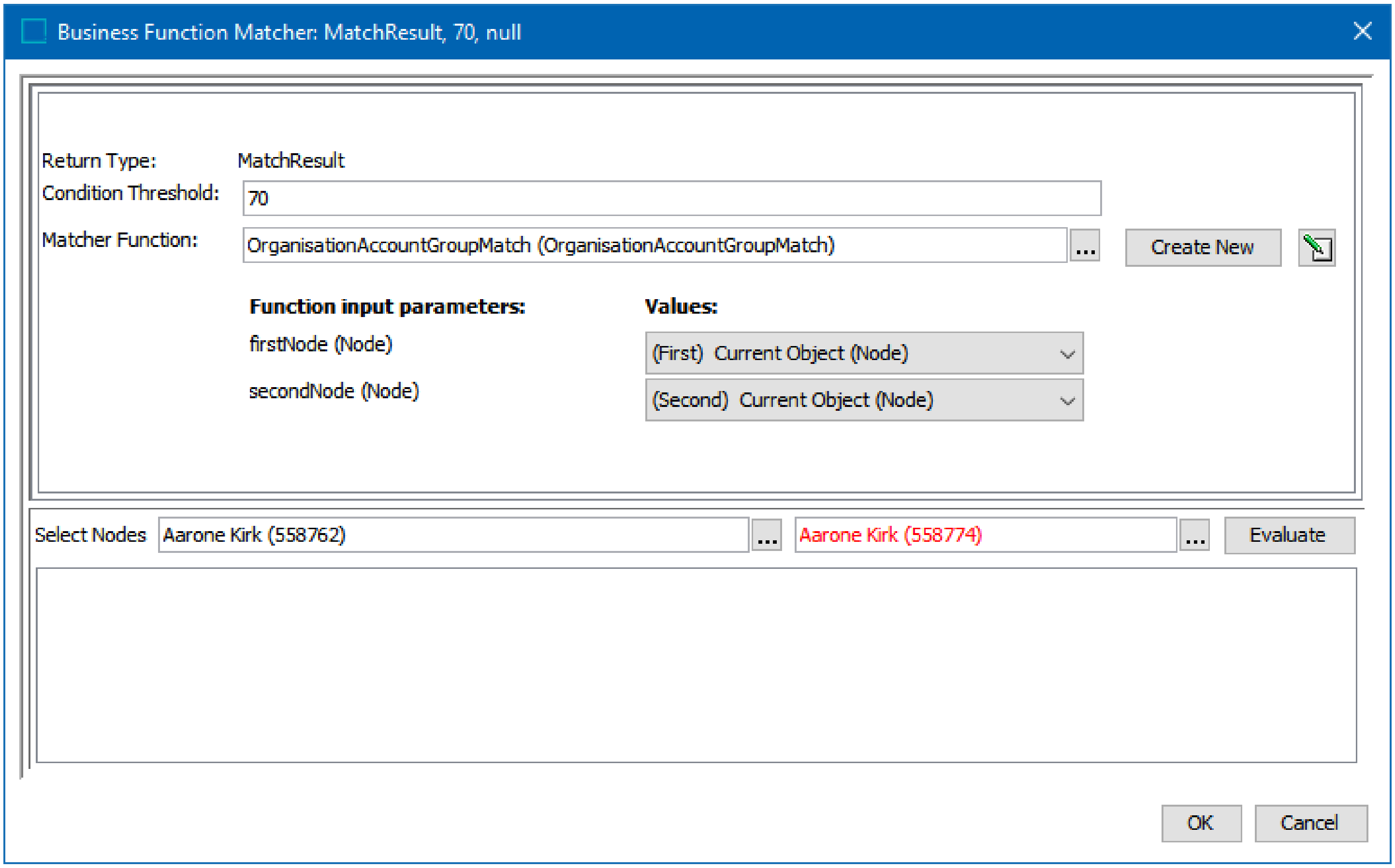
Select Nodes: Aarone Kirk (558762) ... Aarone Kirk (558774) ...

Standard Matchers

When created as a JavaScript or STEP function, 'mcevaluate' and 'evaluate' are used to assess elements from the data and matcher sections of the decision table, and compare their results.

Business Function Matcher

The Business Function Matcher uses a business function to return a match score. The business function is typically written in JavaScript.



Business Function Matcher: MatchResult, 70, null

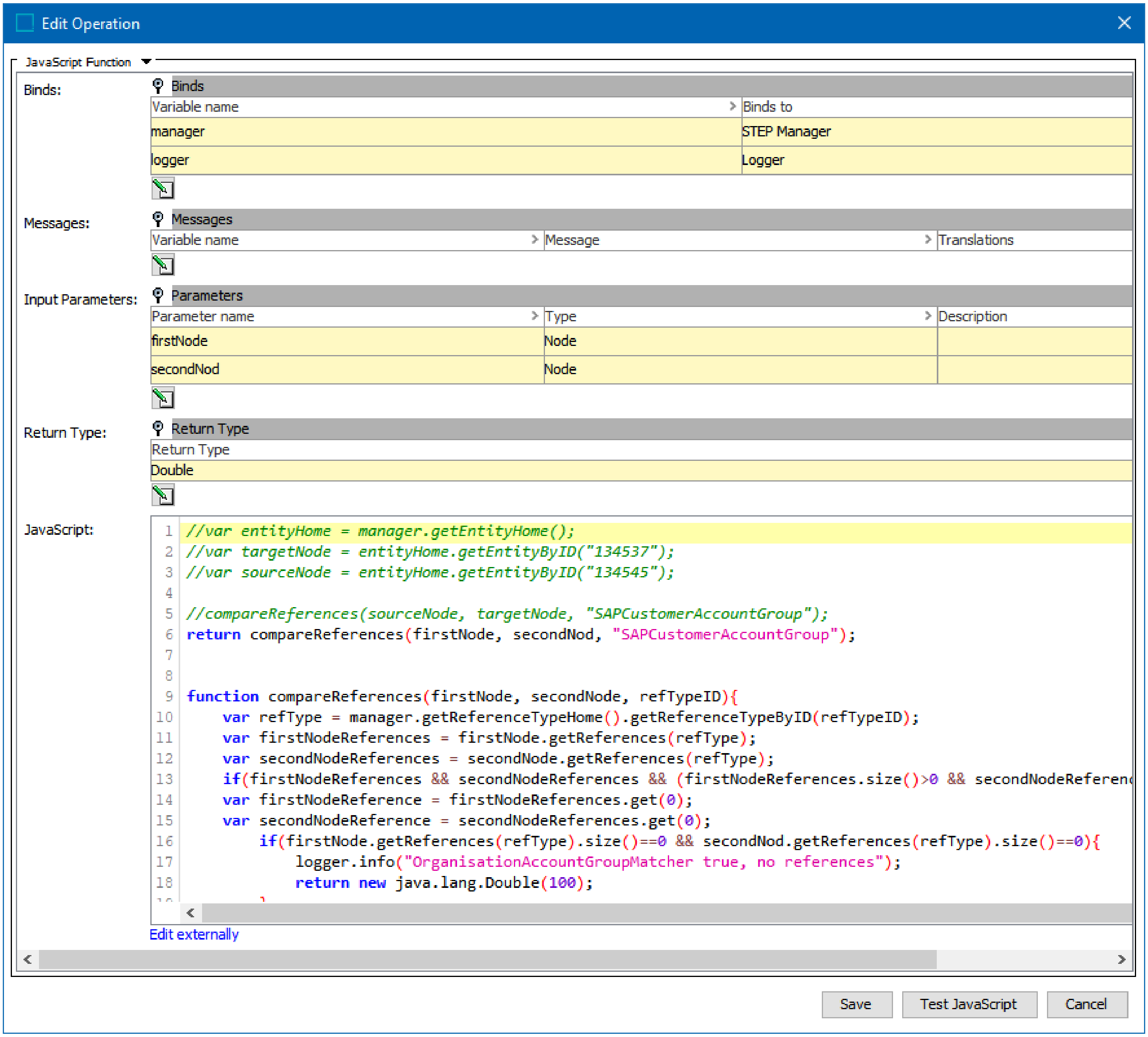
Return Type: MatchResult

Condition Threshold: 70

Matcher Function: OrganisationAccountGroupMatch (OrganisationAccountGroupMatch)

Function input parameters:	Values:
firstNode (Node)	(First) Current Object (Node)
secondNode (Node)	(Second) Current Object (Node)

Select Nodes: Aarone Kirk (558762) | Aarone Kirk (558774)



Variable name	Binds to
manager	STEP Manager
logger	Logger

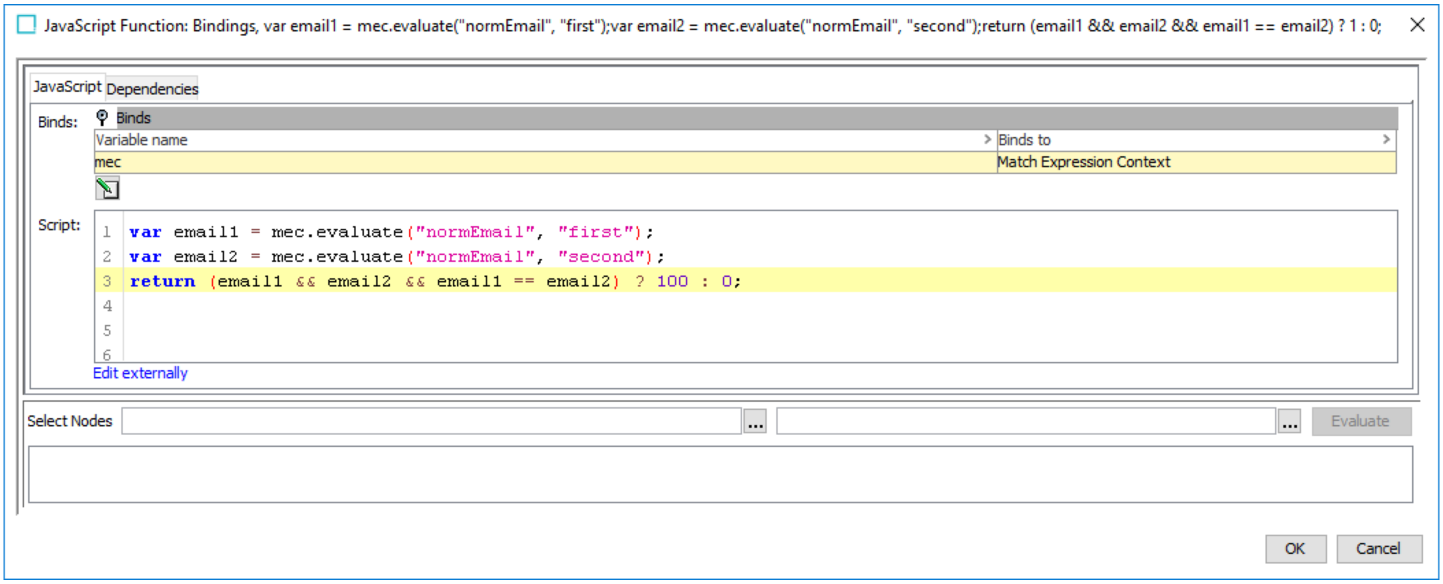
Variable name	Message	Translations

Parameter name	Type	Description
firstNode	Node	
secondNod	Node	

Return Type
Double

```
//var entityHome = manager.getEntityHome();
//var targetNode = entityHome.getEntityByID("134537");
//var sourceNode = entityHome.getEntityByID("134545");

//compareReferences(sourceNode, targetNode, "SAPCustomerAccountGroup");
return compareReferences(firstNode, secondNod, "SAPCustomerAccountGroup");


function compareReferences(firstNode, secondNode, refTypeID){
    var refType = manager.getReferenceTypeHome().getReferenceTypeByID(refTypeID);
    var firstNodeReferences = firstNode.getReferences(refType);
    var secondNodeReferences = secondNode.getReferences(refType);
    if(firstNodeReferences && secondNodeReferences && (firstNodeReferences.size()>0 && secondNodeReferen
    var firstNodeReference = firstNodeReferences.get(0);
    var secondNodeReference = secondNodeReferences.get(0);
        if(firstNode.getReferences(refType).size()==0 && secondNod.getReferences(refType).size()==0){
            logger.info("OrganisationAccountGroupMatcher true, no references");
            return new java.lang.Double(100);
```

JavaScript Function Matcher



JavaScript Function: Bindings, var email1 = mec.evaluate("normEmail", "first");var email2 = mec.evaluate("normEmail", "second");return (email1 && email2 && email1 == email2) ? 1 : 0;

Binds:

Variable name	Binds to
mec	Match Expression Context

Script:

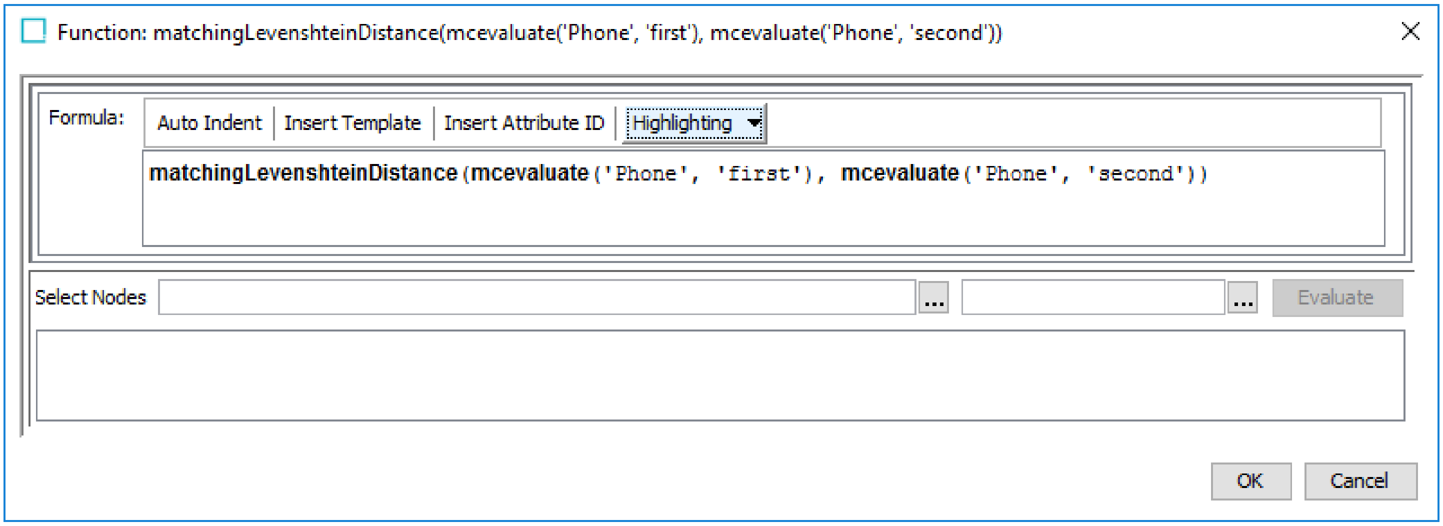
```
var email1 = mec.evaluate("normEmail", "first");
var email2 = mec.evaluate("normEmail", "second");
return (email1 && email2 && email1 == email2) ? 100 : 0;
```

This JavaScript matcher above implements a basic email matcher that performs a plain comparison of the emails by comparing normalized email addresses as text strings.

> **Note:** The matcher does not deal with any special cases such as where the normalizer returns strings that are obviously not emails, like empty strings. Resolving such cases is expected to be handled by the normalizer.

STEP Function Matcher

The STEP Function Matcher uses the language of calculated attributes to produce the match score.



Function: matchingLevenshteinDistance(mcevaluate('Phone', 'first'), mcevaluate('Phone', 'second'))

Formula:

```
matchingLevenshteinDistance(mcevaluate('Phone', 'first'), mcevaluate('Phone', 'second'))
```

Customer Data Matchers

These matcher templates are intended for use in customer data solutions:

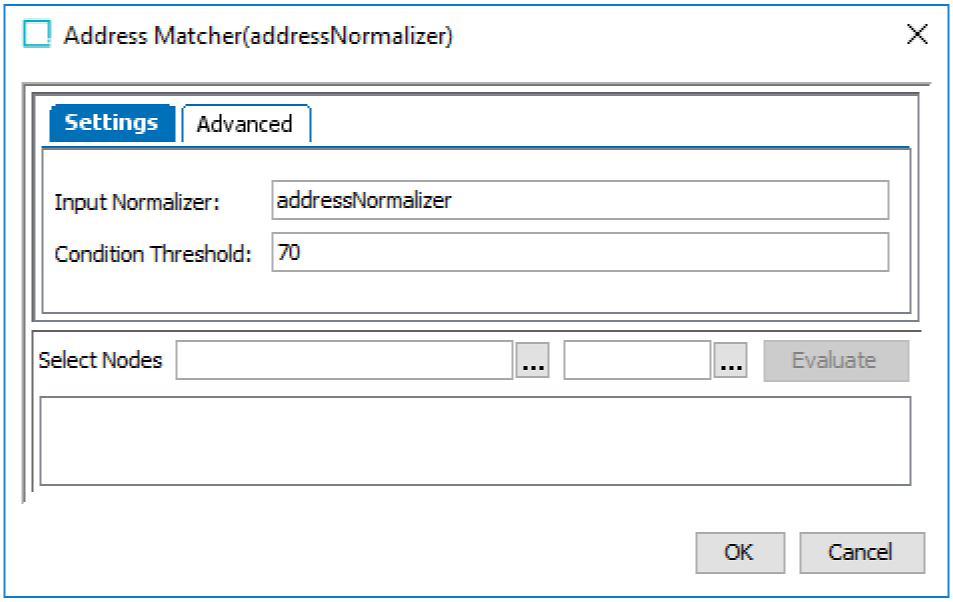
- Address Matcher
- Email Matcher
- Organization Name Matcher
- Person Name Matcher
- Phone Matcher
- Words Matcher

Some matchers give access to lookup tables. For more information on lookup tables, see the **Transformation Lookup Tables** topic in **Resource Materials** documentation.

Address Matcher

The Address Matcher compares the normalized address data of two objects and outputs a rank score based on the weighted sum of relevant data elements and match factors. When applied to a rule, the resulting rank score is evaluated against a condition threshold, and returns 'True' if it meets or exceeds the minimum requirement of the threshold. If the combination of street and postcode, or street and city is a match, the Address Matcher will return a rank score indicating a match.

The Address Matcher configuration is split into two tabs: Settings, where the corresponding normalizer is mapped and the condition threshold is established, and Advanced, where different weights are applied to the relevant data elements and match factors.



Input Normalizer: addressNormalizer

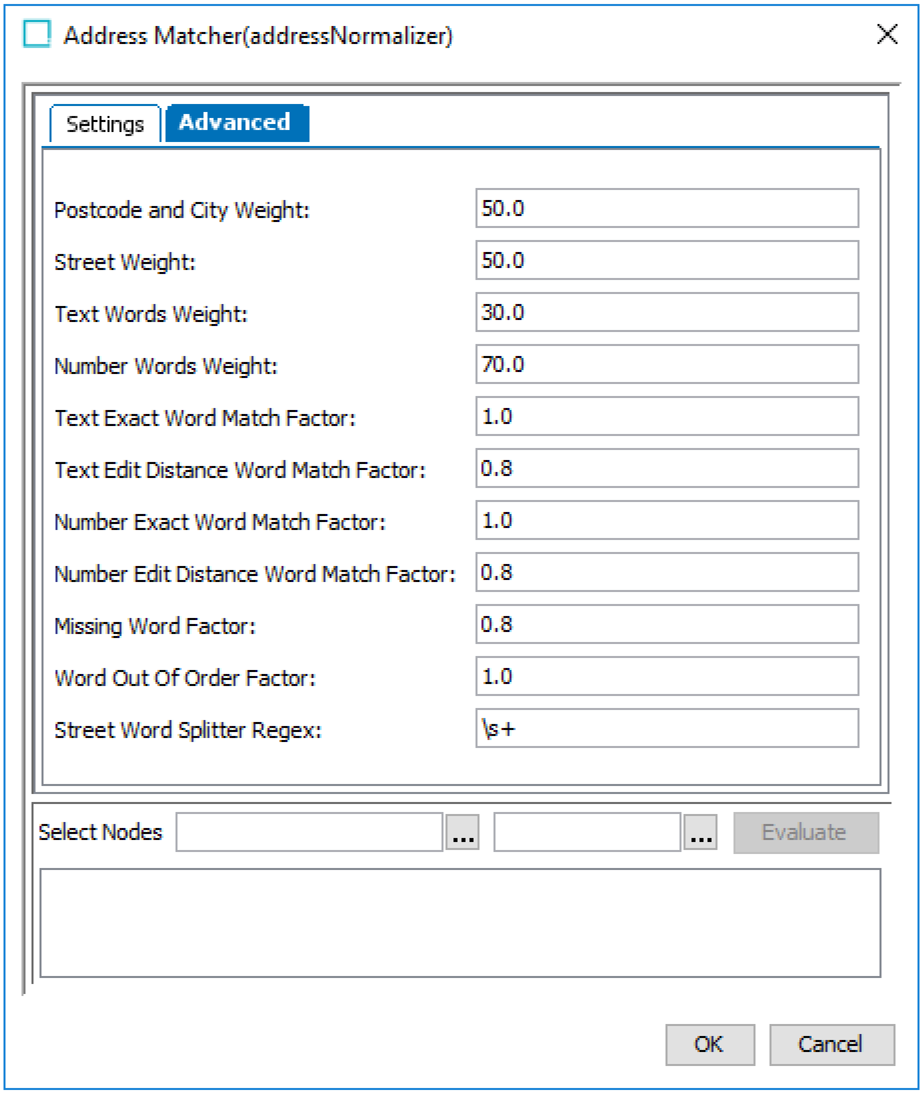
Condition Threshold: 70

1. To configure a matcher for customer address data, click the ellipsis button (...) in the Matcher column to access the configuration.

2. The Address Matcher configuration dialog will open in the 'Settings' tab.
3. In the Input Normalizer parameter, enter the ID of the address normalizer the matcher applies to. This field is case sensitive.
4. In the Condition Threshold parameter, enter the minimum score a matcher must achieve in order to return 'True' on a decision table rule. By default this score is set to '70.'

Note: An empty Condition Threshold should be used if a variable threshold is required between different rules. For example, one rule requires the matcher to return a score greater than '70,' and another rule needs it to be greater than '75.'

5. Navigate to the 'Advanced' tab. All but one of the parameters included on this tab require that a weight be defined. The matcher considers the individual weights of these elements when they are factored together for the rank score.



Postcode and City Weight:	50.0
Street Weight:	50.0
Text Words Weight:	30.0
Number Words Weight:	70.0
Text Exact Word Match Factor:	1.0
Text Edit Distance Word Match Factor:	0.8
Number Exact Word Match Factor:	1.0
Number Edit Distance Word Match Factor:	0.8
Missing Word Factor:	0.8
Word Out Of Order Factor:	1.0
Street Word Splitter Regex:	\s+

A few things to note:

- The final score is a weighted sum of street and postcode, or street and city.
- The value of Street is split into individual words based on the Street Word Splitter Regex.
- The words for the Street value are split between numbers and text, and are compared separately.
- Text words are paired up using Exact, Metaphone 3, and Edit Distance. Text words that are not paired are handled as Missing words.
- Number words are paired up using Exact and Edit Distance. Number words that are not paired are handled as Missing Words.

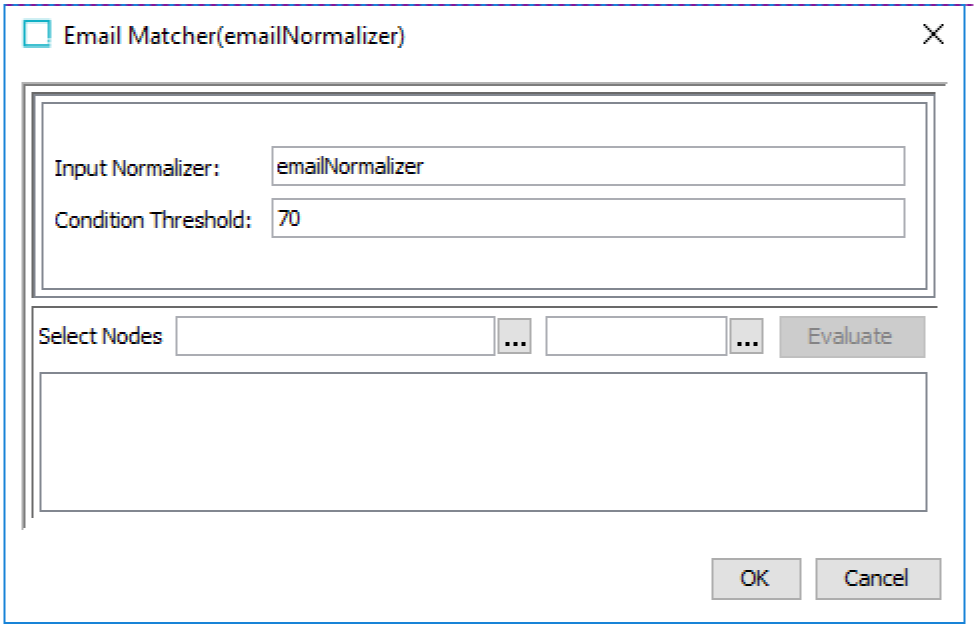
Required parameters include:

- **Postcode and City Weight**: The relative weight of the Postcode / City score versus the Street score.
- **Street Weight**: The relative weight of the Street score versus the Postcode / City. The Street score is a weighted sum of the Number Words score and the Text Words score
- **Text Word Weight**: The relative weight of the Text Words score versus the Number Words score.
- **Number Words Weight**: The relative weight of the Number Words score versus the Text Words score.
- **Text Exact Word Match Factor**: Determines how pairs that are exact matches should influence the final score.
- **Text Edit Distance Word Match Factor**: Determines how words that are paired via edit distance influence the final score.
- **Number Exact Word Match Factor**: Determines how pairs that are exact matches should influence the final score.
- **Number Edit Distance Word Match Factor**: Determines how words that are paired via edit distance influence the final score.
- **Missing Word Factor**: Determines how much unpaired / missing words should penalize the final result.
- **Word Out Of Order Factor**: Determines how much words that appear out of order should penalize the final result.

6. In the Street Word Splitter Regex parameter, enter the RegEx used to split the Street value into words.
7. Click OK when finished.

Email Matcher

The Email Matcher compares the normalized email data of two objects and outputs a rank score. When applied to a rule, the resulting rank score is evaluated against a condition threshold, and returns 'True' if it meets or exceeds the minimum requirement of the threshold. If the email values are a match, the email matcher will return a rank score indicting a match.



Input Normalizer: emailNormalizer

Condition Threshold: 70

1. To configure a matcher for customer email data, click the ellipsis button (...) in the Matcher column to access the configuration.
2. In the Input Normalizer parameter, enter the ID of the email normalizer this matcher applies to. This field is case sensitive.
3. In the Condition Threshold parameter, enter the minimum score a matcher must achieve in order to return 'True' on a decision table rule. By default this score is set to '70.'

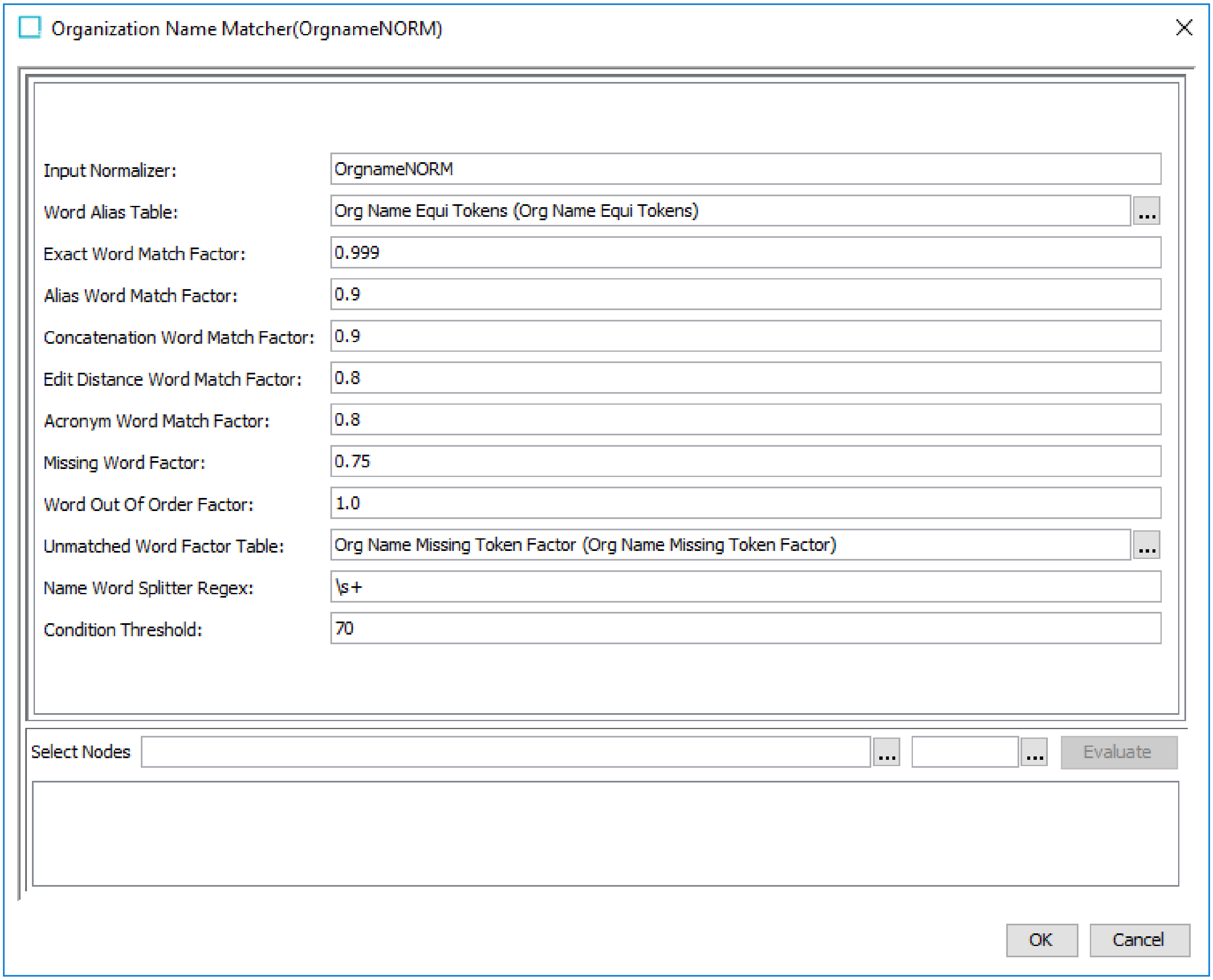
> **Note:** An empty Condition Threshold should be used if a variable threshold is required between different rules. For example, one rule requires the matcher to return a score greater than '70,' and another rule needs it to be greater than '75.'

4. Click OK when finished.

Organization Name Matcher

The Organization Name Matcher compares the normalized organization name data of two objects and outputs a rank score based on the weighted sum of relevant data elements and match factors. When applied to a rule, the resulting rank score is evaluated against a condition threshold, and returns 'True' if it meets or exceeds the minimum requirement of the threshold. If the organization name values are a match, the organization name matcher will return a rank score indicting a match.

1. To configure a matcher for organization name data, click the ellipsis button (...) in the Matcher column to access the configuration.



Organization Name Matcher(OrgnameNORM)

Input Normalizer:	OrgnameNORM
Word Alias Table:	Org Name Equi Tokens (Org Name Equi Tokens)
Exact Word Match Factor:	0.999
Alias Word Match Factor:	0.9
Concatenation Word Match Factor:	0.9
Edit Distance Word Match Factor:	0.8
Acronym Word Match Factor:	0.8
Missing Word Factor:	0.75
Word Out Of Order Factor:	1.0
Unmatched Word Factor Table:	Org Name Missing Token Factor (Org Name Missing Token Factor)
Name Word Splitter Regex:	\s+
Condition Threshold:	70

2. In the Input Normalizer parameter, enter the ID of the organization name normalizer this matcher applies to. This field is case sensitive.
3. In the Word Alias Table parameter, click the ellipsis button (...) and select a lookup table to use for substituting certain words. This substitution takes place after the string has been cut into individual words via the Splitter Regex. This parameter should be used to match words that have the same or similar meaning. For example, legal terms such as inc and incorporated.
4. In the Exact Word Match Factor parameter, determine how pairs that are exact matches should influence the final score.
5. In the Alias Word Match Factor parameter, determine how words paired together via aliases should influence the final score.

6. In the Concatenation Word Match Factor parameter, determine how concatenated organization names paired with non-concatenated organization names impact the final score. For example, this match factor could be configured to match ACME Systems with ACMESystems.
7. In the Edit Distance Word Match Factor parameter, determine how words that are paired via edit distance influence the final score. Typically this match factor is used to catch spelling errors.
8. In the Acronym Word Match Factor parameter, determine how an organization name paired together based off of an acronym influences the final score. For example, the acronym ACME America Inc. could be matched with the organization name Advanced Cellular Medical Engineering of America.
9. In the Missing Word Factor parameter, determine how much unpaired / missing words should penalize the final result.
10. In the Word Out Of Order Factor parameter, determine how much words that appear out of order should penalize the final result.
11. In the Unmatched Word Factor Table parameter, click the ellipsis button (...) and select the relevant lookup table.

    > **Note:** The Unmatched Word Factor Table is a lookup table that assigns factors to certain words.

    By default, unpaired / missing words will penalize the final score using the Missing Word Factor parameter. However, if an unpaired / missing word appears in this table, it will penalize the final score using the factor in the table rather than the factor configured in Missing Word Factor parameter. This can be used to reduce or increase the penalty that certain words, depending on their significance, have on the final score if they are missing.

12. In the Name Word Splitter Regex parameter, enter the RegEx used to split the organization name value into words.
13. In the Condition Threshold parameter, enter the minimum score a matcher must achieve in order to return 'True' on a decision table rule. By default this score is set to '70.'

    > **Note:** An empty Condition Threshold should be used if a variable threshold is required between different rules. For example, one rule requires the matcher to return a score greater than '70,' and another rule needs it to be greater than '75.'

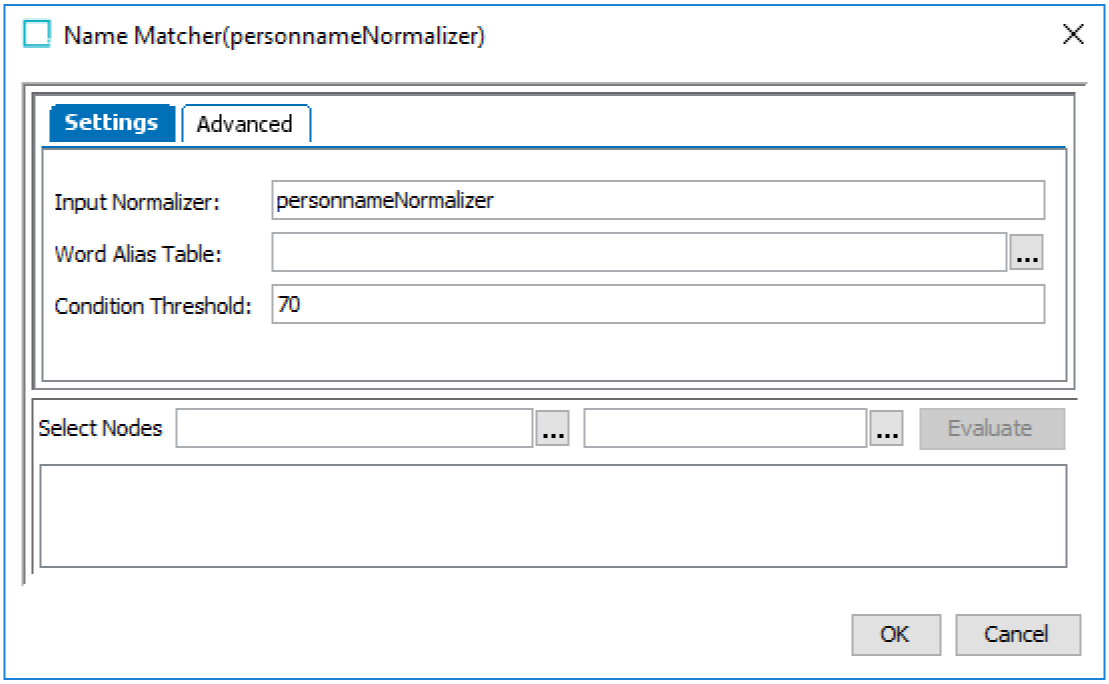
14. Click OK when finished.

Person Name Matcher

The Person Name Matcher compares the normalized name data of two objects and outputs a rank score based on the weighted sum of relevant data elements and match factors. When applied to a rule, the resulting rank score is evaluated against a condition threshold, and returns 'True' if it meets or exceeds the minimum requirement of the threshold. If the combination of first name and middle name, and middle name and last name is a match, the Person Name Matcher will return a rank score indicating a match.

> **Note:** If customer names are represented in a single field rather than split into first name and last name, use the Words Normalizer / Matcher instead. Middle name is not required to use this matcher.

The Person Name Matcher configuration is split into two tabs: Settings, where the corresponding normalizer is mapped and the condition threshold is established, and Advanced, where different weights are applied to the relevant data elements and match factors.



Input Normalizer: personnameNormalizer

Word Alias Table:

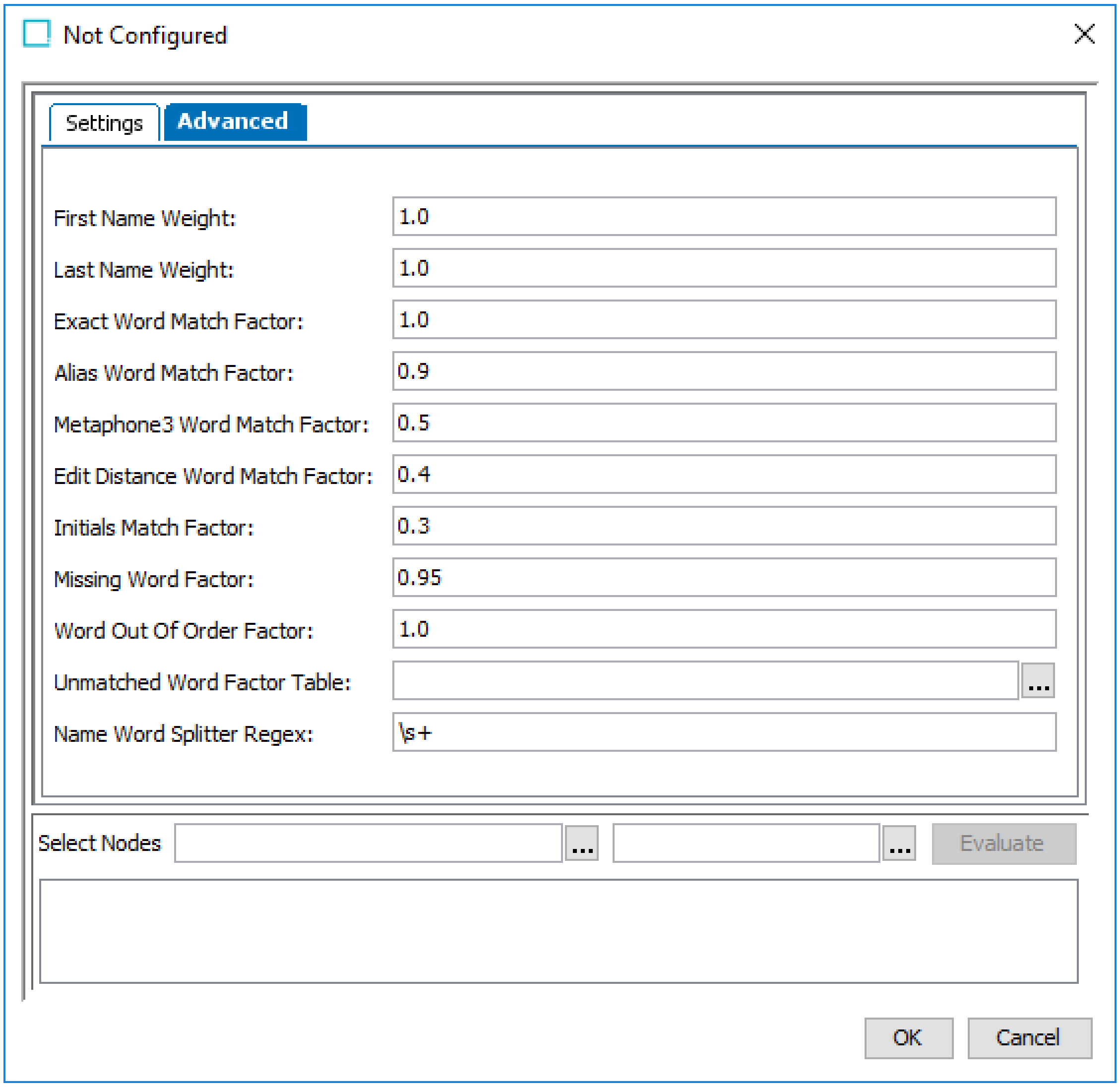
Condition Threshold: 70

1. To configure a matcher for customer name data, click the ellipsis button (...) in the Matcher column to access the configuration. The Person Name Matcher configuration dialog will open in the 'Settings' tab.
2. In the Input Normalizer parameter, enter the ID of the person name normalizer this matcher applies to. This field is case sensitive.
3. In the Word Alias Table parameter, click the ellipsis button (...) and select a lookup table to use for substituting certain words. This substitution takes place after the string has been cut into individual words via the Splitter Regex.
4. In the Condition Threshold parameter, enter the minimum score a matcher must achieve in order to return 'True' on a decision table rule. By default this score is set to '70.'

   > **Note:** An empty Condition Threshold should be used if a variable threshold is required between different rules. For example, one rule requires the matcher to return a score greater than '70,' and another rule needs it to be greater than '75.'

5. Navigate to the 'Advanced' tab. All but two of the parameters included on this tab require that a weight be defined. The matcher considers the individual weights of these elements when they are factored together for

the rank score.



First Name Weight: 1.0
Last Name Weight: 1.0
Exact Word Match Factor: 1.0
Alias Word Match Factor: 0.9
Metaphone3 Word Match Factor: 0.5
Edit Distance Word Match Factor: 0.4
Initials Match Factor: 0.3
Missing Word Factor: 0.95
Word Out Of Order Factor: 1.0
Unmatched Word Factor Table:
Name Word Splitter Regex: \s+

A couple things to note:

- The final score is a weighted sum of the combined first name and middle name, and the combined middle name and last name.
- The First Name, Middle Name, and Last Name values are split into individual words based on the Name Word Splitter Regex.

Required parameters include:

- **First Name Weight**: The relative weight of the first name / middle name score versus the middle name / last name score.
- **Last Name Weight**: The relative weight of the middle name / last name score versus the first name / middle name score.

- **Exact Word Match Factor**: Determines how pairs that are exact matches should influence the final score.
- **Alias Word Match Factor**: Determines how words paired together via aliases should influence the final score.
- **Metaphone3 Word Match Factor**: Determines how words paired together via Metaphone 3 should influence the final score.
- **Edit Distance Word Match Factor**: Determines how words that are paired via edit distance influence the final score.
- **Initials Match Factor**: Determines how words paired together via initials influence the final score.
- **Missing Word Factor**: Determines how much unpaired / missing words should penalize the final result.
- **Word Out Of Order Factor**: Determines how much words that appear out of order should penalize the final result.

7. In the Unmatched Word Factor Table parameter, click the ellipsis button (...) and select the relevant lookup table.

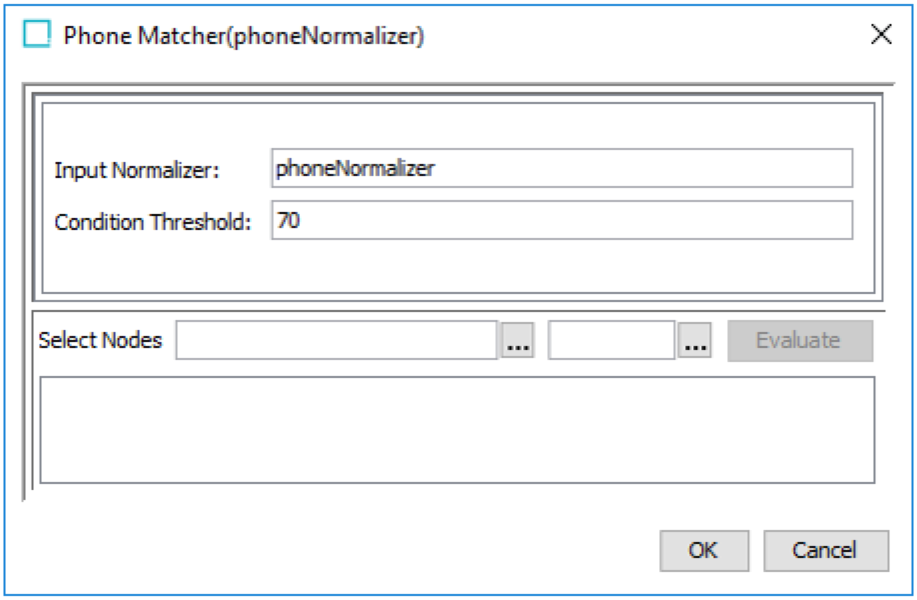
   Note: The Unmatched Word Factor Table is a lookup table that assigns factors to certain words.

   By default, unpaired / missing words will penalize the final score using the Missing Word Factor parameter. However, if an unpaired / missing word appears in this table, it will penalize the final score using the factor in the table rather than the factor configured in Missing Word Factor parameter. This can be used to reduce or increase the penalty that certain words, depending on their significance, have on the final score if they are missing.

8. In the Name Word Splitter Regex parameter, enter the RegEx used to split the first name, middle name, and last name values into words.
9. Click OK when finished.

Phone Matcher

The Phone Matcher compares the normalized phone data of two objects and outputs a rank score. When applied to a rule, the resulting rank score is evaluated against a condition threshold and returns 'True' if it meets or exceeds the minimum requirement of the threshold. If the phone values are a match, the Phone Matcher will return a rank score indicting a match.



Input Normalizer: phoneNormalizer

Condition Threshold: 70

1. To configure a matcher for customer phone data, click the ellipsis button (...) in the Matcher column to access the configuration.
2. In the Input Normalizer parameter, enter the ID of the phone normalizer this matcher applies to. This field is case sensitive.
3. In the Condition Threshold parameter, enter the minimum score a matcher must achieve in order to return 'True' on a decision table rule. By default this score is set to '70.'

   Note: An empty Condition Threshold should be used if a variable threshold is required between different rules. For example, one rule requires the matcher to return a score greater than '70,' and another rule needs it to be greater than '75.'

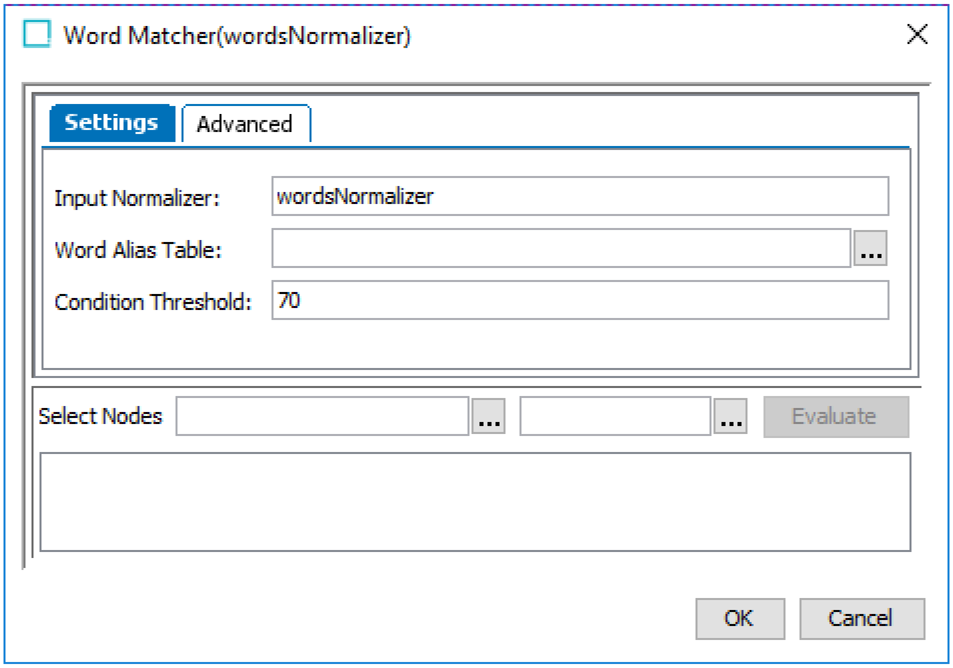
4. Click OK when finished.

Word Matcher

The Word Matcher compares the normalized words data of two objects and outputs a rank score based on the weighted sum of relevant data elements and match factors. When applied to a rule, the resulting rank score is evaluated against a condition threshold, and returns 'True' if it meets or exceeds the minimum requirement of the threshold. If the word values are a match, the Words Matcher will return a rank score indicting a match.

Note: The Word Normalizer / Matcher is a generic multi-word matcher that can represent a wide range of data, such as, customer names and social security numbers.

The Word Matcher configuration is split into two tabs: Settings, where the corresponding normalizer is mapped and the condition threshold is established, and Advanced, where different weights are applied to the relevant data elements and match factors.



Input Normalizer: wordsNormalizer

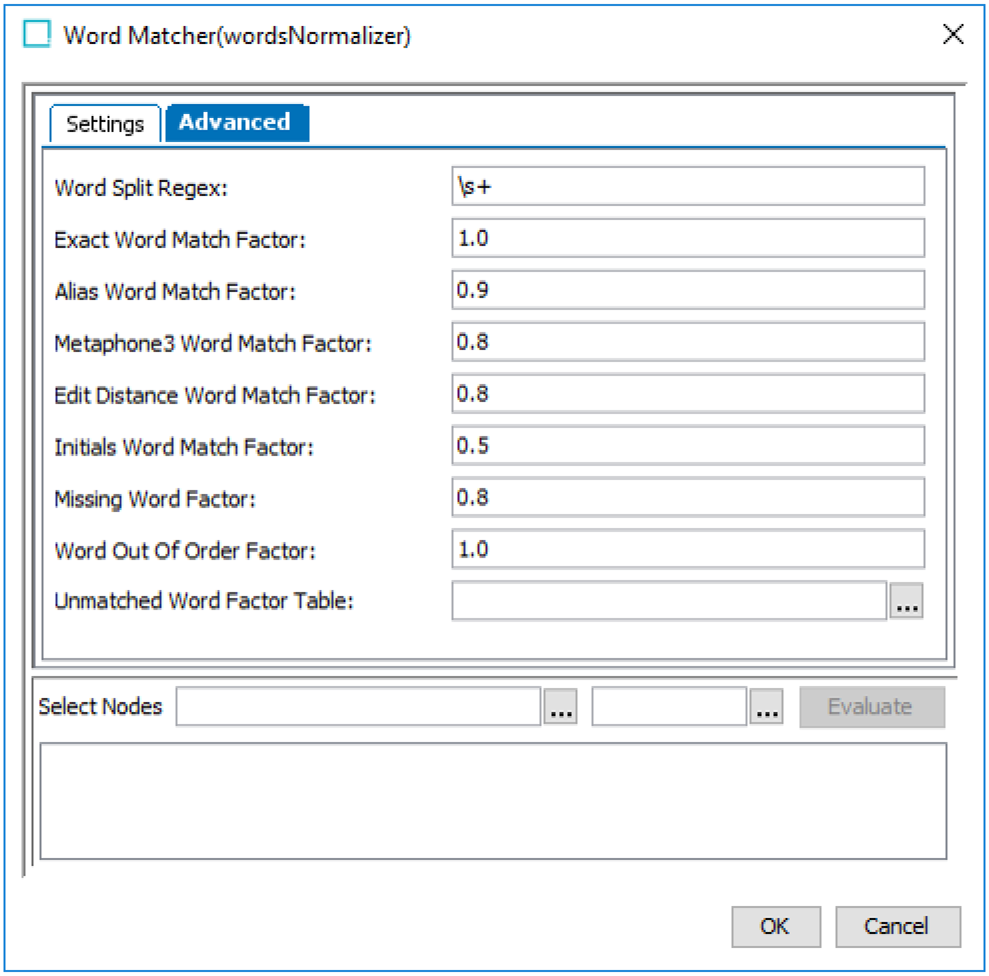
Word Alias Table:

Condition Threshold: 70

1. To configure a matcher for different word normalizers, click the ellipsis button (...) in the Matcher column to access the configuration. The Word Matcher configuration dialog will open in the 'Settings' tab.
2. In the Input Normalizer parameter, enter the ID of the words normalizer this matcher applies to. This field is case sensitive.
3. In the Word Alias Table parameter, click the ellipsis button (...) and select a lookup table to use for substituting certain words. This substitution takes place after the string has been cut into individual words via the Splitter Regex.
4. In the Condition Threshold parameter, enter the minimum score a matcher must achieve in order to return 'True' on a decision table rule. By default this score is set to '70.'

> **Note:** An empty Condition Threshold should be used if a variable threshold is required between different rules. For example, one rule requires the matcher to return a score greater than '70,' and another rule needs it to be greater than '75.'

5. Navigate to the 'Advanced' tab. All but two of the parameters included on this tab require that a weight be defined. The matcher considers the individual weights of these elements when they are factored together for the rank score.



Word Split Regex: \s+

Exact Word Match Factor: 1.0

Alias Word Match Factor: 0.9

Metaphone3 Word Match Factor: 0.8

Edit Distance Word Match Factor: 0.8

Initials Word Match Factor: 0.5

Missing Word Factor: 0.8

Word Out Of Order Factor: 1.0

Unmatched Word Factor Table: ...

6. In the Word Splitter Regex parameter, enter the RegEx used to split the word values into separate words.
7. Required weighed parameters include:
   - **Exact Word Match Factor**: Determines how pairs that are exact matches should influence the final score.
   - **Alias Word Match Factor**: Determines how words paired together via aliases should influence the final score.
   - **Metaphone3 Word Match Factor**: Determines how words paired together via Metaphone 3 should influence the final score.
   - **Edit Distance Word Match Factor**: Determines how words that are paired via edit distance influence the final score.
   - **Initials Word Match Factor**: Determines how words paired together via initials influence the final score.
   - **Missing Word Factor**: Determines how much unpaired / missing words should penalize the final result.
   - **Word Out Of Order Factor**: Determines how much words that appear out of order should penalize the final result.
9. In the Unmatched Word Factor Table parameter, click the ellipsis button (...) and select the relevant lookup table.

> **Note:** The Unmatched Word Factor Table is a lookup table that assigns factors to certain words.

By default, unpaired / missing words will penalize the final score using the Missing Word Factor parameter. However, if an unpaired / missing word appears in this table, it will penalize the final score using the factor in the table rather than the factor configured in Missing Word Factor parameter. This can be used to reduce or increase the penalty that certain words, depending on their significance, have on the final score if they are missing.

10. Click OK when finished.

Extending Customer Data Matchers With JavaScript

For especially complicated solutions, it is possible to expand the capabilities of a customer data matcher via JavaScript. These work in much the same way as basic customer data matchers but allow for more flexibility and expanded functionality.

A typical customer data JavaScript matcher complies with the following basic steps:

1. Uses mc.evaluate to retrieve the output of a desired normalizer.

> **Note:** 'mc' is a bind to the match expression context.

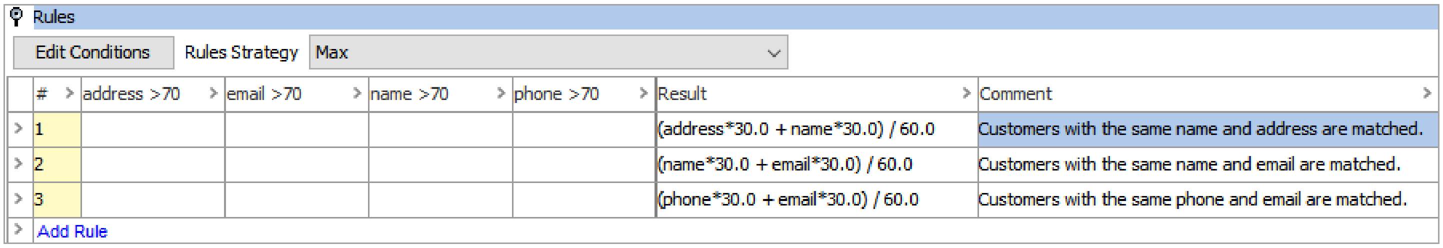
2. Uses an iterator to access the set of values / strings of both objects being matched.
3. Compares those objects and outputs a rank score.

For more information on customer data JavaScript normalizers, see the **Match Criteria Data Elements** topic of this documentation.

Match Criteria Rules

Match criteria rules dictate the final outcome of the matching evaluation. Each rule is evaluated by itself and represents a possible result of a comparison of two records. Only one rule will eventually provide the final score.

The rules strategy determines which rule provides the final Score. With rules strategy ‘First’, the first rule with no condition evaluating to false, provides the score. With rules strategy ‘Max’, the rule with the highest score, with no condition evaluating to false, provides the score.



Rules Strategy: Max

#	address >70	email >70	name >70	phone >70	Result	Comment
1					(address*30.0 + name*30.0) / 60.0	Customers with the same name and address are matched.
2					(name*30.0 + email*30.0) / 60.0	Customers with the same name and email are matched.
3					(phone*30.0 + email*30.0) / 60.0	Customers with the same phone and email are matched.

Consider the very simplistic example above. The strategy is to return the result of the rule with the maximum score. There are then two rules. The first combines address and name into a common score, while the second rule ensures that records sharing a social security number are always matched.

Matchers are, optionally, represented as a condition columns on the rules table, and each row corresponds to a separate rule. The Result column calculates a score of the matched objects.

The conditions control if the result rule formula is used to calculate a possible score. For each condition column, the condition threshold from the Matcher is displayed next to the name of the matcher (provided that matcher has a condition threshold defined).

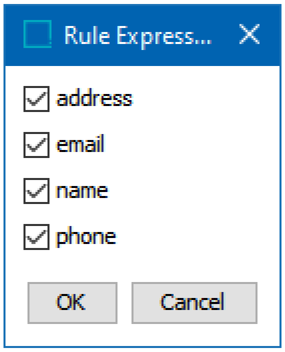
If the rules strategy is 'Max,' the result of all rules with true conditions are calculated, and the maximum result is reported as the Match criteria score.

If the rules strategy is 'First,' then the result of the topmost rule with true conditions is reported as the Match criteria Score.

In the above example, the '>70' value displayed in the top row of the three conditions has been provided by the corresponding matcher configurations, each of which states that if it exceeds a condition threshold of '70' it will return 'True.' In some cases, such as when using a more function-based decision table, the threshold will not be established in the matcher configuration and must instead be defined in the table cell.

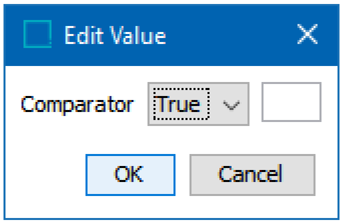
Configuration

1. Edit Conditions: Matchers can be added / removed from the rules table by clicking the Edit Conditions button, and selecting the desired matchers from the 'Rule Expression' dialog.



☑ address
☑ email
☑ name
☑ phone

2. Rules Strategy: The rules strategy determines how the rules are applied when generating a match result. Two options are available from the dropdown: 'First' and 'Max.' Selecting 'First' tells the decision table to look at the rules from top to bottom and base the results on the first rule in which all conditions return 'True.' Alternatively, selecting 'Max' tells the decision table to evaluate all conditions that return 'True' and combine their maximum results.
3. Add Rule: Click the Add Rule link in order to add a row to the table.
4. Rule Condition: If a particular condition should be included in a rule, click the ellipsis button ( ) and add a comparator via the 'Edit Value' popup dialog.

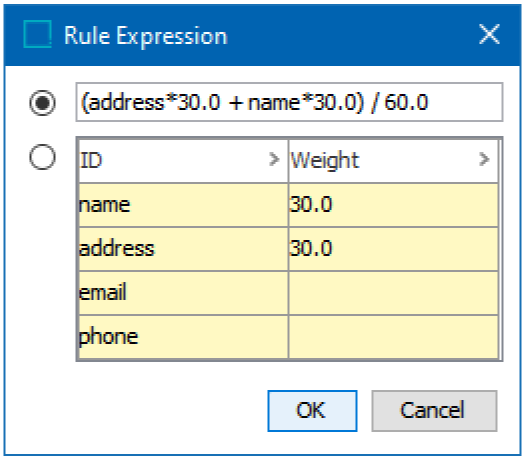


Comparator: True

Alternatively, the comparator can be entered directly into the cell. If the condition threshold is defined in the matcher configuration, select 'True.' If it is not, manually enter the condition threshold.

5. Rule Result: Each decision table rule requires an expression that drives the logic of the resulting score. In most cases, this is as simple as assigning weights to all conditions relevant to the rule. To create an expression, enter the expression directly into the cell.

   Alternatively, click the ellipsis button (...) to access the 'Rule Expression' dialog and click the table radio button. Enter the desired weights for the relevant conditions and click OK. This will automatically generate the relevant expression.



● (address*30.0 + name*30.0) / 60.0

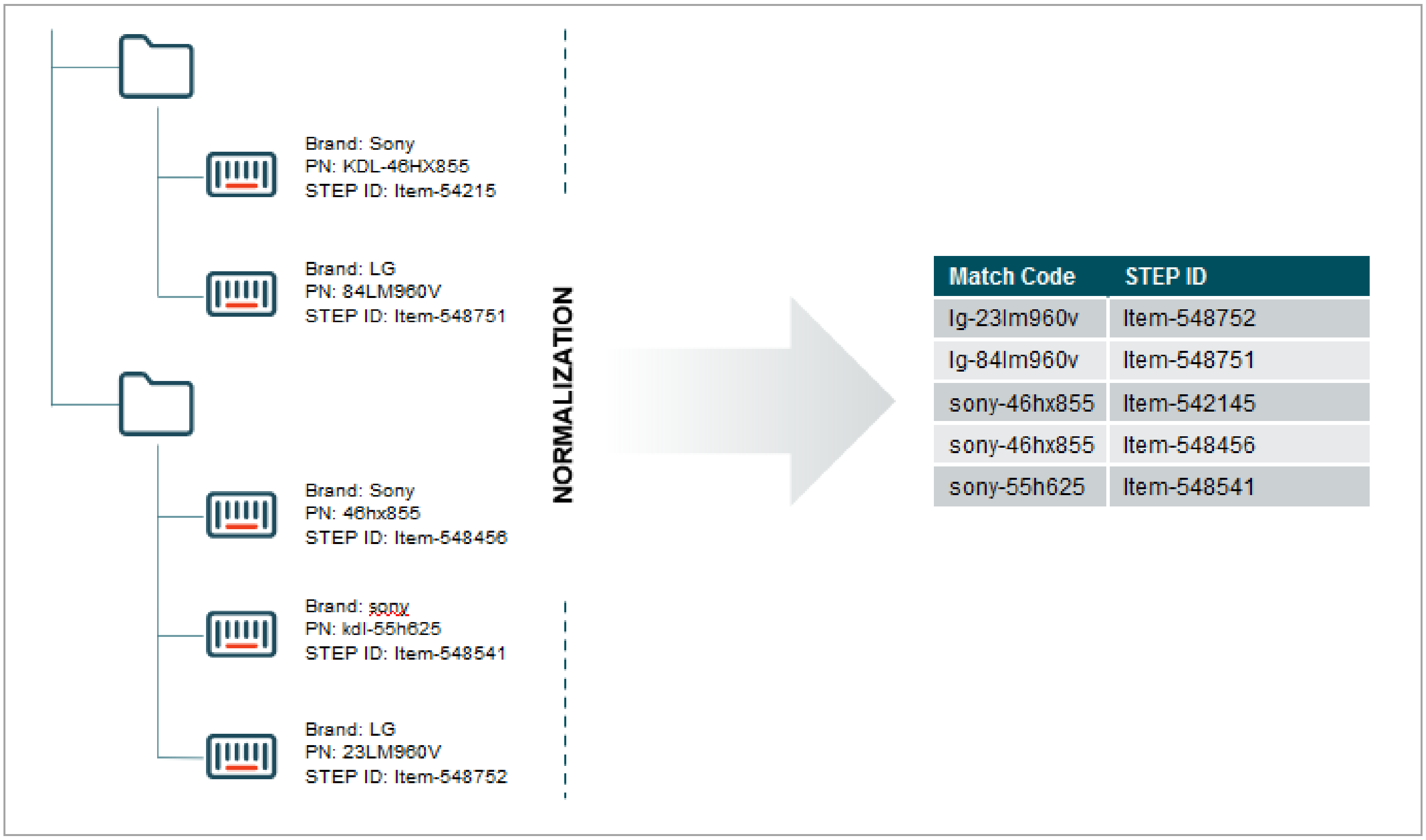
○

ID	Weight
name	30.0
address	30.0
email	
phone	

Match Codes

The purpose of match criteria is to determine if the record at hand matches another record in the database. The database can contain an incredible amount of data, so we need a fast and efficient way to find the records that potentially are matches. That is the purpose of match code. Through match codes, algorithms can compare created results and process records quickly.

A match code is essentially a string (i.e., a text) representing an object. Once generated, these match codes populate an alphabetically sorted table in the system. Rather than comparing every object with every other object in the dataset, only objects with at least one equal match codes will be compared.



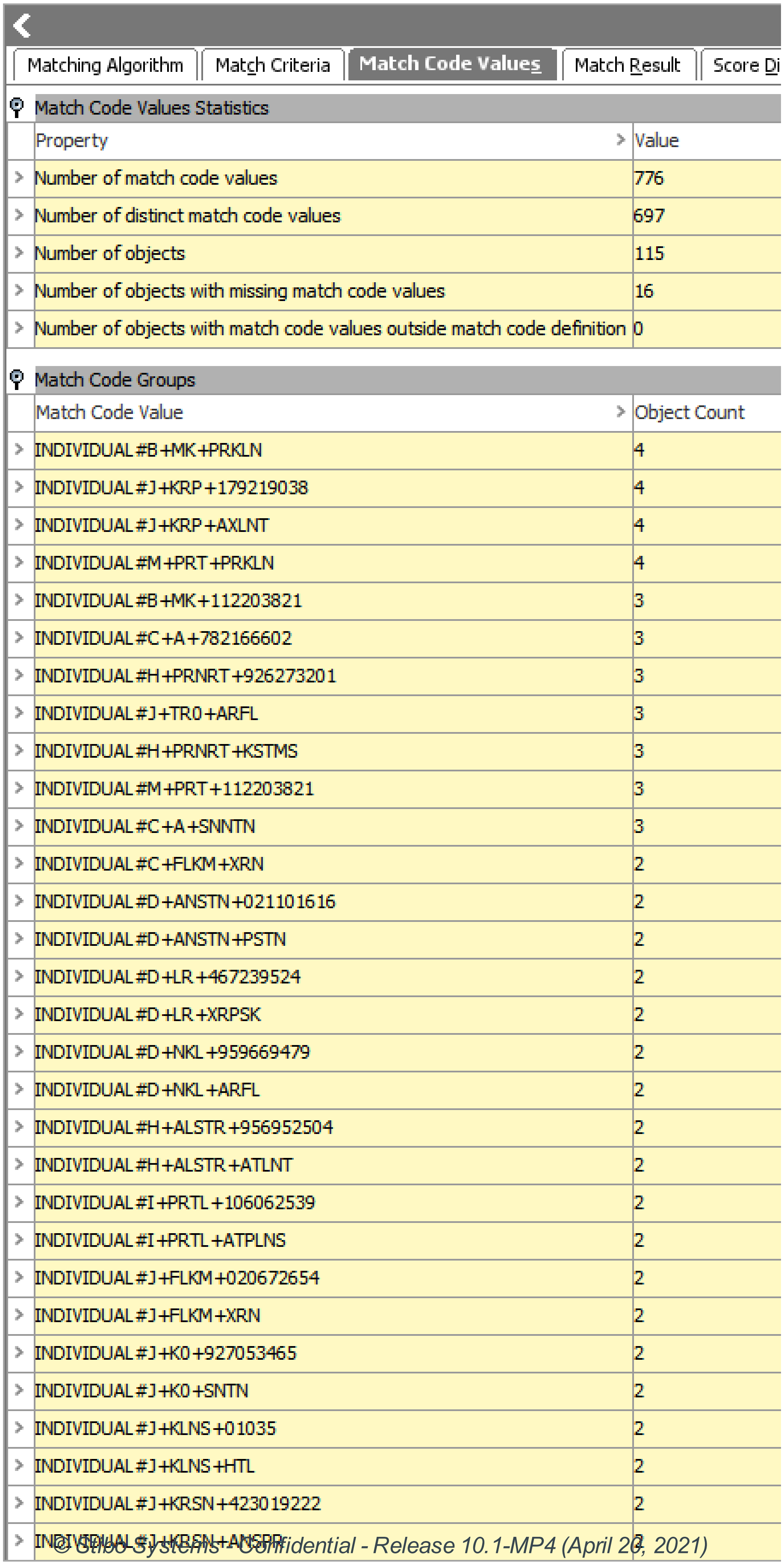
Match Code	STEP ID
lg-23lm960v	Item-548752
lg-84lm960v	Item-548751
sony-46hx855	Item-542145
sony-46hx855	Item-548456
sony-55h625	Item-548541

In the example above, the product with STEP ID Item-548456 is the record-at-hand. By the match code table, we can see that one other object has an identical match code.

It is usually necessary to use several different match codes to ensure matching records are actually compared. Determining which and how many match codes to use is a balance. It is important that matching records share at least one match code. It is undesirable that non-matching records share match codes, as running the full match criteria comparisons on those records will waste system resources.

Match Code Values

On a running system, match code values can be examined in workbench using the match code values tab. It is important that match codes are relatively unique. A group of equal match codes is referred to as a match code group. It is important to ensure that match code groups are small. A match code group exceeding an object count of 10 should be considered a problem.



Match Code Values Statistics

Property	Value
Number of match code values	776
Number of distinct match code values	697
Number of objects	115
Number of objects with missing match code values	16
Number of objects with match code values outside match code definition	0

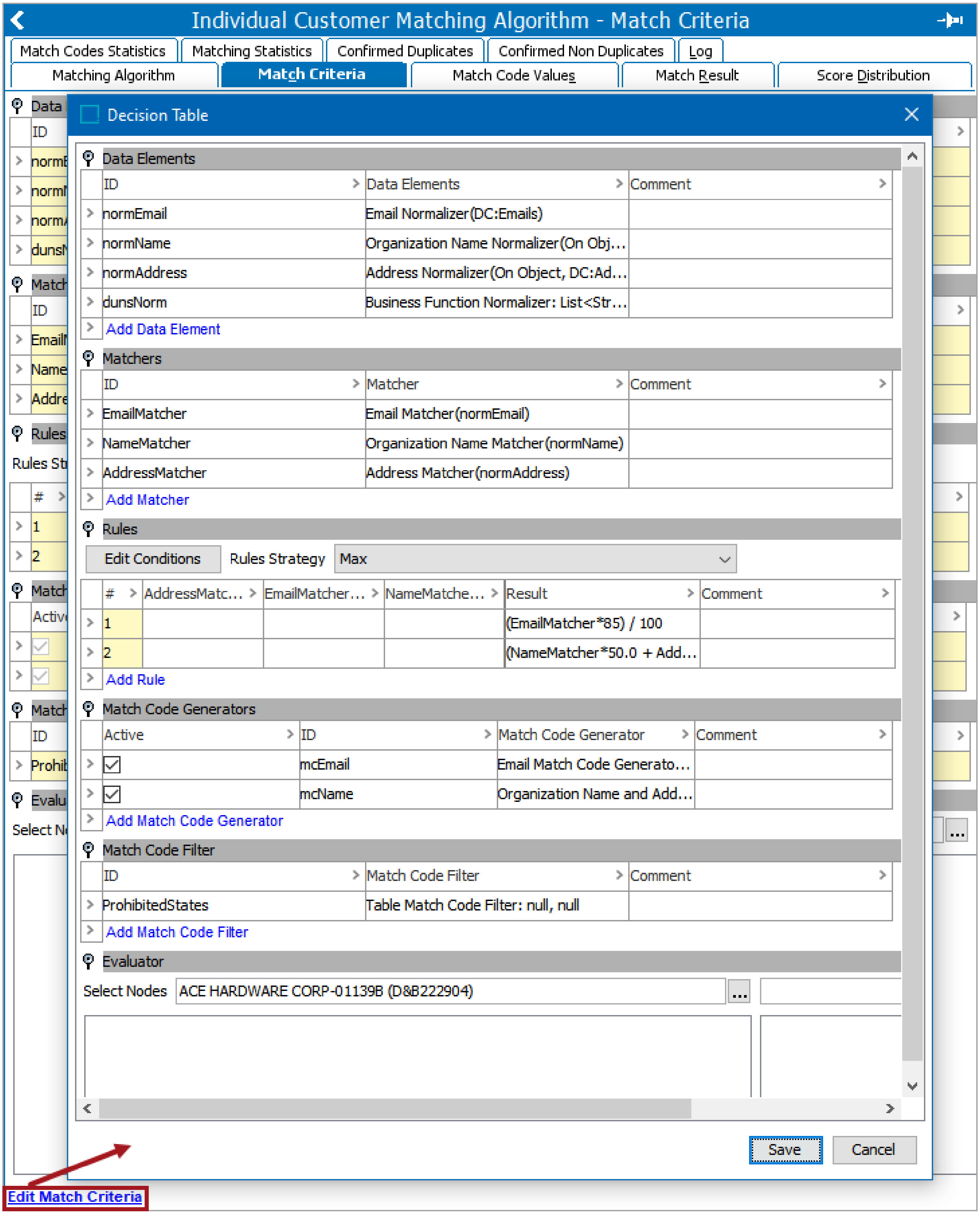
Match Code Groups

Match Code Value	Object Count
INDIVIDUAL#B+MK+PRKLN	4
INDIVIDUAL#J+KRP+179219038	4
INDIVIDUAL#J+KRP+AXLNT	4
INDIVIDUAL#M+PRT+PRKLN	4
INDIVIDUAL#B+MK+112203821	3
INDIVIDUAL#C+A+782166602	3
INDIVIDUAL#H+PRNRT+926273201	3
INDIVIDUAL#J+TR0+ARFL	3
INDIVIDUAL#H+PRNRT+KSTMS	3
INDIVIDUAL#M+PRT+112203821	3
INDIVIDUAL#C+A+SNNTN	3
INDIVIDUAL#C+FLKM+XRN	2
INDIVIDUAL#D+ANSTN+021101616	2
INDIVIDUAL#D+ANSTN+PSTN	2
INDIVIDUAL#D+LR+467239524	2
INDIVIDUAL#D+LR+XRPSK	2
INDIVIDUAL#D+NKL+959669479	2
INDIVIDUAL#D+NKL+ARFL	2
INDIVIDUAL#H+ALSTR+956952504	2
INDIVIDUAL#H+ALSTR+ATLNT	2
INDIVIDUAL#I+PRTL+106062539	2
INDIVIDUAL#I+PRTL+ATPLNS	2
INDIVIDUAL#J+FLKM+020672654	2
INDIVIDUAL#J+FLKM+XRN	2
INDIVIDUAL#J+K0+927053465	2
INDIVIDUAL#J+K0+SNTN	2
INDIVIDUAL#J+KLNS+01035	2
INDIVIDUAL#J+KLNS+HTL	2
INDIVIDUAL#J+KRSN+423019222	2
INDIVIDUAL#J+KRSN+ANSPR	2

- Closely examine the data before configuring a match code. The data profiling tool can provide a lot of valuable information, and if you are planning to use a specific attribute in the match code, always check to which degree the attribute is populated – if values are missing on a lot of objects, the attribute is likely not a good candidate, or at least should not be used alone. Objects with 'empty' value for a match codes will not be compared based on that match code.
- If an attribute is sufficiently unique, like an EAN number, the match code can be based on just that single piece of data.
- If an attribute is less unique, like a name, it would have to be used in combination with other values in order to make good match codes. An example could be the Person Name and Address match code generator available for Customer Data.
- When working with match codes combining several pieces of data, always put the most significant data first. For instance, if deduplicating address objects, put the ZIP code before street and street number, as ZIP codes are geographic, standardized, and mutually exclusive – which most effectively separates your addresses into discrete objects.
- Be sure to normalize the data used in match codes. If, for instance, a manufacturer name is often abbreviated, your match code definition should handle this so that the name is represented the same way in the match codes, regardless of whether it is abbreviated on the source object or not.
- Several match codes can be generated per source object, even by the same match code generator. STEP functions can resolve to a list of multiple match codes, and in JavaScript, an array can be returned. In these cases, each element will be a separate match code. Consider, for example, a customer having several email addresses. Each email address should result in a separate email match code.
- Sometimes an otherwise great identifier has exception cases that should be filtered out. Phone numbers are often very good match code candidates, but a number of contacts at a customer may have provided the reception main number, resulting in a single match code group with hundreds of records. In this case, a match code filter may be applied to the phone match code to remove this exceptional case. For more information, see the **Match Code Filter** section of this topic.

Configuring a Match Code Generator

Match codes are created by match code generators in the Match Criteria tab in the matching algorithm. The following picture shows the match criteria in edit mode.



Data Elements

ID	Data Elements	Comment
normEmail	Email Normalizer(DC:Emails)	
normName	Organization Name Normalizer(On Obj…	
normAddress	Address Normalizer(On Object, DC:Ad…	
dunsNorm	Business Function Normalizer: List<Str…	

Add Data Element

Matchers

ID	Matcher	Comment
EmailMatcher	Email Matcher(normEmail)	
NameMatcher	Organization Name Matcher(normName)	
AddressMatcher	Address Matcher(normAddress)	

Add Matcher

Rules

Rules Strategy: Max

#	AddressMatc…	EmailMatcher…	NameMatche…	Result	Comment
1				(EmailMatcher*85) / 100	
2				(NameMatcher*50.0 + Add…	

Add Rule

Match Code Generators

Active	ID	Match Code Generator	Comment
☑	mcEmail	Email Match Code Generato…	
☑	mcName	Organization Name and Add…	

Add Match Code Generator

Match Code Filter

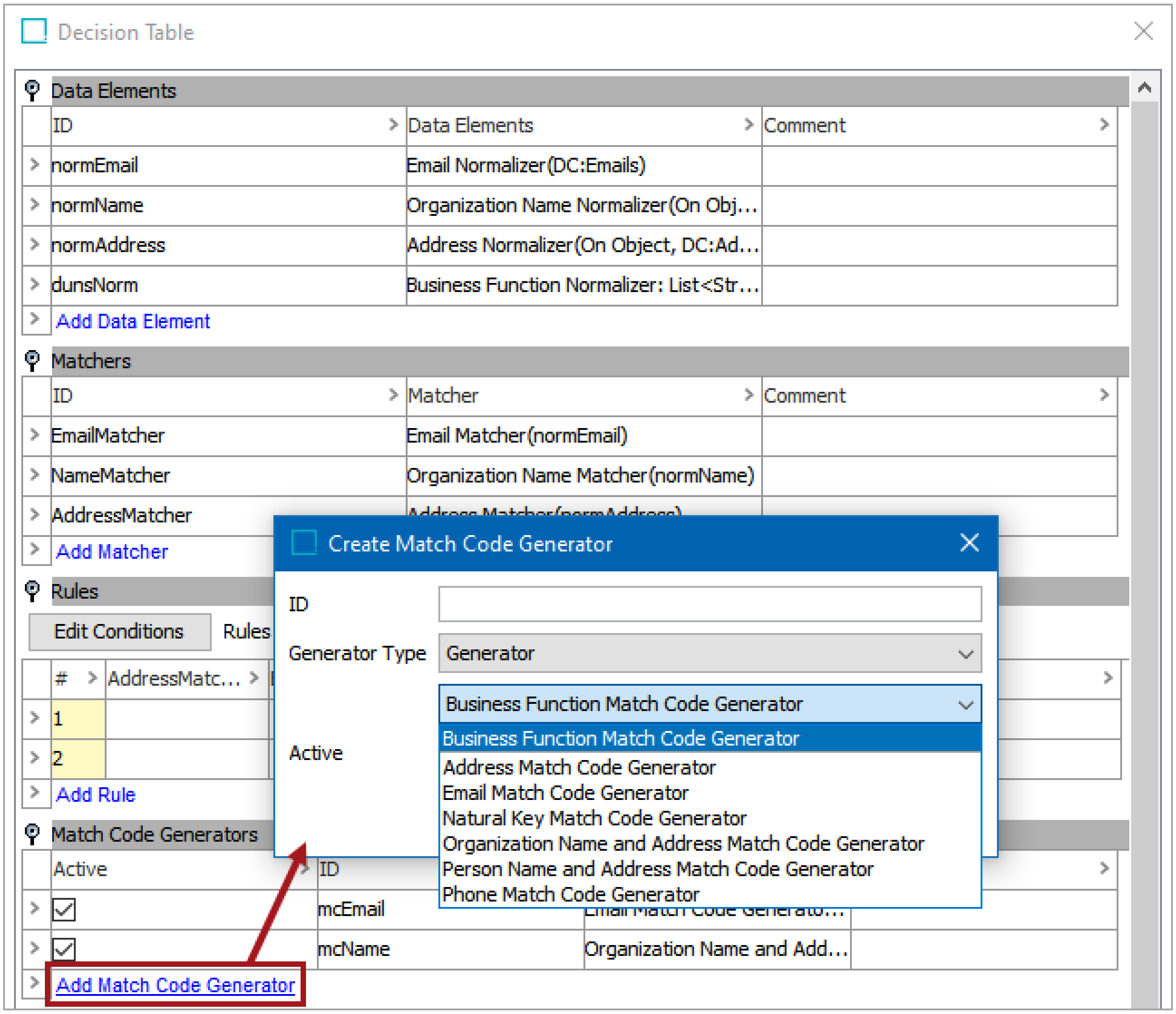
ID	Match Code Filter	Comment
ProhibitedStates	Table Match Code Filter: null, null	

Add Match Code Filter

Evaluator

Select Nodes: ACE HARDWARE CORP-01139B (D&B222904)

To create a match code generator, select Add Match Code Generator.



Decision Table

Data Elements

ID	Data Elements	Comment
normEmail	Email Normalizer(DC:Emails)	
normName	Organization Name Normalizer(On Obj…	
normAddress	Address Normalizer(On Object, DC:Ad…	
dunsNorm	Business Function Normalizer: List<Str…	

Matchers

ID	Matcher	Comment
EmailMatcher	Email Matcher(normEmail)	
NameMatcher	Organization Name Matcher(normName)	
AddressMatcher	Address Matcher(normAddress)	

#	AddressMatc…
1	
2	

Match Code Generators

Active	ID	
☑	mcEmail	Email Match Code Generato…
☑	mcName	Organization Name and Add…

Create Match Code Generator

ID:

Generator Type: Generator

Active

Business Function Match Code Generator

- Business Function Match Code Generator
- Address Match Code Generator
- Email Match Code Generator
- Natural Key Match Code Generator
- Organization Name and Address Match Code Generator
- Person Name and Address Match Code Generator
- Phone Match Code Generator

1. Once created, the match code generator requires configuration. Within the match criteria, click the ellipsis button (...) to specify the various details of the match code generator.
2. Once configured, test the match code generators with a variety of records to ensure that everything is properly configured. To test the match code generator, some codes need to be created. The following example shows several matched records on the previously detailed match code generator. For example purposes, all of these records share the same email. For more information, see the **Configuring Matching Algorithms** topic in this documentation.

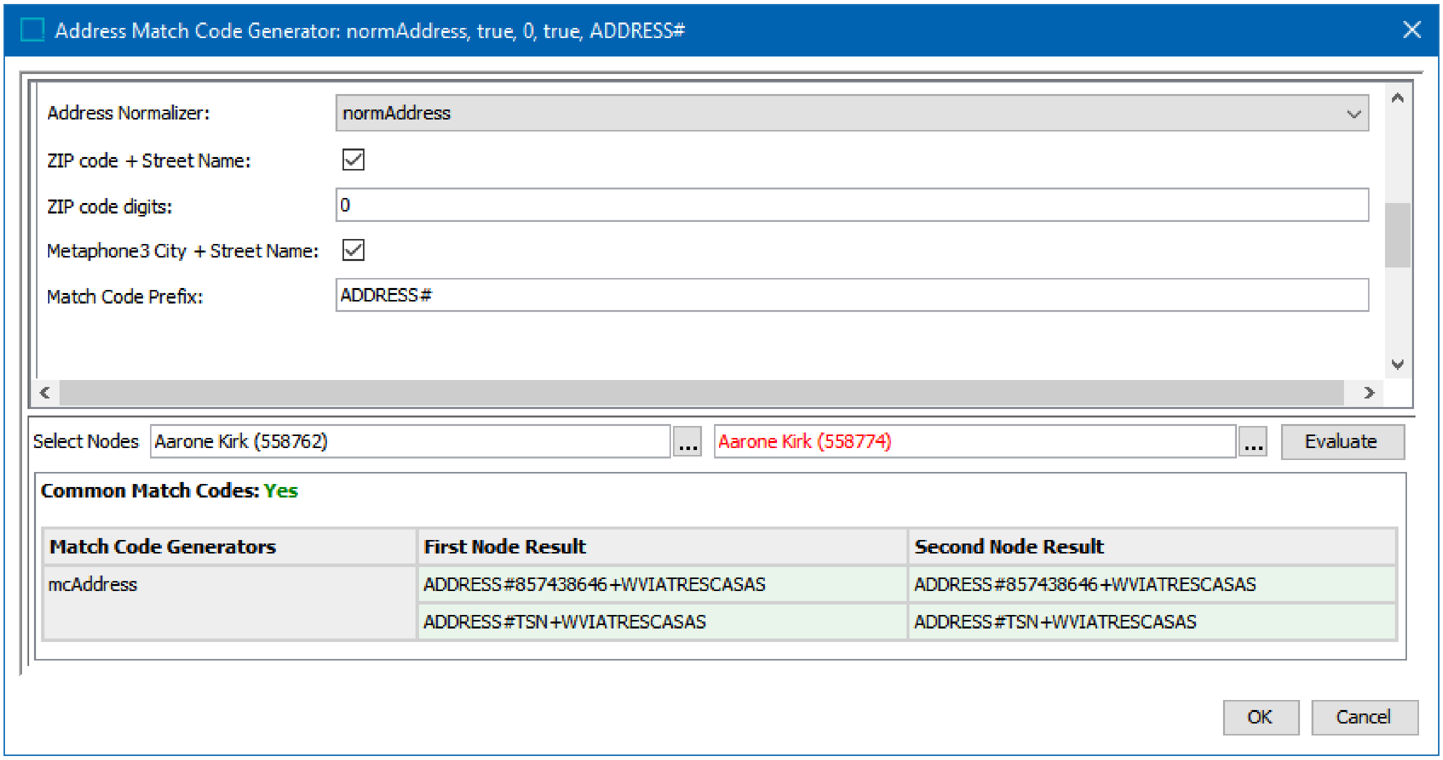
All of the match code generators described below require that users map one or more normalizers to generate codes. For more information, see the **Match Criteria Data Elements** topic in this documentation.

Business Function Match Code Generator

A Business Function match code generator uses a business function to produce match codes.

Address Match Code Generator

The Address match code generator must be mapped to an address normalizer.



Address Match Code Generator: normAddress, true, 0, true, ADDRESS#

Address Normalizer:	normAddress
ZIP code + Street Name:	☑
ZIP code digits:	0
Metaphone3 City + Street Name:	☑
Match Code Prefix:	ADDRESS#

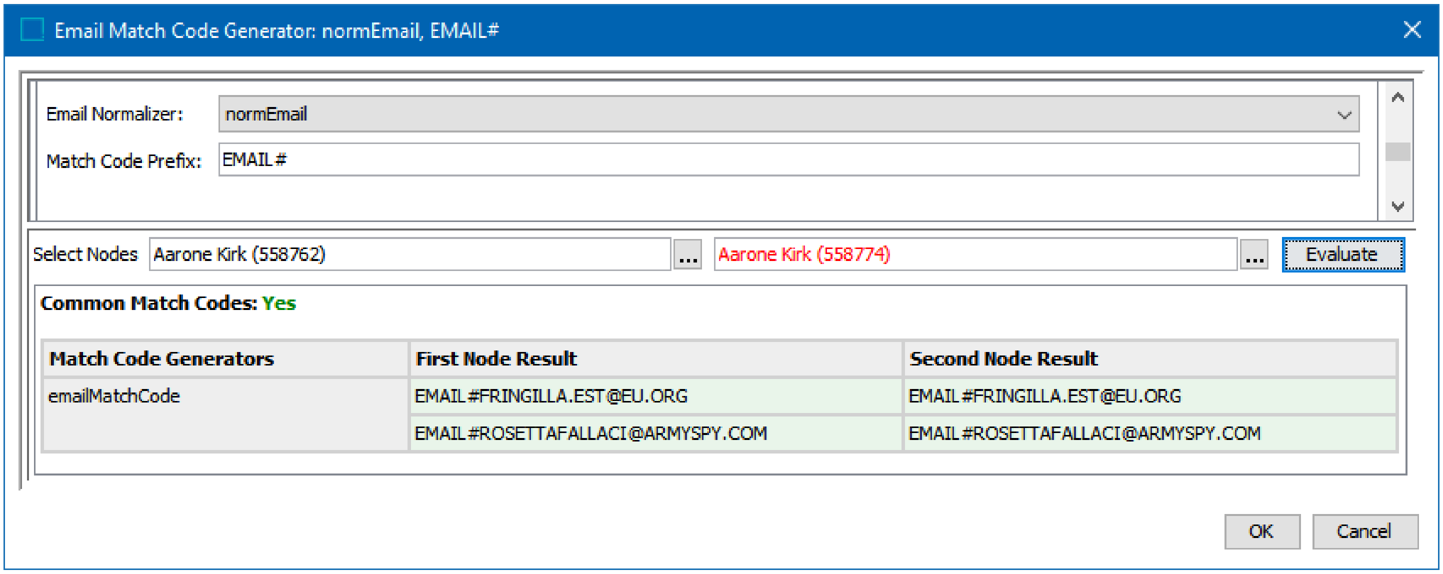
Select Nodes Aarone Kirk (558762) Aarone Kirk (558774)

Common Match Codes: Yes

Match Code Generators	First Node Result	Second Node Result
mcAddress	ADDRESS#857438646+WVIATRESCASAS	ADDRESS#857438646+WVIATRESCASAS
	ADDRESS#TSN+WVIATRESCASAS	ADDRESS#TSN+WVIATRESCASAS

Email Match Code Generator

The Email match code generator must be mapped to an email normalizer.



Email Match Code Generator: normEmail, EMAIL#

Email Normalizer:	normEmail
Match Code Prefix:	EMAIL#

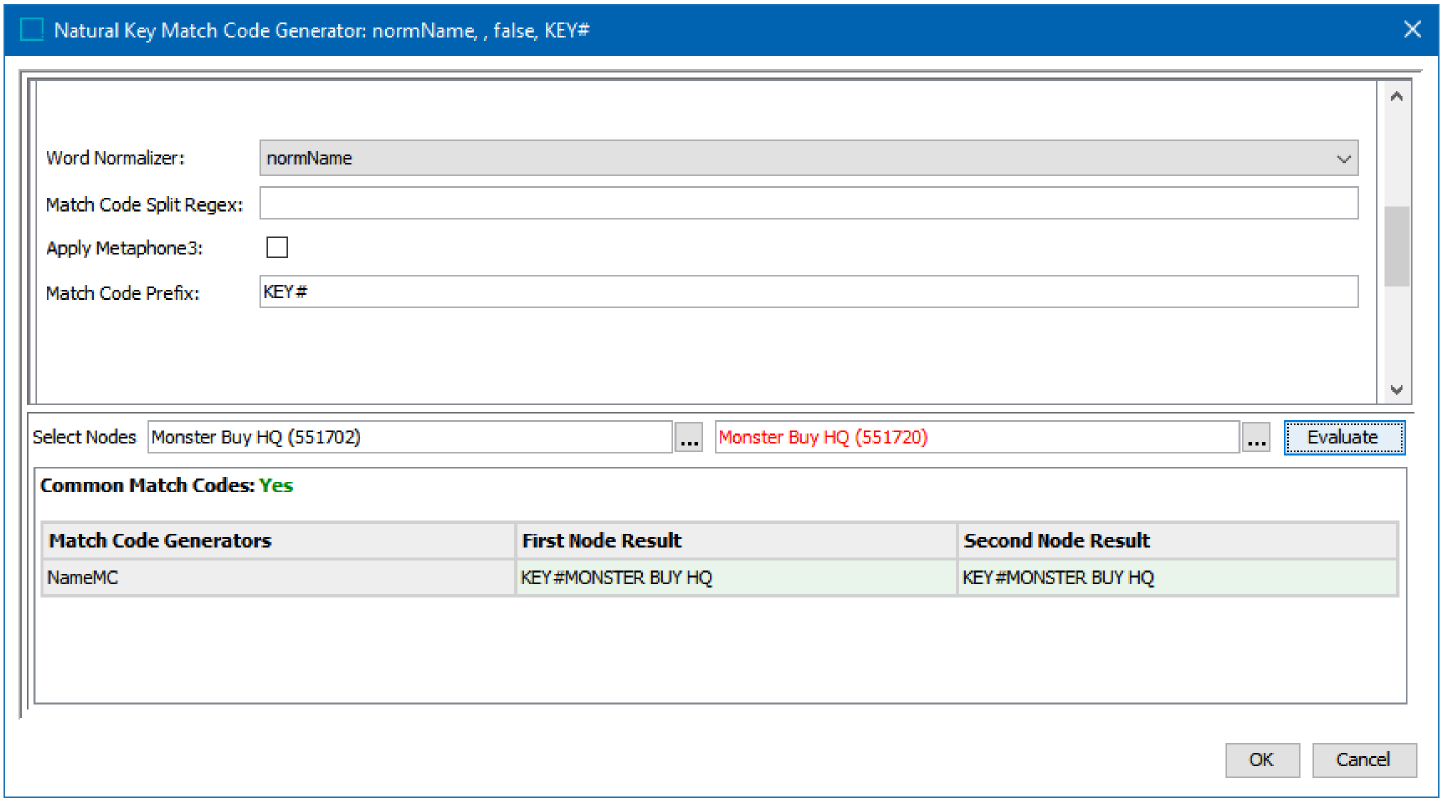
Select Nodes Aarone Kirk (558762) Aarone Kirk (558774)

Common Match Codes: Yes

Match Code Generators	First Node Result	Second Node Result
emailMatchCode	EMAIL#FRINGILLA.EST@EU.ORG	EMAIL#FRINGILLA.EST@EU.ORG
	EMAIL#ROSETTAFALLACI@ARMYSPY.COM	EMAIL#ROSETTAFALLACI@ARMYSPY.COM

Natural Key Match Code Generator

The Natural Key code generator must be mapped to a words normalizer.



Natural Key Match Code Generator: normName, , false, KEY#

Word Normalizer: normName

Match Code Split Regex:

Apply Metaphone3: □

Match Code Prefix: KEY#

Select Nodes Monster Buy HQ (551702) ... Monster Buy HQ (551720) ... Evaluate

Common Match Codes: Yes

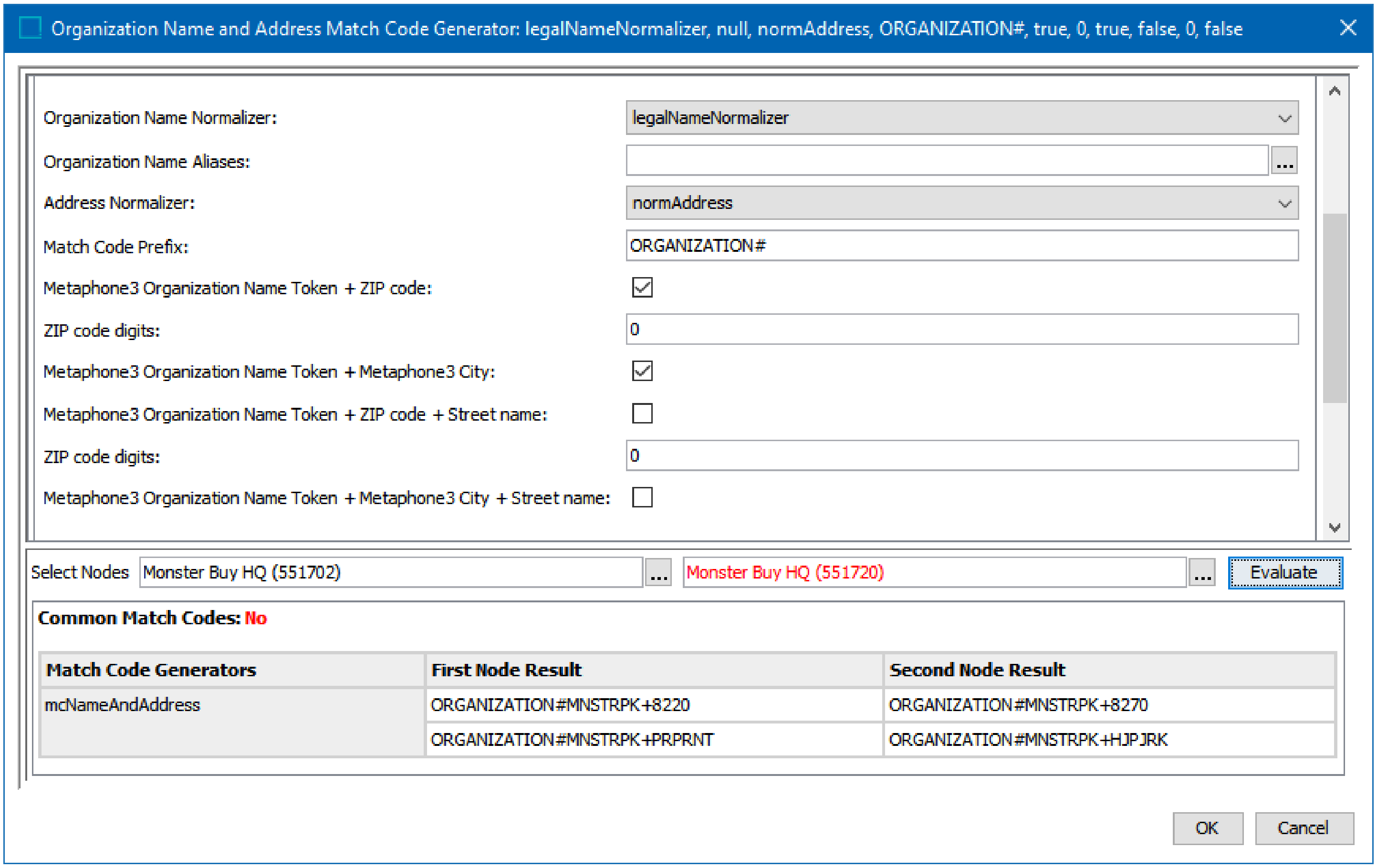
Match Code Generators	First Node Result	Second Node Result
NameMC	KEY#MONSTER BUY HQ	KEY#MONSTER BUY HQ

Organization Name and Address Match Code Generator

It is often useful to match on organization names, but they are often not sufficiently unique. Some supermarket chains would use the same organization name on each of their sites, so it needs to be combined with something more. The combination of organization name and address usually constitutes a good match code.

The organization name and address match code generator must be mapped to an Organization Name normalizer and an Address normalizer.

Organization names are often abbreviated and appended with terms like 'Inc.' that may, at times, be omitted in some systems. For that reason, the organization name should often be followed by a transformation lookup table with aliases.



Organization Name and Address Match Code Generator: legalNameNormalizer, null, normAddress, ORGANIZATION#, true, 0, true, false, 0, false

Organization Name Normalizer:	legalNameNormalizer
Organization Name Aliases:	
Address Normalizer:	normAddress
Match Code Prefix:	ORGANIZATION#
Metaphone3 Organization Name Token + ZIP code:	☑
ZIP code digits:	0
Metaphone3 Organization Name Token + Metaphone3 City:	☑
Metaphone3 Organization Name Token + ZIP code + Street name:	□
ZIP code digits:	0
Metaphone3 Organization Name Token + Metaphone3 City + Street name:	□

Select Nodes Monster Buy HQ (551702) ... Monster Buy HQ (551720) ... Evaluate

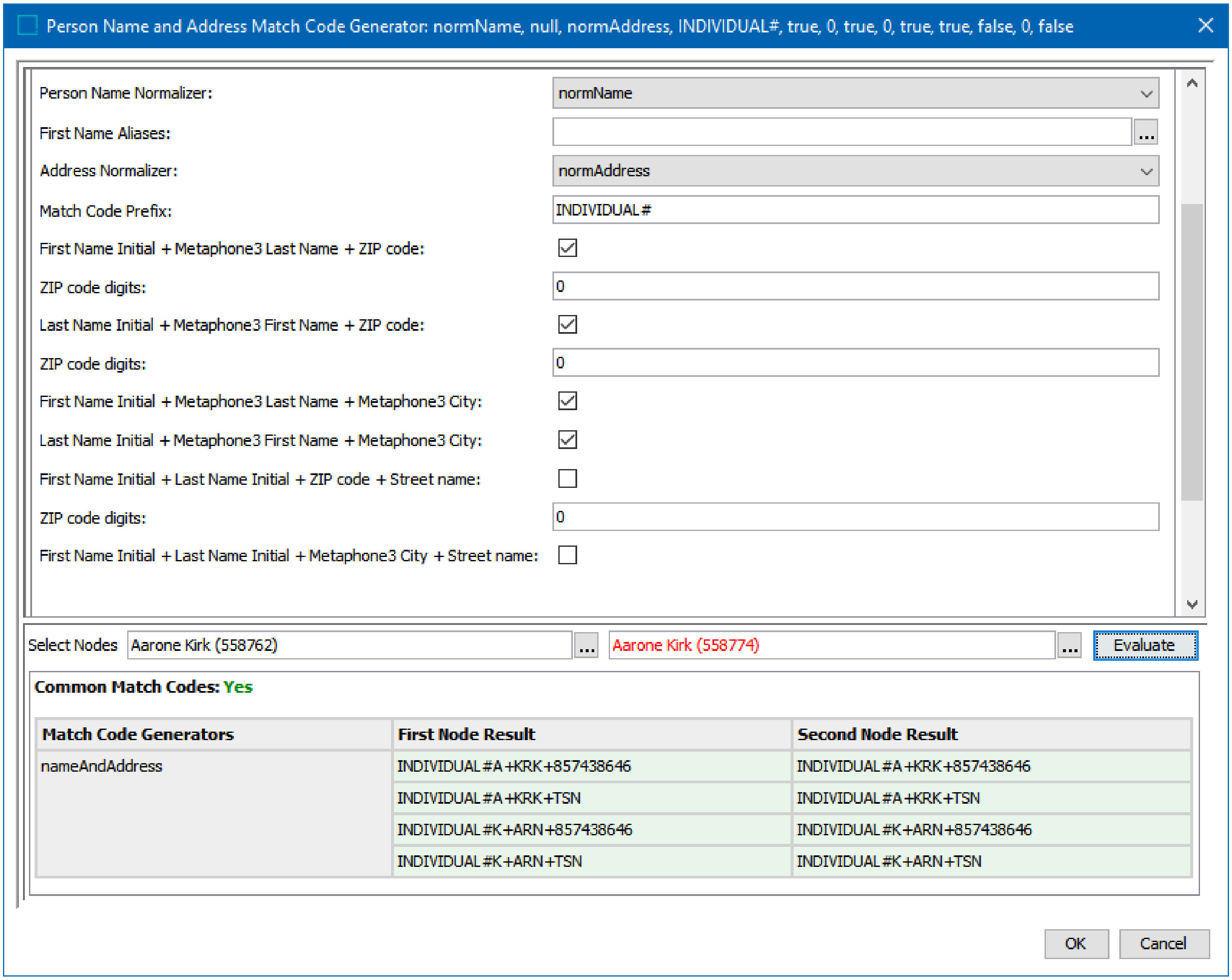
Common Match Codes: No

Match Code Generators	First Node Result	Second Node Result
mcNameAndAddress	ORGANIZATION#MNSTRPK+8220	ORGANIZATION#MNSTRPK+8270
	ORGANIZATION#MNSTRPK+PRPRNT	ORGANIZATION#MNSTRPK+HJPJRK

Person Name and Address Match Code Generator

It is often useful to match on person names, but they are often not sufficiently unique, so it needs to be combined with something more. The combination of person name and address usually constitutes a good match code.

The person name and address match code generator must be mapped to a Person Name normalizer and an Address normalizer.



Person Name and Address Match Code Generator: normName, null, normAddress, INDIVIDUAL#, true, 0, true, 0, true, true, false, 0, false

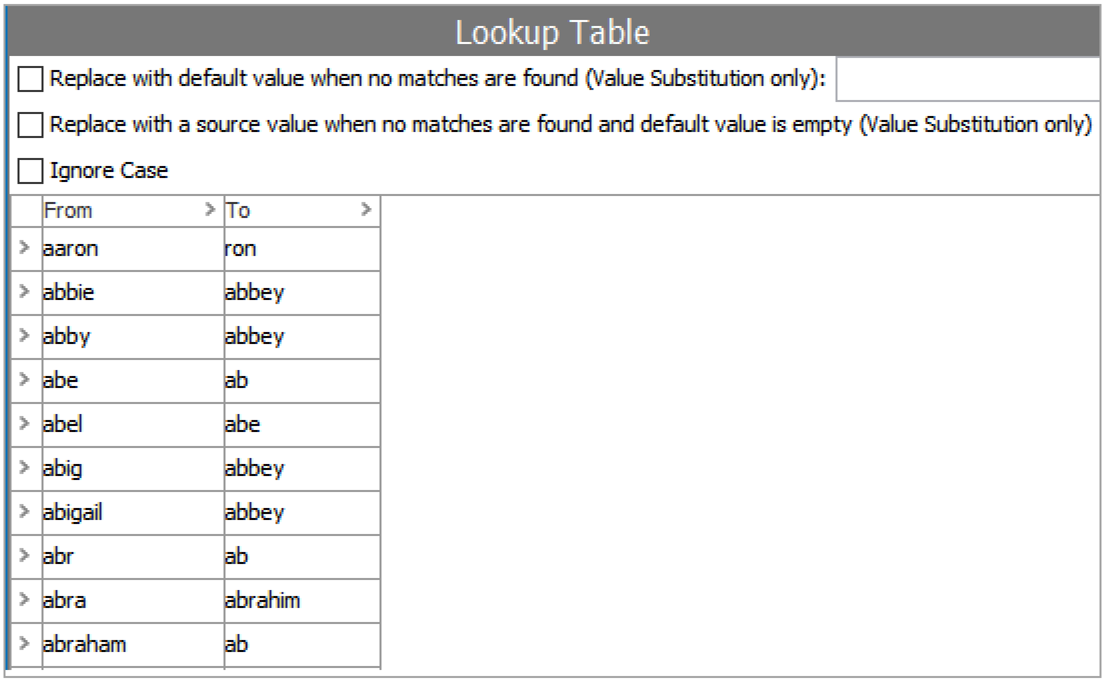
Person Name Normalizer:	normName
First Name Aliases:	
Address Normalizer:	normAddress
Match Code Prefix:	INDIVIDUAL#
First Name Initial + Metaphone3 Last Name + ZIP code:	☑
ZIP code digits:	0
Last Name Initial + Metaphone3 First Name + ZIP code:	☑
ZIP code digits:	0
First Name Initial + Metaphone3 Last Name + Metaphone3 City:	☑
Last Name Initial + Metaphone3 First Name + Metaphone3 City:	☑
First Name Initial + Last Name Initial + ZIP code + Street name:	□
ZIP code digits:	0
First Name Initial + Last Name Initial + Metaphone3 City + Street name:	□

Select Nodes Aarone Kirk (558762) Aarone Kirk (558774)

Common Match Codes: Yes

Match Code Generators	First Node Result	Second Node Result
nameAndAddress	INDIVIDUAL#A+KRK+857438646	INDIVIDUAL#A+KRK+857438646
	INDIVIDUAL#A+KRK+TSN	INDIVIDUAL#A+KRK+TSN
	INDIVIDUAL#K+ARN+857438646	INDIVIDUAL#K+ARN+857438646
	INDIVIDUAL#K+ARN+TSN	INDIVIDUAL#K+ARN+TSN

Person names are sometimes abbreviated or exchanged for call names. For that reason, the person names should often be followed by a transformation lookup table with aliases.



Lookup Table

□ Replace with default value when no matches are found (Value Substitution only):

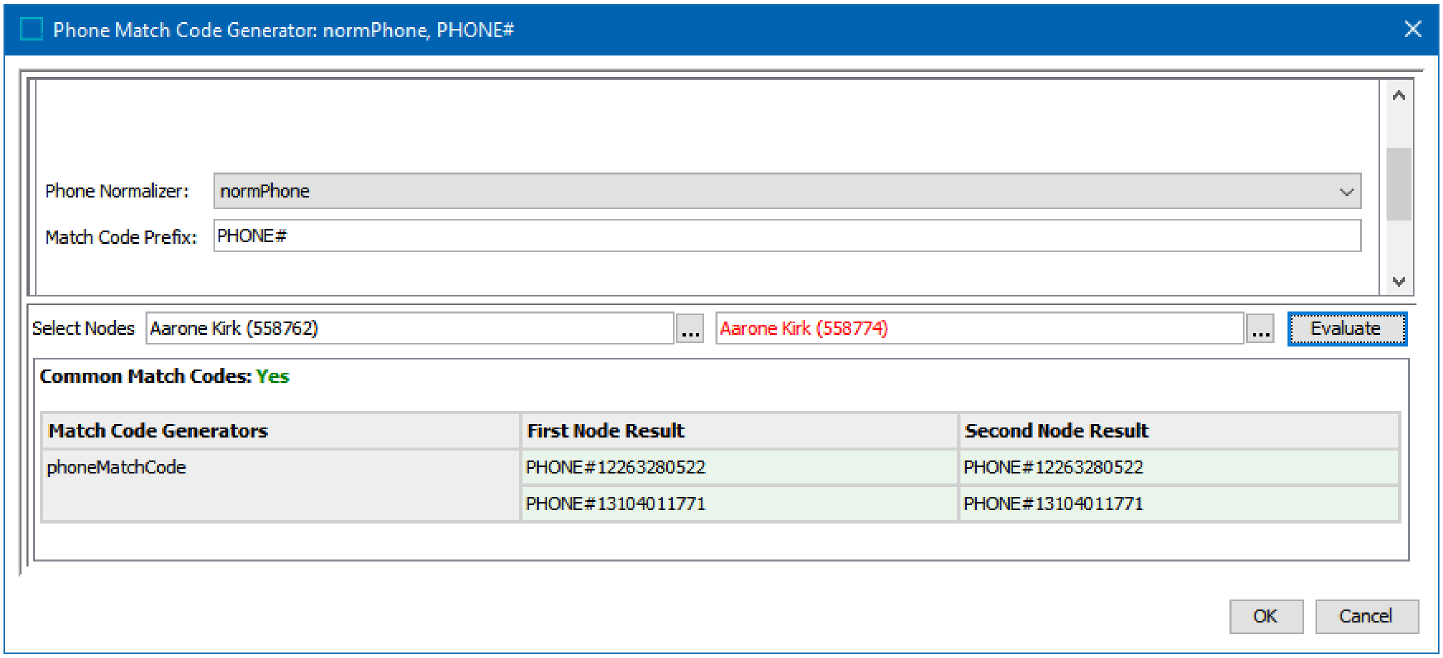
□ Replace with a source value when no matches are found and default value is empty (Value Substitution only)

□ Ignore Case

From	To
aaron	ron
abbie	abbey
abby	abbey
abe	ab
abel	abe
abig	abbey
abigail	abbey
abr	ab
abra	abrahim
abraham	ab

Phone Match Code Generator

The phone match code generator must be mapped to a phone normalizer.



Phone Match Code Generator: normPhone, PHONE#

Phone Normalizer: normPhone

Match Code Prefix: PHONE#

Select Nodes: Aarone Kirk (558762) ... Aarone Kirk (558774) ... Evaluate

Common Match Codes: **Yes**

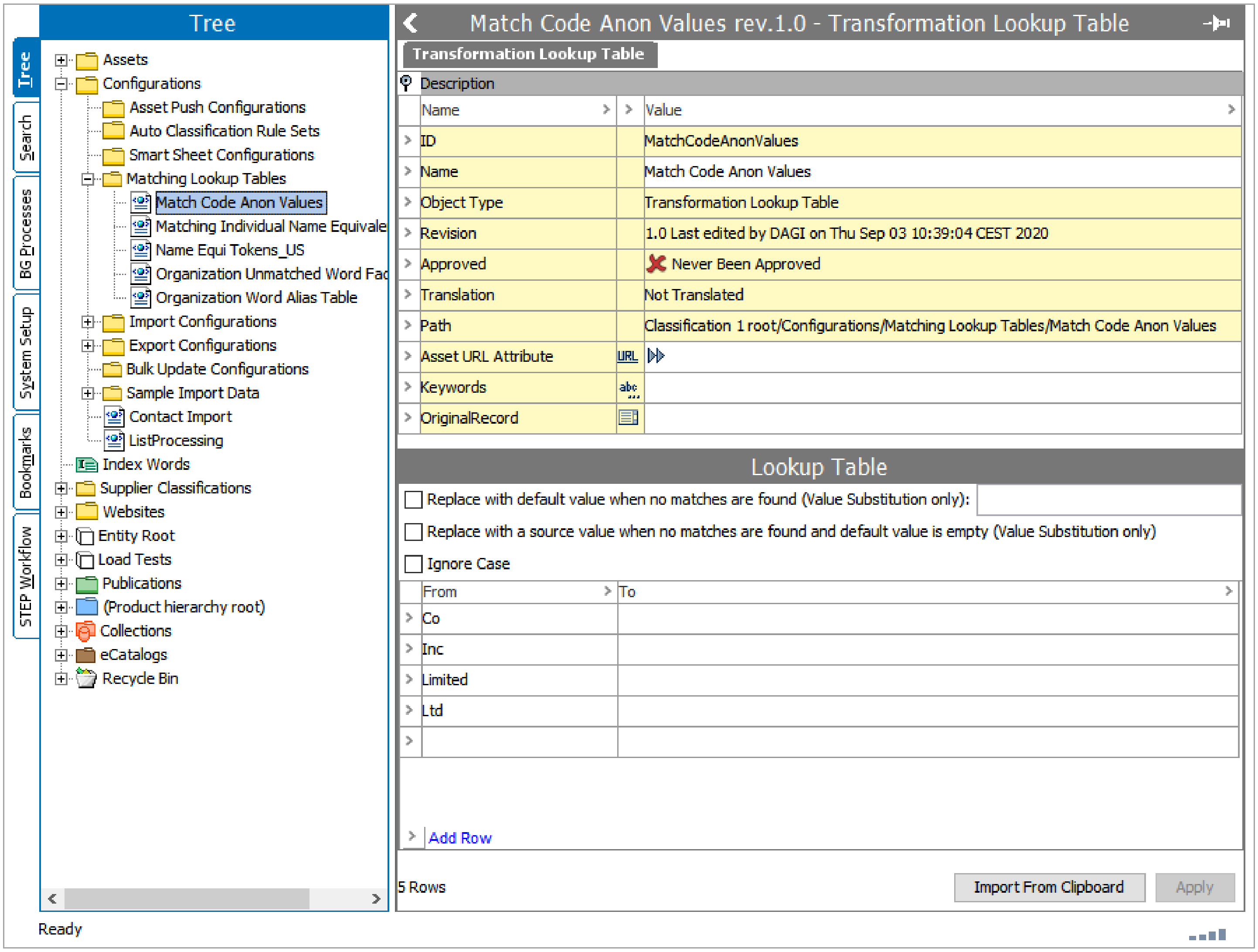
Match Code Generators	First Node Result	Second Node Result
phoneMatchCode	PHONE#12263280522	PHONE#12263280522
	PHONE#13104011771	PHONE#13104011771

Configuring a Match Code Filter

Sometimes exceptions in data create match codes that should really not result in comparing all records in the group. Overly large match code groups can often be found using the Match Code Values tab in the matching algorithm object.

Match code filters can only be used for matching algorithms that have been created with the Embed Match Code checkbox selected.

A match code filter is based on a table of specific match codes that should be filtered out. Create a new transformation lookup table and enter all the match codes to exclude in the 'from' column. Leave the 'to' column empty.



Name	Value
ID	MatchCodeAnonValues
Name	Match Code Anon Values
Object Type	Transformation Lookup Table
Revision	1.0 Last edited by DAGI on Thu Sep 03 10:39:04 CEST 2020
Approved	Never Been Approved
Translation	Not Translated
Path	Classification 1 root/Configurations/Matching Lookup Tables/Match Code Anon Values
Asset URL Attribute	
Keywords	
OriginalRecord	

☐ Replace with default value when no matches are found (Value Substitution only):

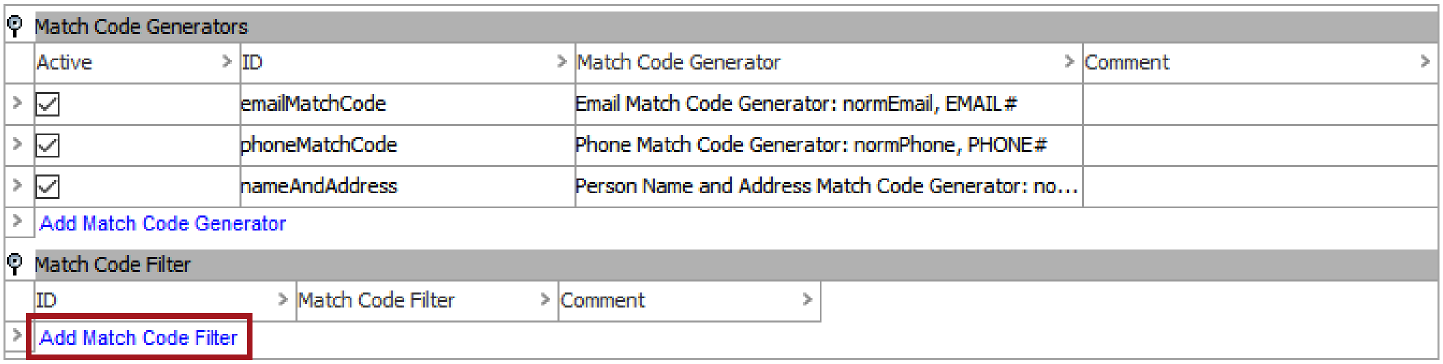
☐ Replace with a source value when no matches are found and default value is empty (Value Substitution only)

☐ Ignore Case

From	To
Co	
Inc	
Limited	
Ltd	

5 Rows

In System Setup, find the matching algorithm for which you want to apply the match code filter. On the Match Criteria tab click 'Edit Match Criteria' link that will open the Decision Table dialog. Find the Match Code Filter section and select 'Add Match Code Filter' link.



Match Code Generators

Active	ID	Match Code Generator	Comment
☑	emailMatchCode	Email Match Code Generator: normEmail, EMAIL#	
☑	phoneMatchCode	Phone Match Code Generator: normPhone, PHONE#	
☑	nameAndAddress	Person Name and Address Match Code Generator: no...	

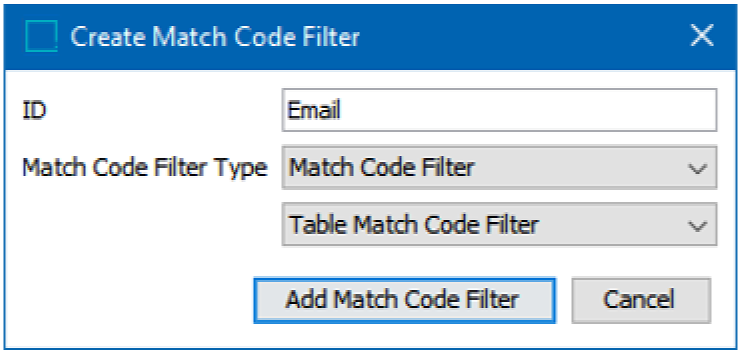
Add Match Code Generator

Match Code Filter

ID	Match Code Filter	Comment

Add Match Code Filter

The Create a Match Code Filter dialog will display. Provide an ID for the filter. The Type dropdown menus have only one option for each. When complete, press the Add Match Code Filter link.



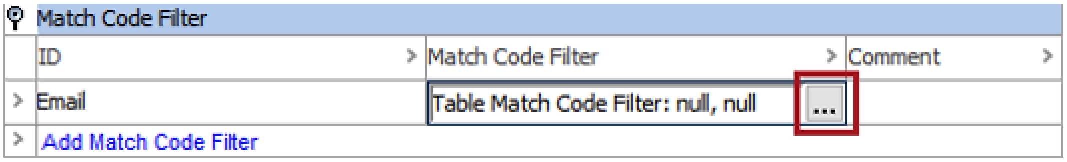
Create Match Code Filter

ID: Email

Match Code Filter Type: Match Code Filter

Table Match Code Filter

Once created, select the field for the table match code filter, and click the ellipsis button (...) to edit the filter.



Match Code Filter

ID	Match Code Filter	Comment
Email	Table Match Code Filter: null, null ...	

Add Match Code Filter

The Table Match Code Filter dialog will display. On this dialog, select the configured transformation table for the filter table.

Add a prefix such as 'EMAIL#' that will be prepended to all match codes. This field can and should be left blank if all the match codes in the transformation lookup table already have this prefix.

The Evaluate button allows users to test if the match codes can be found on the selected nodes.

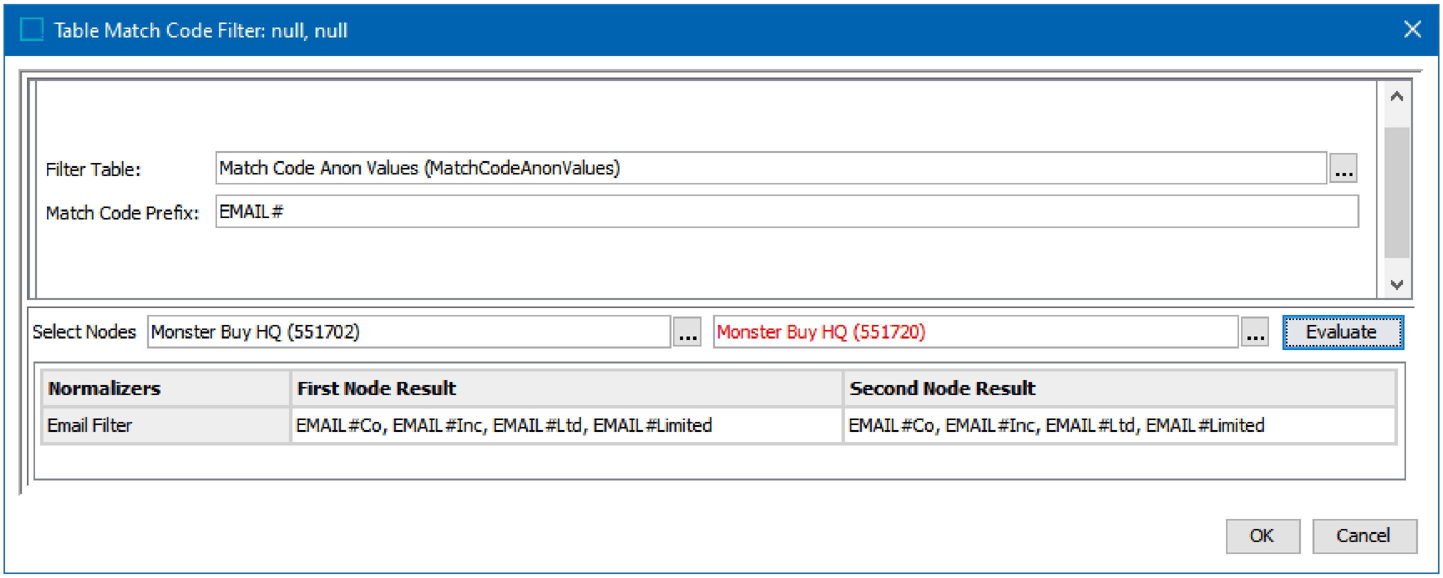


Table Match Code Filter: null, null

Filter Table: Match Code Anon Values (MatchCodeAnonValues)

Match Code Prefix: EMAIL#

Select Nodes: Monster Buy HQ (551702) | Monster Buy HQ (551720)

Normalizers	First Node Result	Second Node Result
Email Filter	EMAIL#Co, EMAIL#Inc, EMAIL#Ltd, EMAIL#Limited	EMAIL#Co, EMAIL#Inc, EMAIL#Ltd, EMAIL#Limited

Select the OK button to close out the Table Match Code Filter dialog. On the Decision Table dialog, the match code filter can be tested on specific nodes in the Evaluator section. In this example, two organization nodes are selected, the user clicks the Evaluate button, and then the filtered results are shown.

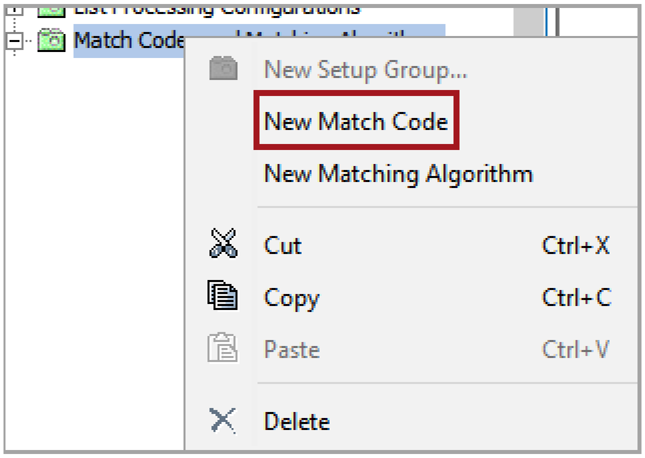
When satisfied with the filtering tests, select the Save button to store your changes.

Configuring an External Match Code

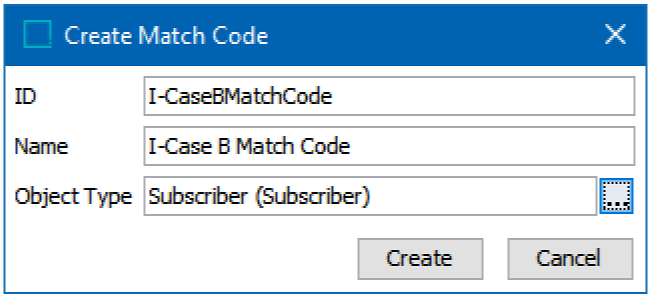
Match codes defined outside the matching algorithm are considered legacy functionality, but are still supported.

The following is the process for manually creating a match code for matching that can only be used for matching algorithms that have been created without the Embed Match Code checkbox selected. It is recommended to use the process described in the **Configuring Matching Algorithms** topic in this documentation.

1. In System Setup, right-click the node configured to house match codes and select New Match Code.



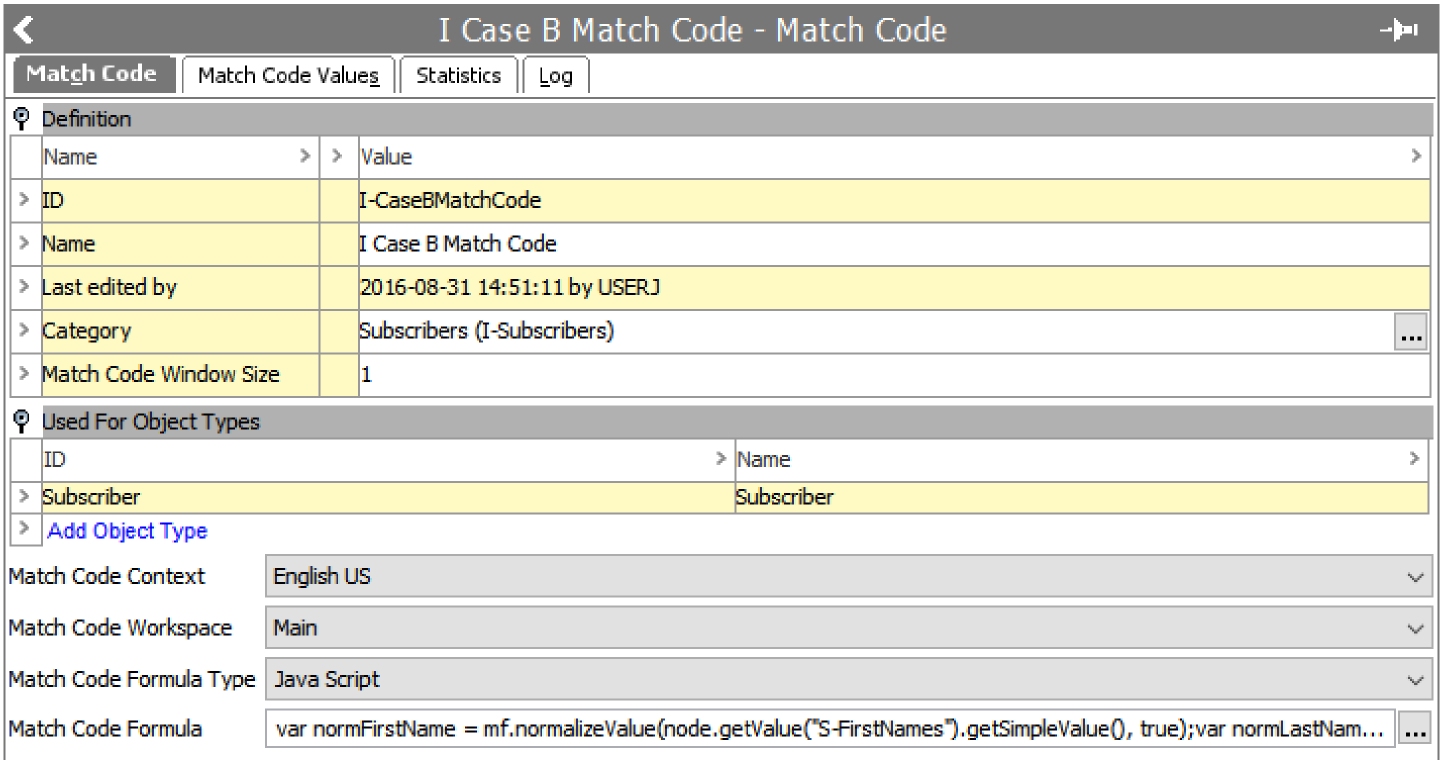
2. In the Create Match Code dialog, define an ID and name for the match code, specify an object type for which this match code applies, and click Create. Additional object types can be identified in the Match Code editor after creation.



Create Match Code

ID	I-CaseBMatchCode
Name	I-Case B Match Code
Object Type	Subscriber (Subscriber)

3. On the new match code editor, navigate to the Match Code tab and click the ellipsis button (...) in the Category field. In the selector that appears, select a node to indicate which objects will have match codes generated.



I Case B Match Code - Match Code

Definition

Name	Value
ID	I-CaseBMatchCode
Name	I Case B Match Code
Last edited by	2016-08-31 14:51:11 by USERJ
Category	Subscribers (I-Subscribers)
Match Code Window Size	1

Used For Object Types

ID	Name
Subscriber	Subscriber

Match Code Context	English US
Match Code Workspace	Main
Match Code Formula Type	Java Script
Match Code Formula	var normFirstName = mf.normalizeValue(node.getValue("S-FirstNames").getSimpleValue(), true);var normLastNam...

4. In the Match Code Window Size field, specify the window size to be used by the matching algorithm.
5. If additional object types are required, in the Used For Object Types section, use the Add Object Type link and selector to identify more object types for the match code.
6. In the Match Code Context field, specify in which context to run the match code formula. This is only required if the data is dimension dependent. By default, the current context will be selected.
7. In the Match Code Workspace field, specify in which workspace to run the match code formula. By default, Main workspace will be selected.

8. In the Match Code Formula Type field, specify JavaScript or Calculated as the format
9. In the Match Code Formula field, click the ellipsis button (...) to open up the formula editor and add your match code formula.

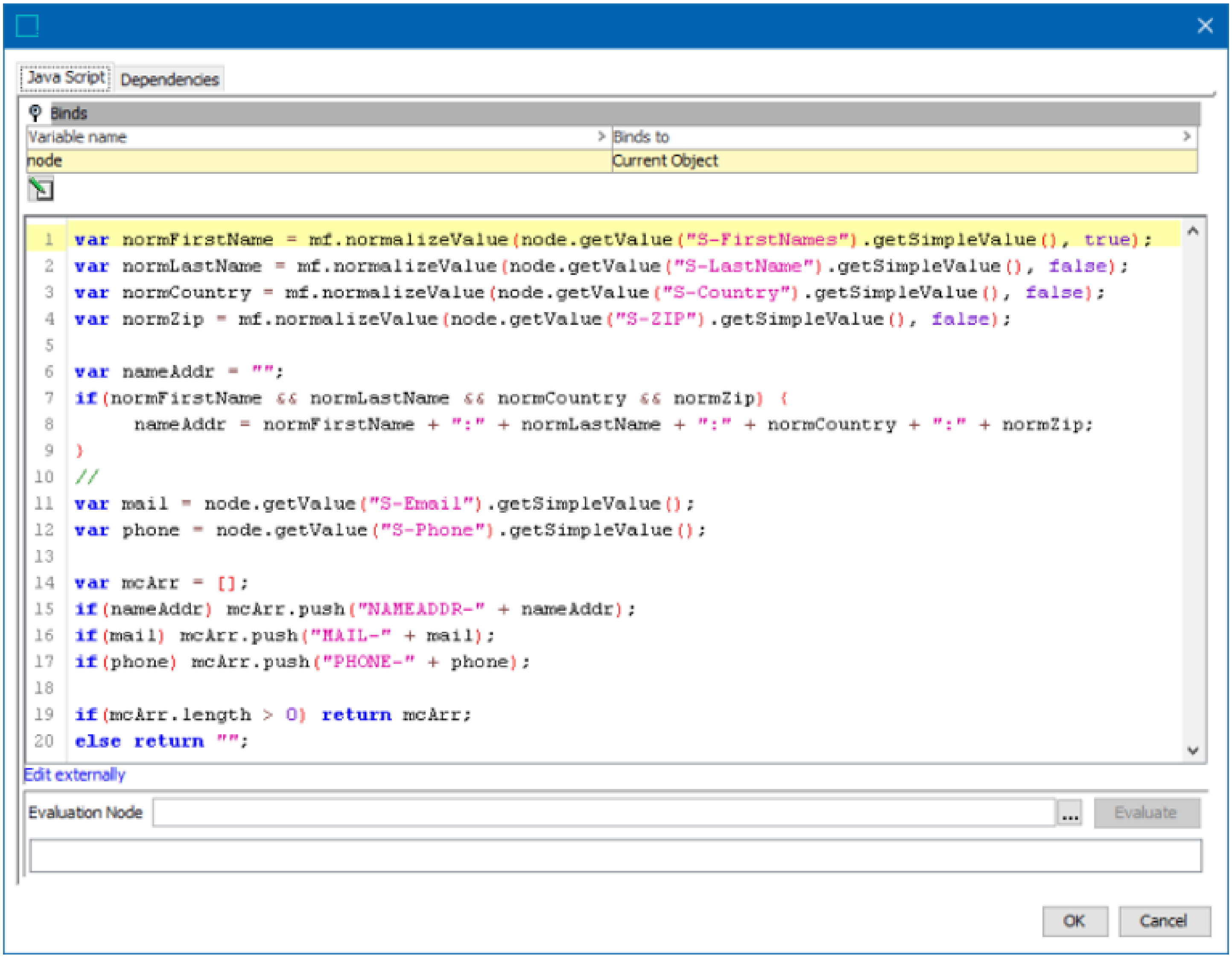
Binds for Match Code Formulas

It is also possible to make use of attributes and values that are created offline by binding them in the match code formula. This is used in cases of offline matching or matching records on import. Once inside the match code formula editor, open the Binds flipper, and click the Edit Binds button. You can declare variables and bind them to a variety of STEP elements / objects, as determined by the selected formula type.

JavaScript Match Code Formula

When using JavaScript, the current object should be bound to a variable. The ultimate goal should be to return the match code value of an object from the JavaScript. If a string is returned, it will be used as a match code value. If a JavaScript array is returned, all values in the array will be used as match code values for that object. Additional utility functions for match codes can be accessed by binding Matching Functions to, for example, the context variable in JavaScript or by binding 'Lookup Table Home' to, for example, 'lth.' For more information, see the **Text Functions** topic in the **Resource Materials** documentation.

Method	Description
context.soundex('Stibo')	Returns the Soundex.
context.metaphone3('Stibo')	Returns the primary value for the Metaphone 3.
context.metaphone3alternate ('Stibo')	Returns the alternate value for the Metaphone 3.
lth.getLookupTableValue('<asset-id>', 'LookupValue')	For more information, see the **Transformation Lookup Tables** topic in the **Resource Materials** documentation.



Variable name	Binds to
node	Current Object

```
var normFirstName = mf.normalizeValue(node.getValue("S-FirstNames").getSimpleValue(), true);
var normLastName = mf.normalizeValue(node.getValue("S-LastName").getSimpleValue(), false);
var normCountry = mf.normalizeValue(node.getValue("S-Country").getSimpleValue(), false);
var normZip = mf.normalizeValue(node.getValue("S-ZIP").getSimpleValue(), false);

var nameAddr = "";
if(normFirstName && normLastName && normCountry && normZip) {
    nameAddr = normFirstName + ":" + normLastName + ":" + normCountry + ":" + normZip;
}
//
var mail = node.getValue("S-Email").getSimpleValue();
var phone = node.getValue("S-Phone").getSimpleValue();

var mcArr = [];
if(nameAddr) mcArr.push("NAMEADDR-" + nameAddr);
if(mail) mcArr.push("MAIL-" + mail);
if(phone) mcArr.push("PHONE-" + phone);

if(mcArr.length > 0) return mcArr;
else return "";
```

Calculated Attribute Match Code Formula

When defining the formula via the calculated attribute language, all functions are available. An object's match code value can be a single string derived from the value of the formula, or it can be a list where all the values in the list are used as match code values for that object.

Below is an example of a simple STEP Function:

The match code value for each object will be a concatenation of the value for a Manufacturer attribute, the string ':' and the value for a ManufacturerPartNumber attribute. The Manufacturer value is normalized via a transformation lookup table with ID 'ManufacturerNormalization.'

```
concatenate(
  replacevaluebylookup("ManufacturerNormalization", value("Manufacturer")),
  ":",
  value("ManufacturerPartNumber")
)
```

If instead you wanted to return two match code values for each object, one for the Manufacturer and one for Manufacturer Part Number, each prefixed with either 'MAN-' or 'MPN-' could be done as follows (this example is without any normalization):

```
listconcatenate(
  concatenate("MAN-", value("Manufacturer")),
  concatenate("MPN-", value("ManufacturerPartNumber"))
)
```

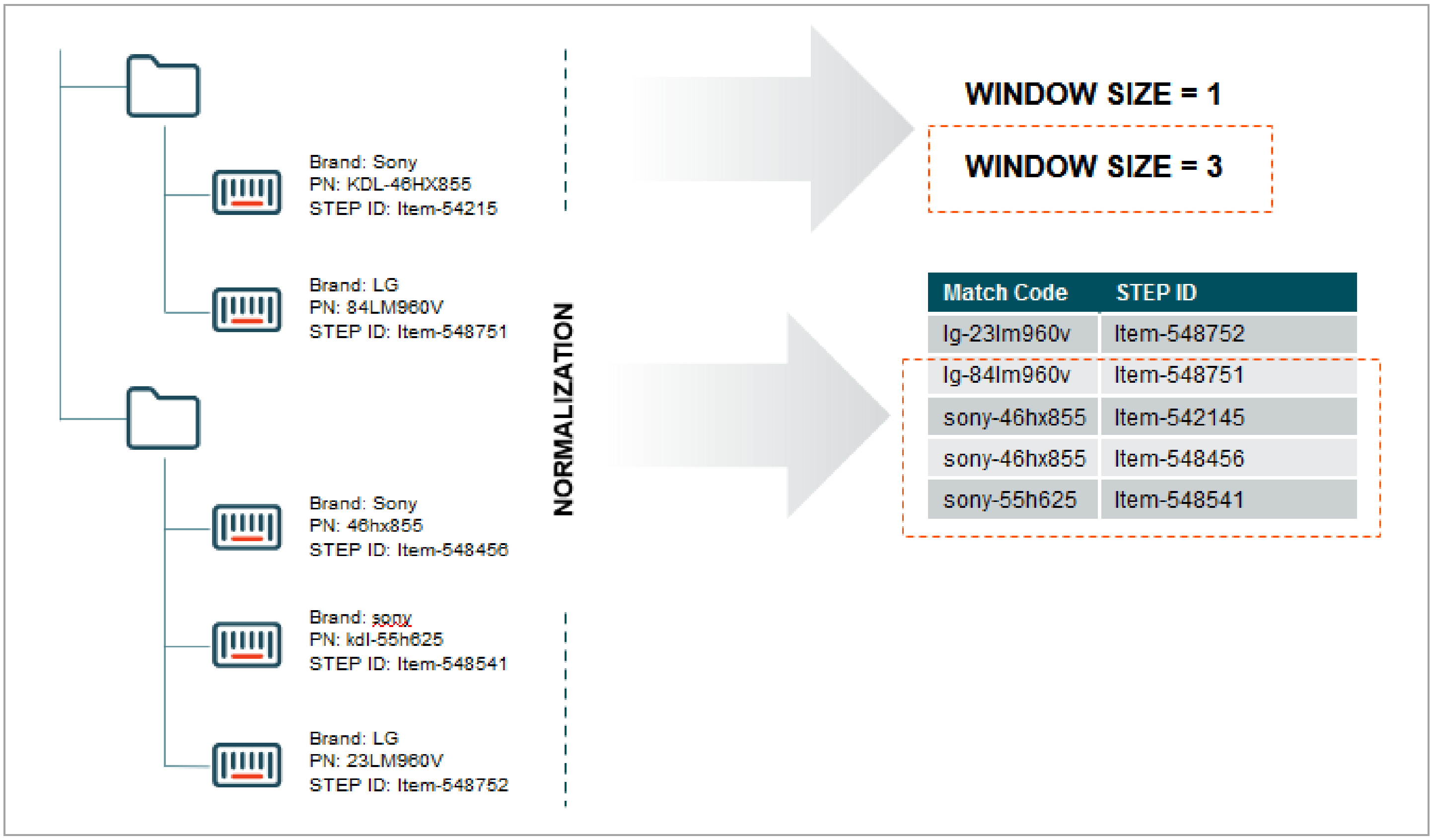
The reason for adding a prefix is to, when at all possible, avoid comparing objects with match code values from completely different domains.

Notice that in the examples above only rudimentary normalization is applied, and nothing is done to handle cases where values are missing. Since we would typically not want match code values only consisting of the hardcoded prefixes, below shows how checks for empty values could be added to the last example:

```
{
  man:= value("Manufacturer"),
  mpn:= value("ManufacturerPartNumber")
}
listconcatenate(
  if(len(man)!=0, concatenate("MAN-", man), ""),
  if(len(mpn)!=0, concatenate("MPN-", mpn), "")
)
```

Window Size

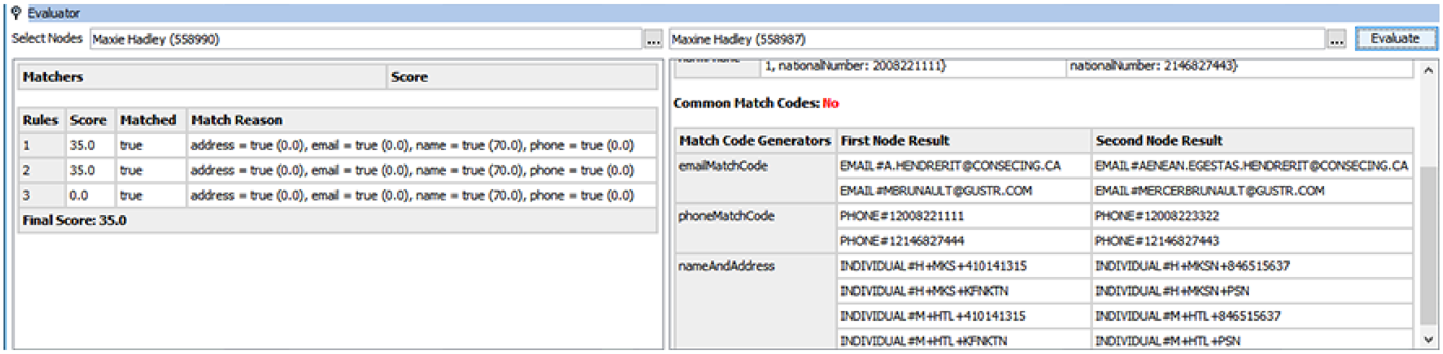
A legacy option on external match codes allows configuring match codes to include near-matches, using what is called a window. With a window size of '3,' Item-548456 would be compared to the object with the match code immediately prior to / following it in the list.



Match Code	STEP ID
lg-23lm960v	Item-548752
lg-84lm960v	Item-548751
sony-46hx855	Item-542145
sony-46hx855	Item-548456
sony-55h625	Item-548541

Evaluator

The evaluator is a tool for diagnosing unexpected results that may be encountered. In the evaluator, select two objects that you want to compare. It reports the results and provides detailed information about how the result was obtained. If additional details are required, the evaluators of the sub components can be used.



Maxie Hadley (558990)

Maxine Hadley (558987)

Matchers	Score

Rules	Score	Matched	Match Reason
1	35.0	true	address = true (0.0), email = true (0.0), name = true (70.0), phone = true (0.0)
2	35.0	true	address = true (0.0), email = true (0.0), name = true (70.0), phone = true (0.0)
3	0.0	true	address = true (0.0), email = true (0.0), name = true (70.0), phone = true (0.0)
Final Score: 35.0			

	1, nationalNumber: 2008221111}	nationalNumber: 2146827443}

Common Match Codes: No

Match Code Generators	First Node Result	Second Node Result
emailMatchCode	EMAIL #A.HENDRERIT@CONSECING.CA	EMAIL #AENEAN.EGESTAS.HENDRERIT@CONSECING.CA
	EMAIL #MBRUNAULT@GUSTR.COM	EMAIL #MERCERBRUNAULT@GUSTR.COM
phoneMatchCode	PHONE#12008221111	PHONE#12008223322
	PHONE#12146827444	PHONE#12146827443
nameAndAddress	INDIVIDUAL #H+MKS+410141315	INDIVIDUAL #H+MKSN+846515637
	INDIVIDUAL #H+MKS+KFNKTN	INDIVIDUAL #H+MKSN+PSN
	INDIVIDUAL #M+HTL+410141315	INDIVIDUAL #M+HTL+846515637
	INDIVIDUAL #M+HTL+KFNKTN	INDIVIDUAL #M+HTL+PSN

Other Match Criteria

Mostly used in legacy implementations, this section describes a number of still-supported alternatives to the decision tables. These match criteria cannot be used with a matching algorithm with embedded match codes.

String Comparison Algorithms

While developing your matching, linking, and merging strategy, a string comparison algorithm can be chosen to serve as the foundation for the matching process. The available string comparison algorithms include:

- **Levenshtein distance -** A metric for how many edits (substitution, insertion, deletion) it takes to make one string look like another. For example, the Levenshtein distance between the strings 'AXR55487' and '8XRT5487' is 2 because the first and fourth digits are different. In STEP terms, the strings would be 75 percent alike (6/8*100).
- **Damerau-Levenshtein distance -** Similar to Levenshtein distance except that the transposition of two adjacent characters counts only as one edit, not two. For example, the Levenshtein distance between the strings 'AA67' and 'A6A7' is 2 while the Damerau Levenshtein distance is 1.
- **Jaro / Jaro-Winkler distance -** Outputs 0 or 1 where 0 is no similarity and 1 an exact match. Note that these algorithms are available and can be made accessible in STEP via JavaScript, but are not currently included in the STEP core.

Note: The Levenshtein / Damerau-Levenshtein distance has to be converted into a percentage manually.

As is often the case, it may be that the preferred string comparison algorithm is not sufficient. To compensate for this, it is possible to define criteria that apply the Levenshtein / Damerau-Levenshtein distance directly to strings you build using STEP functions and automatically output an equality metric. Several criteria can be added and given weights that are used when calculating the total equality. The available criterion types are described as follows.

Multi Word Damerau-Levenshtein Distance

The Multi Word Damerau-Levenshtein distance is equal to the Damerau-Levenshtein distance except that the transposition of two words does not count as an edit. For example, the distance between 'Paul Johnson' and 'Johnson Paul' is 0. This criterion can come in handy when working with names where first name and surname are in the same attribute value yet the order differs from object to object.

Number Distance

The Number Distance criterion returns the relative distance between two numbers expressed as a percentage: lowest number / highest number * 100. For example, with this simple way of calculating a difference, the numbers 1 and 2 will be as different or equal as 50 and 100.

Special cases:

- If one or both strings are not numerical values, the criterion returns '0.'
- If only one of the strings is '0,' the criterion returns '0.'
- If both are '0,' the criterion returns '100.'
- If both strings are negative the calculation is the highest number / lowest number * 100.
- If one value is positive and the other negative, the criterion returns '0.'

The data to apply the number distance calculation to is generated via STEP functions.

JavaScript

The JavaScript criterion allows you to define your own algorithm for comparing objects. The only requirement is that the result is a number between 0 and 100 (as before, representing the percentage of equality).

From the JavaScript criterion, you can draw upon functions defined in business libraries in addition to six objects made available via bindings.

For more information, see the **JavaScript Binds** topic of the **Resource Materials** documentation.

Customizing Match Criteria with JavaScript Functions

In a lot of cases, expanding on the existing normalizers or matchers with functionality specific to the dataset and sources at hand will be necessary. The match criteria offers a number of places where this can be achieved, typically through JavaScript business functions. A range of tools are made available to such JavaScript functions, to support their implementation.

This section provides a number of example normalizers and matchers implemented in JavaScript to showcase some of the tools that are available. These functions can be drawn upon for both pure JavaScript matching algorithms and JavaScript in decision tables.

Important: The below functions are examples and likely cannot be used in their current form for your business case. Test thoroughly with your own data before implementing in your production STEP system.

normalizeValue

This `normalizeValue` function uses JavaScript and regular expressions to make a text lowercase and remove everything but letters and digits.

```
function normalizeValue(value) {

  if(value) {

    var normVal = value + "";

    normVal = normVal.toLowerCase();

    normVal = normVal.replace(/[^\w]|_/g, "");

    return normVal;

  }

  else {

    return "";

  }

}
```

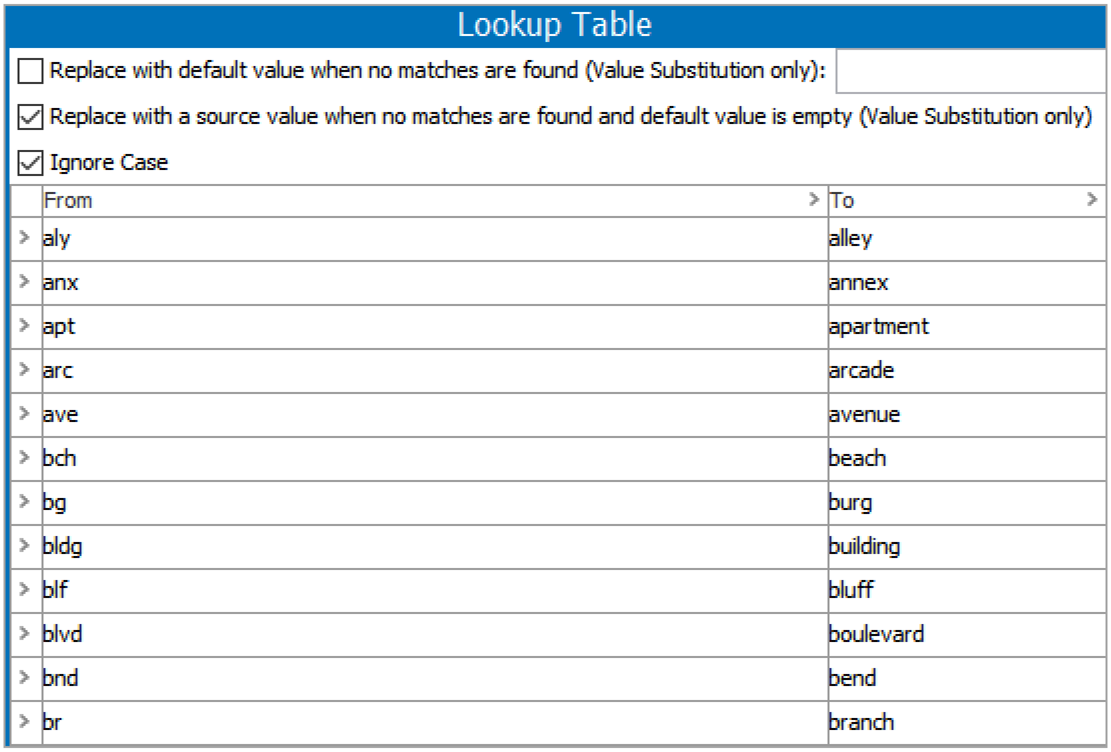
normalizeStreet

This function demonstrates how to access lookup tables. For more information on lookup tables, see the **Transformation Lookup Tables** topic in **Resource Materials** documentation.

The normalizeStreet function applies basic normalization to 'Street' values and uses a transformation lookup table with ID 'AddressAbbreviations' to replace common abbreviations like 'rd,' 'ave,' and 'ap' with their full word counterpart.

The logic reads:

- Convert input to JavaScript string,
- Convert to lowercase,
- Remove all instances of (.), (,), and (#) (more characters should probably be removed, but be careful removing dashes if used in street number ranges),
- Split the string by space characters and loop through the array of words applying the lookup table,
- Piece together the string again and return it.



Lookup Table

☐ Replace with default value when no matches are found (Value Substitution only):

☑ Replace with a source value when no matches are found and default value is empty (Value Substitution only)

☑ Ignore Case

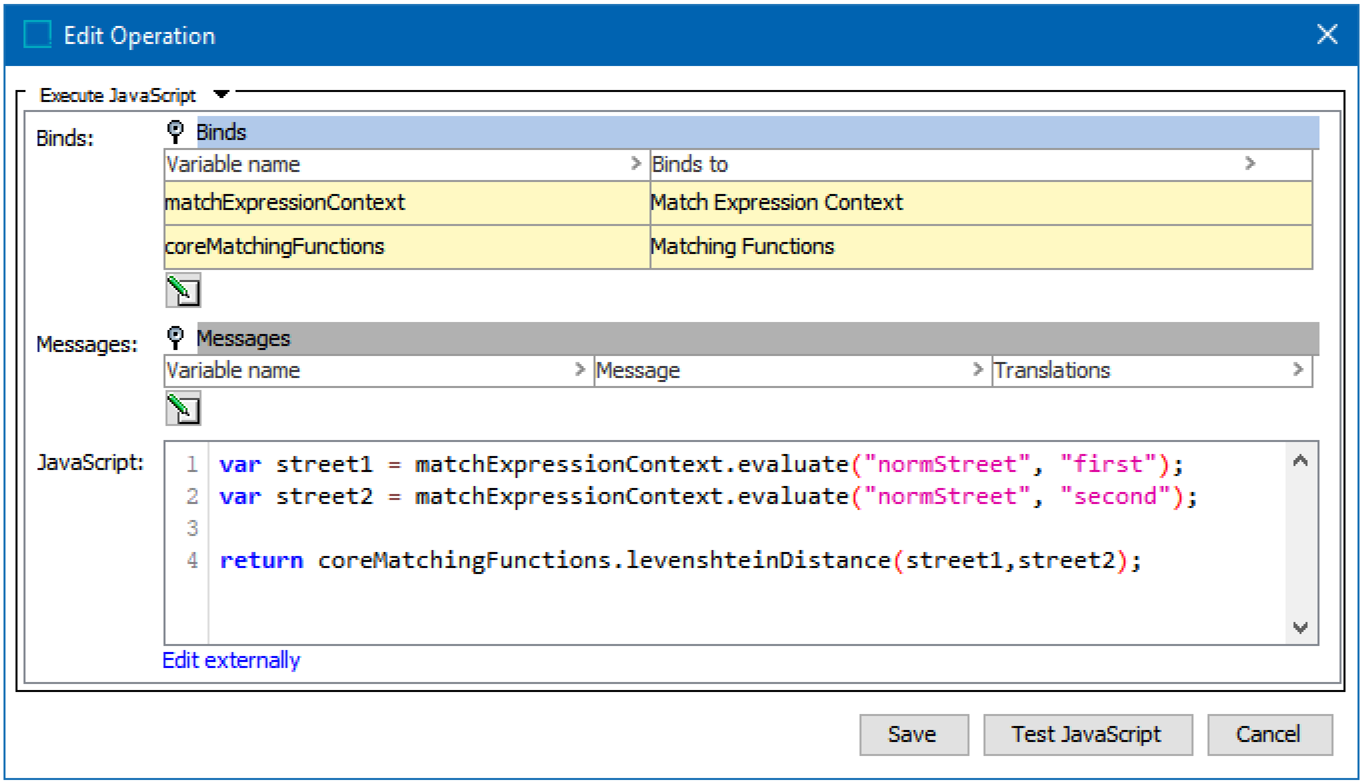
From	To
aly	alley
anx	annex
apt	apartment
arc	arcade
ave	avenue
bch	beach
bg	burg
bldg	building
blf	bluff
blvd	boulevard
bnd	bend
br	branch

```
function normalizeStreet(input, lookupTableHome) {
  var output = "";
  if(input) {
    input = input + "";
    input = input.toLowerCase();
    input = input.replace(/[\.\,#]|_/g, "");
    var inArr = input.split(" ");
    var outArr = [];
    for(var i = 0; i < inArr.length; i++) {
      outArr.push(lookupTableHome.getLookupTableValue("AddressAbbreviations", inArr
      [i]));
    }
    for(var j = 0; j < outArr.length; j++) {
      output += outArr[j];
      if(j != outArr.length - 1) {
              output += " ";
      }
    }
  }
  return output;
}
```

Core Matching Functions

The function below uses the built-in levenshteinDistance function to get the edit distance between normalized street values. 'Matching Functions' is bound to 'coreMatchingFunctions.'

```
var street1 = mec.evaluate("normStreet", "first");
var street2 = mec.evaluate("normStreet", "second");
return coreMatchingFunctions.levenshteinDistance(street1, street2);
```



Binds

Variable name	Binds to
matchExpressionContext	Match Expression Context
coreMatchingFunctions	Matching Functions

Messages

Variable name	Message	Translations

```
var street1 = matchExpressionContext.evaluate("normStreet", "first");
var street2 = matchExpressionContext.evaluate("normStreet", "second");

return coreMatchingFunctions.levenshteinDistance(street1,street2);
```

Match Actions

This page assumes you have read and understood the overview of matching, linking, and merging section, which provides an introduction as to how match actions fit into the bigger picture of this solution. For more information, see the **Matching, Linking, and Merging** topic in this documentation.

The choice of match action defines the entire workflow around the golden records. The following match actions exist:

- Identify Duplicates
- Match and Link
- Match and Merge
- List Processing Deduplicate Records

Identify Duplicates

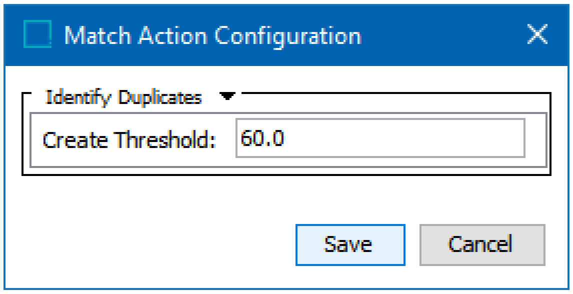
The Identify Duplicates match action helps determine if duplicates exist in a dataset and allows users to confirm, reject, merge, and delete duplicates manually with only limited impact on existing functionality.

Using the Identify Duplicates match action, the matching algorithm will not automatically do anything to identify duplicates. It is possible to set up workflows and UIs for manually merging identified duplicate records in STEP, but if those setups are needed, the Identify Duplicates match action is probably not the best match action.

With the Identify Duplicates match action, as matchable objects are created and modified, events are sent to a matching event processor. In an asynchronous process, the Match Event Processor matches these objects with other matchable objects, as defined by the relevant matching algorithm. When two objects score above the create threshold, a match result is stored for future handling.

Configuration

For the Identify Duplicates match action, the Create Threshold parameter is required and specifies how equal objects must be to be identified as possible duplicates.



Create Threshold: 60.0

Identify Duplicates in Workbench

Identify duplicates can make use of many of the same workbench and Web UI tools as the match and link match action. For more information, see the **Match and Link in Workbench** topic in this documentation.

Identify Duplicates in Web UI

The Web UI supports a number of actions on identified duplicates. See the list below for Web UI topics relevant to the Identify Duplicates solution:

- **Potential Duplicates List** - See the **Match and Link, Potential Duplicates List** topic in the **Web User Interfaces / Web UI Getting Started** documentation.
- **Merging Confirmed Matches** - See the **Match and Link, Merging Confirmed Matches** topic in the **Web User Interfaces / Web UI Getting Started** documentation.
- **Configuring a Deduplication Clerical Review** - See the **Match and Link, Configuring a Deduplication Clerical Review** topic in the **Web User Interfaces / Web UI Getting Started** documentation.

Match and Link

Using an asynchronous process, Match and Link creates and maintains a set of 'golden records' as an aggregation of matching 'source records'.

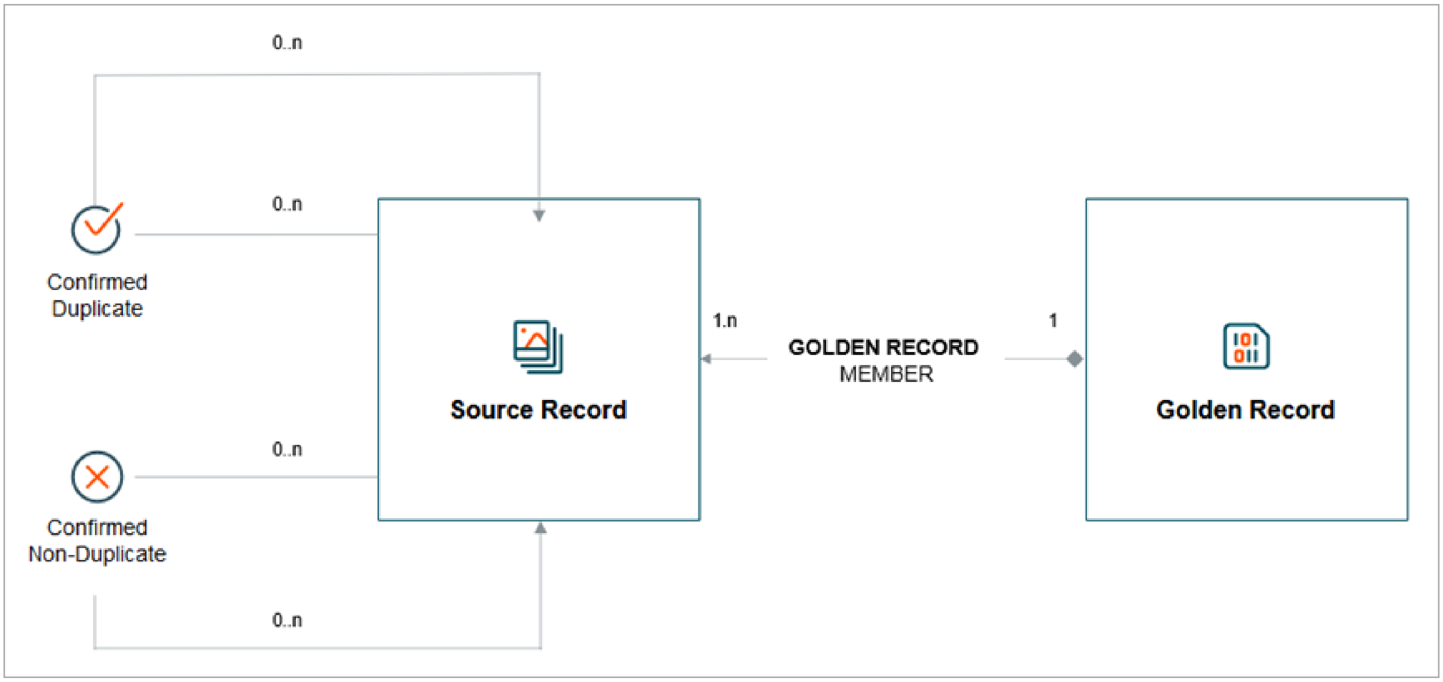
- In Product MDM, Match and Link is commonly used in automating the creation and maintenance of sell-side products as golden records, based on buy-side products as source records.
- In Customer MDM, Match and Link is commonly used for resolving household entities as golden records using individual customer entities as source records.

A detailed setup using Match and Link is described in the **Module - Vendor Data Onboarding** section of the **PIM for Retail** section of the **Product MDM Solution Enablement** documentation.

Data Model

In a Match and Link solution, source records and golden records will be separate records of different object types.

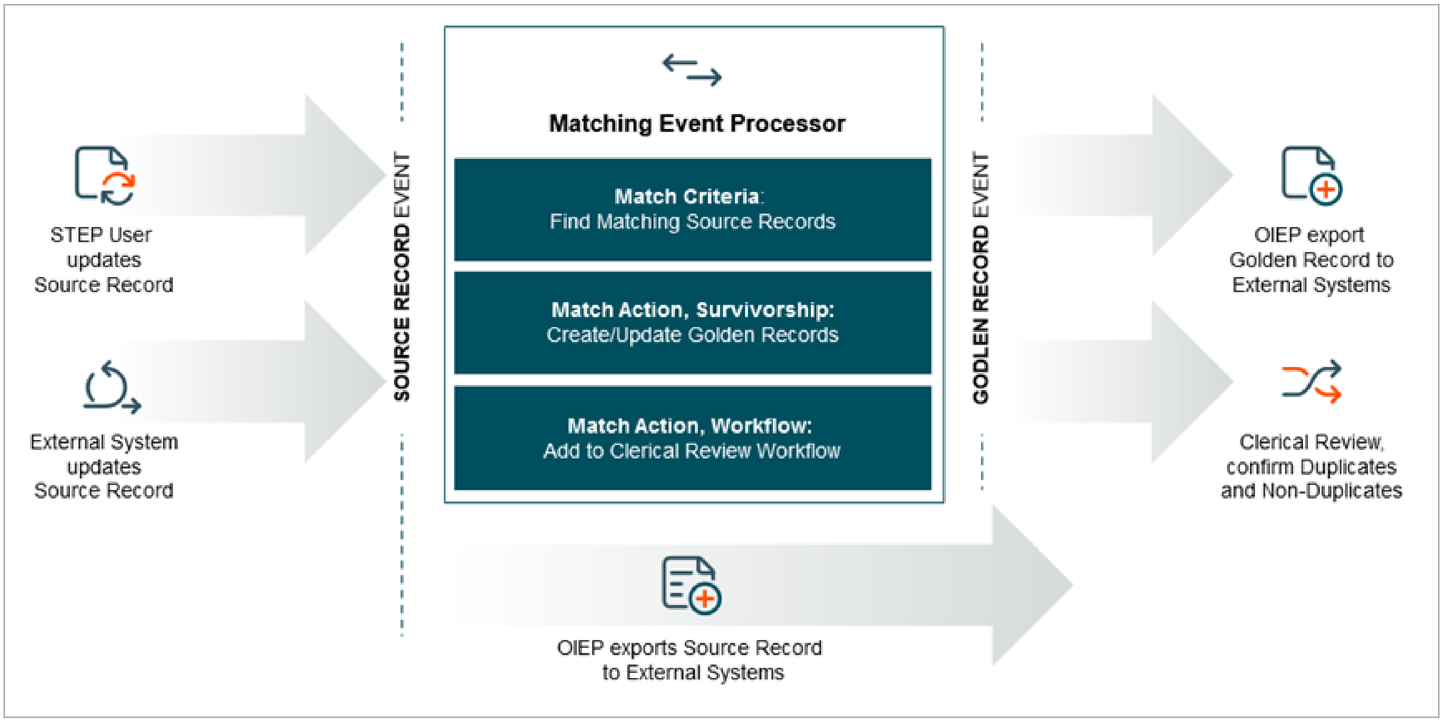
The golden records are created by survivorship rules, and every source record belongs to exactly one golden record.



Confirming a duplicate or non-duplicate in a Match and Link solution results in a reference being created on the source record level. In the Match and Link solution, the Confirmed Duplicate is a reference between two source records which permanently identifies two specific source records as duplicates. The Confirmed Non-Duplicate is the opposite, permanently confirming that two source records should never belong to the same golden record object.

Information Flow

When a user or a source system updates a source record, events are written to a Matching event processor. The Matching event processor lets the matching algorithm run a match on the source record against all other existing source records.


STEP User updates Source Record
External System updates Source Record
SOURCE RECORD EVENT
Matching Event Processor
Match Criteria: Find Matching Source Records
Match Action, Survivorship: Create/Update Golden Records
Match Action, Workflow: Add to Clerical Review Workflow
GODLEN RECORD EVENT
OIEP export Golden Record to External Systems
Clerical Review, confirm Duplicates and Non-Duplicates
OIEP exports Source Record to External Systems

Source records with a match score above an Auto-Link Threshold will be linked to the same golden record. The golden record will be updated with information from all linked source records, according to a set of survivorship rules. For more information, see **Survivorship in Match and Link** topic in this documentation. The resulting golden record updates can trigger events that export the golden record to external systems.

Records with match scores between the Auto-Link Threshold and the Clerical Review Threshold are added to a Clerical Review Workflow. This allows a data steward user to manually identify if this is a Confirmed Duplicate or a Confirmed Non-Duplicate. A decision by the data steward is considered an update to the source record and will invoke the flow again.

The golden record should be considered a system-owned object. Users should not perform manual updates to the golden record since survivorship rules overwrite this information and the golden record may be deleted by the Matching event processor.

It is common to enrich golden records with information through an additional 'internal data' source record, (sometimes referred to as a 'silver record' or an 'enrichment record') that is created and maintained in association to the golden record.

Information from an internal data source record is promoted to the golden record with survivorship rules by the Matching event processor.

Internal Data Source Objects

In Match and Link setups, there is often a need to maintain data on the golden record. Since the golden record is a system-owned object, data maintenance is performed on 'enrichment records' or 'internal data source objects' according to the following rules:

- A unique object type is required, one that is different from the object types of golden record and other source objects.
- Do not generate match codes for internal data source objects.
- In the Matching component model configuration, Source Object Type aspect, add the object type of the internal data source object.
- Golden records should use the same reference types for internal source objects and for other source objects.

To update the golden record automatically when an internal data source object changes:

1. Configure the event processor to listen on events for internal data source objects.
2. Create a business action to find the golden record for the internal data source object, identify one of the other source objects for the golden record, and then generate an event for that object for the event processor.
3. Create an event filter condition that is always false since the original event for the internal data source object will not go onto the queue.

User Actions

Match and Link is supported by a range of tools in workbench and Web UI so the expert user can analyze the results of the matching algorithm and take actions.

The Match and Link specific actions are:

Confirm Duplicates: If two objects are confirmed as duplicates, a reference of the 'Duplicate Reference Type' specified in the component model and in the matching algorithm will be created, the pair will be removed from the 'Match Result' tab, and instead, will show up on the 'Confirmed Duplicates' tab on the matching algorithm.

Confirm Non Duplicates: If two objects are rejected as being duplicates, a reference of the 'Non-Duplicate Reference Type' will be created and the pair will be shown on the 'Confirmed Non Duplicates' tab on the matching algorithm.

It is important to understand that if a pair has been confirmed as duplicate / non-duplicate, the pair will not be considered when the matching algorithm is reapplied, regardless if the data on the objects has changed. The confirmed duplicate / non-duplicate relationship can be updated either via the 'Remove From List' options or by deleting the references.

Manual Merge of source records: If by Identify Duplicates or by 'Link golden record' two objects are confirmed as duplicates, it is possible to manually merge them into a single object.

Configuring Match and Link

This configuration section assumes you already have set up the Matching Component model and match algorithm with match Criteria. For more information, see the **Match Criteria Configuration** topic in this documentation.

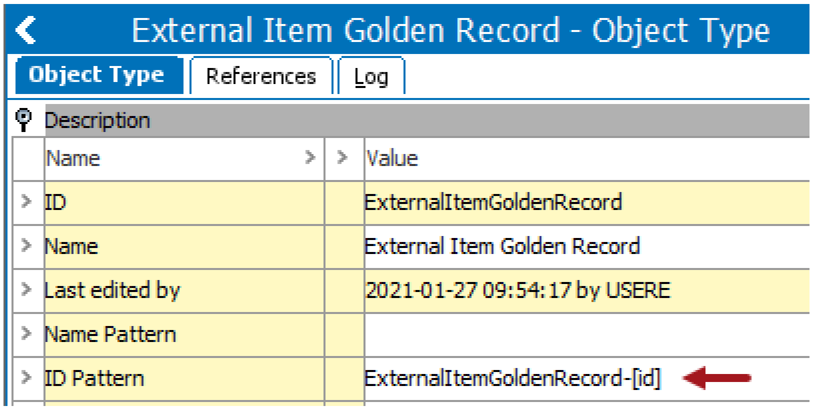
Several elements are required to configure a match algorithm to use the Match and Link match action:

- Set up a match and link component model
- Configure the link golden record match action for the matching algorithm
- Set up an event processor; see match and link event processor section of this topic
- Set up survivorship rules; see the **Survivorship in Match and Link** topic in this documentation.
- Consider setting up match and link in Web UI

Users must add a clerical review widget to the homepage. For more information, see the **Adding Widgets to a Homepage** topic in the **Web User Interfaces / Web UI Getting Started** documentation. For more information on how to optimize the match and link configuration, see the **Matching and Linking Recommendations** topic in this documentation.

Configuring Link Golden Record Object Type

Link golden records automatically created by the matching functionality should be of a different object type than the source objects, e.g., 'ExternalItemGoldenRecord'. The golden record object type must have an auto ID pattern configured.

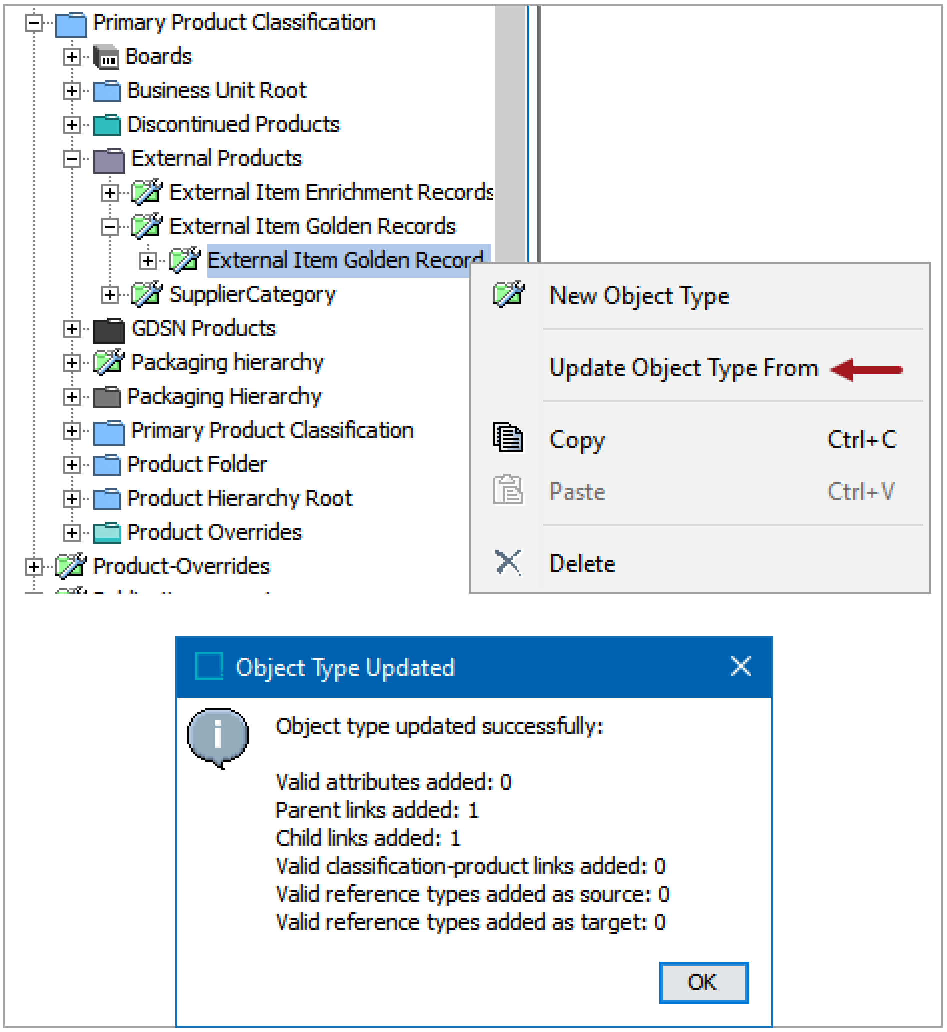


External Item Golden Record - Object Type

Description

Name	Value
ID	ExternalItemGoldenRecord
Name	External Item Golden Record
Last edited by	2021-01-27 09:54:17 by USERE
Name Pattern	
ID Pattern	ExternalItemGoldenRecord-[id]

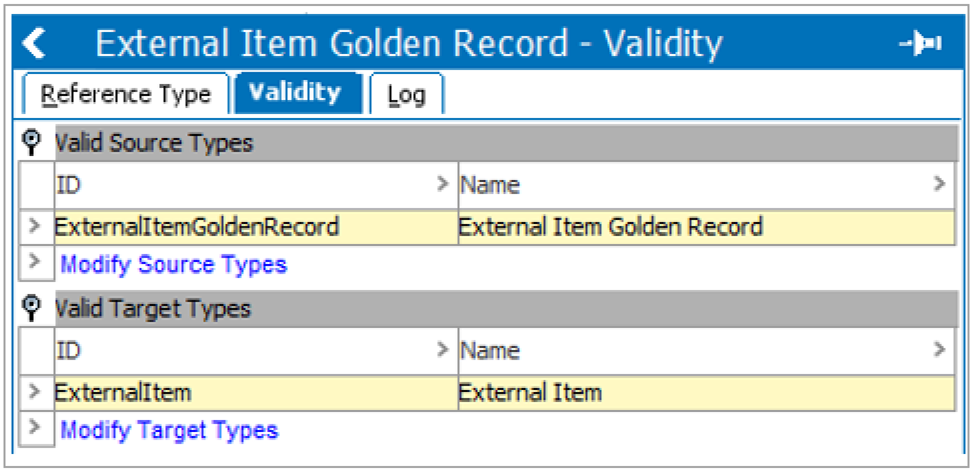
From a validity perspective, if users intend to copy all data from source records, including attribute values and references, the golden record object type should have the same valid attributes and be a valid source for the same reference / link types. An easy way to secure this is via the 'Update Object Type From' context menu option as illustrated in the image below.



Object type updated successfully:

Valid attributes added: 0
Parent links added: 1
Child links added: 1
Valid classification-product links added: 0
Valid reference types added as source: 0
Valid reference types added as target: 0

For golden records to refer back to their source objects, a golden record reference type must be created. The reference type must allow for multiple targets and should be valid from the golden record object type to the source object type.



Valid Source Types

ID	Name
ExternalItemGoldenRecord	External Item Golden Record

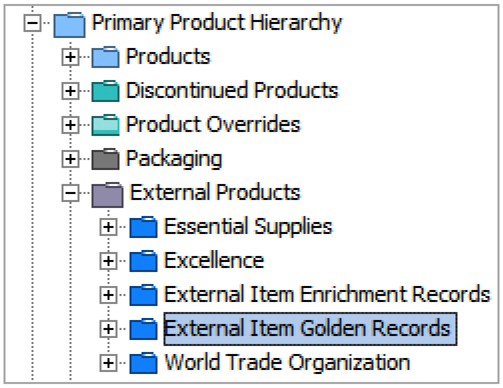
Modify Source Types

Valid Target Types

ID	Name
ExternalItem	External Item

Modify Target Types

In addition to this, a root node for the golden records must be created. Initially, all golden records will be created as immediate children of this node.



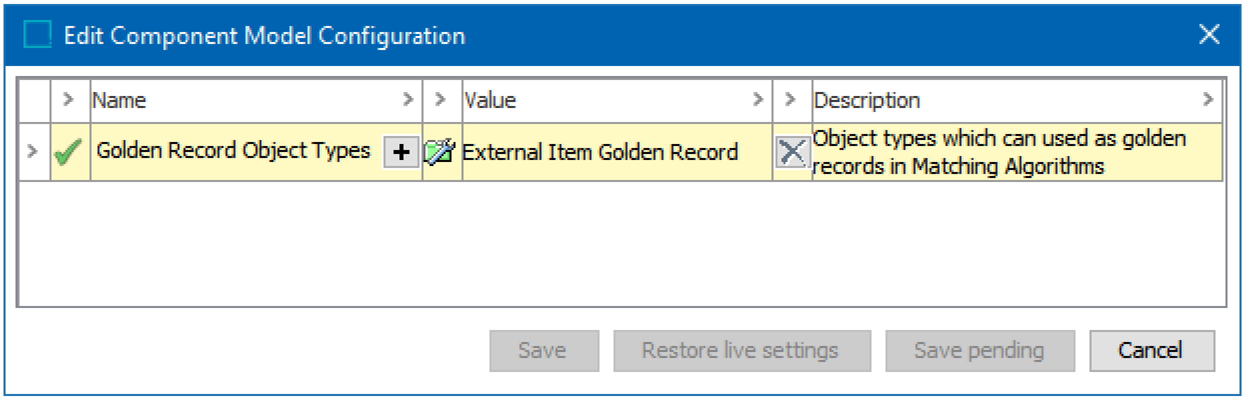
- Primary Product Hierarchy
  - Products
  - Discontinued Products
  - Product Overrides
  - Packaging
  - External Products
    - Essential Supplies
    - Excellence
    - External Item Enrichment Records
    - External Item Golden Records
    - World Trade Organization

Match and Link Component Model

The 'Matching - Link Golden Record' component model lists all the golden record object types applicable to the link golden record solution.

1. In System Setup, expand 'Component Models' and click on the 'Matching - Link Golden Record' node.
2. On the 'Component Model Configuration' tab, click the Edit link.
3. Click the 'plus' button in the 'Golden Record Object Types' field, and in the selection dialog, choose the object types you want to use for golden records in the system. Only the object types that you select can be used as golden records for link golden record configurations. Furthermore, on objects of these object types, the Golden Record tab is automatically enabled. The Golden Record tab shows the golden record together with its member records.

> **Note:** The chosen object types must have all the attribute's reference types and data container types valid for it that survivorship rules promote from source records.



Edit Component Model Configuration

Name	Value	Description
✓ Golden Record Object Types	External Item Golden Record	Object types which can used as golden records in Matching Algorithms

Click the 'X' button to remove the relevant value from the component model. A green checkmark will appear if the component model has a valid configuration.

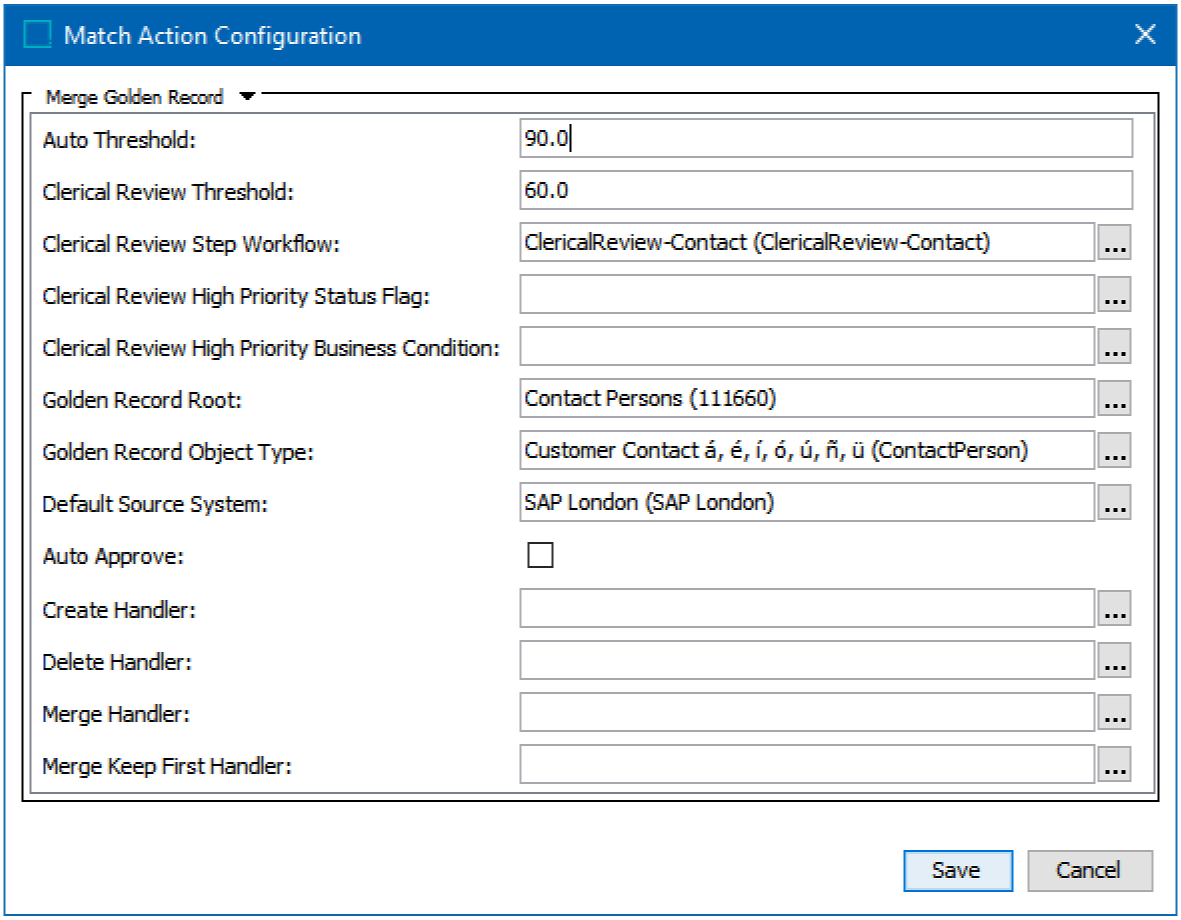
4. Click Save to save changes.

> **Note:** If you need to navigate away from the configuration dialog and the component model is not yet valid (it has an 'X' instead of a checkmark), click Save pending to save your work.

Link Golden Record Match Action

To set up the link golden record match action, follow these steps:

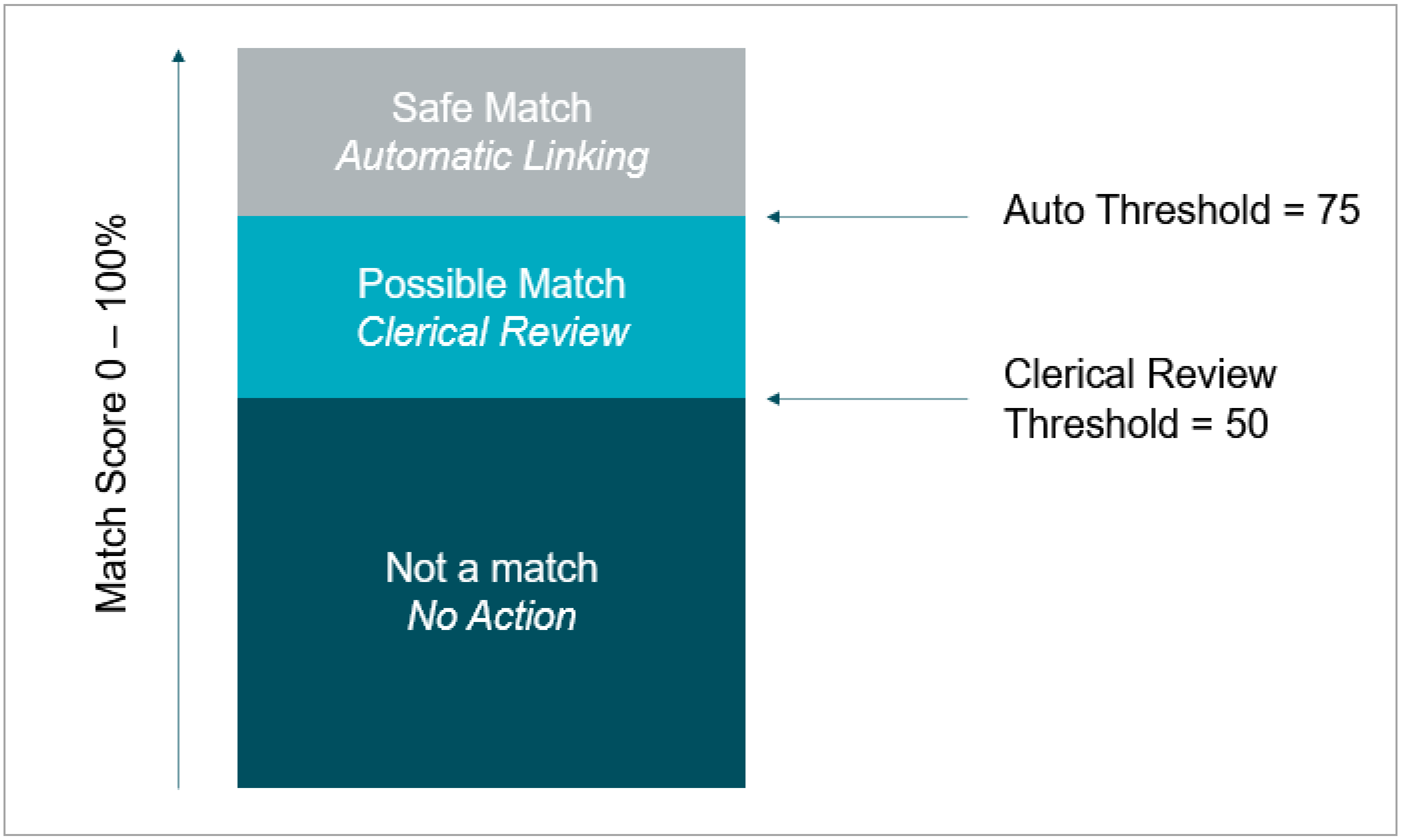
1. On the matching algorithm node, navigate to the Matching Algorithm tab.
2. In the Match Action area, click the Edit link. Once on the Match Action Configuration dialog, several parameters must be configured.



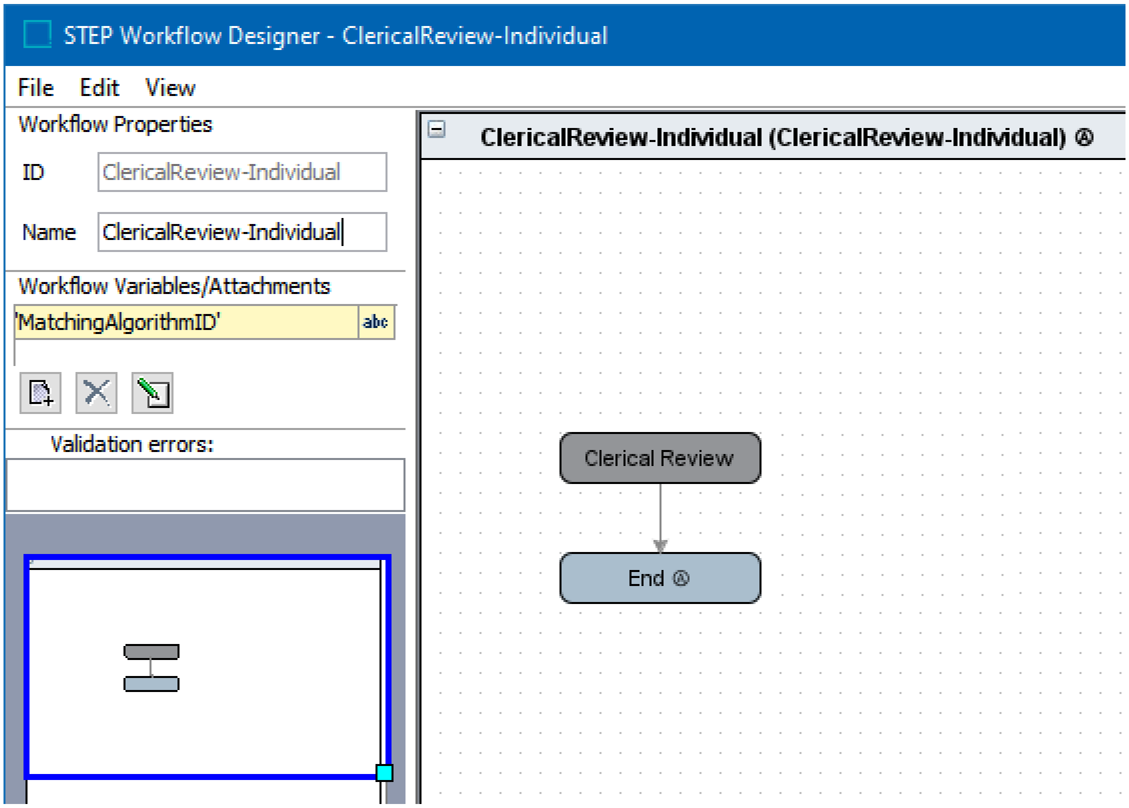
Auto Threshold:	90.0
Clerical Review Threshold:	60.0
Clerical Review Step Workflow:	ClericalReview-Contact (ClericalReview-Contact)
Clerical Review High Priority Status Flag:	
Clerical Review High Priority Business Condition:	
Golden Record Root:	Contact Persons (111660)
Golden Record Object Type:	Customer Contact á, é, í, ó, ú, ñ, ü (ContactPerson)
Default Source System:	SAP London (SAP London)
Auto Approve:	□
Create Handler:	
Delete Handler:	
Merge Handler:	
Merge Keep First Handler:	

3. For the 'Auto Threshold' and 'Clerical Review Threshold' parameters, specify how matches will be evaluated via an equality measurement.
   - The 'Auto Threshold' specifies how equal two source objects have to be to automatically have them linked to the same golden record.
   - The 'Clerical Review Threshold' should be equal to or less than the 'Auto Threshold'. It specifies how equal two objects must be to be considered potential duplicates.

- Objects with an equality metric between the 'Clerical Review Threshold' and the 'Auto Threshold' will be flagged as potential duplicates and enter a clerical review workflow together. If a user then confirms that the objects are in fact duplicates, they will be linked as Confirmed Duplicates, and always be part of the same golden record. Rejected duplicate source records will be linked as Confirmed Non-duplicates, and never contribute to the same golden record.



4. In the 'Clerical Review STEP Workflow' parameter, Click the ellipsis button (...), and select the relevant clerical review workflow. A clerical review workflow can be as simple or elaborate as needed.



STEP Workflow Designer - ClericalReview-Individual

Workflow Properties

ID: ClericalReview-Individual

Name: ClericalReview-Individual

Workflow Variables/Attachments

'MatchingAlgorithmID'

Validation errors:

ClericalReview-Individual (ClericalReview-Individual)

For more information, see the **Creating a Workflow** topic in the **Workflow** documentation.

5. In the Clerical Review High Priority Status Flag parameter, click the ellipsis button (...) and select the STEP workflow status flag that is used to designate high priority tasks in the clerical review workflow.
6. In the Clerical Review High Priority Business Condition parameter, click the ellipsis button (...) and select the business condition that is used to verify if a task is of high priority.

Note: If a Status Flag has been configured, but a Business Condition has not, then the status flags behave as if the theoretical Business Condition evaluated to true. If a Business Condition has been configured, and a Status Flag has not, the Business Condition is ignored.

The Business Condition is evaluated on each object in the Clerical Review task (each potential duplicate) in the context of the matcher and will have access to the Current Object binds.

Note: Though the business condition runs as a part of matching and it involves a clerical review, no matching or Workflow binds are available.

Important: When Status Flags are used in this way, the matching algorithm determines which Status Flags are set (or not set). Due to this, no other Status Flags should be configured in the Clerical Review Workflow.

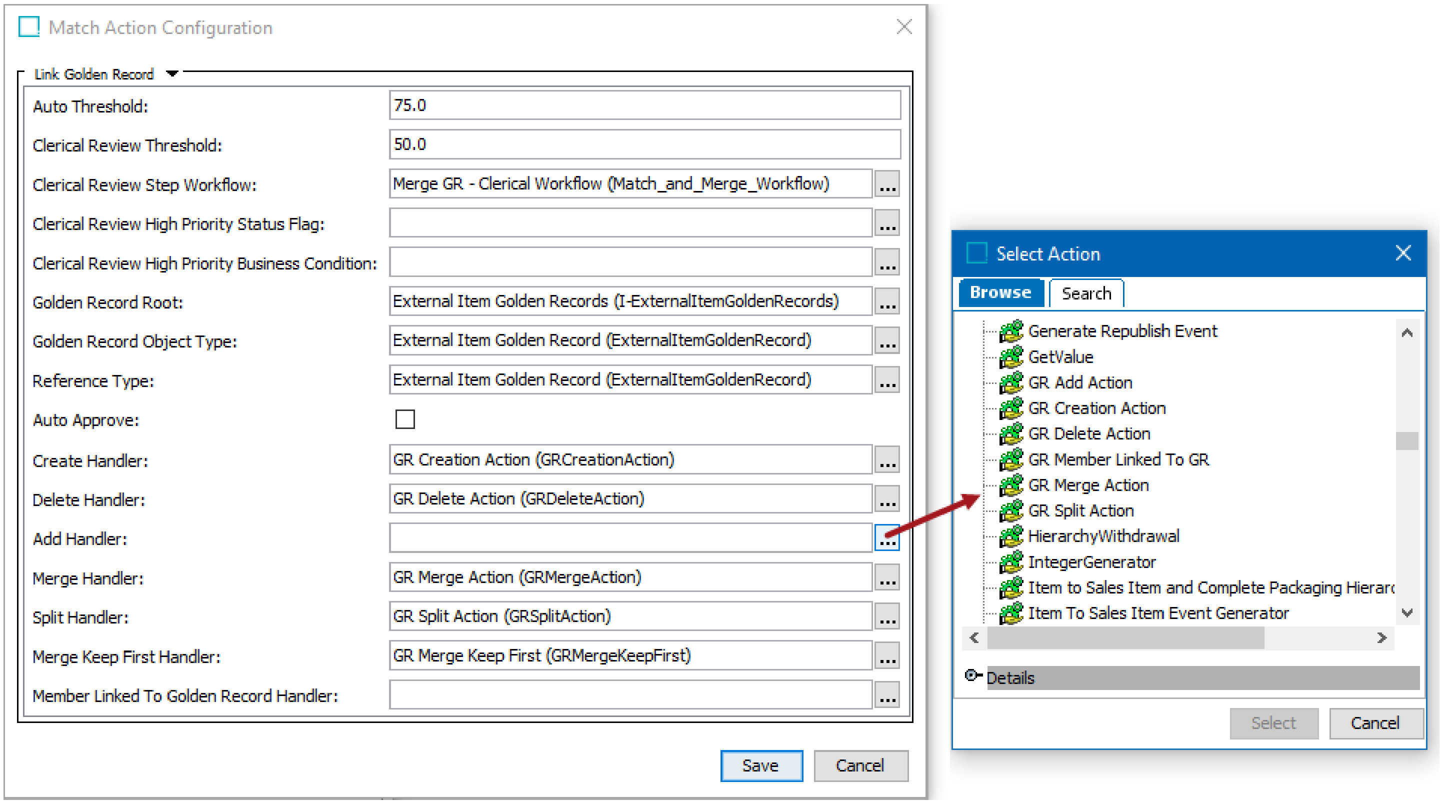
7. For the Golden Record Root and Golden Record Object Type parameters, specify the root node under which golden records should be stored and the golden record object type.
8. Auto Approve is used to automatically approve the golden records being created.

Note: Match and Merge does support the import of records without source system references.

Match Action Handlers for Match and Link

Working with a golden record setup often requires specific actions when a golden record is changed (created, deleted, merged, split, etc.). In these cases, the matching algorithm can be configured to call a business rule via a handler in order to allow for more granular processing of events. For example, when two existing golden records are merged, additional actions may need to occur in addition to the survivorship rules.

The following handlers are available for the matching and linking solution:



Match Action Configuration

Link Golden Record

Auto Threshold:	75.0
Clerical Review Threshold:	50.0
Clerical Review Step Workflow:	Merge GR - Clerical Workflow (Match_and_Merge_Workflow)
Clerical Review High Priority Status Flag:	
Clerical Review High Priority Business Condition:	
Golden Record Root:	External Item Golden Records (I-ExternalItemGoldenRecords)
Golden Record Object Type:	External Item Golden Record (ExternalItemGoldenRecord)
Reference Type:	External Item Golden Record (ExternalItemGoldenRecord)
Auto Approve:	☐
Create Handler:	GR Creation Action (GRCreationAction)
Delete Handler:	GR Delete Action (GRDeleteAction)
Add Handler:	
Merge Handler:	GR Merge Action (GRMergeAction)
Split Handler:	GR Split Action (GRSplitAction)
Merge Keep First Handler:	GR Merge Keep First (GRMergeKeepFirst)
Member Linked To Golden Record Handler:	

Create Handler: The selected business action runs on the golden record after it has been created and has initial source object links, but before survivorship rules run.

Delete Handler: The selected business action runs after the golden record is deleted. For example, when merging two golden records, one is deleted. The delete handler runs after the merge handler, which means that the golden record has no linked source records. Alternatively, in this case, if the delete handler field is blank, then the incoming references of the surviving golden record are re-targeted and re-approved (if they were approved before); the golden record is deleted and, if auto-approve is enabled, the deletion is approved.

Add Handler: The selected business action runs on the golden record after a new source is added, but before any survivorship rules run.

Merge Handler: The selected business action runs when two golden records are merged (because their sources match). The source(s) are moved to the golden record that will be kept and the delete handler is called for the golden record that will be deleted.

Split Handler: The selected business action runs when a golden record is split (because one or more of its sources no longer match). The split handler runs after the new golden record is created and its source record links are updated, but before survivorship rules run. The original and new golden records each reflect the correct source records. The create handler is not called when golden records split.

Merge Keep First Handler: The selected business condition runs when two golden records are being merged and allows identification of the golden record that should be kept. Use the Current Object and Secondary Object binds in the condition and return one of the following options:

- null = default behavior; keep the golden record with most members—f there is an equal number, keep the oldest golden record
- true = golden record bound to Secondary Object is deleted
- false = golden record bound to Current Object is deleted

Member Linked to Golden Record Handler: The selected business action runs on the source object when a source object link changes from one golden record to a new golden record. The handler runs after the sources have been added, but before survivorship rules run.

Match and Link Event Processor

Match and link match algorithm is run in the context of an event processor configured to trigger the matching algorithm.

To create an event processor:

1. Create a matching event processor by following the steps outlined in the **Creating an Event Processors** topic of the **System Setup / Super User Guide** documentation.

   This section guides you to select the type of processing required (generate match code and update and/or run matching algorithm), the matching algorithm(s), establish the schedule, set the queue status, and set the event triggering definitions.

   Enable the matching event processor by following the steps outlined in the **Enabling Event Processors** section of the **Running an Event Processors** topic in the **System Setup / Super User Guide** documentation.

> **Important:** While it is possible to use the same match and link matching algorithm across several event processors, that will usually result in an optimistic locking and/or unique constraint violation when the two processors conflict with one another. Optimistic locking is the implementation where objects are modified concurrently. The result of this is that the system cannot know which value should be represented. To avoid these issues, ensure that each algorithm on the system is run by a single event processor.

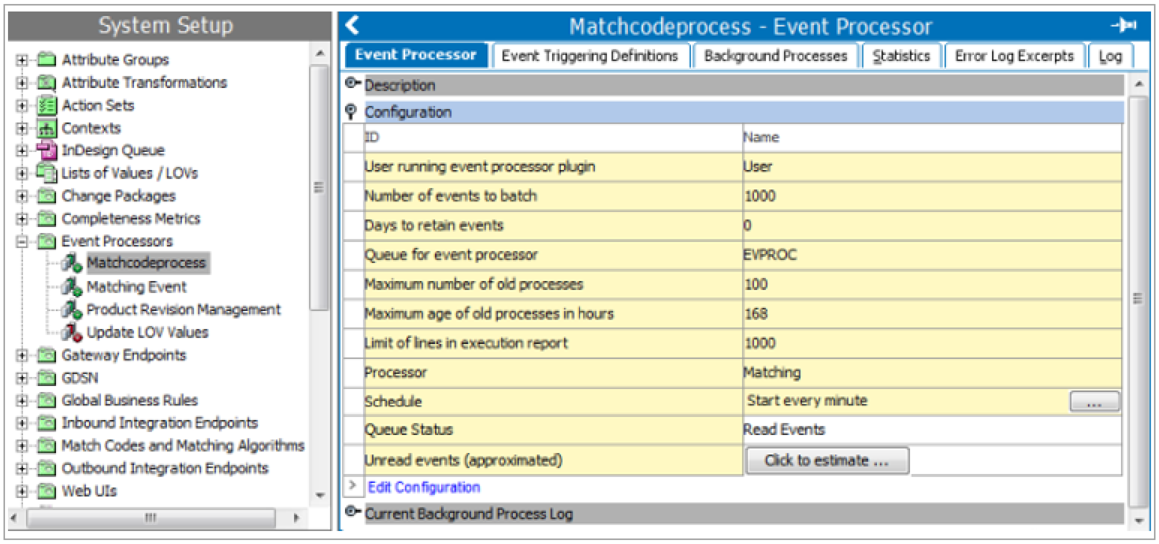
Once set up, an event processor monitors the system for actionable events on specified objects and ensures match codes are regenerated, and runs the matching algorithms in response to any relevant change. For example, consider an object that is subject to a matching algorithm. When the match code assignment or data on that object is approved, the approval can trigger the event processor to regenerate the match code for that object and run the algorithm. Alternatively, events could be passed to the event processor via a republish business rule as part of a workflow or integration.

Event processors keep a background process log, so you can determine when events were processed and what actions were taken in response. Additionally, event processor performance measurements are available on the Statistics tab for both matching algorithms and match code configurations.

Match and Link Event Processor Example

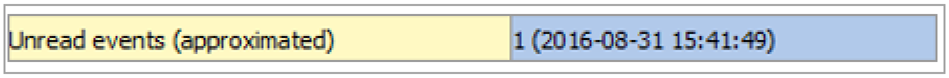
The following example event processor has been configured to:

- Run every minute
- Be triggered by changes on the object types and attributes defined in the match code 'I Case B Match Code'
- Regenerate match codes using 'I Case B Match Code'
- Run the matching algorithm 'I Case B Matching Algorithm DT'



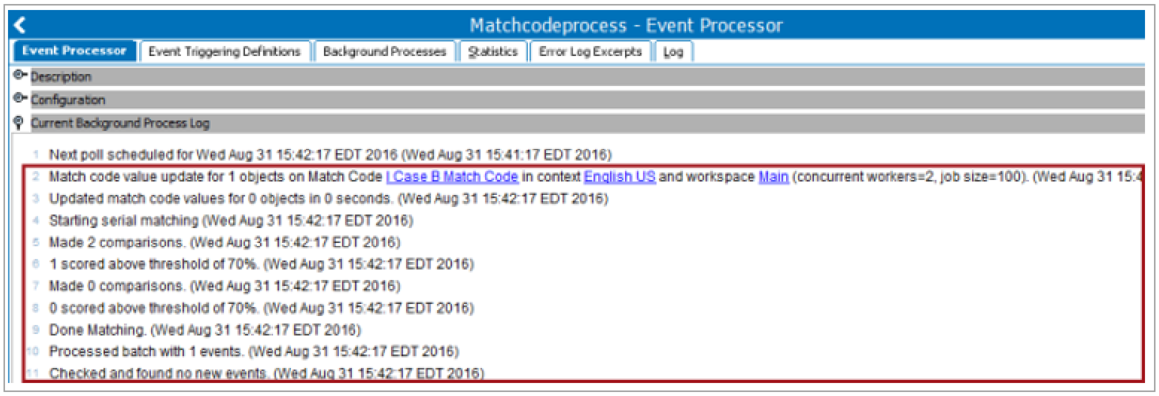
ID	Name
User running event processor plugin	User
Number of events to batch	1000
Days to retain events	0
Queue for event processor	EVPROC
Maximum number of old processes	100
Maximum age of old processes in hours	168
Limit of lines in execution report	1000
Processor	Matching
Schedule	Start every minute
Queue Status	Read Events
Unread events (approximated)	Click to estimate ...

When an entity valid for the match code is updated, the event processor detects the event. On the Event Processor tab, clicking the 'Click to estimate' button reports that an event exists but has not yet been read.



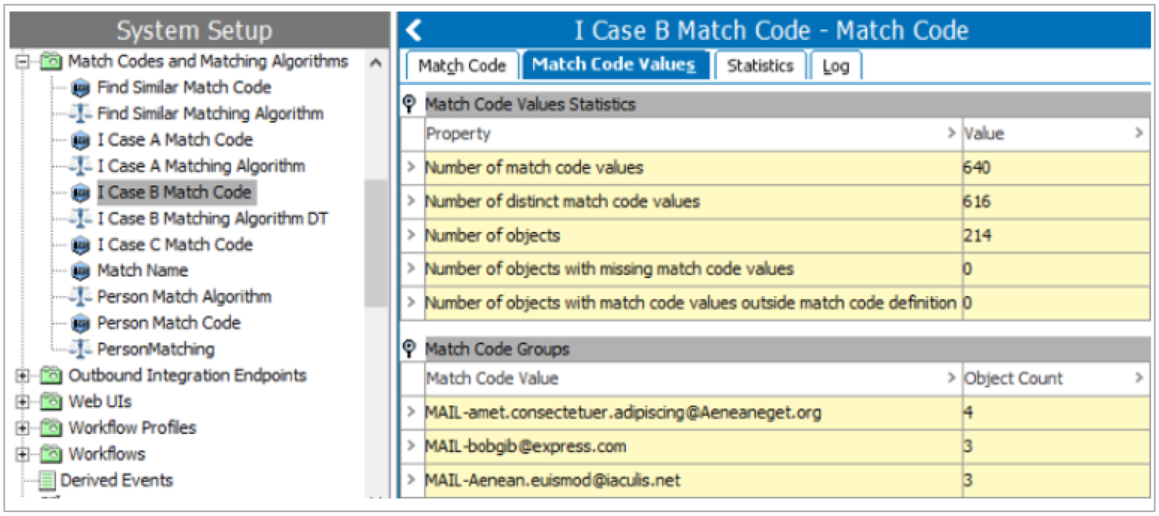
Unread events (approximated)	1 (2016-08-31 15:41:49)

After one minute, the event processor responds to the event, updates the match code values, and then runs the algorithm to determine possible duplicates among the affected object types, based on the defined algorithm threshold.


Matchcodeprocess - Event Processor
Event Processor
Event Triggering Definitions
Background Processes
Statistics
Error Log Excerpts
Log
Description
Configuration
Current Background Process Log

```
Next poll scheduled for Wed Aug 31 15:42:17 EDT 2016 (Wed Aug 31 15:41:17 EDT 2016)
Match code value update for 1 objects on Match Code I Case B Match Code in context English US and workspace Main (concurrent workers=2, job size=100). (Wed Aug 31 15:4
Updated match code values for 0 objects in 0 seconds. (Wed Aug 31 15:42:17 EDT 2016)
Starting serial matching (Wed Aug 31 15:42:17 EDT 2016)
Made 2 comparisons. (Wed Aug 31 15:42:17 EDT 2016)
1 scored above threshold of 70%. (Wed Aug 31 15:42:17 EDT 2016)
Made 0 comparisons. (Wed Aug 31 15:42:17 EDT 2016)
0 scored above threshold of 70%. (Wed Aug 31 15:42:17 EDT 2016)
Done Matching. (Wed Aug 31 15:42:17 EDT 2016)
Processed batch with 1 events. (Wed Aug 31 15:42:17 EDT 2016)
Checked and found no new events. (Wed Aug 31 15:42:17 EDT 2016)
```

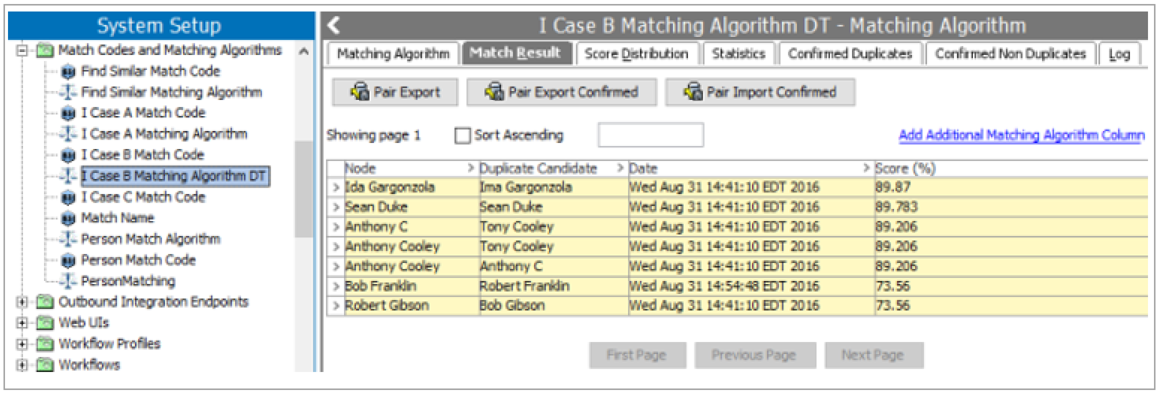
Once complete, you can view the 'I Case B Match Code' match code. On the Match Code Values tab, open the Match Code Values Statistics flipper, and click the **Yes** button to calculate the statistics and display information about the existing match codes values.



Property	Value
Number of match code values	640
Number of distinct match code values	616
Number of objects	214
Number of objects with missing match code values	0
Number of objects with match code values outside match code definition	0

Match Code Value	Object Count
MAIL-amet.consectetuer.adipiscing@Aeneaneget.org	4
MAIL-bobgib@express.com	3
MAIL-Aenean.euismod@iaculis.net	3

You can also check which objects scored above the match action threshold. These objects are now defined as potential duplicates and are displayed on the Match Result tab of the 'I Case B Matching Algorithm DT'.

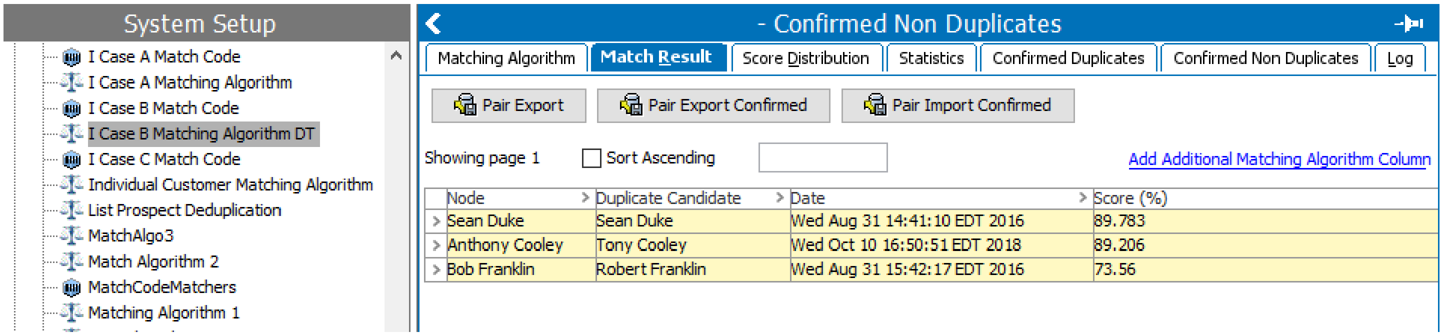


Node	Duplicate Candidate	Date	Score (%)
Ida Gargonzola	Ima Gargonzola	Wed Aug 31 14:41:10 EDT 2016	89.87
Sean Duke	Sean Duke	Wed Aug 31 14:41:10 EDT 2016	89.783
Anthony C	Tony Cooley	Wed Aug 31 14:41:10 EDT 2016	89.206
Anthony Cooley	Tony Cooley	Wed Aug 31 14:41:10 EDT 2016	89.206
Anthony Cooley	Anthony C	Wed Aug 31 14:41:10 EDT 2016	89.206
Bob Franklin	Robert Franklin	Wed Aug 31 14:54:48 EDT 2016	73.56
Robert Gibson	Bob Gibson	Wed Aug 31 14:41:10 EDT 2016	73.56

For more information, see the **Maintaining an Event Processor** topic of the **System Setup / Super User Guide** documentation.

Match and Link in Workbench

When the matching algorithm runs, the possible matches can be viewed on the 'Match Result' tab of the matching algorithm. Workbench supports the matching user actions defined below.



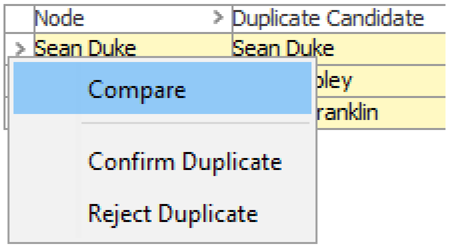
Node	Duplicate Candidate	Date	Score (%)
Sean Duke	Sean Duke	Wed Aug 31 14:41:10 EDT 2016	89.783
Anthony Cooley	Tony Cooley	Wed Oct 10 16:50:51 EDT 2018	89.206
Bob Franklin	Robert Franklin	Wed Aug 31 15:42:17 EDT 2016	73.56

You can merge identified duplicate source records using the Web UI. For more information, see **Match and Link, Merging Confirmed Matches** topic in the **Web UI Setup and User Guide / Web User Interfaces** documentation.

Important: It is recommended that potential duplicates identified through a Match and Merge solution are handled in Web UI. For more information, see the **Match and Merge** topic in this documentation.

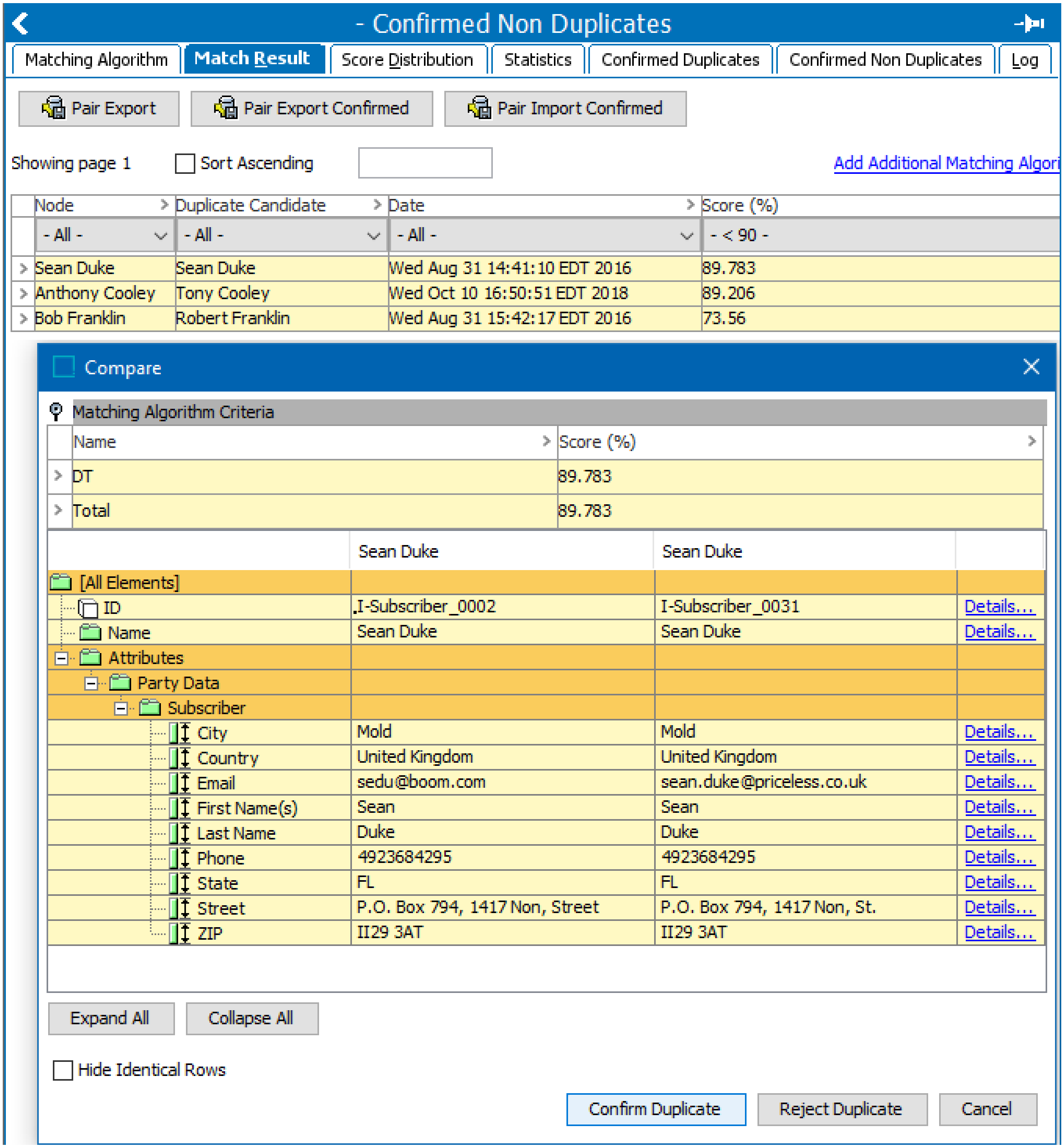
Compare Match Result

To compare an object with its duplicate or non duplicate candidate, on the 'Match Result' or 'Confirmed Non Duplicates' tab, right-click the first column of a row and select the 'Compare' option.



The 'Compare' screen shows the similarities and differences between the paired objects. When accessed via the 'Match Result,' you can confirm or reject duplicates via the 'Confirm Duplicate' and 'Reject Duplicate' buttons.

Right-click a column heading and select 'Filtering enabled' to allow easy navigation and filtering of desired data. Filtering in the following image has been set to include only rows that have a score of less than 90.



Showing page 1 □ Sort Ascending

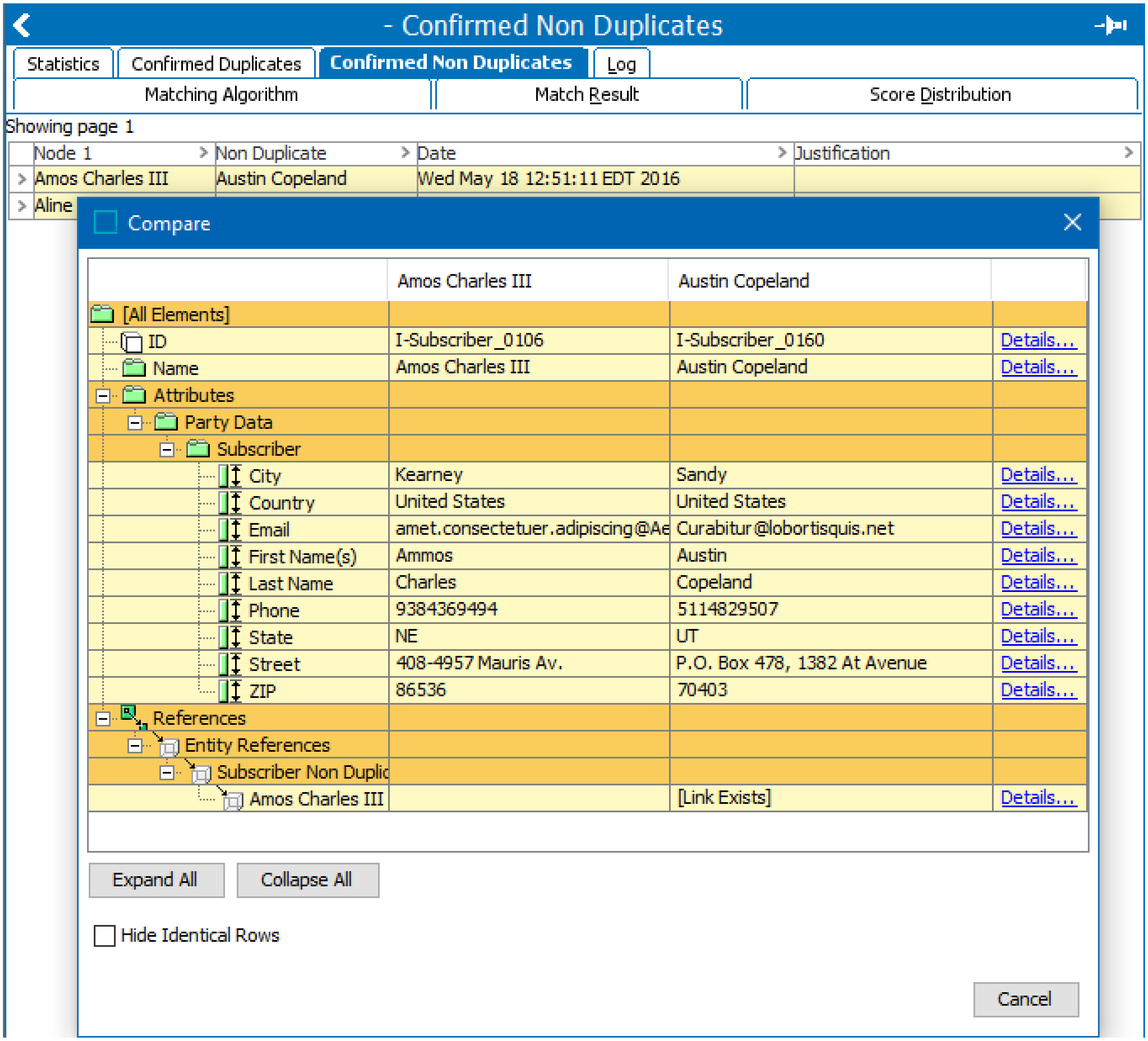
Node	Duplicate Candidate	Date	Score (%)
- All -	- All -	- All -	- < 90 -
Sean Duke	Sean Duke	Wed Aug 31 14:41:10 EDT 2016	89.783
Anthony Cooley	Tony Cooley	Wed Oct 10 16:50:51 EDT 2018	89.206
Bob Franklin	Robert Franklin	Wed Aug 31 15:42:17 EDT 2016	73.56

Name	Score (%)
DT	89.783
Total	89.783

	Sean Duke	Sean Duke	
[All Elements]			
ID	I-Subscriber_0002	I-Subscriber_0031	Details...
Name	Sean Duke	Sean Duke	Details...
Attributes			
Party Data			
Subscriber			
City	Mold	Mold	Details...
Country	United Kingdom	United Kingdom	Details...
Email	sedu@boom.com	sean.duke@priceless.co.uk	Details...
First Name(s)	Sean	Sean	Details...
Last Name	Duke	Duke	Details...
Phone	4923684295	4923684295	Details...
State	FL	FL	Details...
Street	P.O. Box 794, 1417 Non, Street	P.O. Box 794, 1417 Non, St.	Details...
ZIP	II29 3AT	II29 3AT	Details...

□ Hide Identical Rows

When accessed from the 'Confirmed Non Duplicates' tab, you can only view the data, no further actions are available.



Showing page 1

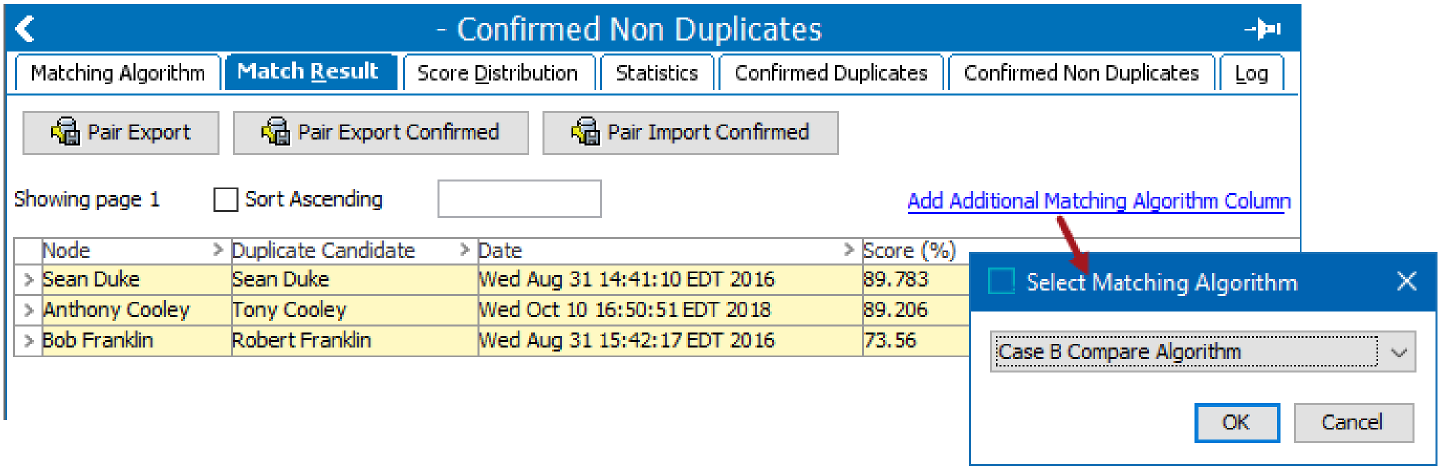
Node 1	Non Duplicate	Date	Justification
Amos Charles III	Austin Copeland	Wed May 18 12:51:11 EDT 2016	
Aline			

	Amos Charles III	Austin Copeland	
[All Elements]			
ID	I-Subscriber_0106	I-Subscriber_0160	Details...
Name	Amos Charles III	Austin Copeland	Details...
Attributes			
Party Data			
Subscriber			
City	Kearney	Sandy	Details...
Country	United States	United States	Details...
Email	amet.consectetuer.adipiscing@Ae	Curabitur@lobortisquis.net	Details...
First Name(s)	Ammos	Austin	Details...
Last Name	Charles	Copeland	Details...
Phone	9384369494	5114829507	Details...
State	NE	UT	Details...
Street	408-4957 Mauris Av.	P.O. Box 478, 1382 At Avenue	Details...
ZIP	86536	70403	Details...
References			
Entity References			
Subscriber Non Duplic			
Amos Charles III		[Link Exists]	Details...

☐ Hide Identical Rows

Adding Additional Matching Algorithm

On the 'Match Result tab, click the **Add Additional Matching Algorithm Column** link to add another matching algorithm to compare the objects. This allows you to review more information about the objects before deciding if they are duplicates or not.



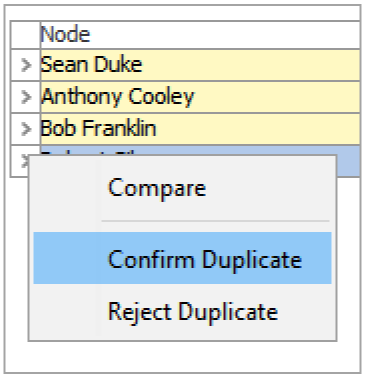
Node	Duplicate Candidate	Date	Score (%)
Sean Duke	Sean Duke	Wed Aug 31 14:41:10 EDT 2016	89.783
Anthony Cooley	Tony Cooley	Wed Oct 10 16:50:51 EDT 2018	89.206
Bob Franklin	Robert Franklin	Wed Aug 31 15:42:17 EDT 2016	73.56

Case B Compare Algorithm

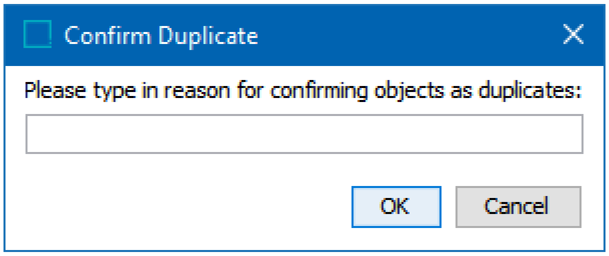
Confirm or Reject a Duplicate

From the 'Match Result' tab, you can compare pairs and mark them as either confirmed duplicates or confirmed non-duplicates.

1. In System Setup, select the relevant matching algorithm, and then click the 'Match Result' tab.
2. Click the row that contains the record being worked, right-click the arrow in the first column and select **Confirm Duplicate** or **Reject Duplicate** from the menu.



3. Provide a reason for the confirmation / rejection and click **OK**. The reason is saved as an attribute value on the corresponding Confirm Duplicate / Confirm Non Duplicate reference.



Please type in reason for confirming objects as duplicates:

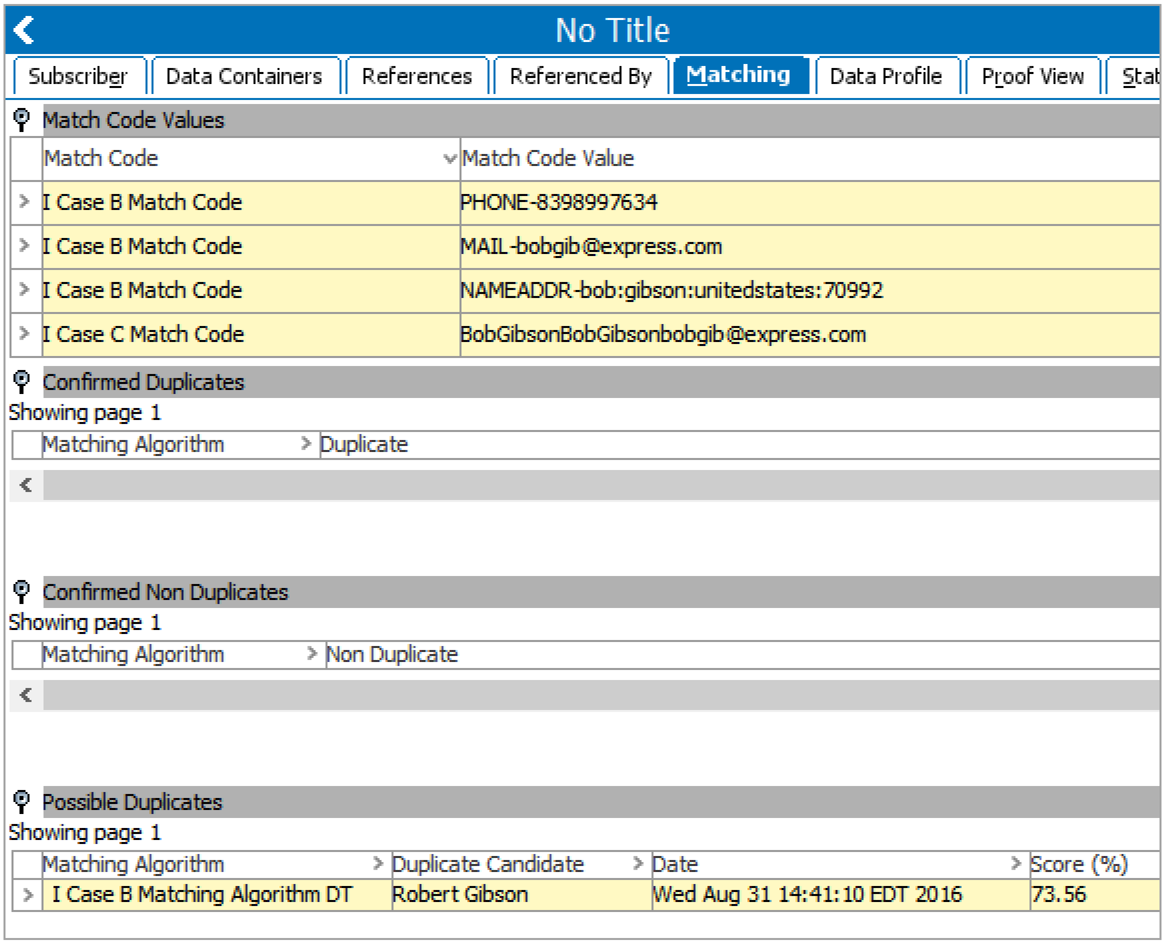
- The **Duplicate** reference type is created between two objects that are manually confirmed as duplicates. This reference means that regardless of how the objects are modified, the matching algorithm always sees them as duplicates.
- The **Non Duplicate** reference type is created between two objects when a duplicate candidate is rejected. This reference means the two objects will never be identified as duplicates by the matching algorithm regardless of how they are modified.
- These references can be manually removed via the 'References' tab of the object in question.

View Matched Objects in Tree

Duplicate information can also be viewed directly on each link golden record source record in the Tree.

Choose a method to view the object:

- In the Tree, select the relevant source record and click the 'Matching' tab.



Match Code Values

Match Code	Match Code Value
I Case B Match Code	PHONE-8398997634
I Case B Match Code	MAIL-bobgib@express.com
I Case B Match Code	NAMEADDR-bob:gibson:unitedstates:70992
I Case C Match Code	BobGibsonBobGibsonbobgib@express.com

Confirmed Duplicates

Showing page 1

Matching Algorithm	Duplicate

Confirmed Non Duplicates

Showing page 1

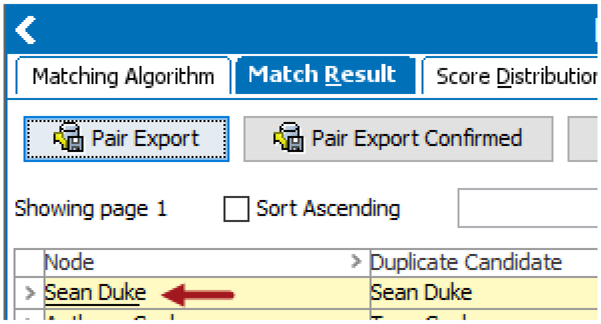
Matching Algorithm	Non Duplicate

Possible Duplicates

Showing page 1

Matching Algorithm	Duplicate Candidate	Date	Score (%)
I Case B Matching Algorithm DT	Robert Gibson	Wed Aug 31 14:41:10 EDT 2016	73.56

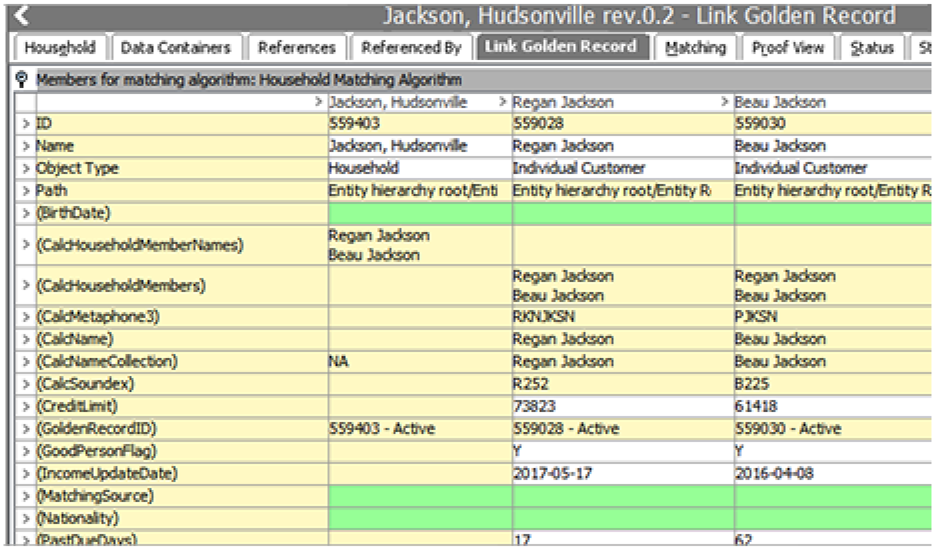
- On the 'Match Result' tab, click the link of the object to open the object editor in Tree.



Showing page 1 ☐ Sort Ascending

Node	Duplicate Candidate
Sean Duke	Sean Duke

- For the link golden record, the 'Link Golden Record' tab display the source records that are linked to it.



Members for matching algorithm: Household Matching Algorithm

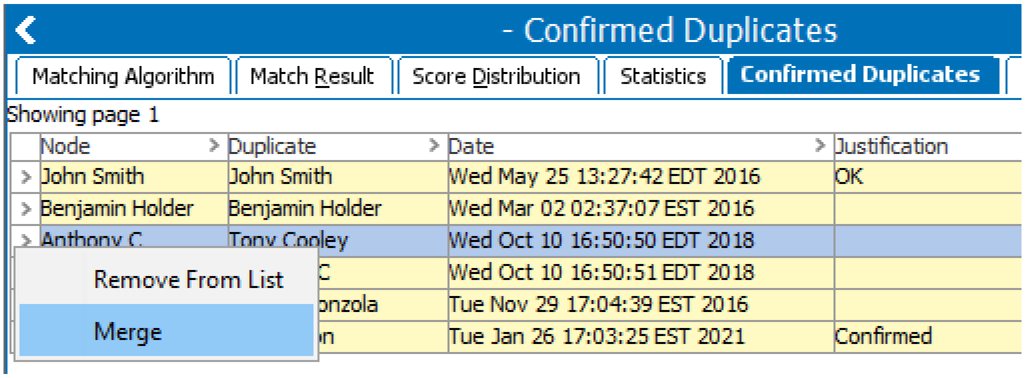
	Jackson, Hudsonville	Regan Jackson	Beau Jackson
ID	559403	559028	559030
Name	Jackson, Hudsonville	Regan Jackson	Beau Jackson
Object Type	Household	Individual Customer	Individual Customer
Path	Entity hierarchy root/Enti	Entity hierarchy root/Entity R	Entity hierarchy root/Entity R
(BirthDate)			
(CalcHouseholdMemberNames)	Regan Jackson Beau Jackson		
(CalcHouseholdMembers)		Regan Jackson Beau Jackson	Regan Jackson Beau Jackson
(CalcMetaphone3)		RKNJKSN	PJKSN
(CalcName)		Regan Jackson	Beau Jackson
(CalcNameCollection)	NA	Regan Jackson	Beau Jackson
(CalcSoundex)		R252	B225
(CreditLimit)		73823	61418
(GoldenRecordID)	559403 - Active	559028 - Active	559030 - Active
(GoodPersonFlag)		Y	Y
(IncomeUpdateDate)		2017-05-17	2016-04-08
(MatchingSource)			
(Nationality)			
(PastDueDays)		17	62

Merge Confirmed Duplicates

The 'Identify Duplicates' or 'Link Golden Record' actions can create two objects that are confirmed duplicates and it is possible to manually merge them into a single object.

> **Important:** Because duplicate source records are deleted during a merge, this should not be used as part of a golden record solution.

1. From the 'Confirmed Duplicates' tab, right-click the first column and choose the **Merge** option.

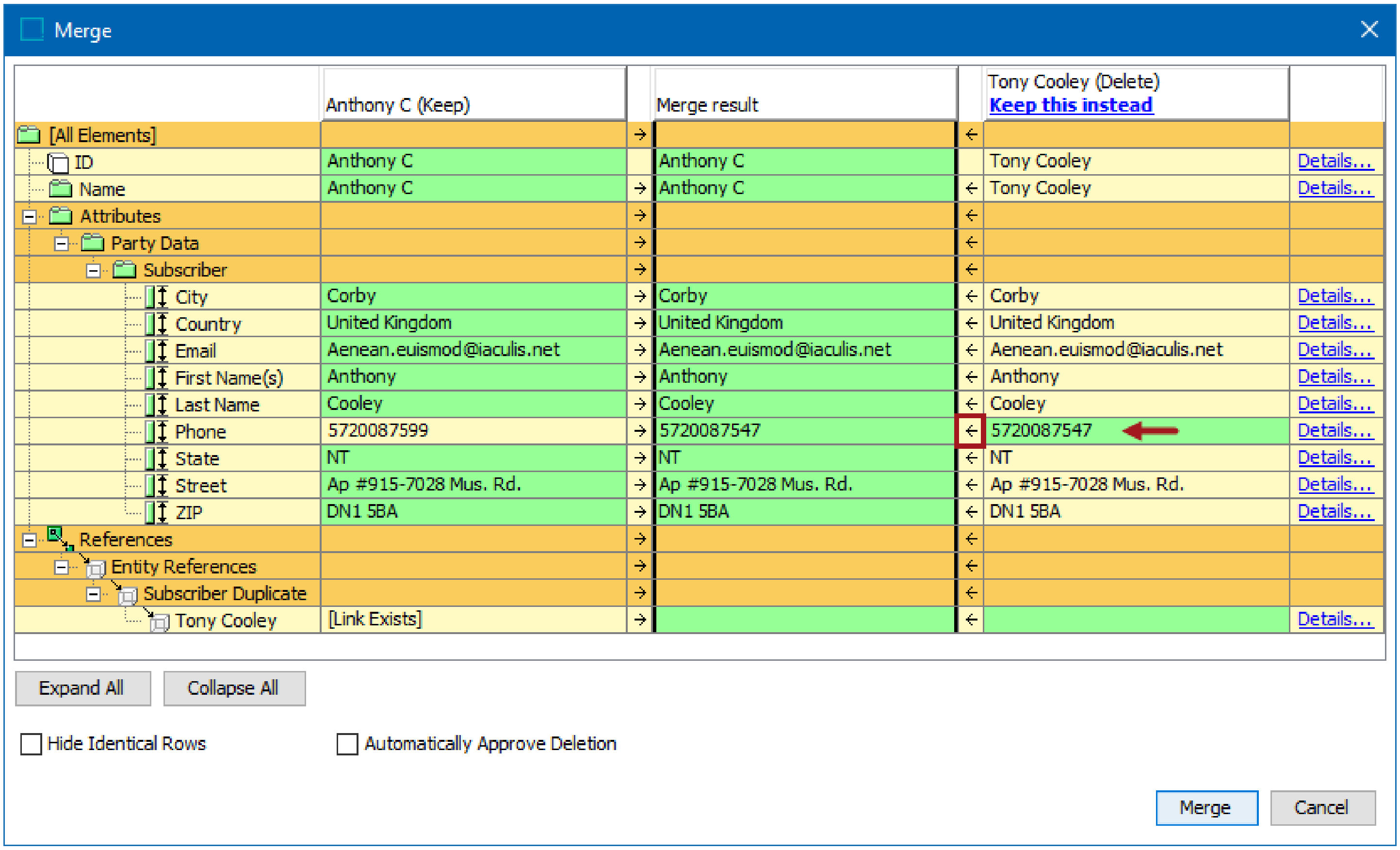


Showing page 1

Node	Duplicate	Date	Justification
John Smith	John Smith	Wed May 25 13:27:42 EDT 2016	OK
Benjamin Holder	Benjamin Holder	Wed Mar 02 02:37:07 EST 2016	
Anthony C	Tony Cooley	Wed Oct 10 16:50:50 EDT 2018	
	C	Wed Oct 10 16:50:51 EDT 2018	
	onzola	Tue Nov 29 17:04:39 EST 2016	
	n	Tue Jan 26 17:03:25 EST 2021	Confirmed

2. On the Merge dialog, review the data and decide which object to keep.

The first column is the data type. The three data columns are: the '(Keep)' data, the data that will remain after the merge (Merge result), and the '(Delete)' data. The green cell background color indicates where data is taken from.



	Anthony C (Keep)		Merge result		Tony Cooley (Delete) Keep this instead	
[All Elements]		→		←		
ID	Anthony C		Anthony C		Tony Cooley	Details...
Name	Anthony C	→	Anthony C	←	Tony Cooley	Details...
Attributes		→		←		
Party Data		→		←		
Subscriber		→		←		
City	Corby	→	Corby	←	Corby	Details...
Country	United Kingdom	→	United Kingdom	←	United Kingdom	Details...
Email	Aenean.euismod@iaculis.net	→	Aenean.euismod@iaculis.net	←	Aenean.euismod@iaculis.net	Details...
First Name(s)	Anthony	→	Anthony	←	Anthony	Details...
Last Name	Cooley	→	Cooley	←	Cooley	Details...
Phone	5720087599	→	5720087547	←	5720087547	Details...
State	NT	→	NT	←	NT	Details...
Street	Ap #915-7028 Mus. Rd.	→	Ap #915-7028 Mus. Rd.	←	Ap #915-7028 Mus. Rd.	Details...
ZIP	DN1 5BA	→	DN1 5BA	←	DN1 5BA	Details...
References		→		←		
Entity References		→		←		
Subscriber Duplicate		→		←		
Tony Cooley	[Link Exists]	→		←		Details...

☐ Hide Identical Rows ☐ Automatically Approve Deletion

- Click the **Details...** link to open a large display of the data on the selected row.
- Click the **Expand All** or **Collapse All** buttons to show or hide the detailed data.
- Check the **Hide Identical Rows** checkbox to show only the rows with different data.
- Check the **Automatically Approve Deletion** checkbox to approve deletion of objects in the 'Delete' column during the merge process and avoid having to manually delete the duplicate record.
- Click the **Keep this instead** link to move all data from the (Delete) column into the Merge result column.
- Click the arrow on an individual row to move only the data from that cell to the Merge result column, as shown for the Phone row.
- When the data in the Merge result column is the record you want to keep, click the **Merge** button to perform the merge and keep a single record.

Merge Considerations

If the object that remains contains no data in any context, the data is taken from the deleted object and merged into the remaining object. Data is defined as:

- Attributes
- Object name
- Reference types
- Object to classification link types
- Table types
- Object to attribute links

Reference and link types do not accumulate. If the reference or link type is already populated in any context nothing is merged from the object that is deleted.

During the merge process, all references to the deleted object are modified to point to the object that remains in the database. This means that the source objects of these references will be modified. 'Automatically Approve Deletion' only approves the deletion of objects and changes to objects due to references that are pointed to another target are not approved.

Match and Link in Web UI

The Web UI supports the user actions described under the **User Actions** section of the **Match and Link** topic in this documentation.

Users must add a clerical review widget to the homepage. For more information, see the **Adding Widgets to a Homepage** topic in the **Web User Interfaces / Web UI Getting Started** documentation.

The following Web UI topics are relevant to configuring a Link Golden Record solution and can be found in the **Web User Interfaces / Web UI Getting Started** documentation:

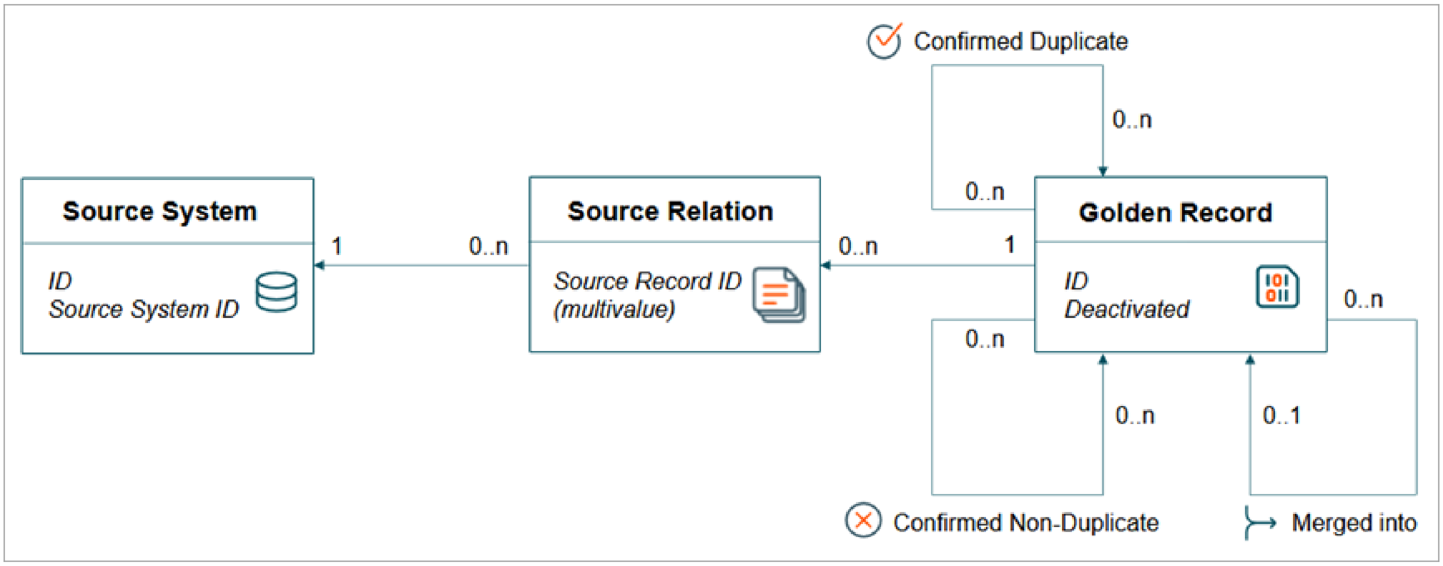
- **Match and Link, Potential Duplicates List**
- **Match and Link, Merging Confirmed Matches**
- **Match and Link, Configuring a Deduplication Clerical Review**

Match and Merge

A match and merge solution takes ownership over the data and is well suited to data hub implementations with any degree of centralized or decentralized management of data.

In the following, we will use an example of maintaining customer records in a match and merge solution to explain the match and merge data flow.

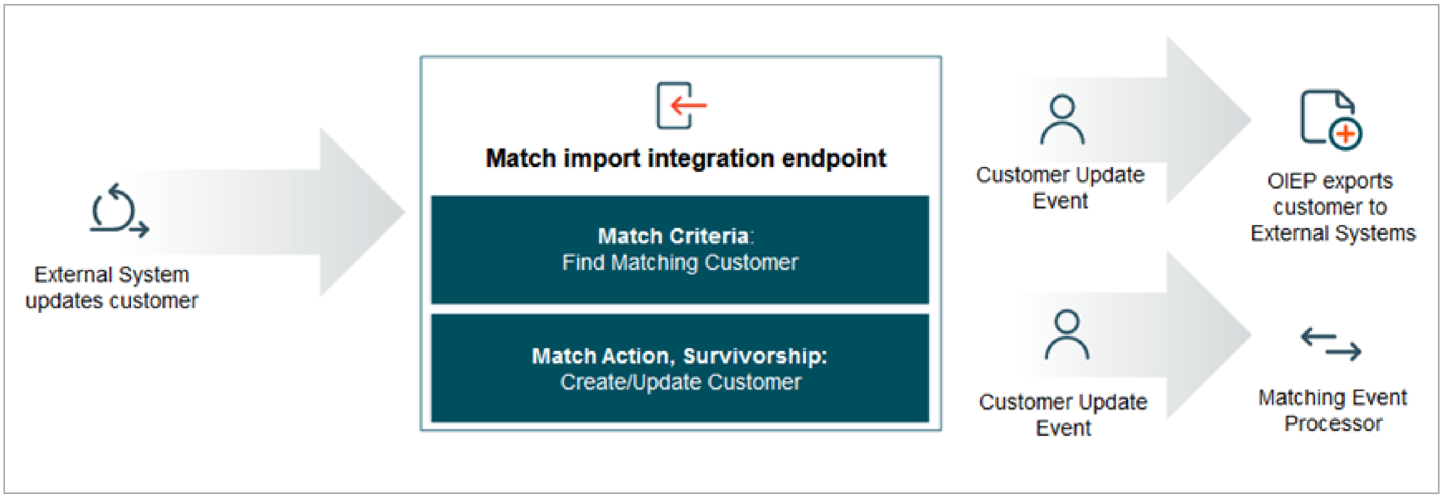
Data Model



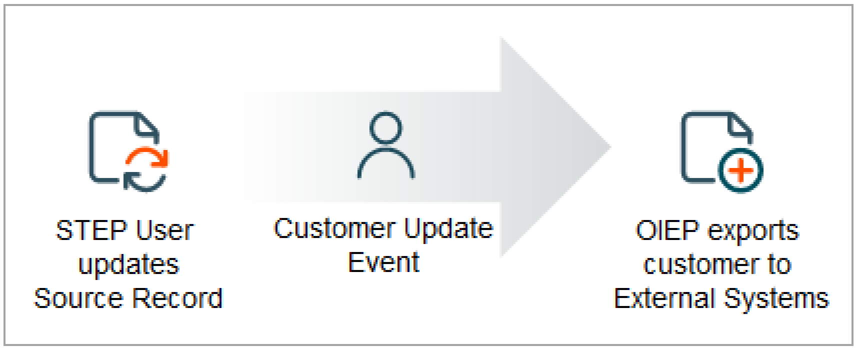
The first thing to note in comparison to a Match and Link Match Action is that the source record and golden record does not use separate object types.

Information Flow

When a customer is created or updated in an external system, the update is delivered to STEP via either Match and Merge Web Service Endpoint or via an Inbound Integration Endpoint configured with the STEP Match and Merge Importer Processing Engine. In both cases, the incoming source record is matched against the existing golden records, and if a match is found, the information from the source record is merged into the relevant golden record using survivorship rules. If this results in updated information, the customer record can be exported back to all external systems. In this way, an update to the customer in any system can be automatically managed for trust and timeliness, and we can ensure that the best possible view of the customer is reflected across the entire ecosystem.


External System updates customer
Match import integration endpoint
Match Criteria: Find Matching Customer
Match Action, Survivorship: Create/Update Customer
Customer Update Event
OIEP exports customer to External Systems
Customer Update Event
Matching Event Processor

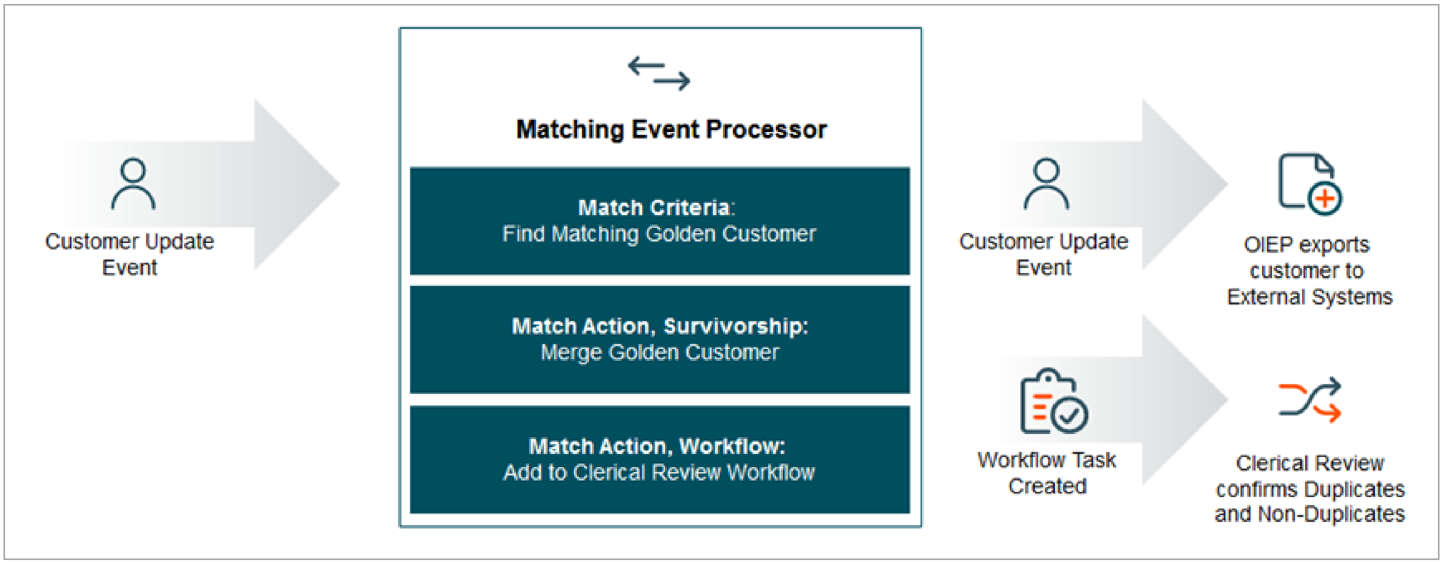
When a user updates the customer in STEP, the update takes place on the golden record itself, and the new trusted record can be exported in the same way as before.


STEP User updates Source Record
Customer Update Event
OIEP exports customer to External Systems

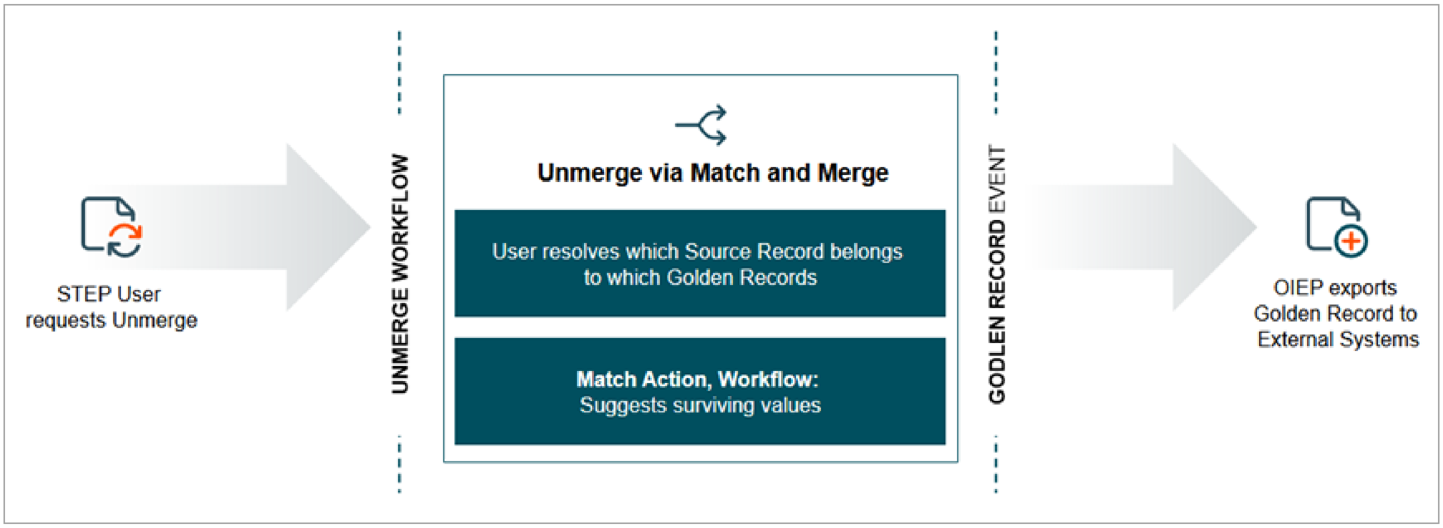
A matching result in a score, which can fall within three groups separated by thresholds. The lowest threshold is the clerical review threshold. A match score below this is considered a non-match. The next threshold is the auto threshold. Anything between these thresholds are possible matches and will be sent to a clerical review to determine if this is indeed a match or not. Anything above the auto threshold is considered a match and will result in the system automatically merging the information.

As golden records are created or updated, a matching event processor compares the golden record to other golden records in the system.

If the match score is higher than the threshold for auto-merge, the matching algorithm will declare one of the records the survivor, and deactivate the other record. The information from the deactivated record will be merged into the surviving record, using the survivorship rules set on the matching algorithm. If the match score is between the auto threshold and the clerical review threshold, the two records will be sent to the clerical review workflow to let a data steward decide if the two customer records should be confirmed as duplicates and merged or confirmed as non-duplicates and be kept separate going forward.


Customer Update Event
Matching Event Processor
Match Criteria: Find Matching Golden Customer
Match Action, Survivorship: Merge Golden Customer
Match Action, Workflow: Add to Clerical Review Workflow
Customer Update Event
OIEP exports customer to External Systems
Workflow Task Created
Clerical Review confirms Duplicates and Non-Duplicates

Even in the best organizations, accidents happen. Should two records be merged by accident, STEP has tools to help resolve the issue. In a data hub that is closely integrated with a multitude of source systems, the process of unmerge may require a range of activities in the workflow, aside from the actual unmerge Web UI. The unmerge Web UI uses both original source records from source systems, revision history, and the match algorithm survivorship rules to help the user more easily determine which values belong to what records during an unmerge.


STEP User requests Unmerge
UNMERGE WORKFLOW
Unmerge via Match and Merge
User resolves which Source Record belongs to which Golden Records
Match Action, Workflow: Suggests surviving values
GODLEN RECORD EVENT
OIEP exports Golden Record to External Systems

Configuring Match and Merge

This configuration section assumes you already have set up the matching component model and match algorithm with match criteria.

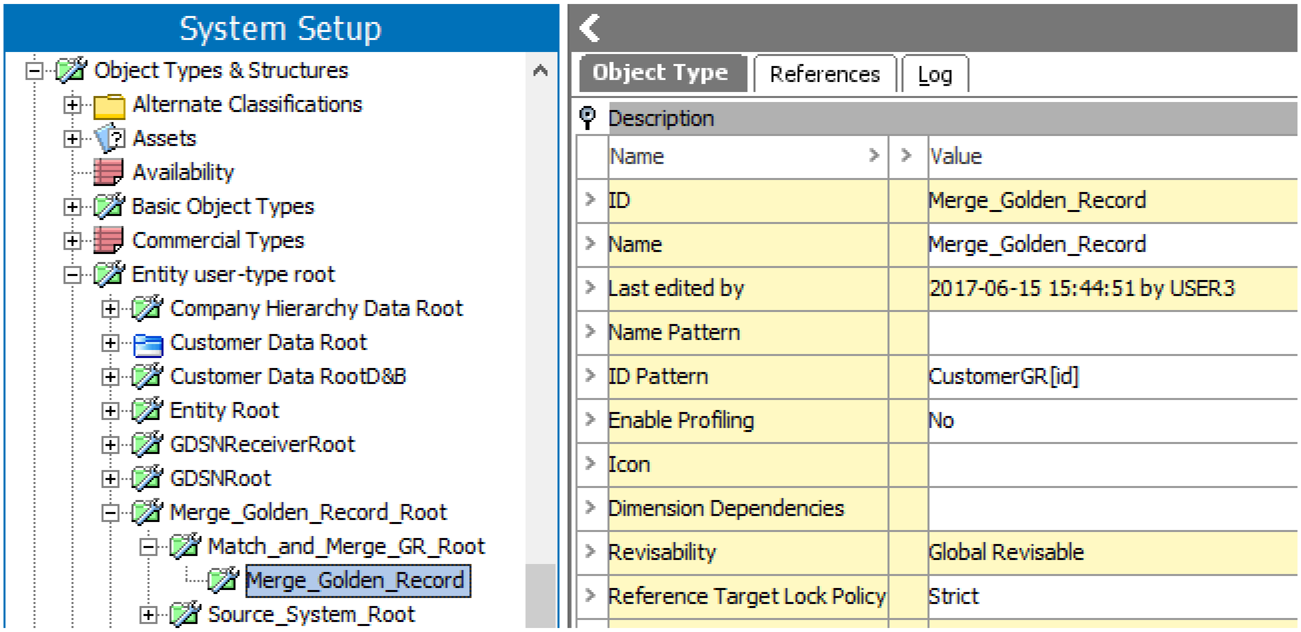
Several elements are required to configure a match algorithm to use the Match and Merge match action:

- Requirements to the golden record object type
- Setup a match and merge component model
- Configure the merge golden record match action for the matching algorithm
- Setup source record input endpoints: match and merge web service endpoint and/or match and merge inbound integration endpoint
- Setup an event processor
- Setup survivorship rules
- Consider setting up source traceability
- Consider if you need an unmerge workflow
- Setup Web UI for traceability, clerical review merge, and clerical review unmerge as needed

The Corner Bar Search and Homepage Search widgets can be used to search for golden records via golden record or source record ID. For more information on these widgets, see the **Corner Bar Search Component** topic or **Homepage Widgets** topic of the **Web User Interfaces / Web UI Getting Started** documentation. Details on how to add a widget to a homepage can be found in the **Adding Widgets to a Homepage** topic in the **Web User Interfaces / Web UI Getting Started** documentation.

Golden Record Object Type

Golden records must be configured before being mapped to the component model and cited in a match action configuration. Considering how golden records are generated, this object type must have an auto ID pattern configuration.



System Setup

- Object Types & Structures
  - Alternate Classifications
  - Assets
  - Availability
  - Basic Object Types
  - Commercial Types
  - Entity user-type root
    - Company Hierarchy Data Root
    - Customer Data Root
    - Customer Data RootD&B
    - Entity Root
    - GDSNReceiverRoot
    - GDSNRoot
    - Merge_Golden_Record_Root
      - Match_and_Merge_GR_Root
        - Merge_Golden_Record
      - Source_System_Root

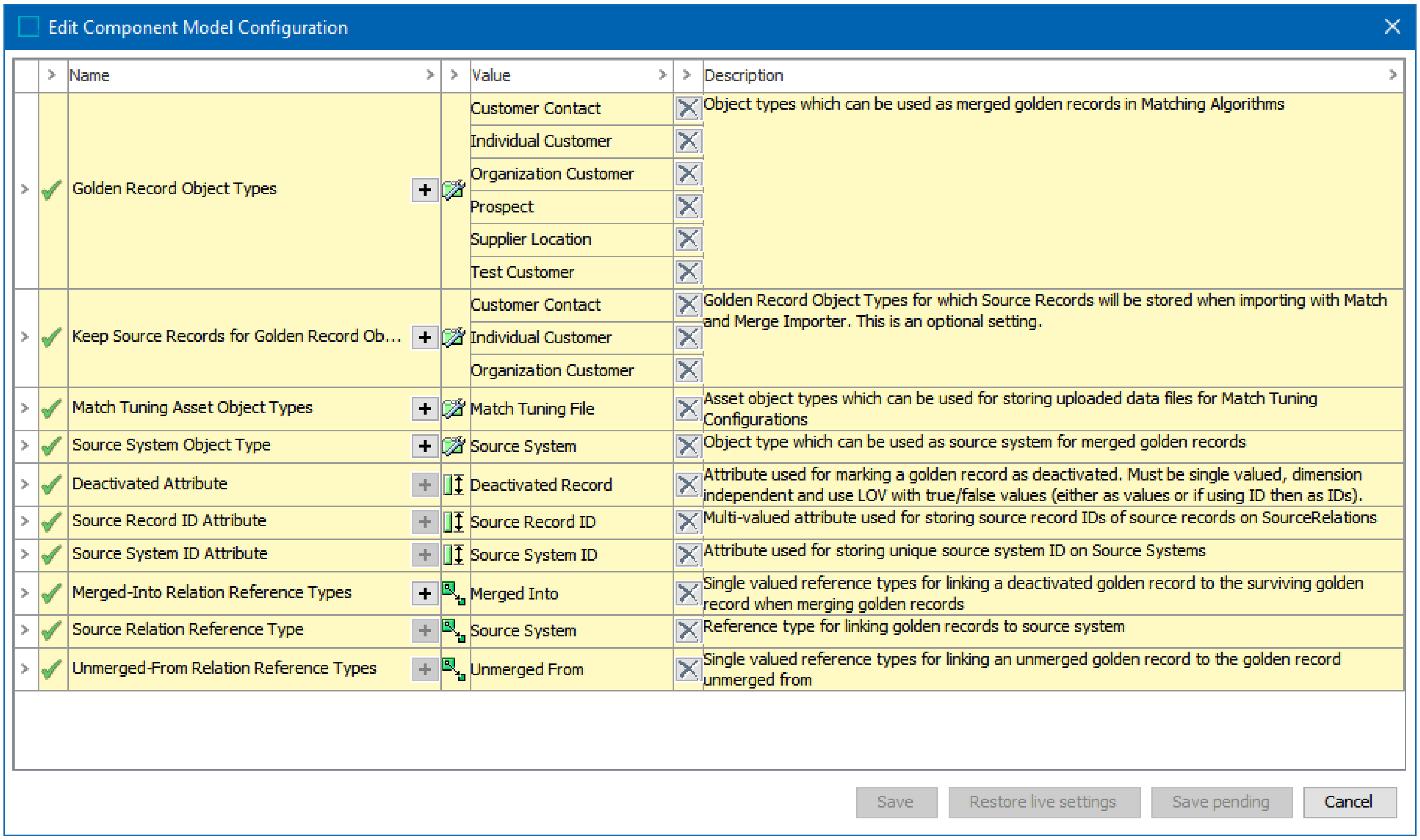
Name	Value
ID	Merge_Golden_Record
Name	Merge_Golden_Record
Last edited by	2017-06-15 15:44:51 by USER3
Name Pattern	
ID Pattern	CustomerGR[id]
Enable Profiling	No
Icon	
Dimension Dependencies	
Revisability	Global Revisable
Reference Target Lock Policy	Strict

Once the object type has been created, the golden record object type must be selected in the 'Matching - Merge Golden Record' component model. For more information on creating object types, see the **Object Types and Structures** topic of the **System Setup / Super User Guide** documentation. For more information on the component model configuration, see the **Component Model** topic in the **System Setup / Super User Guide** documentation.

Match and Merge Component Model

The 'Matching - Merge Golden Record' component model lists all the golden record object types applicable to the merge golden record solution, along with several vital components. Golden record object types applicable to the link golden record solution should not be listed.

1. In System Setup, expand 'Component Models' and click on the 'Matching - Merge Golden Records' node.
2. On the 'Component Model Configuration' tab, click the Edit link.



	Name	Value	Description
✓	Golden Record Object Types	Customer Contact Individual Customer Organization Customer Prospect Supplier Location Test Customer	Object types which can be used as merged golden records in Matching Algorithms
✓	Keep Source Records for Golden Record Ob…	Customer Contact Individual Customer Organization Customer	Golden Record Object Types for which Source Records will be stored when importing with Match and Merge Importer. This is an optional setting.
✓	Match Tuning Asset Object Types	Match Tuning File	Asset object types which can be used for storing uploaded data files for Match Tuning Configurations
✓	Source System Object Type	Source System	Object type which can be used as source system for merged golden records
✓	Deactivated Attribute	Deactivated Record	Attribute used for marking a golden record as deactivated. Must be single valued, dimension independent and use LOV with true/false values (either as values or if using ID then as IDs).
✓	Source Record ID Attribute	Source Record ID	Multi-valued attribute used for storing source record IDs of source records on SourceRelations
✓	Source System ID Attribute	Source System ID	Attribute used for storing unique source system ID on Source Systems
✓	Merged-Into Relation Reference Types	Merged Into	Single valued reference types for linking a deactivated golden record to the surviving golden record when merging golden records
✓	Source Relation Reference Type	Source System	Reference type for linking golden records to source system
✓	Unmerged-From Relation Reference Types	Unmerged From	Single valued reference types for linking an unmerged golden record to the golden record unmerged from

3. Click the 'plus' button for the relevant component aspect to display the selection dialog, and then choose to add an object, attribute, or reference:

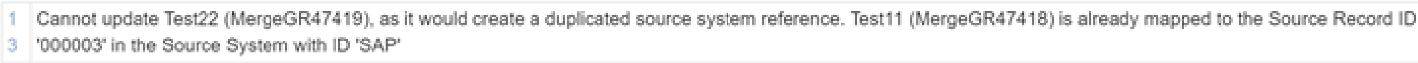
   - **Golden Record Object Type** – Select the object types that can be used as golden records in the system. Only the object types that you select can be used as golden records for Merge Golden Record configurations.
   - **Keep Source Records for Golden Record Object Types** – Select golden record object types for which the source record data should be stored.
   - **Match Tuning Asset Object Types** – Select the asset object types that can be used for storing uploaded data files for match tuning configurations.
   - **Source System Object Type** – Select the object type that can be used as a source system. This source system is referenced by golden records to signify where the record originated.
   - **Deactivated Attribute** – Select the attribute that is used for marking a golden record as deactivated.

     The 'Deactivated Attribute' values are maintained by features in the match and merge match action. It is not advisable to maintain these by other means.

   - **Source Record ID Attribute** – Select the attribute that is used for storing the IDs of source records on golden record objects. This attribute must be multi-valued.

     Source record ID attribute values are copied from source records by features in the match and merge match action. It is not advisable to edit source record id attribute values by other means.

Some level of protection exists against duplicating source record IDs, in which case you may experience:



```
Cannot update Test22 (MergeGR47419), as it would create a duplicated source system reference. Test11 (MergeGR47418) is already mapped to the Source Record ID
'000003' in the Source System with ID 'SAP'
```

- **Source System ID Attribute** – Select the attribute this is used for storing unique source system IDs on their respective source system objects.
- **Merged-Into Relation Reference Types** – Select the reference type(s) that link deactivated golden records to surviving golden records during a merge. This must be a single valued reference type.
- **Source Relationship Reference Type** – Select the reference type that links golden records to source system objects.
- **Unmerged-From Relation Reference Types** – Used for unmerge workflow. For more information, see the **Configuring Unmerge in Workbench** topic in this documentation.

Click the 'X' button to remove the relevant value from the component model. A green checkmark will appear if the component model has a valid configuration. Source relationship references are maintained by features in the match and merge match action. It is not advisable to maintain these by other means.

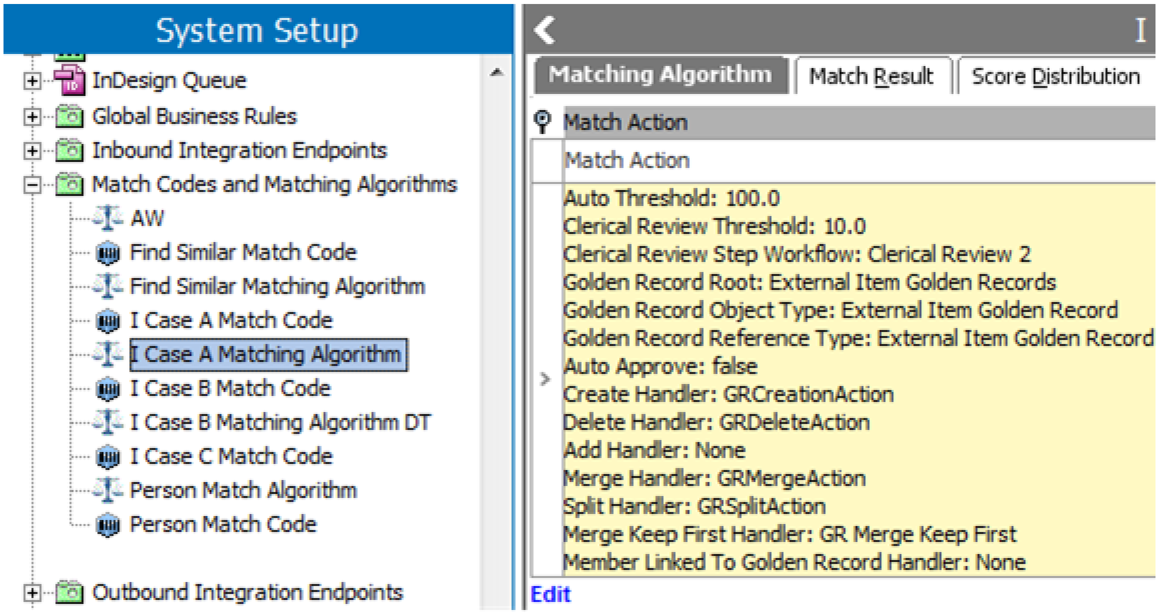
4. Click Save to save changes.

Note: If you need to navigate away from the configuration dialog and the component model is not yet valid (it has an 'X' instead of a checkmark), click Save pending to save your work.

Merge Golden Record Match Action

To set up the merge golden record match action, follow these steps:

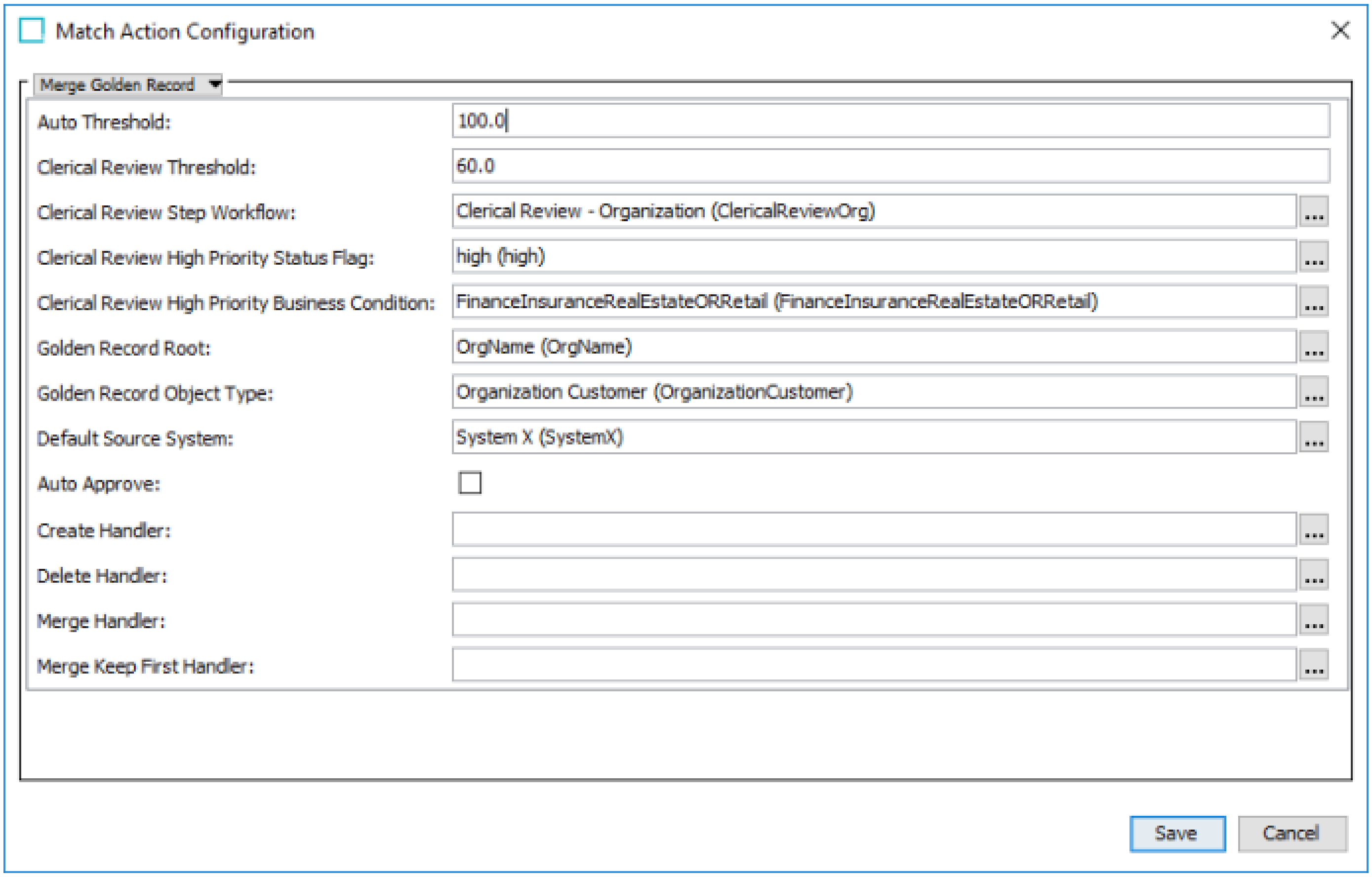
1. On the matching algorithm node, navigate to the 'Matching Algorithm' tab.
2. In the 'Match Action' area, click the Edit link.



Match Action

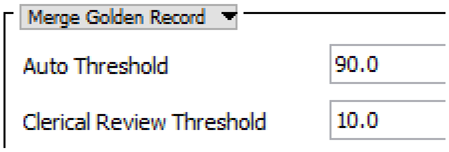
Auto Threshold: 100.0
Clerical Review Threshold: 10.0
Clerical Review Step Workflow: Clerical Review 2
Golden Record Root: External Item Golden Records
Golden Record Object Type: External Item Golden Record
Golden Record Reference Type: External Item Golden Record
Auto Approve: false
Create Handler: GRCreationAction
Delete Handler: GRDeleteAction
Add Handler: None
Merge Handler: GRMergeAction
Split Handler: GRSplitAction
Merge Keep First Handler: GR Merge Keep First
Member Linked To Golden Record Handler: None

Once on the 'Match Action Configuration' screen, several parameters must be configured.



Auto Threshold:	100.0
Clerical Review Threshold:	60.0
Clerical Review Step Workflow:	Clerical Review - Organization (ClericalReviewOrg)
Clerical Review High Priority Status Flag:	high (high)
Clerical Review High Priority Business Condition:	FinanceInsuranceRealEstateORRetail (FinanceInsuranceRealEstateORRetail)
Golden Record Root:	OrgName (OrgName)
Golden Record Object Type:	Organization Customer (OrganizationCustomer)
Default Source System:	System X (SystemX)
Auto Approve:	□
Create Handler:	
Delete Handler:	
Merge Handler:	
Merge Keep First Handler:	

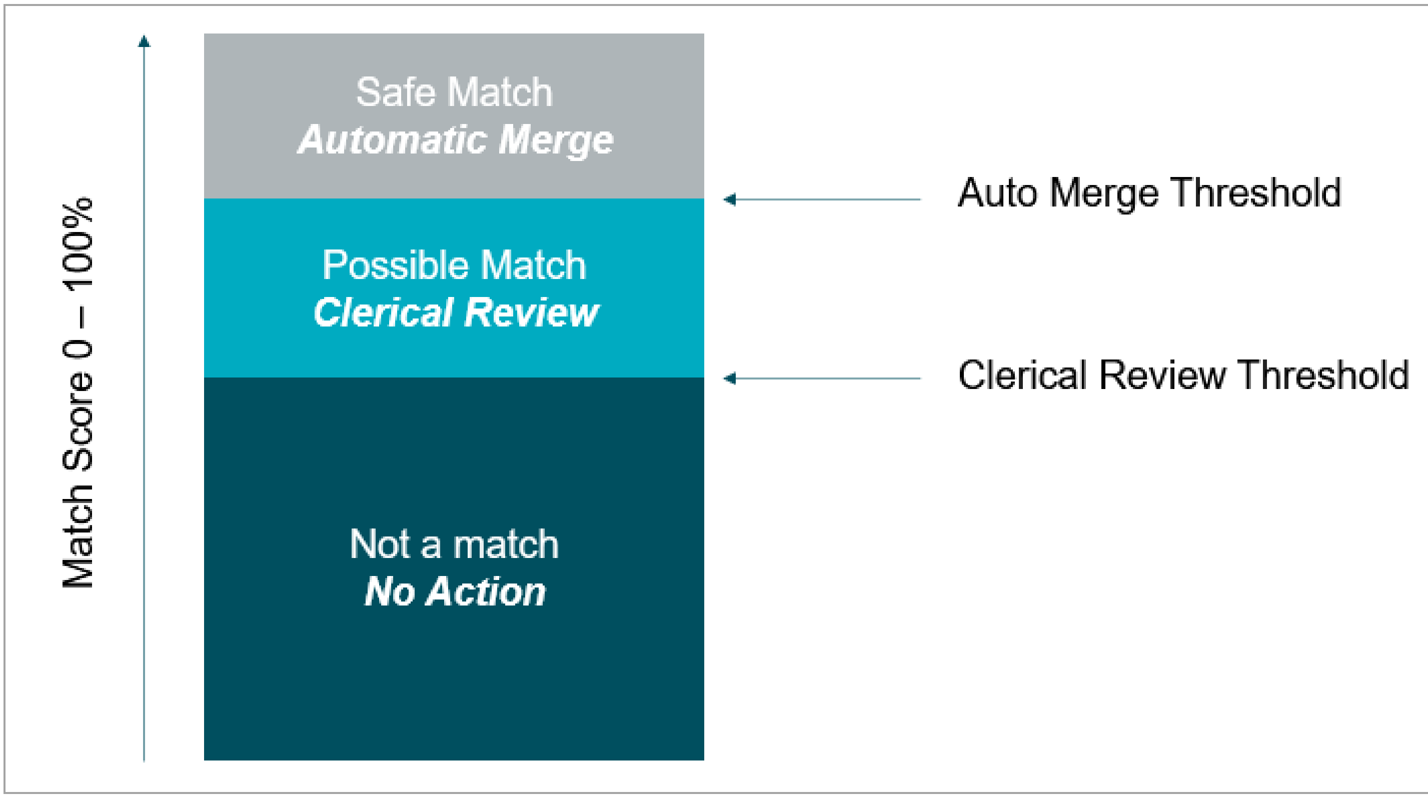
3. For the 'Auto Threshold' and 'Clerical Review Threshold' parameters, specify how matches will be evaluated via an equality measurement.



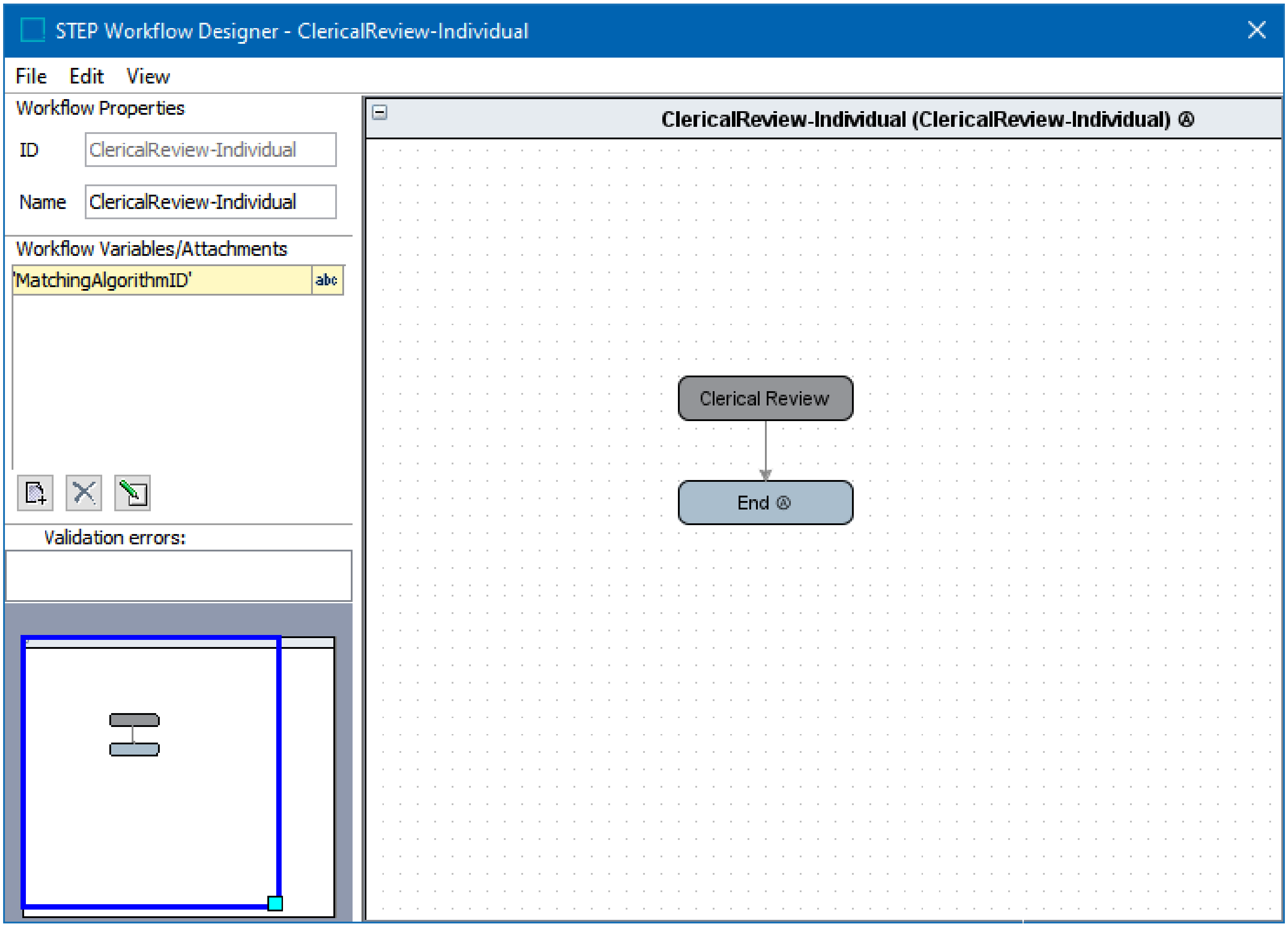
Merge Golden Record ▼

Auto Threshold	90.0
Clerical Review Threshold	10.0

- The 'Auto Threshold' specifies how equal two objects have to be in order to automatically have them merged together.
- The 'Clerical Review Threshold' should be equal to or less than the 'Auto Threshold.' It specifies how equal two objects must be to be considered potential duplicates.
- Objects with an equality metric between the 'Clerical Review Threshold' and the 'Auto Threshold' will be flagged as potential duplicates and enter a clerical review workflow together. If a user then confirms that the objects are in fact duplicates, the objects can be merged together. Records that contribute data but do not survive the merge are deactivated.



4. In the 'Clerical Review STEP Workflow' parameter, click the ellipsis button (...) and select the relevant clerical review workflow. A clerical review workflow can be as simple or elaborate as needed. For more information, see the **Creating a Workflow** topic in the **Workflows** documentation.


STEP Workflow Designer - ClericalReview-Individual
File
Edit
View
ClericalReview-Individual (ClericalReview-Individual)
Clerical Review
End

Workflow Properties

ID ClericalReview-Individual

Name ClericalReview-Individual

Workflow Variables/Attachments

'MatchingAlgorithmID' abc

Validation errors:

5. In the 'Clerical Review High Priority Status Flag' parameter, click the ellipsis button (...) and select the STEP workflow status flag that is used to designate high priority tasks in the clerical review workflow.
6. In the 'Clerical Review High Priority Business Condition' parameter, click the ellipsis button (...) and select the business condition that is used to verify if a task is of high priority.

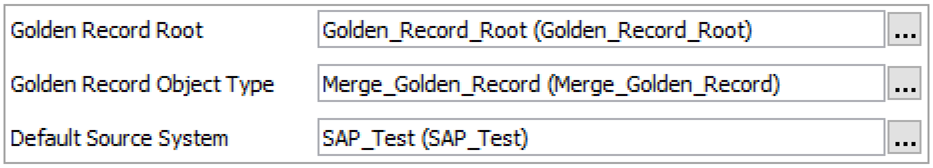
Note: If a status flag has been configured, but a business condition has not, then the status flags behave as if the theoretical business condition evaluated to true. If a business condition has been configured, and a status flag has not, the business condition is ignored.

The business condition is evaluated on each object in the clerical review task (each potential duplicate) in the context of the matcher, and will have access to the 'Current Object' binds.

Note: Though the business condition runs as a part of matching and it involves a clerical review, no matching or workflow binds are available.

Important: When status flags are used in this way, the matching algorithm determines which status flags are set (or not set). Due to this, no other status flags should be configured in the clerical review workflow.

7. For the 'Golden Record Root,' 'Golden Record Object Type,' and 'Default Source System' parameters, specify the applicable golden record root node, golden record root object, and default source system respectively.



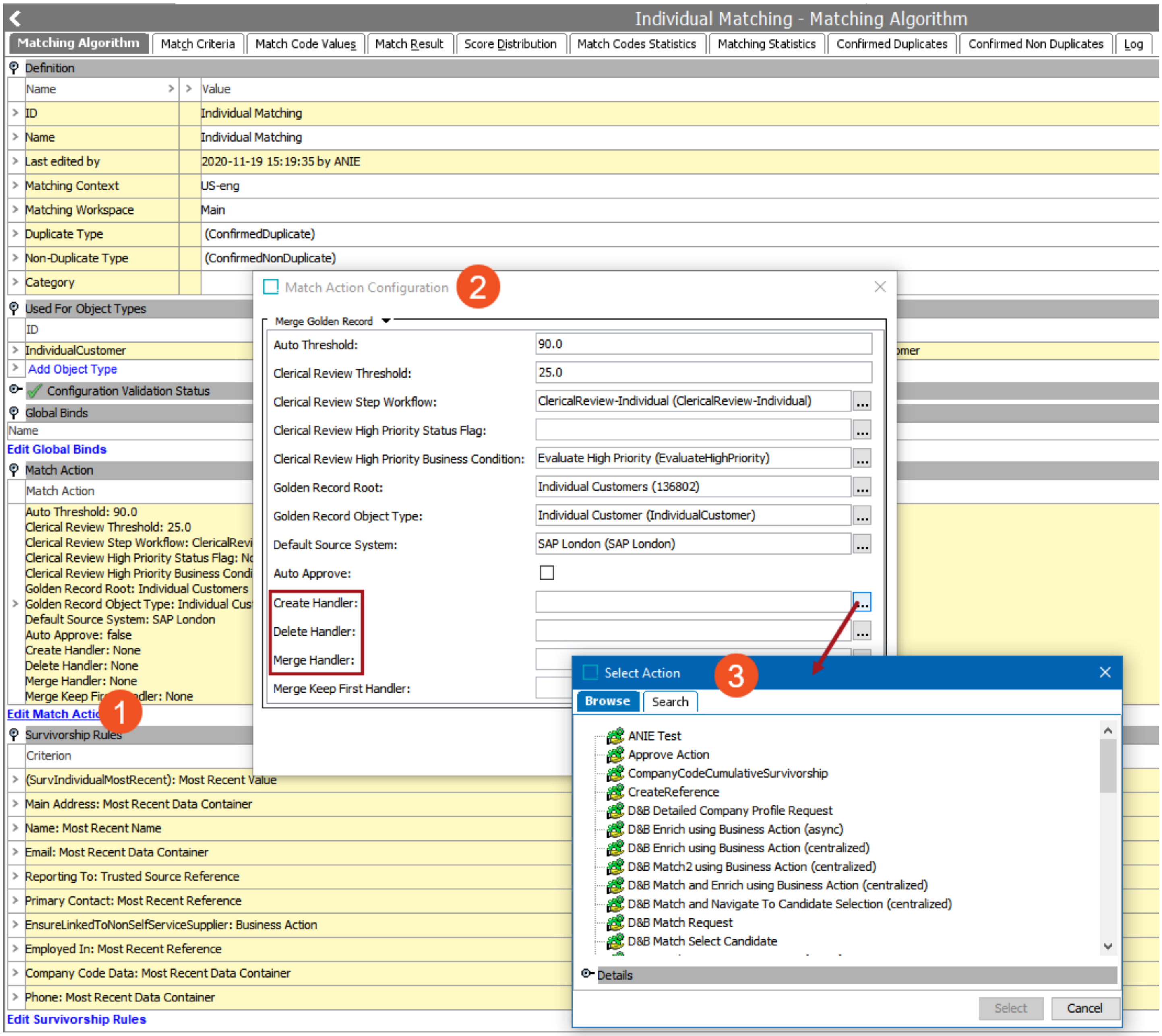
Golden Record Root	Golden_Record_Root (Golden_Record_Root)	...
Golden Record Object Type	Merge_Golden_Record (Merge_Golden_Record)	...
Default Source System	SAP_Test (SAP_Test)	...

8. In the 'Default Source System' parameter, Click the ellipsis button (...) and select the source system that should be used if no source system information is available upon import / merging of records.
9. 'Auto Approve' is used to automatically approve the golden records being created.

Note: Match and Merge does support the import of records without source system references.

Match Action Handlers for Match and Merge

Working with a golden record setup often requires specific actions when a golden record is changed (created, deleted, or merged). In these cases, the matching algorithm can be configured to call a business rule via a handler to allow for more granular processing of events. For example, when two existing golden records are merged, additional actions may need to occur in addition to the survivorship rules.



Individual Matching - Matching Algorithm

Definition

Name	Value
ID	Individual Matching
Name	Individual Matching
Last edited by	2020-11-19 15:19:35 by ANIE
Matching Context	US-eng
Matching Workspace	Main
Duplicate Type	(ConfirmedDuplicate)
Non-Duplicate Type	(ConfirmedNonDuplicate)
Category	

Used For Object Types

ID

IndividualCustomer

Add Object Type

Configuration Validation Status

Global Binds

Name

Edit Global Binds

Match Action

Match Action

Auto Threshold: 90.0
Clerical Review Threshold: 25.0
Clerical Review Step Workflow: ClericalRevi
Clerical Review High Priority Status Flag: No
Clerical Review High Priority Business Condi
Golden Record Root: Individual Customers
Golden Record Object Type: Individual Cus
Default Source System: SAP London
Auto Approve: false
Create Handler: None
Delete Handler: None
Merge Handler: None
Merge Keep Fi[1]dler: None

Edit Match Acti

Survivorship Rules

Criterion

(SurvIndividualMostRecent): Most Recent Value
Main Address: Most Recent Data Container
Name: Most Recent Name
Email: Most Recent Data Container
Reporting To: Trusted Source Reference
Primary Contact: Most Recent Reference
EnsureLinkedToNonSelfServiceSupplier: Business Action
Employed In: Most Recent Reference
Company Code Data: Most Recent Data Container
Phone: Most Recent Data Container

Edit Survivorship Rules

Match Action Configuration [2]

Merge Golden Record

Auto Threshold:	90.0
Clerical Review Threshold:	25.0
Clerical Review Step Workflow:	ClericalReview-Individual (ClericalReview-Individual)
Clerical Review High Priority Status Flag:	
Clerical Review High Priority Business Condition:	Evaluate High Priority (EvaluateHighPriority)
Golden Record Root:	Individual Customers (136802)
Golden Record Object Type:	Individual Customer (IndividualCustomer)
Default Source System:	SAP London (SAP London)
Auto Approve:	□
Create Handler:	
Delete Handler:	
Merge Handler:	
Merge Keep First Handler:	

Select Action [3]

ANIE Test
Approve Action
CompanyCodeCumulativeSurvivorship
CreateReference
D&B Detailed Company Profile Request
D&B Enrich using Business Action (async)
D&B Enrich using Business Action (centralized)
D&B Match2 using Business Action (centralized)
D&B Match and Enrich using Business Action (centralized)
D&B Match and Navigate To Candidate Selection (centralized)
D&B Match Request
D&B Match Select Candidate

The following handlers are available for the match and merge solution:

- **Create Handler**: The selected business action runs on the golden record after it has been created, but before survivorship rules run.

  The supplied golden records are retrieved by the STEP manager with the context and workspace defined by the matching algorithm (or Main workspace if defined as Approved workspace).

  The newly created golden record is bound to the 'Current Object' parameter.

- **Delete Handler**: The selected business action runs after the golden record is deleted. This takes place immediately after the merge handler.

  If no implementations of this handler exist, the default deactivation behavior is to:

  1. Remove all confirmed relations (confirmed duplicates and confirmed non-duplicates).
  2. Reallocate incoming references to surviving golden record and re-approve these references if they were approved already.
  3. Deactivate golden record, copy source information to survivor, and create 'Merged Into' reference to surviving record.
  4. Remove match code values.
  5. Remove links to potential duplicates.
  6. Delete any existing task in clerical review workflow.
  7. If using this handler, then the implementation must account for all the above steps in addition to any other required steps for deletion.

  If auto-approve is enabled, then the surviving golden record will be approved after the business logic is executed.

  The supplied golden records are retrieved by the STEP manager with the context and workspace defined by the matching algorithm or Main workspace if defined as Approved workspace.

  The newly deleted golden record is bound to the 'Current Object' parameter.

- **Merge Handler**: The selected business action runs when two golden records are merged. This handler is invoked after the surviving record has been determined and the record to be deactivated has been merged. Immediately after the call to this handler, the 'Delete Handler' is called to deactivate the relevant golden record.

  The supplied golden records are retrieved by the STEP manager with the context and workspace defined by the matching algorithm or Main workspace if defined as Approved workspace.

  The surviving golden record is bound to the 'Current Object' parameter. The golden record to be deactivated / deleted is bound to the 'Secondary Object' parameter.

- **Merge Keep First Handler**: The selected business condition evaluates when two golden records are being merged and allows identification of the golden record that should survive. If this handler is not used, the default behavior is to keep the golden record that was created first.

  The supplied golden records are retrieved by the STEP manager with the context and workspace defined by the matching algorithm or Main workspace if defined as Approved workspace.

  The surviving golden record is bound to the 'Current Object' parameter. The golden record to be deactivated / deleted is bound to the 'Secondary Object' parameter.

  The business condition should evaluate to 'True' to keep the first golden record or evaluate to 'False' to keep the second golden record.

These handlers are completely optional and may not be applicable to all solutions. For information about writing business rules that require access to two objects, such as two golden records in merge and split cases, see the **Secondary Object Bind** topic in the **Resource Materials** documentation.

Match and Merge Inbound Integration Endpoint

When setting up STEP as a data hub with match and merge, the data often flows into STEP via asynchronous Inbound Integration Endpoints. For information, see the **IIEP - Configure Match and Merge Importer** topic in the **Data Exchange** documentation.

Match and Merge Web Service Endpoint

The match and merge web service endpoint provides a synchronous alternative to the Inbound Integration Endpoint. It also delivers an answer to each request, advising the external system as to the result of the match and merge operation. The request sent to this service will include information such as:

- User name and password for access validation.
- A reference to a STEP context. For more on this, see the **Business Conditions and Contexts with Matching and Merging Match and Merge Web Service Type** section in this topic.
- A reference to a web service endpoint configured with the STEP match and merge type.
- Entity representations of each record to be imported. Any non-duplicates can be declared via the non-duplicate reference types, as defined by the matching component model.

When the request has been received, all incoming records will go through the following process.

1. **Validation -** In this step, the web service type validates incoming data to ensure it satisfies any minimum data requirements, e.g., record has an address or a last name. Records that are not successfully validated will not be stored in STEP and will be rejected.
2. **Standardization -** In this step, incoming data may be standardized, e.g., address standardization.
3. **Matching -** In this step, any existing matching or potentially matching records are identified. The outcome will be either:
    - new or updated golden records in STEP
    - a rejection from the service

In all cases, the web service's response will note:

- if the incoming record was validated
- any potential duplicates that are found
- information on the new / updated record itself
- if the record will be handled manually in a clerical review

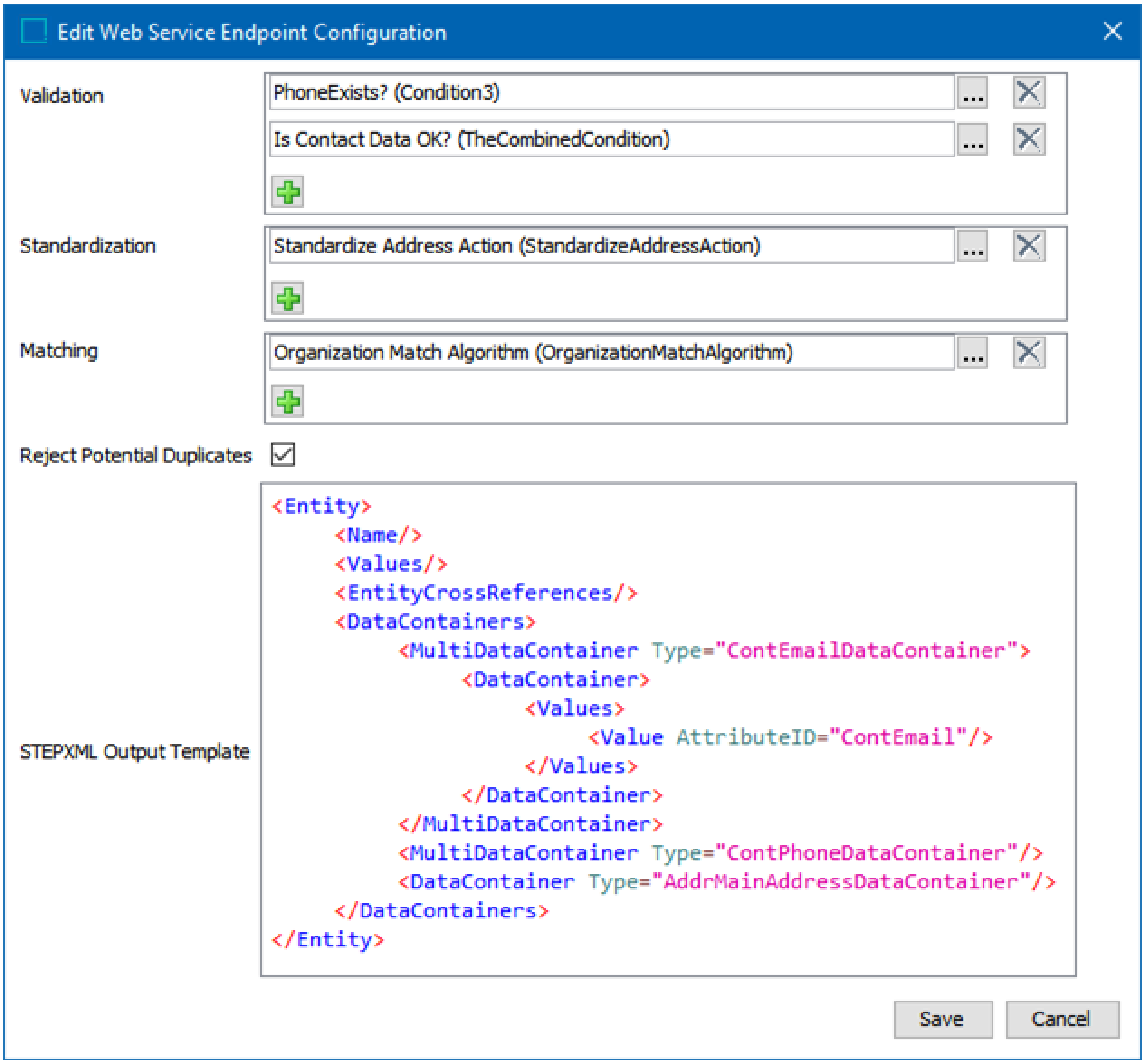
The exact behavior of a match and merge web service endpoint depends on the endpoint configuration.

For more information on web service endpoints, see the **Web Service Endpoint** topic in the **Data Exchange** documentation. More information on this web service type can be found in the STEP API documentation at [system]/sdk or by clicking the STEP API Documentation button on the STEP Start Page. For information on how to navigate to the Advanced STEPXML output template, see the **Advanced STEPXML Format** topic in the **Data Exchange** documentation.

To access the Match and Merge Web Service endpoint for a given system, navigate to the following URL: [your system URL: port]/MatchingWS/matching.

Configuring a Match and Merge Web Service Endpoint

With a new blank web service endpoint, select the 'Edit Web Service Endpoint Configuration' link below the fields. In the 'Edit Web Service Endpoint Configuration' dialog, the following items can be specified as desired:



Validation: PhoneExists? (Condition3)

Is Contact Data OK? (TheCombinedCondition)

Standardization: Standardize Address Action (StandardizeAddressAction)

Matching: Organization Match Algorithm (OrganizationMatchAlgorithm)

Reject Potential Duplicates ☑

STEPXML Output Template

```
<Entity>
     <Name/>
     <Values/>
     <EntityCrossReferences/>
     <DataContainers>
          <MultiDataContainer Type="ContEmailDataContainer">
               <DataContainer>
                    <Values>
                         <Value AttributeID="ContEmail"/>
                    </Values>
               </DataContainer>
          </MultiDataContainer>
          <MultiDataContainer Type="ContPhoneDataContainer"/>
          <DataContainer Type="AddrMainAddressDataContainer"/>
     </DataContainers>
</Entity>
```

To modify a parameter:

- Click the green plus button (➕) to add a criterion (... ☒).
- Click the ellipsis button (...) on a criterion to open the relevant selection dialog box. Make a selection and click

the Select button to continue.

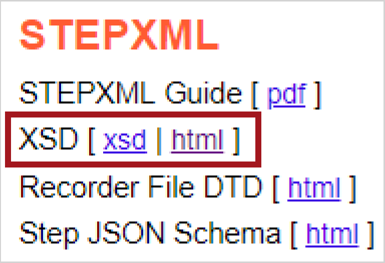
- Click the X button (☒) to delete a criterion.

Parameters available on the web service endpoint include:

- **Validation -** This field is constrained by business conditions that will limit what data is matched. If the data does not conform to any validation criteria denoted, it will reject the record. These operations will result in true or false. For more information, see the **Business Condition** topic in the **Business Rules** documentation.
- **Standardization -** This field is constrained by business actions that make the data being matched conform to a set standard format. For more information, see the **Business Actions** topic in the **Business Rules** documentation.
- **Matching -** This field is where a matching algorithm is set for the records that will be compared. In cases with different object types, there may be multiple matching algorithms. For more information, see the **Configuring Matching Algorithm** topic in the **Matching, Linking, and Merging** documentation.
- **Reject Potential Duplicates -** This option, if selected, will automatically reject potentially duplicated records. This means that if a record is imported into this Web Service, and if the matching algorithm finds any similar records that would lead to the creation of a clerical review task, it will just reject the record creation.
- **STEPXML Output Template -** The output template is an Advanced STEPXML structure that will populate and filter the STEP data in the response. The output template is based on Advanced STEPXML. The highest level tag in the output template should be the <Entity> tag. For example:

```
<Entity>
  <Name/>
  <Values>
    <Value AttributeID="IndLastName"/>
    <Value AttributeID="IndFirstName"/>
  </Values>
  <DataContainers>
    <MultiDataContainer Type="ContEmailDataContainer"/>
    <DataContainer Type="AddrMainAddressDataContainer"/>
  </DataContainers>
</Entity>
```

The XML template must contain a 'ComplexType EntityType' as defined in the STEPXML XSD. This output template is available in the STEP SDK and API documentation from [server]/sdk or by clicking the STEP API Documentation button on the STEP Start Page. It is located under the STEPXML section of this documentation.



STEPXML

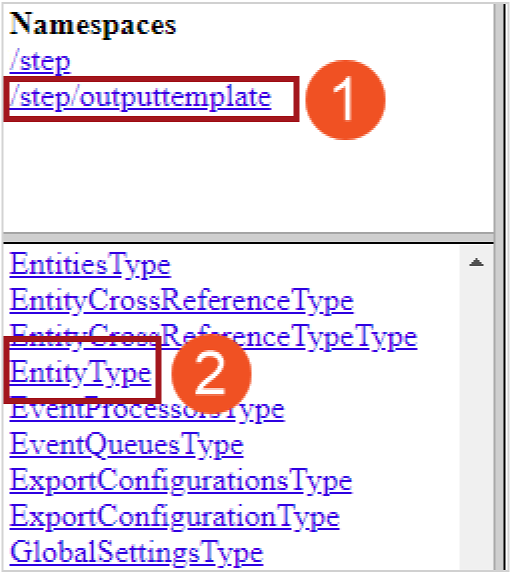
STEPXML Guide [pdf]

XSD [xsd | html]

Recorder File DTD [html]

Step JSON Schema [html]

Under the /step/outputtemplate namespace, select the EntityType line.



Namespaces

/step

/step/outputtemplate

EntitiesType

EntityCrossReferenceType

EntityCrossReferenceTypeType

EntityType

EventProcessorType

EventQueuesType

ExportConfigurationsType

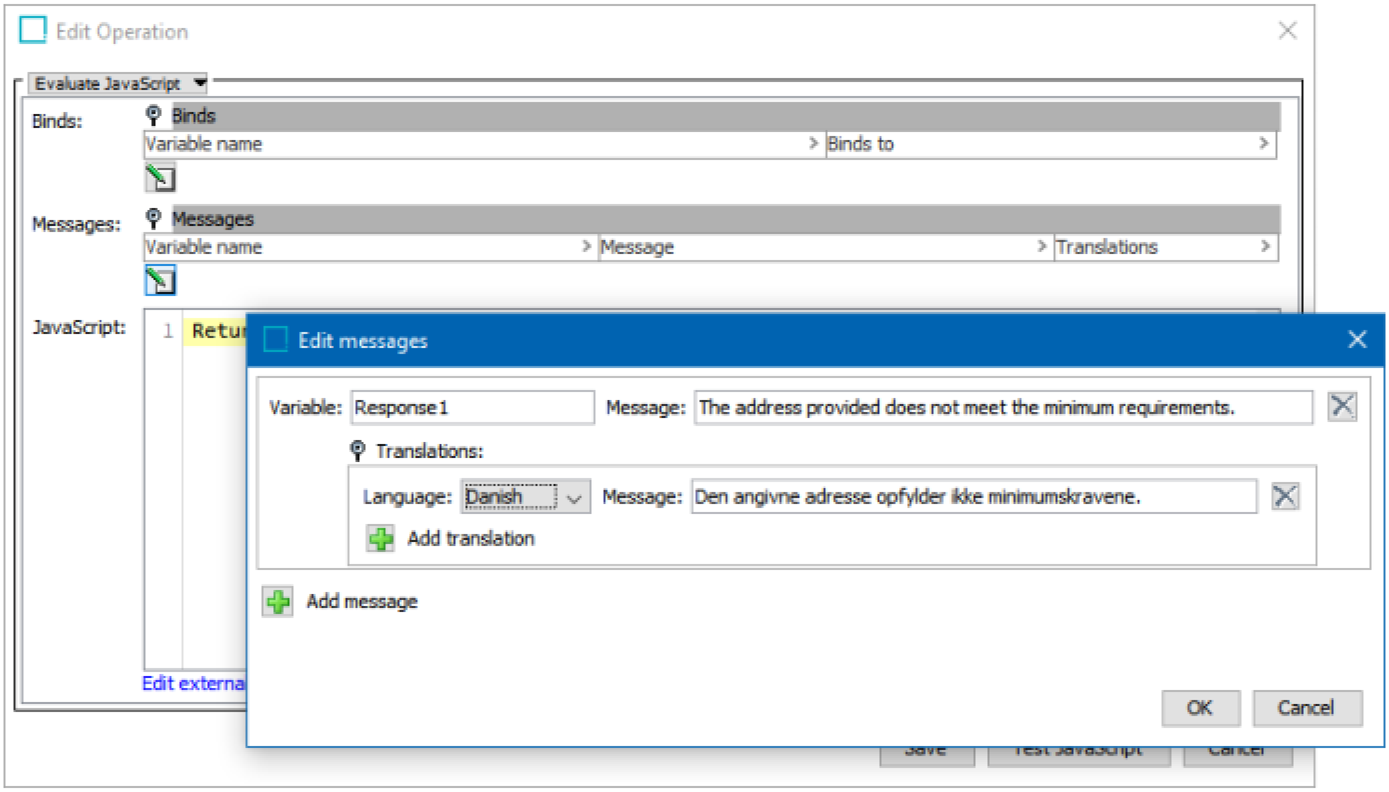
ExportConfigurationType

GlobalSettingsType

For more information on the Advanced STEP XML use, see the **STEPXML Tags and Examples** topic in the **Data Exchange** documentation.

Business Conditions and Contexts with Matching and Merging Match and Merge Web Service Type

When sending a web service request, the request will include the current context. This request is sent to STEP, and STEP returns a value with the context language included. Should the 'Evaluate JavaScript Business Condition' fail, then the included context will signal which translation of the business condition message to include.



Variable: Response1 Message: The address provided does not meet the minimum requirements.

Language: Danish Message: Den angivne adresse opfylder ikke minimumskravene.

For more information, see the **Business Condition: Evaluate JavaScript** topic in the **Business Rules** documentation.

Match and Merge Event Processor

The event processor is responsible for comparing golden records that are already present in the system. It is also the match and merge event processor that initiates possible duplicates into merge clerical review.

There are several reasons why event processors are necessary in match and merge match algorithms. Sometimes, the initial source records do not contain enough information to auto merge, and then both records are created in the system. If the match score surpasses the clerical review threshold, the records will be sent to clerical review. For more information on manual merging, see the **Match and Merge Clerical Review Merge** topic in this documentation. Other times, new updates to old records make it possible to match the records as duplicates.

It is recommended to run the event processor for Match and Merge every minute, and trigger on changes to all data elements used in the match criteria. Unless special conditions exist, the event processor should generate / update match code values and run matching algorithm.

One event processor can be shared across several matching algorithms. This will, however, force these matching algorithms to run in the same process, potentially resulting in suboptimal performance on systems with a high CPU count.

The comparison by event processor happens asynchronous to the actual import, leaving up to a minute between importing a record that should go into the clerical review workflow and the record being visible in said workflow.

Event processors keep a background process log, so you can determine when events were processed and what actions were taken in response. Additionally, event processor performance measurements are available on the Statistics tab for both Matching Algorithms

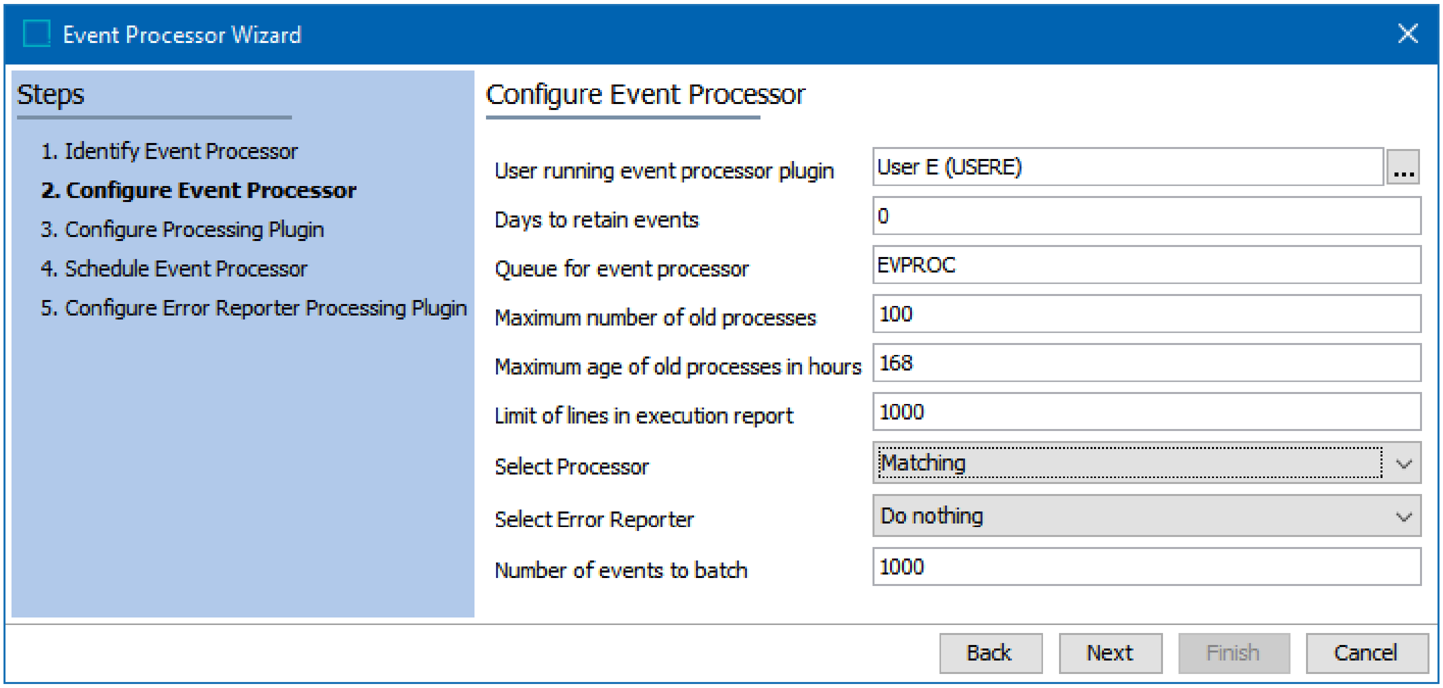
Configuration of Match and Merge Event Processor

To create an event processor include:

1. Create a matching event processor by following the steps outlined in the **Creating Event Processors** topic of the **System Setup / Super User Guide** documentation.

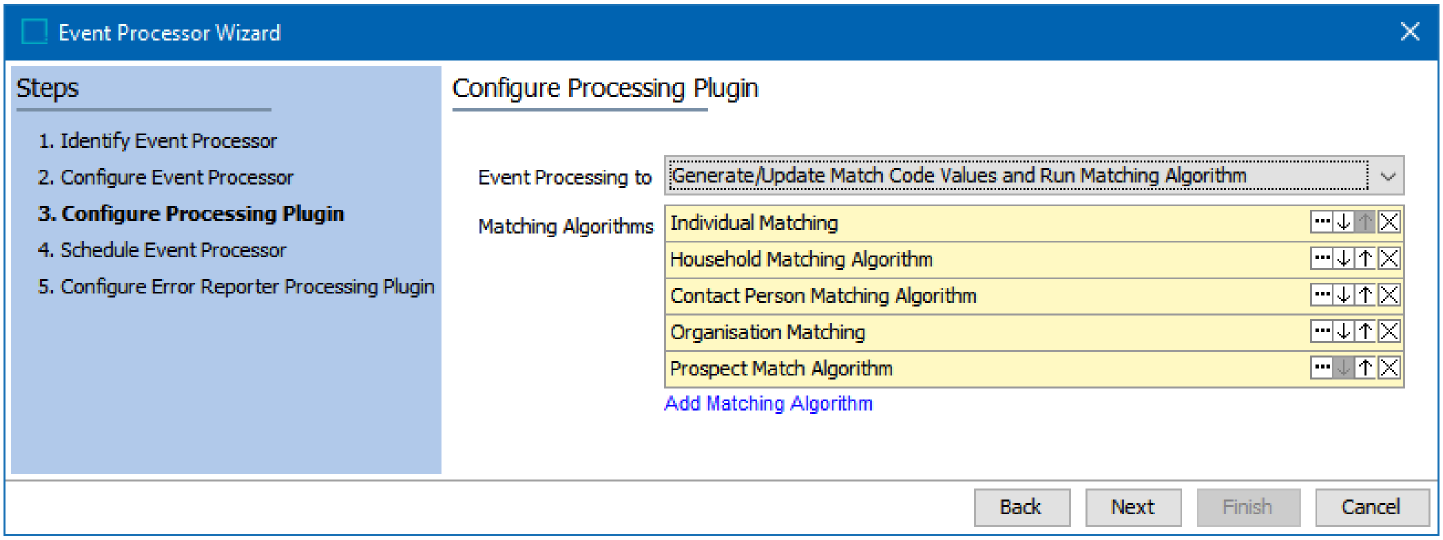
   This section guides you to select the type of processing required (generate match code and update and/or run matching algorithm), the matching algorithm(s), establish the schedule, set the queue status, and set the event triggering definitions.

2. Enable the matching event processor by following the steps outlined in the **Enabling Event Processors** section of the **Running an Event Processors** topic in the **System Setup / Super User Guide** documentation. When creating the event processor, on the second step of the wizard, set the Processor option to 'Matching.'


Event Processor Wizard
Steps
1. Identify Event Processor
2. Configure Event Processor
3. Configure Processing Plugin
4. Schedule Event Processor
5. Configure Error Reporter Processing Plugin
Configure Event Processor
Back
Next
Finish
Cancel

User running event processor plugin	User E (USERE)
Days to retain events	0
Queue for event processor	EVPROC
Maximum number of old processes	100
Maximum age of old processes in hours	168
Limit of lines in execution report	1000
Select Processor	Matching
Select Error Reporter	Do nothing
Number of events to batch	1000

When creating the event processor on the third step of the wizard, set the Event Processing to Generate/Update Match Code Values and Run Matching Algorithm. Add the matching algorithms that the event processor should maintain.



Event Processor Wizard
Steps
1. Identify Event Processor
2. Configure Event Processor
3. Configure Processing Plugin
4. Schedule Event Processor
5. Configure Error Reporter Processing Plugin
Add Matching Algorithm
Back
Next
Finish
Cancel

Configure Processing Plugin

Event Processing to: Generate/Update Match Code Values and Run Matching Algorithm

Matching Algorithms:
- Individual Matching
- Household Matching Algorithm
- Contact Person Matching Algorithm
- Organisation Matching
- Prospect Match Algorithm

Match and Merge Traceability

Match and merge is designed for the data hub, and as such, the engine must know how records are identified by the source systems. In match and merge, this information about a source record is reflected by the Source Record ID Attribute and Source System ID Attribute set in the component model. To set up the component model, see the 'Match and Merge Component Model' section of the **Configuring Match and Merge** topic in this documentation.

- No two records of the same object type in STEP should ever share the same Source Record ID for the same Source System.
- Source Systems may have several IDs on a single record in STEP.
- It is expected that different source systems assign different IDs to the same customer.

The Workbench displays revision data like the Source System ID and Source Record ID using the revision comment field.

Source information can be shown in the Web UI using the **Golden Record Source Information Web UI Component** and the **Golden Record Source Traceability Screen** sections of this topic.

Match and merge imports updates to data directly into golden records, by default discarding non-surviving data. For information about how to retain this data for revision history and improved unmerge capabilities, see **Storing Source Data for Golden Records** section of this topic.

Match and Merge Traceability in Workbench

All merge and unmerge information are displayed in the 'Comment' field along with the source information on the revisions of the individual records.

When merging, the surviving golden record will have the ‘Merged into’ information with the object ID that was merged into this golden record. When unmerging, the IDs of the reactivated or new golden records will be listed in the ‘Unmerged into’ information.

Example 1 - Removing a Record from a Golden Record

The following example shows a golden record with a record mistakenly merged into it and then unmerged.

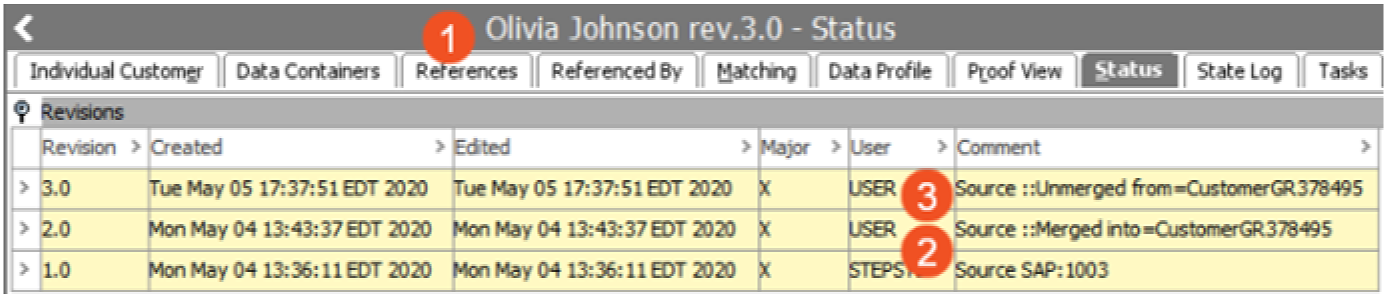
1. **Golden Record Name** — Oliver Johnson
2. **Deactivated Golden Records** — CustomerGR378497 and CustomerGR378499
3. In Revision 2.0 and Revision 3.0, these two deactivated golden records are merged into the Oliver Johnson golden record, leaving the ‘Merged from’ traceability information.
4. In Revision 5.0, the CustomerGR378499 golden record is unmerged from the Oliver Johnson record and reactivated, leaving the ‘Unmerged into’ traceability information.

On an active golden record, the ‘Merged from’ information is stored with the object ID of the golden record into which it was merged. When the golden record is reactivated in an unmerge operation, the ‘Unmerged into’ information is stored as a reference.

Example 2 - Tracing a Records Removal from a Golden Record

The following example shows how the removed record above — CustomerGR378499 — traces unmerging.

1. Currently, the Olivia Johnson, CustomerGR378499, golden record is selected.
2. Olivia Johnson is merged into the active golden record — Oliver Johnson, CustomerGR378495.
3. In Revision 3.0, this merging was reversed and unmerged from the Oliver Johnson record. This action re-activates the Olivia Johnson golden record.



Revision	Created	Edited	Major	User	Comment
3.0	Tue May 05 17:37:51 EDT 2020	Tue May 05 17:37:51 EDT 2020	X	USER	Source ::Unmerged from=CustomerGR378495
2.0	Mon May 04 13:43:37 EDT 2020	Mon May 04 13:43:37 EDT 2020	X	USER	Source ::Merged into=CustomerGR378495
1.0	Mon May 04 13:36:11 EDT 2020	Mon May 04 13:36:11 EDT 2020	X	STEPS	Source SAP:1003

On the Olivia Johnson golden record, the Oliver Johnson record is stored as a reference of the 'Unmerged From' reference type. For more information, see the **Configuring Unmerge in Workbench** topic in this documentation.

Storing Source Data for Golden Records

During a match and merge operation, the imported data is merged directly into golden records. If survivorship rules dictate the existing data should survive, the incoming data is discarded by default. When this data is discarded, a later unmerge of the merged records would leave gaps of missing data after unmerge.

To store the source records, the golden record object type can be added to 'Keep Source Records for Golden Record Object Types' in the Matching-Merge component model.

A Source record is only stored if it includes a source record ID as defined in the Matching-Merge Golden Record component model for the 'Source Record ID Attribute' entry.

Before enabling the storage of source data, ensure all data container keys are correctly defined. This requirement is because changing the key definition later will impact the validity of the existing stored source data. This creates issues where data container keys become incomplete or data container instances are duplicates. There is no manner to find these data container source data as there is with golden records.

Removing a golden record object type from the ‘Keep Source Records for Golden Record Object Types’ will not delete source data that is already stored in the system. After deleting this object type, the system will just stop storing additional source data for that object type.

Source data storage includes revision history and improving the data lineage functionality. For more information, see the 'Value Traceability Popup' section of the **Golden Record Source Traceability Screen** topic in the **Web User Interfaces / Web UI Getting Started** documentation.

Note: Storage of source data is only supported on 'Merge Golden Records' object types.

Important: Without a full import of the source data after enabling the storage of source data, the source data in the system will be incomplete. Future partial updates will complicate the unmerge process since it will be unclear what is the full dataset from each source.

Incremental Updates and Source Data Storage

The technical storage of source data behaves much like if data were imported by standard STEPXML importer. Deleting an attribute value is done by sending an empty tag in STEPXML.

Multi-Valued Data Container without a Defined Data Container Key

Incremental updates of records must always include either all instances or no instances on the data container type.

- Existing instances that are not part of the update are deleted.
- A data container type that is fully excluded in the update is left unchanged.

Multi-Valued Data Containers with a Defined Data Container Key

Incremental updates of records can update a subset of data container instances without including all instances.

- Instances with a matching key are updated.
- If incoming data includes data container instance that match no existing key, a new instance is created.
- Existing data container instances cannot be deleted by import.

Multi-Valued References

- If no matching target is found, a new reference is created.
- Existing reference instances cannot be deleted.

Match and Merge Source Data Storage Impact

Storing source data increases the diskspace used by the underlying storage system. The extent of the increase depends on the frequency of source record updates.

Source data is stored persistently in the system database, thus it will be included in standard backup procedures. Source data does not impact memory use of In-Memory solutions

The ‘Source Record Data Management – Historical Values Cleanup’ event processor described below not only cleans historical values, but it also optimizes the storage to keep disk usage as low as possible simultaneously.

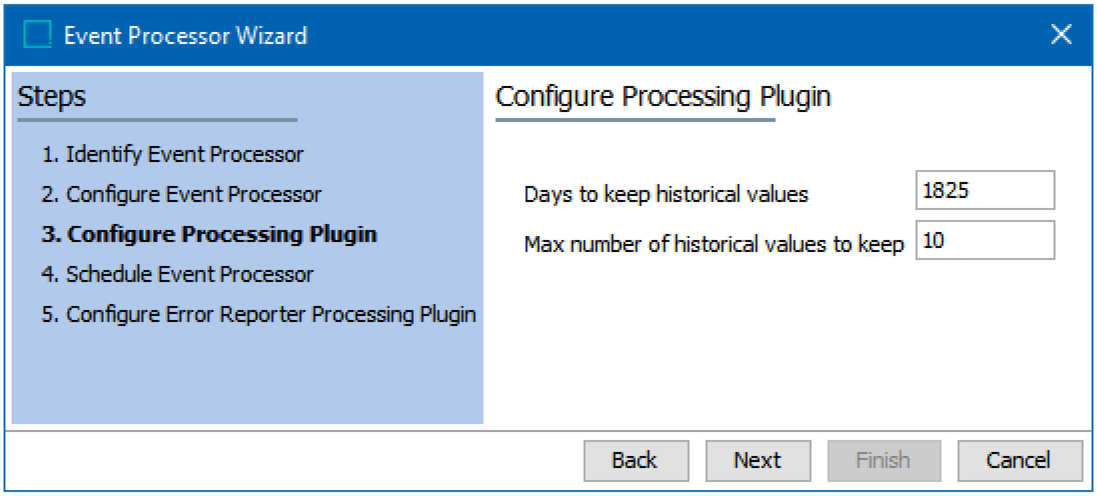
When importing new records, the individual records may require an additional 20-30% of space to accommodate for the source records. However, with minor updates to existing golden records, source data may require 250-

300% of additional disk space for the individual records. To minimize this impact, the ‘Historical Values Cleanup’ event processor should be configured. This event processor will optimize the revisions, so the storage impact is kept to a minimum. Dependent on the average amount of changing values at each import, the extra storage impact for updates could go as low as around 50-80%.

Source Record Storage Historical Values Cleanup

To keep the extra storage impact stable over time, the historical source data values should eventually be purged, much like normal object revision history.

To purge historical source data , create a new event processor of type Source Record Data Management – Historical Values Cleanup.


Event Processor Wizard
Steps
1. Identify Event Processor
2. Configure Event Processor
3. Configure Processing Plugin
4. Schedule Event Processor
5. Configure Error Reporter Processing Plugin
Back
Next
Finish
Cancel

Configure Processing Plugin

Days to keep historical values 1825

Max number of historical values to keep 10

There are two processing plugin configurations:

- **Days to keep historical values** – The number of days historical values are kept before purging
- **Max number of historical values to keep** – The number of values, changing over time, to keep

The data purge occurs if just one of the two configuration criteria are met.

The purging logic is individual per source record and per attribute / reference / data container, so frequent-changing values will not cause other historical values to be purged before they were supposed to be purged.

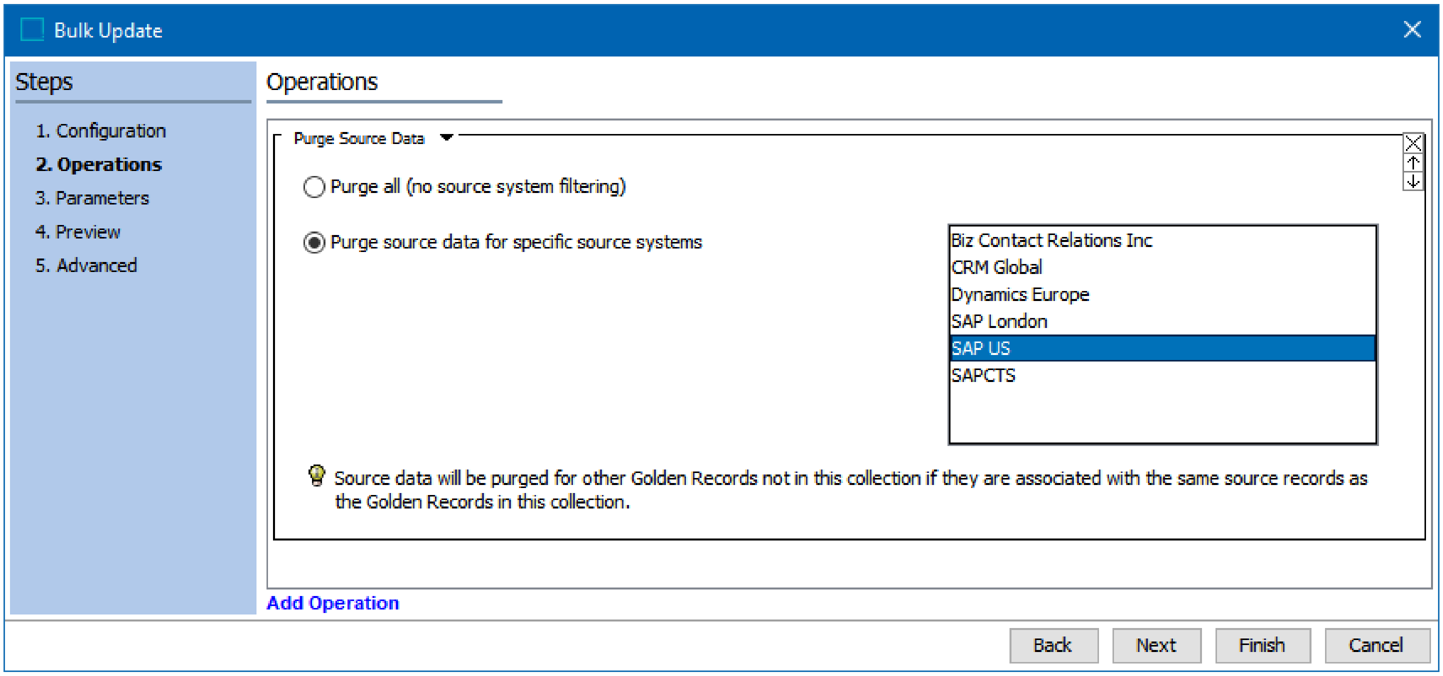
To disable the purging of historical values, but keep the disk usage optimization which is described in the next section, simply set both configuration values to a very high number.

The finished event processor will display in workbench. Before attempting to use the event process, ensure that at least one valid 'Triggering Object Types' are configured under the 'Event Triggering Definitions' tab on the event processor.

Note: If using workspaces, the 'Triggering Workspace' should normally just be the Main workspace.

Source Record Storage Purge using Bulk Update

Source Record Data can be manually purged using a Bulk Update operation. This will purge Source Record Data from selected records, with the option of only purging source data from specific source systems.



Purge Source Data

○ Purge all (no source system filtering)

● Purge source data for specific source systems

- Biz Contact Relations Inc
- CRM Global
- Dynamics Europe
- SAP London
- SAP US
- SAPCTS

Source data will be purged for other Golden Records not in this collection if they are associated with the same source records as the Golden Records in this collection.

This operation has two options:

- Purge all of the source data
- Purge source data for specific source systems

Important: Source data is purged for other golden records not in the collection if they are or were associated with the same source records as the golden records in the collection. This consideration applies for golden records and source records that were part of a merge or unmerge.

Golden Record Source Information Web UI Component

The 'Golden Record Source Information' component is configurable on a node details screen for an entity and displays source record information including:

- Source Record, which displays the ID of the source record.
- Source System, which displays the name of the source system from which the record originated.
- Created, which displays the date the source record was created.
- Last Updated, which displays the date the source record was last updated.

This component offers a general overview of the golden record's history and from which systems it has received data. Once added as a child component on a node details screen, no further configuration is required.

For more information, see the **Node Details Screen** section of the **Web User Interfaces / Web UI Getting Started** documentation.

Golden Record Source Traceability Screen

The 'Golden Record Source Traceability Screen' offers a more comprehensive look at a golden record's revision history than the 'Golden Record Source Information' component. It can be configured with header rows that display the values of attributes, attribute groups, data container attributes, and reference types. This allows the user to track changes to individual aspects of a golden record, it displays from which system the new values originated, and it designates when the changes were made.

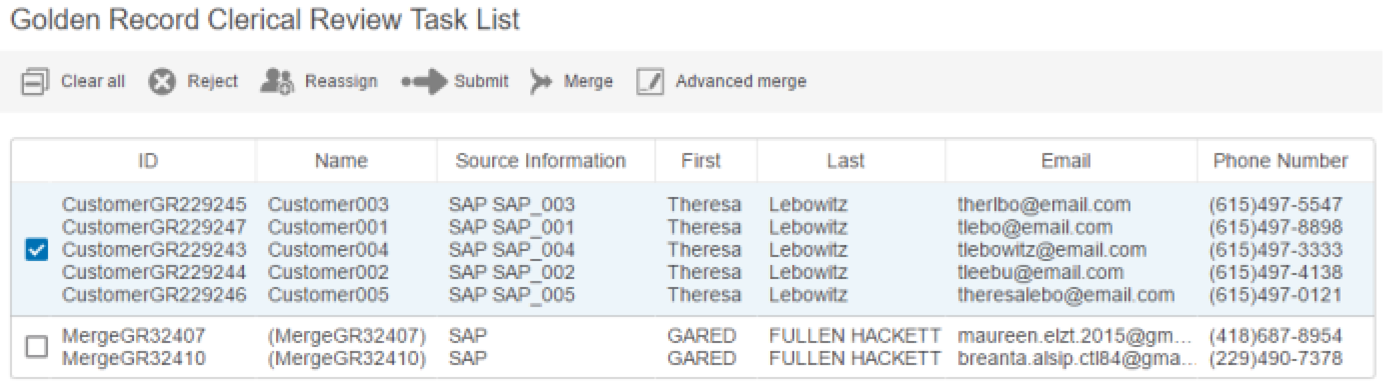
For more information, see the **Golden Record Source Traceability Screen** section of the **Web User Interfaces / Web UI Getting Started** documentation.

Match and Merge Clerical Review - Merge

The match and merge solution is supplemented by a special Web UI clerical review task list and advanced merge screen that assist in clerical reviews for potential duplicates.

The golden record clerical review task list screen displays all potential duplicates found in a specific golden record clerical review workflow or workflow state. From this screen, golden records are grouped into tasks and can be rejected as duplicates via the 'Reject' action button or acknowledged as duplicates and merged together via the 'Merge' or 'Advanced Merge' action buttons. These tasks can also be reassigned to other users via the 'Reassign' action button or submitted to another state in the workflow via the 'Submit' action button.

To arrive at this screen, a golden record must have been flagged as a potential duplicate by the relevant matching algorithm during import. This means that it has fallen within the clerical review threshold of the matching algorithm, and thus, initiated into a clerical review workflow where it can either be rejected as a duplicate or merged with other records. Potential duplicates that are matched together are organized into distinct tasks in the workflow, as pictured below.



Golden Record Clerical Review Task List

	ID	Name	Source Information	First	Last	Email	Phone Number
☑	CustomerGR229245 CustomerGR229247 CustomerGR229243 CustomerGR229244 CustomerGR229246	Customer003 Customer001 Customer004 Customer002 Customer005	SAP SAP_003 SAP SAP_001 SAP SAP_004 SAP SAP_002 SAP SAP_005	Theresa Theresa Theresa Theresa Theresa	Lebowitz Lebowitz Lebowitz Lebowitz Lebowitz	therlbo@email.com tlebo@email.com tlebowitz@email.com tleebu@email.com theresalebo@email.com	(615)497-5547 (615)497-8898 (615)497-3333 (615)497-4138 (615)497-0121
□	MergeGR32407 MergeGR32410	(MergeGR32407) (MergeGR32410)	SAP SAP	GARED GARED	FULLEN HACKETT FULLEN HACKETT	maureen.elzt.2015@gm... breanta.alsip.ctl84@gma...	(418)687-8954 (229)490-7378

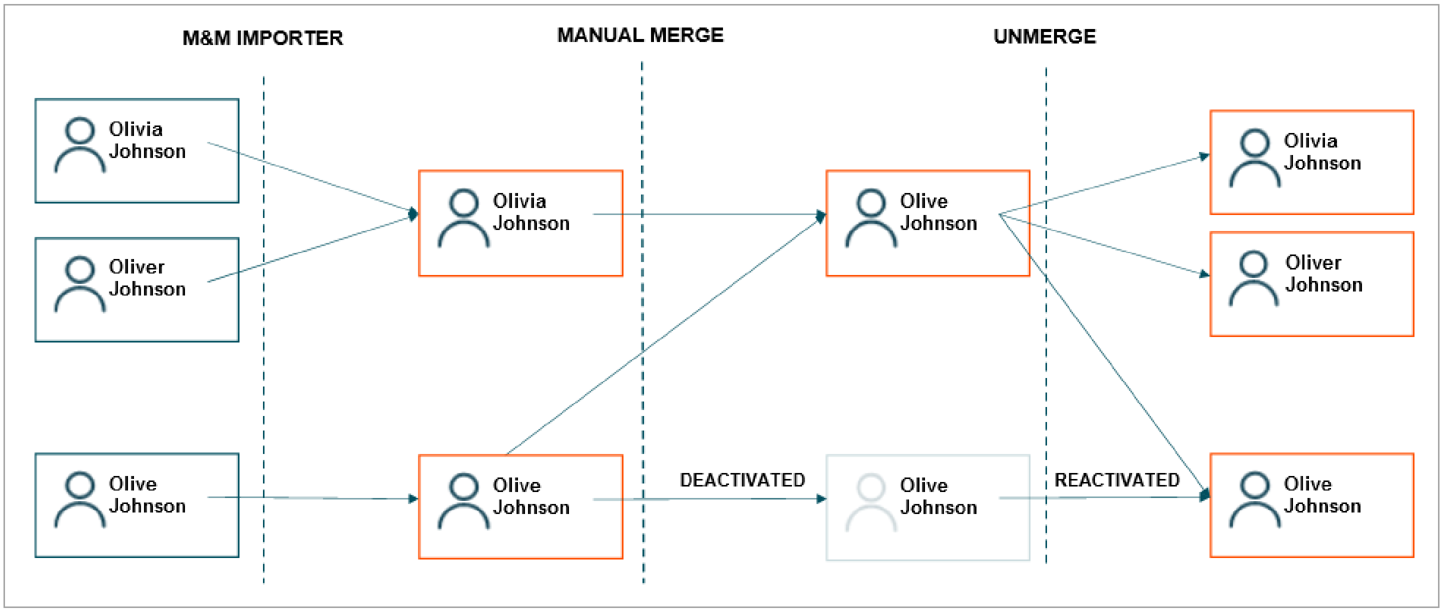
Note: This screen is only to be used with the match and merge solution. The data stewards are the primary users of this screen who can decide if products are duplicates or non-duplicates. A separate user group is recommended to be used for this task.

For information on setting up and using the golden record clerical review screen as well as the advanced Merge feature, see the **Golden Record Clerical Review Task List** section of the **Web User Interfaces / Web UI Getting Started** documentation.

Match and Merge Clerical Review - Unmerge

Unmerge allows users to untangle records that have been wrongly merged. A wrong merge can happen either as the result of a manual action or by auto merge.

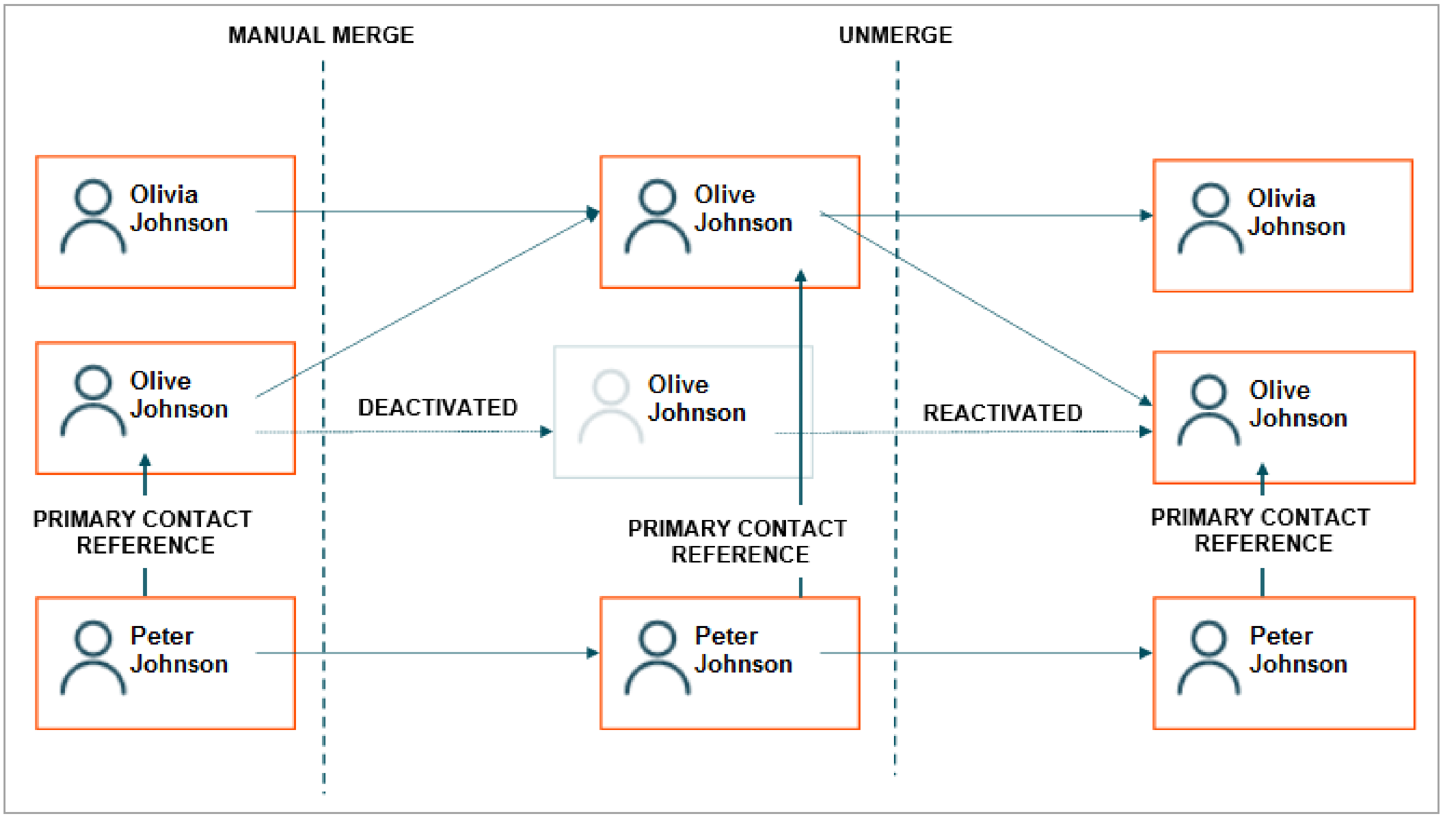
In the example below, the three customer records Olivia, Oliver, and Olive have been merged into one, and the unmerge operation needs to untangle these records, effectively splitting one golden record into separate golden records. In the Unmerge operation, we wish to reactivate the deactivated Olive Johnson record. This functionality is available to incoming records with source record IDs as well as deactivated golden records. The unmerge operation can either be done as part of a workflow or as an ad-hoc operation, and restore the relevant data back to it. Further, we need to move all updates concerning Oliver Johnson to a new record.



When reactivating golden records or moving source records, the unmerge logic will calculate the best possible values for all golden records.

When the user has identified which source records and manual updates belong on each reactivated golden record or moving source record, the unmerge logic will use survivorship rules to calculate the possible values for these golden records. In some scenarios, it is not possible to revert to determine the correct version of the golden record, which is why the second step of the unmerge allows the user to verify and correct them. The user cherry-picks the final values on each golden record.

Any inbound references, such as the one with Peter Johnson pictured below, that were created from a Match and Merge import using source record ID and stored as source records, will attempt to be auto-assigned back to the correct golden record. All inbound references that could not automatically be reassigned will be shown in a dialog when completing the unmerge operation, with a count of how many were left unchanged, grouped by reference type.



The unmerge operation in STEP can either be done as part of a workflow or as an ad-hoc operation.

The logic to revert the changes is not supported on multi-valued data containers and references.

First, it will try to revert values as they were prior to being incorrectly merged. This action is applicable for both merged golden records that are now being reactivated as well as source records that were wrongly auto-merged into the golden record. The reversion logic has two paths of removing values and reverting to the original source records.

For merged golden records, the record is now being reactivated. The 'Merged into' traceability will determine whether or not to revert back to a certain value when the values originally came from either manual entry or imports without source record IDs. For more information on traceability, see the **Match and Merge Traceability** topic in this documentation.

For source records that were incorrectly auto-merged into the golden record during an import, since all existing revisions will have the source information, moving a source record to another golden record will revert those values coming from that particular source.

The logic to revert the changes is not supported on multi-valued data containers and references.

After this operation, the remaining associated source records, if any, are applied to all the golden records using the configured survivorship rules. This will ensure that attributes that had no valid value to revert back to will get the correct original value from the sources written.

When unmerging, the system will restore historical values to the system. These values will get a STEP update timestamp equaling the current time. When reverting to historical values, the system needs to use current timestamps when writing the data to the golden record. This means that value data will seem more recent than it actually is, which could impact functionality like and the survivorship rules therefore might choose the wrong surviving values in a 'most recent' pattern. To avoid this, it is recommended to always use 'Last Edit' attributes when importing and when configuring the survivorship rules. The logic is that if 'Last Edit' attributes are used, unmerge will then also revert these last edit dates, and the later survivorship rules will then choose the correct surviving values.

If 'Auto Approve' is enabled in the match action settings on the matching algorithm and the object type is workspace revisable, the golden records will be auto-approved, and any business conditions and/or business actions enabled for 'on approve' will be evaluated.

Any inbound references that were created from a Match and Merge import using source record ID and stored as source records will attempt to be auto-assigned back to the correct golden record. All inbound references that could not automatically be reassigned will be shown in a dialog when completed the unmerge operation with a count of how many were left unchanged, grouped by reference type.

Note: For imports with source record IDs, enabling and configuring the storage of source record data improves the unmerge result. The data of any imports done before this configuration will not be preserved. For more information, see **Storing Source Data for Golden Records** topic in this documentation.

Unmerge uses the survivorship rules. For more information, see the **Survivorship in Match and Merge** topic in this documentation.

In addition to these items, workflow widgets will be added to the Web UI Hompeage as well as a few additional configuration options.

Note: Unmerge is only supported for object types that are the 'Merge Golden Records' object types.

Important: One aspect that is crucial to truly unmerging records is the storage of source data. This aspect of unmerge requires additional configuration. While it is not required for the basic unraveling of golden records, to truly revert to the original records without the potential for data loss, users should configure this source data storage. For more information, see the **Storing Source Records for Golden Records** topic in the **Data Governance** documentation.

For a breakdown of the actual unmerge process, see the **Unmerge - Distribute Source Records** topic in this documentation.

Note: The unmerge operation is only available in Web UI.

Configuring Unmerge in Workbench

Prior to configuring unmerge, users must ensure that a Match-Merge solution is properly configured on their system. The unmerge process relies on existing matching algorithms as well as configured golden record object types. For more information, see the Match and Merge documentation.

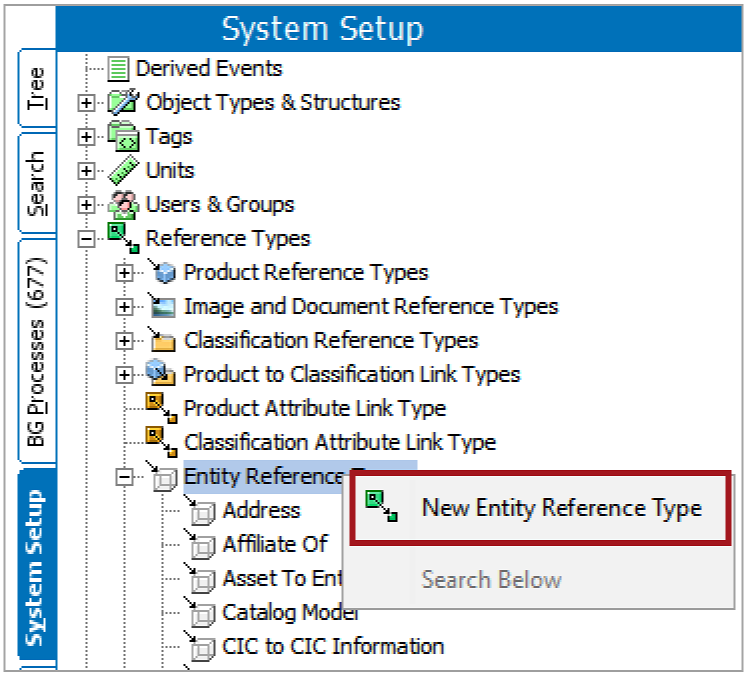
To set up unmerge for Match and Merge, several steps are needed:

- An 'Unmerged From' reference type is needed in the component model.
- In most applications, unmerge has to be part of a workflow.
- Unmerge is available as a screen in Web UI, which requires setup. Also, initiation of unmerge from a Golden Record details screen. For more information, see the **Configuring Unmerge in Web UI** topic in this documentation.

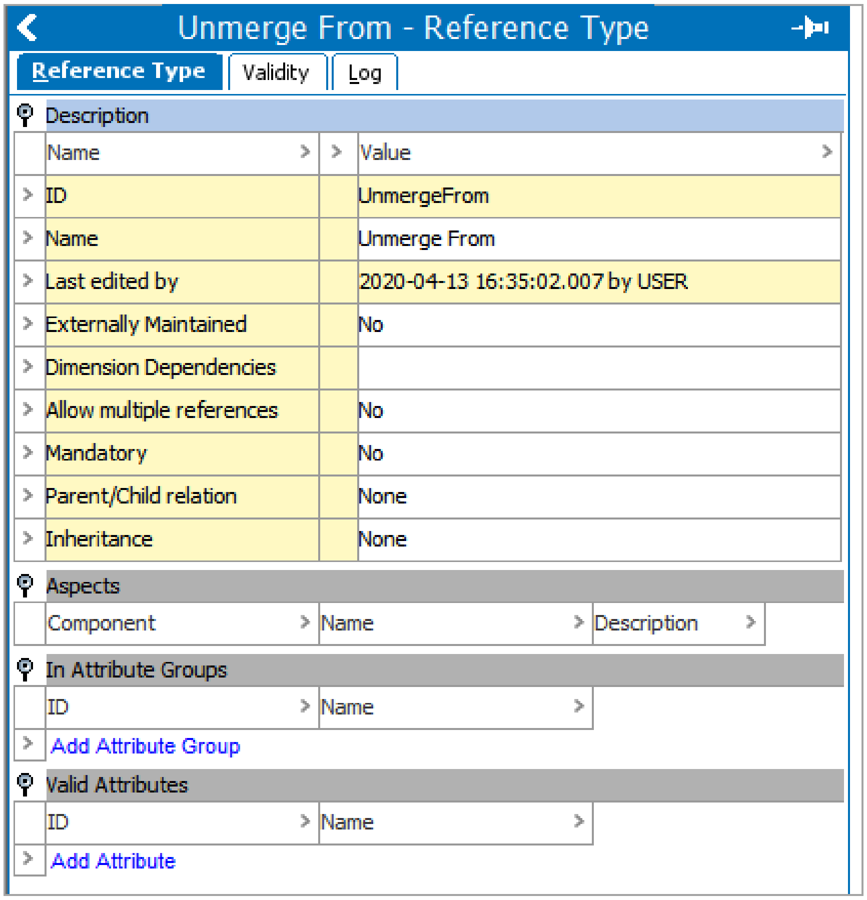
Configuring Unmerge in the Match and Merge Component Model

An 'Unmerged From' Reference Type must be configured in the component model. When data is unmerged from a golden record into another golden record, a reference of this reference type will be created.

1. From the System Setup tab, under 'Reference Types,' create a new Entity-to-Entity Reference Type.



For this reference type, the metadata should be set as shown below:



Unmerge From - Reference Type

Description

Name	Value
ID	UnmergeFrom
Name	Unmerge From
Last edited by	2020-04-13 16:35:02.007 by USER
Externally Maintained	No
Dimension Dependencies	
Allow multiple references	No
Mandatory	No
Parent/Child relation	None
Inheritance	None

Aspects

Component	Name	Description

In Attribute Groups

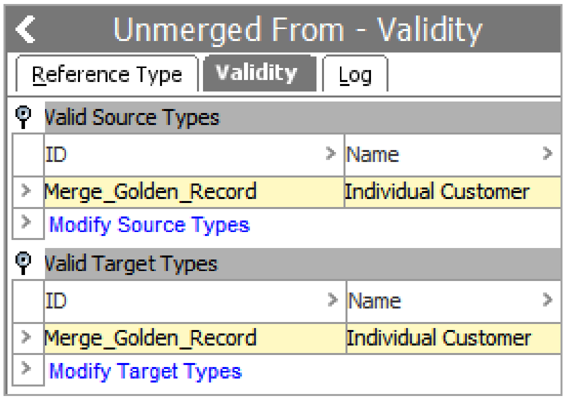
ID	Name

Add Attribute Group

Valid Attributes

ID	Name

Add Attribute



Unmerged From - Validity

Valid Source Types

ID	Name
Merge_Golden_Record	Individual Customer

Modify Source Types

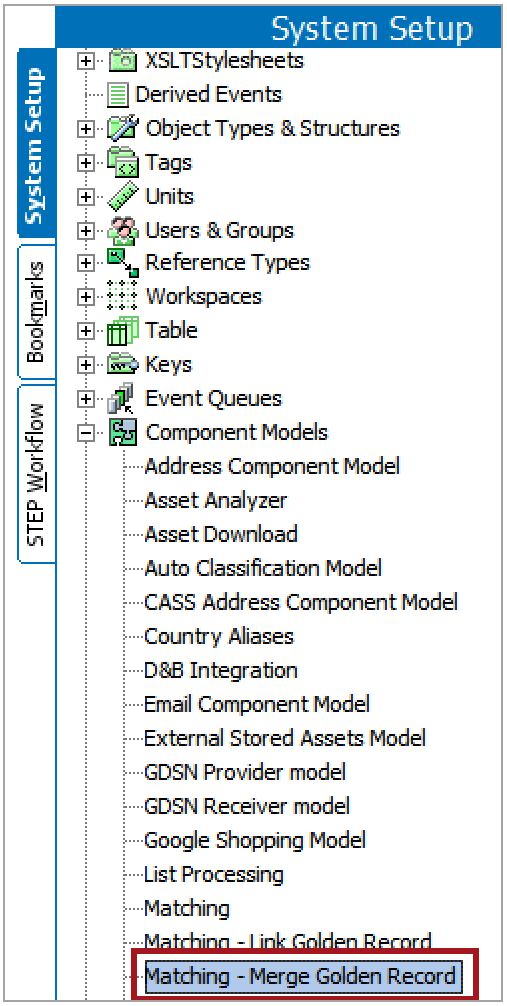
Valid Target Types

ID	Name
Merge_Golden_Record	Individual Customer

Modify Target Types

- **Valid Source Types -** Entity types that should support unmerge
- **Valid Target Types** - Same as the source types
- **Dimension Dependencies -** None
- **Allow Multiple References -** No. Ensure this option is unchecked during initial creation.
- **Externally Maintained** - No
- **Mandatory** - No
- **Sub Product Inheritance Settings -** No Inheritance

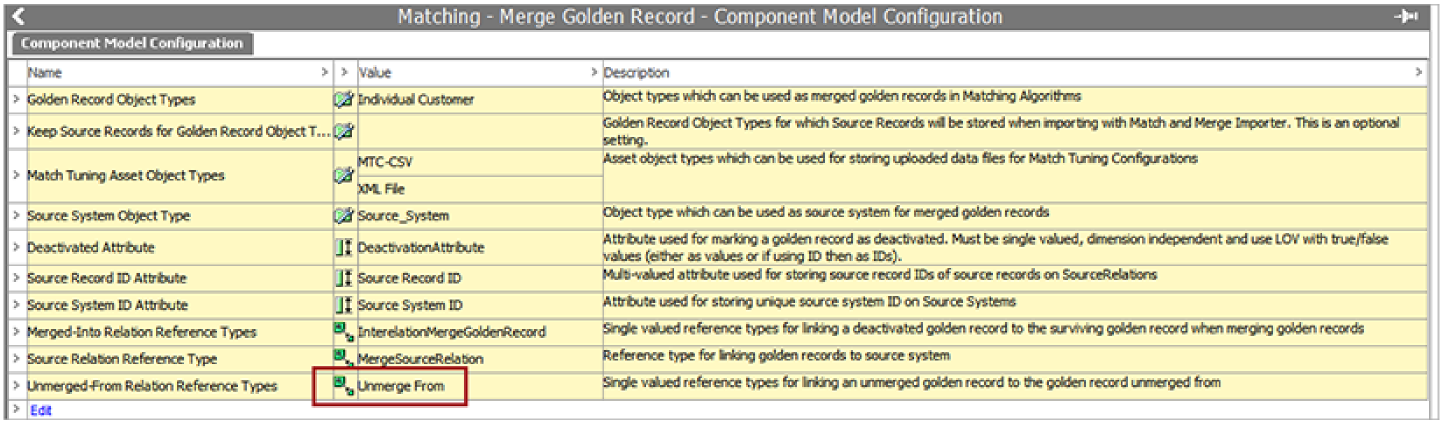
Next, in the Component Model configuration of System Setup, select the Matching-Merge Golden Record option.



System Setup

- XSLTStylesheets
- Derived Events
- Object Types & Structures
- Tags
- Units
- Users & Groups
- Reference Types
- Workspaces
- Table
- Keys
- Event Queues
- Component Models
  - Address Component Model
  - Asset Analyzer
  - Asset Download
  - Auto Classification Model
  - CASS Address Component Model
  - Country Aliases
  - D&B Integration
  - Email Component Model
  - External Stored Assets Model
  - GDSN Provider model
  - GDSN Receiver model
  - Google Shopping Model
  - List Processing
  - Matching
  - Matching - Link Golden Record
  - Matching - Merge Golden Record

On the Matching-Merge Golden Record component model (shown below), select the 'Edit' option to add the 'Unmerge From' reference type created in the previous step to the 'Unmerged-From Relation Reference Types' entry.



Matching - Merge Golden Record - Component Model Configuration

Name	Value	Description
Golden Record Object Types	Individual Customer	Object types which can be used as merged golden records in Matching Algorithms
Keep Source Records for Golden Record Object T...		Golden Record Object Types for which Source Records will be stored when importing with Match and Merge Importer. This is an optional setting.
Match Tuning Asset Object Types	MTC-CSV XML File	Asset object types which can be used for storing uploaded data files for Match Tuning Configurations
Source System Object Type	Source_System	Object type which can be used as source system for merged golden records
Deactivated Attribute	DeactivationAttribute	Attribute used for marking a golden record as deactivated. Must be single valued, dimension independent and use LOV with true/false values (either as values or if using ID then as IDs).
Source Record ID Attribute	Source Record ID	Multi-valued attribute used for storing source record IDs of source records on SourceRelations
Source System ID Attribute	Source System ID	Attribute used for storing unique source system ID on Source Systems
Merged-Into Relation Reference Types	InterelationMergeGoldenRecord	Single valued reference types for linking a deactivated golden record to the surviving golden record when merging golden records
Source Relation Reference Type	MergeSourceRelation	Reference type for linking golden records to source system
Unmerged-From Relation Reference Types	Unmerge From	Single valued reference types for linking an unmerged golden record to the golden record unmerged from
Edit		

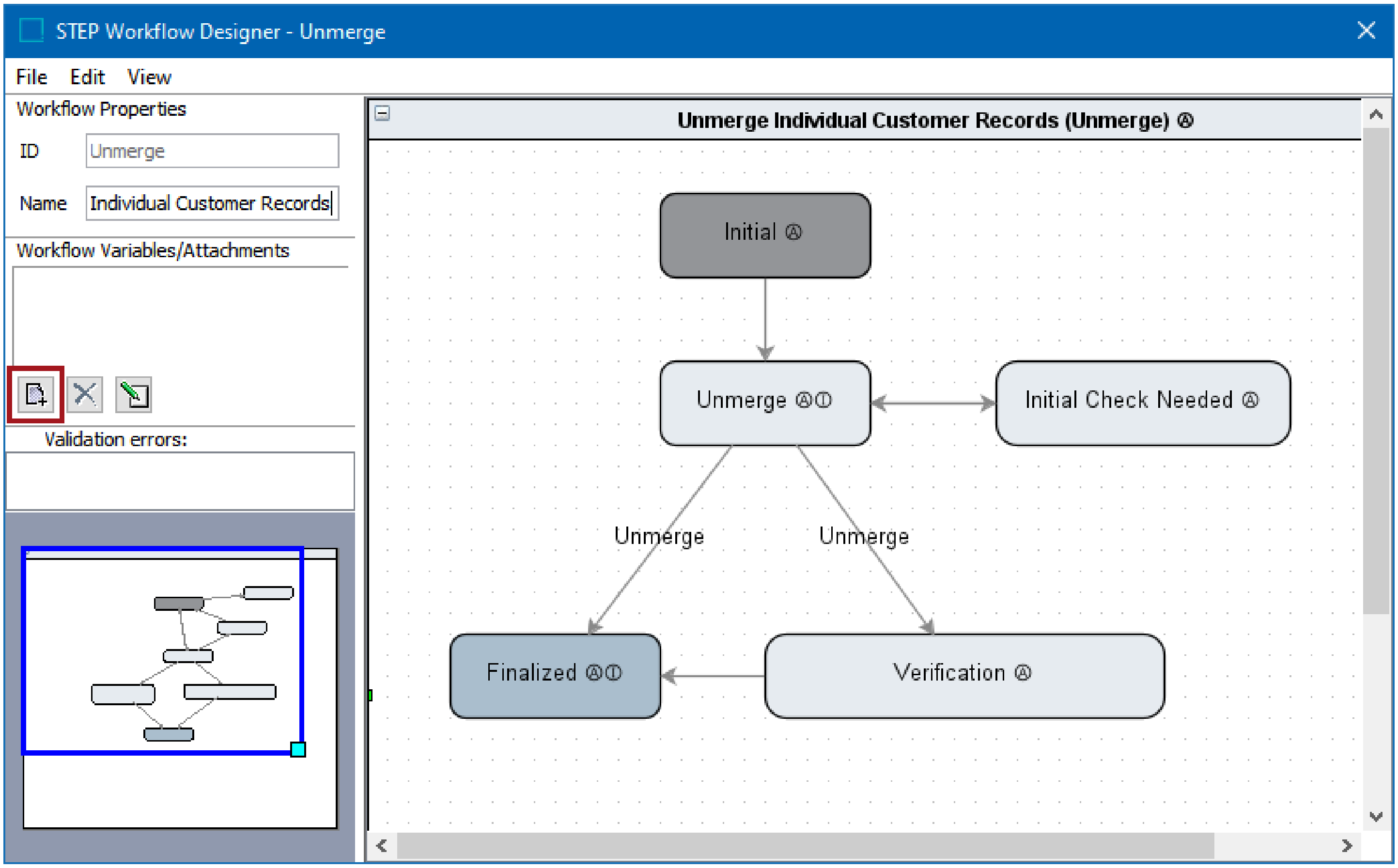
Note: The unmerge reference type's source and targets must match the accepted Golden Record Object Types that are set on this component model. Any other options will cause an error.

Configuring the Unmerge Workflow

The unmerge workflow provides a collaborative process for all unmerge operations between users. The workflow allows users to support the overall process such as preparing some data in the source systems before unmerging and validating data in the downstream systems after the unmerge. This workflow setup is optional since users can perform an ad-hoc unmerge operation as desired.

Note: The following unmerge workflow is an example of a complex unmerge workflow. The only requirements for an unmerge workflow is the matching algorithm ID and the object type validity.

1. Under System Setup, create a new workflow for unmerge. For information on setting up a new workflow, see the **Creating a Workflow** topic in the **Workflows** documentation.
2. On the created workflow, create a new state called 'Unmerge.' Ensure that the initial and final states are added to the workflow as well.
3. In the ‘Workflow Variables/Attachments’ click ‘Add Workflow Variable,’ and set the ID to ‘MatchingAlgorithmID.’ Click OK to close the dialog. For the variable to show up when doing the next step, click 'Save' in the 'File' menu of the workflow designer.


STEP Workflow Designer - Unmerge

File Edit View

Workflow Properties

ID: Unmerge

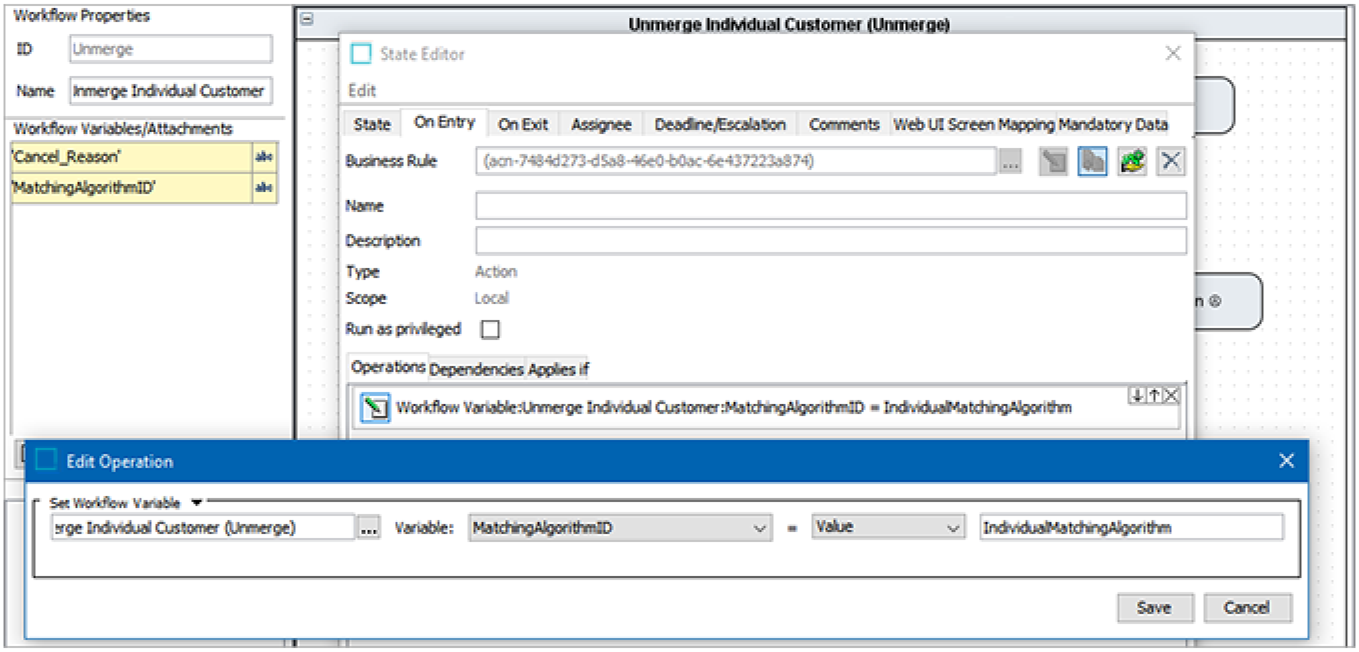
Name: Individual Customer Records

Workflow Variables/Attachments

Validation errors:

Unmerge Individual Customer Records (Unmerge) Ⓐ

4. On the unmerge state, add a new business action under the OnEntry tab. Select 'Set Workflow Variable' as the operation. Select the workflow and the variable MatchingAlgorithmID. The value should be set to the ID of the matching algorithm. For more information, see the **Workflow Variables** topic in the **Workflows** documentation.



Workflow Properties

ID: Unmerge

Name: Inmerge Individual Customer

Workflow Variables/Attachments

- 'Cancel_Reason'
- 'MatchingAlgorithmID'

Unmerge Individual Customer (Unmerge)

State Editor

Edit

State | On Entry | On Exit | Assignee | Deadline/Escalation | Comments | Web UI Screen Mapping | Mandatory Data

Business Rule: (acn-7484d273-d5a8-46e0-b0ac-6e437223a874)

Name:

Description:

Type: Action

Scope: Local

Run as privileged ☐

Operations | Dependencies | Applies if

Workflow Variable:Unmerge Individual Customer:MatchingAlgorithmID = IndividualMatchingAlgorithm

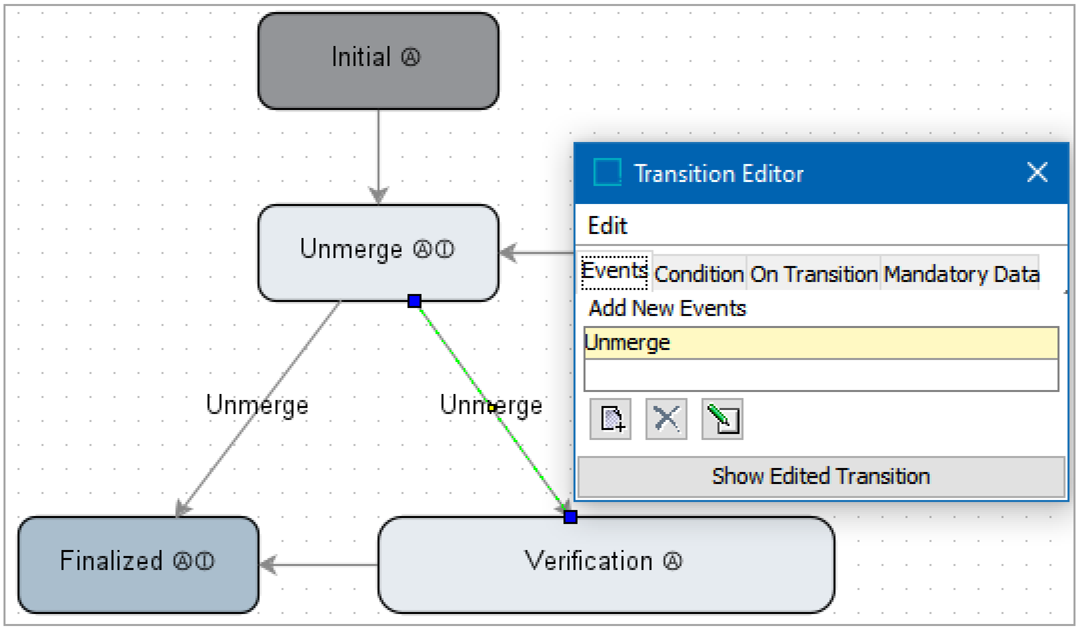
Edit Operation

Set Workflow Variable

erge Individual Customer (Unmerge) ... Variable: MatchingAlgorithmID = Value IndividualMatchingAlgorithm

5. For a complex workflow, if there is more than one transition out of the unmerge state, and not all of these transitions are valid for a completed unmerge operation, then it is recommended that users edit the valid transitions and add events that are specifically named 'Unmerge' to ensure the data flow is clearly understood.

> **Note:** If there are one or more transitions with events named 'Unmerge,' only these transitions will be used when completing the unmerge operation.



Transition Editor

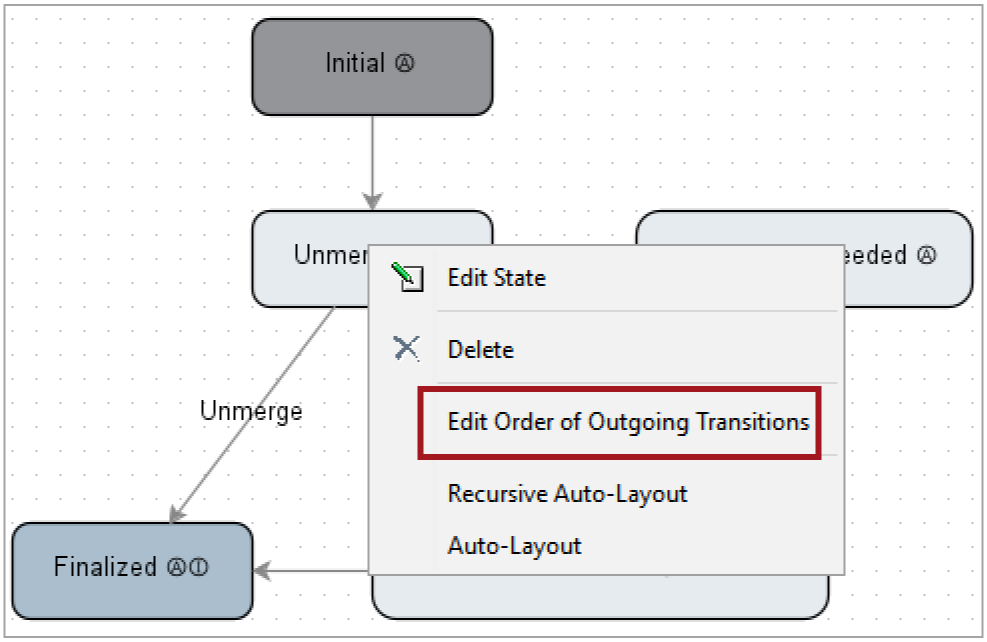
Edit

Events | Condition | On Transition | Mandatory Data

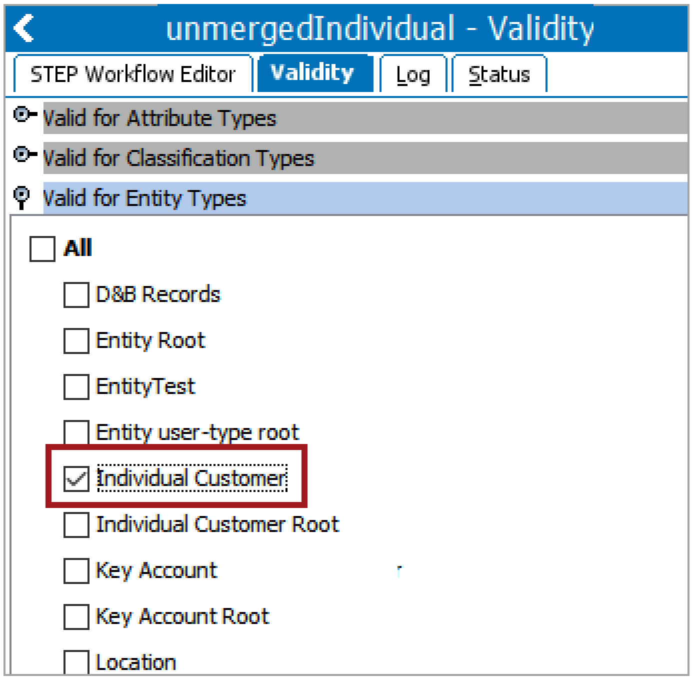
Add New Events

Unmerge

6. When finished configuring the workflow, the order that the transitions are evaluated may be set on the 'Unmerge' state by right-clicking on the state, then selecting 'Edit Order of Outgoing Transitions.'



7. Finally, ensure that the validity for the workflow is set to the unmerge object type on the Validity tab on the workflow in workbench.



unmergedIndividual - Validity

STEP Workflow Editor | Validity | Log | Status

Valid for Attribute Types

Valid for Classification Types

Valid for Entity Types

- ☐ **All**
  - ☐ D&B Records
  - ☐ Entity Root
  - ☐ EntityTest
  - ☐ Entity user-type root
  - ☑ Individual Customer
  - ☐ Individual Customer Root
  - ☐ Key Account
  - ☐ Key Account Root
  - ☐ Location

Configuring Unmerge in Web UI

Prior to configuring unmerge, users must ensure that a match and merge solution is properly configured on their system. The unmerge process relies on existing matching algorithms as well as configured golden record object types. For more information, see the **Configuring Match and Merge** topic in this documentation.

To set up unmerge for match and merge, users will need to consult the Configuring Unmerge in Workbench topic for the component model and workflow configurations.

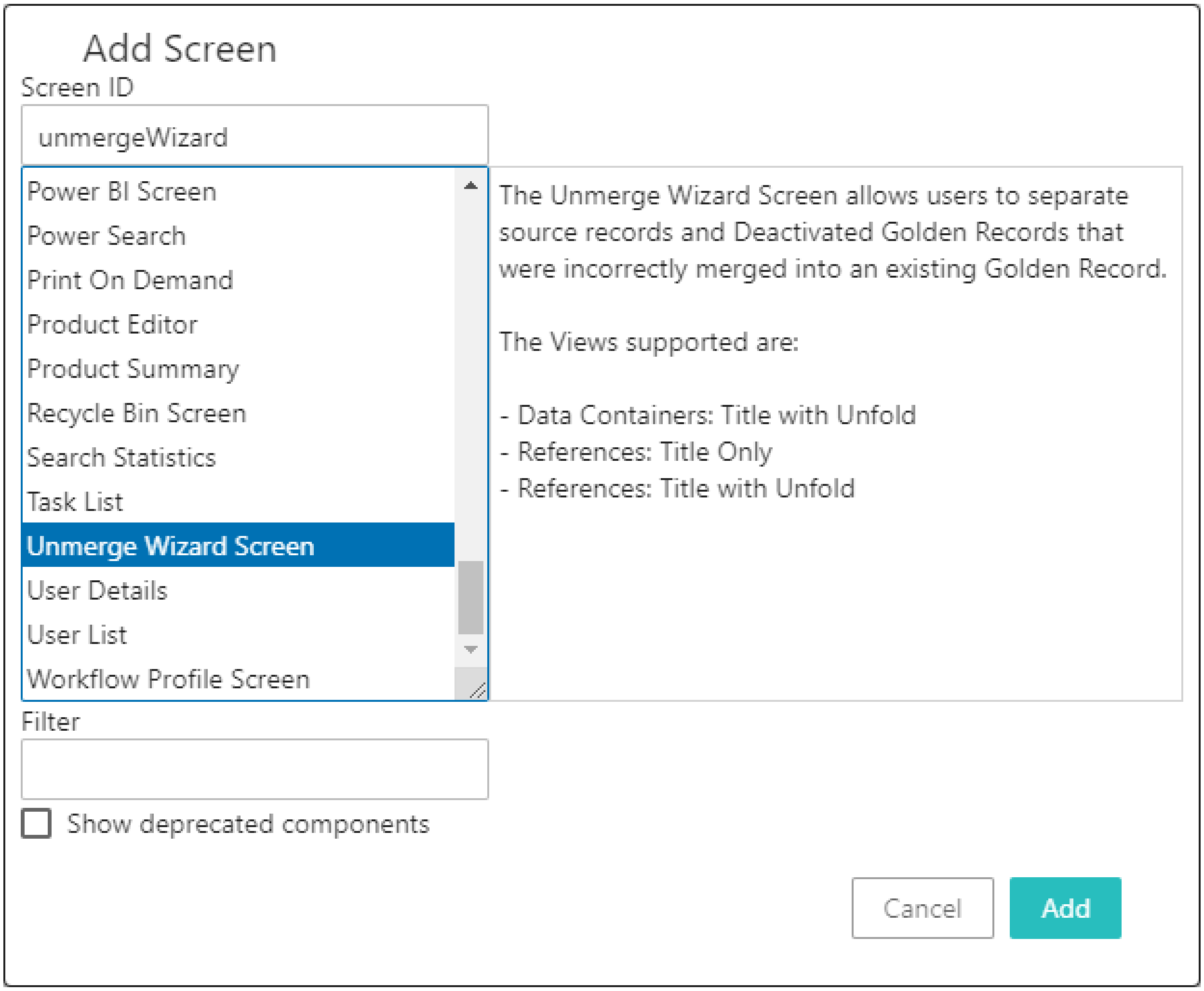
Prerequisites

It is expected that anyone configuring the Unmerge Wizard Screen component is familiar with the Web UI designer as basic concepts for working with the designer are not covered in this section. In addition, the user must have appropriate privileges to access the designer. Additional information can be found in the **Designer Access** section of the **Web User Interfaces / Web UI Getting Started** documentation.

The Unmerge Wizard Screen is a screen type that uses built-in logic to unmerge matched golden records. For information on setting up unmerge workflows, which the Web UI screens require to perform unmerge, see the **Configuring Unmerge for Match and Merge** topic in this documentation.

Adding a New Unmerge Wizard Screen

Steps for creating a new screen are outlined in the **Creating a New Screen** section of the **Design Mode Basics** topic. Select 'Unmerge Wizard Screen' as the screen type (shown in the screenshot below).



Add Screen

Screen ID

unmergeWizard

- Power BI Screen
- Power Search
- Print On Demand
- Product Editor
- Product Summary
- Recycle Bin Screen
- Search Statistics
- Task List
- Unmerge Wizard Screen
- User Details
- User List
- Workflow Profile Screen

The Unmerge Wizard Screen allows users to separate source records and Deactivated Golden Records that were incorrectly merged into an existing Golden Record.

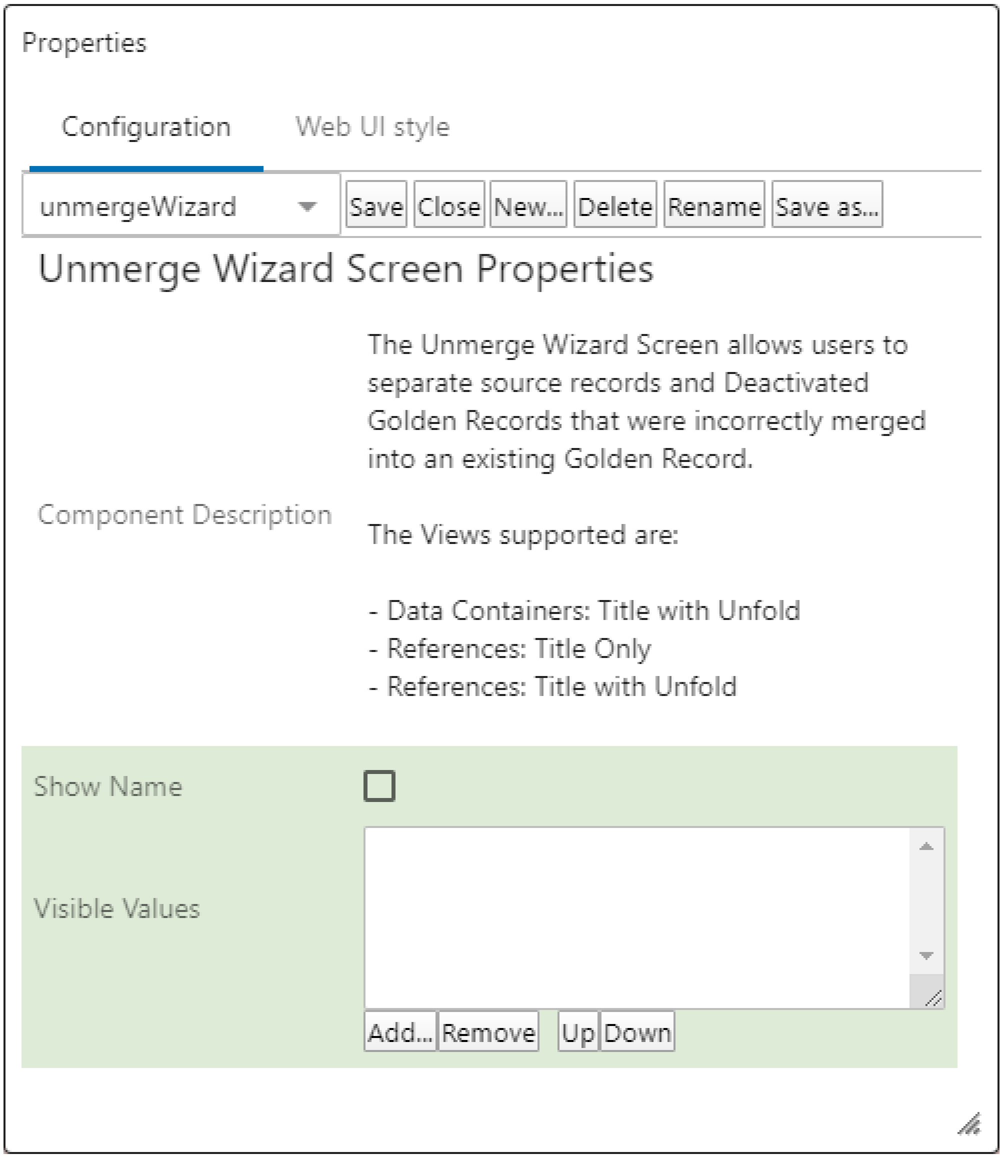
The Views supported are:

- Data Containers: Title with Unfold
- References: Title Only
- References: Title with Unfold

Filter

□ Show deprecated components

Once created, the Unmerge Wizard Screen will need to be configured.



Unmerge Wizard Screen Properties

Component Description: The Unmerge Wizard Screen allows users to separate source records and Deactivated Golden Records that were incorrectly merged into an existing Golden Record.

The Views supported are:

- Data Containers: Title with Unfold
- References: Title Only
- References: Title with Unfold

Show Name: □

Visible Values

1. The 'Show Name' parameter allows users to determine whether the 'Name' of the node displays. Check this box to enable.
2. The 'Visible Values' parameter refers to the values that will be displayed during the unmerged process. Values can be added in this field using the 'Add' button. The order in which these values appear in the field determines the ordering of those values in the Web UI. The ordering may be changed by selecting the relevant value and using the 'Up' and 'Down' buttons.

   The following types may be added:

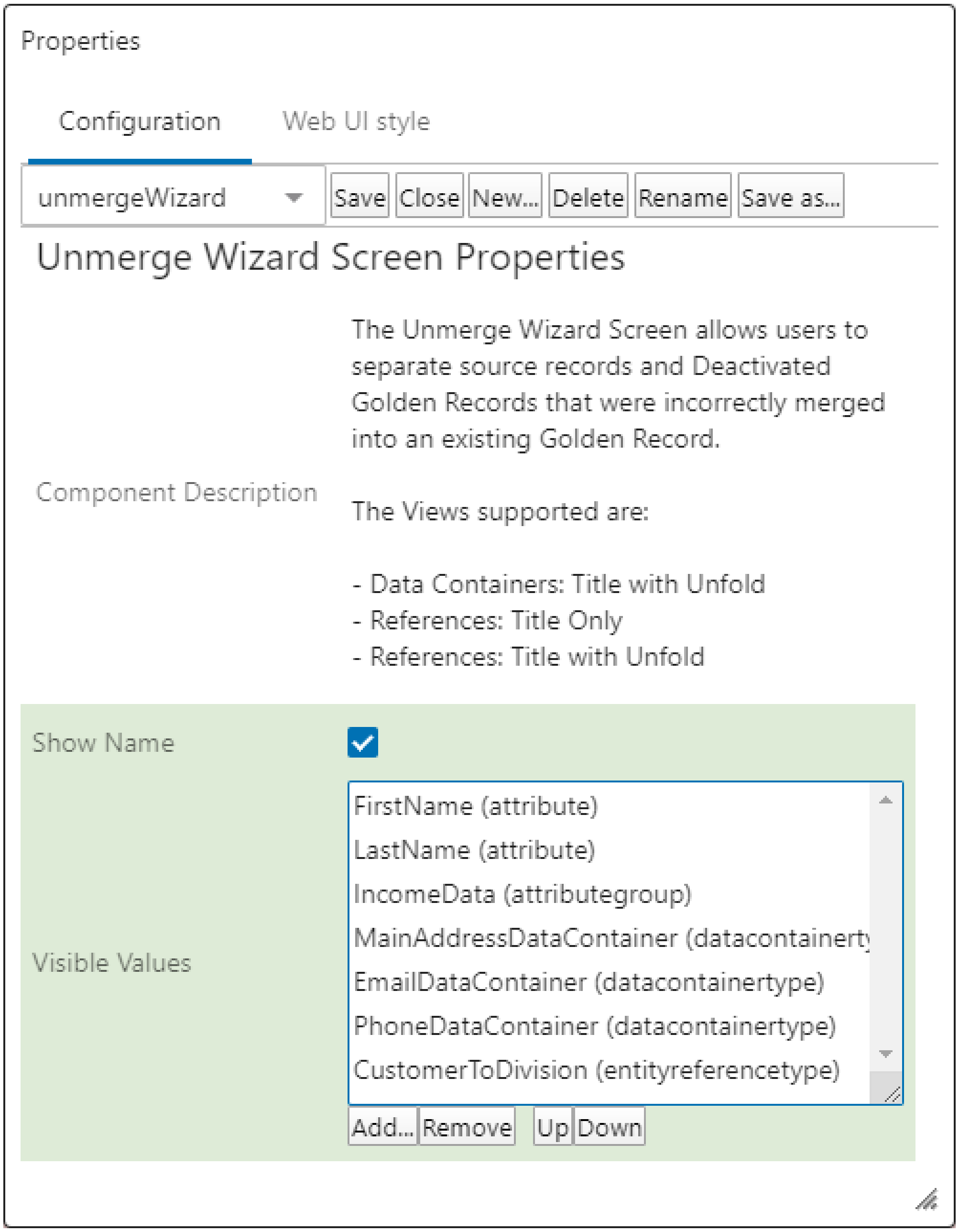
   - **Attributes** will show the selected attributes in the Unmerge Wizard Screen.
   - **Attribute Groups** can be added to show attributes in the group including attributes in nested attribute groups.

Note: Data containers and references that are part of the attribute groups are not supported by this selection and must be configured separately.

- **Data Containers** may be added to the Unmerge Wizard Screen. To add these data containers, navigate to [MAIN] -> [Global Representation List]. If no configuration is found, all attributes and references valid for the data container will be shown.
- **References** also may be added to the Unmerge Wizard Screen. Like the data containers, references are configured in [MAIN] -> [Global Representation List]. If no configuration is found, all attributes valid for the reference will be shown.

Note: All valid data types will be part of the unmerge operation, no matter if they are configured to be visible or not. The attributes for the Visible Values are the values that are displayed on the Unmerge Wizard Screen when unmerging.

The following is a completed Unmerge Wizard Screen configuration.



Unmerge Wizard Screen Properties

Component Description: The Unmerge Wizard Screen allows users to separate source records and Deactivated Golden Records that were incorrectly merged into an existing Golden Record.

The Views supported are:

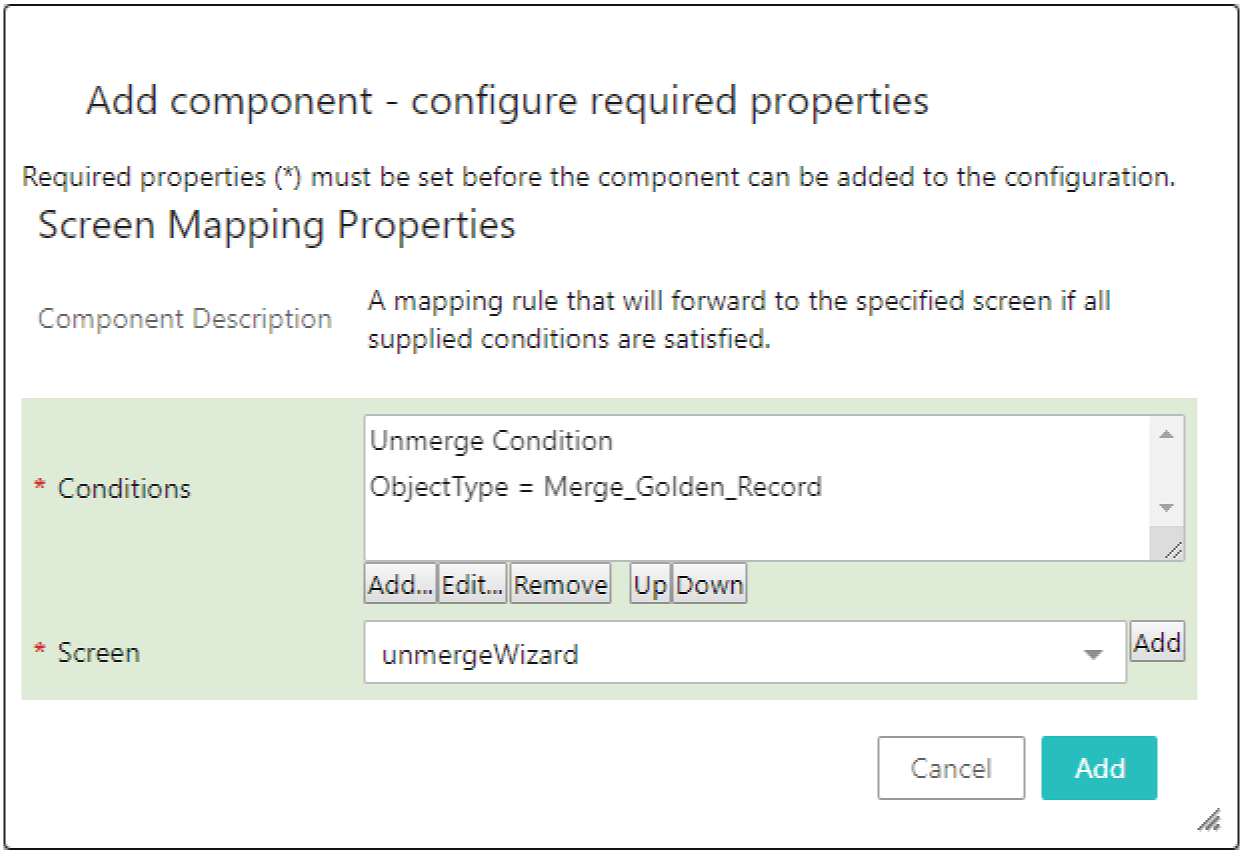
- Data Containers: Title with Unfold
- References: Title Only
- References: Title with Unfold

Show Name: ☑

Visible Values:
FirstName (attribute)
LastName (attribute)
IncomeData (attributegroup)
MainAddressDataContainer (datacontainerty
EmailDataContainer (datacontainertype)
PhoneDataContainer (datacontainertype)
CustomerToDivision (entityreferencetype)

Adding an Unmerge Screen Mapping

Ensure that the newly created Unmerge Wizard Screen is mapped in the Main properties. An example of the mapping is shown below, which uses both the Unmerge Condition with an Object Type condition. For more on mapping screens, see the **Mappings** topic in **Main Properties Overview** section in the **Web User Interfaces / Web UI Setup and User Guide** documentation.



Add component - configure required properties

Required properties (*) must be set before the component can be added to the configuration.

Screen Mapping Properties

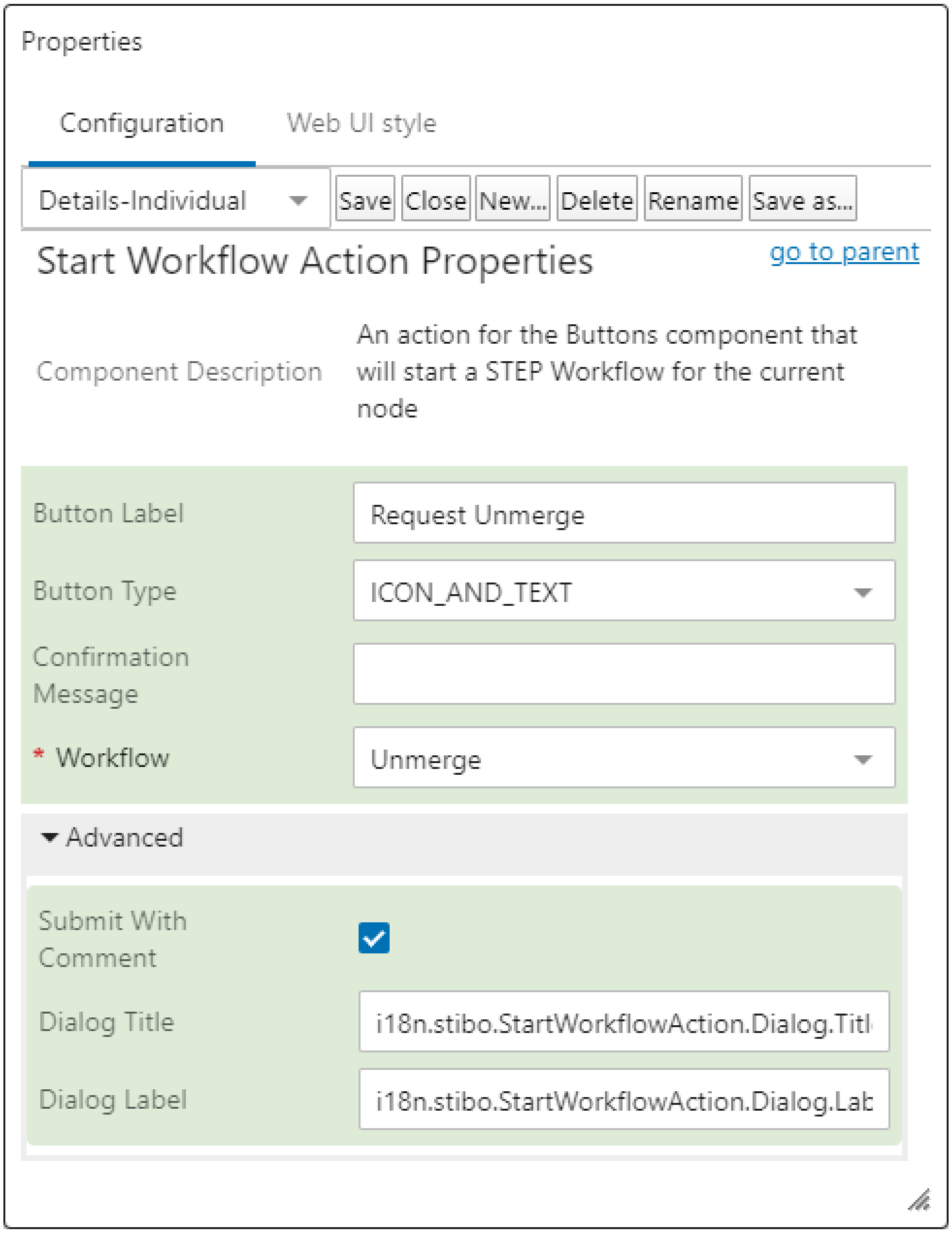
Component Description: A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

* Conditions: Unmerge Condition
ObjectType = Merge_Golden_Record

* Screen: unmergeWizard

Unmerge Buttons on Entity Details Screens

If an unmerge workflow was created and data stewards need to add a selected record to that workflow, add the 'Start Workflow Action' and select the created unmerge workflow. In this example, the button label is ‘Request Unmerge.’ For more information on unmerge workflows, see the **Configuring Unmerge in Workbench** topic in this documentation.



Details-Individual

Start Workflow Action Properties

Component Description	An action for the Buttons component that will start a STEP Workflow for the current node
Button Label	Request Unmerge
Button Type	ICON_AND_TEXT
Confirmation Message	
* Workflow	Unmerge

Advanced

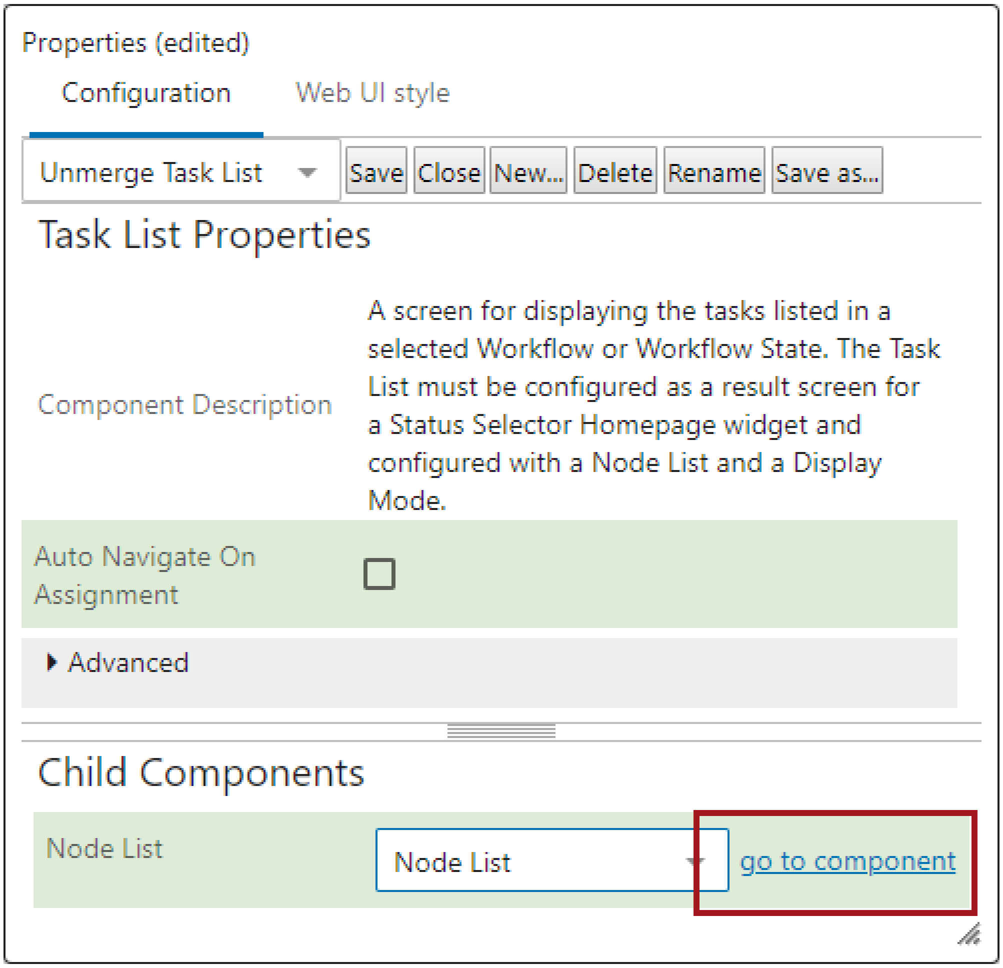
Submit With Comment	☑
Dialog Title	i18n.stibo.StartWorkflowAction.Dialog.Titl
Dialog Label	i18n.stibo.StartWorkflowAction.Dialog.Lat

Under the Advanced flipper, users may enable ‘Submit with Comment.’ This option provides a field that allows users from other workflow states to provide information about the records to aid in unmerging. The comment enters the workflow as a process note, and this note, along with any other process notes is shown to the data steward on the unmerge screen.

Configuring Web UI for Unmerge Workflow

When an unmerge workflow is created, some configurations need to be made to make use of it in Web UI. Create a 'Task List' screen for the unmerge action or edit an existing Task List screen. Steps for creating a new screen are outlined in the **Creating a New Screen** section of the **Design Mode Basics** topic.

On the Task List screen, select the 'go to the component' option on the Node List child component.



Properties (edited)

Task List Properties

Component Description: A screen for displaying the tasks listed in a selected Workflow or Workflow State. The Task List must be configured as a result screen for a Status Selector Homepage widget and configured with a Node List and a Display Mode.

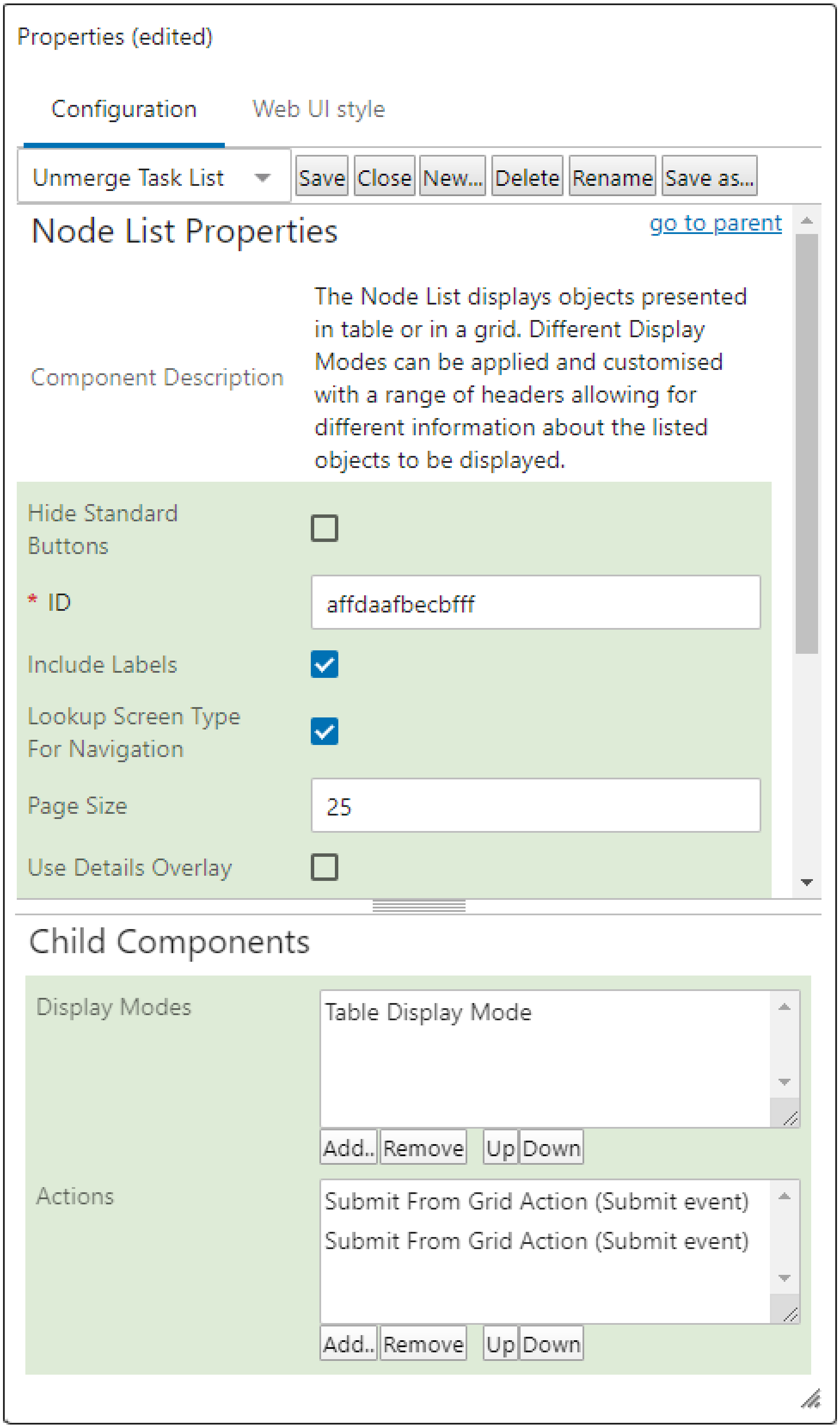
Auto Navigate On Assignment ☐

Child Components

Node List: Node List

In the Node List properties, add a ‘Table Display Mode’ and include the attributes to be shown in the Task List. In the 'Actions' field, add the following:

- **Submit From Grid Action**: This button moves an entity to the next step in the workflow.
- **Delete From Grid Action**: This button takes an entity out of the workflow.



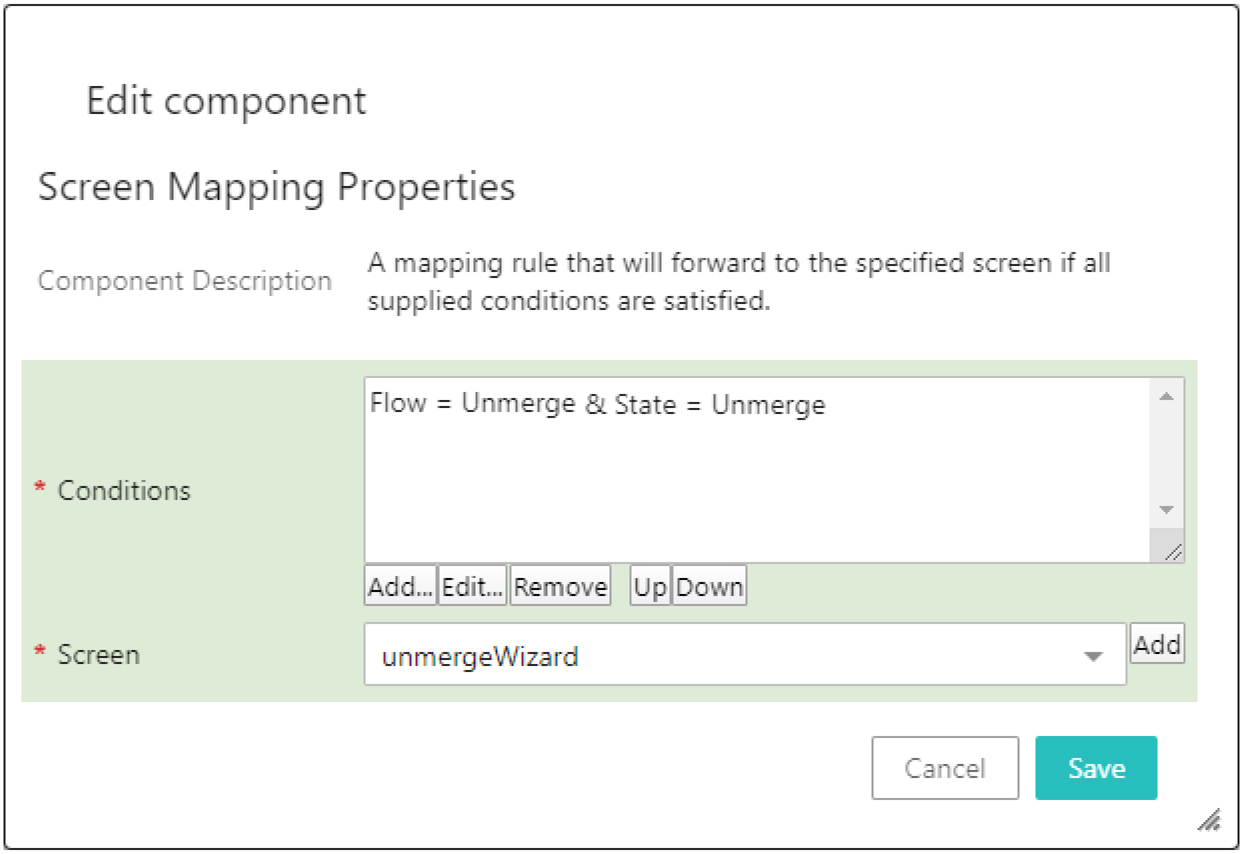
Node List Properties

Component Description	The Node List displays objects presented in table or in a grid. Different Display Modes can be applied and customised with a range of headers allowing for different information about the listed objects to be displayed.
Hide Standard Buttons	☐
* ID	affdaafbecbfff
Include Labels	☑
Lookup Screen Type For Navigation	☑
Page Size	25
Use Details Overlay	☐

Child Components

Display Modes	Table Display Mode
Actions	Submit From Grid Action (Submit event) Submit From Grid Action (Submit event)

To finish the workflow setup, map the unmerge workflow state to the Unmerge Wizard Screen. The workflow and state should be related to the unmerge action. Save the mapping when configured. Below is an example of a completed mapping:



Edit component

Screen Mapping Properties

Component Description: A mapping rule that will forward to the specified screen if all supplied conditions are satisfied.

* Conditions: Flow = Unmerge & State = Unmerge

* Screen: unmergeWizard

Creating an Unmerge Workflow Widget

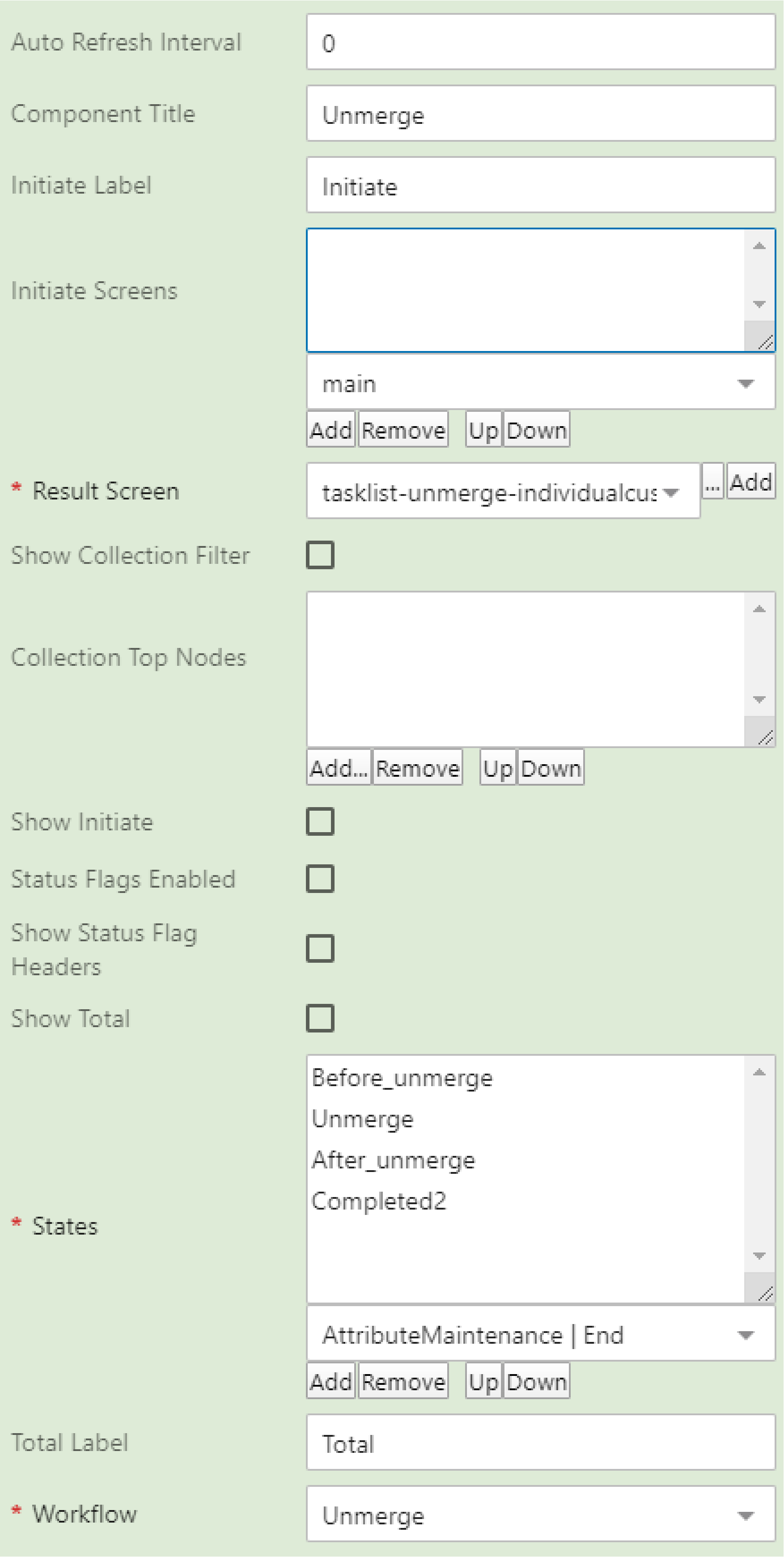
Homepage widgets allow users to execute tasks on a range of features from a single screen with each widget representing a different function or functions. The unmerge process supports the ability to access its functionality via a homepage widget as well. To do this, first add a Status Selector Homepage Widget and then configure it for the unmerge workflow. The steps required to enable this feature are listed below:

For more information on widgets, see the **Homepage Widgets** topic in the Web **User Interfaces / Web UI Getting Started** documentation.

The following fields on the Status Selector Homepage Widget Properties should be populated:

- **Result Screen**: the Unmerge Tasklist as created in this topic
- **States**: add the states as configured in the unmerge workflow
- **Workflow**: the created unmerge workflow

The following is an example of a configured Status Selector Homepage Widget.



Auto Refresh Interval	0
Component Title	Unmerge
Initiate Label	Initiate
Initiate Screens	main
* Result Screen	tasklist-unmerge-individualcus
Show Collection Filter	□
Collection Top Nodes	
Show Initiate	□
Status Flags Enabled	□
Show Status Flag Headers	□
Show Total	□
* States	Before_unmerge Unmerge After_unmerge Completed2 AttributeMaintenance | End
Total Label	Total
* Workflow	Unmerge

Unmerge - Distribute Source Records

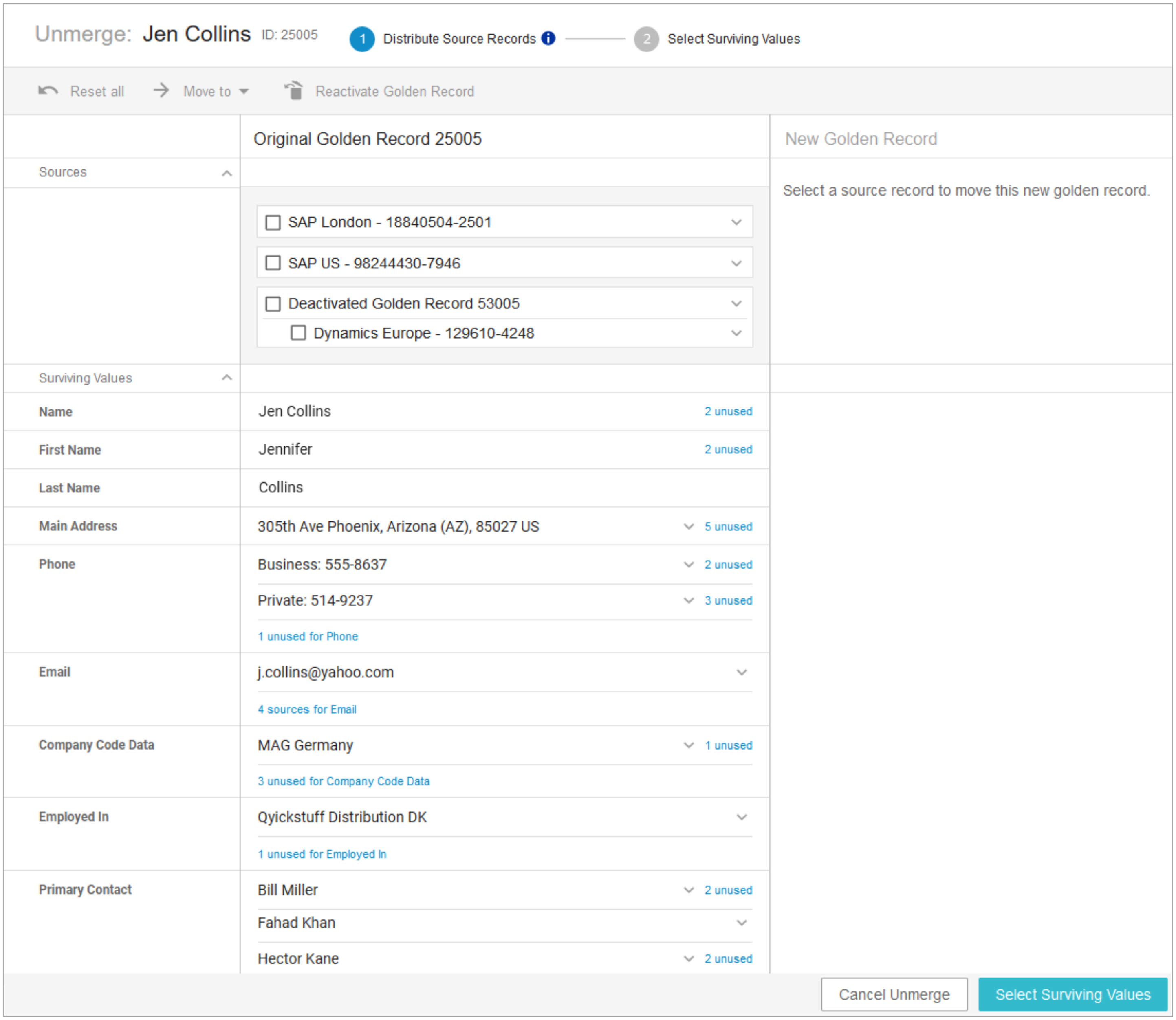
This topic explains the Distribute Source Records setup of the Unmerge Wizard. This step must be performed before moving to the Select Survivorship Values step.

Prerequisites

This topic assumes that users have already set up unmerge functionality, both in Web UI as well as in the workbench. For more information about the setup, see the **Configuring Unmerge in Workbench** topic and the **Adding a New Unmerge Wizard Screen** section of the **Configuring Unmerge in Web UI** topic in this documentation.

Using the Unmerge Functionality

In Web UI, navigate to an applicable entity node to be unmerged, and click the ‘Unmerge’ button. The Unmerge Wizard Screen will display.



Unmerge: Jen Collins ID: 25005

1 Distribute Source Records — 2 Select Surviving Values

Original Golden Record 25005

New Golden Record

Sources

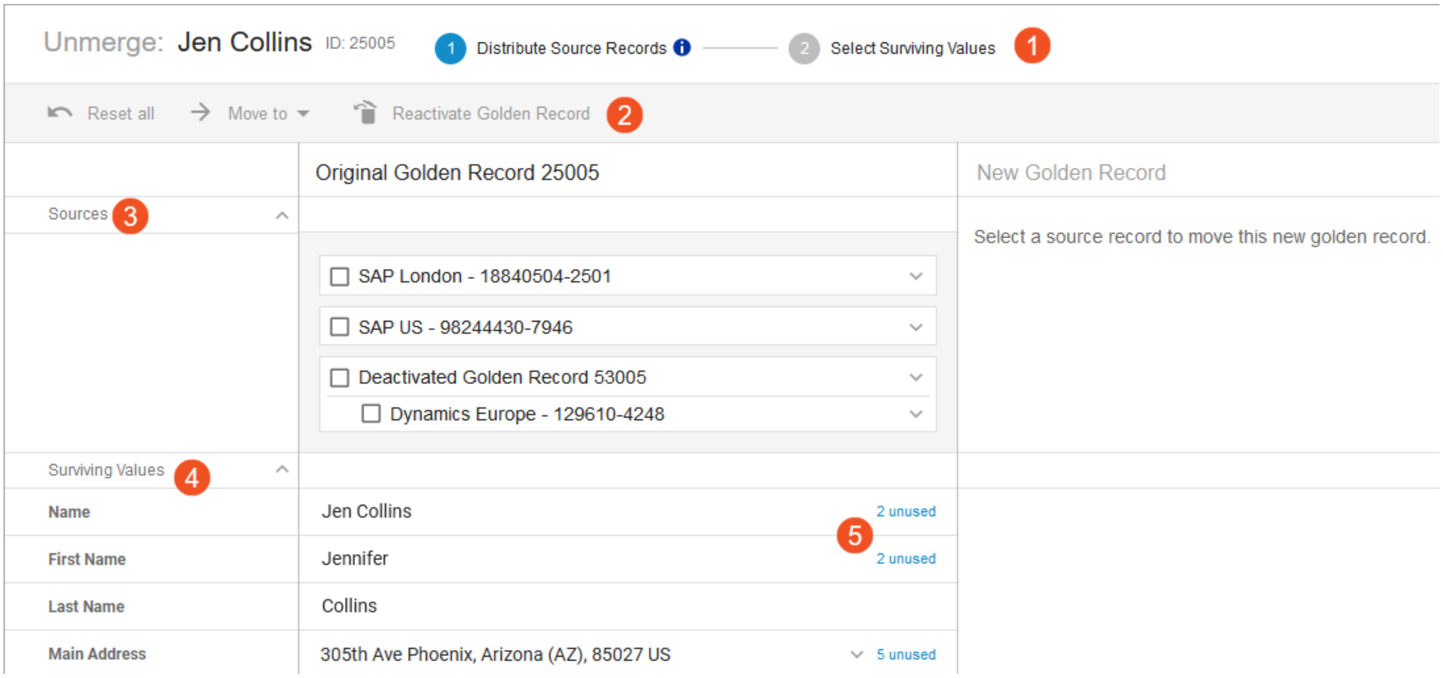
Select a source record to move this new golden record.

- □ SAP London - 18840504-2501
- □ SAP US - 98244430-7946
- □ Deactivated Golden Record 53005
  - □ Dynamics Europe - 129610-4248

Surviving Values

Name	Jen Collins	2 unused
First Name	Jennifer	2 unused
Last Name	Collins	
Main Address	305th Ave Phoenix, Arizona (AZ), 85027 US	5 unused
Phone	Business: 555-8637	2 unused
	Private: 514-9237	3 unused
	1 unused for Phone	
Email	j.collins@yahoo.com	
	4 sources for Email	
Company Code Data	MAG Germany	1 unused
	3 unused for Company Code Data	
Employed In	Qyickstuff Distribution DK	
	1 unused for Employed In	
Primary Contact	Bill Miller	2 unused
	Fahad Khan	
	Hector Kane	2 unused

The unmerge screen comprises the following elements.


Unmerge: Jen Collins ID: 25005
1 Distribute Source Records
2 Select Surviving Values
1
Reset all
Move to
Reactivate Golden Record
2
Sources
3
Surviving Values
4
5

Original Golden Record 25005

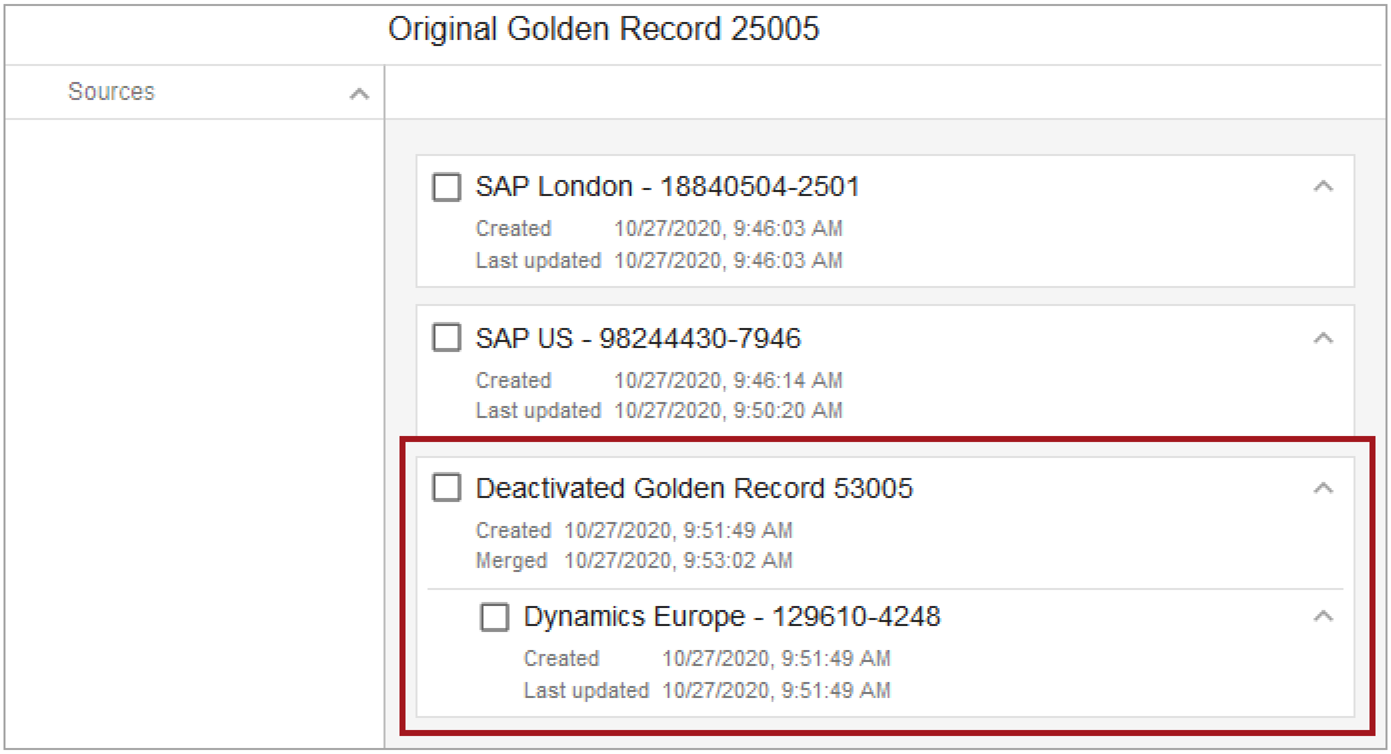
New Golden Record

Select a source record to move this new golden record.

- ☐ SAP London - 18840504-2501
- ☐ SAP US - 98244430-7946
- ☐ Deactivated Golden Record 53005
  - ☐ Dynamics Europe - 129610-4248

Surviving Values	Original Golden Record 25005	
Name	Jen Collins	2 unused
First Name	Jennifer	2 unused
Last Name	Collins	
Main Address	305th Ave Phoenix, Arizona (AZ), 85027 US	5 unused

1. **Unmerge Steps -** This component shows where in the Unmerge process the user is currently.
2. **Action Bar** contains three elements:
   - Reset all: The action will revert the screen back to the original state. All actions / changes that were made will be lost.
   - Move to: When one or more source records are selected, this action will move them to another golden record or a new golden record can be created.
   - Reactivate Golden Record: When a golden record has been merged into another golden record, this action allows users to reactivate it. A deactivated golden record must be selected to perform this action, and any source records associated with this record will be moved as well.
3. **Sources -** This element shows the records that will be used for determining the original record. This section shows all deactivated golden records that were merged into the golden record being unmerged. Once a record is expanded, the created date and the merged date or last updated date are shown.



Original Golden Record 25005

Sources

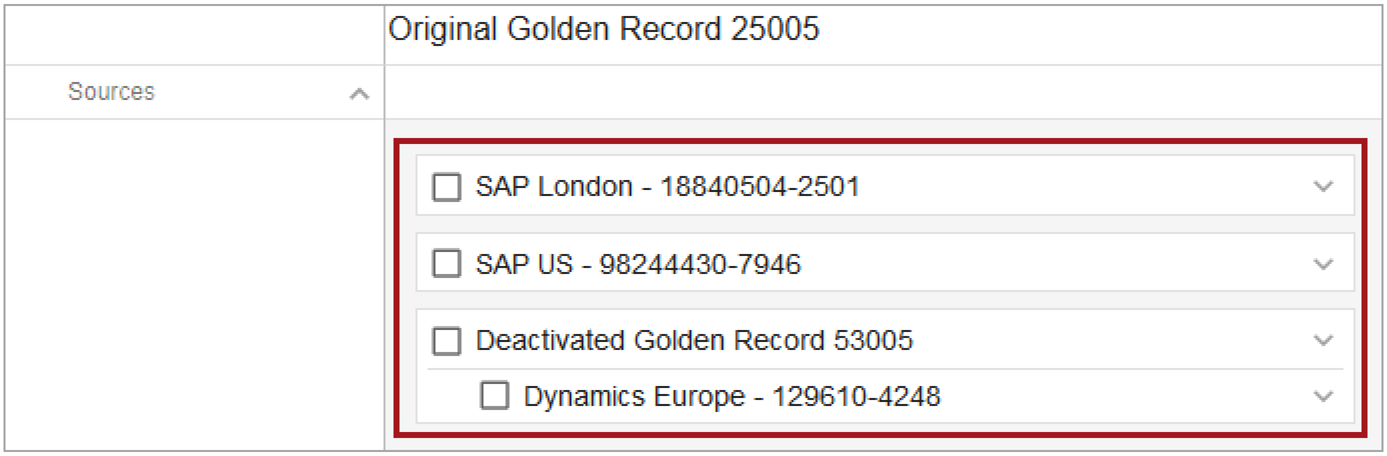
□ SAP London - 18840504-2501
Created 10/27/2020, 9:46:03 AM
Last updated 10/27/2020, 9:46:03 AM

□ SAP US - 98244430-7946
Created 10/27/2020, 9:46:14 AM
Last updated 10/27/2020, 9:50:20 AM

□ Deactivated Golden Record 53005
Created 10/27/2020, 9:51:49 AM
Merged 10/27/2020, 9:53:02 AM

□ Dynamics Europe - 129610-4248
Created 10/27/2020, 9:51:49 AM
Last updated 10/27/2020, 9:51:49 AM

If the system uses source record IDs when importing, those source records are shown in combination with the deactivated golden records. All source records shown are actively ‘assigned’ to the golden record being unmerged, but those that were previously actively assigned to the deactivated golden records are shown underneath them. This representation shows that the values might move along if the deactivated golden records are to be reactivated. When these records are expanded, the creation data as well as the last updated date will be shown.



Original Golden Record 25005

Sources

□ SAP London - 18840504-2501

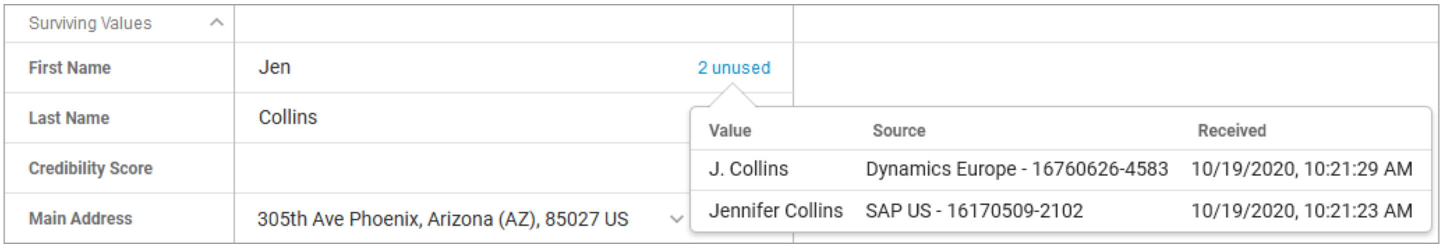
□ SAP US - 98244430-7946

□ Deactivated Golden Record 53005

□ Dynamics Europe - 129610-4248

4. **Surviving Values -** This element shows the values that were configured to display and provides a preview of the surviving data after a completed unmerge operation.

5. **Unused Values -** The unused values are calculated based on data from deactivated golden records and Source data. Since the data on the deactivated Golden Record could be a mix from several sources and often will be identical to the surviving value it's important to make sure source data storage is enabled to get the complete overview of unused values. For more information, see the **Storing Source Data for Golden Records** topic in this documentation.



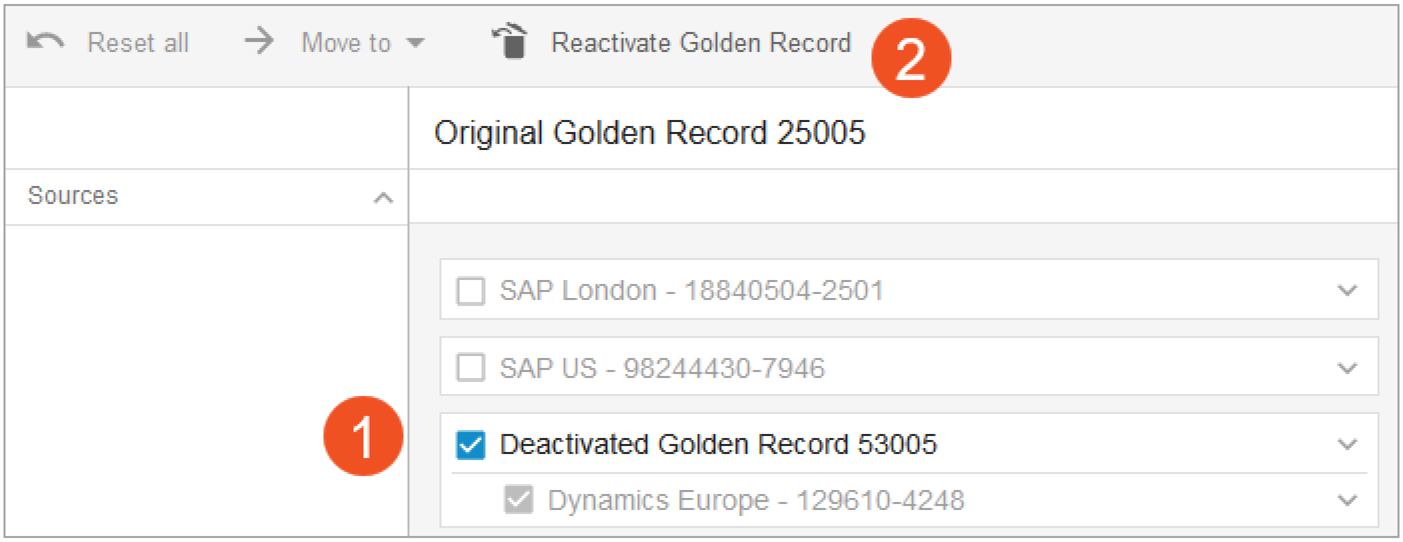
Surviving Values		
First Name	Jen	2 unused
Last Name	Collins	
Credibility Score		
Main Address	305th Ave Phoenix, Arizona (AZ), 85027 US	

Value	Source	Received
J. Collins	Dynamics Europe - 16760626-4583	10/19/2020, 10:21:29 AM
Jennifer Collins	SAP US - 16170509-2102	10/19/2020, 10:21:23 AM

- The blue text link on the right side next to the surviving values shows how many unselected attribute values that did not survive are associated with the record. Clicking on this text will show the values, and for each record, each belongs to a popup dialog with the value, source, and timestamp.
- The blue link under the surviving values will show the unused instances of references and data containers. References are grouped by the reference target and data containers are grouped by the defined data container key.
- If no key is defined, the link will just show the available sources in an ungrouped list.

Reactivating a Deactivated Golden Record

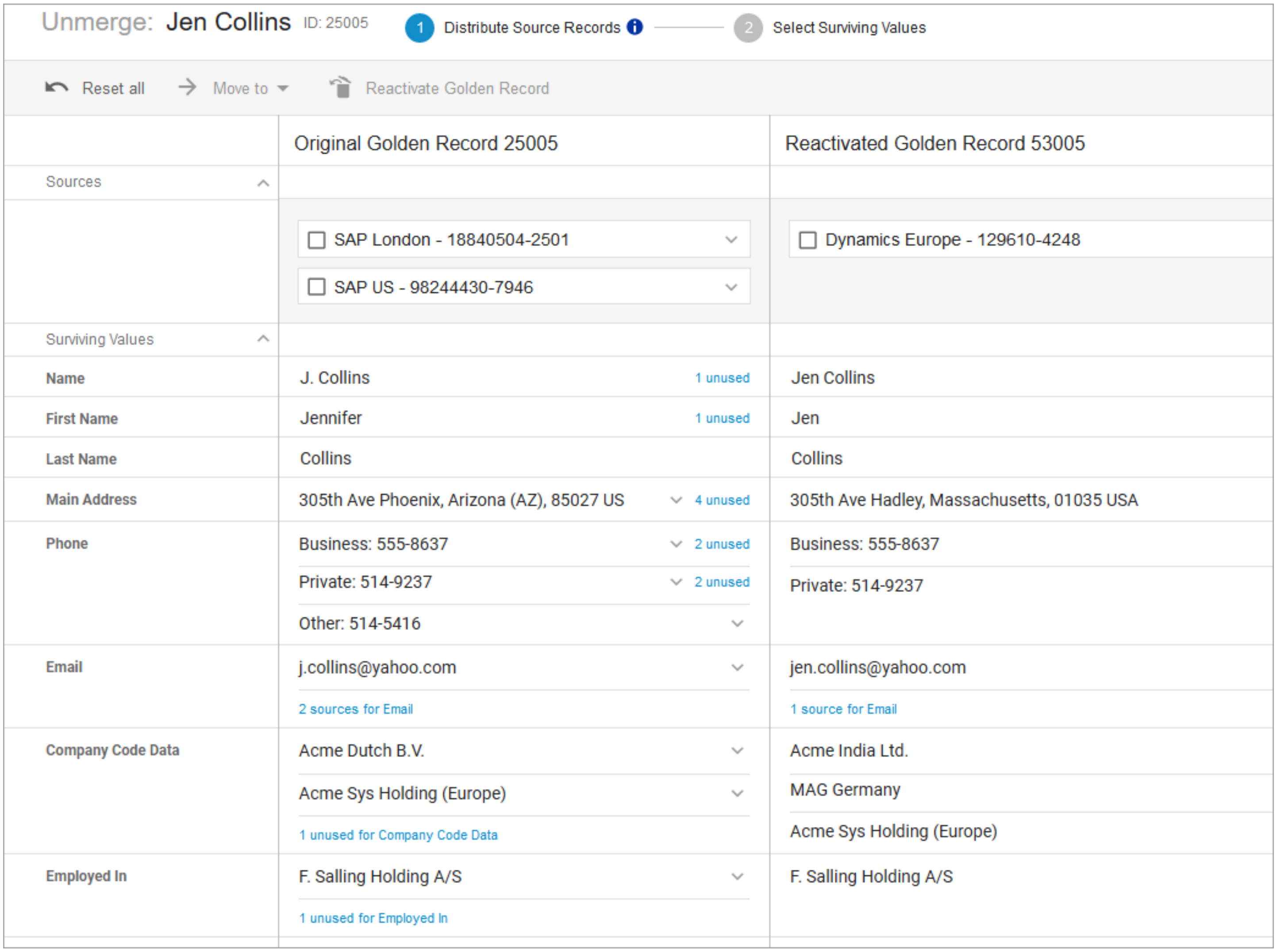
To reactivate a golden record, select the desired deactivated record, and then, select the 'Reactivate Golden Record' toolbar action.



Original Golden Record 25005

- □ SAP London - 18840504-2501
- □ SAP US - 98244430-7946
- ☑ Deactivated Golden Record 53005
  - ☑ Dynamics Europe - 129610-4248

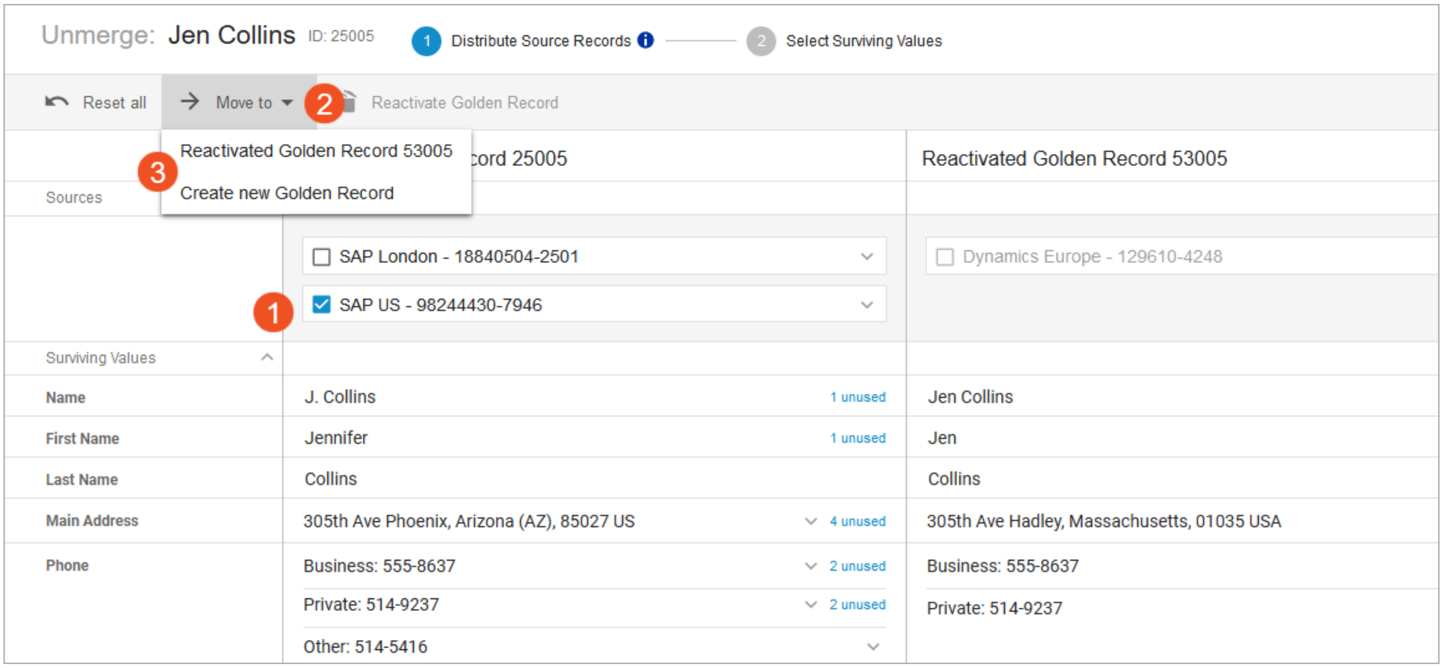
The unmerge logic will calculate the result for both of the selected golden records and show the result.


Unmerge: Jen Collins ID: 25005
1 Distribute Source Records
2 Select Surviving Values
Reset all
Move to
Reactivate Golden Record

	Original Golden Record 25005	Reactivated Golden Record 53005
Sources	☐ SAP London - 18840504-2501 ☐ SAP US - 98244430-7946	☐ Dynamics Europe - 129610-4248
Surviving Values		
Name	J. Collins 1 unused	Jen Collins
First Name	Jennifer 1 unused	Jen
Last Name	Collins	Collins
Main Address	305th Ave Phoenix, Arizona (AZ), 85027 US 4 unused	305th Ave Hadley, Massachusetts, 01035 USA
Phone	Business: 555-8637 2 unused Private: 514-9237 2 unused Other: 514-5416	Business: 555-8637 Private: 514-9237
Email	j.collins@yahoo.com 2 sources for Email	jen.collins@yahoo.com 1 source for Email
Company Code Data	Acme Dutch B.V. Acme Sys Holding (Europe) 1 unused for Company Code Data	Acme India Ltd. MAG Germany Acme Sys Holding (Europe)
Employed In	F. Salling Holding A/S 1 unused for Employed In	F. Salling Holding A/S

Moving a Source Record

To move one or more source records to another golden record or to a new golden record, select the desired source records, and then select the ‘Move To’ action button.



Unmerge: Jen Collins ID: 25005

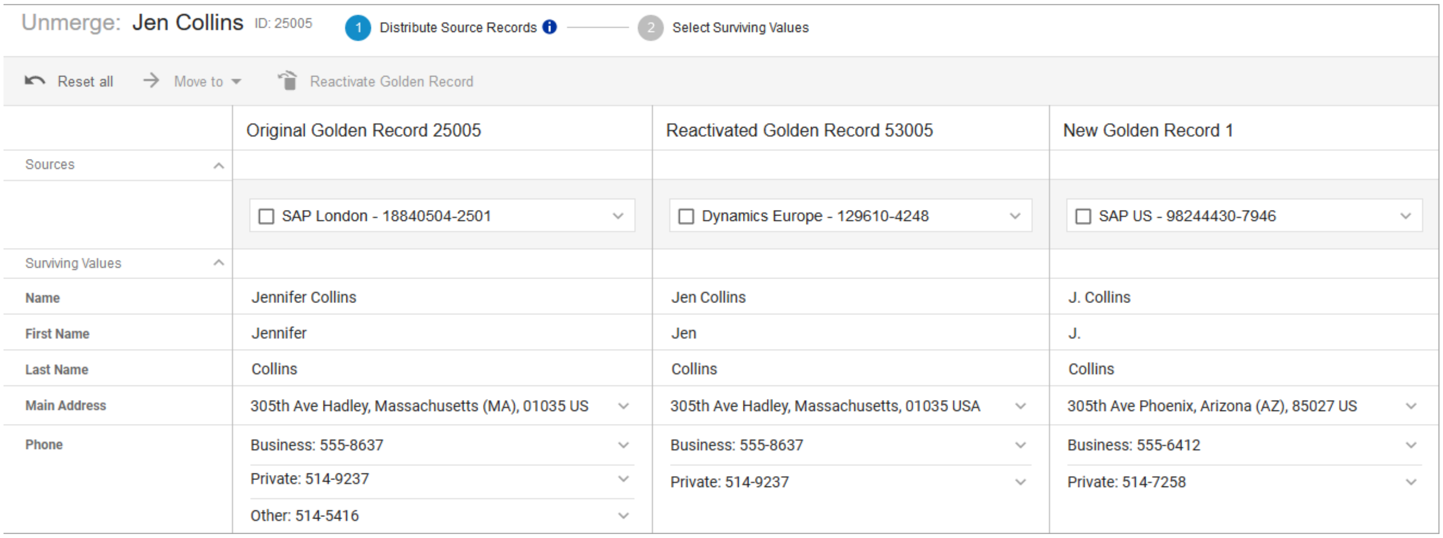
1 Distribute Source Records — 2 Select Surviving Values

Reset all | Move to 2 | Reactivate Golden Record

3 Reactivated Golden Record 53005
Create new Golden Record

	cord 25005	Reactivated Golden Record 53005
Sources		
	☐ SAP London - 18840504-2501 1 ☑ SAP US - 98244430-7946	☐ Dynamics Europe - 129610-4248
Surviving Values		
Name	J. Collins 1 unused	Jen Collins
First Name	Jennifer 1 unused	Jen
Last Name	Collins	Collins
Main Address	305th Ave Phoenix, Arizona (AZ), 85027 US 4 unused	305th Ave Hadley, Massachusetts, 01035 USA
Phone	Business: 555-8637 2 unused Private: 514-9237 2 unused Other: 514-5416	Business: 555-8637 Private: 514-9237

The unmerge logic will calculate the result for all golden records and show the result.



Unmerge: Jen Collins ID: 25005

1 Distribute Source Records — 2 Select Surviving Values

Reset all | Move to | Reactivate Golden Record

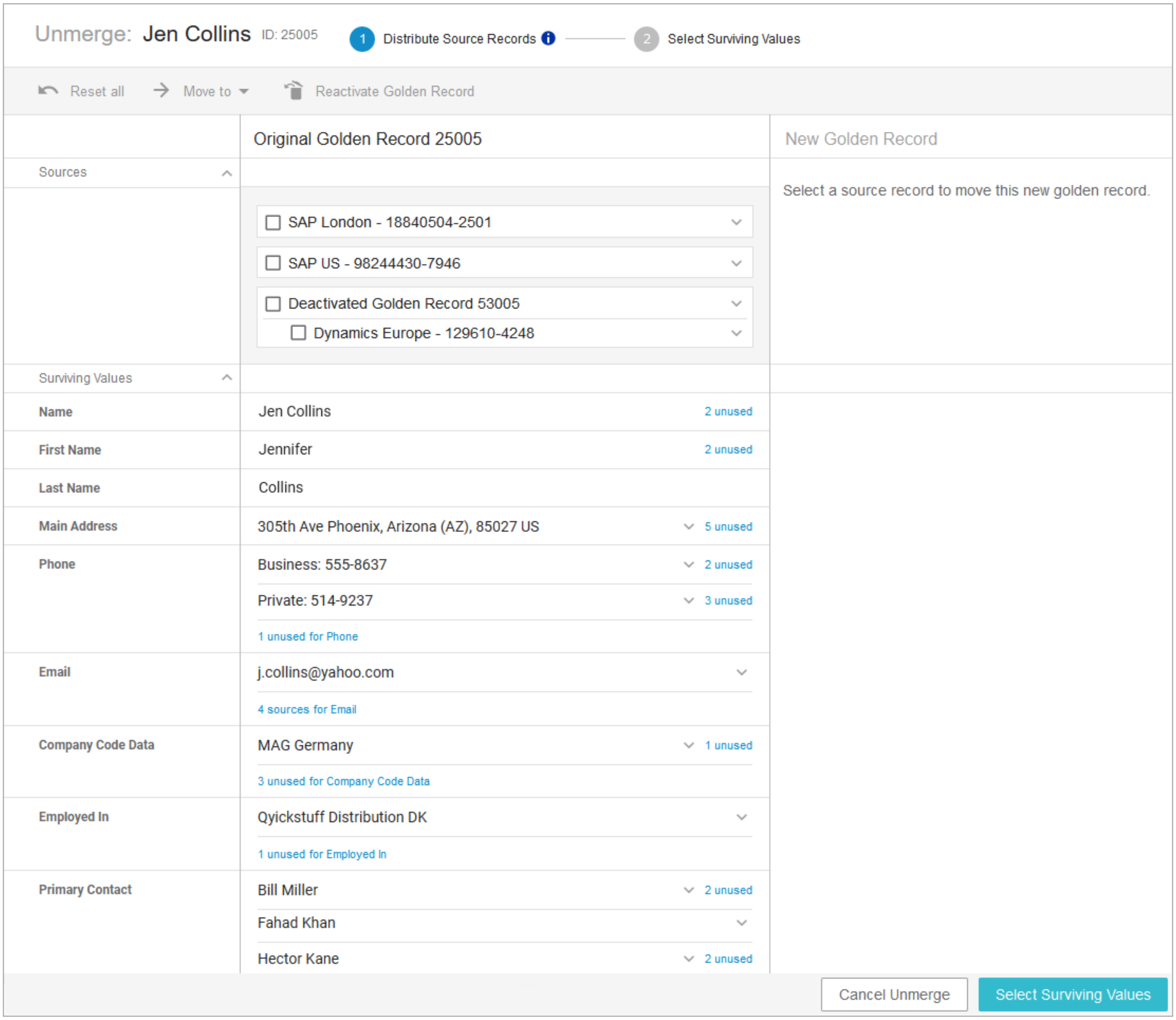
	Original Golden Record 25005	Reactivated Golden Record 53005	New Golden Record 1
Sources			
	☐ SAP London - 18840504-2501	☐ Dynamics Europe - 129610-4248	☐ SAP US - 98244430-7946
Surviving Values			
Name	Jennifer Collins	Jen Collins	J. Collins
First Name	Jennifer	Jen	J.
Last Name	Collins	Collins	Collins
Main Address	305th Ave Hadley, Massachusetts (MA), 01035 US	305th Ave Hadley, Massachusetts, 01035 USA	305th Ave Phoenix, Arizona (AZ), 85027 US
Phone	Business: 555-8637 Private: 514-9237 Other: 514-5416	Business: 555-8637 Private: 514-9237	Business: 555-6412 Private: 514-7258

Unmerge - Selecting Surviving Values

This topic assumes that users have already performed the Distribute Source Records step of the Unmerge Wizard. For more information, see the **Unmerge -Distribute Source Records** topic in this documentation.

Once the new record is created, either from a merged record or a reactivated golden record, select the 'Select Surviving Values' button at the bottom of the Unmerge Wizard screen. The second step of the Unmerge Wizard will display which allows users to select which value survives per record.

The available values in the dropdown controls are calculated based on data from deactivated Golden Records and Source data. Since the data on the deactivated Golden Record could be a mix from several sources and often will be identical to the surviving value it’s important to make sure source data storage is enabled to get the complete overview of possible values. For more information, see the **Storing Source Data for Golden Records** topic in this documentation.



Unmerge: Jen Collins ID: 25005

Original Golden Record 25005

New Golden Record

Select a source record to move this new golden record.

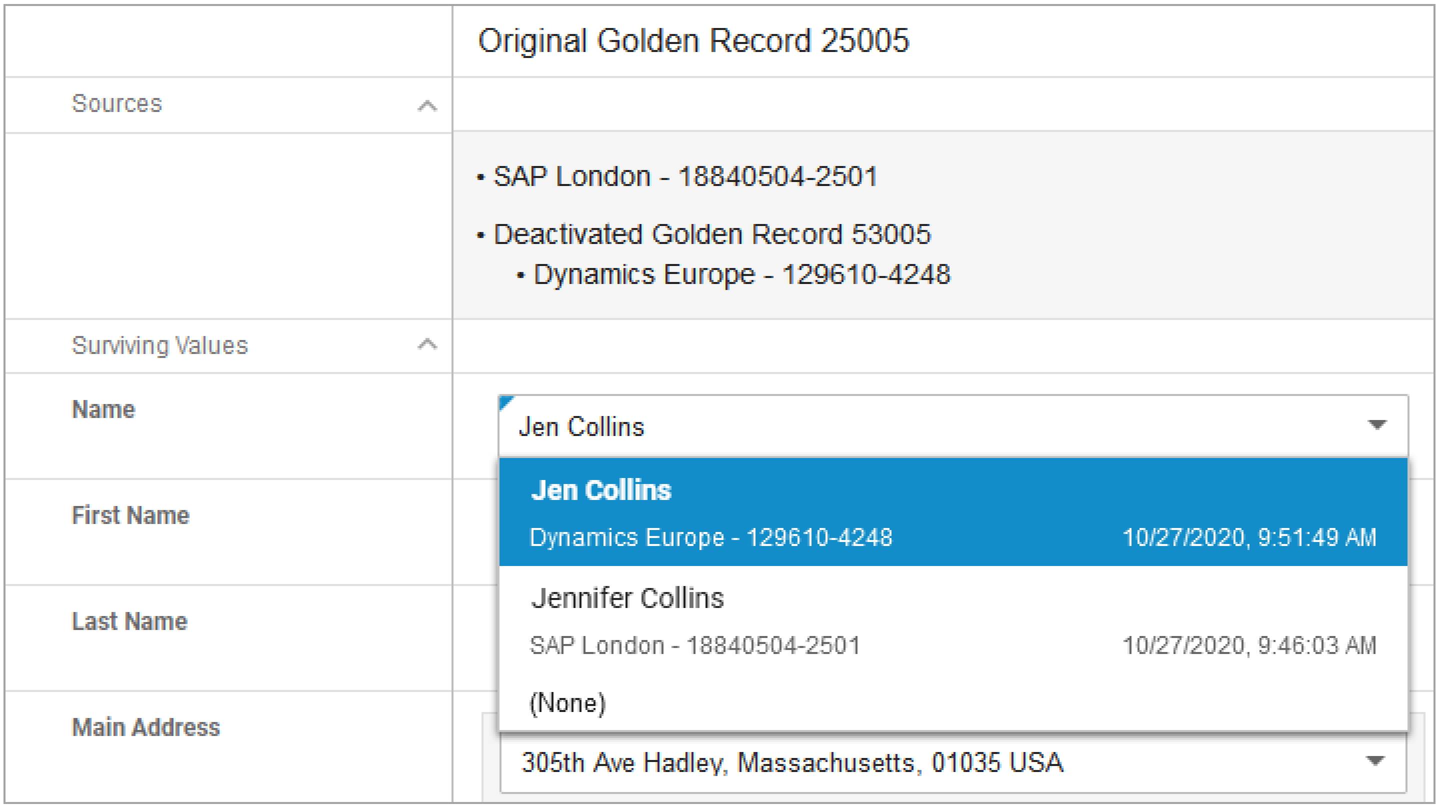
Sources

- □ SAP London - 18840504-2501
- □ SAP US - 98244430-7946
- □ Deactivated Golden Record 53005
  - □ Dynamics Europe - 129610-4248

Surviving Values

Name	Jen Collins	2 unused
First Name	Jennifer	2 unused
Last Name	Collins	
Main Address	305th Ave Phoenix, Arizona (AZ), 85027 US	5 unused
Phone	Business: 555-8637	2 unused
	Private: 514-9237	3 unused
	1 unused for Phone	
Email	j.collins@yahoo.com	
	4 sources for Email	
Company Code Data	MAG Germany	1 unused
	3 unused for Company Code Data	
Employed In	Qyickstuff Distribution DK	
	1 unused for Employed In	
Primary Contact	Bill Miller	2 unused
	Fahad Khan	
	Hector Kane	2 unused

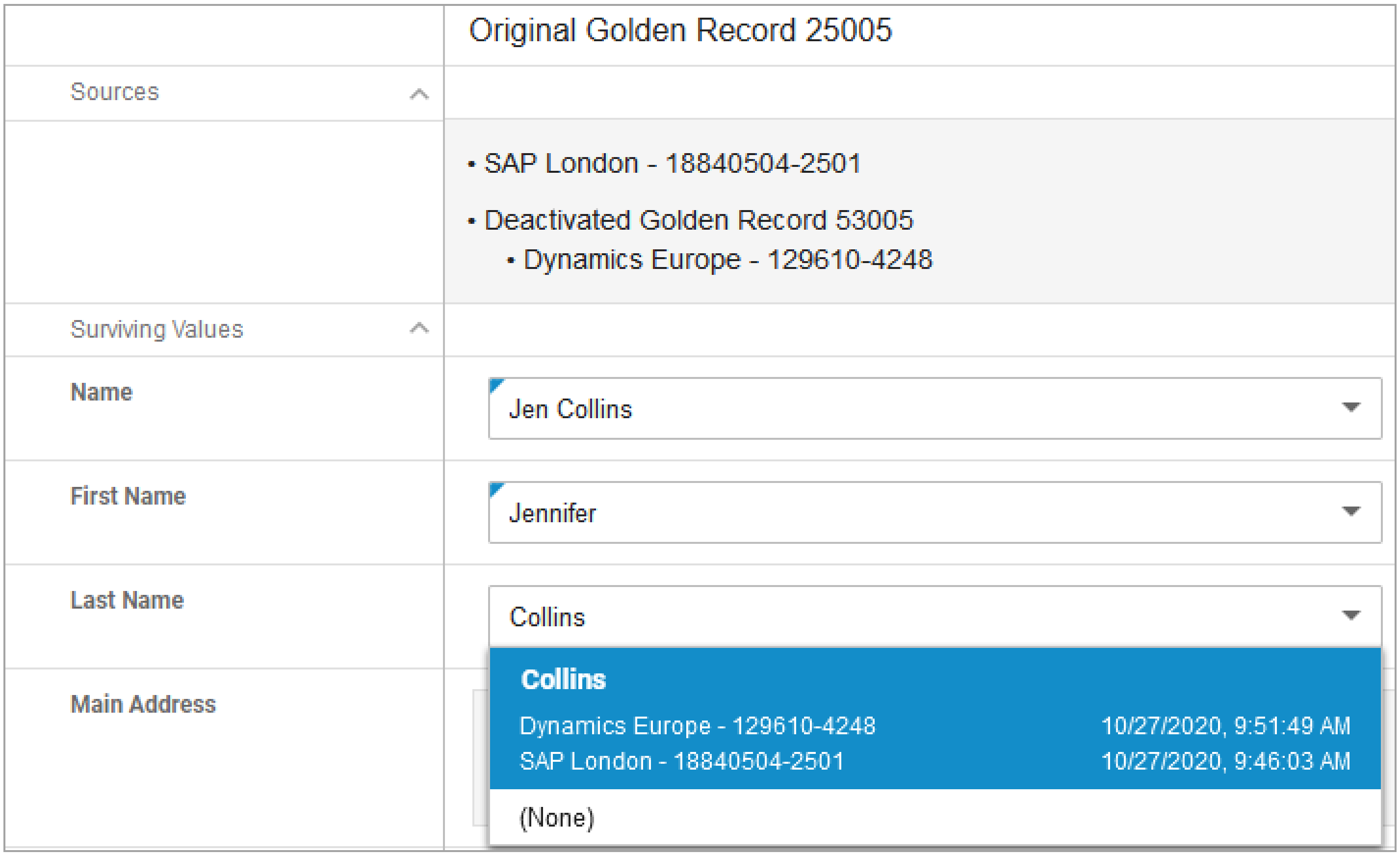
The fields with the marker in the top left denote that multiple values exist for this field.



	Original Golden Record 25005
Sources	
	• SAP London - 18840504-2501 • Deactivated Golden Record 53005 • Dynamics Europe - 129610-4248
Surviving Values	
Name	Jen Collins
First Name	
Last Name	
Main Address	305th Ave Hadley, Massachusetts, 01035 USA

Jen Collins	Dynamics Europe - 129610-4248	10/27/2020, 9:51:49 AM
Jennifer Collins	SAP London - 18840504-2501	10/27/2020, 9:46:03 AM
(None)		

Fields without this marker will only display the value displayed or '(None)' to remove any saved values.

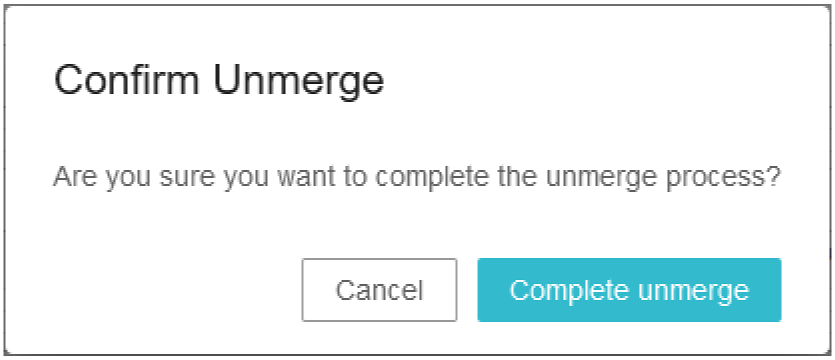


	Original Golden Record 25005
Sources	
	• SAP London - 18840504-2501 • Deactivated Golden Record 53005 • Dynamics Europe - 129610-4248
Surviving Values	
Name	Jen Collins
First Name	Jennifer
Last Name	Collins
Main Address	

Collins

Collins	
Dynamics Europe - 129610-4248	10/27/2020, 9:51:49 AM
SAP London - 18840504-2501	10/27/2020, 9:46:03 AM
(None)	

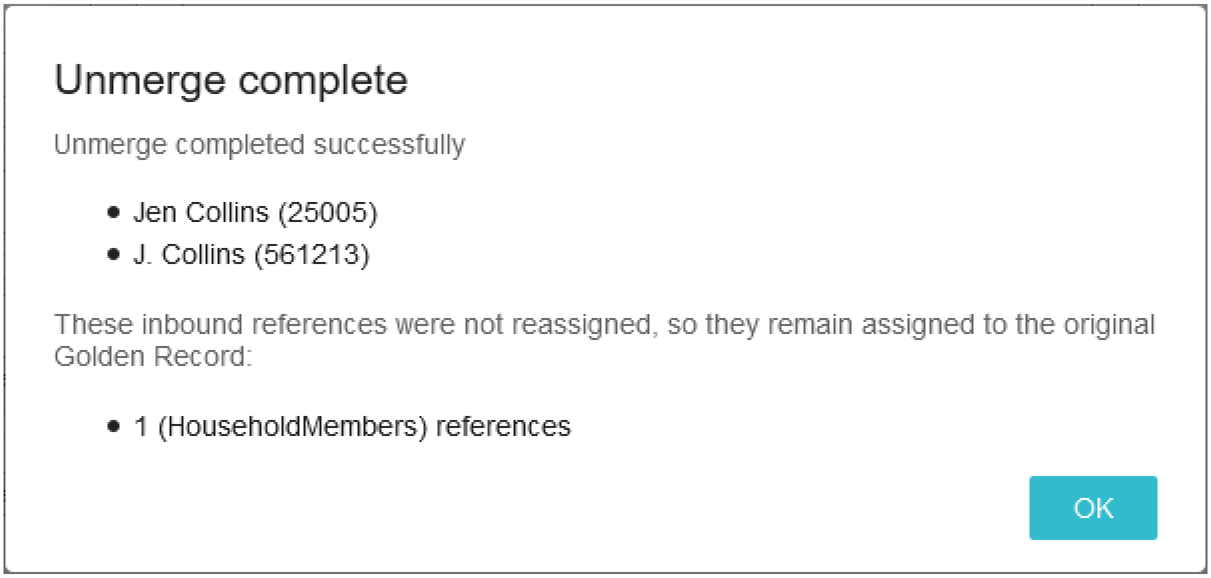
Once the desired values for each record are selected, select the 'Complete Unmerge' button at the bottom of the Ummerge wizard screen. Once selected a confirmation prompt will display. Select 'Complete unmerge' if ready to complete the process.



Confirm Unmerge

Are you sure you want to complete the unmerge process?

Once confirmed and completed, a summary of the Unmerge process will display.



Unmerge complete

Unmerge completed successfully

- Jen Collins (25005)
- J. Collins (561213)

These inbound references were not reassigned, so they remain assigned to the original Golden Record:

- 1 (HouseholdMembers) references

Match and Merge Match Tuning

A match tuning configuration allow data stewards to evaluate and fine-tune a matching algorithm for better accuracy when importing source records. With this tool, users can analyze data and iterate on the matching algorithm before ever running an import.

The first step in tuning the algorithm is to run a STEP data profile on files mapped to the match tuning configuration. You will be looking for data entries that are good for matching your records together, if these values are always populated, and if they need to be normalized before they can be used.

The match tuning configuration uses standard data profiling tools, but because the data being profiled originates from outside the system, some features, such as bulk update, search, and saving collections, are not available.

For more information on data profiles, see the **Data Profiles** topic of the **Data Profiling** documentation. For use case examples, see the **Data Governance** topic in the **Customer MDM Solution Enablement** documentation.

Next, the match tuning configuration can evaluate the matching algorithm via the Evaluate Matching Algorithm action. This action runs two reports–pair export and match codes export–in a background process that can assist the data steward in evaluating the data by showing which match code groups have been created and which source records have been matched together, based on the data profile.

Using these reports, the data steward can continue to iterate on the matching algorithm until it is satisfactory.

List Processing Deduplicate Records

The list processing component allows users to import, export, and manage rented customer records lists. These lists can aid marketing in their initiatives to target specific segments of their customers based on data stored on the lists. With the list processing component, users can import a list, assess the quality of the list, remove unwanted records, and then export the modified lists.

For more information, see the **List Processing** topic in the **Data Preparation** documentation.

Golden Records Survivorship Rules

Survivorship rules determine the outcome of merging two records by declaring which values survive for each attribute, reference, and data container on the golden record. The application of survivorship differs slightly across match actions, but the overall principles remain the same. When merging records, the surviving attribute values are selected by survivorship rules.

When selecting which values survive, the basic strategy is often to either trust some sources more than other sources or to preserve the most recent updates. These kinds of rules are called 'Most Recent' and 'Trusted Source.' More special survivorship rules can be implemented using business actions. It is possible to apply different survivorship rules to groups of attributes or attributes and references individually, so that, for example, the value of one attribute follows a trusted source rule while the other attributes follow a most recently rule.

Survivorship rules are defined independently for an object's name, its references, its data containers, and its attributes / attribute groups.

A set of configurable rules exists, but if special business logic needs to be accounted for, a business action survivorship rule can be implemented to apply surviving values to golden records.

For more information on configuring survivorship rules, see the **Configuration of Survivorship Rules** topic in this documentation.

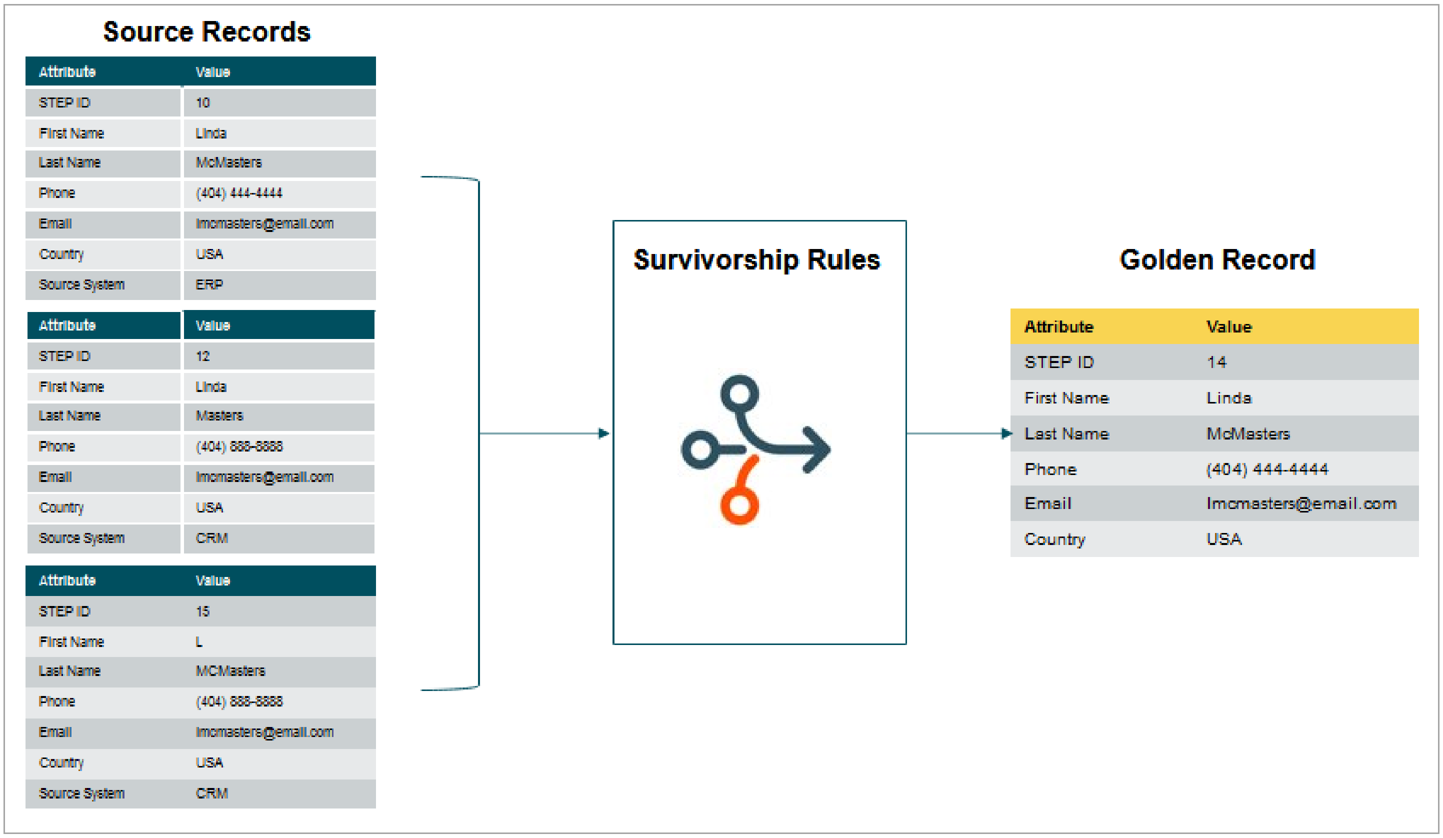
For more information on how survivorship rules work with match and link, see the **Survivorship with Match and Link** topic in this documentation.

For more information on how survivorship rules work with match and merge, see the **Survivorship with Match and Merge** topic in this documentation.

Survivorship in Match and Link

In a match and link solution, source records are products or entities that already exist in STEP. The golden record is a new product or entity, created and populated by the survivorship rules.

When survivorship rules run in a match and link solution, the number of sources is unknown; there could be one or many sources. This lack of information is especially important to remember if writing Business Action survivorship rules.



Source Records

Attribute	Value
STEP ID	10
First Name	Linda
Last Name	McMasters
Phone	(404) 444-4444
Email	lmcmasters@email.com
Country	USA
Source System	ERP

Attribute	Value
STEP ID	12
First Name	Linda
Last Name	Masters
Phone	(404) 888-8888
Email	lmcmasters@email.com
Country	USA
Source System	CRM

Attribute	Value
STEP ID	15
First Name	L
Last Name	MCMasters
Phone	(404) 888-8888
Email	lmcmasters@email.com
Country	USA
Source System	CRM

Survivorship Rules

Golden Record

Attribute	Value
STEP ID	14
First Name	Linda
Last Name	McMasters
Phone	(404) 444-4444
Email	lmcmasters@email.com
Country	USA

Match and link survivorship rules are only ever run in the context of an event processor; they are not used when merging source records.

Golden records should not be merged in a match and link solution. That would conflict with the general rule that the golden record is not to be directly edited.

Trusted Source

To use the trusted source survivorship rule, information about the source, e.g., what system / supplier the object originated from, must be available on the source objects. This attribute is defined in the general Matching component model as the 'Data Source Attribute.' This attribute should typically be a mandatory LOV-based description attribute that does not allow users to add values. For more information, see the **Matching Component Model** section of the **Match Criteria Configuration** topic in this documentation.

Information from a source outside the list of trusted sources is not copied to the golden record during a trusted source survivorship rule evaluation. Information on a record without a source attribute is not copied to the golden record by trusted source survivorship rules.

For more information, see the **Configuration of Survivorship Rules** topic in this documentation.

Most Recent

The 'Most Recent' survivorship rule strategy takes the most recent data from a golden record's source objects.

What is most recent can be qualified either by the revision date in STEP or by a 'Last Edited' date attribute. The latter makes it possible to promote data based on the time of edit in source systems.

For more information, see the **Configuration of Survivorship Rules** topic in this documentation.

Business Action Rule

It is quite common that special rules for survivorship exist in implementations. In these cases, it is necessary to implement that survivorship logic in business actions, which can be run as survivorship rules.

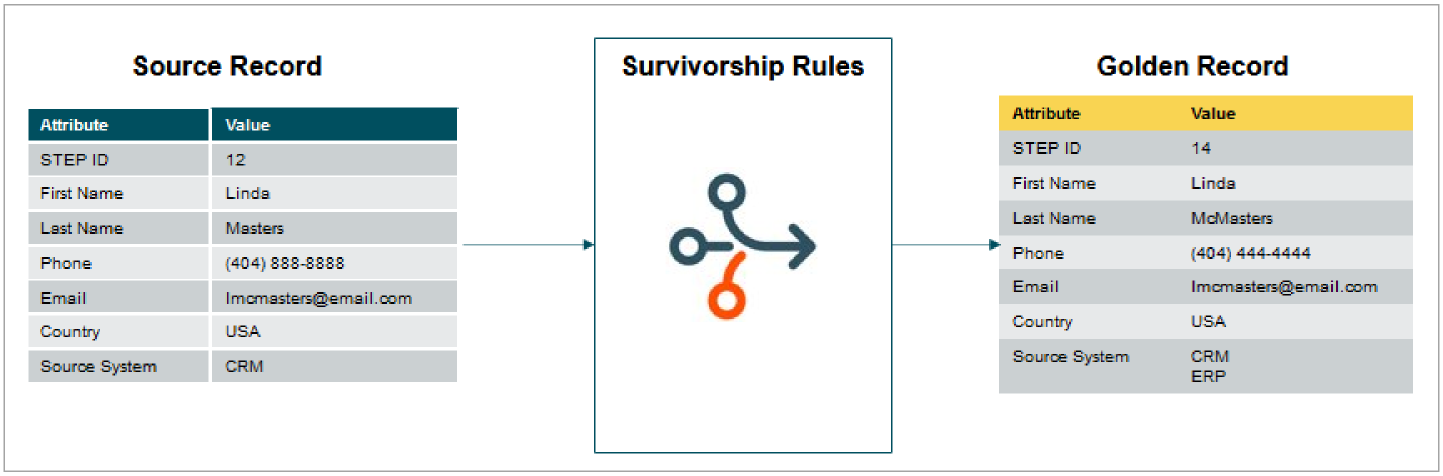
Note: A survivorship rule should never update values outside the golden record.

For more information, see the **Business Actions** topic in the **Business Rule** documentation.

Survivorship in Match and Merge

In match and merge, survivorship rules promote information from exactly one source to exactly one target by comparing information from the source with information from the target and writing the relevant updates to the target.

- In the match and merge importer and match and merge web service endpoint, information is promoted from incoming entities to existing or newly created golden records.
- In matching event processing and in the clerical review Web UI, information is promoted from non-surviving golden records to surviving golden records as those records are merged.
- In unmerge Web UI actions, as the association between source records and golden records are changed, the content of the resulting golden records is resolved.



Source Record

Attribute	Value
STEP ID	12
First Name	Linda
Last Name	Masters
Phone	(404) 888-8888
Email	lmcmasters@email.com
Country	USA
Source System	CRM

Golden Record

Attribute	Value
STEP ID	14
First Name	Linda
Last Name	McMasters
Phone	(404) 444-4444
Email	lmcmasters@email.com
Country	USA
Source System	CRM ERP

Match and Merge Survivorship during Unmerge

Survivorship in unmerge are run to:

- Suggest the values to survive on the golden record that were present before unmerge but existed after a number of sources have been removed from it.
- Suggest the values to survive on a new golden records created by unmerging a number of sources.
- Suggest the values to survive on a reactivated golden record after moving a number of sources to it.

Updating a Golden Record Created through Unmerging

The unmerge process is done when erroneously flagged duplicates are merged together. In this use case, the corrected golden record, created by removing the false sources, will be updated based on the values selected for survivorship.

1. In the unmerge UI, a user removes a number of source records and golden records that do not belong to the record.

2. The algorithm removes values originating from the removed sources. Those values no longer belong on the golden record.
3. The algorithm attempts to restore those cleaned values from revision history, applying the value as it was before it was set to the now cleaned value. This step will not happen for multivalued references and data containers.
4. Finally, the algorithm applies survivorship for all available source records to the golden record. These applications of survivorship rules will function as 'Match and Merge Survivorship update - when import merges with existing record.'

Using Survivorship Rules within the Unmerge Process

If using a golden record that was created from unmerging individual sources, then the process will use survivorship rules like in the previous golden record updating scenario.

1. In the unmerge UI, the user removes a number of sources form a golden record to create this new golden record.
2. The unmerge algorithm sorts the source records associated with the new golden record by the time of editing the records and applies the changes, starting with the oldest source.
3. The survivorship of the oldest source, when applied, will work like the 'Match and Merge Survivorship when Import creates new record' operation.
4. The newer source records, when applied, will work like 'Match and Merge Survivorship update - when import merges with existing record' operation.

Unmerging a Golden Record from Another Golden Record

1. In the unmerge UI, the user removes a falsely merged golden record from another golden record using the unmerge UI.
2. The unmerged golden record is reactivated, and it is assumed to have the attribute values it had when it was merged.
3. The algorithm removes any values that originated from any removed sources. Those values no longer belong on the reactivated golden record.
4. The algorithm attempts to restore those cleaned values from revision history, applying the value as it was before it was set to the now cleaned value. This step will not happen for multivalued references and data containers.
5. Finally, the algorithm applies survivorship for all available source records to the golden record. These applications of survivorship rules will function as 'Match and Merge Survivorship update - when import merges with existing record' operation.

For more information on unmerge, see **Match and Merge Clerical Review - Unmerge** topic in this documentation.

Trusted Source

Trusted source survivorship rules trusts some source systems over others. The rule is configured with a list of the available source systems in the sequence of trust. The systems with lower trust rankings are not able to overwrite values set by higher trust systems.

The source system information this depends on is an integral part of match and merge and is defined in the component model. For more information on how source information is tracked in match and merge, see the **Match and Merge Traceability** topic in this documentation.

Important: Information from a source outside the list of trusted sources is regarded as untrusted, and information from it is not copied to the golden record during trusted source survivorship.

For trusted source, match and merge has the limitation that it is not possible to take the value from a less trusted source. The value from a more trusted source disappears from the source record it came from since that information is not available during the survivorship evaluation.

For more information, see the **Configuration of Survivorship Rules** topic in this documentation.

Most Recent

The 'Most Recent' survivorship rule strategy lets the most recent data from all contributing records survive to the final golden record.

The 'Most Recent' record can be qualified either by the revision date in STEP or by a 'Last Edited' date attribute. The latter makes it possible to promote data based on the time of edit in source systems.

In match and merge, it is differentiated if a given value on an existing golden records comes from a source system or not. If the value does not come from a source system, the revision date is always used in the determination of the most recent value. This makes it possible to do manual edits. This logic applies to the object name, attribute values, references, data containers, attribute values on data containers, attribute values on references, and references on data containers.

In the 'Match and Merge Importer' as well as in the 'Match and Merge Web Service Endpoint,' it is possible to promote the deletion of attribute values on existing golden records by sending an empty value element in the STEP XML. For example, the following STEPXML would void the 'FirstName' attribute value:

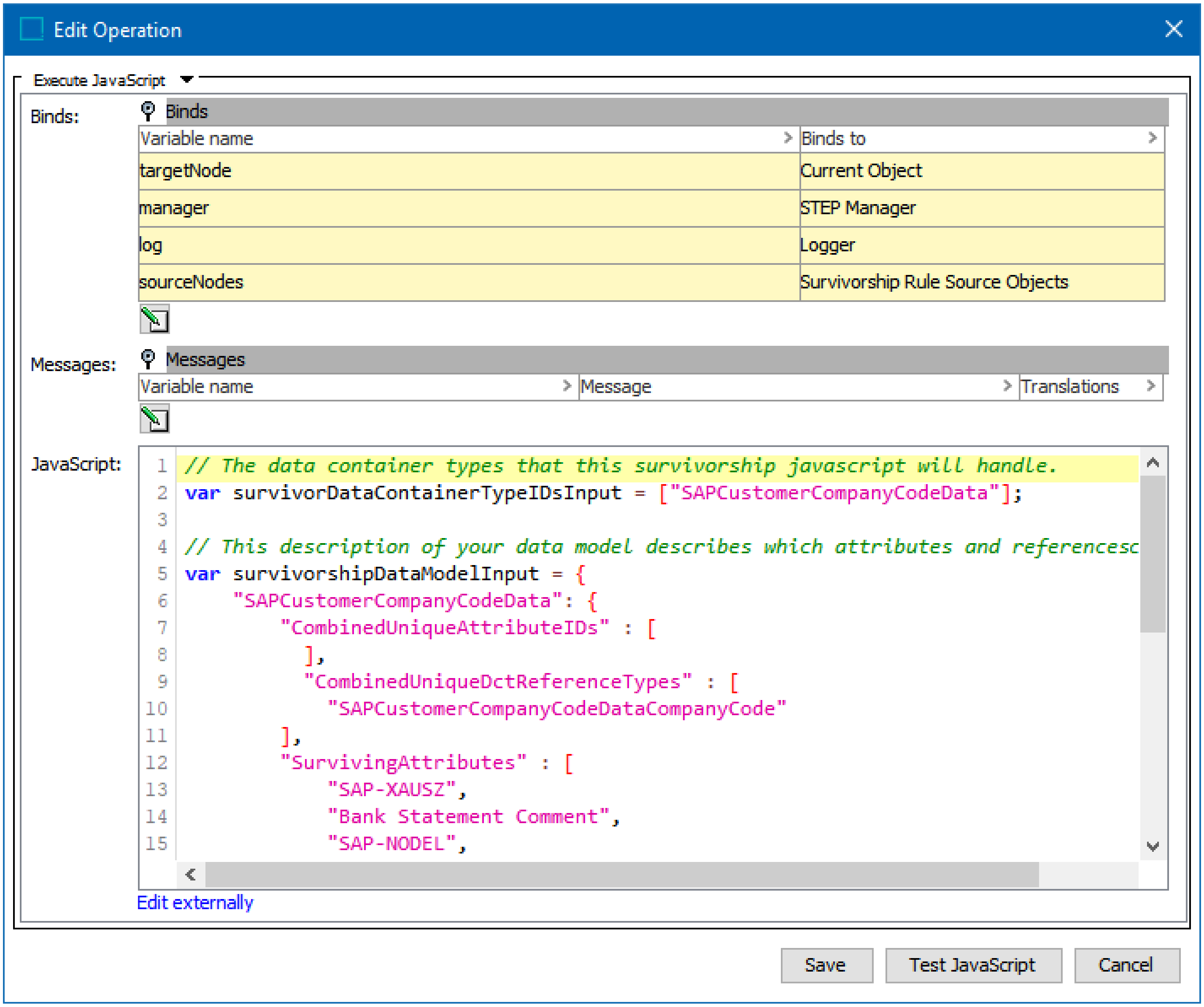
```
<Value AttributeID=”FirstName”></Value>
```

For more information, see the **Configuration of Survivorship Rules** topic in this documentation.

Match and Merge Business Action Rules

It is quite common that special rules for survivorship exist in implementations. In these cases, it is necessary to implement survivorship logic in business actions, which can be run as survivorship rules.

Note: A survivorship rule should never update values outside the golden record.



Variable name	Binds to
targetNode	Current Object
manager	STEP Manager
log	Logger
sourceNodes	Survivorship Rule Source Objects

Variable name	Message	Translations

```
// The data container types that this survivorship javascript will handle.
var survivorDataContainerTypeIDsInput = ["SAPCustomerCompanyCodeData"];

// This description of your data model describes which attributes and referencesc
var survivorshipDataModelInput = {
    "SAPCustomerCompanyCodeData": {
        "CombinedUniqueAttributeIDs" : [
          ],
          "CombinedUniqueDctReferenceTypes" : [
            "SAPCustomerCompanyCodeDataCompanyCode"
        ],
        "SurvivingAttributes" : [
            "SAP-XAUSZ",
            "Bank Statement Comment",
            "SAP-NODEL",
```

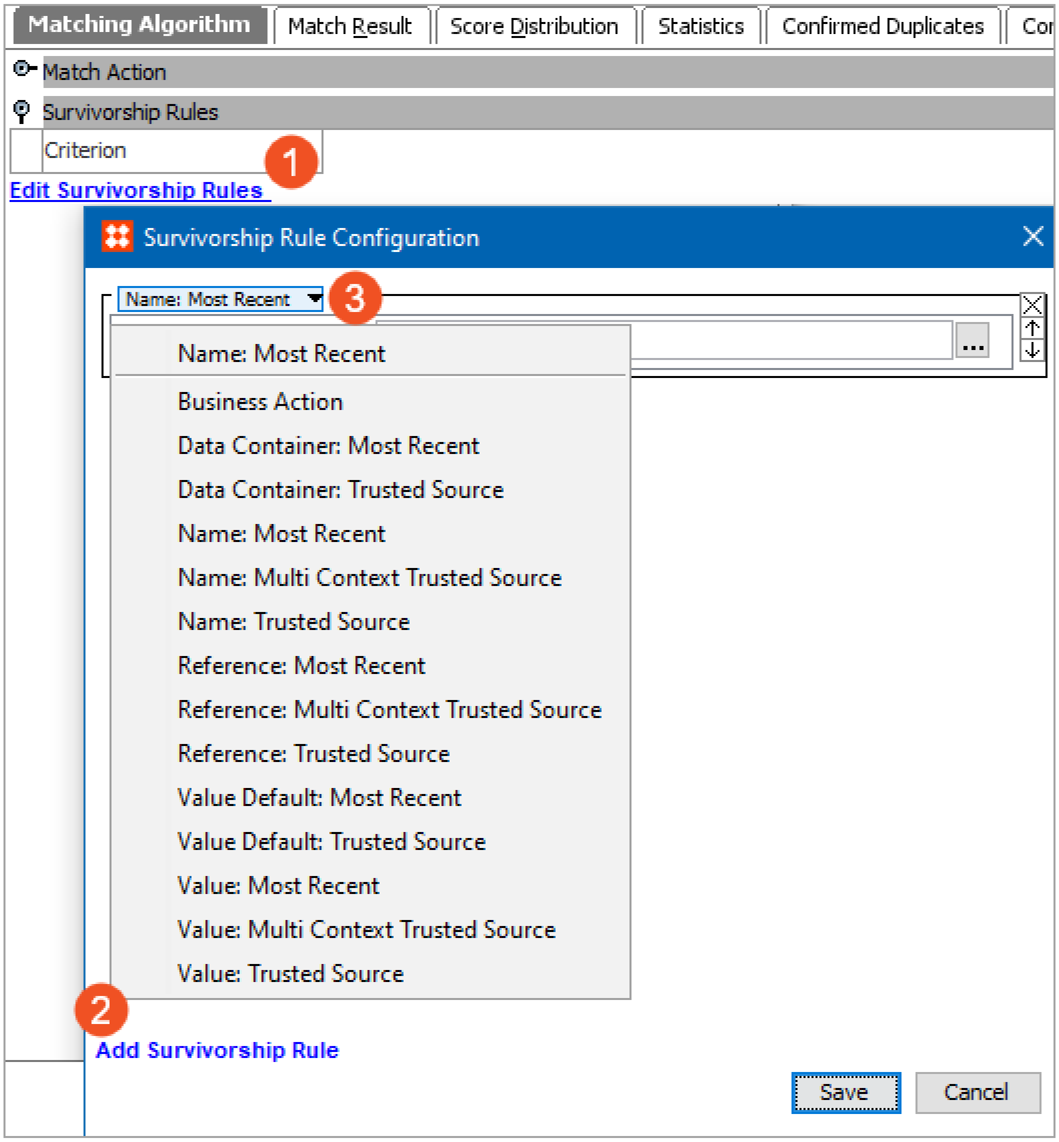
For business action survivorship rules in match and merge, some special binds are available. For more information, see the **Business Actions** topic in the **Business Rules** documentation, the **Survivorship Rule Source Objects Bind** topic, and the **Match and Merge Survivorship Context Bind** topic in the **Resource Materials** documentation.

Note: In match and merge survivorship rules, source records, as well as the target golden record, are in the system as non-persistent objects. A lot of operations that are available in the API are not applicable to non-persistent objects and will fail. Operations related to reading and modifying attributes values, references, and data containers can be used. Examples of operations that cannot be used are approval and workflow-related operations.

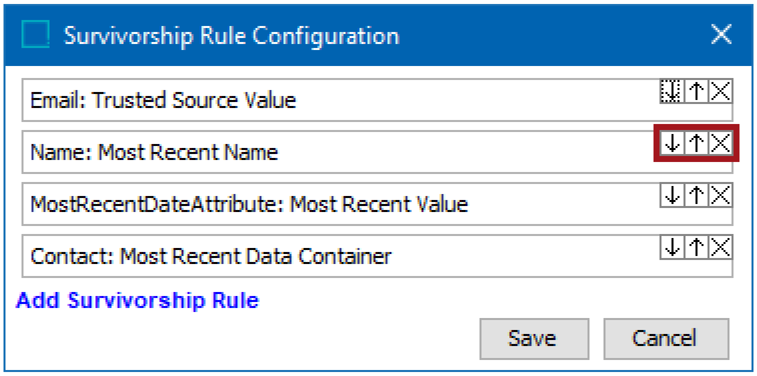
In match and merge, it is not possible to implement a trusted source pattern with the business action survivorship rule as the source information for an existing value on the golden record is not available in the JavaScript API.

Configuration of Survivorship Rules

To add a survivorship rule, on the Matching Algorithm tab, open the 'Survivorship Rules' flipper and click the 'Edit Survivorship Rules' link. In the 'Survivorship Rule Configuration' dialog, click the 'Add Survivorship Rule' link, and select the required rule from the dropdown.



When multiple rules are added, they are executed in order from top-to-bottom. Use the down and up buttons to change the order of the rules. Use the X button to remove a rule.


Survivorship Rule Configuration

Email: Trusted Source Value

Name: Most Recent Name

MostRecentDateAttribute: Most Recent Value

Contact: Most Recent Data Container

Name Rules

The following rules are available for an object name to a golden record.

Name: Most Recent

This rule can be used for merge or link strategies and it specifies that 'Name' should be taken from the source object with the most recent name. No configuration is required for this rule. The analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers.

- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.

Name: Multi Context Trusted Source

This rule can be used for link strategy only and considers data that is dimension dependent. The analysis is performed for all contexts / qualifiers (a set of one or more dimension points, like country and language) in STEP. The available parameters determine which name is promoted to the golden record.

- Comma separated list of trusted sources: Enter a comma-separated list of all trusted sources, starting with the most trusted source, then the next-most, and so on. Content is taken from the first trusted source with data. If content does not exist for any of the trusted sources, nothing will be promoted to the golden record.

- Promote single source only: When checked, the name from the most trusted source is used for all contexts / qualifiers, which prevents an empty name value in the golden record as long as one of the trusted sources has a name. For example, when only the French language, France country context has a name value, that value would be written into all other contexts. Alternatively, when not checked, each context / qualifier supplies its own name, including empty values, when found.

- Prefer dimension point specific names: When checked, only a local name is promoted. When not checked, available inherited content is promoted if a local name does not exit.

Name: Trusted Source

This rule can be used for merge or link strategies and determines that 'name' is taken from the most trusted source. The analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers.

- **Comma-separated list of trusted sources:** Enter a comma-separated list of all trusted sources, starting with the most trusted source, then the next-most, and so on. Content is taken from the first trusted source with data. If content does not exist for any of the trusted sources, nothing will be promoted to the golden record.
- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.

Value Rules

The following rules are available for promoting values to a golden record.

Value Default: Most Recent

This rule can be used for merge or link strategies and determines that the value is taken from the source with the most recent date for all attributes. No attribute or attribute group selection is required for this rule. The Analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers.

- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.

Value Default: Trusted Source

This rule can be used for merge or link strategies and determines that the value is taken from the most trusted source for all attributes. No attribute or attribute group selection is required for this rule. The analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers.

- **Comma-separated list of trusted sources:** Enter a comma-separated list of all trusted sources, starting with the most trusted source, then the next-most, and so on. Content is taken from the first trusted source with data. If content does not exist for any of the trusted sources, nothing will be promoted to the golden record.
- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most

recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.

Value: Most Recent

This rule can be used for merge or link strategies and it specifies that value should be taken from the source object with the most recent value. The analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers.

- **Attribute / Attribute Group:** Click the ellipsis button (...) and select a single attribute or all attributes in a specific group for which the rule applies.
- **The group with the latest value change always survives:** If this option is selected, then all values of an attribute group will survive when the group contains the attribute with the most recent timestamp among all compared attribute groups.
- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.
- **Survive incoming empty values:** If this option is selected, then when an empty value is imported, it will replace any existing value. This works for IIEP and Imports.

Value: Multi Context Trusted Source

This rule can only be used for link strategies and considers data that is dimension dependent. The analysis is performed for all contexts / qualifiers (a set of one or more dimension points, like country and language) in STEP. The available parameters determine which value is promoted to the golden record.

- **Comma-separated list of trusted sources:** Enter a comma-separated list of all trusted sources, starting with the most trusted source, then the next-most, and so on. Content is taken from the first trusted source with data. If content does not exist for any of the trusted sources, nothing will be promoted to the golden record.
- **Attribute / Attribute Group:** Click the ellipsis button (...) and select a single attribute or all attributes in a specific group for which the rule applies.
- **Promote single source only:** When checked, content from the most trusted source is used for all contexts / qualifiers, which prevents empty values in the golden record as long as one of the trusted sources has content. For example, when only the French language, France country context has a value, that value would be written into other contexts that are blank. When not checked, each context / qualifier supplies its own content, including empty values when found.
- **Prefer dimension point specific values:** When checked, only local values are promoted for the selected

attribute / attribute group. When not checked, available inherited content is promoted if a local value does not exit for the selected attribute / attribute group.

> **Note:** If both the 'Prefer dimension point specific values' and the 'Promote single source only' options are checked, then 'Promote single source only' takes precedence, and only values from that source are promoted for the selected attribute / attribute group.

- **Promote inherited values:** When checked, inherited values are written to the golden record for the selected attribute / attribute group only if the golden record object type is valid. When not checked, only local values are written to the golden record for the selected attribute / attribute group.

Value: Trusted Source

This rule can be used for merge or link strategies and determines that the value is taken from the most trusted source. The analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers.

- **Comma-separated list of trusted sources:** Enter a comma-separated list of all trusted sources, starting with the most trusted source, then the next-most, and so on. Content is taken from the first trusted source with data. If content does not exist for any of the trusted sources, nothing will be promoted to the golden record.
- **Attribute / Attribute Group:** Click the ellipsis button (...) and select a single attribute or all attributes in a specific group for which the rule applies.
- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.

Data Container Rules

The following rules are available for promoting data container values to a golden record.

Data Container: Most Recent

This rule can be used for merge or link strategies and allows the most recent data container instances and their attribute values to be promoted to the golden records. Analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers. The following parameters must be configured:

- **Business Condition**: Click the ellipsis button (...) and select a business condition that is valid for the golden record object type. If the source record should always overwrite the golden record, the condition should return true. Otherwise, this condition must be a JavaScript rule that uses the 'Pairs of Attributes' bind to compare data container instances on source records with data container instances on golden records when survivorship rules are applied. For more information and an example of the bind, see the **Pair of Attribute Values Bind** topic in the **Resource Materials** documentation.

> **Note:** If the data container type being merged has a Data Container Key configured, there is no need to configure a business condition. For more information on data container keys, see the **Data Container Keys** topic in the **System Setup / Super User Guide** documentation.

- **Data Container Type:** Click the ellipsis button (...) and select the relevant data container type.

- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.

> **Note:** Survivorship rules look at Last Edit Date attributes on the entities themselves before they look for them within a data container. Additionally, in the case of multi-value data container types, it takes the newest date from all data containers of the type in question.

Data Container: Trusted Source

This rule can be used for merge or link strategies and allows data container instances and their attribute values that originate from the specified trusted source(s) to be promoted to the golden records. Analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers. The following parameters must be configured:

- **Business Condition:** Click the ellipsis button (...) and select a business condition that is valid for the golden record object type. If the source record should always overwrite the golden record, the condition should return true. Otherwise, this condition must be a JavaScript rule that uses the 'Pairs of Attributes' bind to compare data container instances on source records with data container instances on golden records when survivorship rules are applied. For more information and an example of the bind, see the Pair of Attribute Values Bind section in the Resource Materials documentation.

> **Note:** If the data container type being merged has a Data Container Key configured, there is no need to configure a business condition. For more information on data container keys, see the **Data Container Keys** topic in the **System Setup / Super User Guide** documentation.

- **Comma-separated list of trusted sources:** Enter a comma-separated list of all trusted sources, starting with the most trusted source, then the next-most, and so on. Content is taken from the first trusted source with data. If content does not exist for any of the trusted sources, nothing will be promoted to the golden record.

- **Data Container Type:** Click the ellipsis button (...) and select the relevant data container type.

- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.

> **Note:** Survivorship rules look at Last Edit Date attributes on the entities themselves before they look for them within a data container. Additionally, in the case of mult-value data container types, it takes the newest date from all data containers of the type in question.

Data Container Keys Survivorship Rule Principles For Inconsistent Keys

When configuring survivorship rules to account for data containers that have inconsistent keys, there are a few principles to consider:

- Data containers with inconsistent keys will not survive a merge using standard survivorship rules. It does not matter if the key is incomplete or duplicated.
- When updating a target with a duplicate key, the data container with the lowest internal STEP ID survives.
- Survivorship rules will never write an incomplete key.
- Survivorship rules will never add a data container instance that has a duplicated key.

Reference Rules

The following rules are available for promoting references / links to a golden record.

Reference: Most Recent

This rule can be used for merge or link strategies and it allows selecting the three reference / link types that should be promoted from the source object with the most recent reference / link. Analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers.

- **Reference Type:** Required. Click the ellipsis button (...) to specify the valid reference / link type from the source objects you are handling. When this is the only field populated, a reference / link of the same type pointing to the same target will be promoted to the golden record.

- **Golden Record Reference Type:** Optional. If the objects the source objects are pointing to also have golden records, you can configure the new golden record to point to this golden record rather than the source object's original target. Click the ellipsis button (...) to specify the reference type that links the target golden records and target source objects.

- **Mapping Reference Type:** Optional. When this field is not populated, the reference or link created for the golden record will be of the same type as the source object's reference / link. Click the ellipsis button (...) to specify a reference / link type mapped to this reference / link type.

- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.

Reference: Multi Context Trusted Source

This rule can be used for link strategies only and considers data that is dimension dependent. Analysis is performed for all contexts / qualifiers (a set of one or more dimension points, like country and language) in STEP. The available parameters determine which reference / link is promoted to the golden record.

- **Comma-separated list of trusted sources:** Enter a comma-separated list of all trusted sources, starting with the most trusted source, then the next-most, and so on. Content is taken from the first trusted source with data. If content does not exist for any of the trusted sources, nothing will be promoted to the golden record.
- **Reference Type:** Required. Click the ellipsis button (...) to specify the valid reference / link type from the source objects you are handling. When this is the only field populated, a reference / link of the same type pointing to the same target will be promoted to the golden record.
- **Golden Record Reference Type:** Optional. If the objects the source objects are pointing to also have golden records, you can configure the new golden record to point to this golden record rather than the source object's original target. Click the ellipsis button (...) to specify the reference type that links the target golden records and target source objects.
- **Mapping Reference Type:** Optional. When this field is not populated, the reference or link created for the golden record will be of the same type as the source object's reference / link. Click the ellipsis button (...) to specify a reference / link type mapped to this reference / link type.
- **Promote single source only:** When checked, content from the most trusted source is used for all contexts / qualifiers, which prevents empty values in the golden record as long as one of the trusted sources has content. For example, when only the French language, France country context has a value, that value would be written into other contexts that are blank. When not checked, each context / qualifier supplies its own content, including empty values when found.
- **Accumulative promotions:** When checked, all multi-valued references / links and their metadata from multiple source records are written to the golden record. When not checked, only all multi-valued references / links and their metadata from the most trusted source records are written to the golden record. This option should not be used when a single-valued reference / link type is selected.

> **Note:** If both the 'Accumulative promotions' and the 'Promote single source only' options are checked, then 'Promote single source only' takes precedence, and only references / links from that source are promoted.

- **Prefer dimension point specific references:** When checked, only local references / links are promoted. When not checked, available inherited content is promoted if a local reference / link does not exit. This option can be used in conjunction with the 'Accumulative promotions' option, in order to determine which reference / link to promote when multiple source records have references / links to the same target object.

> **Note:** If both the 'Prefer dimension point specific references' and the 'Promote single source only' options are checked, then 'Promote single source only' takes precedence, and only references from that source are promoted.

- **Promote inherited references:** When checked, inherited references / links are written to the golden record only if the golden record object type is valid for the selected reference type. When not checked, only local references are written to the golden record.
- **Promote reference suppressions:** When checked, suppressed references / links are written to the golden record. When not checked, suppressed references / links are ignored.

Reference: Trusted Source

This rule can be used for merge or link strategies and includes the same options available as the 'Reference: Most Recent' rule, with the addition of a list of trusted sources. Analysis is performed in the single context / workspace selected in the algorithm, and that data is promoted across all contexts / qualifiers.

- **Comma-separated list of trusted sources:** Enter a comma-separated list of all trusted sources, starting with the most trusted source, then the next-most, and so on. Content is taken from the first trusted source with data. If content does not exist for any of the trusted sources, nothing will be promoted to the golden record.
- **Reference Type:** Required. Click the ellipsis button (...) to specify the valid reference / link type from the source objects you are handling. When this is the only field populated, a reference / link of the same type pointing to the same target will be promoted to the golden record.
- **Golden Record Reference Type:** Optional. If the objects the source objects are pointing to also have golden records, you can configure the new golden record to point to this golden record rather than the source object's original target. Click the ellipsis button (...) to specify the reference type that links the target golden records and target source objects.
- **Mapping Reference Type:** Optional. When this field is not populated, the reference or link created for the golden record will be of the same type as the source object's reference / link. Click the ellipsis button (...) to specify a reference / link type mapped to this reference / link type.
- **Last Edit Date Attribute:** When no attribute is selected, the most recent date is the STEP object revision timestamp when the given element of the survivorship rule entered STEP. Optionally, click the ellipsis button (...) and select the attribute that holds the value to be used as the last edit date when determining the most recent source record to promote to the golden record. The timestamp is taken from the object when the selected attribute is valid for this object. When the selected attribute is not valid for the object, the value is taken from the given element of the survivorship rule, for example, a data container object or a reference object.

Business Action Rule

This rule can be used for merge or link strategies. Click the ellipsis button (...) to specify a business action to run on golden records when survivorship rules are applied.

When using a JavaScript business action with the merge golden record solution, you will need the 'Survivorship Rules Source Object' bind. This bind grants the script access to the temporary source objects so that any relevant values can be promoted from them to the surviving golden records.

Note: When data is promoted to a golden record, it is done across all contexts (the evaluation is performed in the context and workspace selected on the algorithm) and only the data for which survivorship rules exist will be promoted. Inherited and calculated values are not used.

Note: When writing JavaScript survivorship rules be aware that if the source has a reference to itself, that reference has already had its target moved to the surviving record before survivorship rules are run.

Important: With the match and merge solution, survivorship rules should only update the surviving entity and not other entities referencing the survivor. This is also the recommended behavior in match and link.

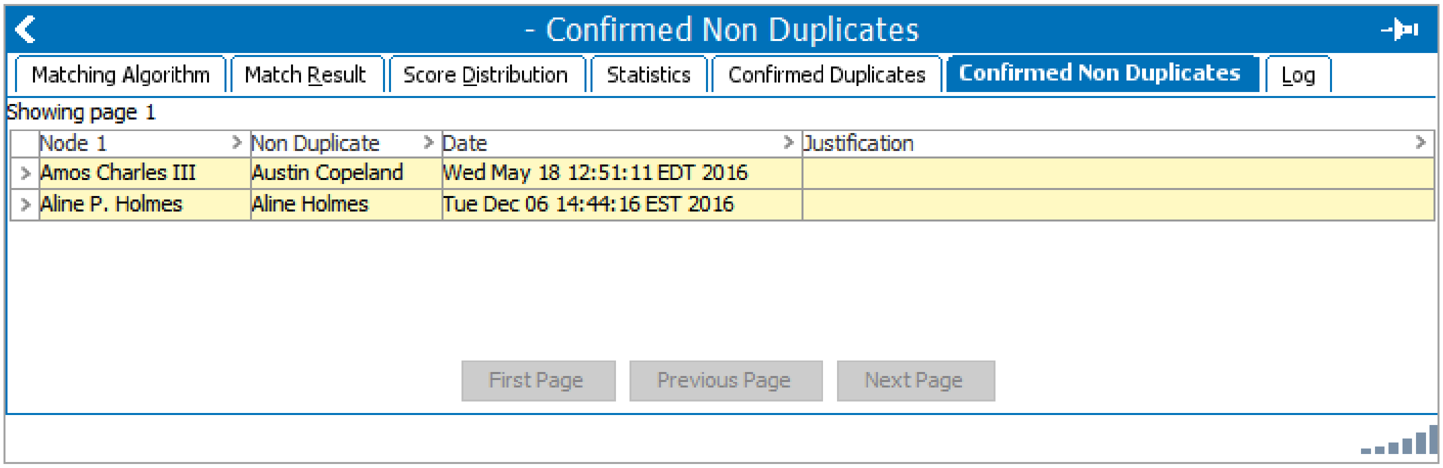
Tuning and Monitoring a Matching Algorithm

The matching algorithm has a number of tools that can be helpful in tuning and monitoring the results of the matching algorithm.

Confirmed Duplicates and Confirmed Non Duplicates

A confirmed duplicate is represented by a specific reference type in the system, and such a reference between two records is the result of someone manually reviewing and confirming a match.

You can get an overview of all confirmed duplicates and non-confirmed duplicates in the appropriate tabs in the workbench.



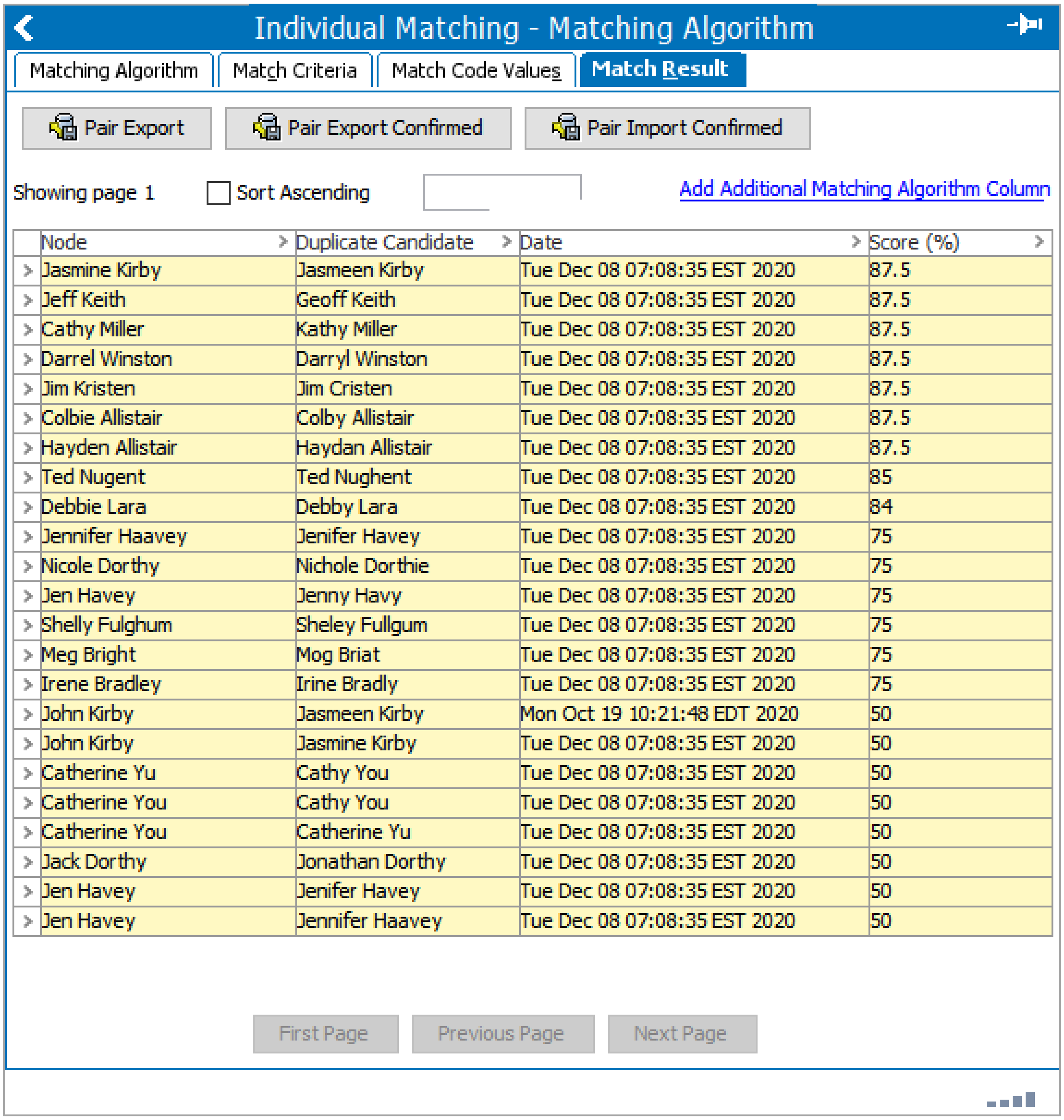
Showing page 1

Node 1	Non Duplicate	Date	Justification
Amos Charles III	Austin Copeland	Wed May 18 12:51:11 EDT 2016	
Aline P. Holmes	Aline Holmes	Tue Dec 06 14:44:16 EST 2016	

Note: For Match and Merge Match Action, the Confirmed Duplicate reference is deleted by the merge operation, and thus the 'Confirmed Duplicates' tab is almost always empty.

Match Result Tab

When a matching algorithm has first been applied, the identified matches are displayed on the 'Match Result' tab of the matching algorithm. Three options related to fine-tuning the matching algorithm are available: 'Pair Export,' 'Pair Import Confirmed,' and 'Pair Export Confirmed.'



Node	Duplicate Candidate	Date	Score (%)
Jasmine Kirby	Jasmeen Kirby	Tue Dec 08 07:08:35 EST 2020	87.5
Jeff Keith	Geoff Keith	Tue Dec 08 07:08:35 EST 2020	87.5
Cathy Miller	Kathy Miller	Tue Dec 08 07:08:35 EST 2020	87.5
Darrel Winston	Darryl Winston	Tue Dec 08 07:08:35 EST 2020	87.5
Jim Kristen	Jim Cristen	Tue Dec 08 07:08:35 EST 2020	87.5
Colbie Allistair	Colby Allistair	Tue Dec 08 07:08:35 EST 2020	87.5
Hayden Allistair	Haydan Allistair	Tue Dec 08 07:08:35 EST 2020	87.5
Ted Nugent	Ted Nughent	Tue Dec 08 07:08:35 EST 2020	85
Debbie Lara	Debby Lara	Tue Dec 08 07:08:35 EST 2020	84
Jennifer Haavey	Jenifer Havey	Tue Dec 08 07:08:35 EST 2020	75
Nicole Dorthy	Nichole Dorthie	Tue Dec 08 07:08:35 EST 2020	75
Jen Havey	Jenny Havy	Tue Dec 08 07:08:35 EST 2020	75
Shelly Fulghum	Sheley Fullgum	Tue Dec 08 07:08:35 EST 2020	75
Meg Bright	Mog Briat	Tue Dec 08 07:08:35 EST 2020	75
Irene Bradley	Irine Bradly	Tue Dec 08 07:08:35 EST 2020	75
John Kirby	Jasmeen Kirby	Mon Oct 19 10:21:48 EDT 2020	50
John Kirby	Jasmine Kirby	Tue Dec 08 07:08:35 EST 2020	50
Catherine Yu	Cathy You	Tue Dec 08 07:08:35 EST 2020	50
Catherine You	Cathy You	Tue Dec 08 07:08:35 EST 2020	50
Catherine You	Catherine Yu	Tue Dec 08 07:08:35 EST 2020	50
Jack Dorthy	Jonathan Dorthy	Tue Dec 08 07:08:35 EST 2020	50
Jen Havey	Jenifer Havey	Tue Dec 08 07:08:35 EST 2020	50
Jen Havey	Jennifer Haavey	Tue Dec 08 07:08:35 EST 2020	50

To determine how well different versions of a matching algorithm work, you will need a set of confirmed duplicates and confirmed non-duplicates that can function as a truth table that the algorithms can be tested against: that is, pairs where human users have inspected the data, and for each pair, determined whether they are duplicates or not. This truth table can be built directly from the 'Match Result' tab by confirming or rejecting matched pairs.

Alternately, the 'Pair Export' and 'Pair Import Confirmed' options can be used.

> **Note:** For Match and Merge solutions, the Pair Import and Export tools are not applicable for early evaluations. Instead, such solutions should make use of the Match Tuning functionality in order to adjust matching algorithms. For more information, see the Match Tuning documentation.

Pair Export

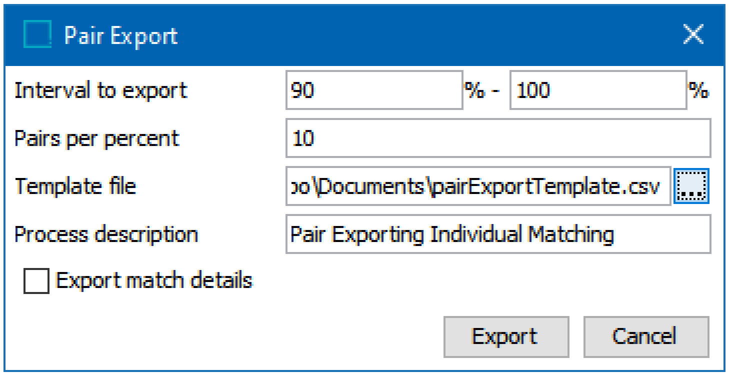
With the 'Pair Export' option, a CSV file is produced that can be used for manual, offline confirmation / rejection of matched pairs.

The file has a header and the following standard columns:

- **<Pair>** - One row per source object and the 'Pair' info is used to indicate which objects belong together. The first two rows will have the value '1,' the next two rows will have '2,' and so on.
- **<Match y n>** - Column used to indicate whether pairs are matches or not. A value is only required for the first object in a pair.
- **<Equality>** - The calculated equality between the two objects.
- **<ID>** - ID of the object in the current row.
- **<Name>** - Name of the object in the current row.
- **<URL>** - STEP URL of the object in the current row.

> **Note:** No template is required for the initial export. The export itself will create a CSV file with the above details.

Additionally, for people to work with the data offline, attribute values should be included in the file. For this purpose, a template file with a semicolon-separated list of attribute IDs must be prepared in advance and selected in the 'Pair Export' dialog as shown below.

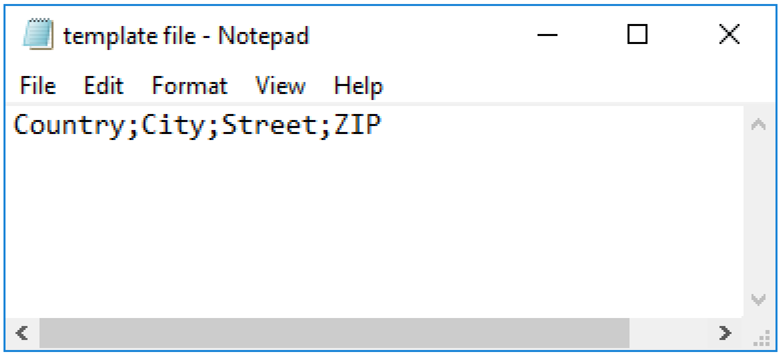


Interval to export: 90 % - 100 %
Pairs per percent: 10
Template file: ɔo\Documents\pairExportTemplate.csv
Process description: Pair Exporting Individual Matching
☐ Export match details

In the Pair Export dialog, specify the following:

- **Interval to export:** Specify an interval that includes pairs expected to be both matches and non-matches, as well as pairs that are not clear matches or non-matches. Only pairs with scores within this interval are exported.
- **Pairs per percent:** Specify the maximum number of pairs to be exported for each percentage point.
- **Template file:** Select the file (created beforehand) that contains the required attribute values. The format of the file is the attribute IDs separated by semicolons (;).

  This file should be a basic text document such as the one pictured below:



```
Country;City;Street;ZIP
```

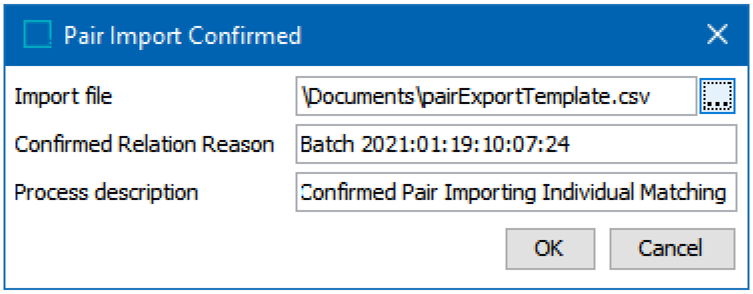
- **Process description:** Provide a description for the background process found under the Background Process tab.
- **Export Match Details:** Check this parameter to include columns with part scores from decision table comparators and sub decision tables.

The exported file can be opened in Excel, and the decisions can be entered in the <Match y n> column.

Pair Import Confirmed

Once the file exported via the pair export option has been populated with matches, it can be imported via the 'Pair Import Confirmed' option.

Rather than use the match column, the 'Pair Import Confirmed' process uses its own columns for identification purposes but does not import any other data. This way you can avoid reverting values updated elsewhere since the pair export was performed.



Import file: \Documents\pairExportTemplate.csv

Confirmed Relation Reason: Batch 2021:01:19:10:07:24

Process description: Confirmed Pair Importing Individual Matching

In the Pair Import Confirmed dialog, specify the following:

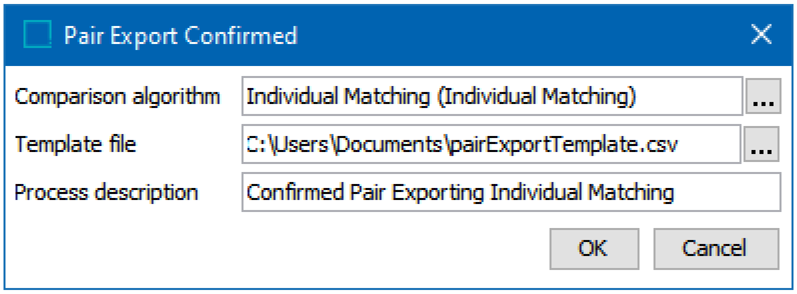
- **Import File:** Select the file that you want to import. This must be a CSV file produced by a pair export process, using a semicolon delimiter. Keep the header line.
- **Confirmed Relation Reason:** Provide a reason for why two objects have been confirmed as duplicates or non-duplicates. This reason is saved on each confirmed relation as a meta data attribute and can be viewed on the matching tab of the relevant objects.
- **Process description:** Provide a description for the background process.

The file import will create 'Confirmed Duplicate' / 'Confirmed Non Duplicate' references between the pair objects.

Pair Export Confirmed

The 'Pair Export' option is used when you want to compare two versions of a matching algorithm against each other and against the confirmed duplicates / non duplicates truth table constructed manually or via the steps described above.

When using this functionality, it is assumed that you have duplicated an earlier version of your matching algorithm, and now wish to compare that to your fine-tuned version of your matching algorithm. Along with a template file like the one produced for the 'Pair Export' option, the fine-tuned matching algorithm must be selected in the 'Pair Export Confirmed' dialog as shown below.



Comparison algorithm: Individual Matching (Individual Matching)

Template file: C:\Users\Documents\pairExportTemplate.csv

Process description: Confirmed Pair Exporting Individual Matching

In the 'Pair Export Confirmed' dialog, specify the following:

- **Comparison Algorithm:** Select the matching algorithm that you want to compare the selected algorithm to.
- **Template File:** Select the file (created beforehand) that contains the required attribute values. The format of the file is the attribute IDs, separated by semicolons (;).
- **Process description:** Provide a description for the background process.

Confirmed Matches Distribution Tool

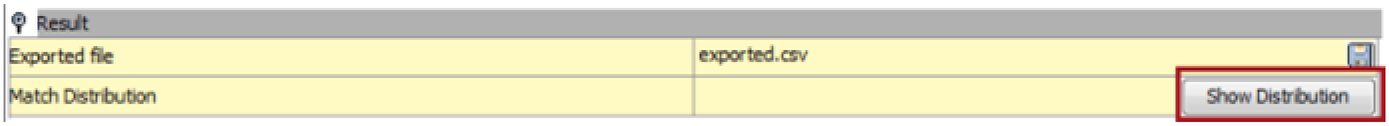
Once the Pair Export Confirmed background process has finished, a CSV file with the comparison results will be produced and the Match Distribution tool made available. This tool can be used to visualize the differences between the match algorithms and compare their accuracy.

When reviewing the results, you may find false negatives and false positives, which are the errors produced by the algorithm when compared to the manually reviewed pairs. Ideally, the count should be 0, which is the goal of fine-tuning the algorithms. However, even if the count is 0, it does not mean that the algorithm is perfect. The reliability of the result depends on the amount of data in the data set and the representativeness of the data.

If you find that the fine-tuned version of the matching algorithm produces fewer 'False Positives' and 'False Negatives,' you can either copy the logic to the original matching algorithm or let the fine-tuned version replace the original one.

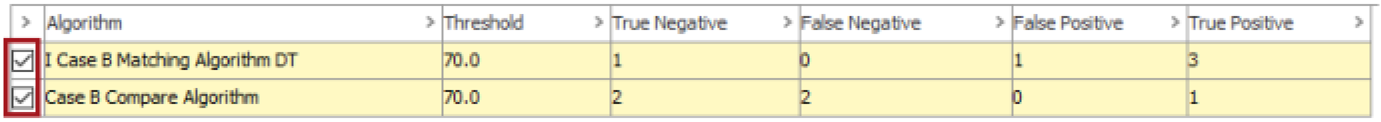
To access the Distribution Tool:

1. On 'BG Processes,' expand 'Matching Pair Export,' and then select the relevant confirmed export process.
2. At the bottom right corner, click 'Show Distribution.' The 'Confirmed Matches Distribution' window opens.



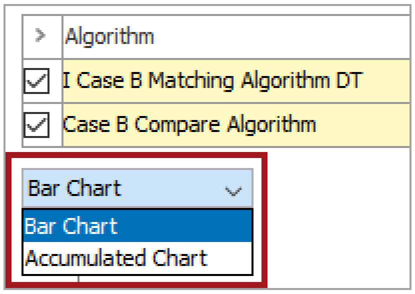
Result	
Exported file	exported.csv
Match Distribution	

3. Select the checkbox to view the algorithm data in the chart.



	Algorithm	Threshold	True Negative	False Negative	False Positive	True Positive
☑	I Case B Matching Algorithm DT	70.0	1	0	1	3
☑	Case B Compare Algorithm	70.0	2	2	0	1

4. From the list, select whether to view the data in a bar chart or an accumulated chart.



	Algorithm
☑	I Case B Matching Algorithm DT
☑	Case B Compare Algorithm

The table shows the following information about each algorithm:

- **Algorithm:** The ID of the algorithm.
- **Threshold:** The threshold that is used to distinguish between positives and negatives.
- **True Negative:** The number of comparisons that were classified as a non-match, both manually, and by the algorithm.
- **False Negative:** Count of comparisons that were manually classified as a match but were classified as a non-match by the algorithm because the scores were below the threshold.
- **False Positive:** Count of comparisons that were manually classified as a non-match but were classified as a match by the algorithm because the scores were above the threshold.
- **True Positive:** Count of comparisons that were classified as a match both manually and by the algorithm.

The Bar Chart and the Accumulated Chart

The colors used in the charts are unique and identified in the chart legend. Common for both charts, green represents relations that have been manually confirmed as duplicates, and red represents relations that have been manually confirmed as non-duplicates.

The threshold of the algorithm is shown as a vertical line.

Generally, the red bars are displayed to the left of the threshold indicator and the green bars to the right. If green bars are displayed to the left of the threshold indicator, they represent false negatives, and if red bars are displayed to the right of the threshold indicator, they represents false positives.

The bars only have a resolution of 1 percent point. Therefore, you cannot always read the exact number of false positives and false negatives directly from the graph. However, the exact number is listed in the table above it.

- **The Bar Chart**

  The bars in the chart show the frequency of the scores of the selected algorithm. The bar chart can either show a single algorithm or two algorithms in a special compare mode that enables a detailed comparison of the two algorithms.

  When clicking a bar, you can view a table that contains an extract of the corresponding data from the CSV file. This enables you to perform a quick inspection of the attribute values of the pairs.

  If you need to investigate the algorithm behavior further for a given pair, click the binocular icon() . The matching algorithm editor is opened with the relevant pair selected in the System Setup tab.

  You can then use the Evaluator or open the individual criteria to view more detailed information about the pair.

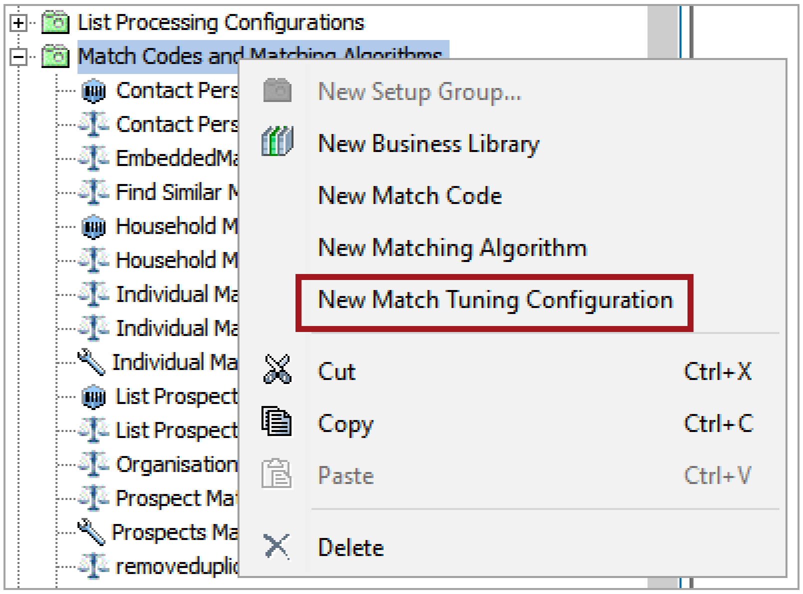
- **The Accumulated Chart**

  The accumulated chart shows the accumulated score frequency for the algorithms. The manually classified matches are green and accumulated to the right of the threshold line. The manually classified no-matches are red and are accumulated to the left. The accumulated chart is useful when you want to compare the matching abilities of two algorithms because it is easy to evaluate the number of scores up to a certain point. The chart is also useful for identifying a good threshold value.

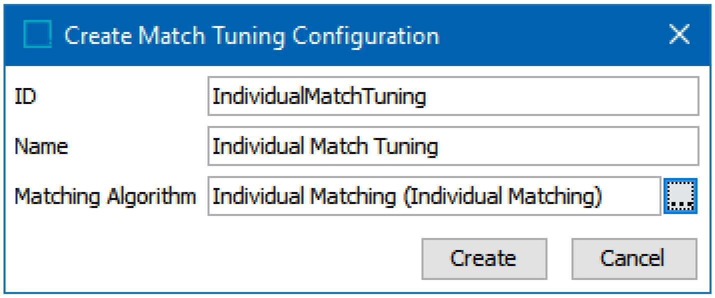
Match Tuning Configuration

Before a Match Tuning configuration can be created, ensure that the basic object type for Match Tuning configurations under 'System Setup > Object Types & Structures' has been made valid for a relevant setup group. For more information, see the **Creating an Object Type** topic in the **System Setup / Super User Guide** documentation.

1. In the System Setup tab, right-click the node configured to house matching tuning configurations and select 'New Match Tuning Configuration.'



2. In the 'Create Match Tuning Configuration' dialog, define an ID and name for the configuration, and specify a matching algorithm to test, then click 'Create.'

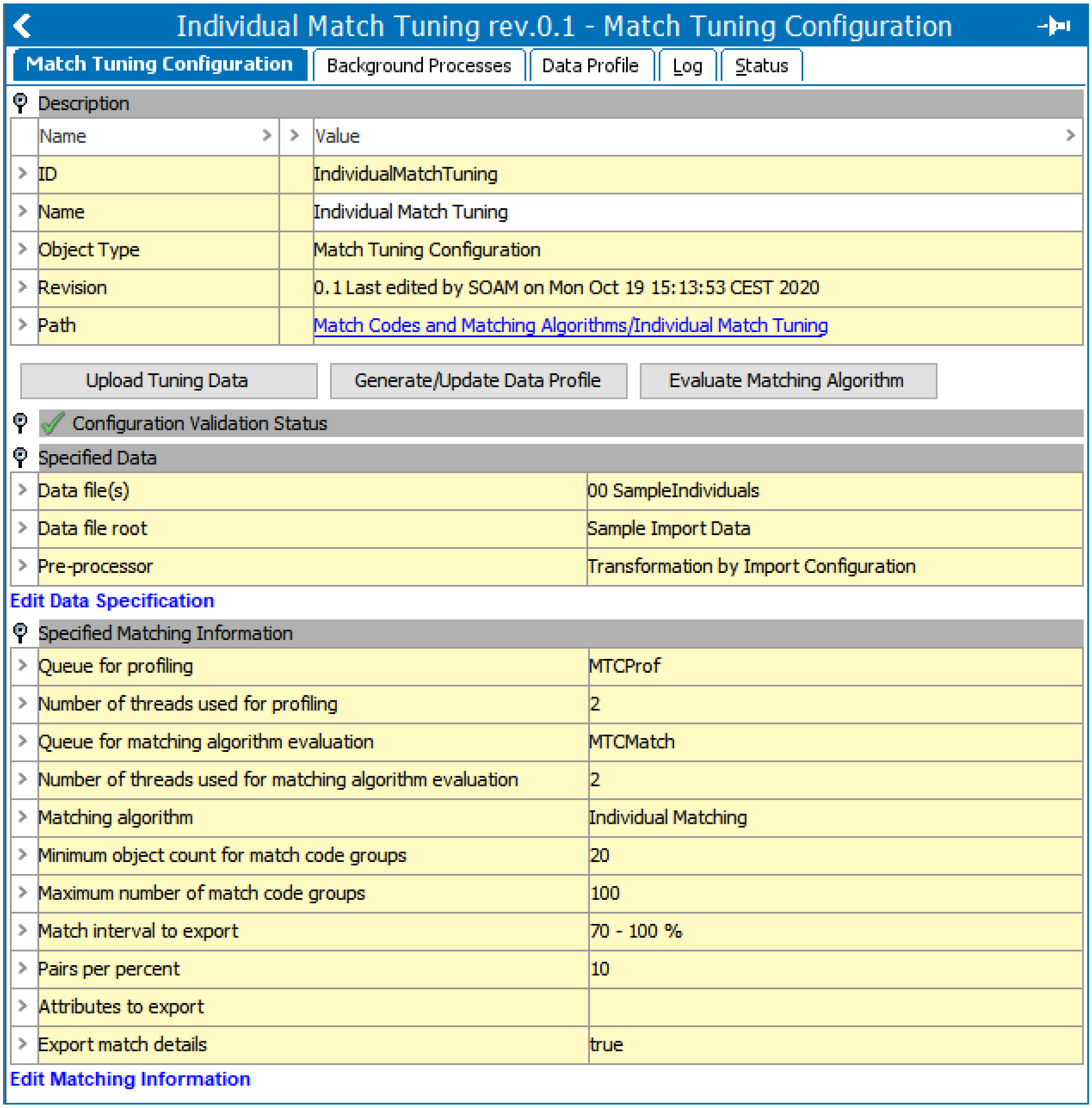

Create Match Tuning Configuration

ID: IndividualMatchTuning

Name: Individual Match Tuning

Matching Algorithm: Individual Matching (Individual Matching)

3. Click on the 'Match Tuning Configuration' tab to view the overall configuration.



Individual Match Tuning rev.0.1 - Match Tuning Configuration
Match Tuning Configuration
Background Processes
Data Profile
Log
Status

Description

Name	Value
ID	IndividualMatchTuning
Name	Individual Match Tuning
Object Type	Match Tuning Configuration
Revision	0.1 Last edited by SOAM on Mon Oct 19 15:13:53 CEST 2020
Path	Match Codes and Matching Algorithms/Individual Match Tuning

Upload Tuning Data
Generate/Update Data Profile
Evaluate Matching Algorithm

✓ Configuration Validation Status

Specified Data

Data file(s)	00 SampleIndividuals
Data file root	Sample Import Data
Pre-processor	Transformation by Import Configuration

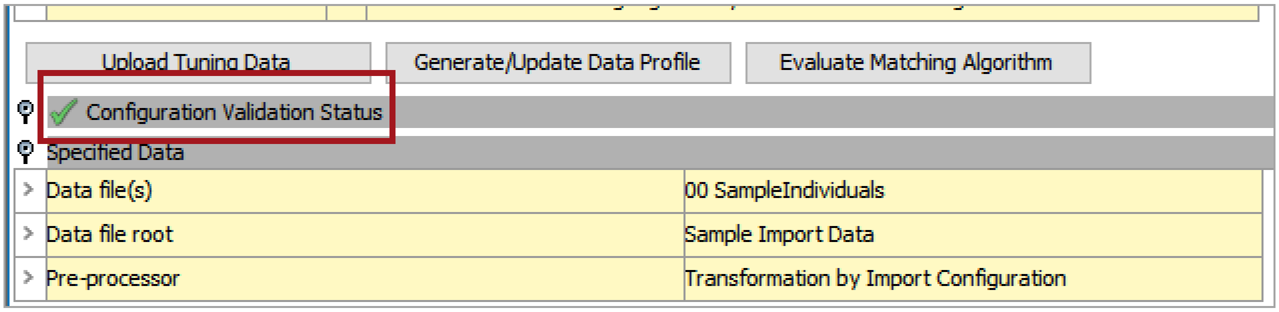
Edit Data Specification

Specified Matching Information

Queue for profiling	MTCProf
Number of threads used for profiling	2
Queue for matching algorithm evaluation	MTCMatch
Number of threads used for matching algorithm evaluation	2
Matching algorithm	Individual Matching
Minimum object count for match code groups	20
Maximum number of match code groups	100
Match interval to export	70 - 100 %
Pairs per percent	10
Attributes to export	
Export match details	true

Edit Matching Information

4. In the 'Configuration Validation Status' flipper, a check mark will appear if the configuration is valid. If it is not, an 'X' will appear and error information will be provided.


Upload Tuning Data
Generate/Update Data Profile
Evaluate Matching Algorithm
Configuration Validation Status
Specified Data

Data file(s)	00 SampleIndividuals
Data file root	Sample Import Data
Pre-processor	Transformation by Import Configuration

5. Before tuning data can be profiled, an asset object type must be created to store the data. Additionally, this asset object type must be mapped to the 'Match Tuning Asset Object Type' parameter in the Matching - Merge Golden Record component model.

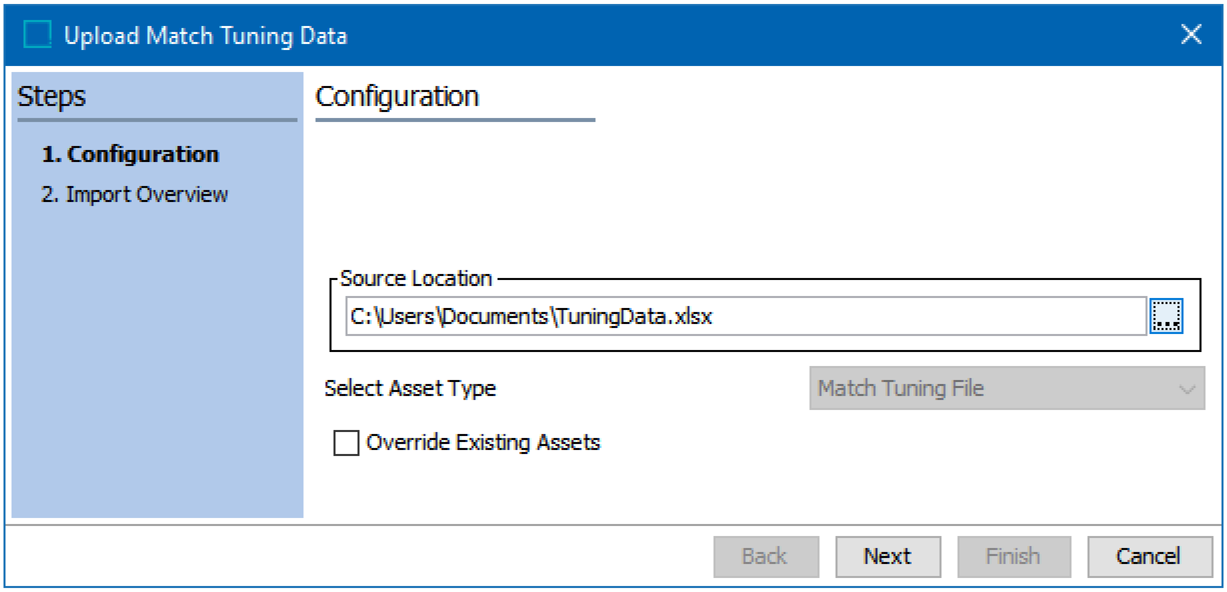
   For more information, see the **Matching Component Model** section of the **Match Criteria Configuration** topic in this documentation.

   A root file must also be configured to store the tuning data. Navigate to the 'Specified Data' flipper on the 'Match Tuning Configuration' tab and click 'Edit.' In the 'Specified Data' dialog, under the 'Data file root' parameter, click the ellipsis button (...) to specify the root file where the tuning data is stored. For more information on this dialog, see Step 7 below.

   > **Important:** This asset object type must have its Reference Target Lock Policy parameter set to 'Strict.' For more information on this parameter, see the **Reference Target Lock Policy on Object Types** topic of the **System Setup / Super User Guide** documentation.

6. Once an asset object type has been established, tuning data can be uploaded. To do so, click the 'Upload Tuning Data' button on the 'Match Tuning Configuration' tab. In the wizard that displays, navigate to the 'Source Location' parameter and click the ellipsis button (...) to specify a data file.

   In the 'Select Asset Type' parameter, specify the desired asset type among those configured in the previous step. Check the 'Override Existing Assets' box if previously uploaded tuning data should be overwritten.


Upload Match Tuning Data

Steps

1. **Configuration**
2. Import Overview

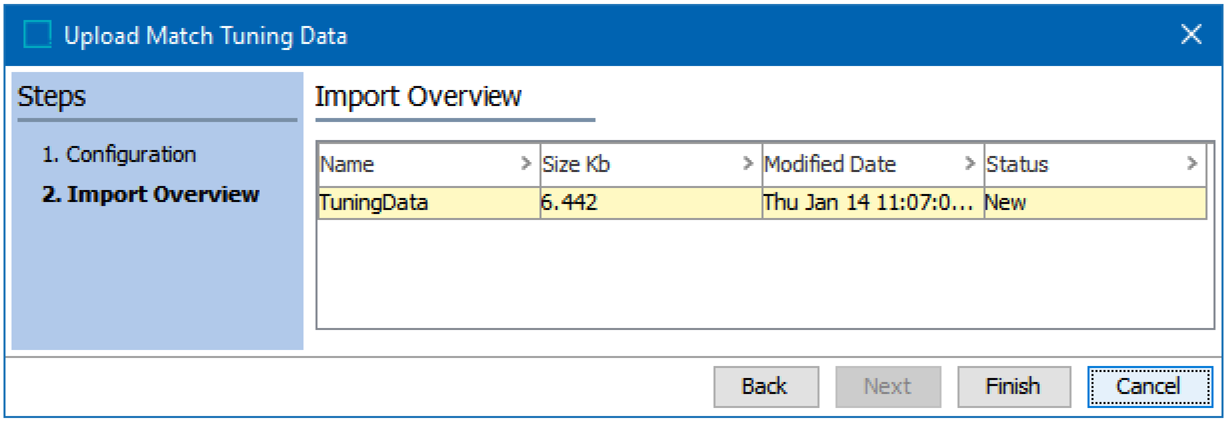
Configuration

Source Location: C:\Users\Documents\TuningData.xlsx

Select Asset Type: Match Tuning File

□ Override Existing Assets

If desired, click 'Next' to view the Import Overview, then click 'Finish' to upload the tuning data.


Upload Match Tuning Data

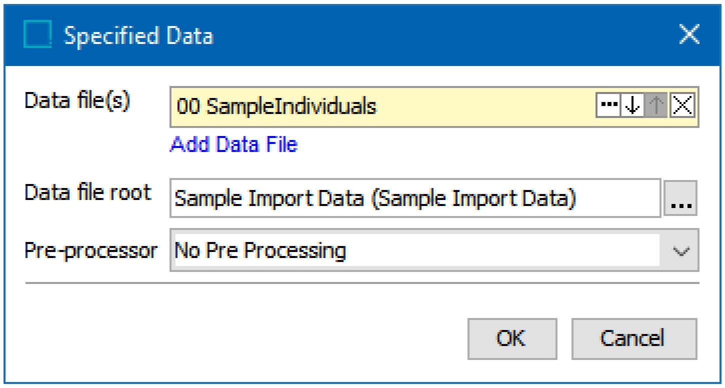
Steps

1. Configuration
2. **Import Overview**

Import Overview

Name	Size Kb	Modified Date	Status
TuningData	6.442	Thu Jan 14 11:07:0...	New

7. To specify tuning data to profile, navigate to the 'Specified Data' flipper and click the 'Edit Data Specification' link. Click the 'Add Data File' link and select one or more data files among those uploaded in the previous step which reside under the specified root. If the file is not a STEPXML, a pre-processor can be used to convert the data.



Specified Data

Data file(s): 00 SampleIndividuals

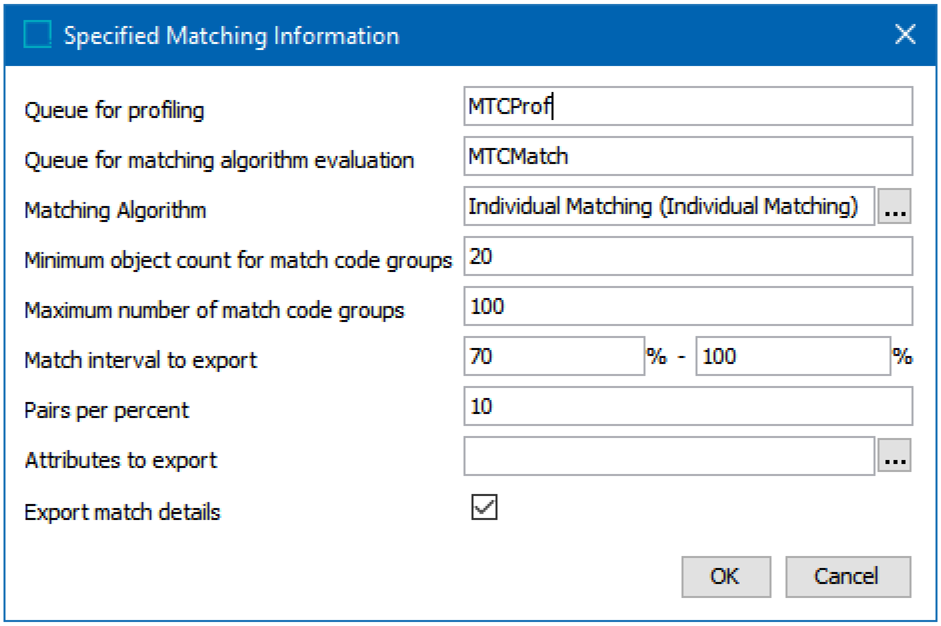
Add Data File

Data file root: Sample Import Data (Sample Import Data)

Pre-processor: No Pre Processing

For more information on converting the CSV / Excel files in this way, see the **IIEP - Configure Match and Merge Importer Processing Engine** topic of the **Data Exchange** documentation.

8. Before generating the data profile, the 'Data Profile' tab should be configured. For more information on configuring a data profile, see the **Data Profiles** topic of the **Data Profiling** documentation.
9. Click the 'Generate / Update Data Profile' button on the 'Match Tuning Configuration' tab to generate the data profile.
10. To specify matching algorithm details, navigate to the 'Specified Matching Information' flipper and click 'Edit.' In the 'Specified Matching Information' dialog, fill out the relevant parameters:



Specified Matching Information

Queue for profiling	MTCProf
Queue for matching algorithm evaluation	MTCMatch
Matching Algorithm	Individual Matching (Individual Matching)
Minimum object count for match code groups	20
Maximum number of match code groups	100
Match interval to export	70 % - 100 %
Pairs per percent	10
Attributes to export	
Export match details	☑

- **Queue for profiling -** The background process queue created for profiling.
- **Queue for matching algorithm evaluation -** The background process queue created for matching algorithm evaluation.
- **Matching Algorithm -** Click the ellipsis button (...) and browse or search for the matching algorithm the match tuning configuration should test.

- **Minimum object count for match code groups** - Enter the minimum number of objects to be exported per match code group.
- **Maximum number of match code groups** - Enter the maximum amount of match code groups the tuning data can generate.
- **Match interval to export** - Specify an interval that includes pairs expected to be both matches and non-matches, as well as pairs that are not clear matches or non-matches. Only pairs with scores within this interval are exported.
- **Pairs per percent** - Enter the maximum number of pairs to be exported for each percentage point.
- **Attribute to export** - Click the ellipsis button (...) and select the attribute values that should be exported.
- **Export match details** - Check the box to add additional columns with part scores from decision table comparators and sub decision tables.

11. Once Specified Matching Information has been configured, click the Evaluate Matching Algorithm button on the Match Tuning Configuration tab. This will start a background process that creates a pair export file and match codes export file.

Find Similar

Before creating new objects, matching algorithms can be used to search for similar existing objects in STEP, ensuring duplicates are not created.

Matching logic can be applied to three different 'Search Before Create' methods:

- The 'Duplicate Handler' on an 'Initiate Item' workflow screen in Web UI. For more information, see the **Initiate Item Screen** topic in **Web User Interfaces / Web UI Getting Started** documentation.
- The 'Find Similar' search on an 'Add Reference' action in Web UI. For more information, see the **Add Reference Action** topic in the **Web User Interfaces / Web UI Getting Started** documentation.
- 'GetSimilarObject' SOAP web service request. For more information, see the **getSimiliarObjects** topic in the **CMDM Solution Enablement** documentation.

The key to Find Similar functionality is the matching setup that the customer creates and uses for duplicate handling. Every time a user enters data into the search fields and clicks OK, the Find Similar search checks the match code values involved, executes the relevant matching algorithm, and provides a set of results, if any are found. If a user is not getting the expected results, one area to assess is the algorithm configured in the 'Duplicate Handler' parameter in the 'Add Reference Action' properties. Two bind types work with the Find Similar functionality:

1. First Match Object
2. Second Match Object

A relevant match code and matching algorithm needs to be set up before attempting to use the Find Similar Search tab. For more information about setting up and using matching algorithms, see the **Configuring Matching Algorithms** topic of this documentation.

The matching logic is applied by comparing potential new objects with that of existing objects. More specifically, match codes are generated for the incoming objects, compared to existing objects with similar match codes, and if matches are found, a list is returned of all matched objects with rank scores that met or exceeded the configured threshold in the request. Using this list, the user can decide whether to create a new object or use an existing one.

Find Similar in Web UI

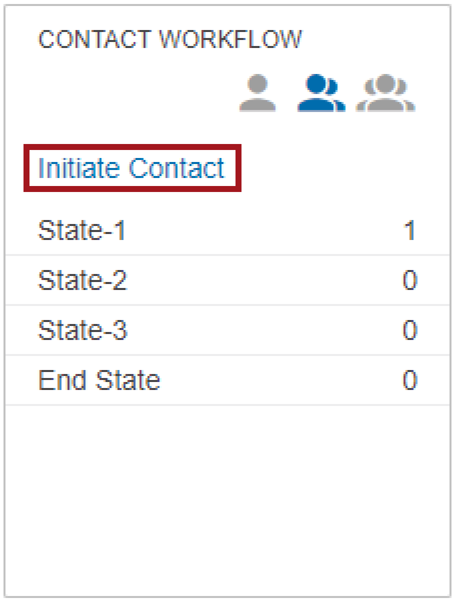
To avoid users creating duplicate objects in the STEP system, users can use the 'Find Similar Action' to identify similar objects prior to initiating new ones, when working in Initiate Item screens in Web UI. Also, users can use 'Find Similar' functionality to search for and identify similar objects prior to adding references and creating target objects within a Multi-Reference Editor.

Using Find Similar on an Initiate Item Screen

This example shows how Find Similar is used on an 'Initiate Item' screen, including a 'Store Single Referenced Target' component.

The 'Store Single Referenced Target' component is intended to find a similar record on 'Initiate Item' workflow state. The component can be used to create a single reference target with attributes and create a reference to it from the matched record.

For example, a user creates a Contact object by clicking 'Initiate Contact' on the 'Status Selector Homepage Widget.'

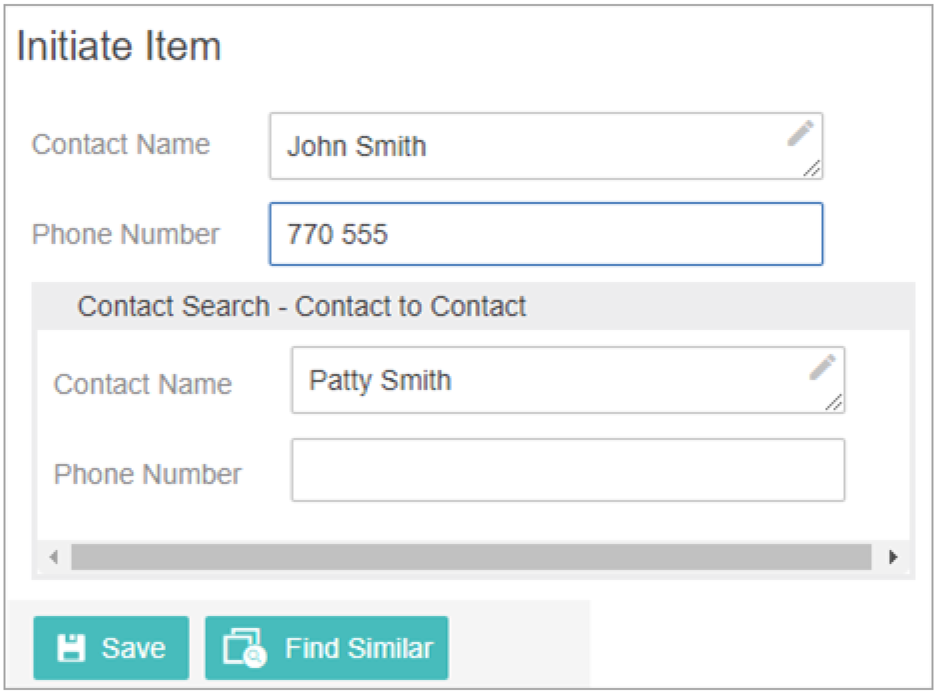


CONTACT WORKFLOW

Initiate Contact

State-1	1
State-2	0
State-3	0
End State	0

On the Initiate Screen, the user enters data into the direct object search fields and into the referenced object search fields and then clicks the 'Find Similar' button.



Initiate Item

Contact Name: John Smith

Phone Number: 770 555

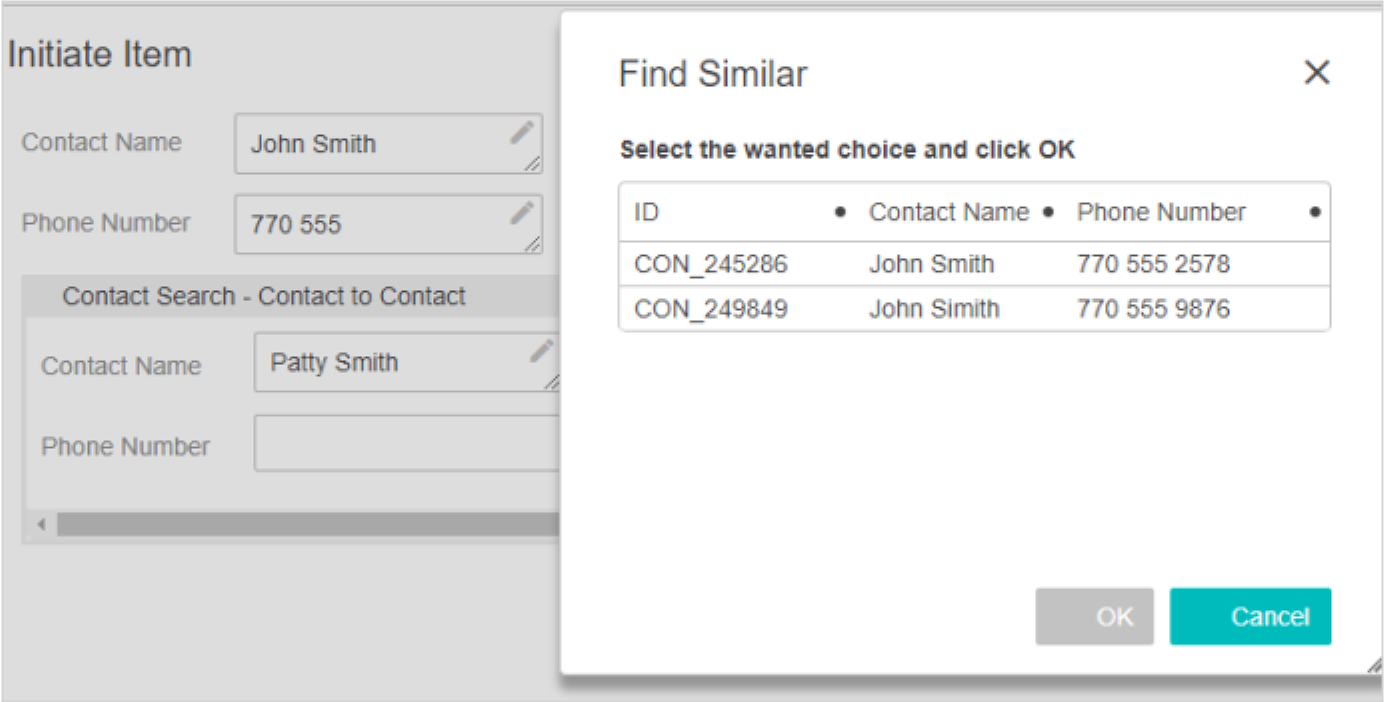
Contact Search - Contact to Contact

Contact Name: Patty Smith

Phone Number:

The algorithm runs in the background, and the configured Dialog List Screen displays. Be aware that this is not a standard search, and the results are based on the matching algorithm running in the background.

Important: As stated at the beginning of this topic, the key to Find Similar functionality is the matching setup that the customer creates and uses for duplicate handling. This is important because the attributes being searched must be part of the list of attributes that match codes are generated for; if not, the search will not work as expected.



Initiate Item

Contact Name	John Smith
Phone Number	770 555

Contact Search - Contact to Contact

Contact Name	Patty Smith
Phone Number	

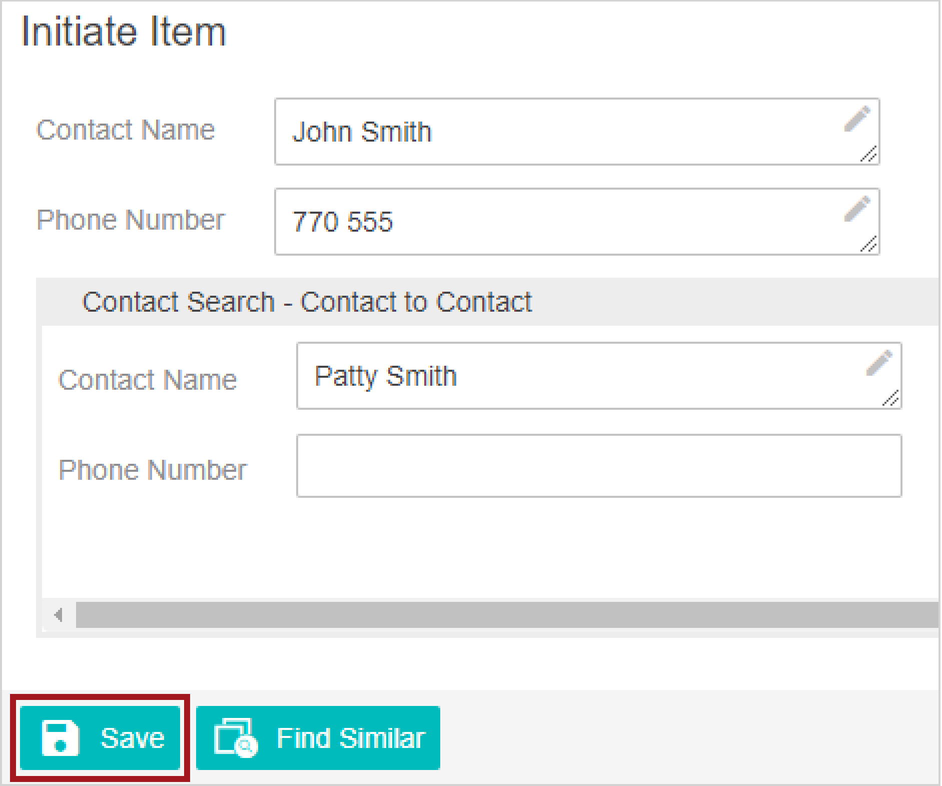
Find Similar

Select the wanted choice and click OK

ID	Contact Name	Phone Number
CON_245286	John Smith	770 555 2578
CON_249849	John Simith	770 555 9876

A maximum of fifty (50) objects are shown at one time on the results list. The user can select one of the objects shown in the results list. After choosing an item, select the OK button. Clicking OK takes the user directly to that object via the appropriate Web UI screens.

If the user wants to create a new object with the search data entered, the user clicks Cancel to exit the results list. On the Initiate Screen, the user will click Save and be taken directly to the new object. The data entered into the attribute value fields, which are configured in the Store Single ReferencedTarget Properties, is also used to generate the referenced object and the reference between these objects is also created.

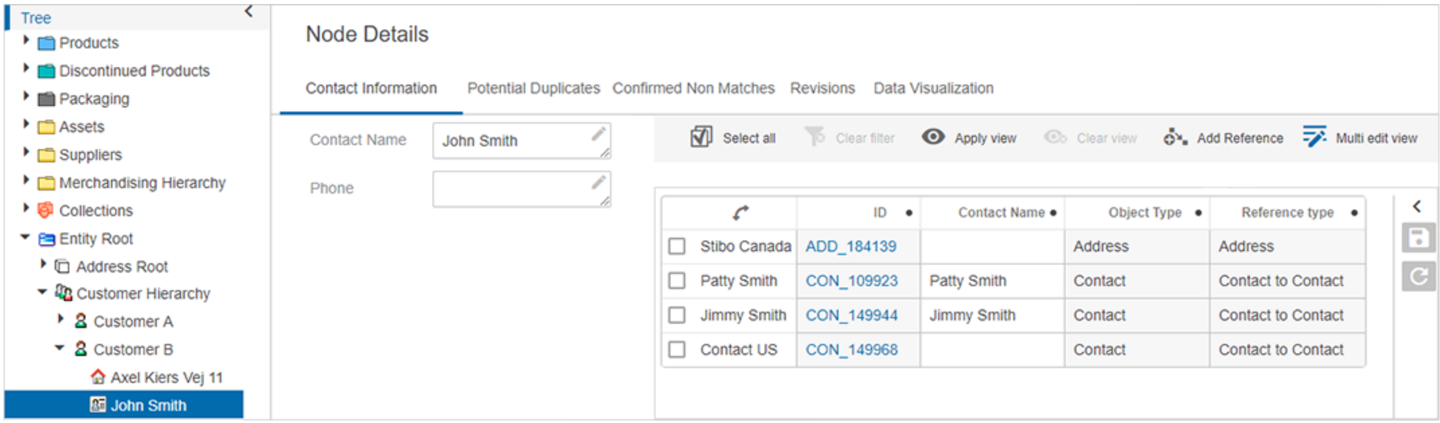


Initiate Item

Contact Name	John Smith
Phone Number	770 555

Contact Search - Contact to Contact

Contact Name	Patty Smith
Phone Number	



Node Details

Contact Information | Potential Duplicates | Confirmed Non Matches | Revisions | Data Visualization

Contact Name	John Smith
Phone	

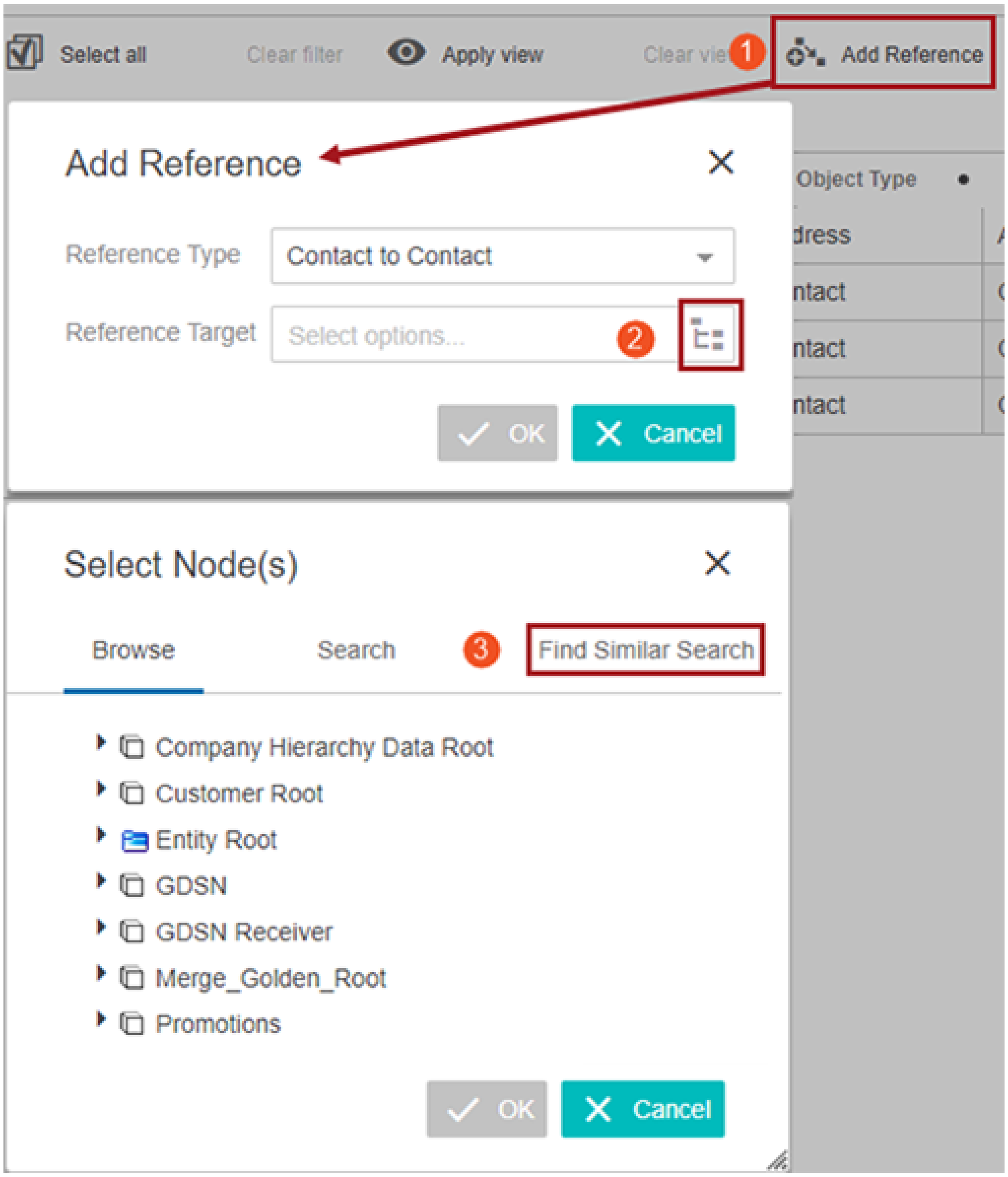
		ID	Contact Name	Object Type	Reference type
□	Stibo Canada	ADD_184139		Address	Address
□	Patty Smith	CON_109923	Patty Smith	Contact	Contact to Contact
□	Jimmy Smith	CON_149944	Jimmy Smith	Contact	Contact to Contact
□	Contact US	CON_149968		Contact	Contact to Contact

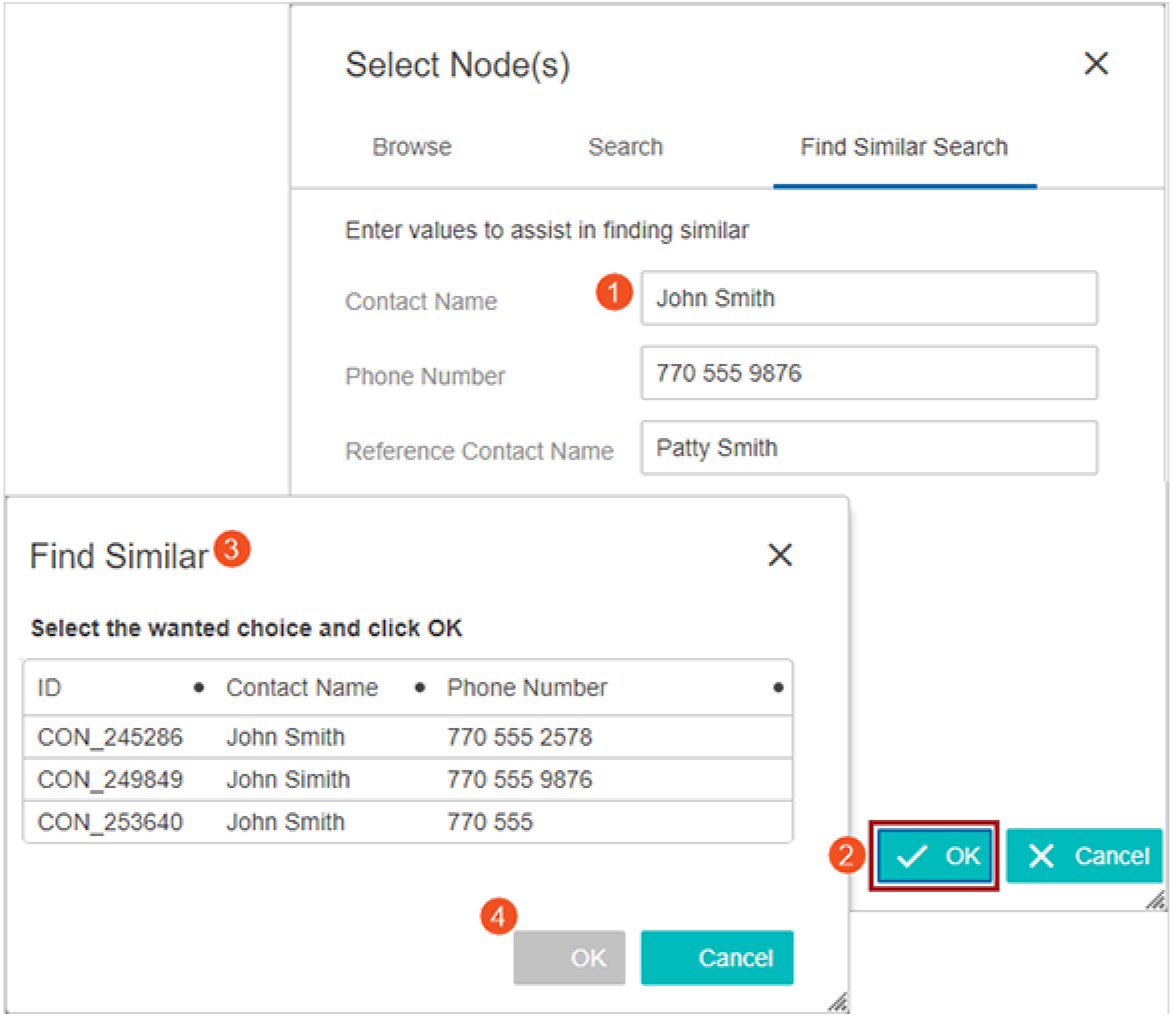
If applicable, the new object will automatically be initiated into a workflow or workflows based on existing workflow rules. For more information about workflows and auto-initiation, see the **Auto-Initiation of Tasks in Workflows** topic in the **Workflows** documentation. It is advisable to also make use of standard STEP tools like business rules and workflows to manage the referenced objects that are created in this manner.

Using Find Similar on an Add Reference Action

The following section details an example of a configured 'Find Similar Search' tab.

While using a 'Multi-Reference Editor' component, click 'Add Reference,' and then, click the node picker icon on the 'Add reference' dialog that appeared. The Find Similar Search tab is displayed in the 'Select Node(s)' dialog. The end user enters data into the configured attribute fields and clicks OK. The algorithm runs in the background and the configured Dialog List Screen displays. Remember that this is not a standard search and results are based on the matching algorithm running in the background.





Enter values to assist in finding similar

Contact Name: John Smith

Phone Number: 770 555 9876

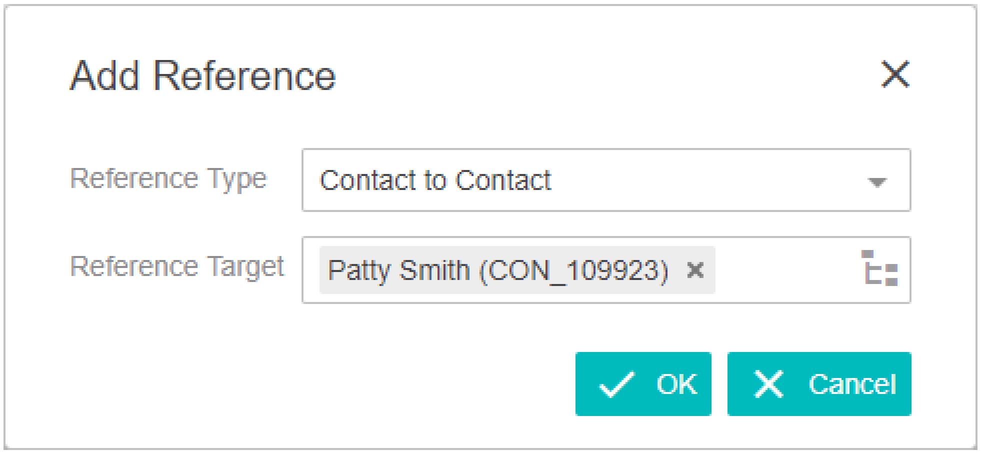
Reference Contact Name: Patty Smith

Select the wanted choice and click OK

ID	Contact Name	Phone Number
CON_245286	John Smith	770 555 2578
CON_249849	John Simith	770 555 9876
CON_253640	John Smith	770 555

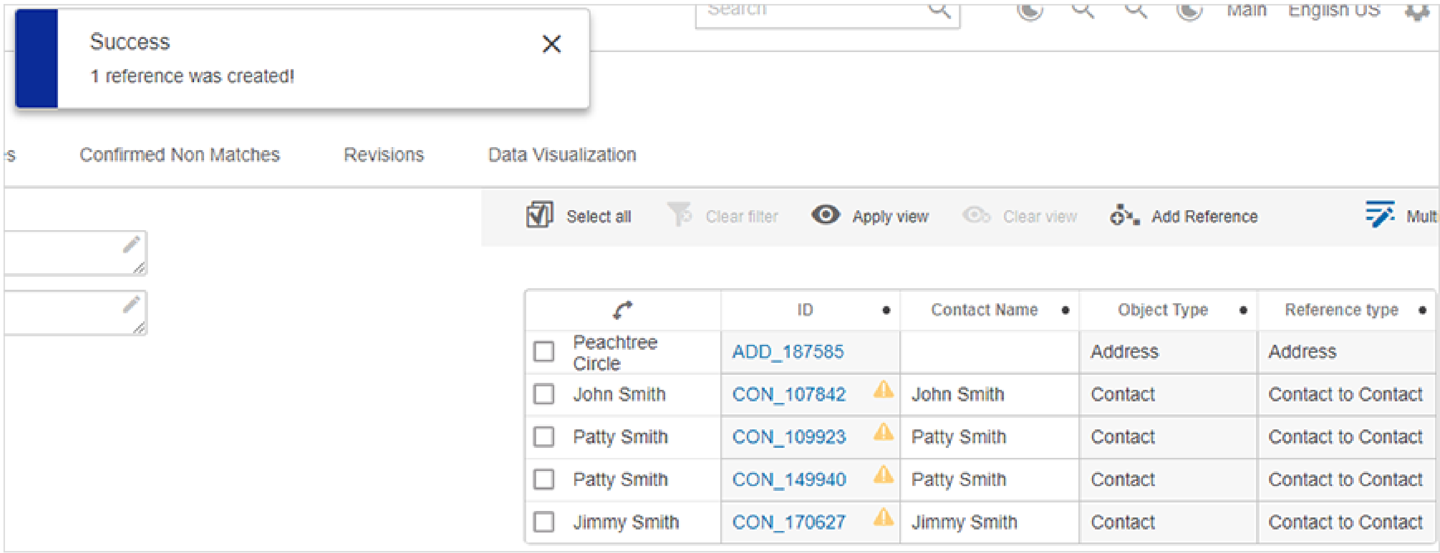
Important: As stated previously in this topic, the key to Find Similar functionality is the matching setup that the customer creates and uses for duplicate handling. This is important because the attributes being searched must be part of the list of attributes that match codes are generated for. If not, the search will not work as expected.

A maximum of fifty (50) objects are shown at one time on the results list. If the end user finds a reference on the results list that they want to use, they click in the row they want. When that selection is made, the OK button will be available. The end users clicks 'OK' and now that reference is saved. If the user does not find a result to use, they can click cancel and create a new reference using the Create or Create from Template functions (both described previously in this topic).



Add Reference

Reference Type	Contact to Contact
Reference Target	Patty Smith (CON_109923) ×



Success

1 reference was created!

	ID	Contact Name	Object Type	Reference type
□ Peachtree Circle	ADD_187585		Address	Address
□ John Smith	CON_107842 ⚠	John Smith	Contact	Contact to Contact
□ Patty Smith	CON_109923 ⚠	Patty Smith	Contact	Contact to Contact
□ Patty Smith	CON_149940 ⚠	Patty Smith	Contact	Contact to Contact
□ Jimmy Smith	CON_170627 ⚠	Jimmy Smith	Contact	Contact to Contact

Note: Find Similar functionality is also available when working with company data hierarchies via the Hierarchy Representations component (found in Main (Screen) Properties). For more information on the setup, which is similar to Add Reference, see the **Optional Configurations for the Hierarchy Representation Component** topic in the **Web User Interfaces / Web UI Getting Started** documentation.

Configure Match Codes and Matching Algorithm for Use with Find Similar

When configured to work with getSimilarObject web service, special considerations should be made for the solution's match codes and matching algorithm.

Important: A getSimilarObject request can only return ninety-nine results at a time.

Match Codes

Attribute value binds in the match code definition should be created specifically for getSimilarObject cases.

For example:

```
if ( node == null ) {

   //generate matchcodes for getSimilarObject

}

else {

   //generate matchcodes for existing objects

}
```

> **Note:** When used for getSimilarObject, it is safe to establish large match code groups without impacting performance.

For more information, see the **Match Codes** topic of the documentation.

Matching Algorithm

Matching algorithm global binds should be configured to map the attributes used in the SOAP request. 'Mcevaluate' / 'evaluate' should be used in the match criterion's STEP function / JavaScript function to retrieve these values when the current object returns null.

> **Note:** It is recommended practice to use the decision table match criteria for this purpose.

When decision tables are used as the match criteria, any configured party data normalizers require that all configured attributes also exist as global binds. This applies to both explicitly configured attributes and for those configured via component models.

So, for example, if the Address Normalizer is used for getSimilarObjects, the country, region, city, postcode, and street attributes must have corresponding global binds.

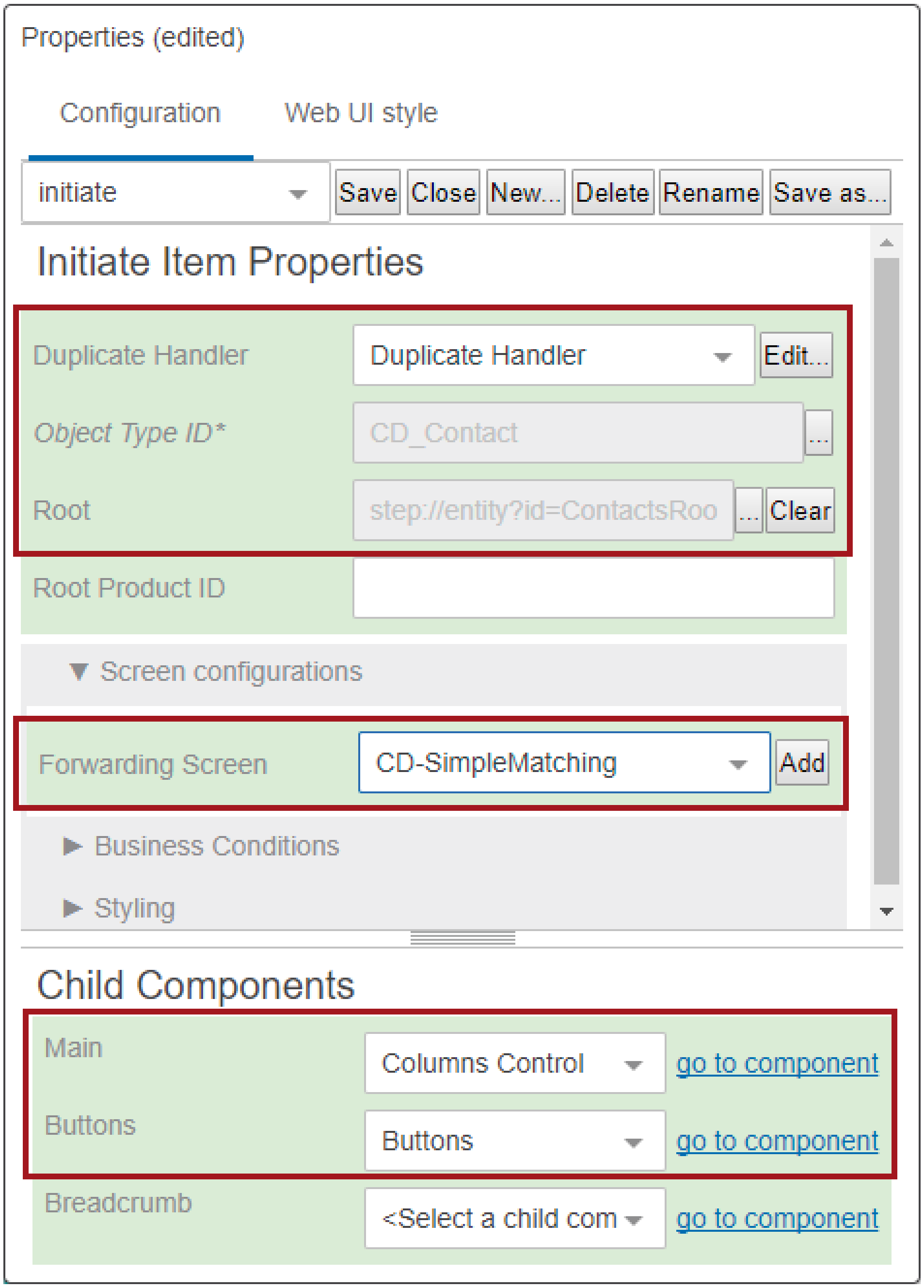
> **Important:** For customer data normalizers, the name of the global bind must match the ID of the corresponding attribute.

Customer data normalizers are compatible with getSimilarObject so long as global binds are set on the matching algorithm. For more information on global binds and match criteria, see the **Configuring Matching Algorithms** topic of the documentation.

Configuring Find Similar on an Initiate Item Screen

Prerequisites

All the steps provided in this topic assume the Web UI designer is in design mode and on the applicable Properties screen prior to starting the configuration process. It is also assumed that all users have the proper privilege to work with these features. For more information about privileges and user setup, see the **Users and Groups** topic and **Adding User Privileges for a Group** topic of the **System Setup / Super User Guide** documentation.



Properties (edited)

Initiate Item Properties

Duplicate Handler	Duplicate Handler
*Object Type ID**	CD_Contact
Root	step://entity?id=ContactsRoo
Root Product ID	

▼ Screen configurations

Forwarding Screen	CD-SimpleMatching

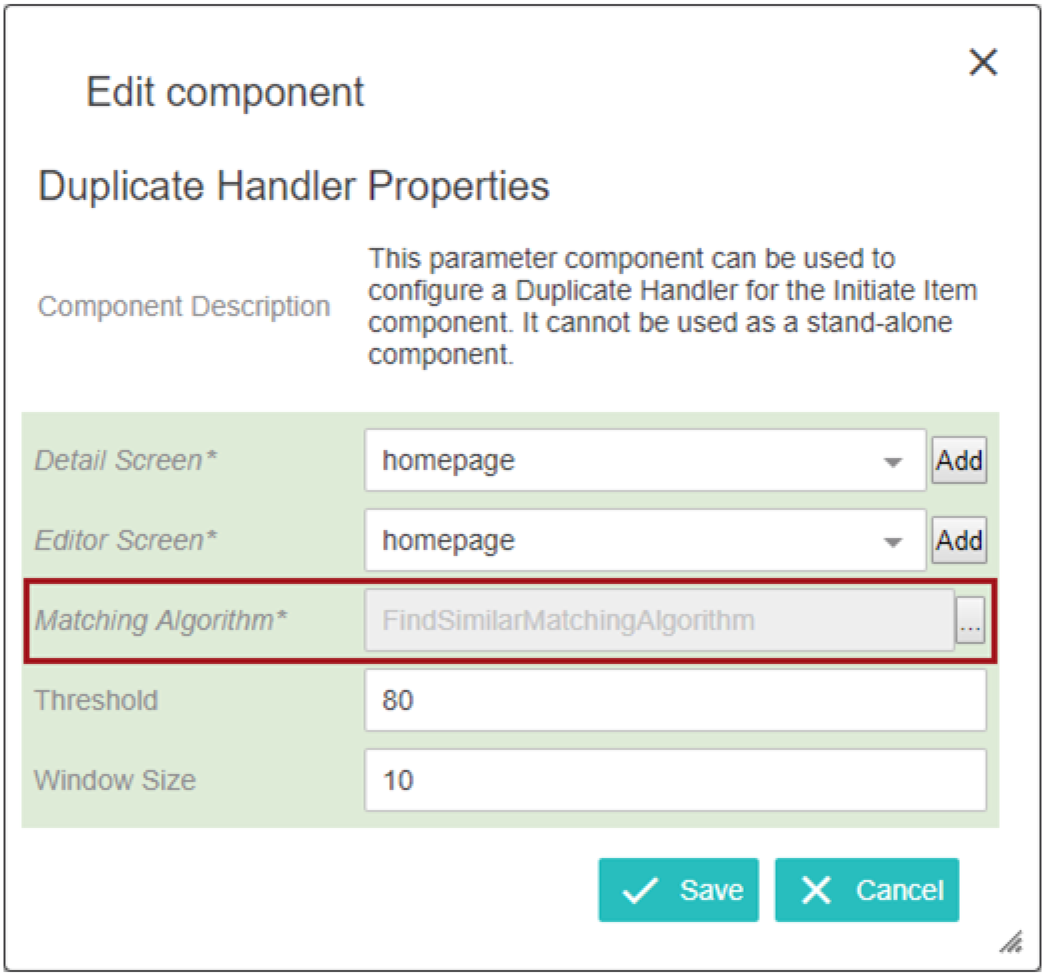
Child Components

Main	Columns Control	go to component
Buttons	Buttons	go to component
Breadcrumb	<Select a child com	go to component

1. Create a new 'Initiate Item' screen or use an existing one.
2. In 'Initiate Item Properties,' select 'Duplicate Handler' in the dropdown for the 'Duplicate Handler' parameter.

   The Duplicate Handler Properties screen will display. The only required setting is the matching algorithm parameter.

1. Click the ellipsis button (...) to the right of the value field and select a matching algorithm. Click Save.



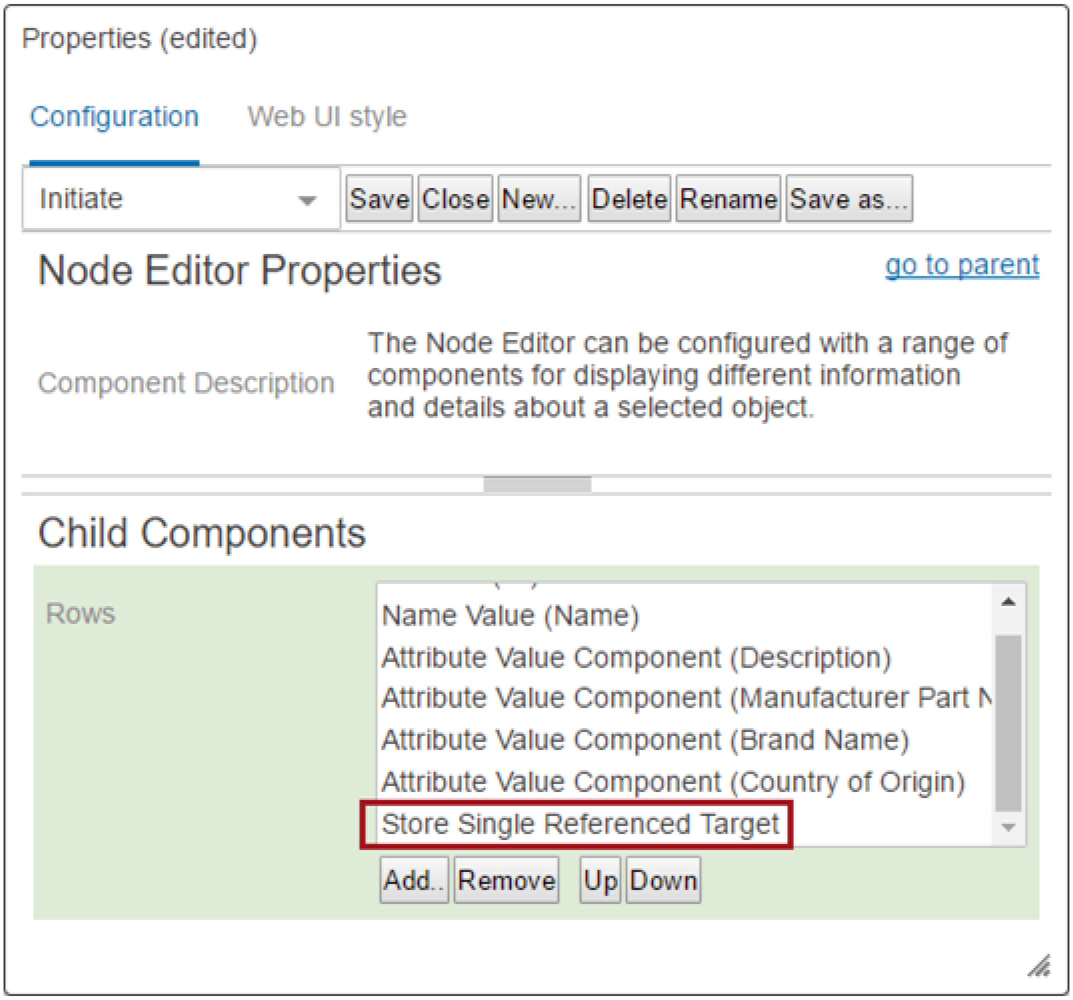
Edit component

Duplicate Handler Properties

Component Description	This parameter component can be used to configure a Duplicate Handler for the Initiate Item component. It cannot be used as a stand-alone component.
*Detail Screen**	homepage
*Editor Screen**	homepage
*Matching Algorithm**	FindSimilarMatchingAlgorithm
Threshold	80
Window Size	10

2. On the Initiate Item properties dialog, if the default Object Type ID is not the correct option, click the ellipsis button (...) to the right of the Object Type ID parameter to select the object type of the objects that will be initiated into the system.
3. Click the ellipsis button (...) to select a Root directory. New items will be created below the designated root folder. A default value will be populated.
4. Select a 'Forwarding Screen' option. This screen is what is shown once an object is created.
5. For the 'Main' child components option, choose 'Columns Control' from the dropdown. Click on 'go to component' to configure this option. Select 'Node Editor' in the Columns Control Properties.

6. Double click the Node Editor component to go to the Node Editor Properties dialog. On this dialog, the search attributes need to be configured within the Child Components > Rows value field using the add and remove buttons. These search attributes are for searching direct objects.



Properties (edited)

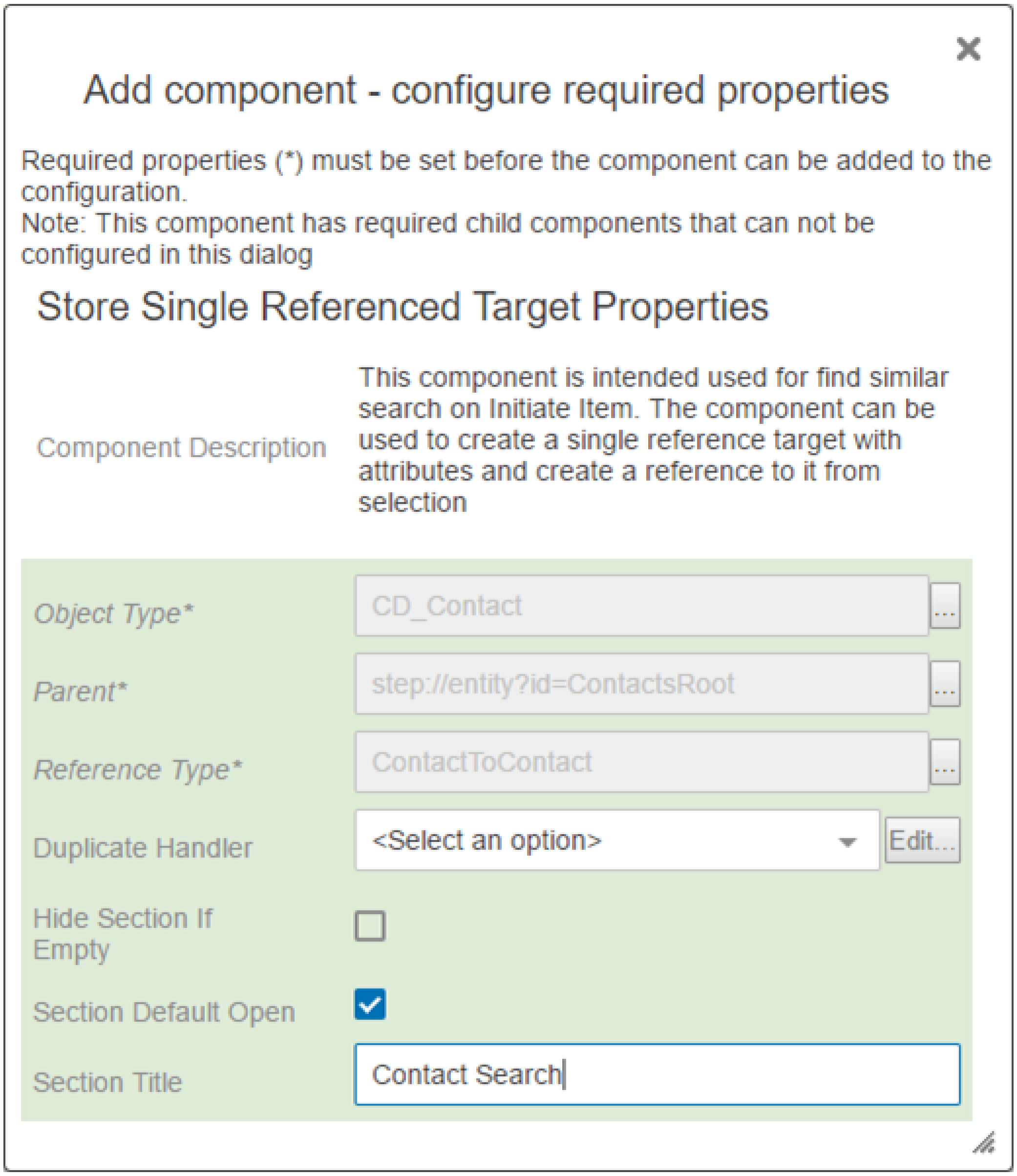
Node Editor Properties

Component Description: The Node Editor can be configured with a range of components for displaying different information and details about a selected object.

Child Components

Rows:
Name Value (Name)
Attribute Value Component (Description)
Attribute Value Component (Manufacturer Part N
Attribute Value Component (Brand Name)
Attribute Value Component (Country of Origin)
Store Single Referenced Target

7. If the use case requires searching on referenced objects in addition to direct objects, users must add the 'Store Single Referenced Target' component. Double click on 'Store Single Referenced Target' to configure the component. The 'Object Type,' 'Parent,' 'Reference Type,' and 'Child Component > Rows' are required parameters / components. Go through each parameter one-by-one to complete the configuration.



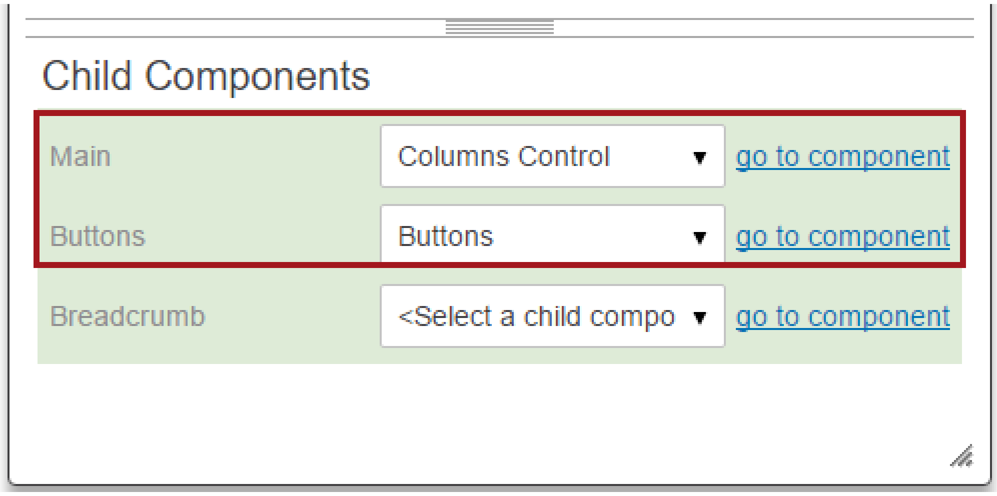
Add component - configure required properties

Required properties (*) must be set before the component can be added to the configuration.
Note: This component has required child components that can not be configured in this dialog

Store Single Referenced Target Properties

Component Description	This component is intended used for find similar search on Initiate Item. The component can be used to create a single reference target with attributes and create a reference to it from selection
*Object Type**	CD_Contact
*Parent**	step://entity?id=ContactsRoot
*Reference Type**	ContactToContact
Duplicate Handler	<Select an option>
Hide Section If Empty	□
Section Default Open	☑
Section Title	Contact Search

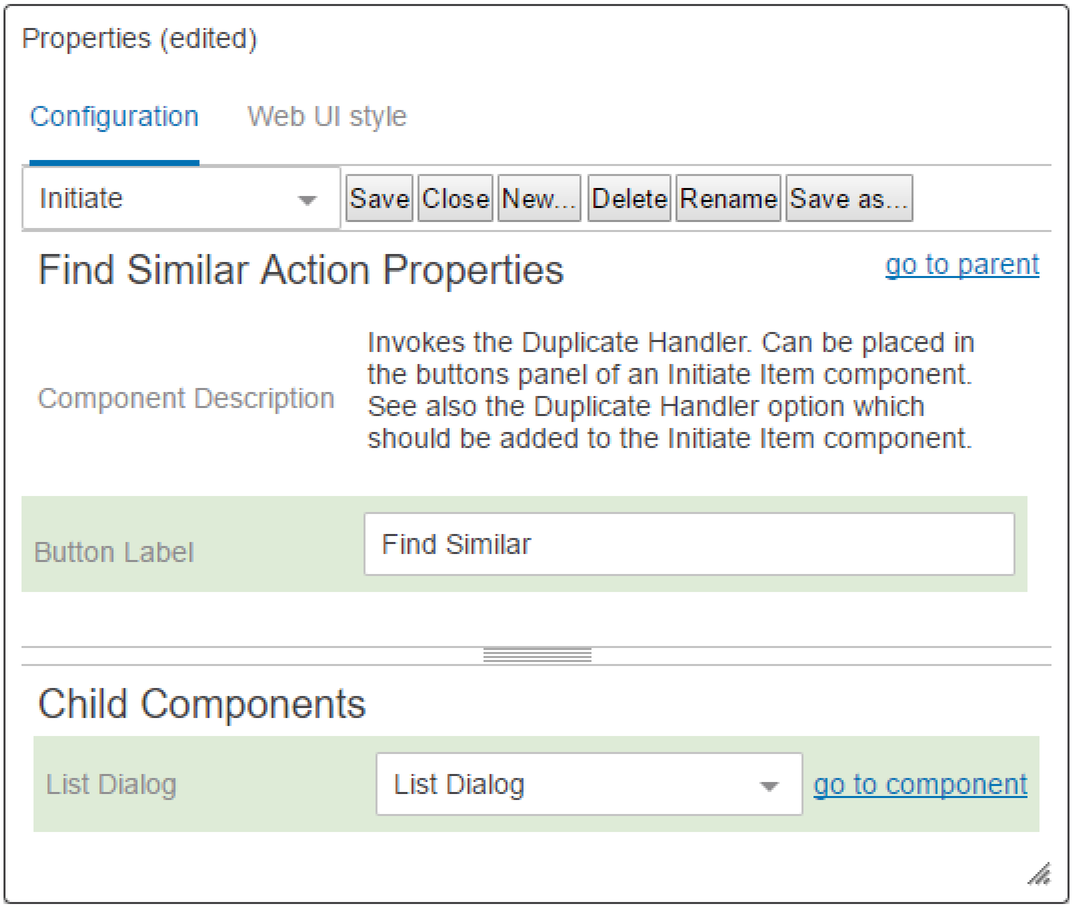
8. On the parent screen's 'Initiate Item Properties' screen, go to 'Child Component > Buttons' and click on 'go to component.' In the 'Actions' value field, select 'Add' and select 'Find Similar Action.'



Child Components

Main	Columns Control	go to component
Buttons	Buttons	go to component
Breadcrumb	<Select a child compo	go to component

9. Double click on the 'Find Similar Action' title and open the 'Find Similar Action Properties' dialog.



Find Similar Action Properties

Component Description: Invokes the Duplicate Handler. Can be placed in the buttons panel of an Initiate Item component. See also the Duplicate Handler option which should be added to the Initiate Item component.

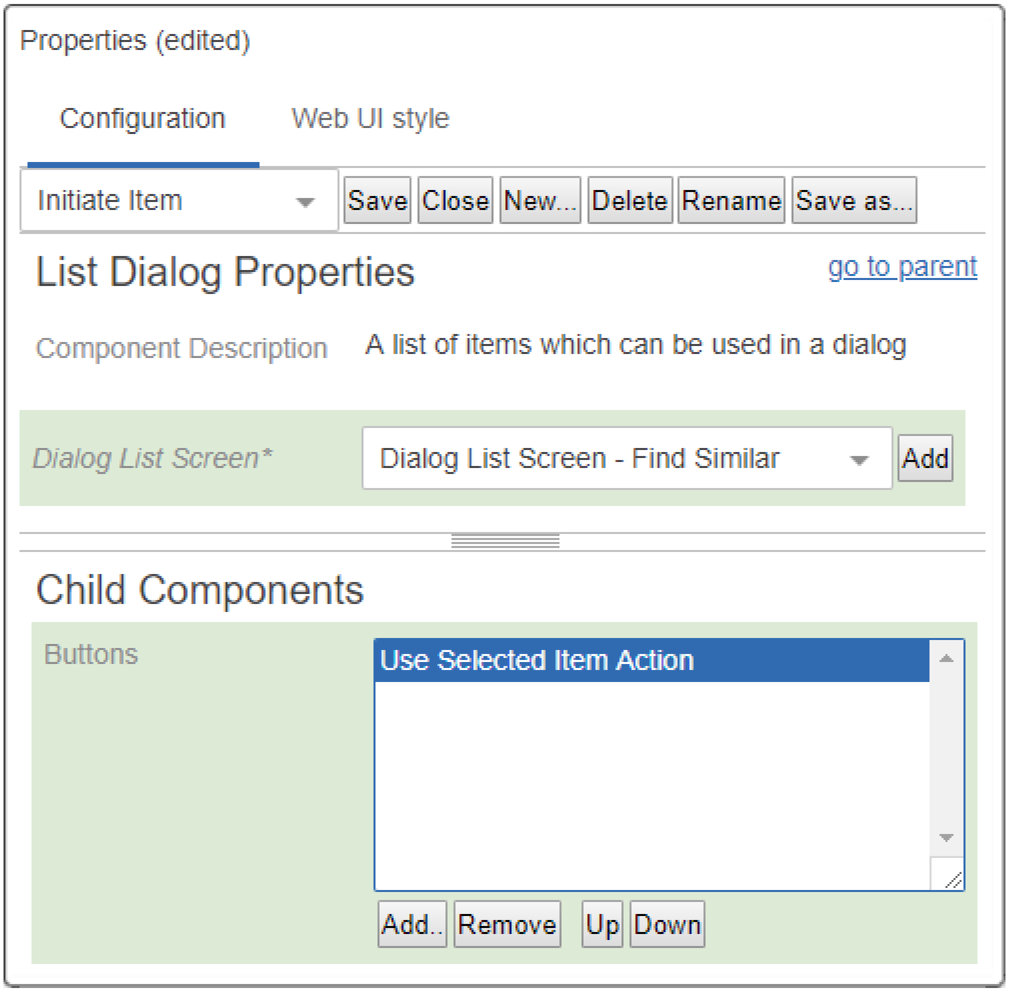
Button Label: Find Similar

Child Components

List Dialog	List Dialog	go to component

10. Edit the Button Label, if desired. Under Child Components > List Dialog, select 'List Dialog' from the dropdown menu.

11. On 'List Dialog' Properties, select a Dialog List screen. If a Dialog List screen does not exist, follow the next two steps to create one.
12. Click Add to the right of the Dialog List screen value field.
13. Create a new screen by selecting Dialog List screen from the list of available screen types / components. Enter a screen ID, and then, click Add. The screen ID will automatically populate in the Dialog List Screen value field.



List Dialog Properties

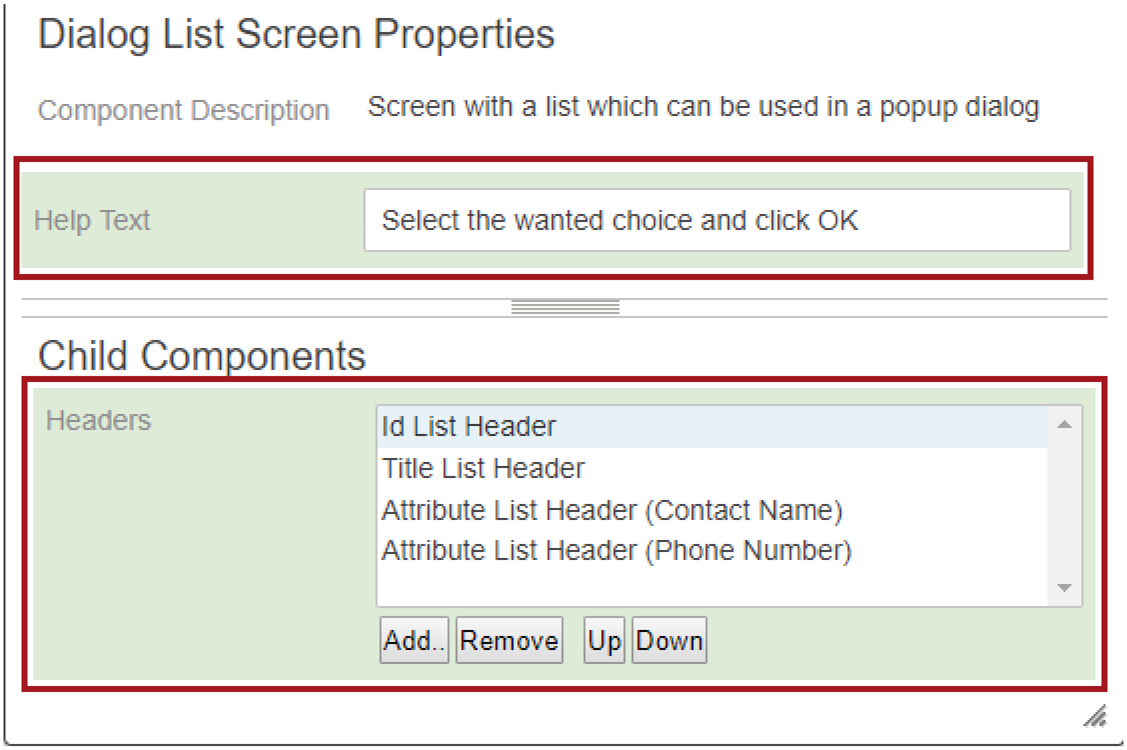
Component Description: A list of items which can be used in a dialog

Dialog List Screen*: Dialog List Screen - Find Similar

Child Components

Buttons: Use Selected Item Action

14. On the List Dialog Properties screen, in the Child Components > Buttons field, click Add and select Use Selected Item Action. Click Add, and then click Save in the designer window.
15. Before exiting design mode, select the Dialog List Screen ID of the screen you just created from the dropdown menu. In the example above, the screen ID is 'Dialog List Screen - Find Similar.'
16. Configure the Child Components > Headers section of the Dialog List Screen Properties. Click Add under the value field to set up the attributes that will display in the Find Similar Search tab for the results list. The Help Text is editable and will display at the top of the results list.



Dialog List Screen Properties

Component Description: Screen with a list which can be used in a popup dialog

Help Text: Select the wanted choice and click OK

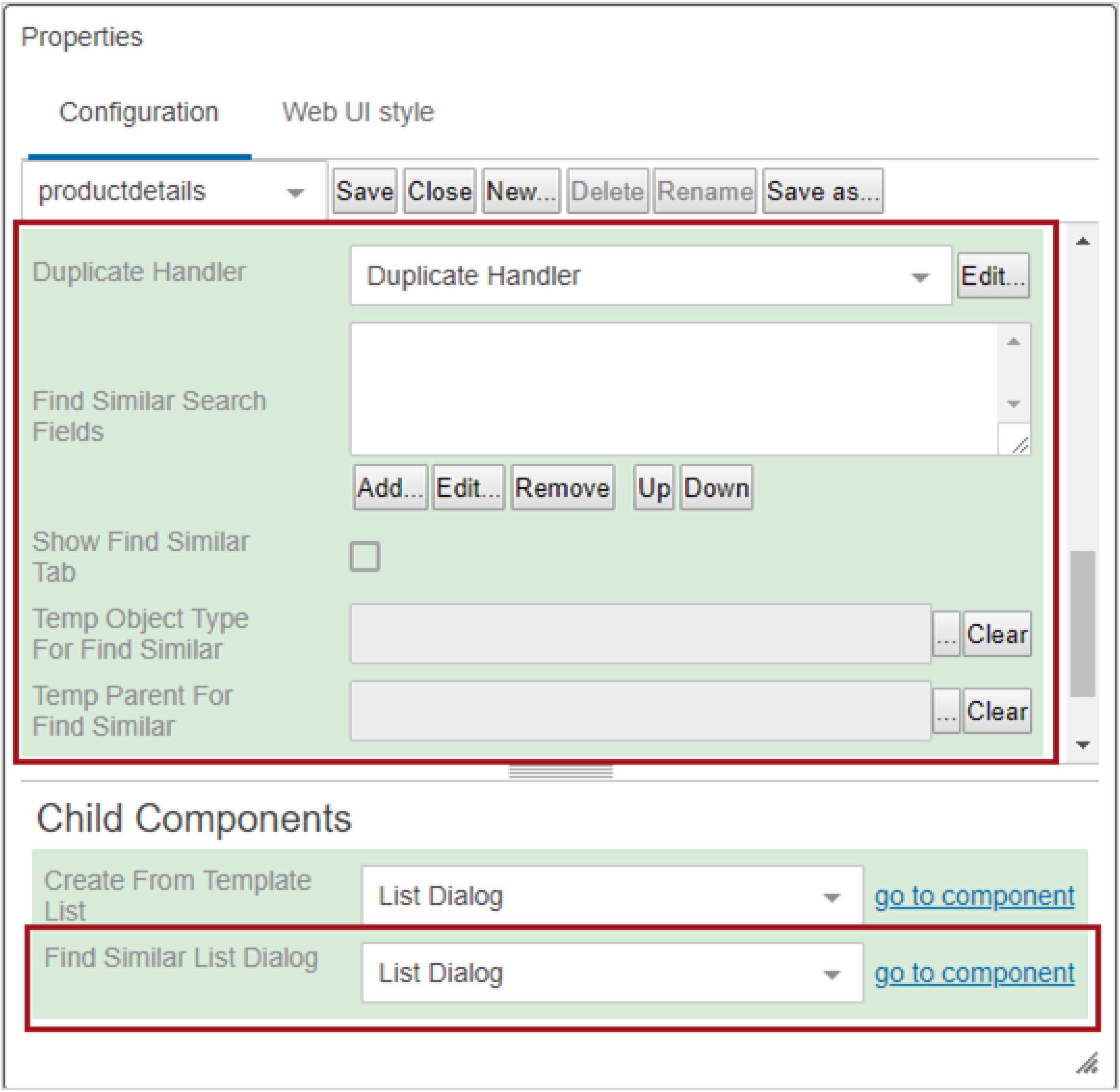
Child Components

Headers:
- Id List Header
- Title List Header
- Attribute List Header (Contact Name)
- Attribute List Header (Phone Number)

17. Click Save and Close design mode to return to normal Web UI mode.

Configuring Find Similar on an Add Reference Action

This setup information is also available in the **Add Reference Action** topic of the **Web User Interfaces / Using a Web UI** documentation.



productdetails

Duplicate Handler: Duplicate Handler

Find Similar Search Fields

Show Find Similar Tab ☐

Temp Object Type For Find Similar

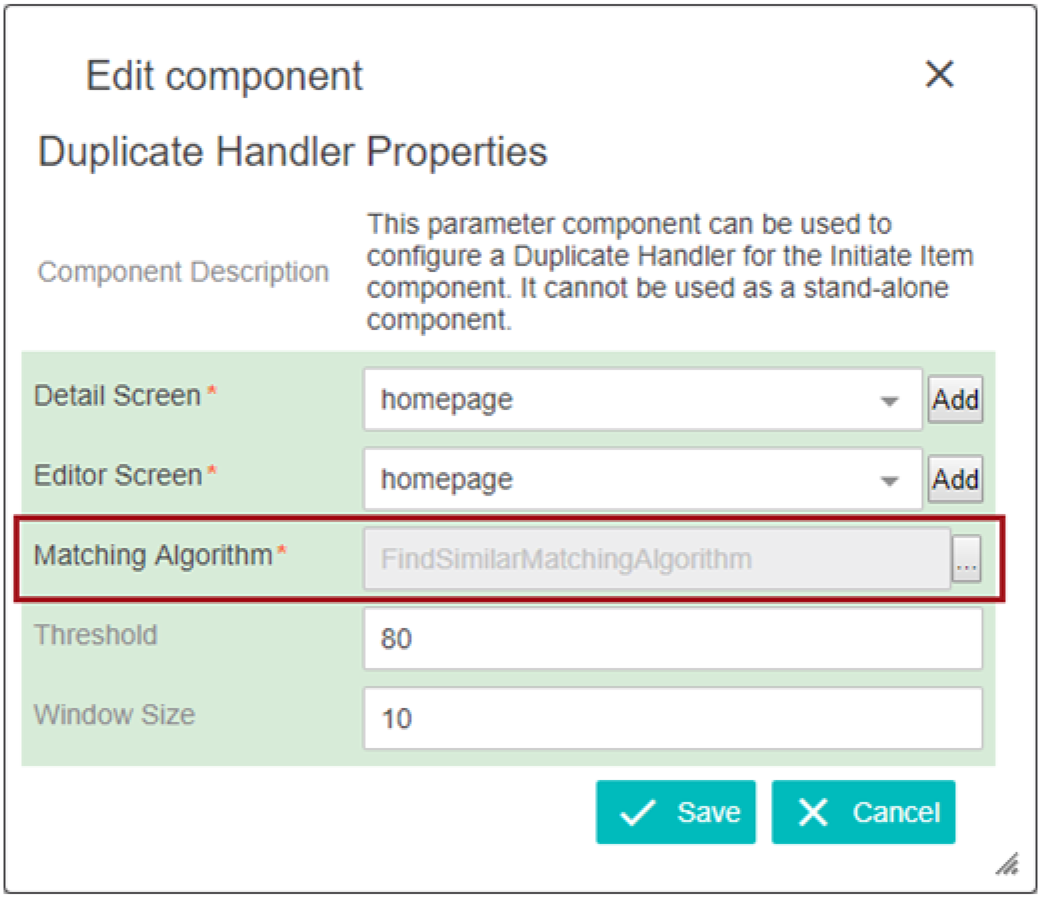
Temp Parent For Find Similar

Child Components

Create From Template List: List Dialog

Find Similar List Dialog: List Dialog

1. Edit the properties for an existing Add Reference Action configured for a Multi-Reference Editor, or add an Add Reference Action and complete the configuration.
2. In the Add Reference Action Properties, select Duplicate Handler in the dropdown for the Duplicate Handler parameter.
3. The Duplicate Handler Properties screen will display. The only required setting is the Matching Algorithm parameter. Click the ellipsis button (...) to the right of the value field and select a matching algorithm. Click Save.

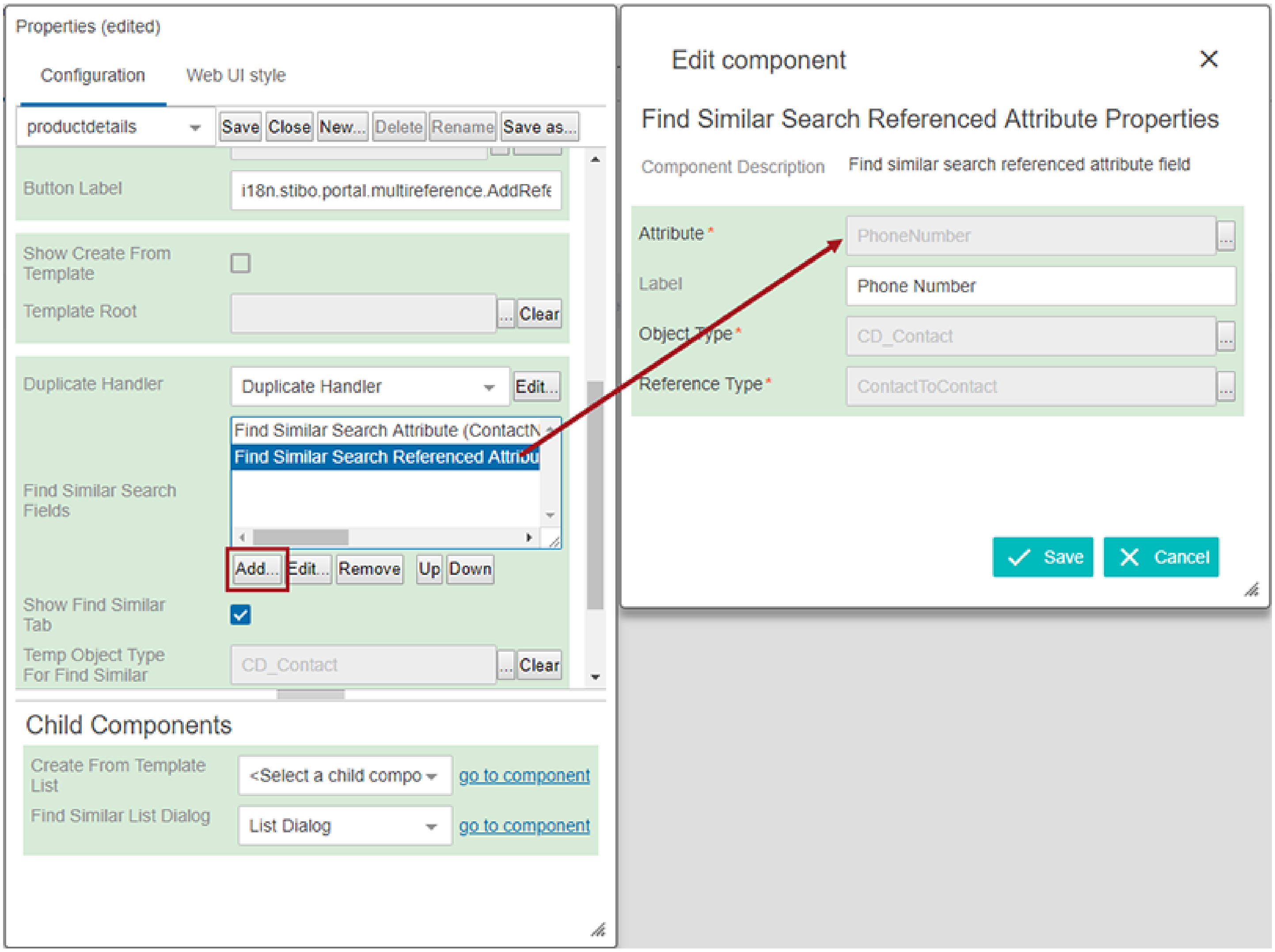


Edit component

Duplicate Handler Properties

Component Description	This parameter component can be used to configure a Duplicate Handler for the Initiate Item component. It cannot be used as a stand-alone component.
Detail Screen *	homepage
Editor Screen *	homepage
Matching Algorithm *	FindSimilarMatchingAlgorithm
Threshold	80
Window Size	10

4. On the 'Add Reference Action Properties' again, click Add under the Find Similar Search Fields value box. Add, remove, and re-order Attribute and Referenced Attributes as desired. Remember that these search fields will need to be incorporated into the matching algorithm configured in the Duplicate Handler parameter.
5. It is important to fill in the Label fields during the Find Similar Search Fields configuration. These labels will appear on the Find Similar Search tab. If adding a Find Similar Search Referenced Attribute, fill in the Attribute, Label, Object Type, and Reference Type values. Save all changes.



productdetails

Button Label: i18n.stibo.portal.multireference.AddRefe

Show Create From Template: □

Template Root:

Duplicate Handler: Duplicate Handler

Find Similar Search Fields: Find Similar Search Attribute (ContactN
Find Similar Search Referenced Attribu

Show Find Similar Tab: ☑

Temp Object Type For Find Similar: CD_Contact

Child Components

Create From Template List: <Select a child compo

Find Similar List Dialog: List Dialog

Find Similar Search Referenced Attribute Properties

Component Description: Find similar search referenced attribute field

Attribute*: PhoneNumber

Label: Phone Number

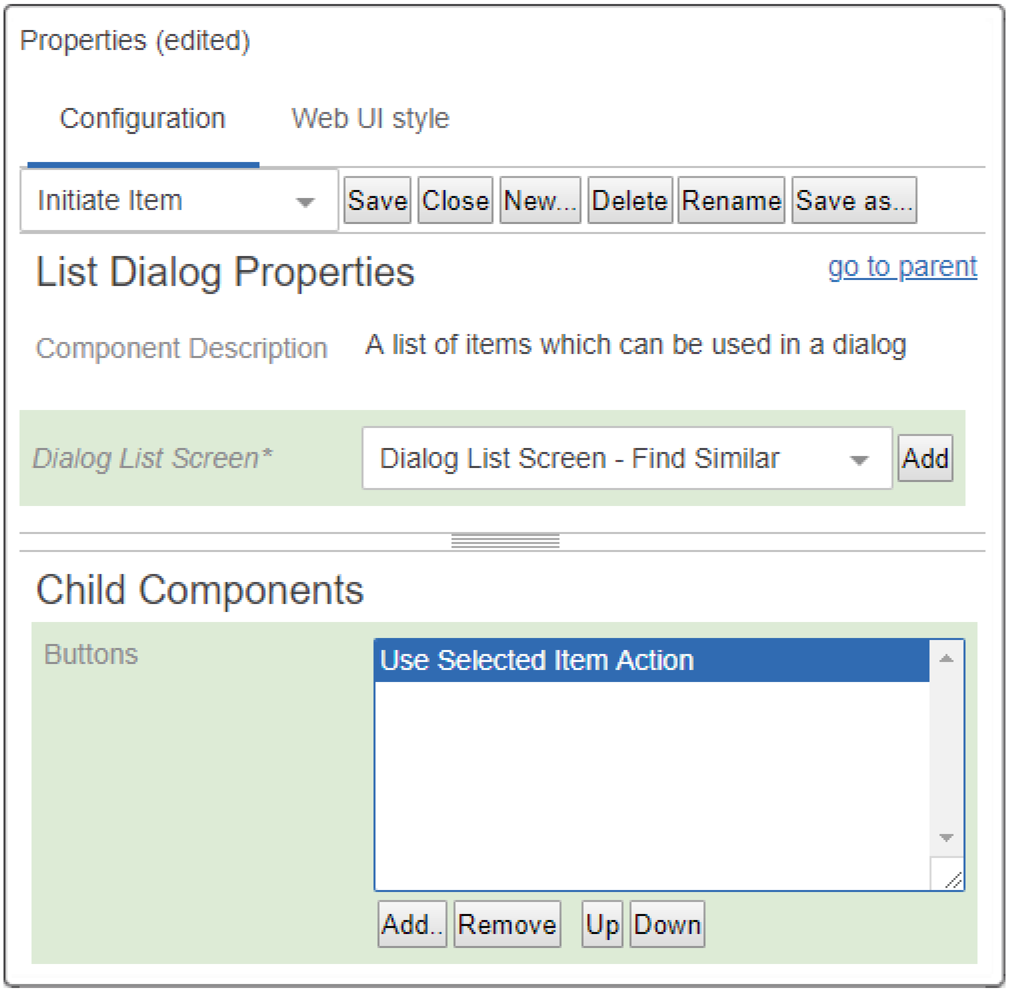
Object Type*: CD_Contact

Reference Type*: ContactToContact

6. Enable the Show Find Similar Tab setting back on the Add Reference Action Properties.
7. Click the ellipsis button (**...**) to make a selection for the Temp Object Type For Find Similar parameter and the Temp Parent For Find Similar parameter. (When the matching algorithm is run, it creates temporary objects based on the input in the search fields. These objects are then used in the algorithm to compare and find similar objects. The temporary objects need a parent and object type to be created. After the user finishes the operation, the temporary objects are deleted by the system. These parameters have to be configured for the functionality to work. The object types of both the direct objects and the referenced objects need to be made valid under the location used for the Temp Parent For Find Similar parameter.)

Important: The node selected for Temp Parent For Find Similar in the configuration, must also be included in the category specified in the match code definition if a category is specified. If this is not done, the match codes will not generate properly, and the match results will be incorrect.

8. Under Child Components > Find Similar List Dialog, select List Dialog from the dropdown menu.
9. On List Dialog Properties, click Add to the right of the Dialog List Screen value field.
10. Create a new screen by selecting Dialog List Screen from the list of available screen types / components. Enter an easily identifiable Screen ID, click Add. The screen ID will automatically populate in the Dialog List Screen value field.
11. If a Dialog List Screen already exists, skip the previous two steps, and on the List Dialog Properties, select the Dialog List Screen using the dropdown.
12. On the List Dialog Properties screen, in the Child Components > Buttons field, click Add and select Use Selected Item Action. Click Add, and then click Save in the designer window.



List Dialog Properties

Component Description: A list of items which can be used in a dialog

Dialog List Screen*: Dialog List Screen - Find Similar

Child Components

Buttons: Use Selected Item Action

13. Before exiting design mode, select the Dialog List Screen ID of the screen you just created from the dropdown menu. In the example above, the screen ID is Dialog List Screen - Find Similar.
14. Configure the Child Components > Headers section of the Dialog List Screen Properties. Click Add under the value field to set up the attributes that will display in the Find Similar Search tab for the results list. The Help Text is editable and will display at the top of the results list.
15. Click Save and Close design mode to return to normal Web UI mode.

Find Similar Webservice (getSimilarObjects)

To avoid users creating duplicate objects in source systems, integrators can make use of the getSimilarObject SOAP web service.

The following section discusses how the getSimilarObjects web service sends a call when performing a Search Before Create operation. The getSimilarObjects web service is a part of the SOAP API.

The getSimilarObjects request defines the criteria for the match and the information to be returned. The following should be supplied in the call.

- **Access Context** - This parameter contains the username and password for the user accessing the system. It may optionally contain the context and workspace as well.
- **Values** - The values supplied are used by the matching engine for comparison. The property URL points to the URL of the attribute ID in the system that the value should be associated with for comparison.
- **Object Type URL** - This parameter is the URL of the object type in the system which will be used as a base for comparison.
- **Matching Algorithm URL** - The URL of the matching algorithm in the system to perform the comparison.
- **Export Configuration XML** - This optional section defines the information in XML format of the potential duplicates to be returned in the response. If excluded from the request, the STEP ID, STEP URL, Title, Super Type, Object Type URL, and Score will be returned. The records will be returned in order of highest score to lowest score.
- **Search Threshold** - The score threshold of potential duplicates to be returned. If the search threshold is 70, only records that match the supplied values with a score of 70 or above will be returned in the response. The Clerical Review Threshold and the Auto Threshold defined in the matching algorithm are ignored.
- **Max Count** - The maximum number of potential duplicates to return. If the matching algorithm identifies 100 records that score above the Search Threshold and the Max Count is set to 10, only the top 10 scoring records will be returned in the response.

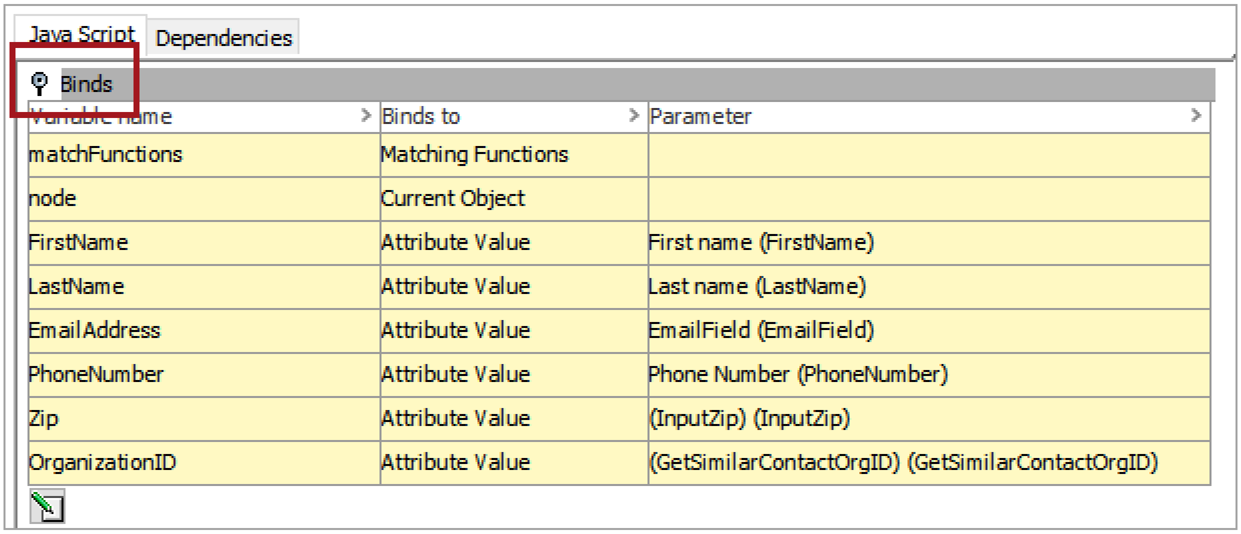
Complete documentation for Web Services functionality related to getSimilarObjects can be found in the SOAP API documentation at [system]/sdk or by clicking the **STEP API Documentation** button on the Start Page. This topic provides some basic information and an example of a simple SOAP request but should not be considered comprehensive.

Note: getSimilarObjects rely on the match algorithm to search, and so match codes must exist and be up-to-date on the records, and match criteria must be set up in the system for the 'getSimilarObjects' call to work.

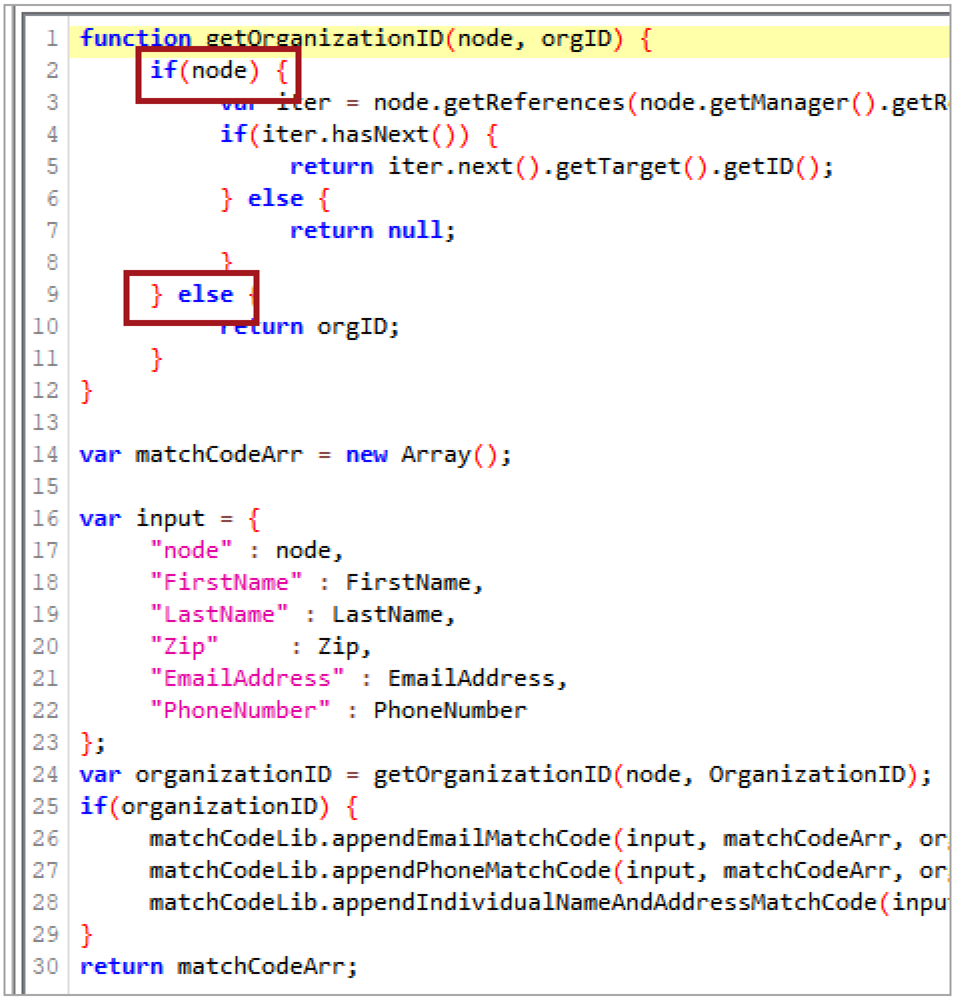
Node Binds and the getSimilarObjects Web Service

When a getSimilarObjects call is made, a node–a permanent STEP object–is not created in the system. This non-permanent state means that the match code cannot get a hold of the 'Current Object,' and the match algorithm cannot get a hold of the `first()` node via the 'Match Expression Context.' For results to be returned, the matching engine needs a way to compare the values in the call to the values on the existing system nodes. This issue is solved through binds.

Binds associate incoming values to attributes in the system allowing the matching engine to make the appropriate comparisons. All values used in the match code should be defined under the binds flipper. Match codes should make use of the `if(node){}else{}` function for values not on the current object such as the reference being used in the code below.



Variable name	Binds to	Parameter
matchFunctions	Matching Functions	
node	Current Object	
FirstName	Attribute Value	First name (FirstName)
LastName	Attribute Value	Last name (LastName)
EmailAddress	Attribute Value	EmailField (EmailField)
PhoneNumber	Attribute Value	Phone Number (PhoneNumber)
Zip	Attribute Value	(InputZip) (InputZip)
OrganizationID	Attribute Value	(GetSimilarContactOrgID) (GetSimilarContactOrgID)



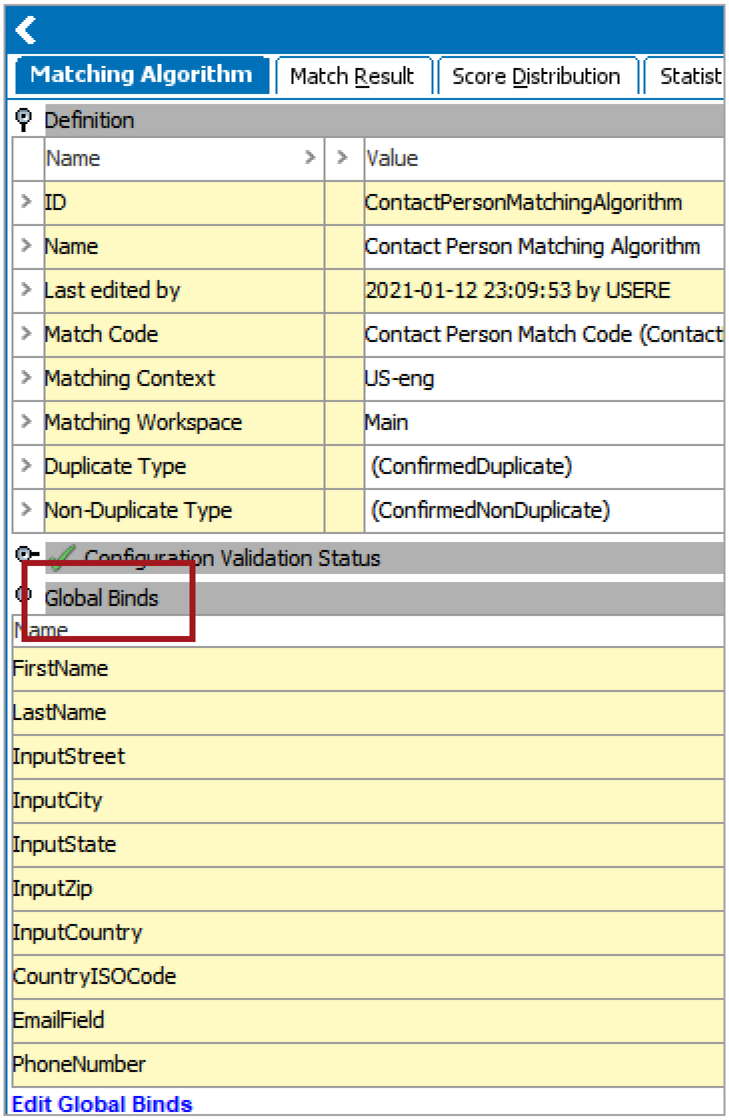
```
function getOrganizationID(node, orgID) {
    if(node) {
        var iter = node.getReferences(node.getManager().getR
        if(iter.hasNext()) {
            return iter.next().getTarget().getID();
        } else {
            return null;
        }
    } else {
        return orgID;
    }
}

var matchCodeArr = new Array();

var input = {
    "node" : node,
    "FirstName" : FirstName,
    "LastName" : LastName,
    "Zip"      : Zip,
    "EmailAddress" : EmailAddress,
    "PhoneNumber" : PhoneNumber
};
var organizationID = getOrganizationID(node, OrganizationID);
if(organizationID) {
    matchCodeLib.appendEmailMatchCode(input, matchCodeArr, or
    matchCodeLib.appendPhoneMatchCode(input, matchCodeArr, or
    matchCodeLib.appendIndividualNameAndAddressMatchCode(inpu
}
return matchCodeArr;
```

All values used in the match algorithm should be defined under the 'Global Binds' flipper. This expectation applies to both explicitly configured attributes and for those configured via component models. If customer data normalizers are used, the name of the global bind must match the ID of the corresponding attribute.

Note: If using the 'Address Normalizer' in the match algorithm, the following attributes defined in the Address component model should be bound to the match algorithm:

- Input Street
- Input City
- Input State
- Input Zip
- Input Country
- Country ISO Code

Matching using binds is not optimized for the In-Memory Database Component.



Definition

Name	Value
ID	ContactPersonMatchingAlgorithm
Name	Contact Person Matching Algorithm
Last edited by	2021-01-12 23:09:53 by USERE
Match Code	Contact Person Match Code (Contact
Matching Context	US-eng
Matching Workspace	Main
Duplicate Type	(ConfirmedDuplicate)
Non-Duplicate Type	(ConfirmedNonDuplicate)

Configuration Validation Status

Global Binds

Name
FirstName
LastName
InputStreet
InputCity
InputState
InputZip
InputCountry
CountryISOCode
EmailField
PhoneNumber

Edit Global Binds

For more documentation for Web Services functionality related to GetSimilarObjects, click the **STEP API Documentation** button on the STEP Start Page and see the SOAP API section. For information on JavaScript binds, see the **JavaScript Binds** topic in the **Resource Materials** documentation.