ORACLE

8

Data Mapping

Objectives

After completing this lesson, you should be able to:

- Map or transform data by using the OIC Mapper tool
- Create XSL expressions using XSLT functions and operators
- Describe and use the OIC Recommendations Tool
- Edit and import advanced XSL files into OIC
- Register and use custom JavaScript functions
- Create and invoke OIC Lookups

Agenda

- OIC Data Mapper
- OIC Recommendations Engine
- Advanced Transformation Options
- Using OIC Lookups

Integration Development (Review)

Define Connections

Build Integration Flow Logic

Map Data

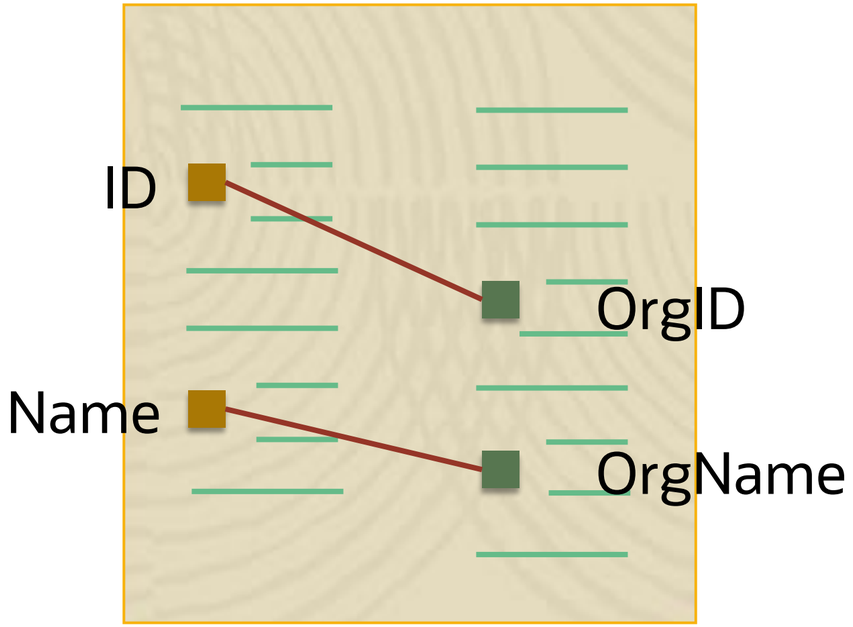
Use the OIC Data Mapper to define payload for each invoked connection.

Activate Integration

Monitor Integrations

OIC Data Mapping

- A visual mapper enables you to define data mapping between different data structures by using drag and drop.
- Automatically populate target data structures with information pulled from one or more source data structures.
- Transformation is supported between XML and XML.



Supported Features

- Transformation:
  - XSLT visual data mapper tool
  - XPath functions (standard & custom): Define how data is modified if needed
  - OIC Lookups (specialized): Using Domain Value Maps
  - Expression Builder: For more complex expressions
- Repeat a Target element to map from different sources
- Create for-each statement automatically
- A recommend option to automate some mappings

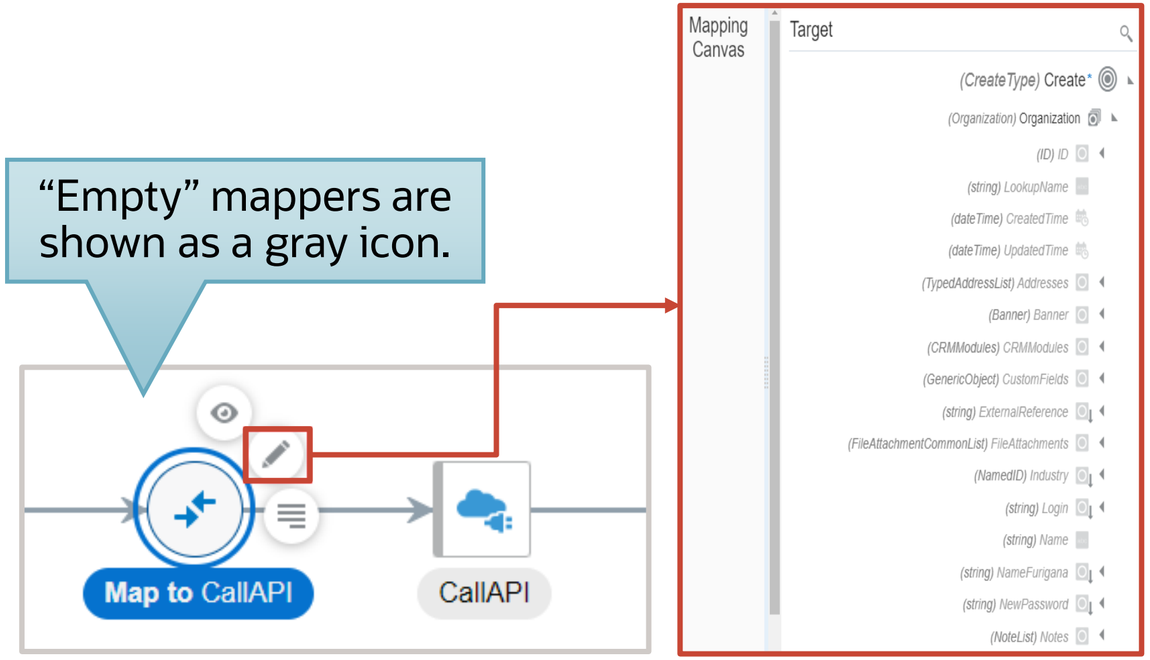
Launching the OIC Data Mapper (Review)

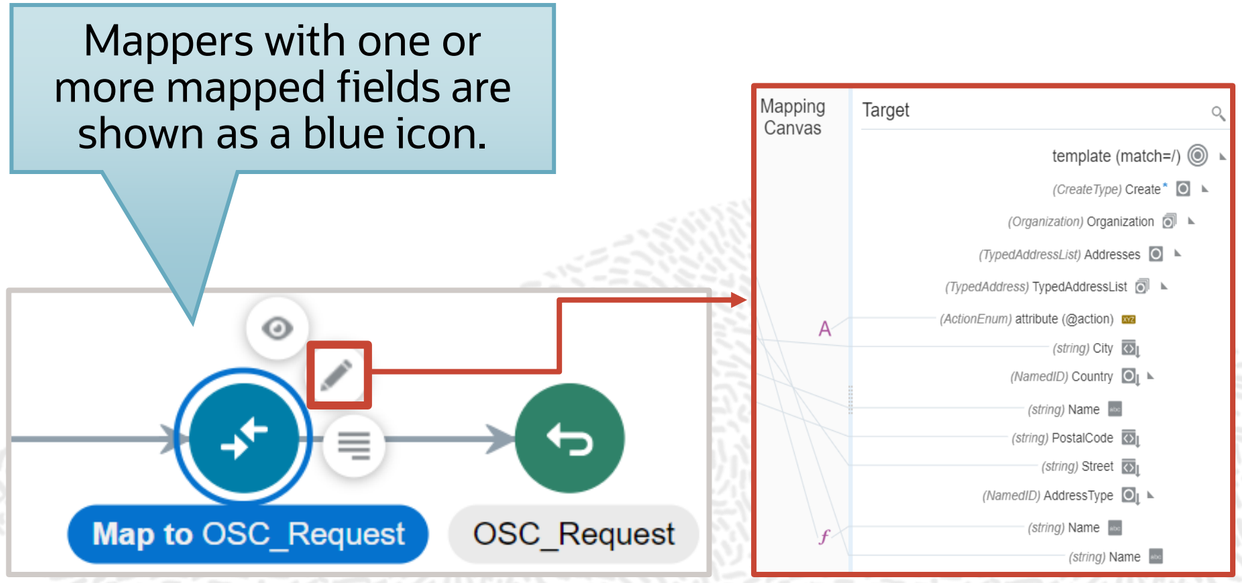
Click the **Map** action icon to reveal three smaller icons and then click the pencil edit icon:

- Preceding an *Invoke* or *Call* action
- Preceding a *Return* or *Callback* action

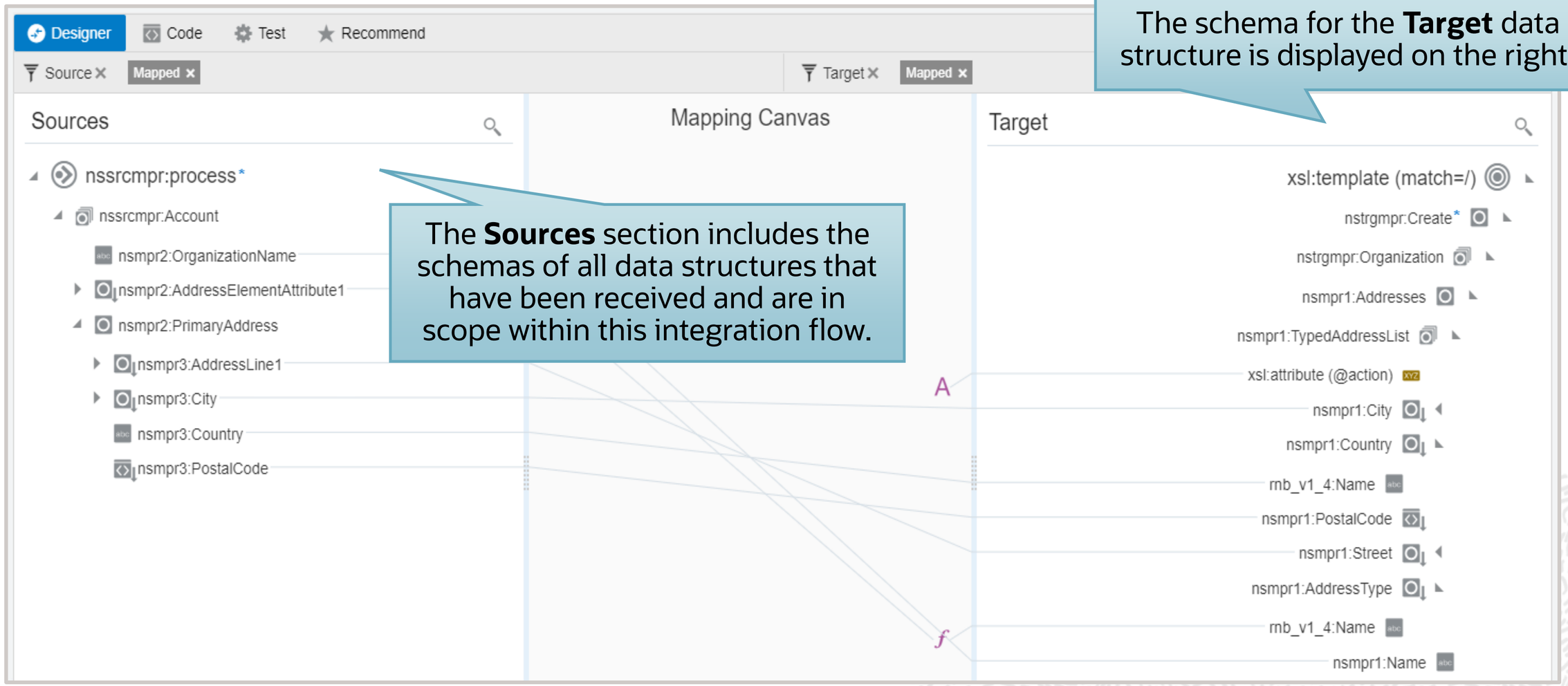
- View

- Edit





OIC Data Mapper: Designer View



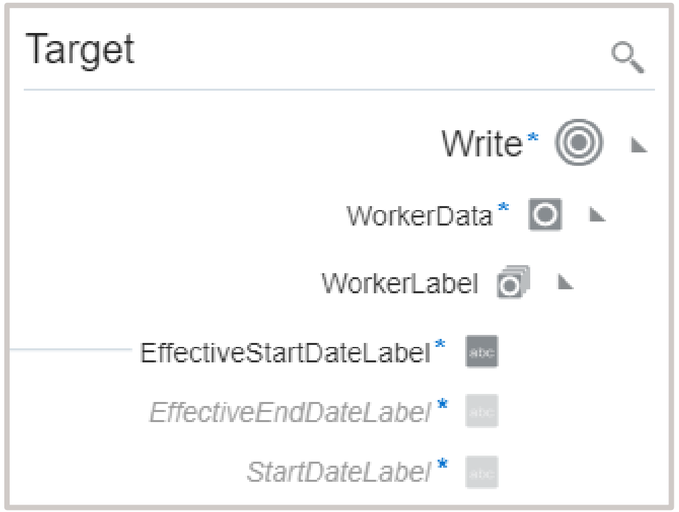
The schema for the **Target** data structure is displayed on the right.

The **Sources** section includes the schemas of all data structures that have been received and are in scope within this integration flow.

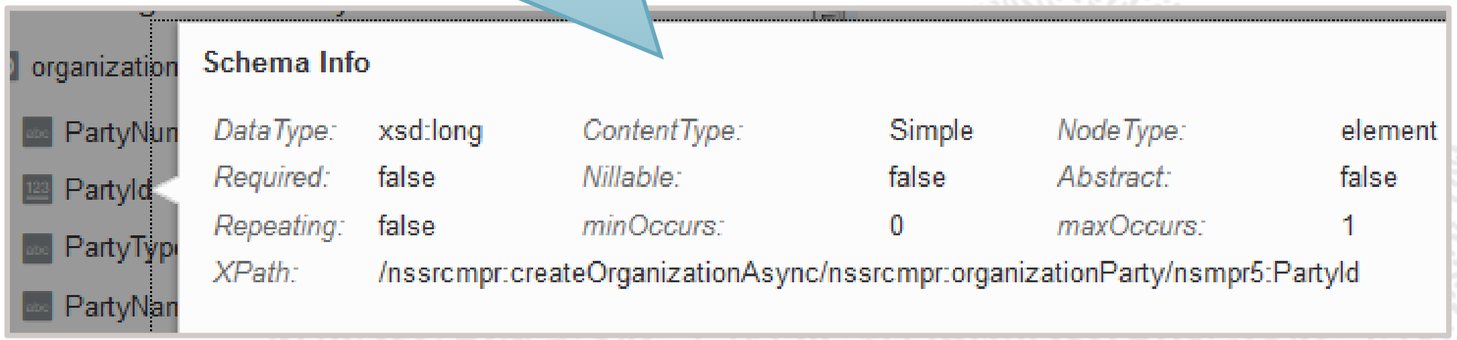
Schema Support

Transformation maps use eXtensible Stylesheet Language (XSL).

- All Source and Target data objects are represented internally as XML structures.
- Mapping supports both qualified and unqualified schemas.
- Required fields are identified by a blue asterisk (*).

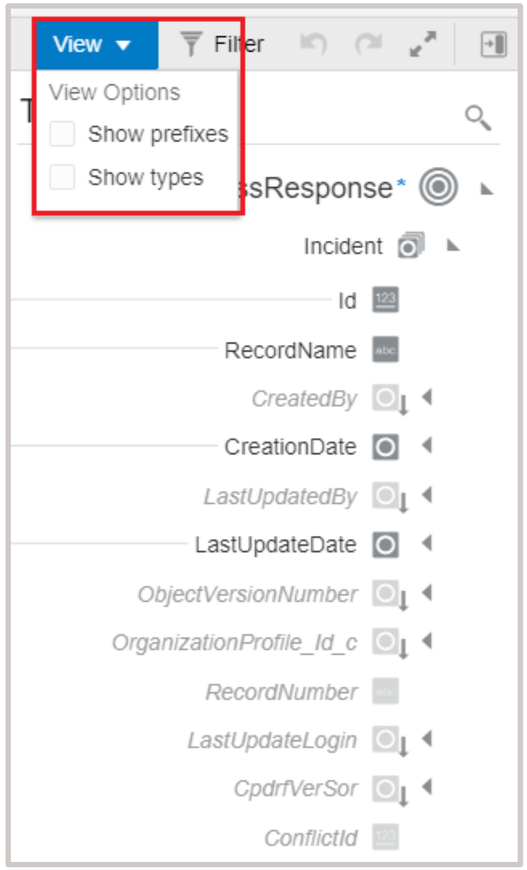


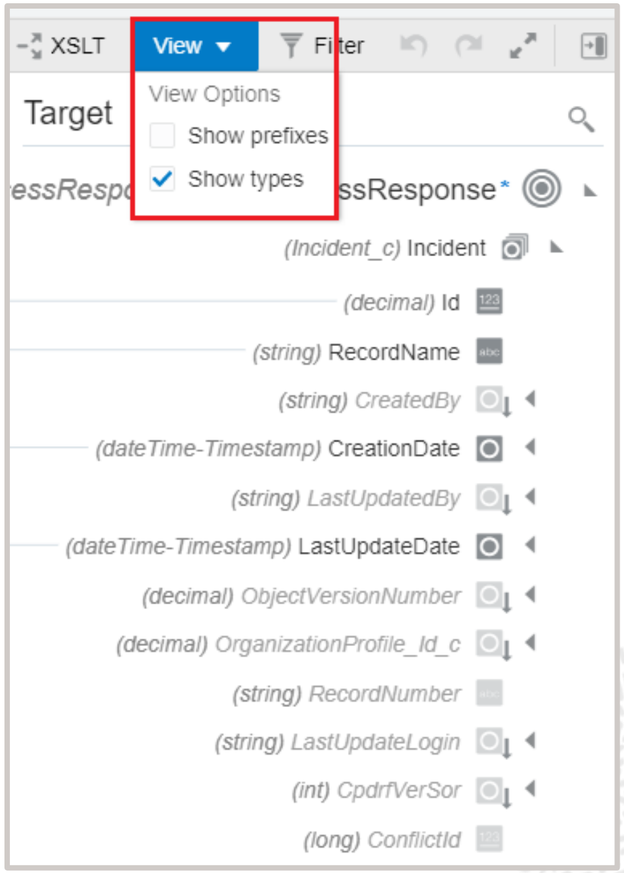
Right-click an element or attribute and select **Node Info** for more details.

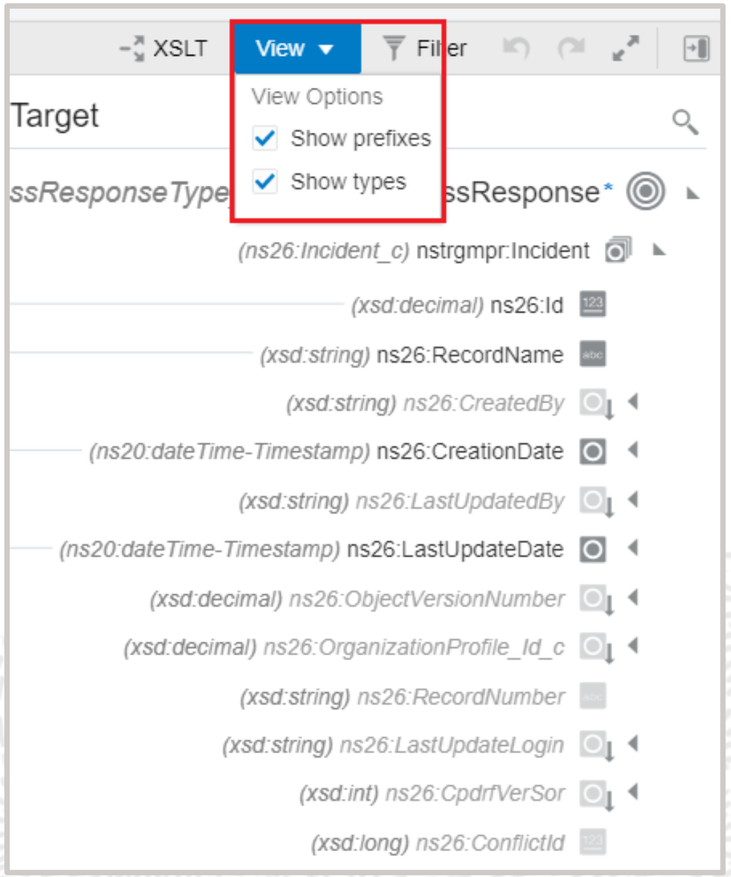


Data Structure View Options

Click the **View** button to select display options for data element nodes and attributes.

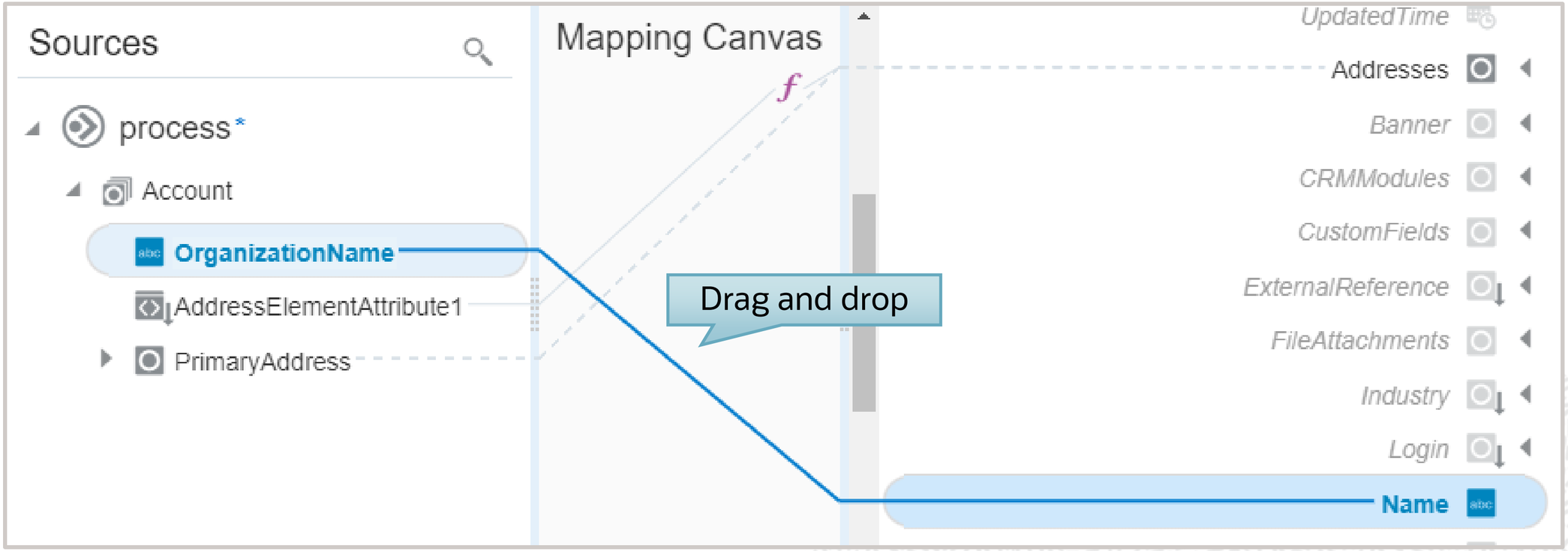






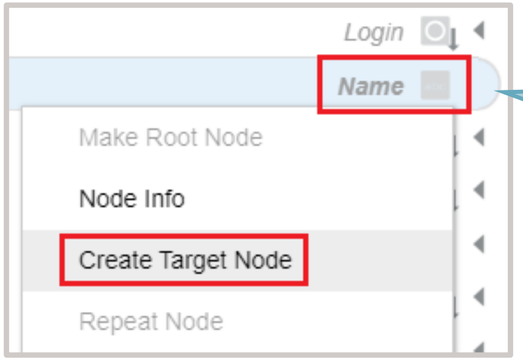
Basic Field Mapping: Drag & Drop

Click a field in a **Source** data field and drag it to the required **Target** field.



Basic Field Mapping: Using the Expression Builder

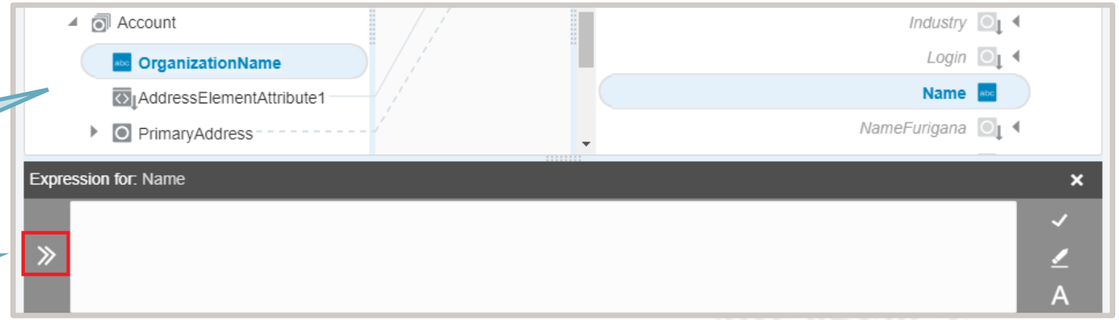
Right-click an empty **Target** field and select *Create Target Node* which reveals the Expression Builder for that element.

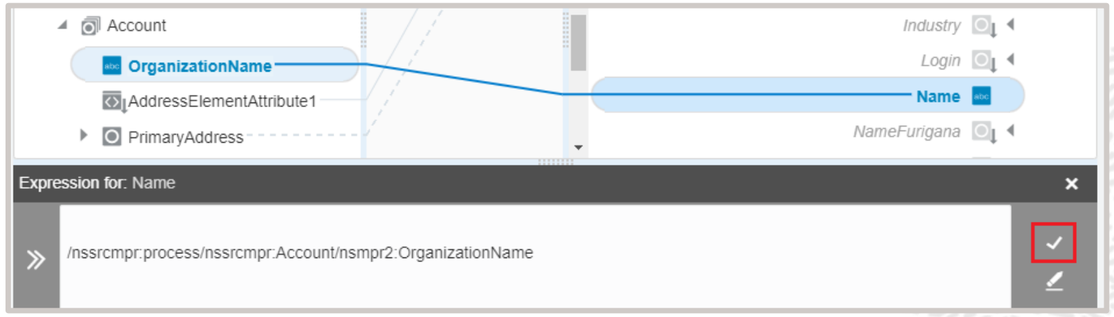


Right-click

Select a **Source** data field

Use the shuttle button to insert expression

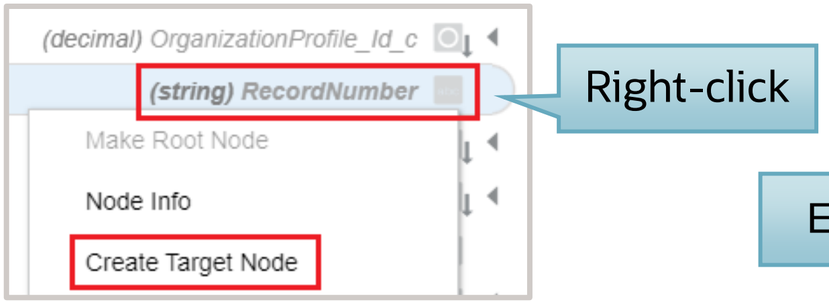


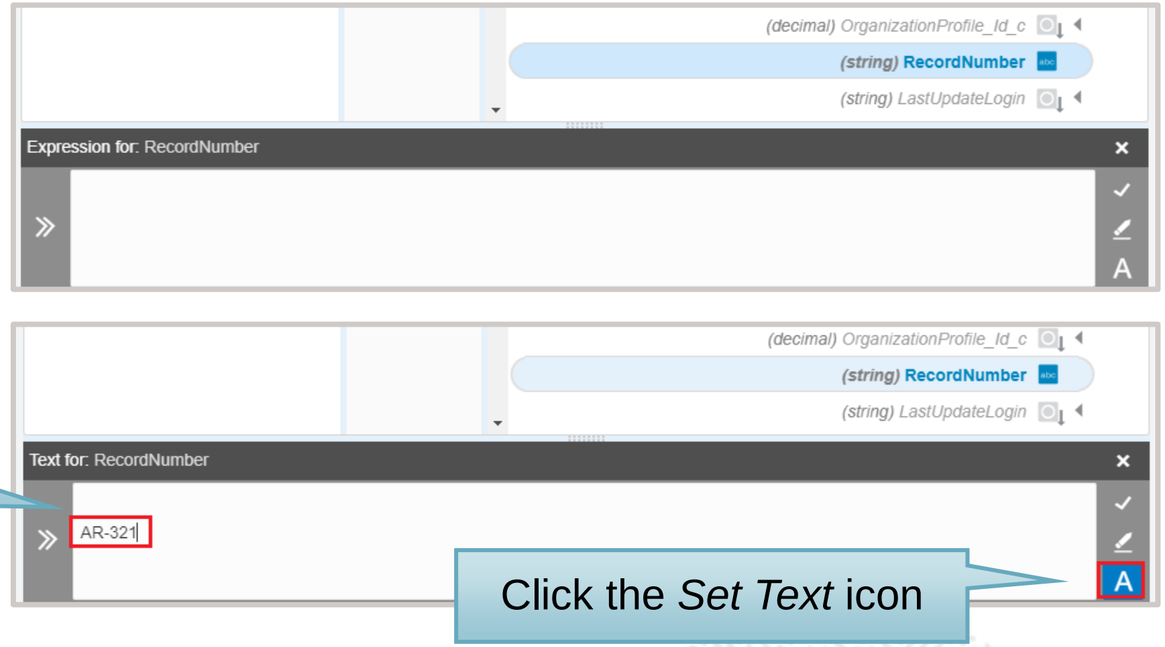


Click the check mark to save the mapping

Using the Set Text Mode

Right-click an empty **Target** field and select *Create Target Node* which reveals the Expression Builder for that element.

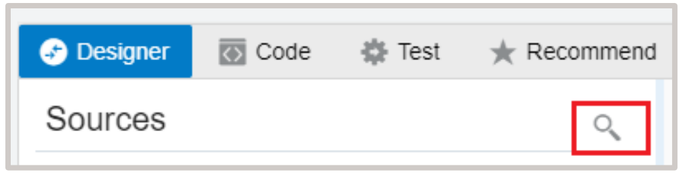


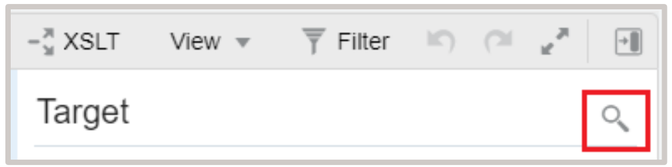


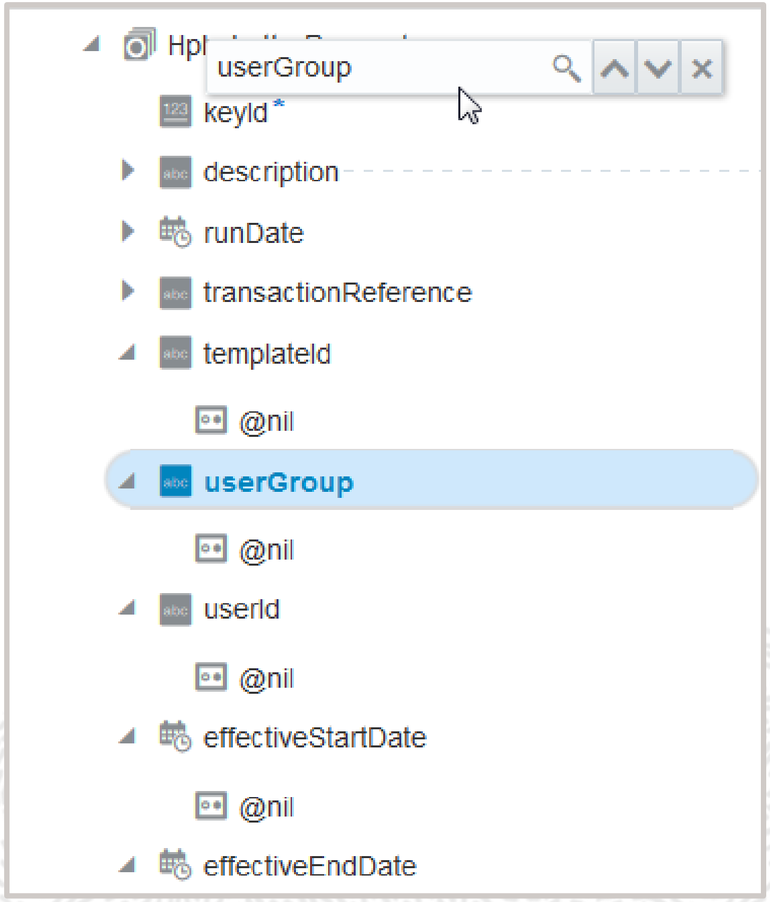
Search Data Fields

You can search for specific element nodes or attributes in either the Source or Target structure.

- Click the *Search* icon
- Enter a full or partial name
- Click the **V** icon to advance
- Click the **X** icon to close the Search bar



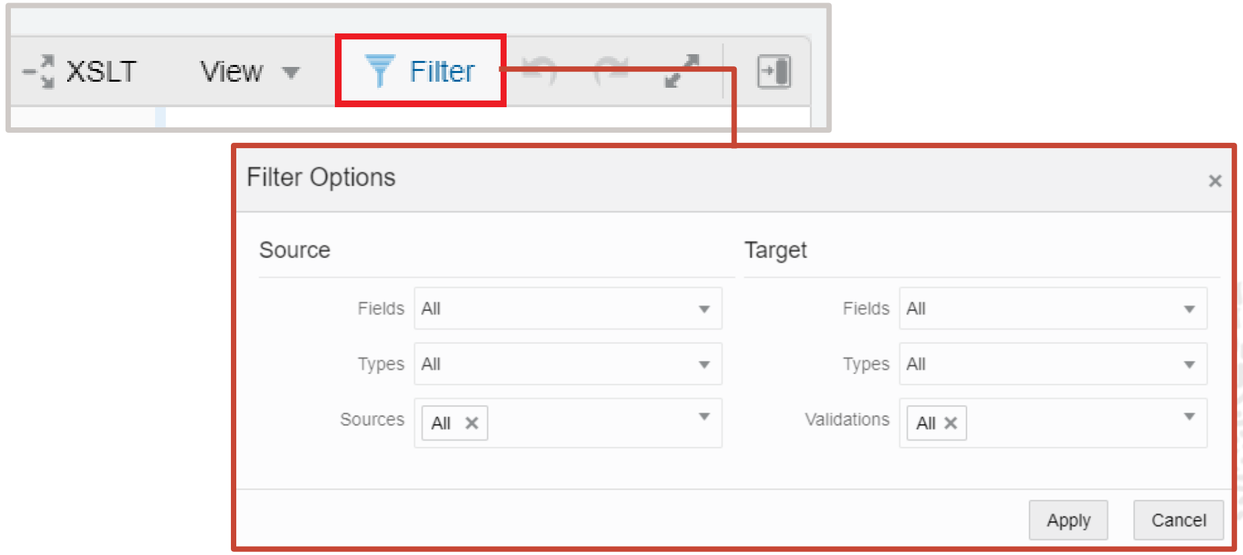


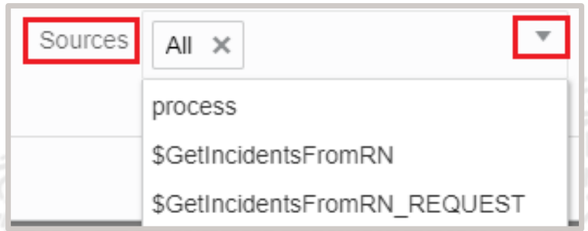


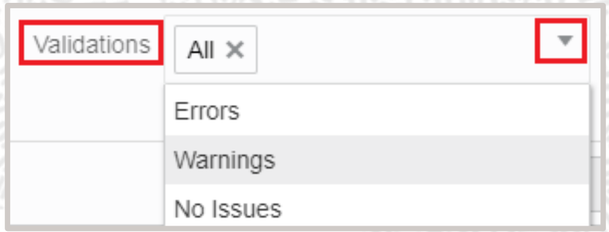
Filter the Source or Target Data Structures

You can filter the display of the source and/or target structures.

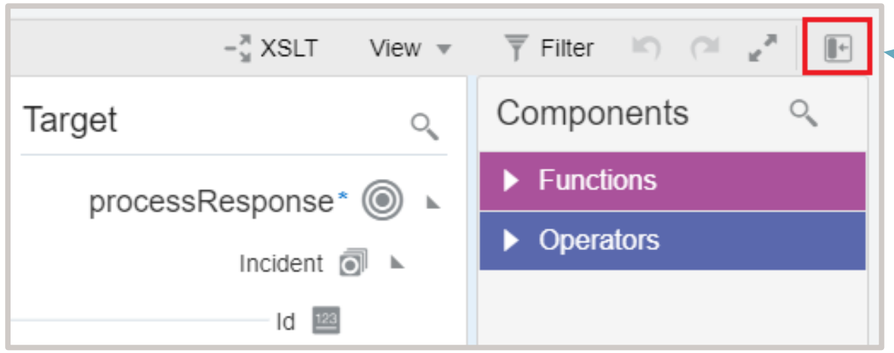
- Fields (*mapped, unmapped*)
- Types (*required, custom*)
- Sources
- Target Validations (*errors, warnings, no issues*)





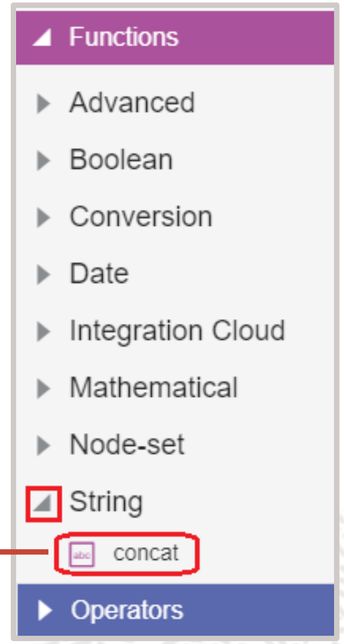


Using XPath Functions and Operators

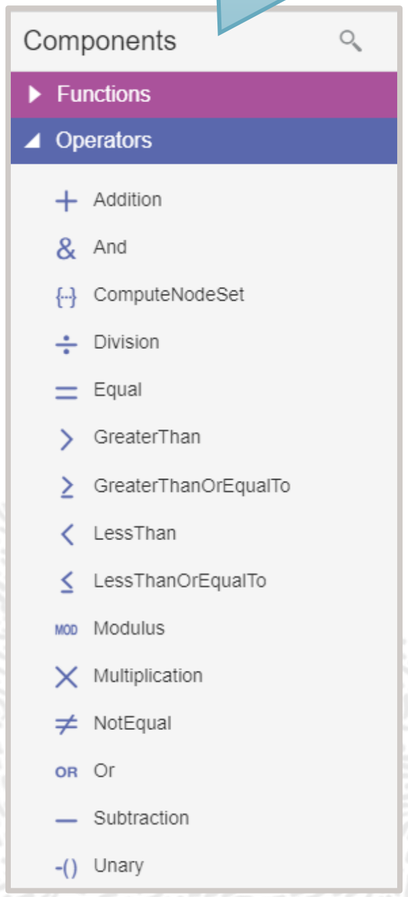


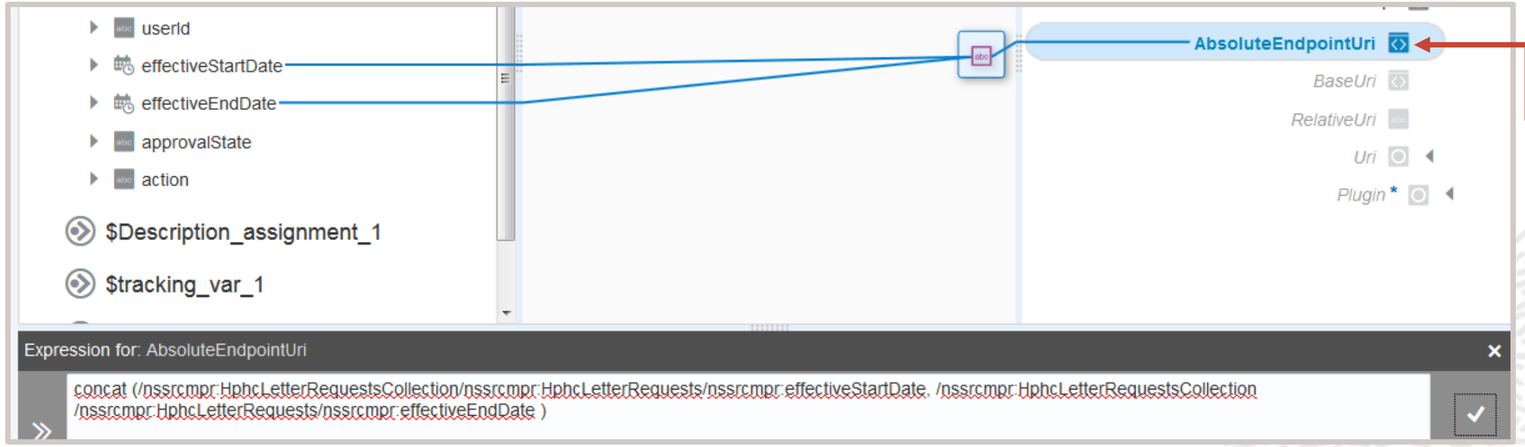
Click the *Toggle functions* icon to expose the **Components** panel.

Expand a category to locate the function, then drag to the target element or to the Expression box.



Expand to display XPath operators

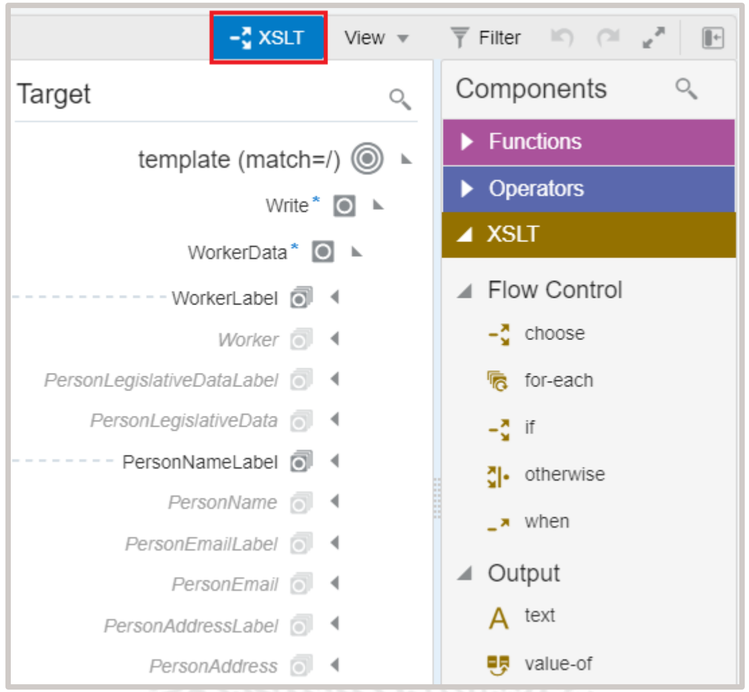




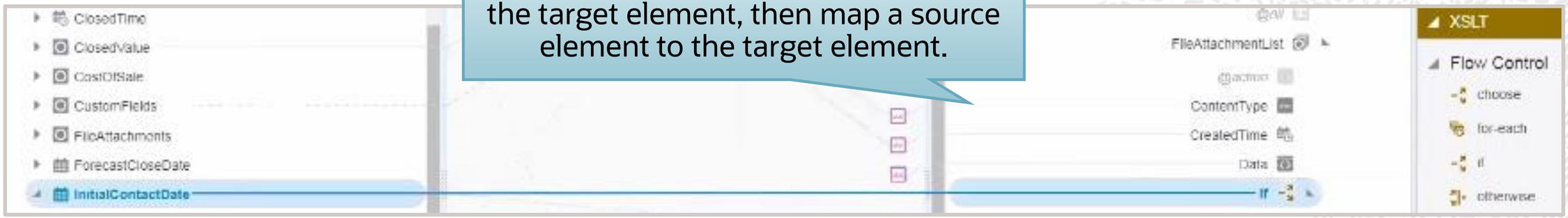
Working With XSLT Statements

Click the **XSLT** (or **Advanced**) button to add the XSLT header to the *Components* panel.

- Expand the XSLT header tab.
- Browse for and drag the appropriate statement onto the target element node.
- If the element is grayed out (is a ghost node), right-click the element and select *Create Target Node*.



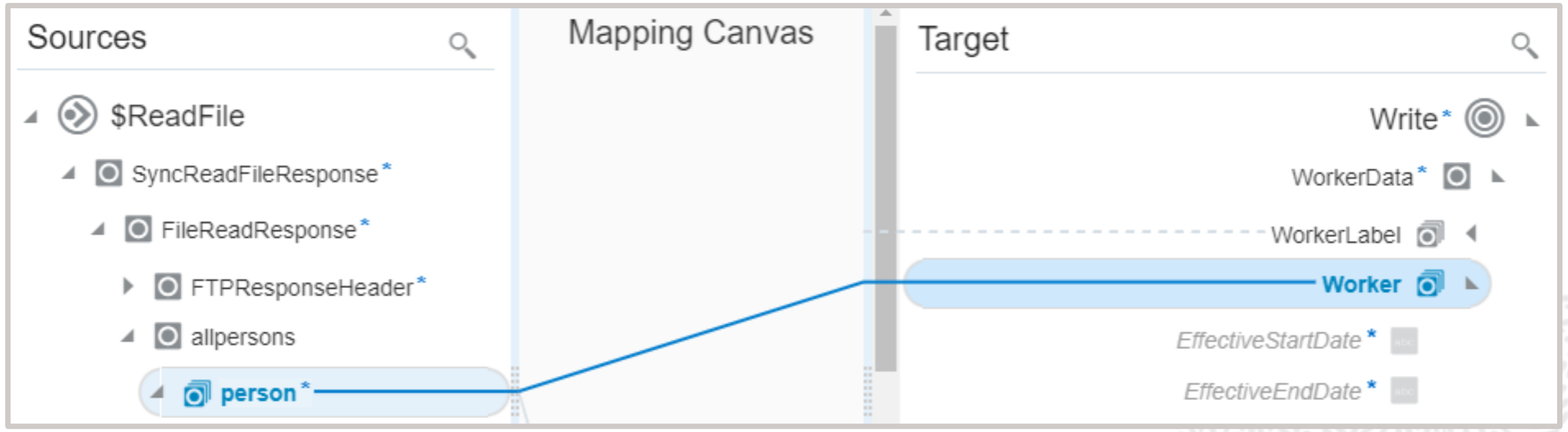
For example, drag an **if** statement to the target element, then map a source element to the target element.

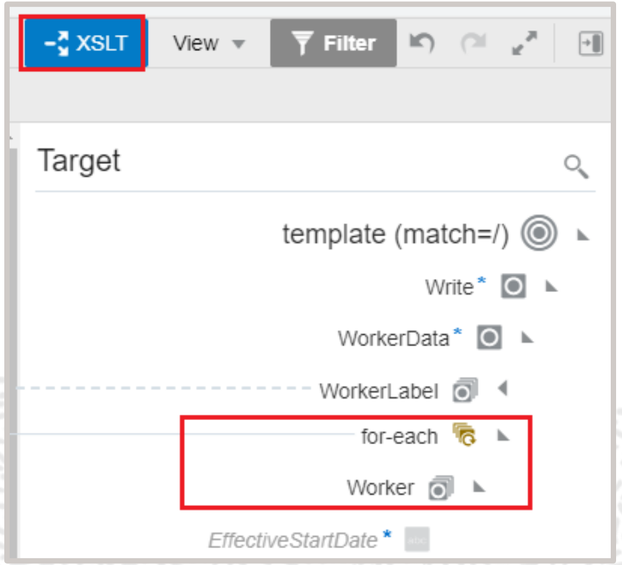


Automatically Create for-each Statements

You can automatically create for-each statements when mapping between repeatable source and target elements in the mapper.

- Repeatable elements are identified by this icon: 
- Toggle the XSLT button to view created for-each statement

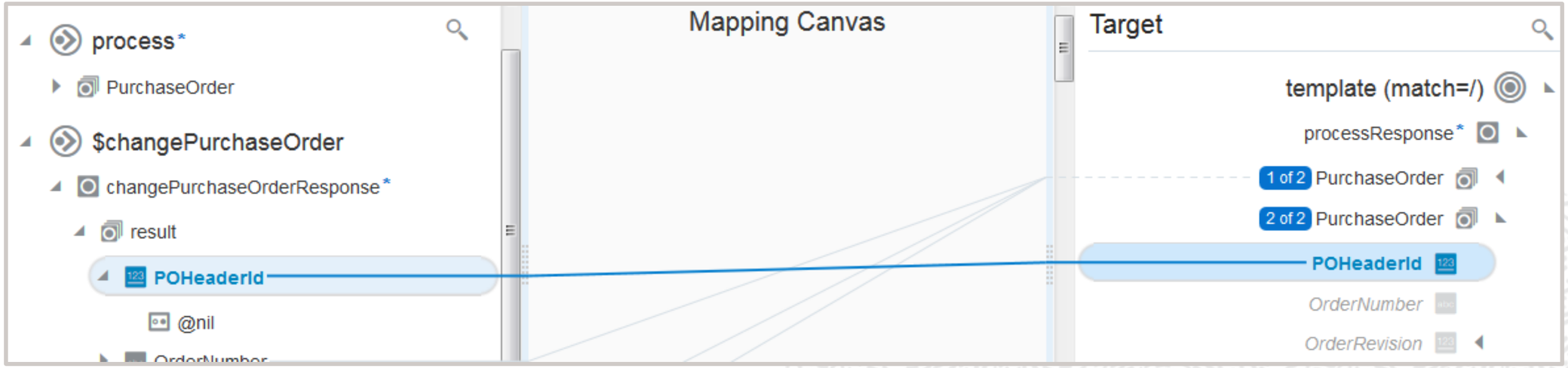




Repeat a Target Element to Map to Different Sources

You can repeat a target element in the mapper, enabling you to map different sources to the same target element.

- Elements defined in the target schema with the *maxOccurs* attribute set to a value greater than one can be repeated.



Accessing the XSL Stylesheet

Click the **Code** tab to display the source stylesheet.

If desired, copy contents to be viewed in a separate application.

Copy and paste all contents and save as an .xsl file

Data (D:) › temp

OSC_Map.xsl
XSL Stylesheet
6.99 KB

Designer | Code | Test | Recommend

```
17                      <oracle-xsl-mapper:param name="GetIncidentsFromRN_REQUEST" xml:id="id_43"/>
18                  </oracle-xsl-mapper:source>
19                  <oracle-xsl-mapper:source type="XSD" xml:id="id_44">
20                      <oracle-xsl-mapper:schema
location="../../processor_63/resourcegroup_64/ICSIntegrationMetadata.xsd" xml:id="id_45"/>
21                      <oracle-xsl-mapper:rootElement name="metadata"
namespace="http://www.oracle.com/2014/03/ic/integration/metadata" xml:id="id_46"/>
22                      <oracle-xsl-mapper:param name="self" xml:id="id_47"/>
23                  </oracle-xsl-mapper:source>
24              </oracle-xsl-mapper:mapSources>
25              <oracle-xsl-mapper:mapTargets xml:id="id_7">
26                  <oracle-xsl-mapper:target type="WSDL" xml:id="id_8">
27                      <oracle-xsl-mapper:schema
location="../../application_18/outbound_19/resourcegroup_20/OSC_Request_REQUEST.wsdl" xml:id="id_9"/>
28                      <oracle-xsl-mapper:rootElement name="processResponse"
namespace="http://xmlns.oracle.com/cloud/adapter/osc/OSC_Request_REQUEST/types" xml:id="id_10"/>
29                  </oracle-xsl-mapper:target>
30              </oracle-xsl-mapper:mapTargets>
31              <!--GENERATED BY ORACLE XSL MAPPER 12.1.2.0.0-->
32          </oracle-xsl-mapper:schema>
33          <!--User Editing allowed BELOW this line - DO NOT DELETE THIS LINE-->
34          <xsl:param name="GetIncidentsFromRN" xml:id="id_25"/>
35          <xsl:param name="GetIncidentsFromRN_REQUEST" xml:id="id_48"/>
36          <xsl:param name="self" xml:id="id_49"/>
37          <xsl:param name="tracking_var_1" xml:id="id_50"/>
38          <xsl:param name="tracking_var_2" xml:id="id_51"/>
39          <xsl:param name="tracking_var_3" xml:id="id_52"/>
40          <xsl:template match="/" xml:id="id_11">
41              <nstrgmpr:processResponse xml:id="id_12">
42                  <nstrgmpr:Incident>
43                      <ns26:Id>
44                          <xsl:value-of select="/nstrgmpr:process/nstrgmpr:Account/ns25:PartyId"/>
45                      </ns26:Id>
46                      <ns26:CreationDate>
47                          <xsl:value-of
select="$GetIncidentsFromRN/nsmpr0:QueryObjectsResponse/nsmpr0:Incident/rnb_v1_4:CreatedTime"/>
48                      </ns26:CreationDate>
49                  </nstrgmpr:Incident>
50              </nstrgmpr:processResponse>
51          </xsl:template>
52      </xsl:stylesheet>
```

Testing Your Mappings

Click the **Test** button to open testing tool

Click to generate random inputs

Map

Designer | Code | Test | Recommend | Clear | Generate Inputs | Execute

Source:createOrganizationAsync

```
1  <?xml version='1.0' encoding='UTF-8'?>
2  <createOrganizationAsync xmlns="http://xmlns.oracle.com/apps/cdm/foundation
   /parties/organizationService/applicationModule/types/">
3     <organizationParty>
4        <PartyNumber xmlns="http://xmlns
   /organizationService/">PartyNumber1</P
5        <PartyId xmlns="http://xmlns.ora
   /organizationService/">2</PartyId>
6        <PartyType xmlns="http://xmlns.c
   /organizationService/">ORGANIZATION</P
7        <PartyName xmlns="http://xmlns.c
   /organizationService/">PartyName3</Par
8        <LastUpdatedBy xmlns="http://xmlns.orac  /cdm/foundation
   /parties/organizationService/">LastUpdatedB  stUpdatedBy>
9        <ValidatedFlag xmlns="http://xmlns.oracle.com/apps/cdm/foundation
   /parties/organizationService/">true</ValidatedFlag>
10       <LastUpdateLogin xmlns="http://xmlns.oracle.com/apps/cdm/foundation
   /parties/organizationService/">LastUpdateLogin5</LastUpdateLogin>
11       <CreationDate xmlns="http://xmlns.oracle.com/apps/cdm/foundation/parties
   /organizationService/">2019-06-22T20:22:54.650</CreationDate>
12       <RequestId xmlns="http://xmlns.oracle.com/apps/cdm/foundation/parties
   /organizationService/">6</RequestId>
13       <LastUpdateDate xmlns="http://xmlns.oracle.com/apps/cdm/foundation
   /parties/organizationService/">2019-06-22T20:22:54.650</LastUpdateDate>
14       <CreatedBy xmlns="http://xmlns.oracle.com/apps/cdm/foundation/parties
   /organizationService/">CreatedBy7</CreatedBy>
15       <OrigSystemReference xmlns="http://xmlns.oracle.com/apps/cdm/foundation
   /parties/organizationService/">OrigSystemReference8</OrigSystemReference>
16       <HQBranchIndicator xmlns="http://xmlns.oracle.com/apps/cdm/foundation
   /parties/organizationService/">HQBranchIndicator9</HQBranchIndicator>
17       <DUNSNumberC xmlns="http://xmlns.oracle.com/apps/cdm/foundation/parties
   /organizationService/">DUNSNumberC10</DUNSNumberC>
18       <SICCode xmlns="http://xmlns.oracle.com/apps/cdm/foundation/parties
   /organizationService/">SICCode11</SICCode>
19       <JgzzFiscalCode xmlns="http://xmlns.oracle.com/apps/cdm/foundation
   /parties/organizationService/">JgzzFiscalCode12</JgzzFiscalCode>
20       <Address1 xmlns="http://xmlns.oracle.com/apps/cdm/foundation/parties
```

Replace random generated inputs or copy/paste other test payloads

Target:

```
1  <?xml version = '1.0' encoding = 'UTF-8'?>
2  <nstrgmpr:Create xmlns:rnn_v1_2="urn:nullfields.ws.rightnow.com/v1_2"
   xmlns:rnw_v1_2="urn:wsdl.ws.rightnow.com/v1_2" xmlns:ns1="http://xmlr
   /cloud/adapter/rightnow/RightnowAdapter1_REQUEST"
   xmlns:ns4="http://www.oracle.com/XSL/Transform/java/oracle.tip.dvm.Lo
   xmlns:rnf_v1_2="urn:faults.ws.rightnow.com/v1_2" xmlns:ns5="http://ww
   /XSL/Transform
   /java/com.bea.wli.sb.resources.icsxpathfunctions.ICSInstanceTrackingF
   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:xml="http://www.w
   /XML/1998/namespace" xmlns:rnm_v1_2="urn:messages.ws.rightnow.com/v1_2"
   xmlns:dvm="http://www.oracle.com/XSL/Transform
   /java/com.bea.wli.sb.functions.dvm.DVMFunctions"
   xmlns:rna_v1_2="urn:metadata.ws.rightnow.com/v1_2"
   xmlns:rng_v1_2="urn:generic.ws.rightnow.com/v1_2"
   xmlns:rno_v1_2="urn:objects.ws.rightnow.com/v1_2"
   xmlns:rnb_v1_2="urn:base.ws.rightnow.com/v1_2"
   xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
   xmlns:nstrgmpr="http://xmlns.oracle.com/cloud/adapter/rightnow
   /RightnowAdapter1_REQUEST/types" xmlns:soap="http://schemas.xmlsoap.org
   /wsdl/soap/">
3     <nstrgmpr:Organization>
4        <rnb_v1_2:ID id="2"/>
5        <rnb_v1_2:LookupName>PartyName3</rnb_v1_2:LookupName>
6        <rnb_v1_2:CreatedTime>2019-06-22T20:22:54.650</rnb_v1_2:CreatedTime>
7        <rnb_v1_2:UpdatedTime>2019-06-22T20:22:54.650</rnb_v1_2:UpdatedTime>
8        <rno_v1_2:Addresses>
9           <rno_v1_2:TypedAddressList>
10             <rno_v1_2:Country>
11                <rnb_v1_2:ID id="Country14"/>
12             </rno_v1_2:Country>
13             <rno_v1_2:Street>Address113</rno_v1_2:Street>
14          </rno_v1_2:TypedAddressList>
15       </rno_v1_2:Addresses>
16    </nstrgmpr:Organization>
17 </nstrgmpr:Create>
```

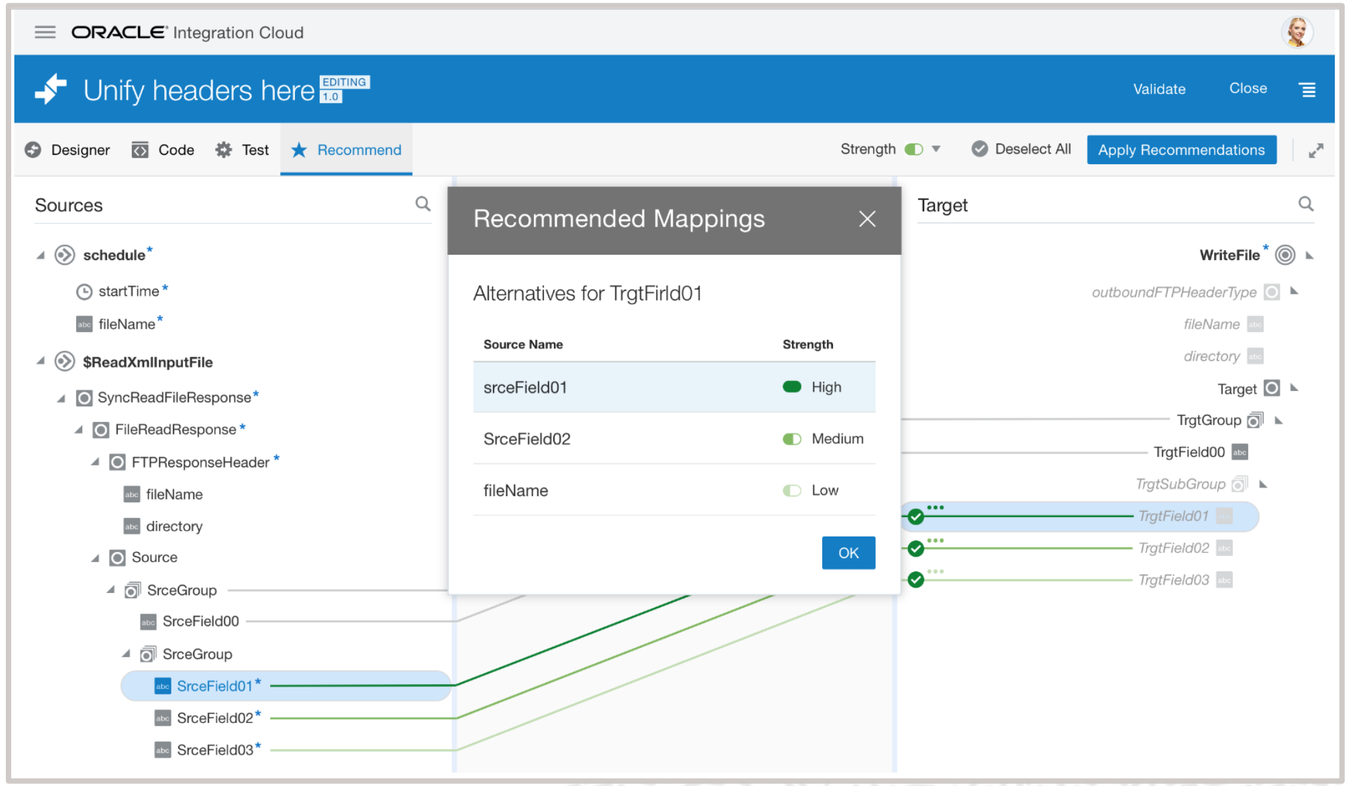
Click to execute then view the resulting target data structure

Agenda

- OIC Data Mapper
- OIC Recommendations Engine
- Advanced Transformation Options
- Using OIC Lookups

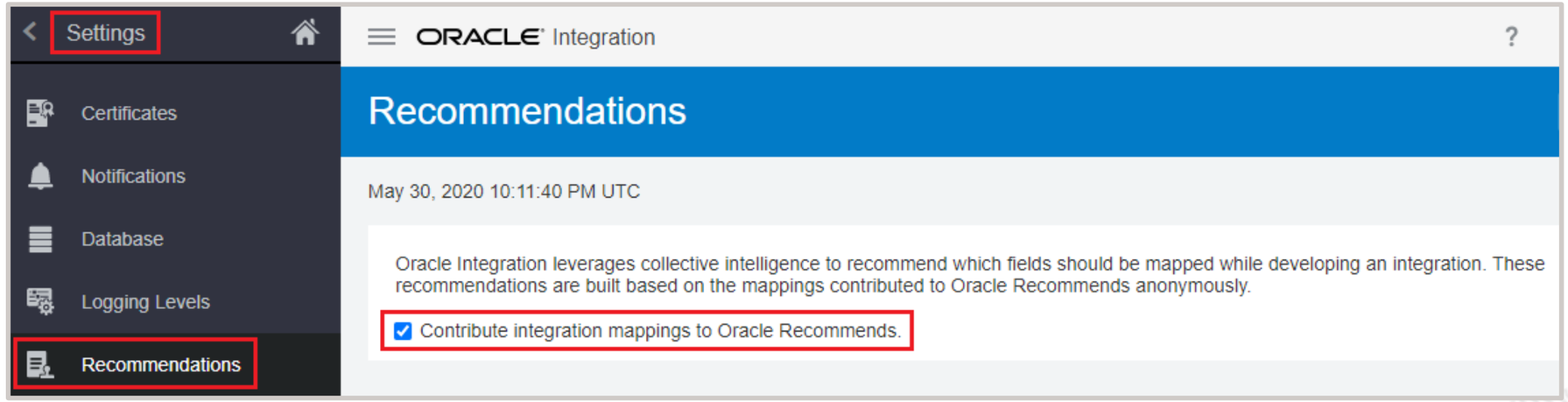
Data Mapping Recommendations

- AI and ML powered to save up to 60 percent of development effort
- Auto mapping and transformation to connect applications faster
- Machine learning guidance in OIC based on community usage
- Recommended Mappings highlight proven and popular best practices

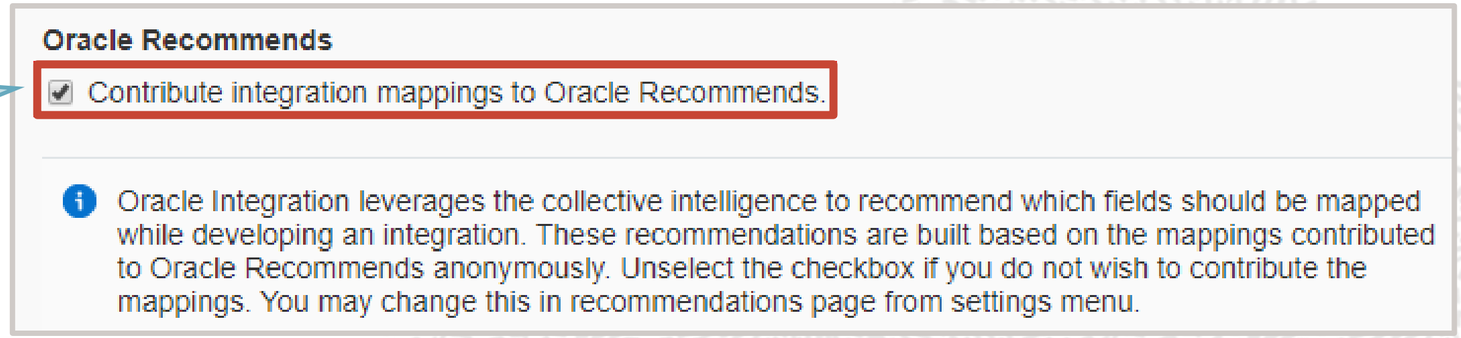


Enabling the Oracle Recommendations Engine

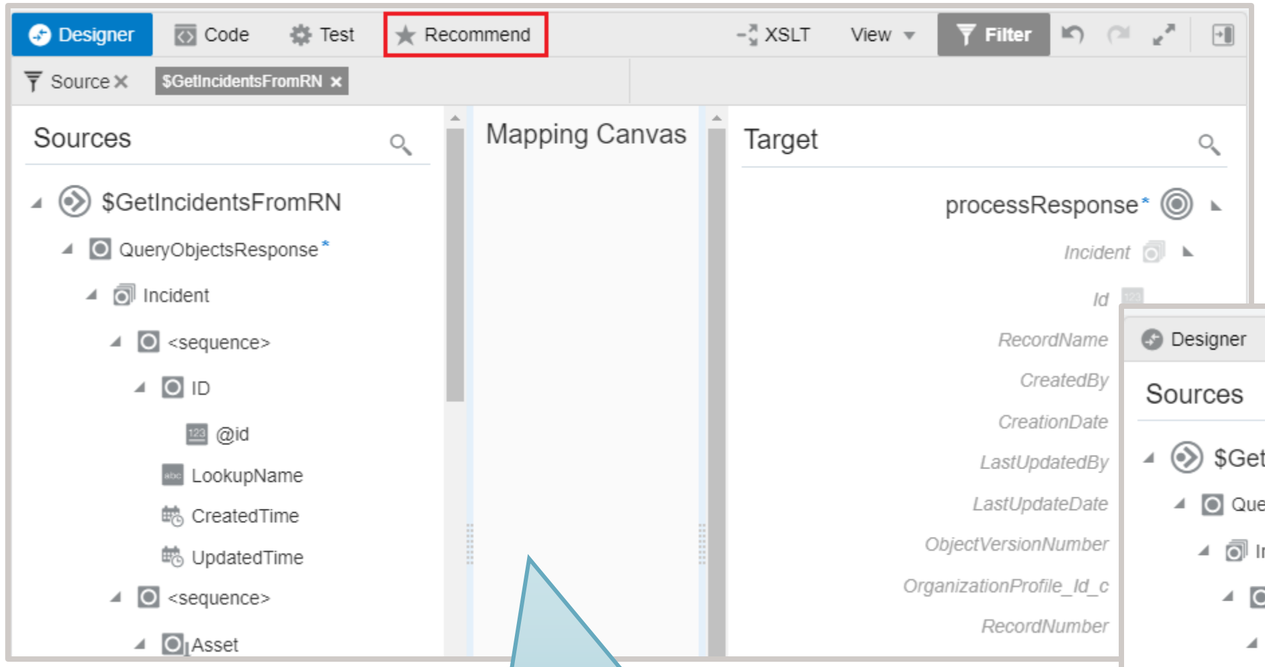
By default, the recommendations engine is enabled. When enabled, all integrations on that instance are published to the recommendations engine.



However, you can deselect this option for individual integrations upon activation.

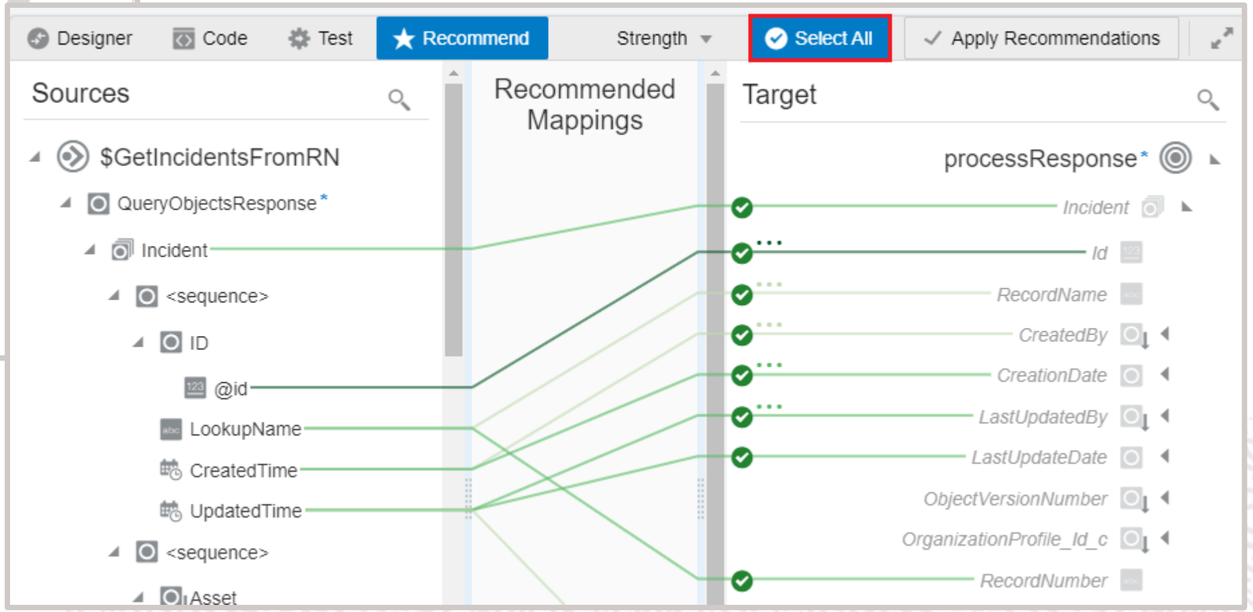


Launching the Mapper Recommendations



Open the mapper, and then click the **Recommend** tab

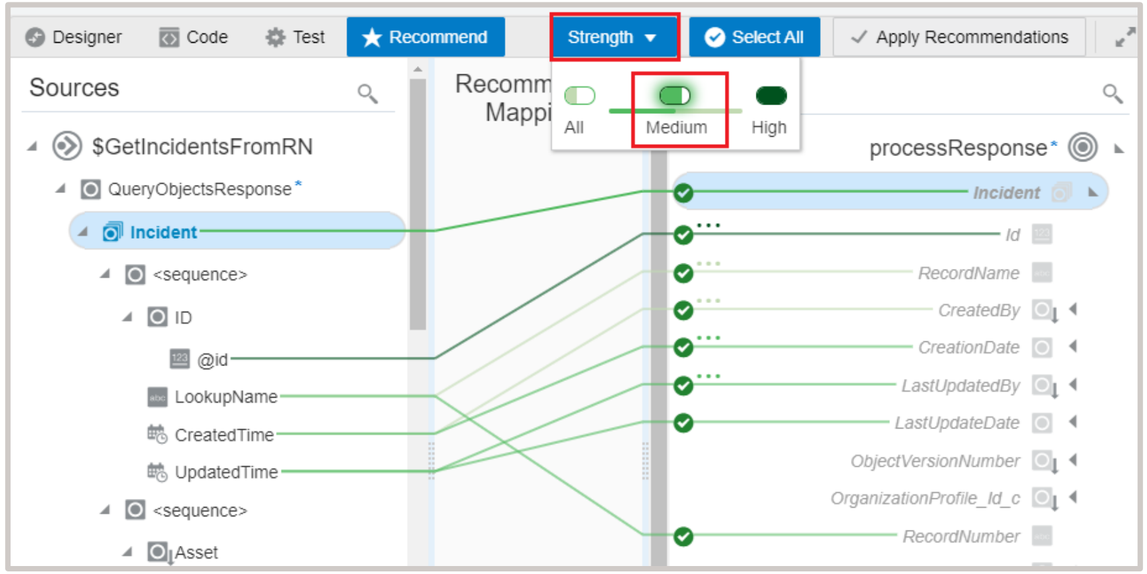
By default, all recommendations are selected.

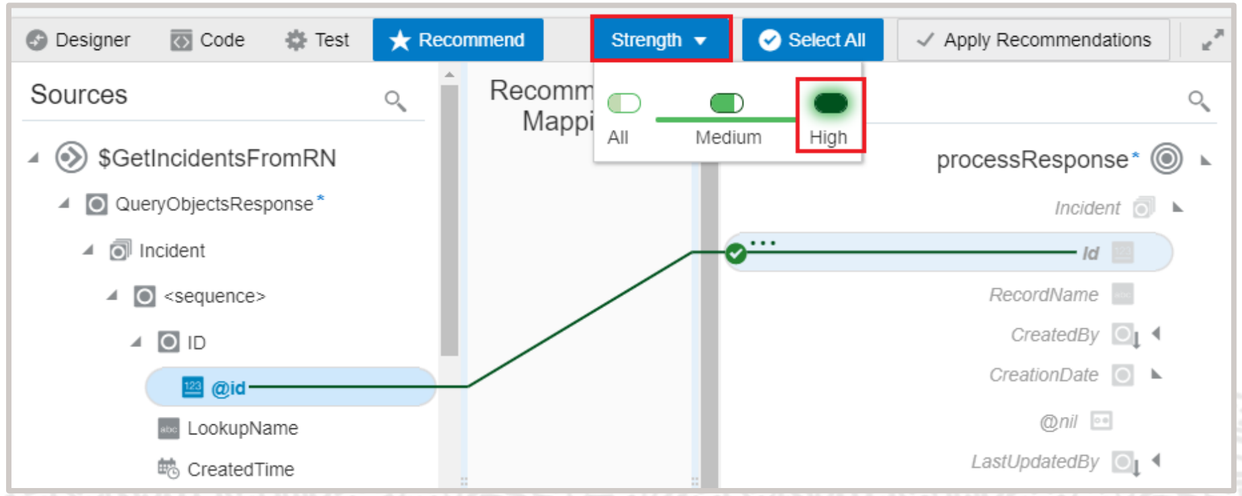


Displaying By Strength

Recommendations are categorized by three strength levels: Low, Medium, High.

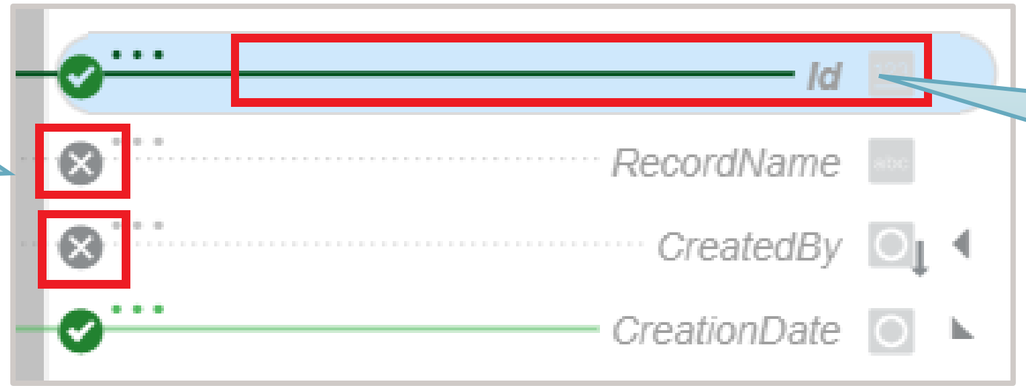
- Select Medium or High to remove the display of lower strength recommendations.



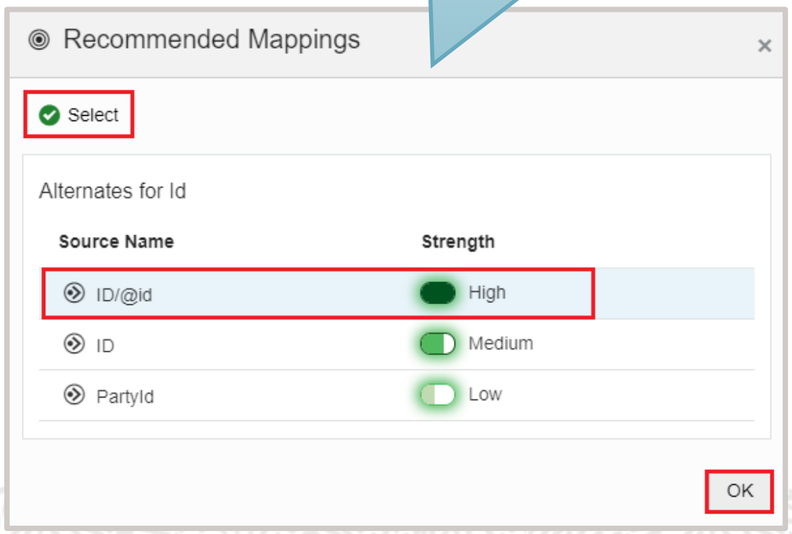


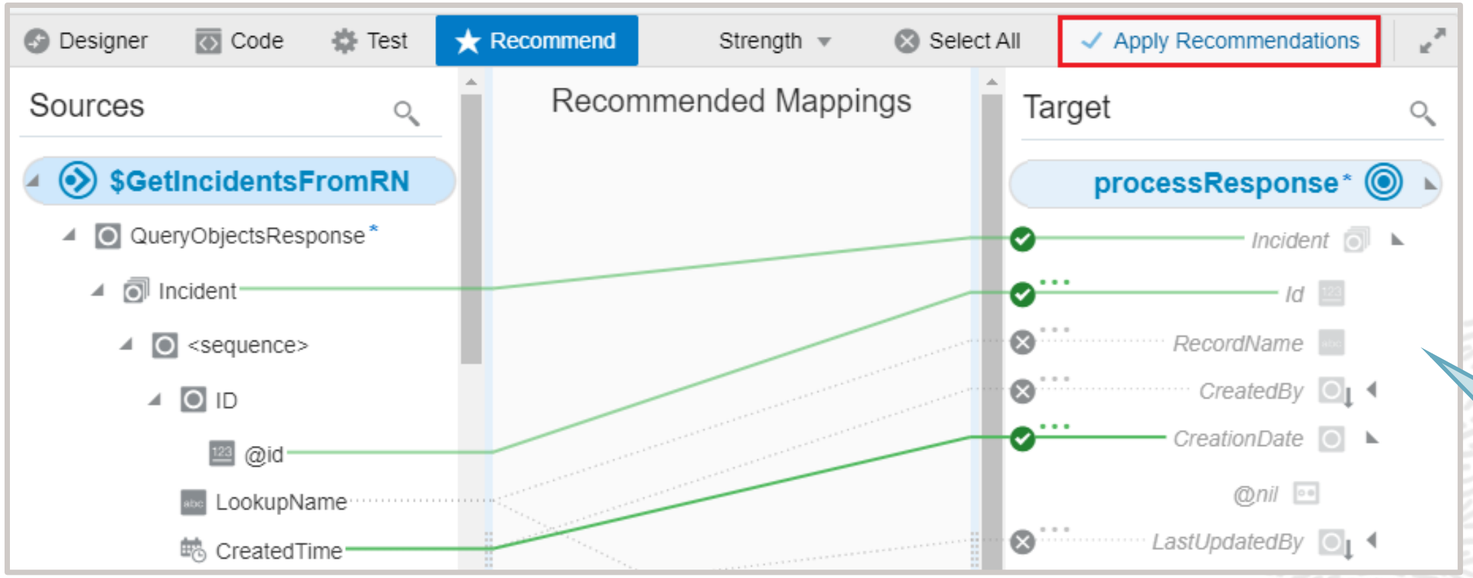
Selecting and Accepting Recommendations

Click a target icon to deselect or reselect a recommendation…



…or open the *Recommended Mappings* box to select recommended alternatives.



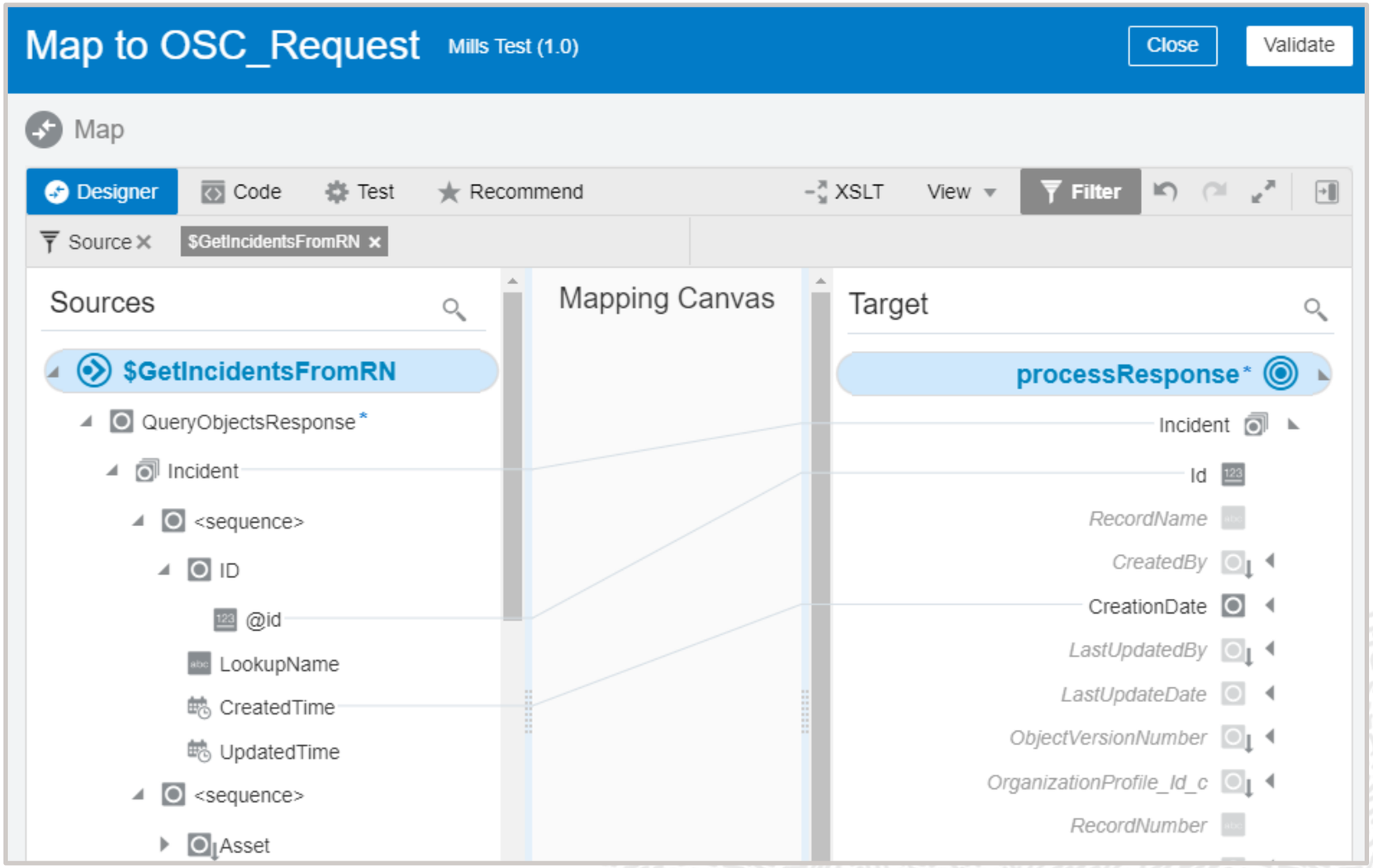


When finished, click **Apply Recommendations** to accept and apply selected mappings.

Complete the Data Mapping

Selected recommendations are applied and the mapper returns to **Designer** view

Continue to edit additional mappings (*as required*), then click Validate to save.



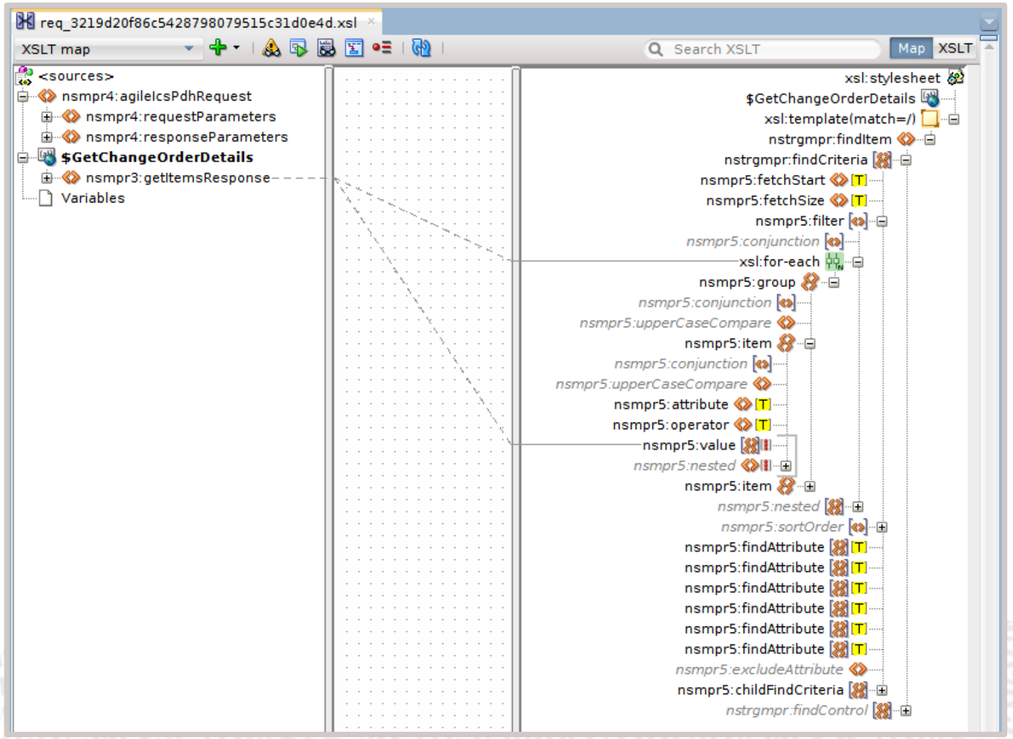
Agenda

- OIC Data Mapper
- OIC Recommendations Engine
- Advanced Transformation Options
- Using OIC Lookups

Creating Transformations Outside of OIC

Why might you choose to edit a transformation outside of the OIC Mapper tool?

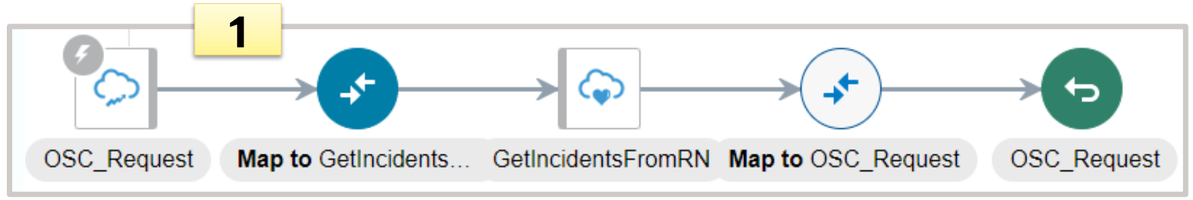
- Some mappings can be done faster using an external tool.
  - “If” and “choose-when-otherwise” conditionals
  - Large sections of similar/repeating elements
- More advanced functions and capabilities *(not supported in the OIC Mapper)* can be implemented.
  - Variables
  - Advanced selectors
  - Templates

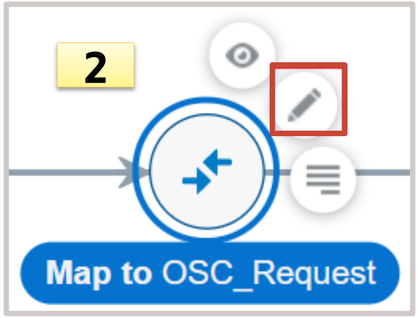


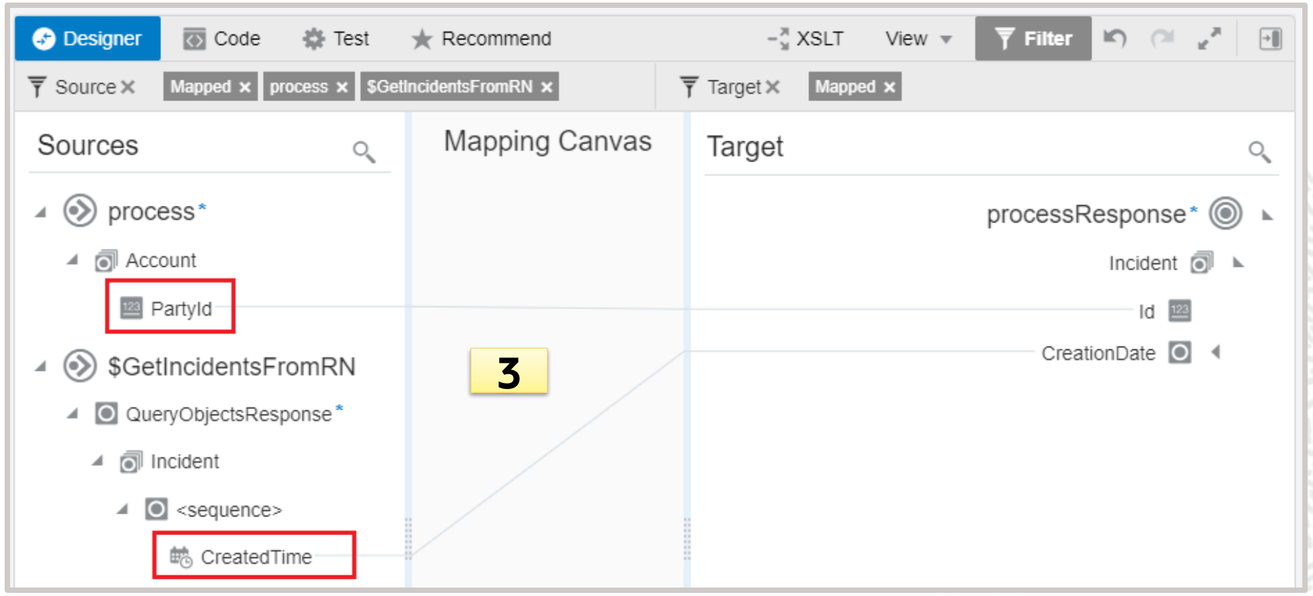
Preparing for XSL Export

Prior to export, complete the following:

1. Design the integration flow logic.
2. Open the “empty” Data Mapper.
3. Map one or more data values from each source schema to the target schema. *This provides visibility for all required schemas in the exported XSL.*

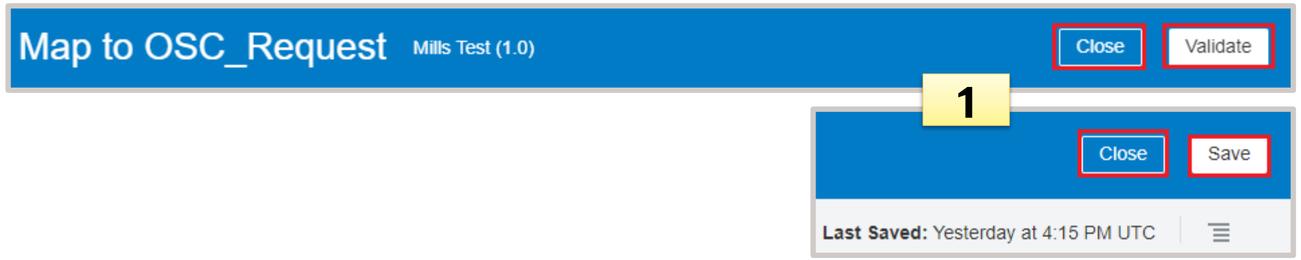


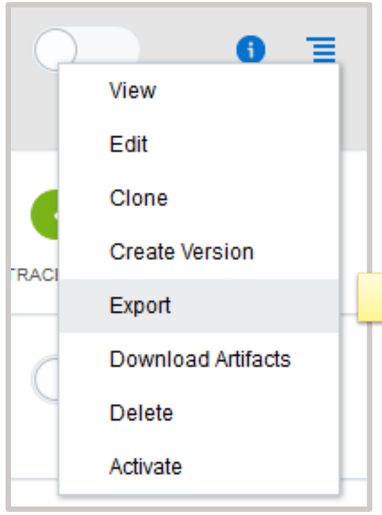




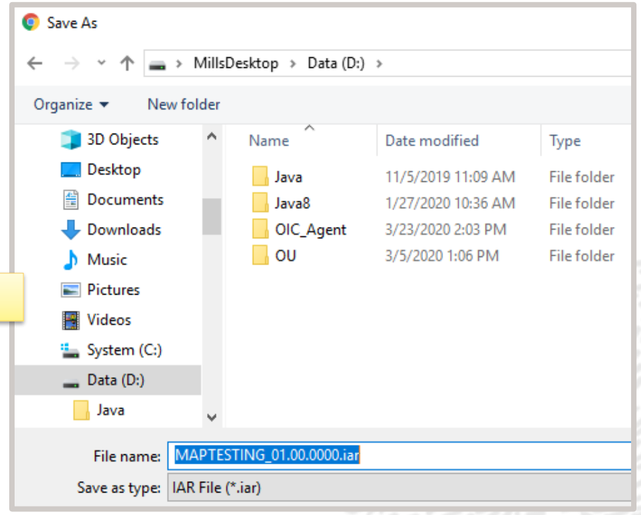
Exporting the XSL File

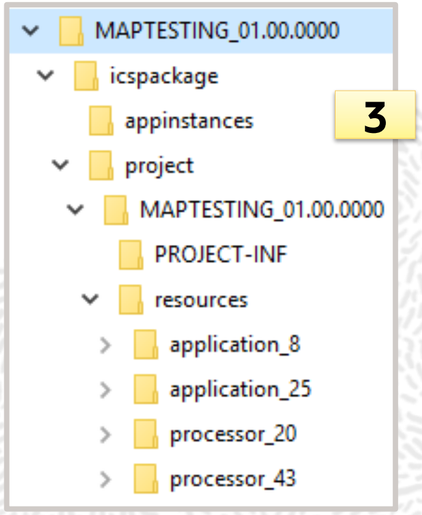
1. Prior to export, complete the following:
   - Validate and then close the mapper.
   - Save and then close the integration.
2. Export the entire integration.
3. Extract the .iar archive *(if you wish to manually view the file).*





2

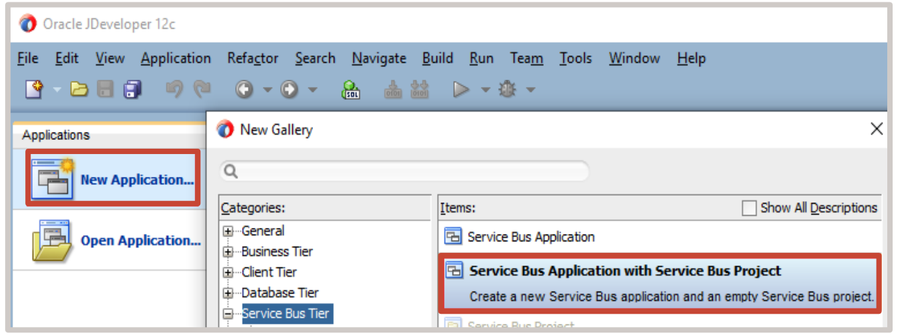




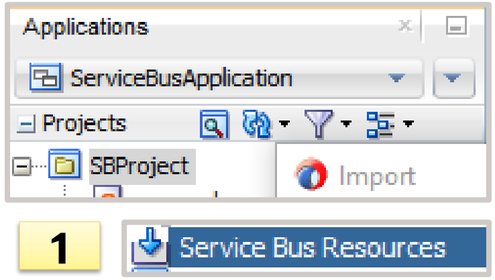
3

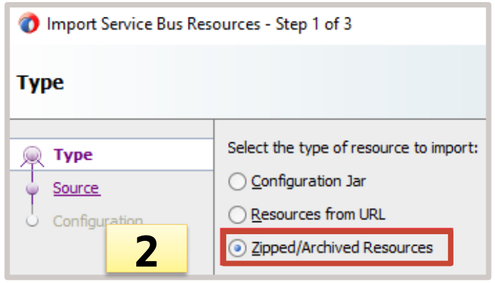
Setting Up JDeveloper to Edit the XSL File

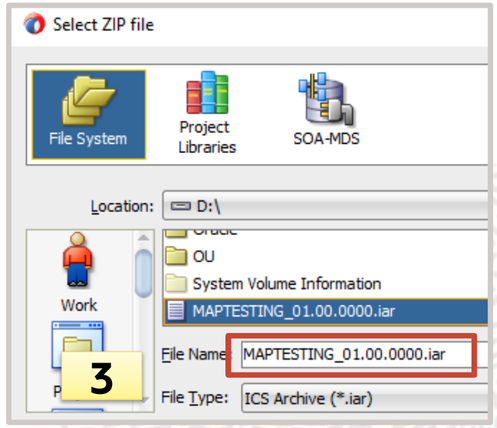
- Create an Oracle Service Bus application and project in Oracle JDeveloper.

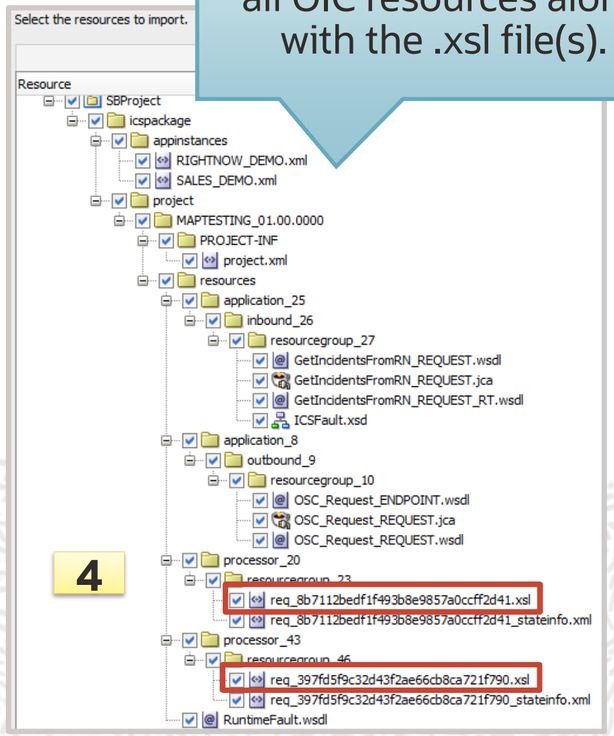


- Import the integration archive to the OSB project.







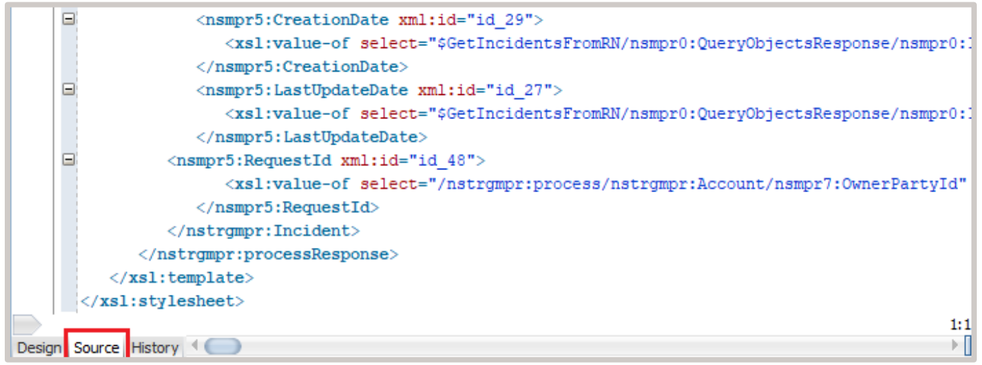


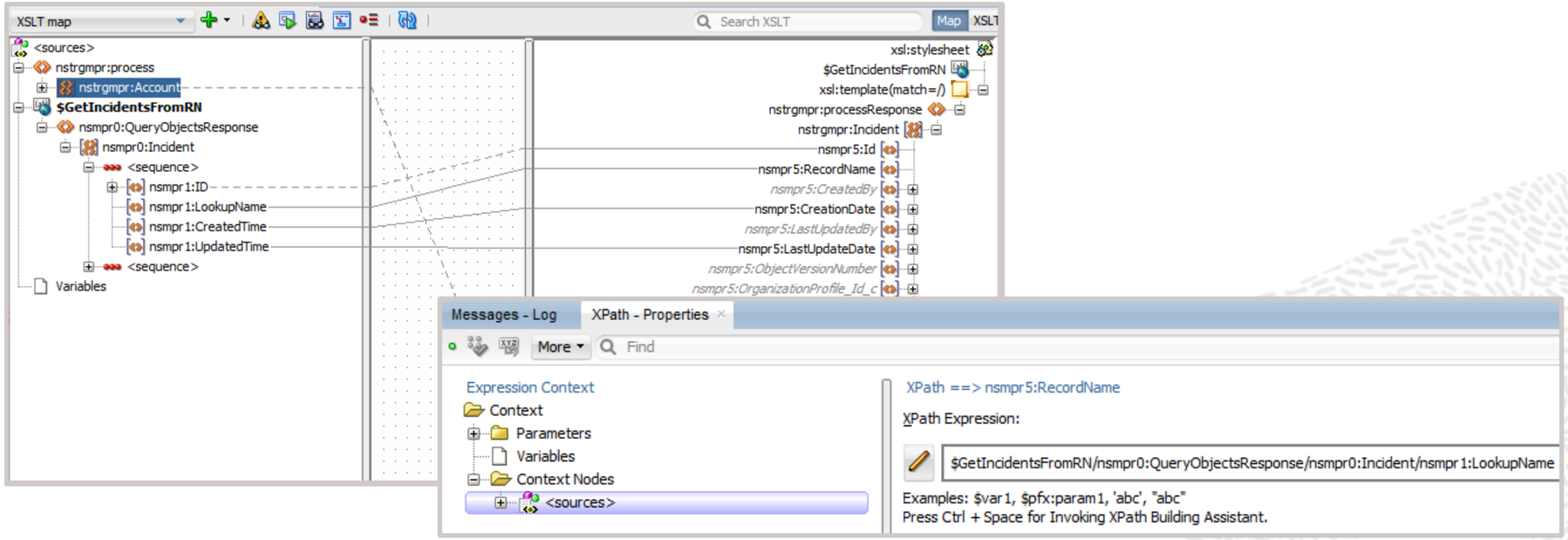
Best practice is to import all OIC resources along with the .xsl file(s).

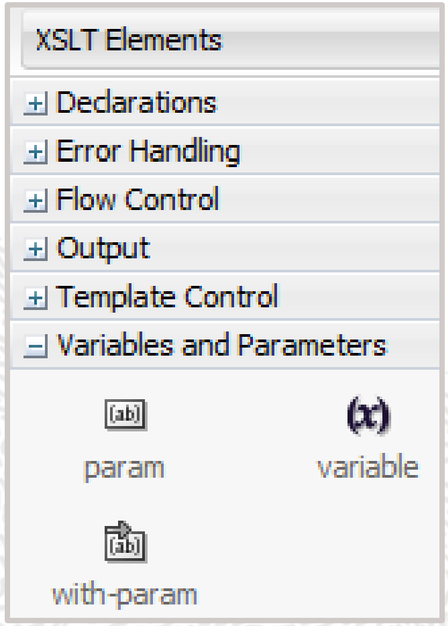
Using the XSLT Editor

Locate and open to edit the .xsl file:

- Use the **Design** editor or edit the **Source** directly.
- Switch between the **Map** view and the **XSLT** view.
- Leverage any of the XSLT elements.

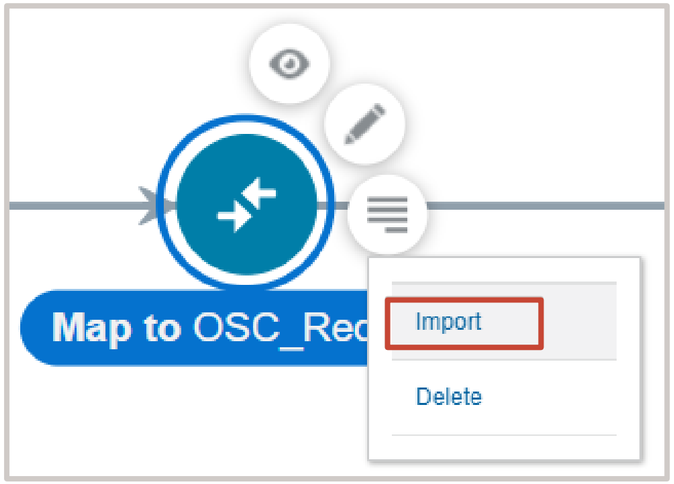


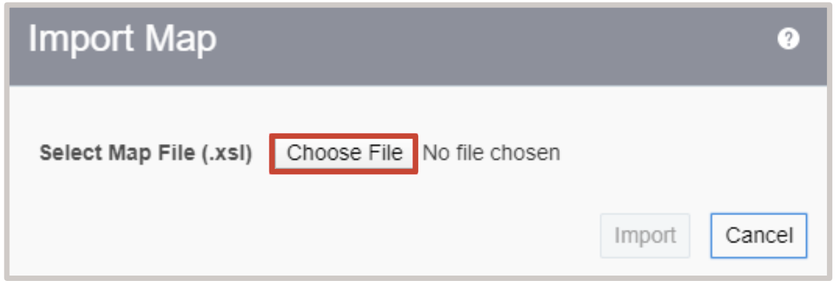




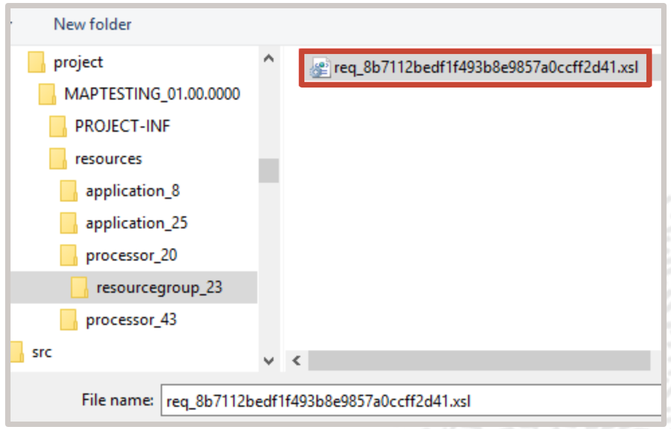
Import the Edited XSL File into OIC

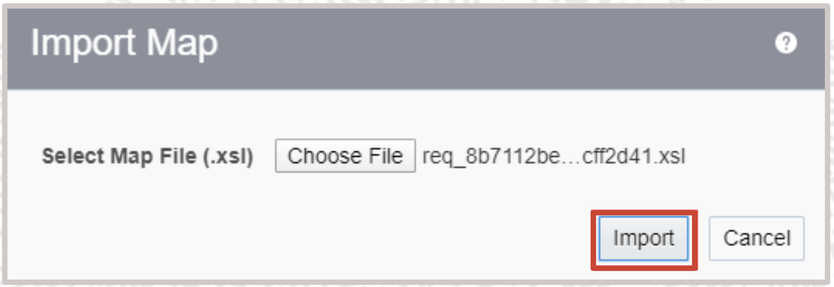
- Save the edited transformation.
- Return to the OIC Map action and click **Import**.





The mapper is now locked (see notes section).

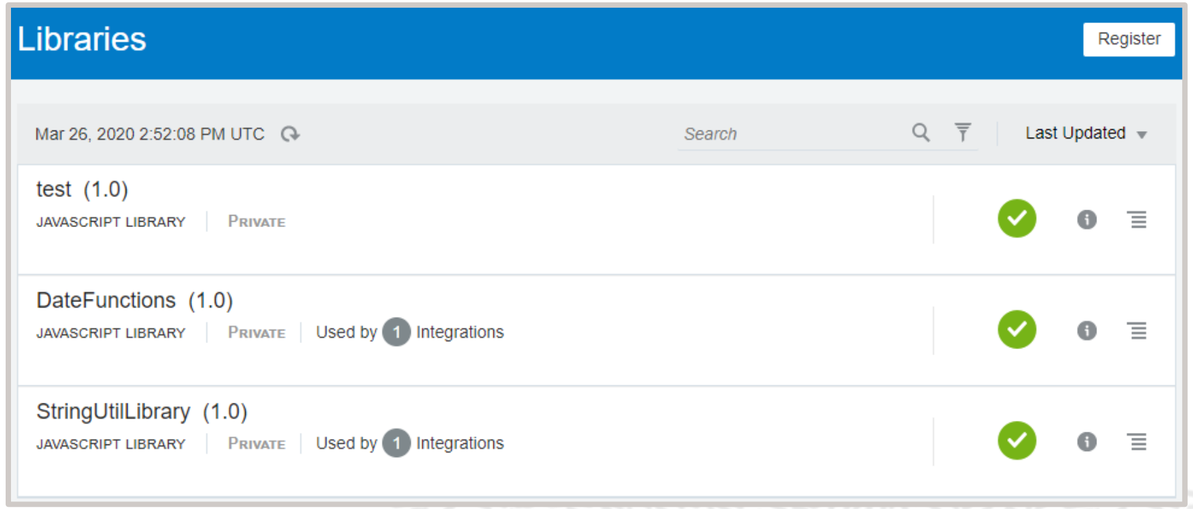




Using Custom JavaScript Functions

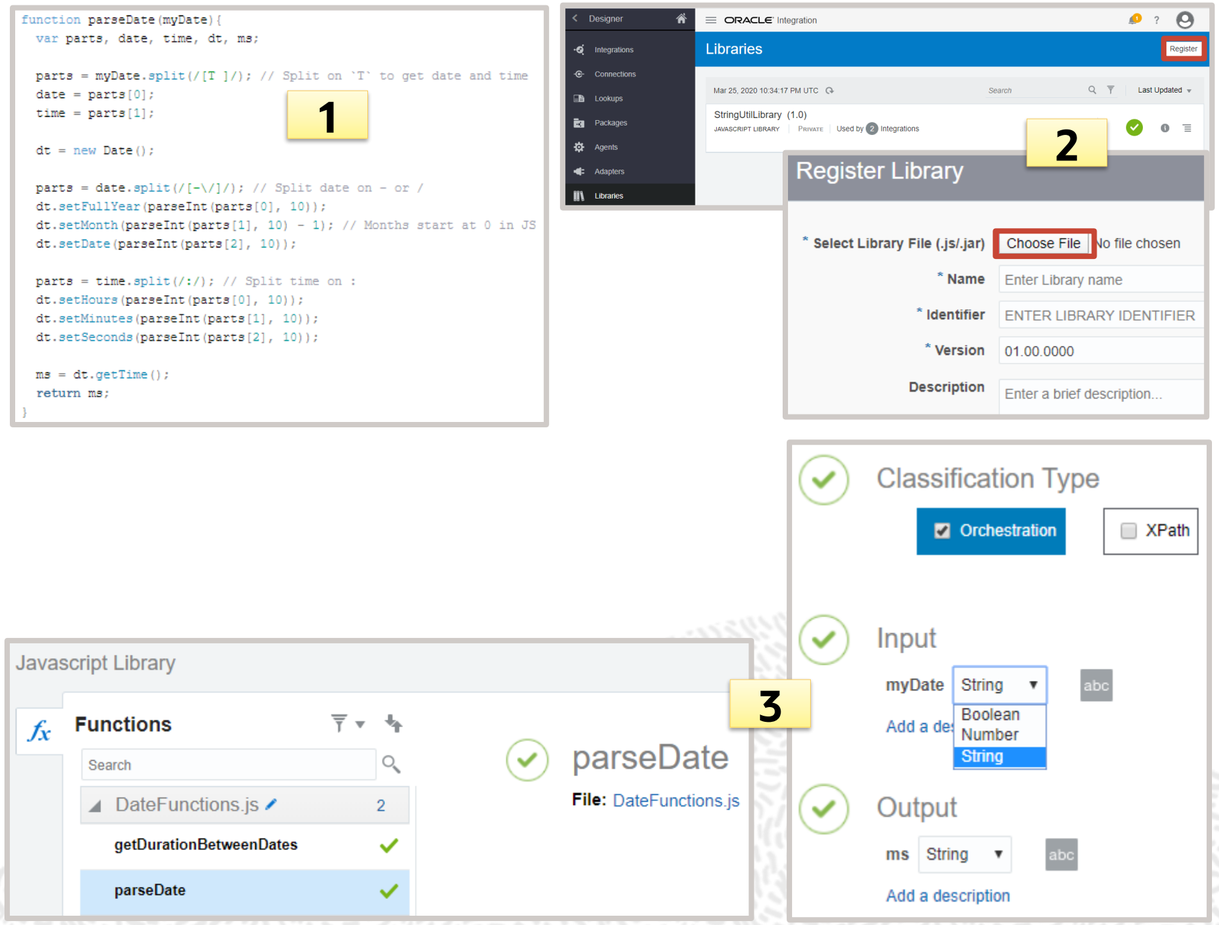
More advanced transformation requirements can be facilitated by using custom functions written in JavaScript. Two options are offered in OIC:

- Register as a custom XPath function
  - Added to the list of XPath functions in a **User Defined** folder
  - Available within any XSLT statement
- Register as a JavaScript function call
  - Available within any Orchestration integration flow



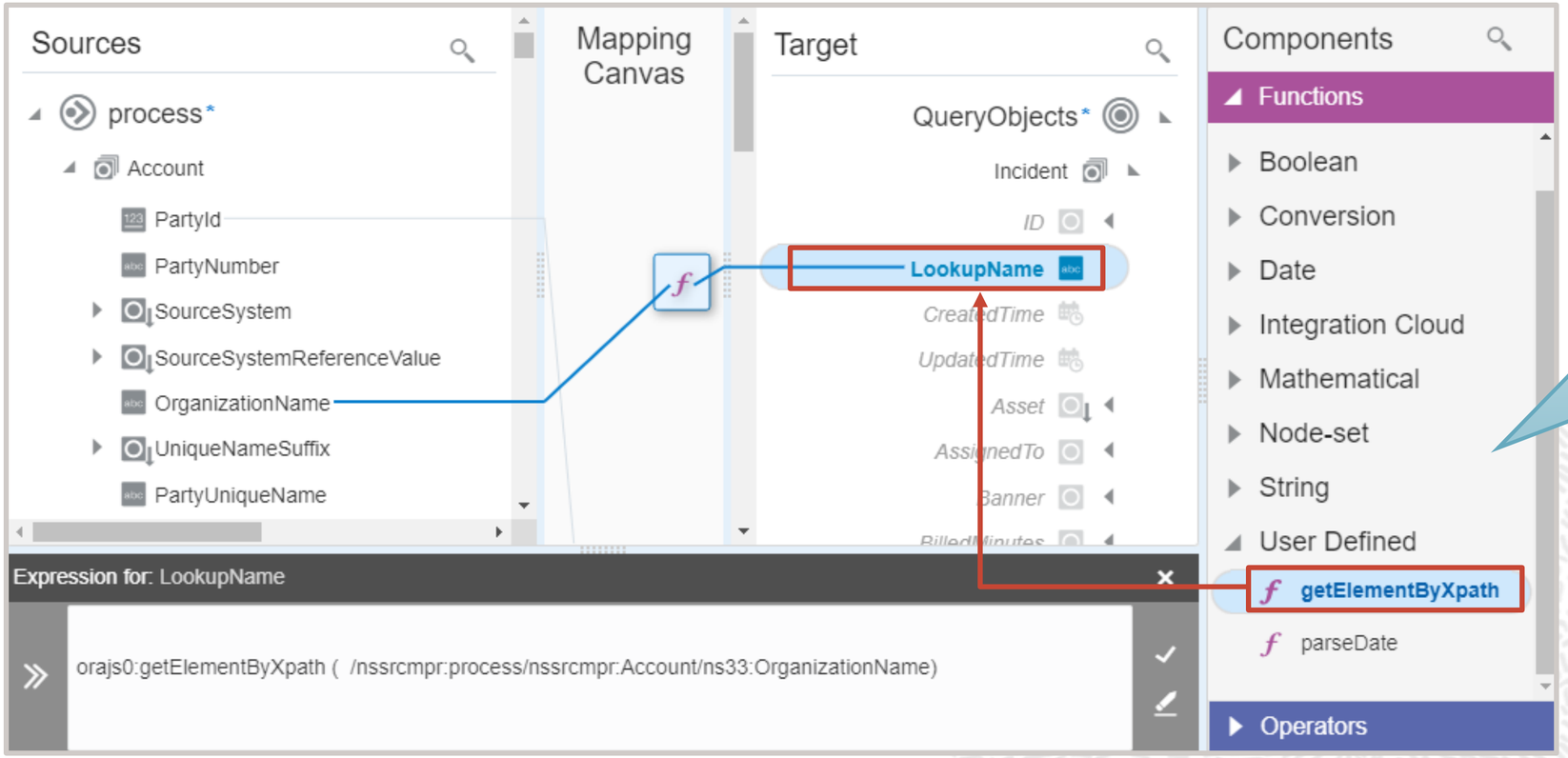
Adding JavaScript Libraries

1. Create new custom JavaScript functions
   - Save in one or more separate .js files
2. Register a new library in OIC
   - Upload a **.js** file or a **.jar** of multiple files
3. Define each function within the library
   - Classify as **Orchestration** or **XPath**
   - Define data types for **Input** and **Output** parameters
   - Optionally define parameter descriptions



Using Custom XPath Functions in the OIC Mapper

- Locate the custom function and drag and drop onto the target element.
- Provide source arguments as required.

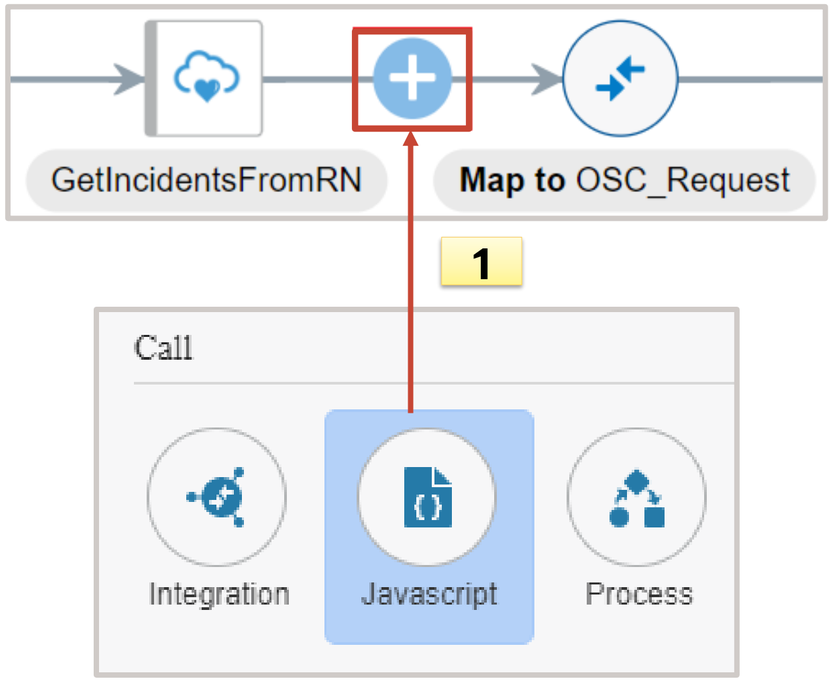


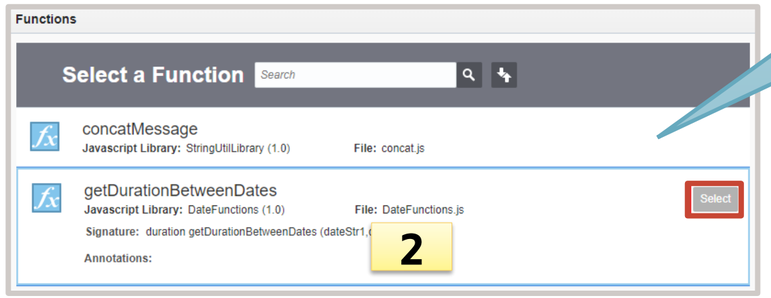
All library functions that have been classified as *XPath* will appear in the **User Defined** category.

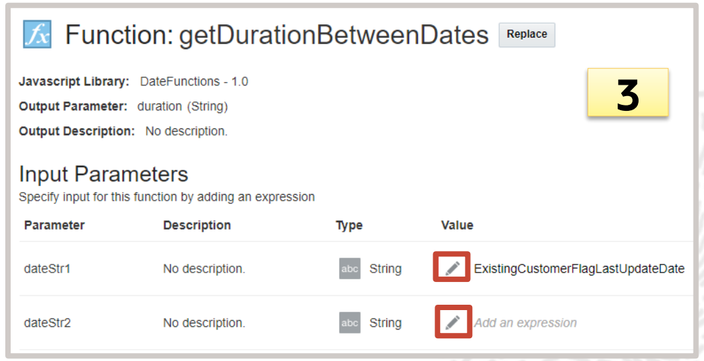
Executing Orchestration JavaScript Functions

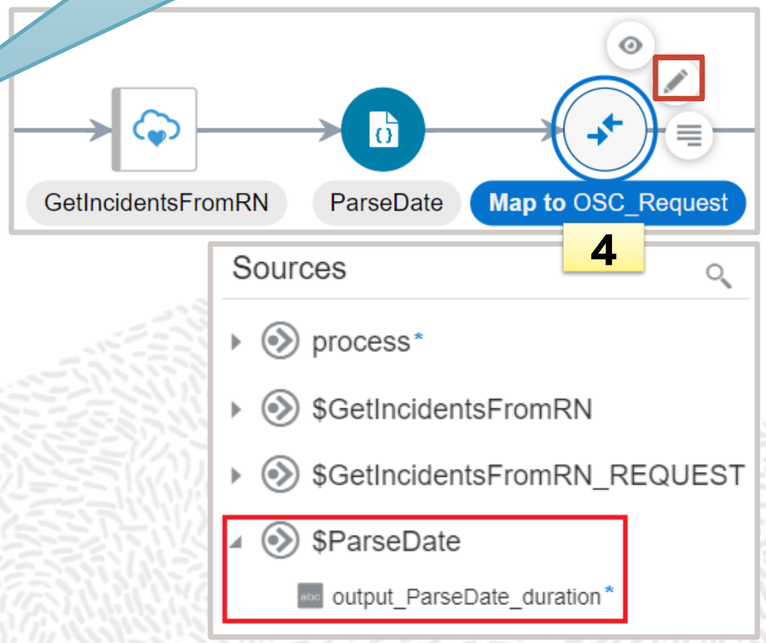
1. Drag the **Javascript** Call action to the integration flow.
2. Select the custom function.
3. Use the Expression Builder to define parameter values.
4. The return value is now available as a Source data element.

All Library functions that have been classified as **Orchestration** will appear in the list of callable functions.









Agenda

- OIC Data Mapper
- OIC Recommendations Engine
- Advanced Transformation Options
- Using OIC Lookups

The Challenge

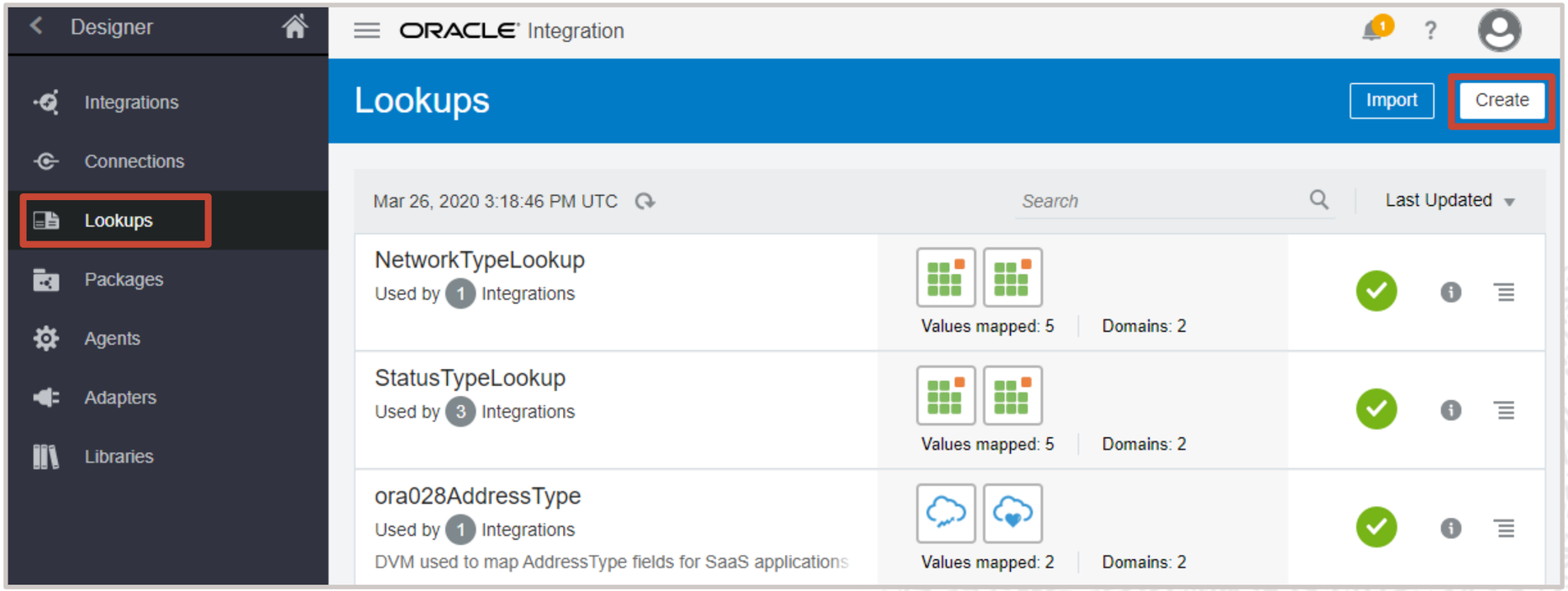
Information	System A	System B
Organization identifier	ID	OrgID
Organization name	Name	OrgName
…	…	…

OIC Lookups

- Use lookups to create in-memory “tables” that map the different terms used to describe the same item across your applications.
- Lookups are:
  - Reusable
  - Based on static definitions
    - Values are specified at design time.
    - Value map tables are looked up for values at run time.
- Example: Country Codes

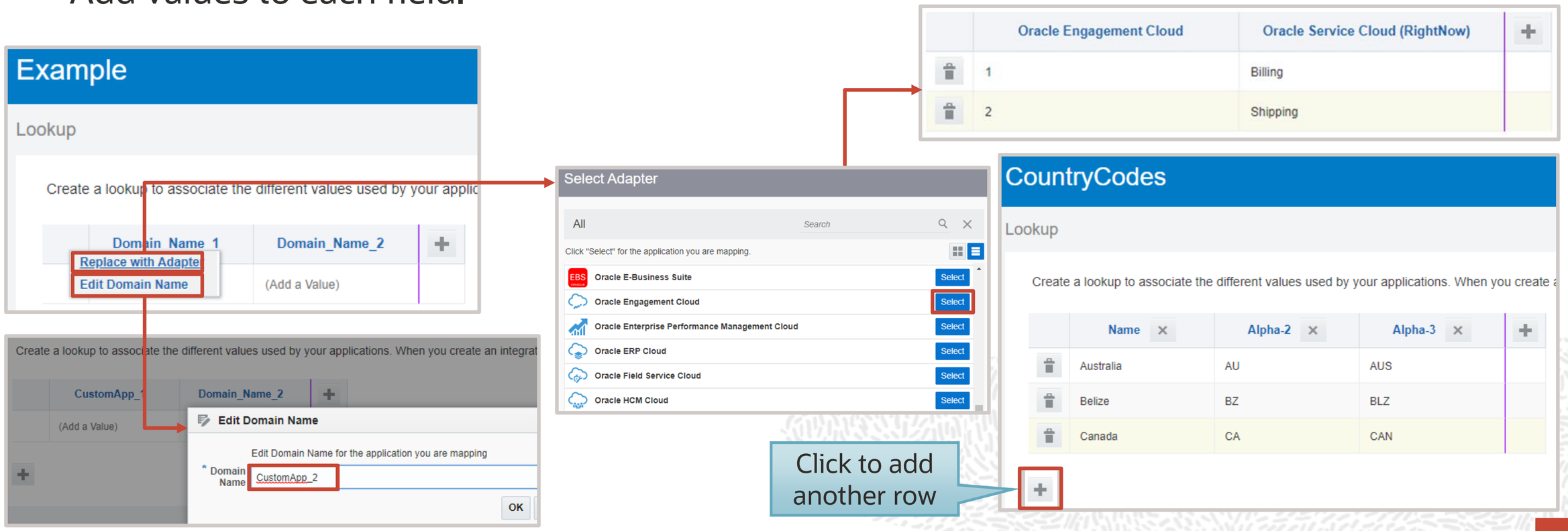
Getting Started

- On the toolbar, click **Designer**.
- On the Designer Portal, click **Lookups**.
- Click **Create**.



Creating Lookups

- Select the adapter type of the on-premises or SaaS application.
  - Alternatively, provide a unique domain name for the application or data format type.
- Add values to each field.



Editing Lookups

- Lookups can be cloned or exported to a CSV file for easy reuse.
- CSV files can be edited and then imported back to OIC.

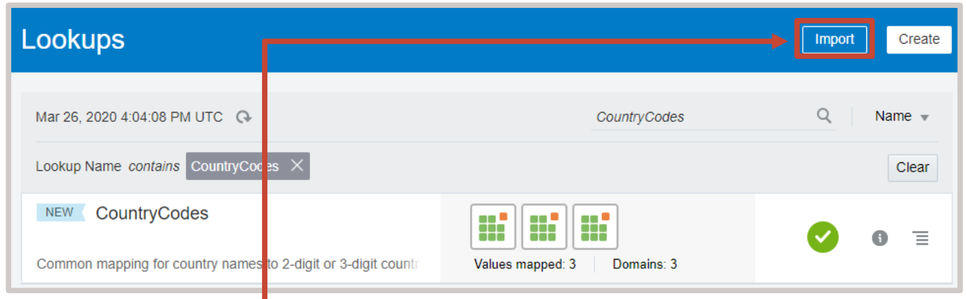
	A	B	C
1	DVM	CountryCo	Common
2	Name	Alpha-2	Alpha-3
3	Australia	AU	AUS
4	Belize	BZ	BLZ
5	Canada	CA	CAN

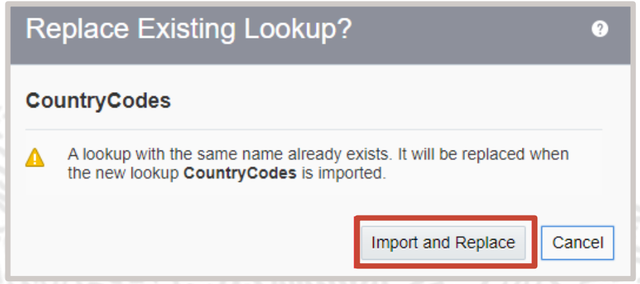
3 rows

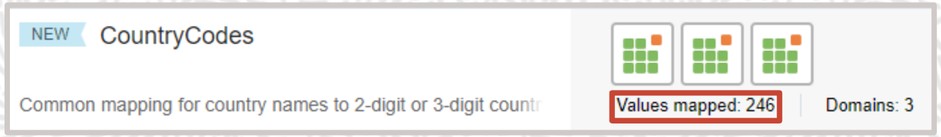
Edit CSV

	A	B	C
1	DVM	CountryCo	Common
2	Name	Alpha-2	Alpha-3
3	Algeria	DZ	DZA
4	American	AS	ASM
5	Andorra	AD	AND
6	Angola	AO	AGO
7	Anguilla	AI	AIA
8	Antarctica	AQ	ATA
9	Antigua a	AG	ATG
10	Argentina	AR	ARG
11	Armenia	AM	ARM
12	Aruba	AW	ABW
13	Australia	AU	AUS
14	Austria	AT	AUT
15	Azerbaija	AZ	AZE
16	Bahamas	BS	BHS

246 rows



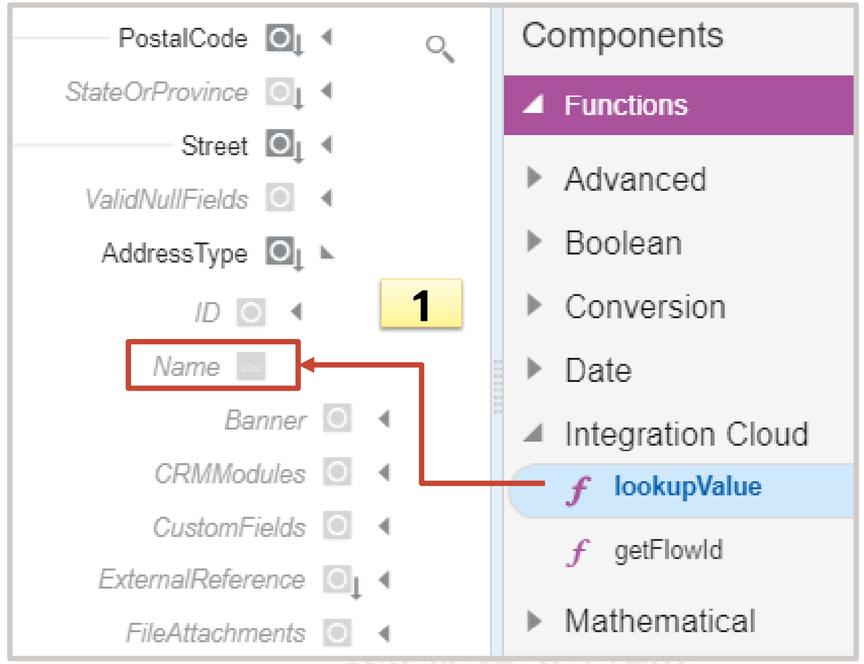


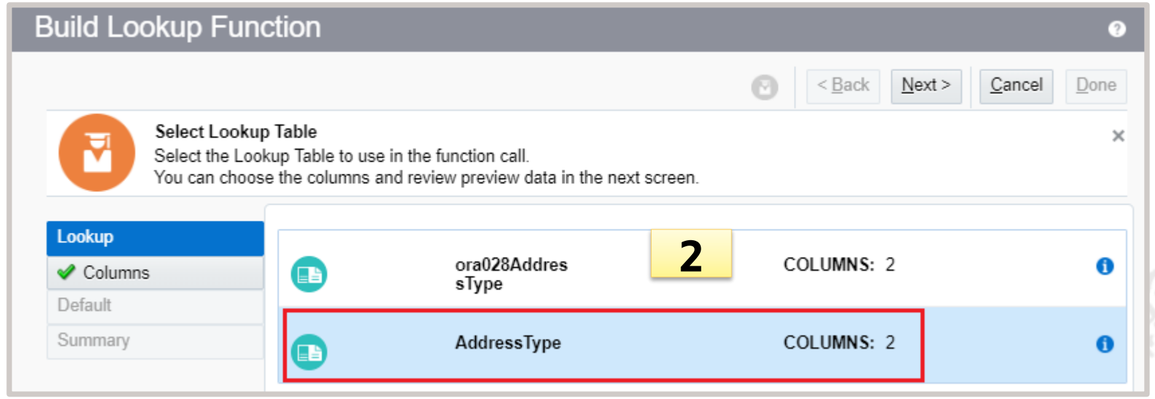


Changes made to lookups already used in active integrations take effect immediately *(within 2 minutes)*. There is **no** need to re-activate integrations using a changed lookup.

Referencing Lookups in the OIC Data Mapper

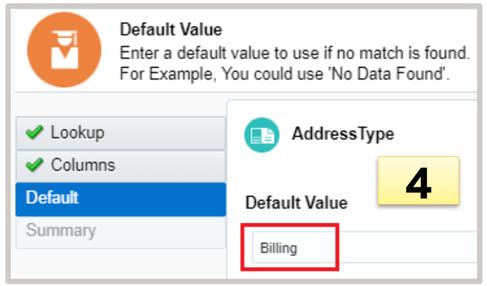
1. Locate the **lookupValue** function and drag and drop onto the target element.
   - This will launch the **Build Lookup Function** wizard.
2. Select the lookup and click *Next*.
3. Choose the **Source** and **Target** Columns.

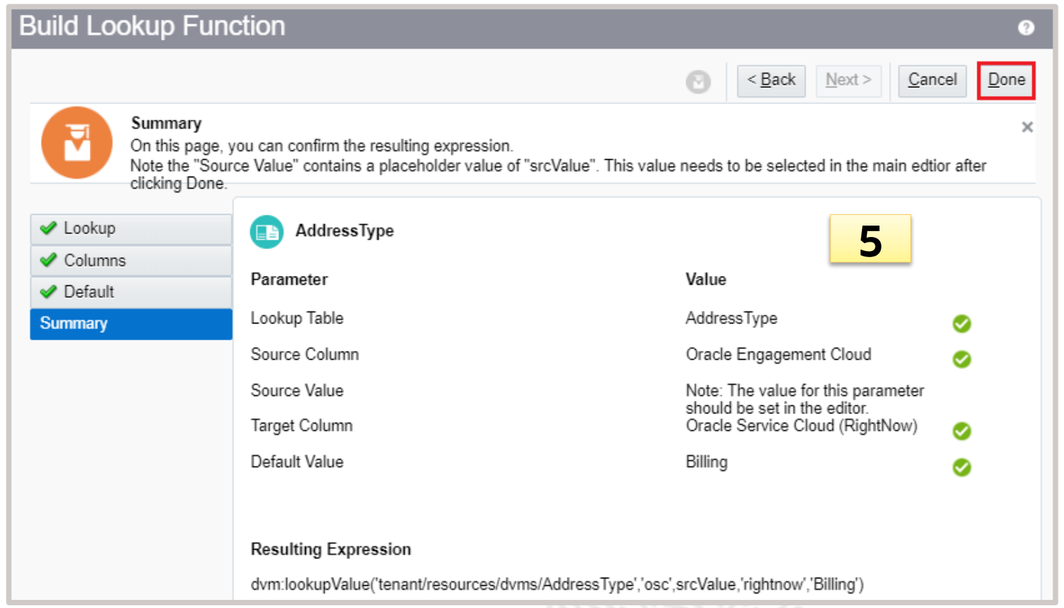


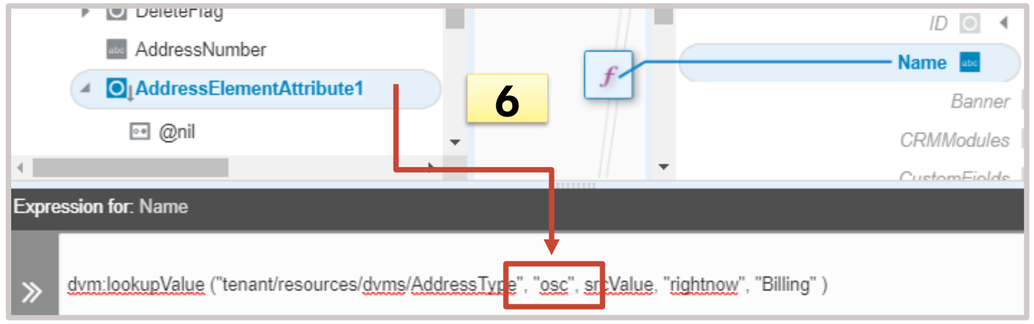


Referencing Lookups in the OIC Data Mapper

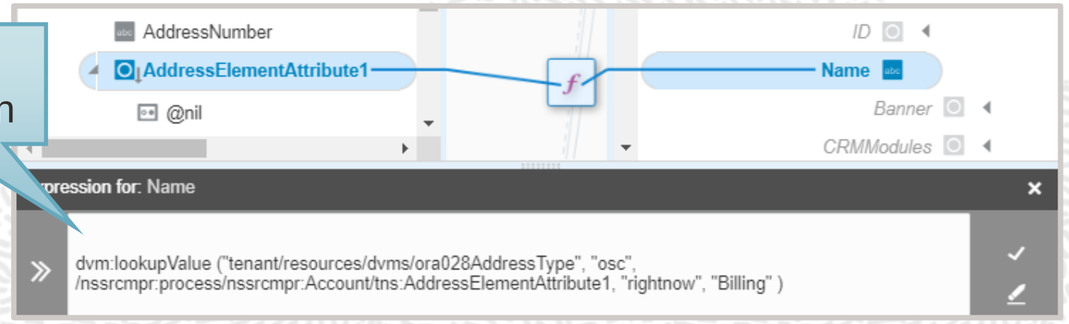
4. Provide a default value.
5. Review the Summary page and click *Done*.
6. Locate and map the Source value to the third parameter and click the *Save* check mark.







Final expression



Summary

In this lesson, you should have learned how to:

- Map or transform data by using the OIC Mapper tool
- Create XSL expressions using XSLT functions and operators
- Describe and use the OIC Recommendations Tool
- Edit and import advanced XSL files into OIC
- Register and use custom JavaScript functions
- Create and invoke OIC Lookups

Practice 8-1: Creating Pub/Sub Integrations with Lookups

This practice includes:

- PART 1 – Creating a Lookup Component
- PART 2 – Creating the Publish Integration Flow
- PART 3 – Creating the Subscribe Integration Flow
  - Configuring Data Mapping using a Lookup function
- PART 4 – Testing the Publish and Subscribe Integrations

Creating a Subscriber Integration (Review)

Once the Publisher integration has been configured, you can create the Subscriber integration to process the message.

- Click the Map icon and then click the + Create icon to launch the OIC Data Mapper.