Oracle® Cloud

Using the Oracle Mapper with Oracle Integration
Contents

Preface

Audience v
Documentation Accessibility v
Related Resources v
Conventions vi

1 Get Started with the Mapper

About Mappings 1-1
About Mapping Data Between Applications 1-3
Build Expressions with the Mapping Builder 1-4

2 Map Data

Create Mappings 2-1
Accept Mapping Recommendations with the Recommendations Engine 2-2
   Enable the Oracle Recommendations Engine 2-2
   Accept Target Element Mapping Recommendations 2-2
Repeat a Target Element to Map to Different Sources 2-5
Create for-each Statements Automatically 2-7
Delete Individual Mappings 2-9
Find Data Fields 2-10
Browse and Filter the Source or Target Data Structure 2-11
Map Multiple Source Structures to a Target Structure 2-13
Deep Copy Parent Nodes from Source to Target 2-14
Set Default Values in the Mapper 2-15
Test Your Mappings 2-16

3 Build Expressions

Build Mapping Statements 3-1
Repeat Elements with the for-each Statement 3-2
Convert a Comma-Separated String into a List of Entries with a for-each XSL Element 3-4
Create Conditional Mappings 3-4
Use XPath Functions 3-5
  Support for BPEL XPath Extension Functions 3-6
  Use the length Parameter in the Substring Function in the Mapper 3-7
  Add a New Line Between Variables in a concat Operation 3-7
Reference Lookups 3-9
Create the lookupValue Function 3-11
  Access the Build Lookup Function Wizard 3-11
  Select the Lookup Table 3-12
    What You Can Do from the Select Lookup Table Page 3-12
    What You See on the Select Lookup Table Page 3-12
  Select the Source and Target Columns 3-12
    What You Can Do from the Source and Target Columns Page 3-13
    What You See on the Source and Target Columns Page 3-13
Specify the Default Value 3-13
  What You Can Do from the Default Value Page 3-13
  What You See on the Default Value Page 3-13
Review Your Lookup Table Selections 3-14
  What You Can Do from the Lookup Table Summary Page 3-14
  What You See on the Lookup Table Summary Page 3-14
Delete Mapping Statements 3-15
Preface

*Using the Oracle Mapper with Oracle Integration* describes how to use the mapper to map source data structures to target data structures.

**Note:**

The information in this guide applies to all of your Oracle Integration instances. It doesn’t matter which edition you’re using, what features you have, or who manages your cloud environment. You’ll find what you need here, including notes about any differences between the various flavors of Oracle Integration when necessary.

**Topics**

- Audience
- Documentation Accessibility
- Related Resources
- Conventions

**Audience**

*Using the Oracle Mapper with Oracle Integration* is intended for users who want to use the mapper to map source data structures to target data structures.

**Documentation Accessibility**


**Access to Oracle Support**

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit [http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info](http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info) or visit [http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs](http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs) if you are hearing impaired.

**Related Resources**

See these Oracle resources:
• Oracle Cloud
  http://cloud.oracle.com
• Using Integrations in Oracle Integration

Conventions

The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td>monospace</td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
Get Started with the Mapper

Review the following topics to learn how the mapper works to map source data structures to target data structures.

Topics

- About Mappings
- About Mapping Data Between Applications
- Build Expressions with the Mapping Builder

About Mappings

One of the key tasks to any integration is defining how data is transferred, or mapped, between two applications.

In most cases, the messages you want to transfer between the applications in an integration have different data structures. A visual mapper enables you to map fields between applications by dragging source fields onto target fields. When you open the mapper for a request or response message in an integration, the data structures are automatically populated with the information pulled from the source and target connections. You can expand and load data structure levels on demand to display additional levels. There is no limit on the levels of display.

The maps you create are called transformation maps, and use the eXtensible Stylesheet Language (XSL) to describe the data mappings, which lets you perform
complex data manipulation and transformation. A standard set of XPath functions are provided for you to define how data is modified when moving from one application to another. A specialized function is also provided for you to reference lookups directly from the mapper.

The mapper supports both qualified and unqualified schemas (that is, schemas without elementFormDefault="qualified"). Elements and attributes with and without namespace prefixes are also supported.

Substitution groups in schemas are supported. You can see all the substitutable elements in a base element in the mapper, and select the one to use.

Elements and attributes for which mapping is required are identified by a blue asterisk (*) to the left of their names. To display only required fields, click the Filter icon, select Required Fields, and click Apply.

You can also place your cursor over elements and attributes to display specific schema details such as the data type, if mapping is required, and so on. When you place your cursor over some elements and attributes, additional custom annotations can also be displayed. These annotations are currently only available with the Oracle Sales Cloud Adapter. The Oracle Sales Cloud Adapter obtains this information from the applications and annotates it in the integration WSDL. This information is then read and made visible as annotations in the mapper (for example, title and description). This information can help you better understand what data is being mapped.
About Mapping Data Between Applications

Once you create an integration and have the source and target connections in place, you can define how data is mapped between the two data structures.

The mapper appears with the source data structure on the left and the target data structure on the right.

1. To map fields directly, click a field in the source and drag it to the corresponding field in the target.
The name of the source field appears in the target **Mapping** column, and a green check mark icon appears next to both fields. The most recently mapped fields are connected by a green line. Click the green check mark of other sources and targets to see their current mappings.

2. To use XPath functions in your mapping, see **Using XPath Functions** for instructions.

3. To use lookups in your mapping, see **Referencing Lookups** for instructions.

4. When you are done mapping data, click **Close**, then click **Apply** to save your changes.

5. On the toolbar, click **Save**.

6. Click the map icon between the response operations, and repeat the above steps to map the response. Make sure to save the integration when you are done.

You can also configure the primary tracking variable and both custom field tracking variables (update and access values) in orchestrated integrations in the mapper. You can map tracking variables to output variables or create complex expressions for an assign or switch activity. The primary and two custom parameters are available in the **From** palette, but only the two custom parameters are present on the **To** palette for the assign activity.

- All tracking variables are of type string (all that assignments support).
- All three tracking variable entries are present even if you choose not to model them. The name and XPath can be empty for tracking.
- Editing or deleting the tracking variables only updates the name and XPath nodes in that particular tracking variable element.
- The tracking variables have static names. Therefore, it is possible to set a tracking variable somewhere in the flow, but not initialize it with a value and a name in the Tracking dialog.
- The primary tracking variable cannot be assigned any value in between the flow.
- You cannot create a new variable with the same names as any of the statically name tracking variables.

**Build Expressions with the Mapping Builder**

Use the Mapping Builder to create and modify advanced mappings. For example, you can loop through repeating elements, create conditional statements, use XPath...
functions, and reference lookups. The Mapping Builder is displayed when you click a field in the target data structure of the mapper.

See Build Expressions.
2

Map Data

Use the intelligent data mapper to drag fields from the source structure to the target structure to map elements between the two.

Topics

• Create Mappings
• Repeat a Target Element to Map to Different Sources
• Create for-each Statements Automatically
• Delete Individual Mappings
• Find Data Fields
• Browse and Filter the Source or Target Data Structure
• Map Multiple Source Structures to a Target Structure
• Deep Copy Parent Nodes from Source to Target
• Test Your Mappings

Create Mappings

You can map fields directly from the source data structure to the target data structure in the mapper.

To create mappings:

1. In the middle of the integration, click the Mapper icon for the request, response, or fault map to edit.
2. Click Create.
3. To map fields directly, perform one of the following steps:
   a. Click a field in the source and drag it to the corresponding field in the target.
   b. Click the source and the target fields, and then click Map+.

The name of the source field appears in the target Mapping column, and a green check mark icon appears next to both fields. The most recently mapped fields are connected by a green line. Click the green check mark of other sources and targets to see their current mappings. You cannot drag and drop onto a target that is already mapped.
Accept Mapping Recommendations with the Recommendations Engine

You can accept the target element recommendations of the recommendations engine when creating mappings. This eliminates the need to analyze and perform each individual source-to-target mapping. The findings of the recommendations engine are particularly useful when you have a new integration in which mapping has not yet been created. You can also use the recommendations engine with previously-created mappings.

Topics

- Enable the Oracle Recommendations Engine
- Accept Target Element Mapping Recommendations

Enable the Oracle Recommendations Engine

By default, the recommendations engine is not enabled. You must enable this feature. When enabled, all integrations on this instance are published to the recommendations engine.

To enable the recommendations engine:

1. In the navigation pane, click Integrations, then click the < arrow next to DESIGNER.
2. Click Settings > Recommendations.
3. Select the Publish integration mappings to Oracle Recommendations Engine check box, then click Save in the upper right corner.

Accept Target Element Mapping Recommendations

The mapper includes a recommendations engine for creating mappings. This eliminates the need to analyze and perform each individual source-to-target mapping. The findings of the recommendations engine are particularly useful when you have a
new integration in which mapping has not yet been created. You can also use the recommendations engine with previously-created mappings.

Note:

Ensure that you first configure the recommendations engine. See Enable the Oracle Recommendations Engine.

To use the mapping recommendations of the recommendations engine:

1. Go to the Integrations page, and find the integration in which to use the mapping recommendations of the recommendations engine.
2. Open the mapper.
3. Click Recommend in the upper right corner of the target tree.

A dialog shows the target tree node starting with the root node and then shows each of its child nodes with a recommendation. To maintain the hierarchy of the tree, the parent node of a child is displayed even though it has no recommendation. Parents have no selectable check boxes. Child nodes include a selectable check box. The number of recommendations is displayed at the bottom of the dialog. The Oracle Recommendations engine also indicates if there are no suggested recommendations.

4. Note the following additional recommendations features:
   - The Mapping Recommendation column displays the nodes for which recommendations are identified. If a target node has more than one recommended mapping, a Show Alternates link is displayed. Click the link to view and select from the available recommendations. Up to five alternates are displayed.
- The All Sourced By dropdown list by default displays the initial recommendations from all sources. You can filter by all sources or narrow your filtering by selecting from the following sources.
  - Oracle: Mapping recommendations that came from Oracle.
  - Customer: Mapping recommendations in which others have created similar mappings in their integrations.
- The Relevance column weighs the significance of the recommendation on a scale from one to five stars. The more stars, the more relevant the target recommendation. Place your cursor over the score to display a popup with information about the score, the source (Oracle, customer, or both), and usage count. Use the filter list to narrow the display by relevance (number of stars).

5. Review the recommendations and select as follows:
   - Select the check boxes of specific nodes that you want to map.
   - Select the Target check box in the upper left corner to automatically select all nodes.

   The Recommendations Selected value at the bottom of the page is increased.

6. Click OK to exit the recommendations dialog and return to the mapper.

   The recommendations you selected are displayed in the mapper.

7. Right-click an individual row of an unselected node in the mapper, and select Recommend. This action shows the recommendations for that specific node.

   A dialog is displayed with the recommendations for the selected node and any of its child nodes. If you select a parent, you receive recommendations for any of its child nodes that have not yet been selected. Any previously recommended nodes that you selected are not displayed.

8. Select the check box and click OK to add any recommendations.

   You are returned to the mapper.
9. When complete, click **Close** to exit the mapper. A message prompts you to save your mappings.

## Repeat a Target Element to Map to Different Sources

You can repeat a target element in the mapper. This enables 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.

To repeat a target element to map to different sources:

1. In the target data structure, right-click the element to repeat, and select **Repeat Element**. Elements that can be repeated are identified by a special icon with two bars to the left of the name. When you place your cursor over these elements, the words **Repeating Element** are displayed.

   ![Repeating Element Icon]

   The element is repeated and displayed below the existing element. Elements that are repeated show the count (for example, **(1 of 2)** for the existing element and **(2 of 2)** for the repeated element. You can repeat an element multiple times.

2. Expand the existing and repeated elements to see that the attributes in each element are repeated.
3. Click the child attribute of the existing element (for this example, named languageCode under EBOName (1 of 2) to access the Mapping Builder.

4. Replace Drag and Drop or Type here with a value (for this example, en is entered for English).

5. Click Save, then click Close.

6. Click the child attribute of the repeated element (for this example, named languageCode under EBOName (2 of 2) to access the Mapping Builder.

7. Replace Drag and Drop or Type here with a value (for this example, fr is entered for French).

8. Click Save, then click Close.

The assigned values are displayed in the Mapper column.
Create for-each Statements Automatically

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

To automatically create for-each statements:

1. In the Source panel, identify the repeatable source and target elements to which to map. A repeatable element is identified by a special icon with two bars to the left of the name. When you place your cursor over these elements, the words Repeating Element are displayed. For this example, there are parent and child book and title repeatable elements in the Source panel and parent and child item and name repeatable elements in the Target panel.

2. In the Source panel, map the child repeatable element to the child target repeatable element (for this example, the source repeatable element title is mapped to the target repeatable element name).

3. In the Mapping column of the Target panel, a for-each statement is automatically created with a selectable value (for this example, named title).

   Click the for-each statement to access the Mapping Builder.

   A for-each statement with an absolute path is displayed.
4. Click **Close** to exit the Mapping Builder.

5. In the **Source** panel, identify the *parent* repeatable source and target elements to which to map (for this example, **book** and **item**).

6. In the source data structure, map the *parent* repeatable element to the *parent* target repeatable element (for this example, the source repeatable element **book** is mapped to the target repeatable element **item**).

The mapper creates a second **for-each** statement to loop through the **book** element and place the mapping into the **item** element. This statement does not include a value to select because parent elements do not typically contain attributes to map.

7. Click the **for-each** function to access the Mapping Builder. The **for-each** statement is displayed. Note that the absolute path of the mapping is converted to a relative path.

8. Click **Close** to exit the Mapping Builder.

9. Map the source attribute to the target attribute (for this example, **lang** is mapped to **lang**).
10. In the **Mapping** column, click the attribute to access the Mapping Builder.

11. In the upper right corner, select **Actions > Show Content** to display the execution code.

   The first **for-each** selects the books and the second **for-each** selects the book titles.

12. Click **Close** to exit the Mapping Builder.

13. Click **Test**.

14. Enter the source payload and click **Execute**. For this example, the payload includes two book titles. Each title is displayed in both English and Spanish.

   The titles in English and Spanish for both books are displayed in the **Output** panel.

---

**See** *Repeat a Target Element to Map to Different Sources.*

### Delete Individual Mappings

You can delete individual source-to-target mappings with the mapper.

To delete individual mappings:
1. To remove a mapping, right-click the row of the target to which to remove and select **Delete Mapping**.

2. Click **Yes** when prompted to confirm.

   If you attempt to delete a parent mapping that includes child mappings, you are prompted to confirm that you want to delete all the child mappings inside the parent mapping.

## Find Data Fields

The mapper displays the source data structure on the left and the target data structure on the right. You can search for specific field names in either structure.

To find data fields:

1. Type the full or partial name in the source or target data structure, and click the **Find** icon.
The tree scrolls to the first match.

2. Click the **Find** icon again to scroll to the next match.

**Browse and Filter the Source or Target Data Structure**

The mapper displays the source data structure on the left and the target data structure on the right. You can filter the display of either structure.

To browse and filter the source or target data structure:

1. To expand or collapse parent nodes one level at a time:
   a. Right-click the parent node and select **Expand** or **Collapse**.
   b. Click the **Expand** or **Collapse** icon to the left of the parent node.

2. To expand or collapse nodes under a parent, right-click a row and select **Expand** or **Collapse All Below**.

3. To promote a node to the top, right-click a row and select **Show as Top**. To return to the original view, select the **Up** arrow in the first row, then select the data structure hierarchy to access.
4. To filter the source side, click **Filter** to specify map filtering options based on the following criteria, then click **Apply**. You can select one option from each section. When the **Filter** icon is displayed in blue, this indicates that the filter has been modified from its default settings and is active. To reset the filter option to its default values, click **Reset All**.

- Current field status (show all fields, mapped fields, or unmapped fields (applies to target nodes to which a source, variable, or function has been mapped).
- How fields are defined (all fields, required fields, or custom fields you created in a prebuilt Oracle integration that was edited in customization mode). You can select to filter both required and custom fields together.
- The source data structures in the integration. This option is only displayed if there are multiple source data structures in this mapper. You can select to filter all data structures together.

5. On the target side, click **Filter** to specify map filtering options based on the following criteria, then click **Apply**. You can select one option from each section. When the **Filter** icon is displayed in blue, this indicates that the filter has been modified from its default settings and is active. To reset the filter option to its default values, click **Reset All**

- Current field status (show all fields or mapped fields (applies to target nodes to which a source, variable, or function has been mapped).
- How fields are defined (all fields, required fields, or custom fields you created in a prebuilt Oracle integration that was edited in customization mode). You can select to filter both required and custom fields together.
- The origin of mappings (all, standard, or custom).
- Whether the mappings are valid. You can select to filter multiple options together.
Map Multiple Source Structures to a Target Structure

You can map fields from multiple source structures to a single target structure in certain parts of integrations (for example, integrations in which message enrichment points have been added or integrations with a response mapping). This action applies to the creation of new maps.

To map multiple source structures to a target structure:

1. In the mapper, note that two source structures are displayed:
   - The initial request mapping source (process)
   - The secondary request enrichment mapping source ($RequestEnrichmentApplicationObject)

2. Expand the initial source data structure and drag appropriate source fields to target fields.
Expand the secondary source data structure and drag appropriate source fields to target fields.

3. To test the mappings, see Test Your Mappings.
4. When complete, click Save, then click Exit Mapper.

Deep Copy Parent Nodes from Source to Target

You can deep copy a similar parent node from the source to the target using one drag and drop action. This eliminates the need to map every individual leaf node in large mappings. You can then override individual leaf mappings on a case-by-case basis. Similar is defined as a source and target having the same QName (name and namespace) and data type. A similar parent node can also be mapped if both source and target nodes are repeating elements. While performing deep copy, repeating parent nodes are mapped with the for-each tag while the nonrepeating parent nodes are not mapped. The leaf element and attributes are mapped with the value-of tag. You can only perform deep copying on the mapper page. Deep copying of functions on the Build Mappings page is not supported.

1. Access the mapper in an integration.
2. Drag the similar parent node from the source to the target.
   The mapper identifies that the selected repeating elements are similar and asks if you want to map similar descendents.
3. Click Yes. You can also select a checkbox to remember your preferences for this session.

For this example, the source TypedAddressList element is mapped to the target TypedAddressList element. Attributes and leaf elements are mapped. Nonleaf elements are not mapped, such as Country. Nonrepeating parents cannot be mapped, but their children can be mapped.
4. Edit leaf mappings on a case-by-case basis. However, if you delete the parent, everything is deleted.

Set Default Values in the Mapper

You may have scenarios in which you need to set some fields to default values. The mapper contains a set of functions (for example, the when function available under Mapping Components > XSL Elements) that you can use to set default values.

For example, the following payload is sent to Oracle Sales Cloud from an integration.

```xml
<typ:process>
    <typ:Opportunity>
        <opp:ConflictId>10</opp:ConflictId>
        <opp:CurrencyCode>USD</opp:CurrencyCode>
        <opp:Name>ORG</opp:Name>
    </typ:Opportunity>
</typ:process>
```

In the payload for the conflictID, you can set the default value in the mapper using the when function.

```xml
<xsl:choose/>
<xsl:when test="nssrcmpr:process/opportunity/conflictID">
    <xsl:value-of select="/nssrcmpr"process/opportunity/conflictID"/>
    <xsl:otherwise/> 1000
</xsl:when>
```

This syntax checks if the <opp:conflictId> node is present in the payload. If so, it assigns that value. Otherwise, it adds the default value, which in this case is 1000.
Test Your Mappings

Once you complete designing your mappings, you can test it by entering sample content of the message to process in the mapping tester. When you execute the test, the output is generated from the sample content.

To test a mapping:

1. In the mapper toolbar, click **Test**.

   The mapping tester appears. The names of the source and target data structures for your mapping are displayed at the top. If your mapping includes multiple source data structures, both names are displayed. The primary source is displayed first, followed by the secondary source. Two instances are also generated and displayed for repeating nodes.

2. In the **Input** panel, enter the payload you want to test. You can manually enter the payload, copy and paste the payload, or click **Generate Inputs** to automatically generate the payload.

   If your mapping includes multiple source data structures, payloads for both sources can be generated and displayed by clicking the source name.
Note:

If the payload is very large, it is not automatically generated and you receive the following error message:

Payload could not be generated for the "$SourceApplicationObject" schema due to excessive size and a lack of system memory

3. Expand the display of the input payload by dragging the splitter in the middle of the page to the right. You can also click the splitter to toggle between the input and output payloads.

4. Scroll through the input payload and note the following details:
   - Unbounded, repeating elements are displayed multiple times.
• Schemas of up to 20 levels in depth can be displayed.
• Random values are automatically generated for payload elements. Based on the data type of the element, the correct values (for example, numerical or string values) are generated.
• You can manually edit the randomly-generated values, as necessary.

5. Click **Execute** to generate results in the **Output** panel.

6. Review the results in the **Output** panel to ensure that your input payload was processed correctly.

7. Test your mapping and, as necessary, return to the mapper to make mapping changes and to the Mapping Builder to make mapping logic changes, such as changing the XSL elements or functions used.

8. To clear the **Input** and **Output** panels, click **Clear**.

9. When testing is complete, click **Close**.
Build Expressions

Use the Mapping Builder to create and modify advanced mappings. For example, you can loop through repeating elements, create conditional statements, use XPath functions, and reference lookups.

Topics

• Build Mapping Statements
• Repeat Elements with the for-each Statement
• Create Conditional Mappings
• Use XPath Functions
• Reference Lookups
• Create the lookupValue Function
• Delete Mapping Statements

Build Mapping Statements

This section describes how to build mapping statements.

To build mapping statements:

1. Drag a source to a target to create a mapping.
2. Click the target node to launch the Mapping Builder.
   
   If the target is not mapped, you see two statements: the target tag and a new empty statement with the text Drag and Drop or Type here....
3. Drag a source element onto the empty row.

The value-of mapping statement appears.
4. Click the **Edit** icon, or right click and select **Edit**. The statement expands to show the select and source statements.

5. Change the mapping as necessary by dragging a different source element onto the source statement.

6. Click **Close**.

### Repeat Elements with the for-each Statement

The for-each statement allows you to loop through repeating elements.

To repeat elements with the for-each statement:

1. Drag a source to a target to create a simple mapping.
2. Click the target node to launch the Mapping Builder.
3. Click **Mapping Components**.
   
   You can either find or browse for the function.
   
   - To find the function, type `for-each` in the **Find** field and click **Find**.
   - To browse for the function, expand the **XSL Elements** folder and locate `for-each`.
4. Drag the **for-each** function onto the target tag statement.  
   The **for-each** statement appears as a parent of the target tag. A new empty row is created with the text **Drag and Drop or Type here...**

5. Click **Source**.
6. Drag a repeating source element onto the empty row.  
   This uses the source element as the basis for the loop.
7. Click **Save**.
Convert a Comma-Separated String into a List of Entries with a for-each XSL Element

You can convert a comma-separated string into a list of entries to iterate through using a for-each XSL element.

Assume you have a requirement to split a comma-separated field into individual data. For example:

```xml
<xsl:variable name="RMCSLedgerSet"
  select="nsmpr3:lookupValue('tenant/resources/dvms/Lookup_DataStore',
    'DataStore', 'RMCSLedgers', 'Link', '')" xml:id="id_17"/>

<nstrgmpr:execute xml:id="id_16">
  <nstrgdfl:submitJobWithOutputRequestMultiple xml:id="id_25">
    <xsl:for-each select="$RMCSLedgerSet" xml:id="id_29">
      <nstrgdfl:submitJobWithOutputRequestSingle xml:id="id_26">
        <nstrgdfl:inboundRequestID xml:id="id_27">
          RMCSLedgerSet has the comma-delimited values US_USD_PRM', 'GB_GBP_PRM.

          To satisfy this requirement, use a for-each XSL element and select the create-nodeset-from-delimited-string function in the Build Mappings page of the mapper. Provide the lookupValue as the delimited string and ',' as the delimiter.

          <xsl:for-each select = "oraext:create-nodeset-from-delimited-string('something',
            string(nsmpr0:lookupValue('tenant/resources/dvms/Lookup_DataStore',
              'RMCSLedgers', 'Link', ')), ',', '))">
```

Create Conditional Mappings

The if and choose statements are two ways to create conditions. If statements allow you to specify a single condition. Choose/when/otherwise statements allow you to specify multiple conditions, similar to if/then/else.

To create conditional mapping:

1. Drag a source to a target to create a mapping.
2. Click the target node to launch the Mapping Builder.
3. Click **Mapping Components**.
   - You can either find or browse for the function.
   - To find the function, type if or choose in the **Find** field, and click **Find**.
To browse for the function, expand the XSL Elements folder and locate if or choose.

4. Drag the if or choose function onto the target tag statement. The statement appears as a parent of the target tag. A new empty row is created with the text Drag and Drop or Type here....

5. Click Source. Then do one of the following:
   - Drag a source element onto the empty row to use a source element as the basis for the test condition.
   - Select an XPath function as the basis for the test condition.
   - Use a literal as the basis for the test condition by typing in the empty row.

6. For choose, you may specify additional when and otherwise conditions.
   a. Right-click the choose statement and select Insert Child. A new empty row is displayed with the text Drag and Drop or Type here...
   b. Drag and drop when or otherwise from the Mapping Components list.

7. Click Save.

Use XPath Functions

You can use a variety of XPath functions in your data mappings to transform the data you transfer between applications.

To use an XPath function:
1. Drag a source to a target to create a mapping.
2. Click the target node to launch the Mapping Builder.
3. Click Mapping Components.
You can either find or browse for the function:

- To find the function, type the full or partial name of the function in the Find field, and click the Find icon.
- To browse for the function, expand the Functions folder and subcategory folders to locate the function.

4. Drag the function onto the mapping statement.
   
   If the function has a single parameter, it automatically inserts the existing mapping expression into the parameter.
   
   If the function has multiple parameters, it prompts you to select which parameter into which to insert the existing mapping expression.

5. Select a parameter or None if you want to remove the existing mapping expression, and click OK.

6. Click Save and Close.

Support for BPEL XPath Extension Functions

The mapper supports some BPEL XPath extension functions that you can include in your data mappings. The functions are available in both orchestrated integrations and basic map data integrations.

In the Source tree, expand the following categories under Functions. Place your cursor over a specific function to invoke a message that describes the syntax to use.
<table>
<thead>
<tr>
<th>Under Functions...</th>
<th>Available BPEL XPath Extension Functions</th>
</tr>
</thead>
</table>
| **Expand Functions > Advanced** | • create-nodeset-from-delimited-string  
• decodeBase64  
• encodeBase64  
• generate-guid  
• lookup-xml  
• parseEscapedXML  
• parseXML |
| **Expand Functions > Mathematical** | • max-value-among-nodeset  
• min-value-among-nodeset  
• square-root |
| **Expand Functions > String** | • compare-ignore-case  
• create-delimited-string  
• format-string  
• get-content-as-string  
• get-localized-string  
• index-within-string  
• last-index-within-string  
• left-trim  
• right-trim |

**Use the length Parameter in the Substring Function in the Mapper**

When you drag the `substring` function into the mapper, it may appear that you cannot add a value to the third parameter `length`. This is not the case. This is an optional parameter.

To add a value to the `length` parameter, perform the following steps:

1. Drag and drop the `substring` function into the mapper. Only required parameters are automatically dropped.
2. Right-click `substring`, and select **Insert Child** to add an empty row for the optional parameter.
3. Once the third row is available, you can drag and drop or type into the field to complete the optional parameter.

**Add a New Line Between Variables in a concat Operation**

You can add a new line between variables in a `concat` operation by manually editing the XSL file outside of Oracle Integration. For example:

First Variable

Second Variable

To add a new line between variables in a `concat` operation:
1. Export the integration.

2. Open the XSL file to be modified.

3. Use one of the following four solutions to add a new line:

   \[
   \text{xsl:value-of} \\
   \text{select} = "\text{concat}(/\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field1Name1}, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field1Value1}, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field2Name1}, $textObj, $textObj, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field2Value1})" \text{xml:id} = \text{id}_68
   \]

   \[
   \text{xsl:value-of} \\
   \text{select} = "\text{concat}(/\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field1Name1}, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field1Value1}, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field2Name1}, $breakObj, $breakObj, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field2Value1})" \text{xml:id} = \text{id}_69
   \]

   \[
   \text{xsl:value-of} \\
   \text{select} = "\text{concat}(/\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field1Name1}, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field1Value1}, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field2Name1}, $blankObj, $blankObj, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field2Value1})" \text{xml:id} = \text{id}_70
   \]

   \[
   \text{xsl:value-of} \\
   \text{select} = "\text{concat}(/\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field1Name1}, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field1Value1}, /\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field2Name1}, '\&#10;', '\&#10;', '/\text{nssrcmpr:execute/nssrcmpr:customObjects}_24/\text{nsmpri2:Field2Value1})" \text{xml:id} = \text{id}_71
   \]
4. Save the XSL file.
5. Import the XSL file into Oracle Integration by clicking the **Import** option on the map action in an integration.
6. Save the integration.

**Reference Lookups**

A special lookup function in the mapper enables you to call a lookup from a mapping to determine the value to populate into a field when transferring data between applications.

To reference a lookup from a data mapping:

1. Drag the source to a target to create a simple mapping.
2. Click the target node to launch the Mapping Builder.
3. In the Mapping Builder, click **Mapping Components**.
4. Type `lookupValue` in the **Find** field, and click **Find**.
5. Drag the function onto the mapping statement.

The mapper prompts you to select a previously created lookup.
6. Search by lookup name. 
When you select a lookup in the **Lookup Tables** column, you see preview data for that lookup on the right based on the connections you selected as the source and target.

7. Select a lookup and click **Use**.

8. The system automatically populates the following parameters in the lookup function:
   - `dvmLocation`: with the lookup name you selected
   - `srcColumn`: with the source application type
   - `srcValue`: with the existing mapping expression
   - `targetColumn`: with the target application type

9. Enter a `defaultValue` parameter. This is the value sent to the target if the lookup function is unable to find a match for the value passed from the source.

10. Click **Save** and **Close**.
Create the lookupValue Function

You can create the parameter values for the lookupValue function with the Build Lookup Function wizard. This wizard enables you to define the lookup table, source column, target column, and default value to use in the function. For these parameter values to be selectable in the wizard, you must have already created a lookup on the Lookups page.

Topics

- Access the Build Lookup Function Wizard
- Select the Lookup Table
- Select the Source and Target Columns
- Specify the Default Value
- Review Your Lookup Table Selections

Access the Build Lookup Function Wizard

The Build Lookup Function wizard for creating the lookupValue function parameter values is accessible from the mapper.

To access the Build Lookup Function wizard:

1. Go to the Integrations page.
2. Open an integration.
3. Click the mapper to edit.
   The mapper is displayed.
4. Click the value in the Mapping column for the mapping on which to use lookups.

Note:

You must already have created lookups to use this wizard. See Creating a Lookup of Using Integrations in Oracle Integration.
The Build Mappings page is displayed.

5. Expand Mapping Components > Functions > ICS or use the Search utility to find the lookupValue function.

6. Drag the lookupValue function on top of the expression on the right side.

The Build Lookup Function wizard is displayed. To create the lookupValue function parameter values, see section Select the Lookup Table.

Select the Lookup Table

Select the lookup table to use in the lookupValue function.

Topics

- What You Can Do from the Select Lookup Table Page
- What You See on the Select Lookup Table Page

What You Can Do from the Select Lookup Table Page

Select the lookup table to use in the function.

Note:

You must already have created a lookup. Otherwise, no lookups are displayed for selection.

What You See on the Select Lookup Table Page

The following table describes the key information on the page.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lookup Table</td>
<td>Select the lookup table to use in the function. You can view the lookup description by clicking the information icon in the table. This can guide you in selecting the required lookup table. The number of columns defined in the lookup is also displayed.</td>
</tr>
</tbody>
</table>

Select the Source and Target Columns

Select the source and target columns to use in the lookupValue function.

Topics

- What You Can Do from the Source and Target Columns Page
- What You See on the Source and Target Columns Page
What You Can Do from the Source and Target Columns Page

Select the source and target columns to use in the `lookupValue` function.

The `lookupValue` function requires one source column and one target column.

What You See on the Source and Target Columns Page

The following table describes the key information on the page.

When you select a source and target column, the values available with the columns are displayed.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Source Column</td>
<td>Click the source column header to select from a list of available columns for this lookup table. The data included with the selected column is displayed. Both adapter and domain name columns are displayed.</td>
</tr>
<tr>
<td>Select Target Column</td>
<td>Click the target column header to select from a list of available columns for this lookup table. The data included with the selected column is displayed. Both adapter and domain name columns are displayed.</td>
</tr>
</tbody>
</table>

Specify the Default Value

Select the default value to use in the `lookupValue` function.

Topics

- What You Can Do from the Default Value Page
- What You See on the Default Value Page

What You Can Do from the Default Value Page

Enter the default value to use if no match is found. If there is no match that satisfies all the search values, the lookup fails and the default value is returned.

What You See on the Default Value Page

The following table describes the key information on the page.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td>Enter a default value to use if no match is found (for example, an actual default value to use or an error message such as No Value Found).</td>
</tr>
</tbody>
</table>
Review Your Lookup Table Selections

You can review the lookup table values to use in the `lookupValue` function on the Summary page.

Topics

• What You Can Do from the Lookup Table Summary Page
• What You See on the Lookup Table Summary Page

What You Can Do from the Lookup Table Summary Page

You can review the lookup table values from the Summary page. The Summary page is the final wizard page after you have completed your configuration.

What You See on the Lookup Table Summary Page

The following table describes the key information on the page.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter and Value Table</td>
<td>Displays a summary of the parameters and values you defined on previous pages of the wizard.</td>
</tr>
<tr>
<td></td>
<td>To return to a previous page to update any values, click the appropriate tab in the left panel or click Back.</td>
</tr>
</tbody>
</table>
Delete Mapping Statements

You can delete one or all mapping statements from the **Actions** menu in the Mapper Builder.

To delete mapping statements:

1. Drag the source to a target to create a simple mapping.
2. Click the target node to launch the Mapping Builder.
3. To delete a single statement, right-click a statement or click the **Actions** menu and select **Delete**.
4. To delete all statements, click the **Actions** menu and select **Delete All**.

---

### Element Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resulting Expression</td>
<td>Displays the expression you defined on the previous pages of the wizard. The <code>lookupValue</code> function takes the following format:</td>
</tr>
</tbody>
</table>

```plaintext
lookupValue(dvmLocation, srcColumn, srcValue, targetColumn, defaultValue)
```

Where:

- **dvmLocation**: The lookup table selected on the Select Lookup Table page.
- **srcColumn**: The source column selected on the Select Columns page.
- **srcValue**: The source value you enter in the **New Condition** field of the Expression Builder after completing this wizard. Click **Done** to complete this wizard, then define the `srcValue` parameter value.
- **targetColumn**: The target column selected on the Select Columns page.
- **defaultValue**: The default value entered on the Default Value page.

For example, a defined `lookupValue` function after you have completed the wizard and defined the `srcValue` parameter value in the Expression Builder can look as follows:

```plaintext
dvm:lookupValue('tenant/resources/dvms/Country','rightnow','US','mysoap','No data found')
```