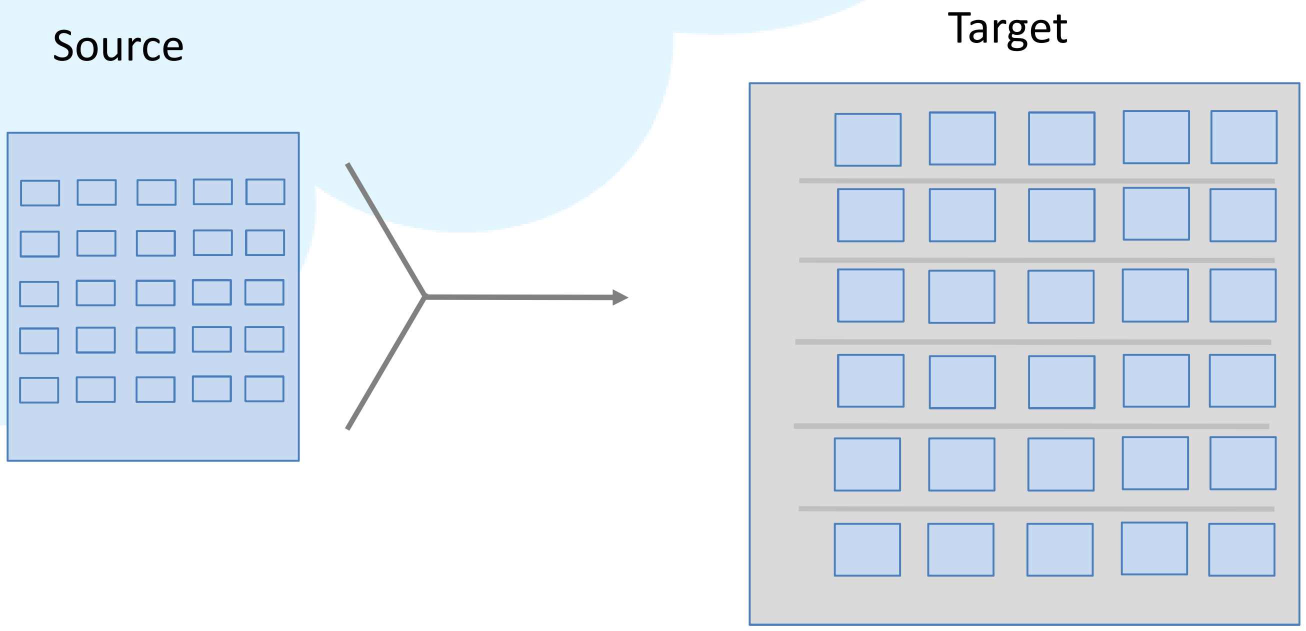
Leading the **Integration** Revolution

Your business problems have changed.
Why hasn't your integration solution?

ADEPTIA

Mapping Scenarios

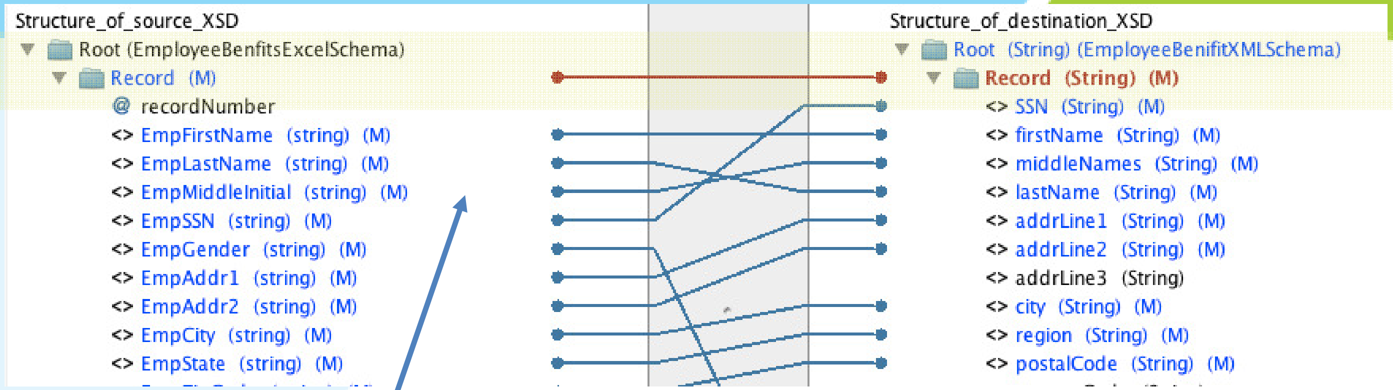
Example 1: Mapping scenario – Applying ‘For Each’ condition



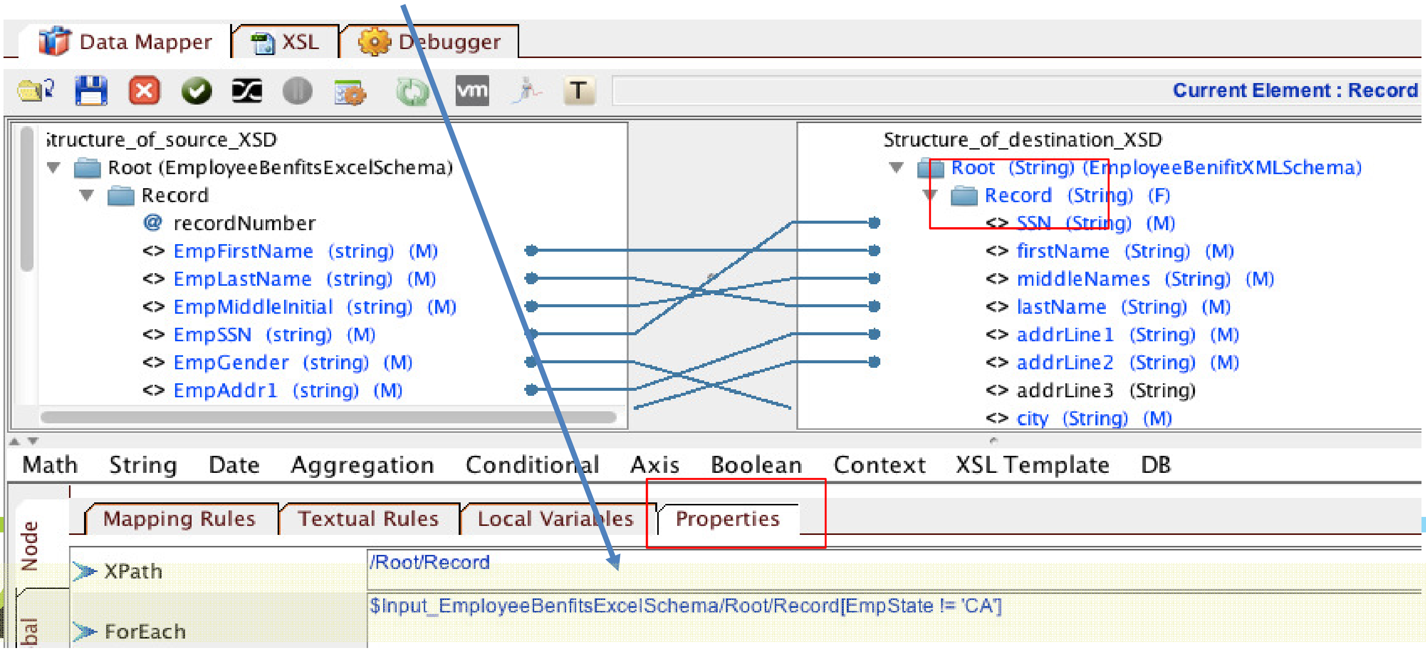
Example: Map source records into target. Apply ‘For Each’ condition to pull all the records from the source and map to target fields.

Refer to Mapper Help on how to apply mapping functions such as Constants, String, Math, Conditional, Aggregation, Date, Variables etc.

Example 1: Mapping scenario – Applying 'For Each' condition

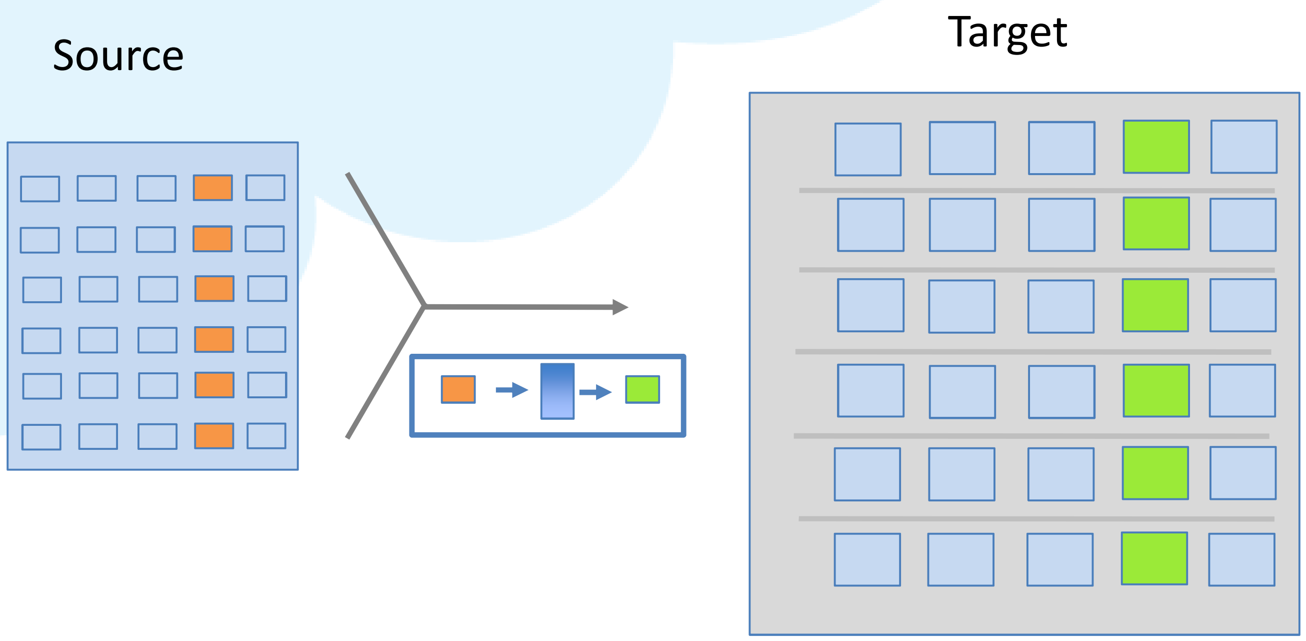

Structure_of_source_XSD
Root (EmployeeBenfitsExcelSchema)
Record (M)
@ recordNumber
<> EmpFirstName (string) (M)
<> EmpLastName (string) (M)
<> EmpMiddleInitial (string) (M)
<> EmpSSN (string) (M)
<> EmpGender (string) (M)
<> EmpAddr1 (string) (M)
<> EmpAddr2 (string) (M)
<> EmpCity (string) (M)
<> EmpState (string) (M)
Structure_of_destination_XSD
Root (String) (EmployeeBenifitXMLSchema)
Record (String) (M)
<> SSN (String) (M)
<> firstName (String) (M)
<> middleNames (String) (M)
<> lastName (String) (M)
<> addrLine1 (String) (M)
<> addrLine2 (String) (M)
<> addrLine3 (String)
<> city (String) (M)
<> region (String) (M)
<> postalCode (String) (M)

By connecting Source 'Record' node to the Target 'Record' node applies a ForEach condition that would pick all the record from the Source and map to the Target schema. If this ForEach condition is not applied then only one record would be picked from the Source.
If we need to apply a filter condition on the ForEach such as 'Select records where State is not equal to CA' then use the following approach.
First select the Target Record node then go to its Properties and put your cursor in the ForEach field. On the ForEach field in the Properties panel double click on the Source Record node and then add a predicate with the logic as shown below. Refer to Mapper Help for more information.



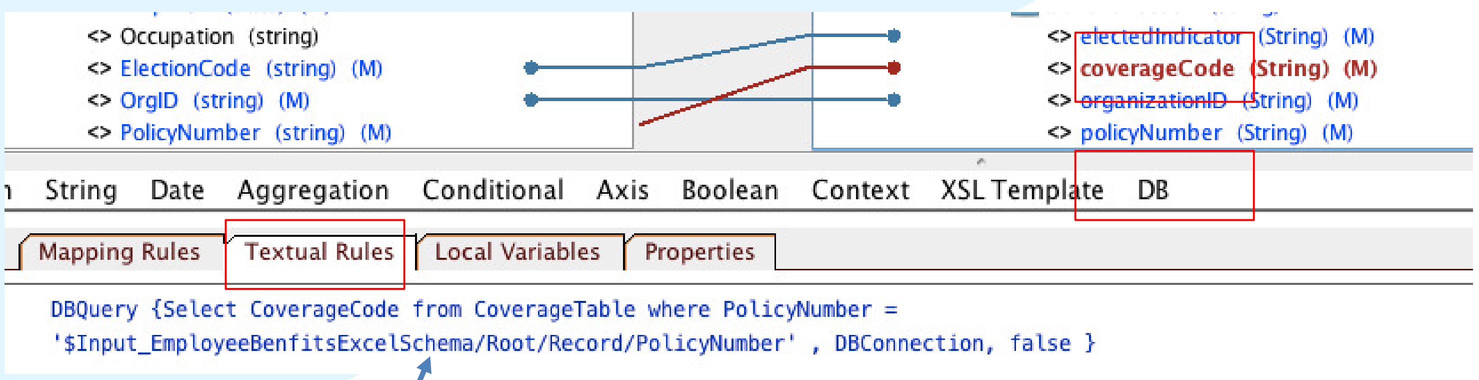
XPath: /Root/Record
ForEach: $Input_EmployeeBenfitsExcelSchema/Root/Record[EmpState != 'CA']

Example 2: Mapping scenario – Applying 'Lookup' condition



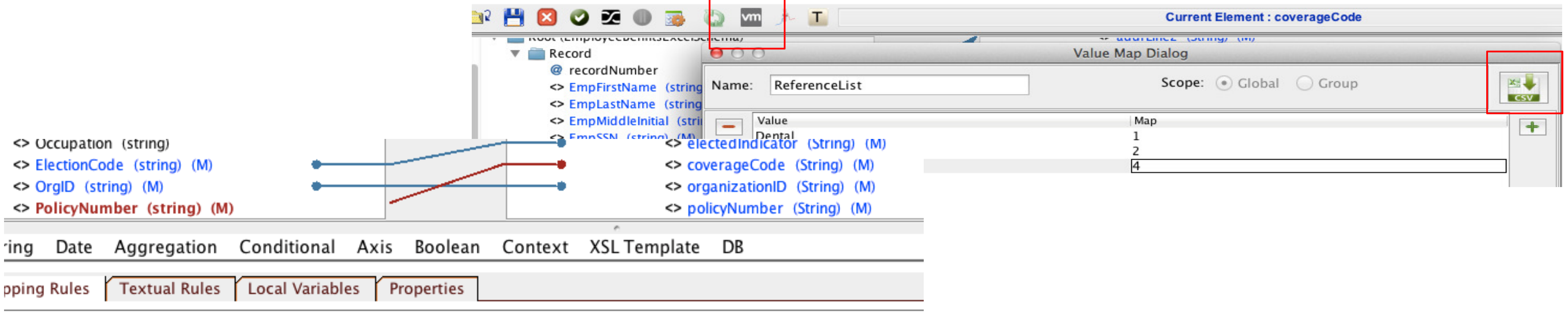
Example: Apply a lookup on a database reference table or a Value Map based on a source field value and map the result into a target field.

Example 2: Mapping scenario – Applying ‘Lookup’ condition


<> Occupation (string)
<> ElectionCode (string) (M)
<> OrgID (string) (M)
<> PolicyNumber (string) (M)
<> electedIndicator (String) (M)
<> coverageCode (String) (M)
<> organizationID (String) (M)
<> policyNumber (String) (M)
String Date Aggregation Conditional Axis Boolean Context XSL Template DB
Mapping Rules
Textual Rules
Local Variables
Properties

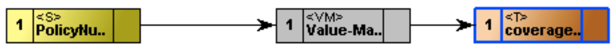
```
DBQuery {Select CoverageCode from CoverageTable where PolicyNumber =
'$Input_EmployeeBenfitsExcelSchema/Root/Record/PolicyNumber' , DBConnection, false }
```

Select the target field and then go to Textual Rules and select DB. It will show a function that takes three parameters. In the first parameter write the SQL query and put the Source field that is going to be used for the lookup in the Where clause. Use single quote if the source field is of type text. The second parameter uses the DB variable name (refer to Help on how to create the DB connection variable). The third parameter is either True or False. True means that more than one records in the result set, False means only one record in the result set. The return response from the query will be mapped to the target field.
If the reference data is not in a Database but in a CSV file or if you want to create a new Reference list within Mapping then use the Value Map (VM) Function as shown below. Refer to Mapper Help on how to use Value Map.


Current Element : coverageCode
Record
@ recordNumber
<> EmpFirstName (string
<> EmpLastName (string
<> EmpMiddleInitial (stri
Value Map Dialog
<> Occupation (string)
<> ElectionCode (string) (M)
<> OrgID (string) (M)
<> PolicyNumber (string) (M)
<> electedIndicator (String) (M)
<> coverageCode (String) (M)
<> organizationID (String) (M)
<> policyNumber (String) (M)
ring Date Aggregation Conditional Axis Boolean Context XSL Template DB
pping Rules
Textual Rules
Local Variables
Properties

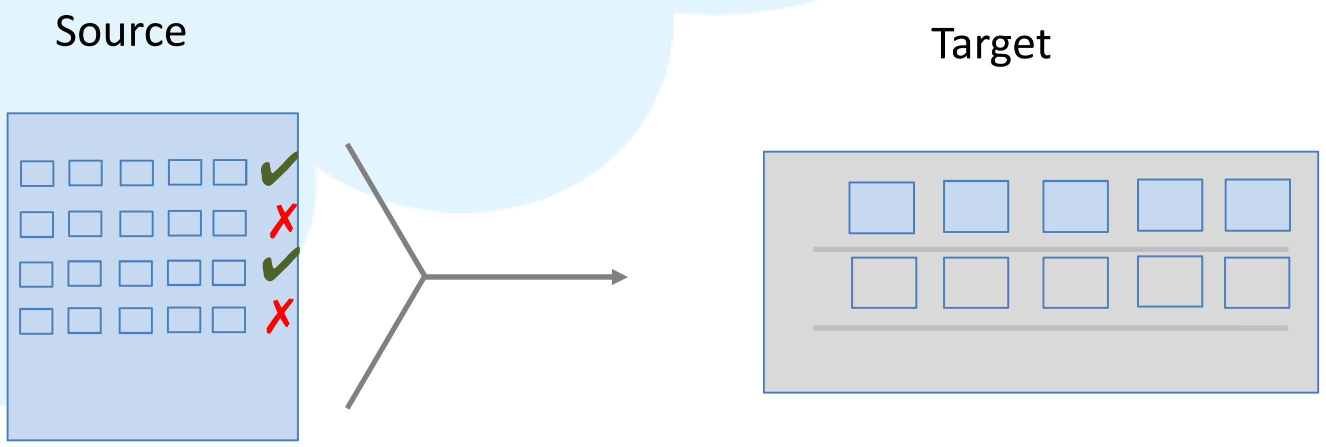
Name: ReferenceList
Scope: ◉ Global ○ Group

Value	Map
Dental	1
	2
	4



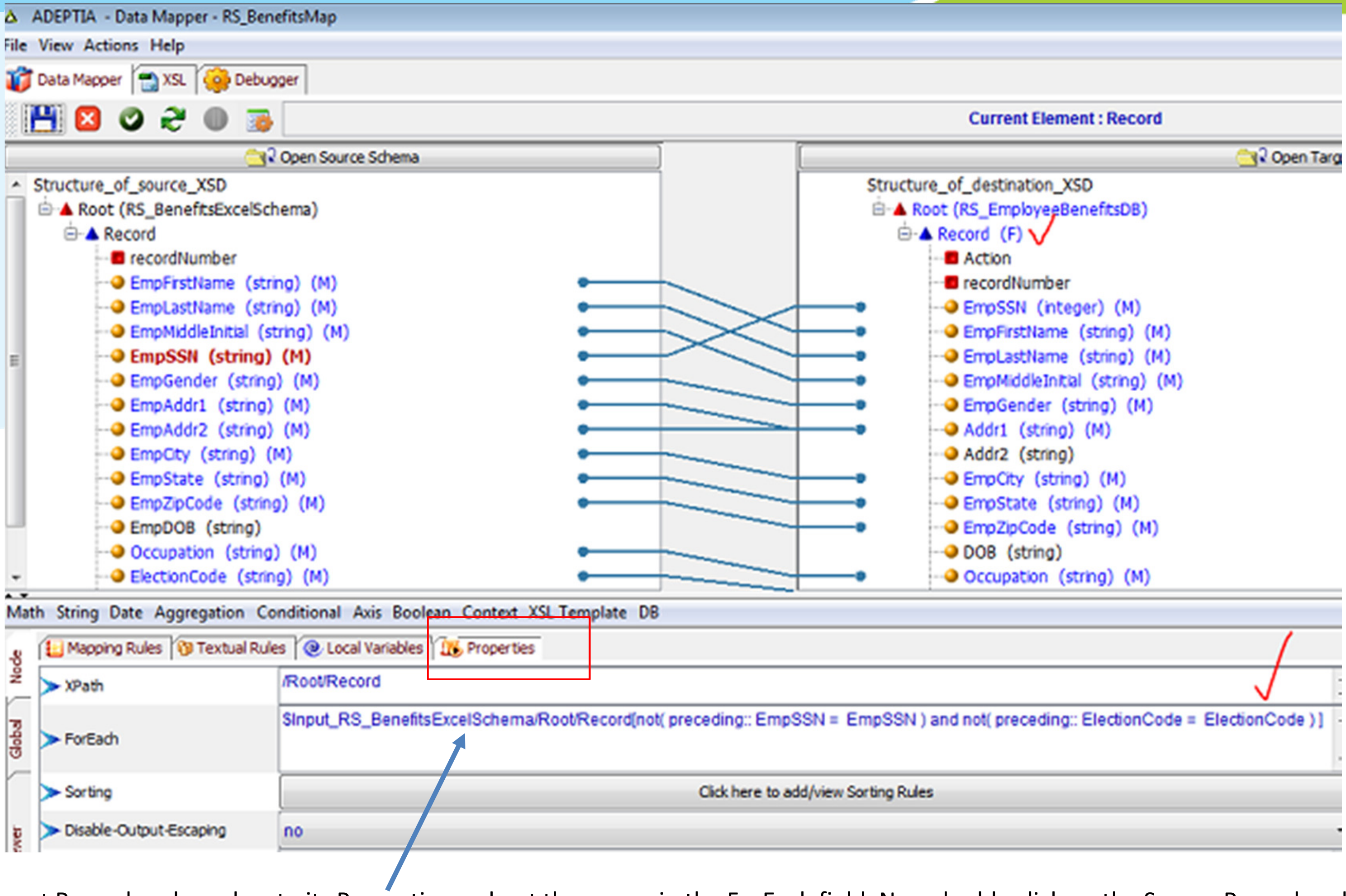
ADEPTIA

Example 3: Mapping scenario – Applying ‘Duplicate handling’ condition



Example: Map source records into target. Apply ‘For Each’ and duplicate handling condition to pull only unique records from source and map to target.

Example 3: Mapping scenario – Applying ‘Duplicate handling’ condition


ADEPTIA - Data Mapper - RS_BenefitsMap

File View Actions Help

Data Mapper | XSL | Debugger

Current Element : Record

Open Source Schema

Structure_of_source_XSD
- Root (RS_BenefitsExcelSchema)
  - Record
    - recordNumber
    - EmpFirstName (string) (M)
    - EmpLastName (string) (M)
    - EmpMiddleInitial (string) (M)
    - **EmpSSN (string) (M)**
    - EmpGender (string) (M)
    - EmpAddr1 (string) (M)
    - EmpAddr2 (string) (M)
    - EmpCity (string) (M)
    - EmpState (string) (M)
    - EmpZipCode (string) (M)
    - EmpDOB (string)
    - Occupation (string) (M)
    - ElectionCode (string) (M)

Open Targ

Structure_of_destination_XSD
- Root (RS_EmployeeBenefitsDB)
  - Record (F) ✓
    - Action
    - recordNumber
    - EmpSSN (integer) (M)
    - EmpFirstName (string) (M)
    - EmpLastName (string) (M)
    - EmpMiddleInitial (string) (M)
    - EmpGender (string) (M)
    - Addr1 (string) (M)
    - Addr2 (string)
    - EmpCity (string) (M)
    - EmpState (string) (M)
    - EmpZipCode (string) (M)
    - DOB (string)
    - Occupation (string) (M)

Math String Date Aggregation Conditional Axis Boolean Context XSL Template DB

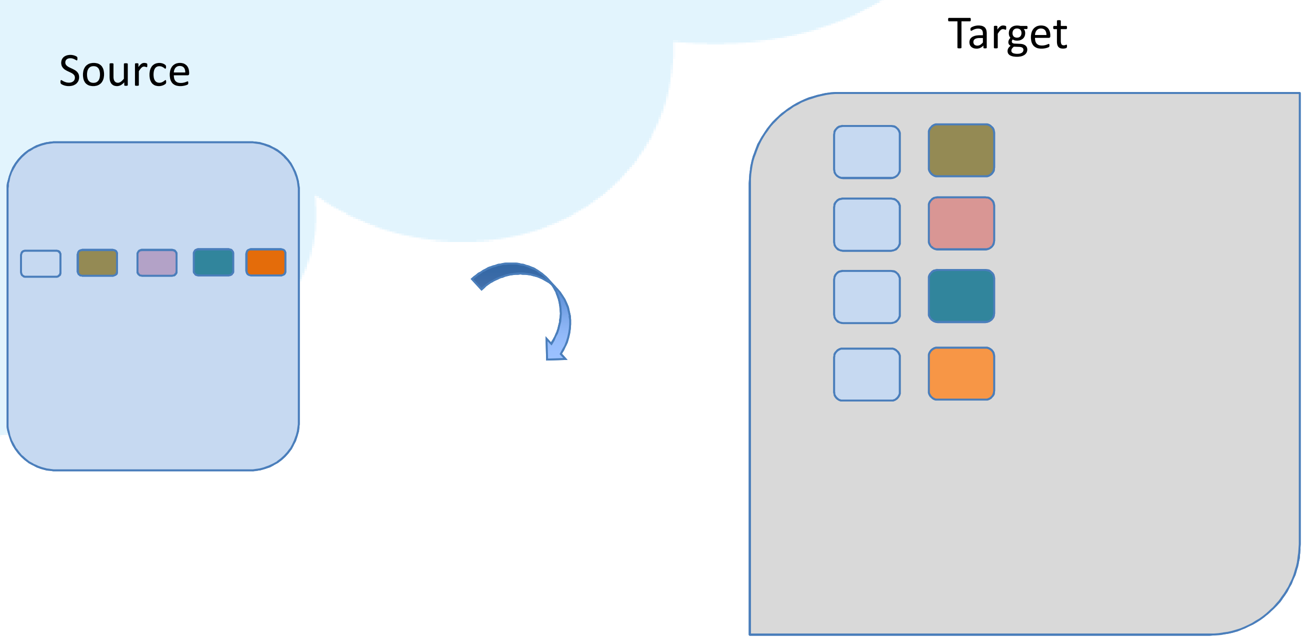
Mapping Rules | Textual Rules | Local Variables | Properties

XPath	/Root/Record ✓
ForEach	$Input_RS_BenefitsExcelSchema/Root/Record[not(preceding:: EmpSSN = EmpSSN) and not(preceding:: ElectionCode = ElectionCode)]
Sorting	Click here to add/view Sorting Rules
Disable-Output-Escaping	no

Select the Target Record node and go to its Properties and put the cursor in the ForEach field. Now double click on the Source Record node and Then apply the ‘preceding’ Axis function. Here the example shown above is to pick only those records whose SSN and ElectionCode do not match with any preceding records from the Source. If the condition is met then record will be sent to Target else it will be considered a duplicate and will be filtered out.

ADEPTIA

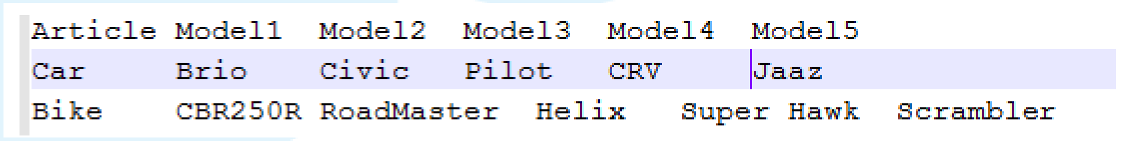
Example 4: Mapping scenario – Pivot single source record to multiple target records.



Example: Pivot single source record to multiple target records.
Repeat the same rule for remaining source records.

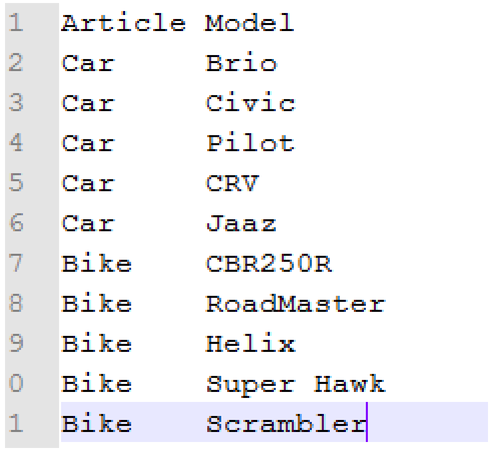
Example 4: Mapping scenario – Pivot single source record to multiple target records.

A car brand that maintains its product listing as :



```
Article Model1  Model2     Model3   Model4     Model5
Car     Brio    Civic      Pilot    CRV        Jaaz
Bike    CBR250R RoadMaster  Helix    Super Hawk  Scrambler
```

The ouput required should be as :



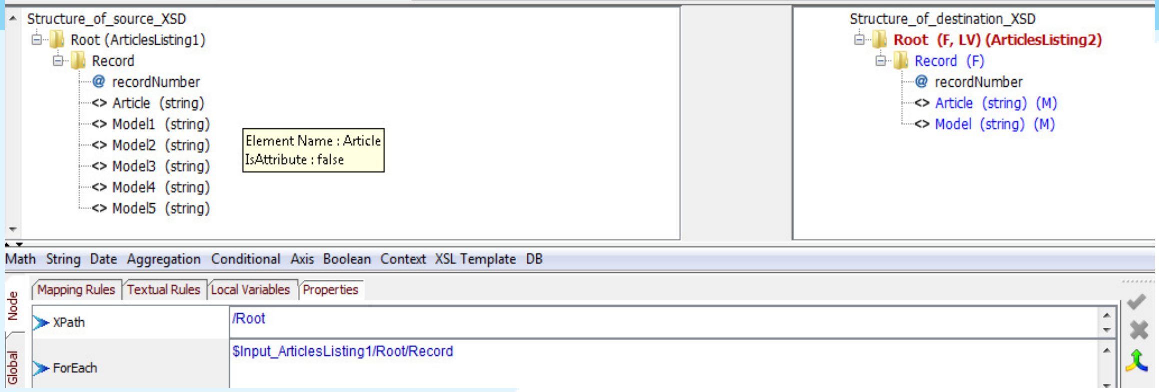
```
Article Model
Car     Brio
Car     Civic
Car     Pilot
Car     CRV
Car     Jaaz
Bike    CBR250R
Bike    RoadMaster
Bike    Helix
Bike    Super Hawk
Bike    Scrambler
```

The above mentioned scenario can be achieved by pivoting single source record to multiple target records in Data Mapper as shown below.

ADEPTIA

Example 4: Mapping scenario – Pivot single source record to multiple target records.

1) Apply for- each at the root level of target schema with the XPath of the Source Record.



Structure_of_source_XSD
- Root (ArticlesListing1)
  - Record
    - @ recordNumber
    - <> Article (string)
    - <> Model1 (string)
    - <> Model2 (string)
    - <> Model3 (string)
    - <> Model4 (string)
    - <> Model5 (string)

Element Name : Article
IsAttribute : false

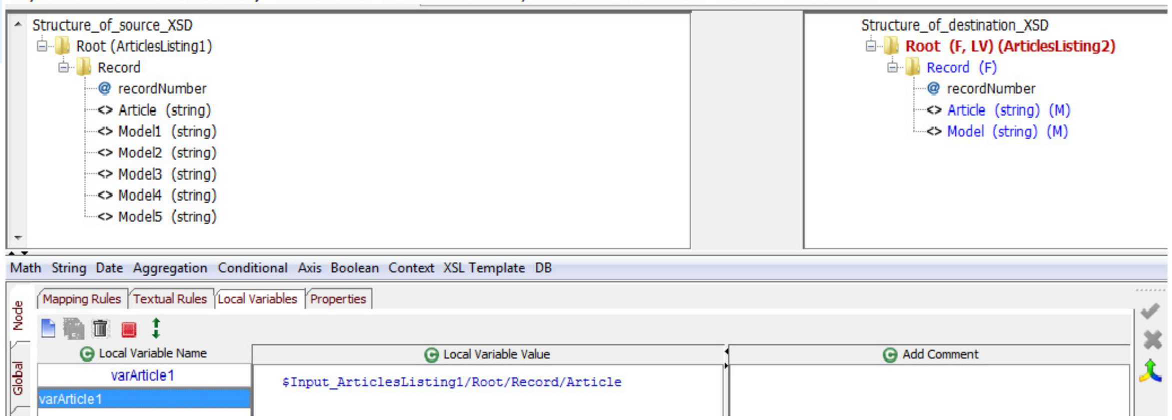
Structure_of_destination_XSD
- Root (F, LV) (ArticlesListing2)
  - Record (F)
    - @ recordNumber
    - <> Article (string) (M)
    - <> Model (string) (M)

Math String Date Aggregation Conditional Axis Boolean Context XSL Template DB
Mapping Rules | Textual Rules | Local Variables | Properties
Node
Global

XPath	/Root
ForEach	$Input_ArticlesListing1/Root/Record

Create a ForEach on Root level of the Target Data to iterate for each source Record.`

2) Create a local variable say '***varArticle1***' at the root level , to select the schema '***Article***' field value.



Structure_of_source_XSD
- Root (ArticlesListing1)
  - Record
    - @ recordNumber
    - <> Article (string)
    - <> Model1 (string)
    - <> Model2 (string)
    - <> Model3 (string)
    - <> Model4 (string)
    - <> Model5 (string)

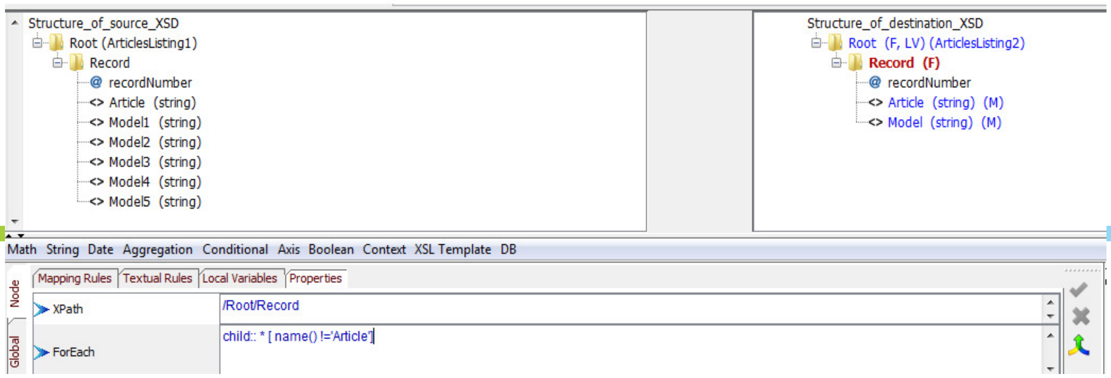
Structure_of_destination_XSD
- Root (F, LV) (ArticlesListing2)
  - Record (F)
    - @ recordNumber
    - <> Article (string) (M)
    - <> Model (string) (M)

Math String Date Aggregation Conditional Axis Boolean Context XSL Template DB
Mapping Rules | Textual Rules | Local Variables | Properties
Node
Global

Local Variable Name	Local Variable Value	Add Comment
varArticle1	`$Input_ArticlesListing1/Root/Record/Article`	
varArticle1		

Create a local variable at the root element of the target Schema to store the 'Article' field value.

3) Now Apply for- each on the record level of target Schema.



Structure_of_source_XSD
- Root (ArticlesListing1)
  - Record
    - @ recordNumber
    - <> Article (string)
    - <> Model1 (string)
    - <> Model2 (string)
    - <> Model3 (string)
    - <> Model4 (string)
    - <> Model5 (string)

Structure_of_destination_XSD
- Root (F, LV) (ArticlesListing2)
  - Record (F)
    - @ recordNumber
    - <> Article (string) (M)
    - <> Model (string) (M)

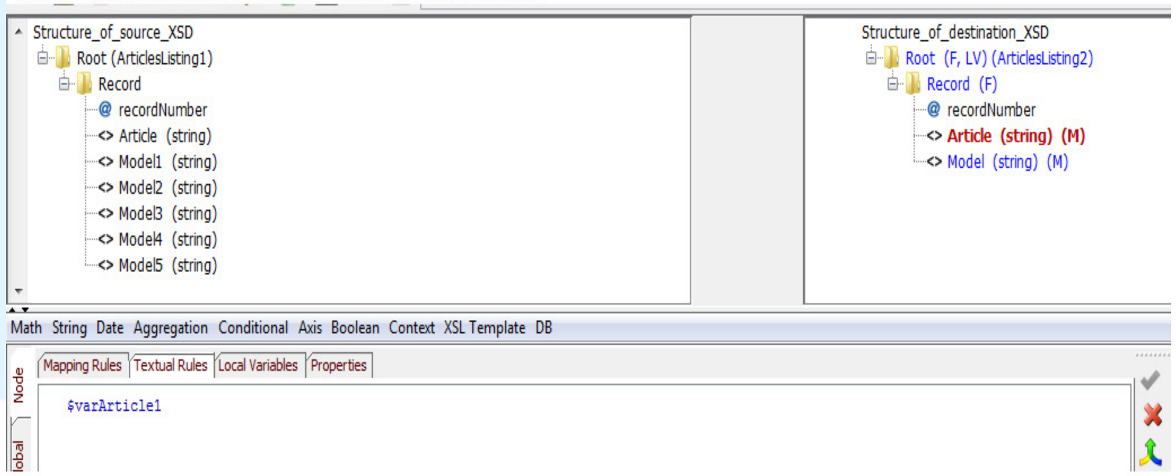
Math String Date Aggregation Conditional Axis Boolean Context XSL Template DB
Mapping Rules | Textual Rules | Local Variables | Properties
Node
Global

XPath	/Root/Record
ForEach	child:: * [name() !='Article']

Now apply for-each on the target record to iterate over all the child elements whose name is not 'Article'.

Example 4: Mapping scenario – Pivot single source record to multiple target records.

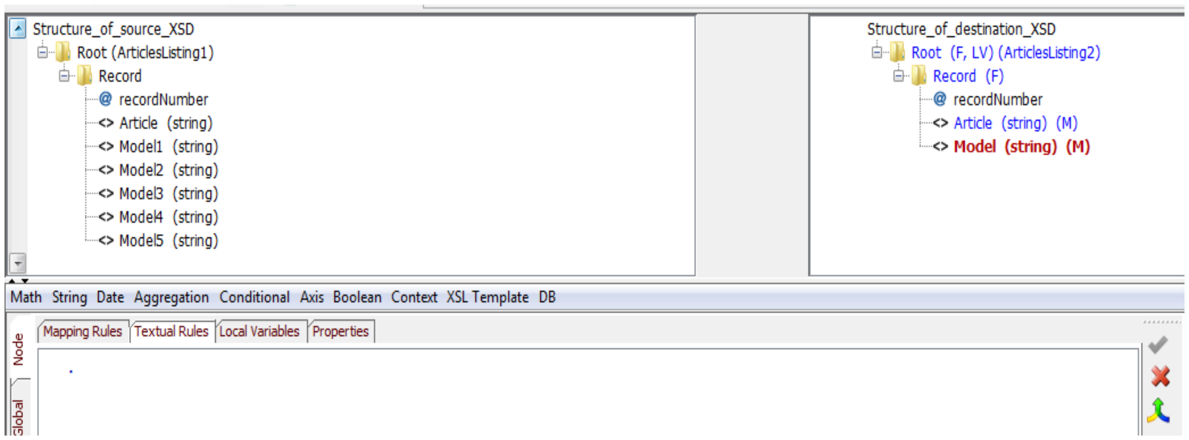
4) Apply mapping on the Article element of the target Schema.



```
$varArticle1
```

Use the local variable created at the root level of the target schema to map the 'Article' element of target schema.

5) Apply Mapping on Model element of the target schema.

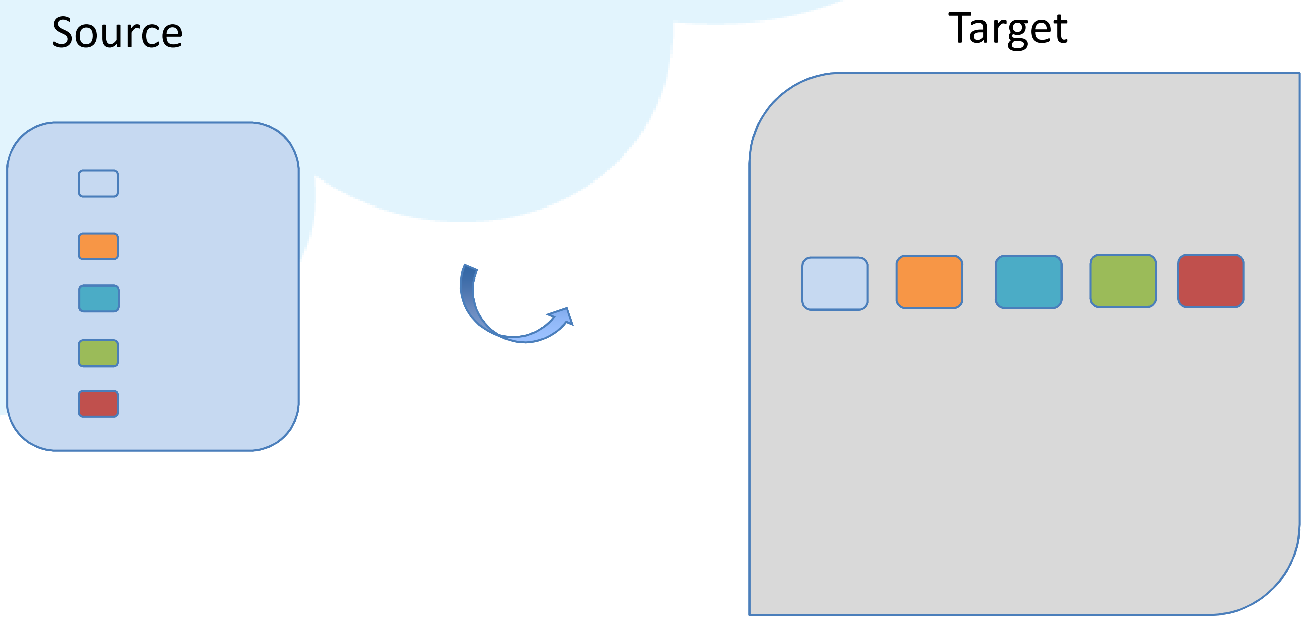


```
.
```

Map the Model element value of the target Schema by applying a "." in textual rule. This will pick values from Model one by one.

ADEPTIA

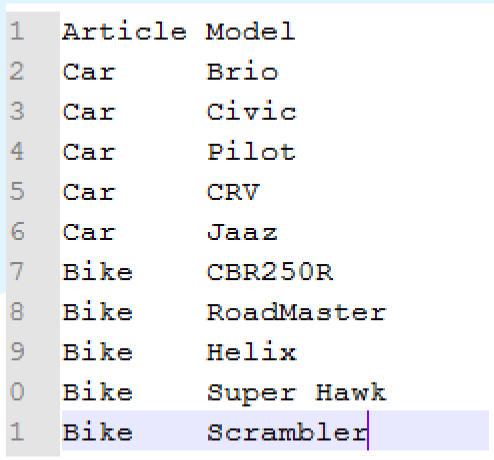
Example 5: Mapping scenario – Pivot multiple source records into a single target record.



Example: Pivot multiple source records into a single target record. Repeat the same rule for remaining source records that repeat after 5 rows. In other words one target record comprises of 5 source records block.

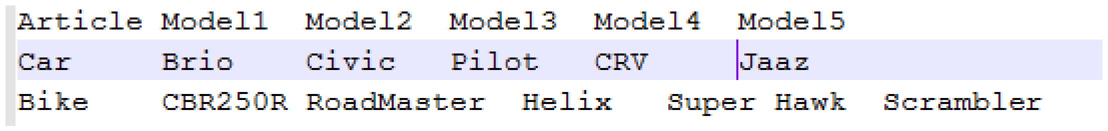
Example 5: Mapping scenario – Pivot multiple source records into a single target record.

A car brand that maintains its product listing as:



```
Article Model
Car     Brio
Car     Civic
Car     Pilot
Car     CRV
Car     Jaaz
Bike    CBR250R
Bike    RoadMaster
Bike    Helix
Bike    Super Hawk
Bike    Scrambler
```

The ouput required should be as :

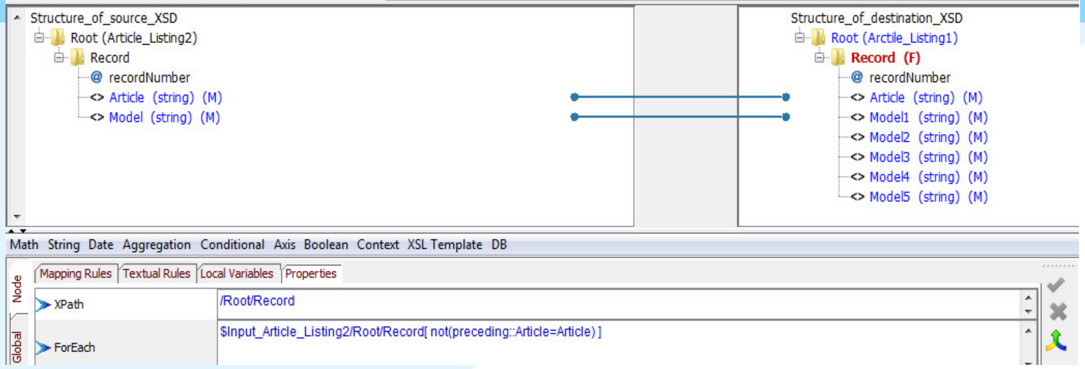


```
Article Model1  Model2  Model3  Model4  Model5
Car     Brio    Civic   Pilot   CRV     Jaaz
Bike    CBR250R RoadMaster  Helix   Super Hawk  Scrambler
```

The above mentioned scenario can be achieved by pivoting multiple source record to single target record in Data Mapper as shown below.

Example 5: Mapping scenario – Pivot multiple source records into a single target record.

1) Apply for-each at the target record with the Xpath of the Source Record.

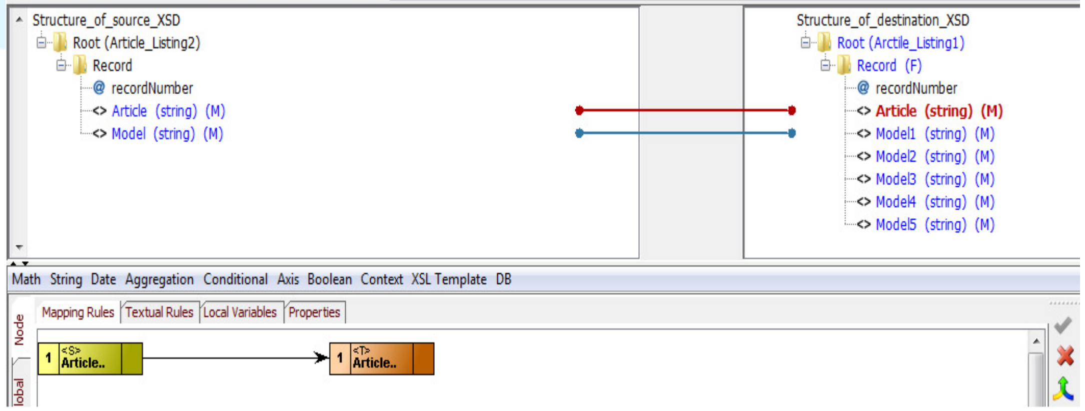


XPath /Root/Record

ForEach $Input_Article_Listing2/Root/Record[not(preceding::Article=Article)]

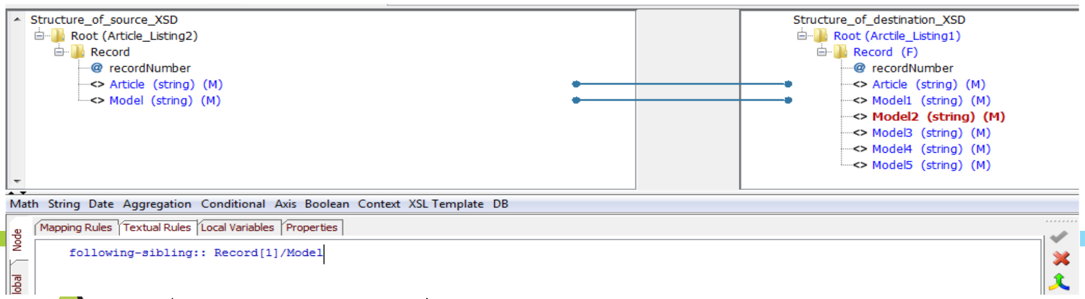
Apply 'For Each' and duplicate handling condition to pull only unique (Article)records from source and map to target.

2) Apply one to one mapping at target field name '***Article***' and '***Molde1***' elements of target Schema



Apply one to one Mapping for Article and Model1 field of target Schema.

3) Apply mapping at Model2 and other element of target schema .

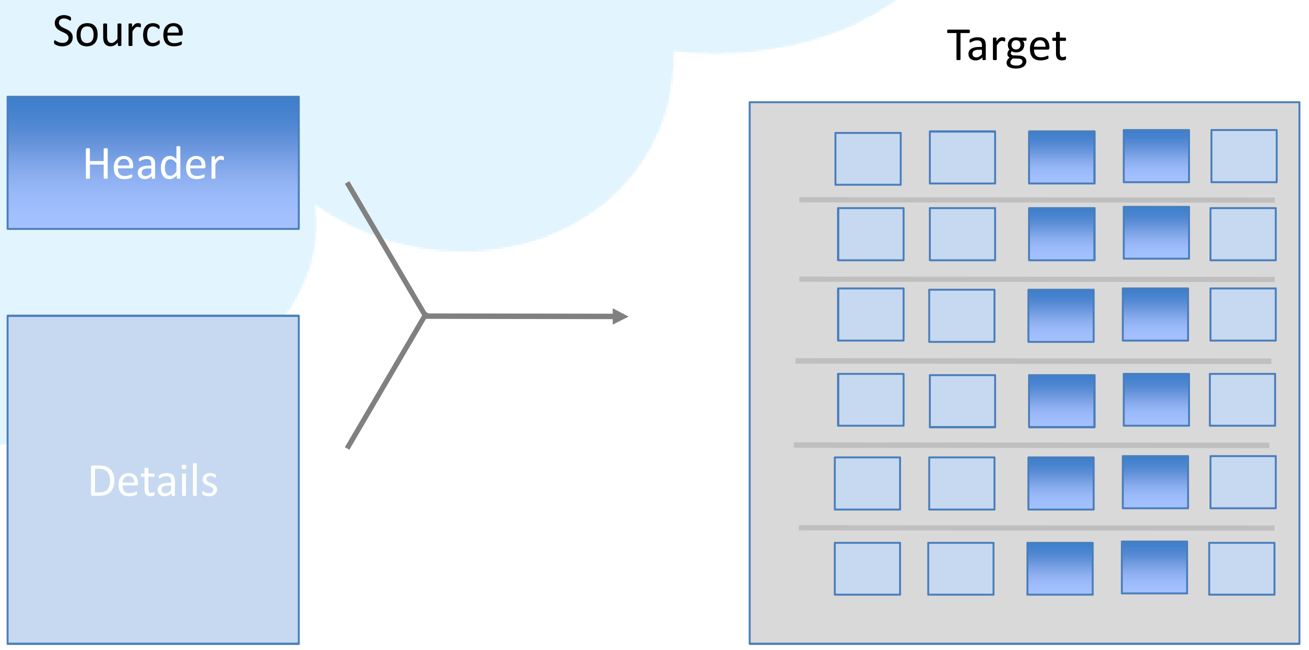


```
following-sibling:: Record[1]/Model
```

Here we are applying mapping for field name'Model2' of target Schema to fetch value of the Model field following the first source Record.
Use the following-sibling function to fetch records following Record 1 and map to 'Model2' .Similarly use following sibling to fetch records following Record 2 and map to target field 'Model3'.Thus similarly map target field 'Model4' and 'Model5' to fetch the Records following Record 3 and Record4 correspondingly using the following – sibling function.

ADEPTIA

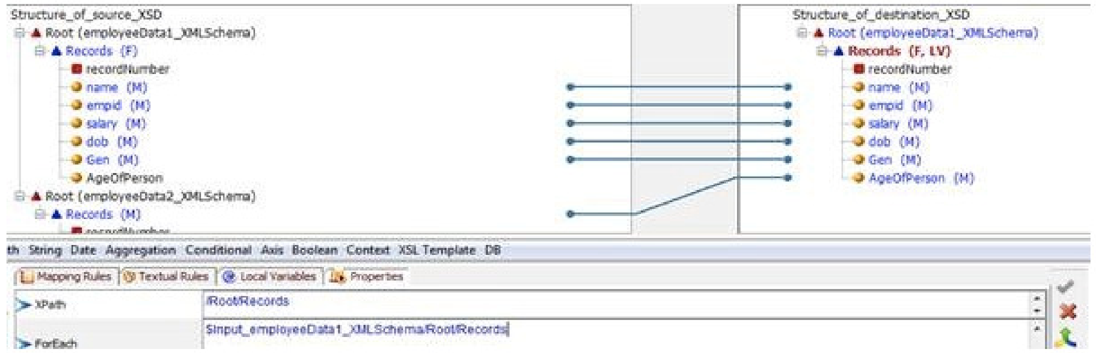
Example 6: Mapping scenario – Merging two sources by common key



Example: Two sources (header & details) merged by a common key into a target. The number of records in target equal the number of records in details. Each target record contains values from header record based on a Join condition using a common key available in the two sources.

Example 6: Mapping scenario – Merging two sources by common key

1) Apply for-each at the target record with the xPath of the first schema.



XPath /Root/Records

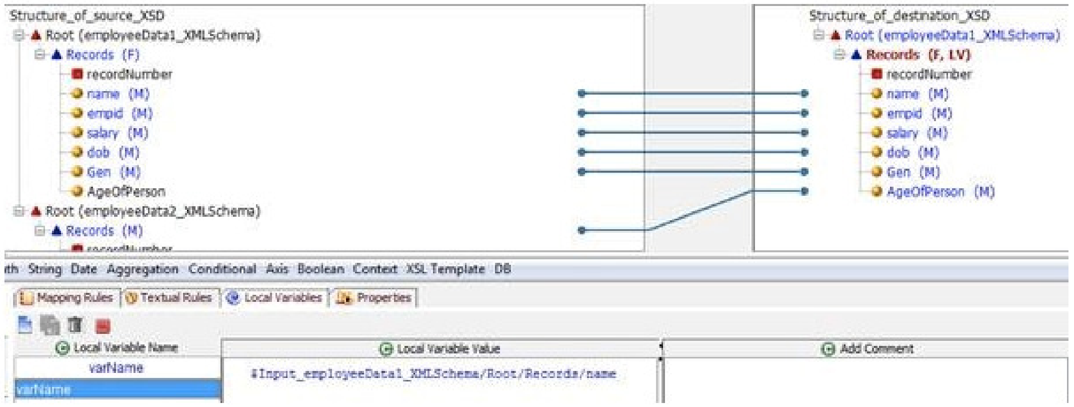
ForEach $Input_employeeData1_XMLSchema/Root/Records

Source Data1 contains the details.

Source Data2 contains the headers.

Create a ForEach on the Source Data 1.
Idea is that that Target record will iterate on each Detail record and the field in the target schema that needs a value from the Header will be obtained by a lookup.

2) Create a local variable say "***varName***" at the record level, to select first schema "***name***" field value.



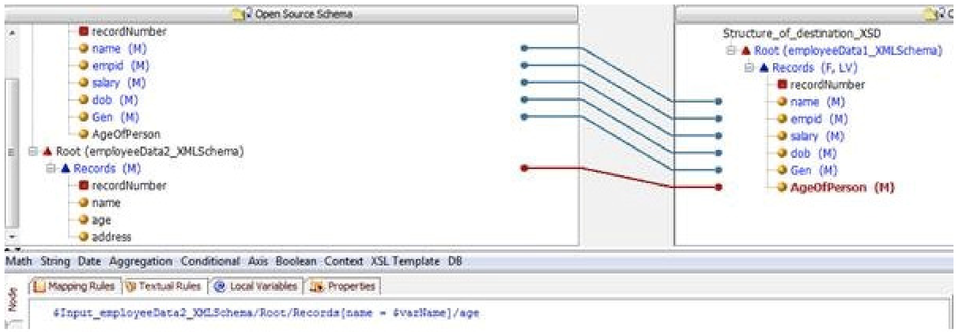
Local Variable Name	Local Variable Value	Add Comment
varName	$Input_employeeData1_XMLSchema/Root/Records/name	

varName

Create a local variable in the Target Record node and assign the value from field Name. Here Name is a **common field** in the two sources.

3) Now apply the below rule at the element "***AgeOfPerson***" (element where common data from second source needs to be mapped)

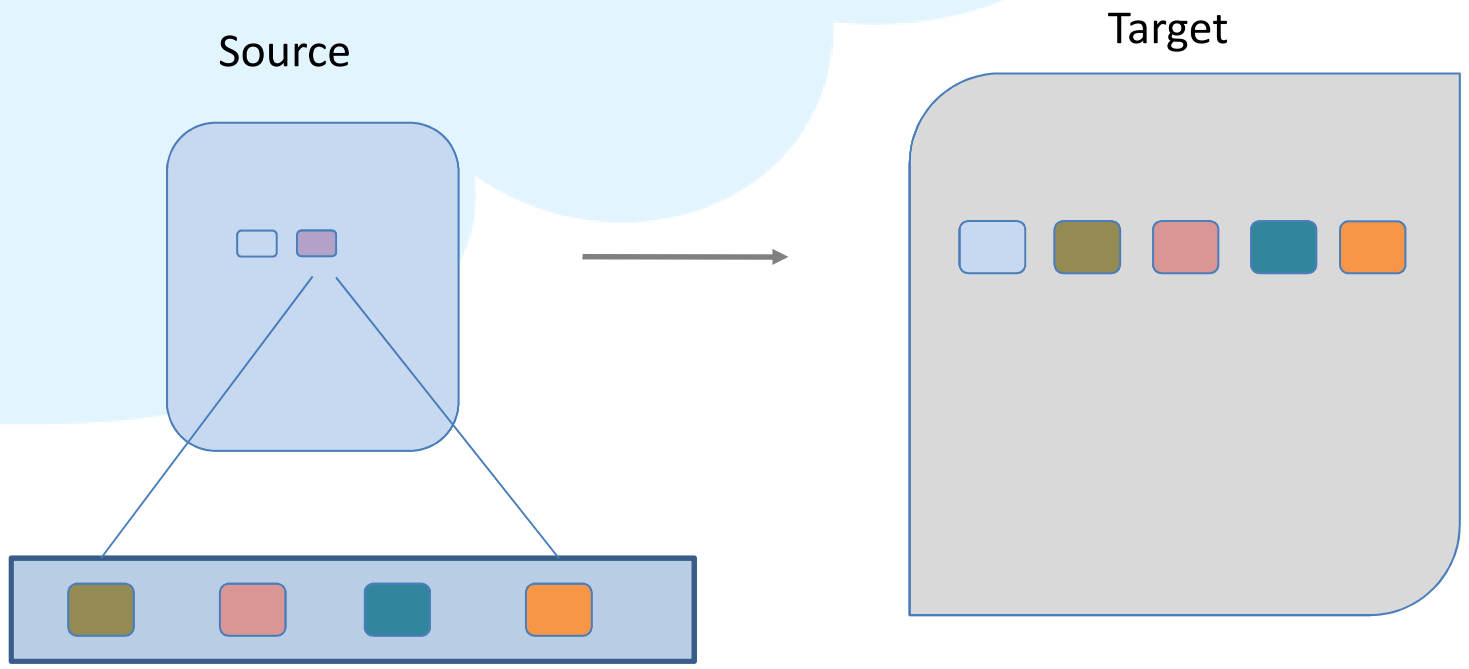
$Input_employeeData2_XMLSchema/Root/Records[name = $varName]/age.



```
$Input_employeeData2_XMLSchema/Root/Records[name = $varName]/age
```

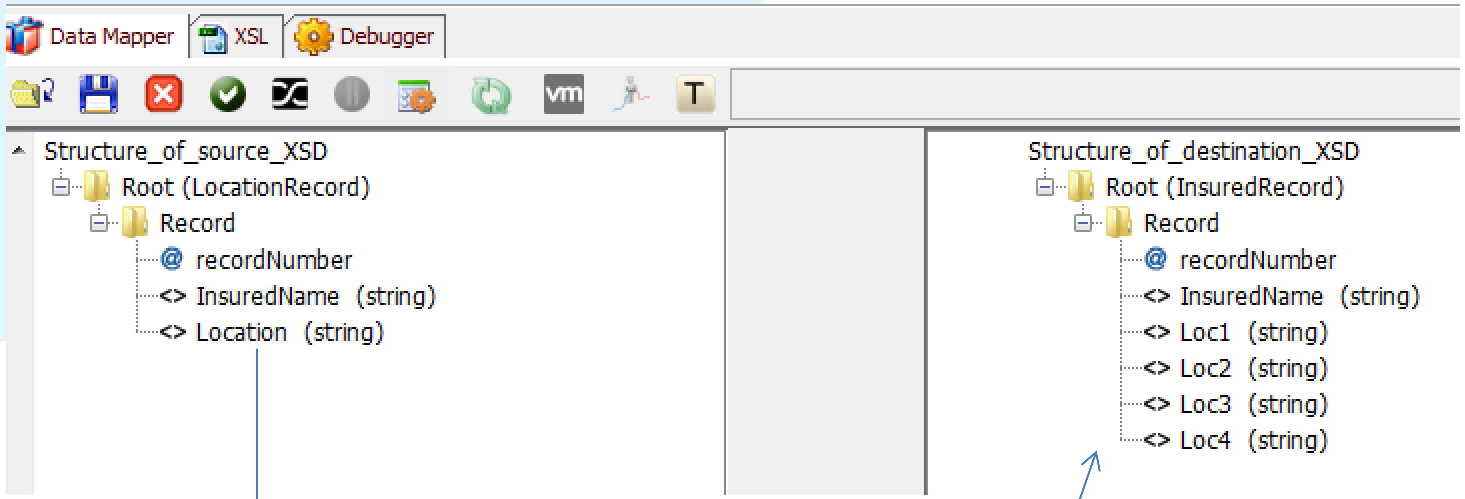
Here we are looking for Age in Source Data 2 based on Name which is coming from the variable defined above. Use a Predicate function here.

Example 7: Mapping scenario – Use Tokenize function to split a source field value into multiple parts and map each of those parts to different fields in the target



Example: Use Tokenize function to split a source value into multiple target fields and repeat the same rule on all the source records.

Example 7: Mapping scenario – Use Tokenize function to split a source field value into multiple parts and map each of those parts to different fields in the target



Structure_of_source_XSD
- Root (LocationRecord)
  - Record
    - @ recordNumber
    - <> InsuredName (string)
    - <> Location (string)

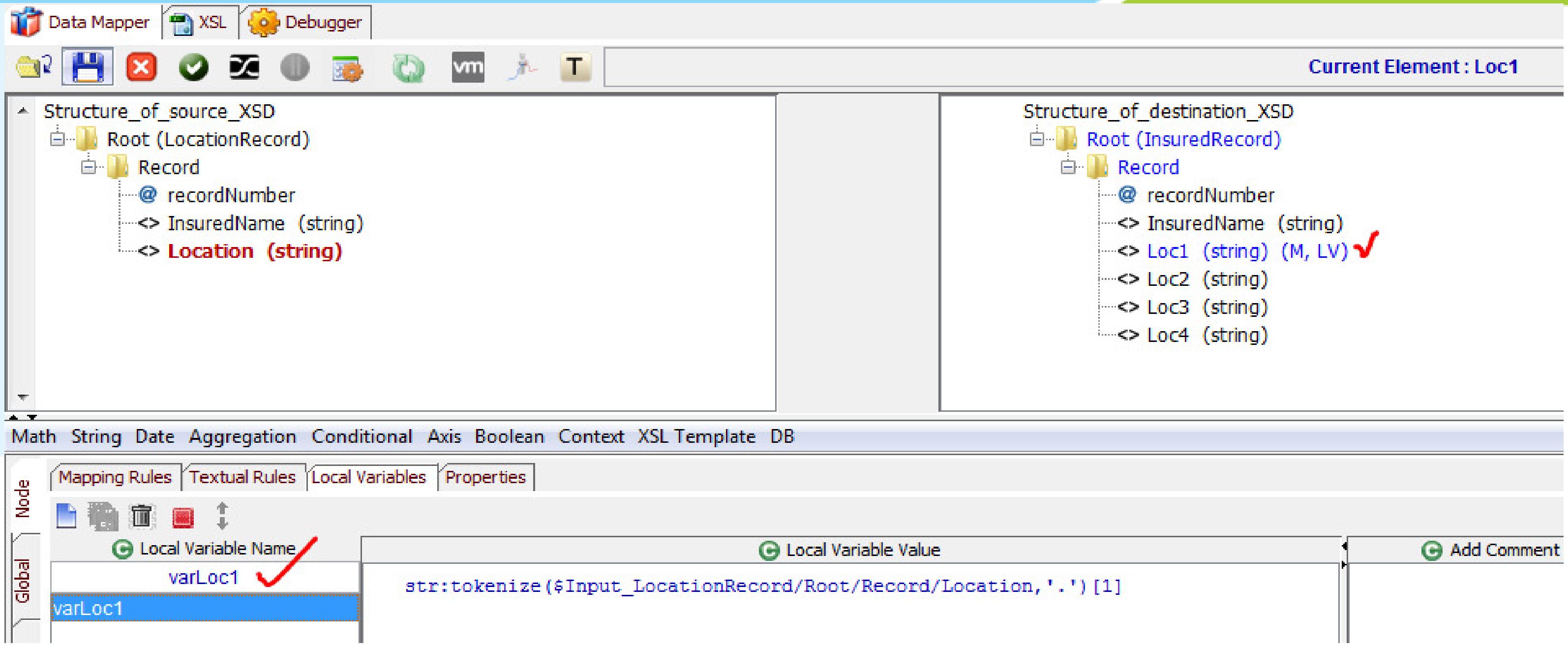
Structure_of_destination_XSD
- Root (InsuredRecord)
  - Record
    - @ recordNumber
    - <> InsuredName (string)
    - <> Loc1 (string)
    - <> Loc2 (string)
    - <> Loc3 (string)
    - <> Loc4 (string)

Suppose Location in the source has a value as WV.L2.D3.F1 (also the pattern can change)

Each of the values separated by (.) need to be mapped to its related Target field. Thus Loc1 is WV, Loc2 is L2, Loc3 is D3, Loc4 is F1

ADEPTIA

Example 7: Mapping scenario – Use Tokenize function to split a source field value into multiple parts and map each of those parts to different fields in the target



```
str:tokenize($Input_LocationRecord/Root/Record/Location,'.')[1]
```

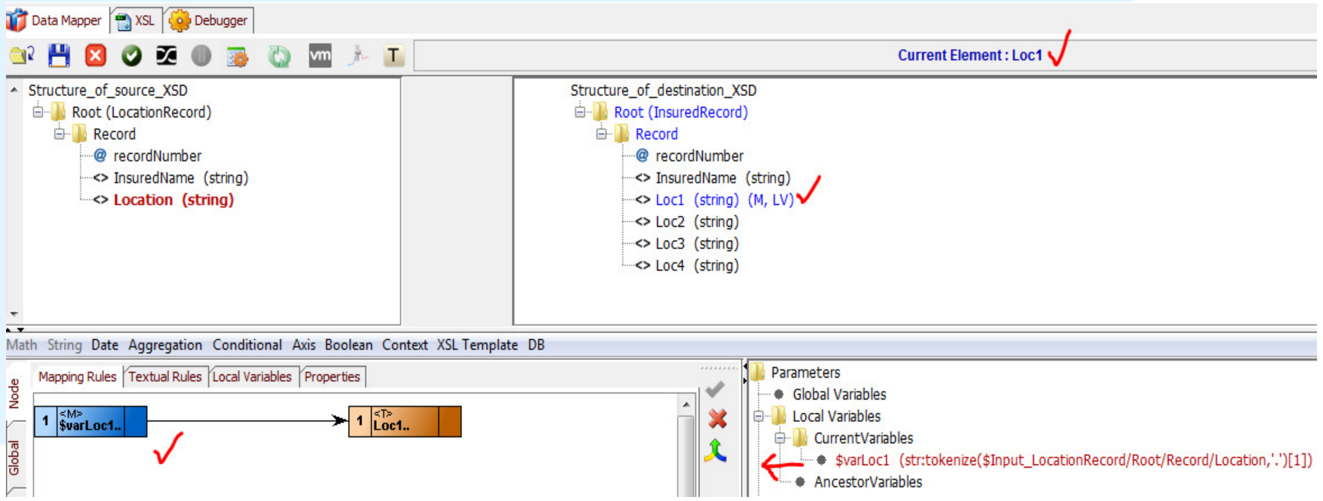
Create a local variable "var1" on the target field like "Loc1" in which use the below function:
str:tokenize($Input_SourceSchema/Root/Record/Location,'.')[1]
Map this local variable "var1" to the target field "Loc1" using the graphical or textual rule.

Similarly create another local variable "var2" on the second target field like "Loc2" in which use the below function:
str:tokenize($Input_SourceSchema/Root/Record/Location,'.')[2]
Map this local variable "var2" to the target field "Loc2" using the graphical or textual rule.

Follow the above steps for the rest of the Target Loc fields.

ADEPTIA

Example 7: Mapping scenario – Use Tokenize function to split a source field value into multiple parts and map each of those parts to different fields in the target

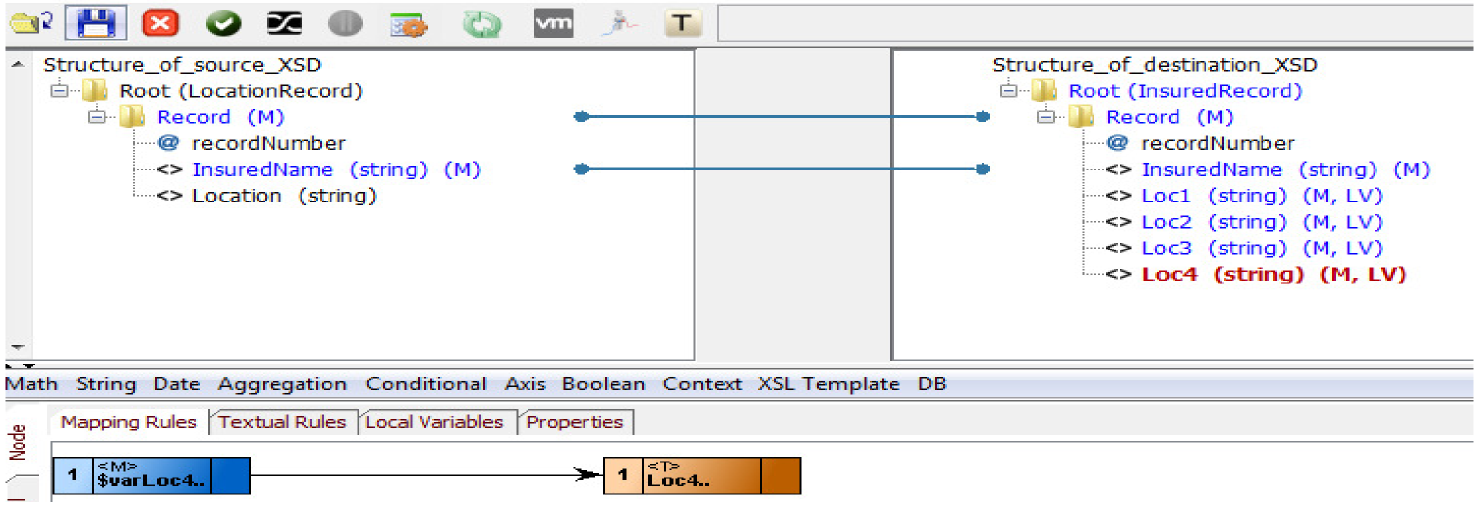


Current Element : Loc1 ✓

```
Structure_of_source_XSD
  Root (LocationRecord)
    Record
      @ recordNumber
      <> InsuredName  (string)
      <> Location  (string)
```

```
Structure_of_destination_XSD
  Root (InsuredRecord)
    Record
      @ recordNumber
      <> InsuredName  (string)
      <> Loc1  (string)  (M, LV) ✓
      <> Loc2  (string)
      <> Loc3  (string)
      <> Loc4  (string)
```

```
Parameters
  Global Variables
  Local Variables
    CurrentVariables
      $varLoc1  (str:tokenize($Input_LocationRecord/Root/Record/Location,'.')[1])
    AncestorVariables
```



```
Structure_of_source_XSD
  Root (LocationRecord)
    Record  (M)
      @ recordNumber
      <> InsuredName  (string)  (M)
      <> Location  (string)
```

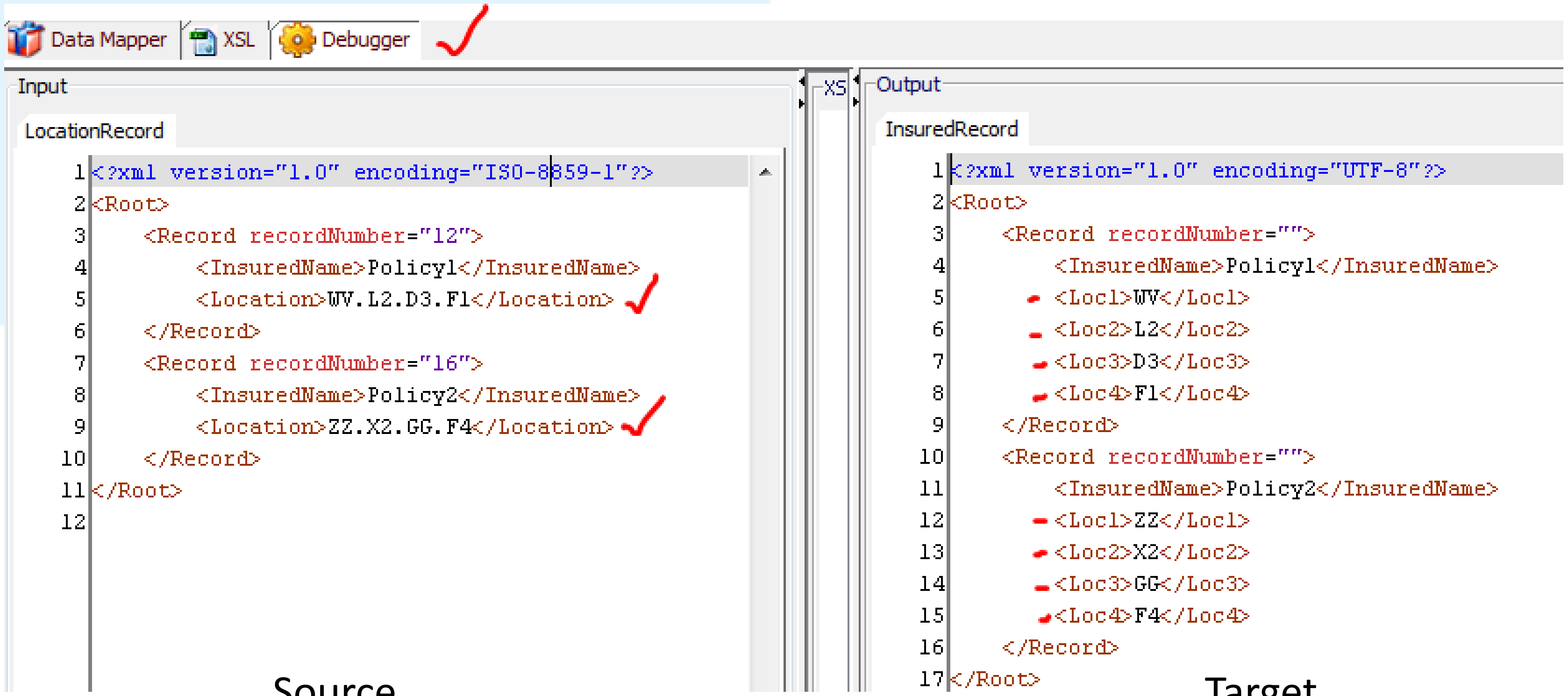
```
Structure_of_destination_XSD
  Root (InsuredRecord)
    Record  (M)
      @ recordNumber
      <> InsuredName  (string)  (M)
      <> Loc1  (string)  (M, LV)
      <> Loc2  (string)  (M, LV)
      <> Loc3  (string)  (M, LV)
      <> Loc4  (string)  (M, LV)
```

Final map

ADEPTIA

Example 7: Mapping scenario – Use Tokenize function to split a source field value into multiple parts and map each of those parts to different fields in the target

Result



```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Root>
    <Record recordNumber="12">
        <InsuredName>Policy1</InsuredName>
        <Location>WV.L2.D3.F1</Location>
    </Record>
    <Record recordNumber="16">
        <InsuredName>Policy2</InsuredName>
        <Location>ZZ.X2.GG.F4</Location>
    </Record>
</Root>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<Root>
    <Record recordNumber="">
        <InsuredName>Policy1</InsuredName>
        <Loc1>WV</Loc1>
        <Loc2>L2</Loc2>
        <Loc3>D3</Loc3>
        <Loc4>F1</Loc4>
    </Record>
    <Record recordNumber="">
        <InsuredName>Policy2</InsuredName>
        <Loc1>ZZ</Loc1>
        <Loc2>X2</Loc2>
        <Loc3>GG</Loc3>
        <Loc4>F4</Loc4>
    </Record>
</Root>
```

Source

Target

ADEPTIA

Leading the **Integration** Revolution

Your business problems have changed.
Why hasn't your integration solution?

Thank You!

ADEPTIA