



Connecting the
Data-Driven Enterprise >



Lab Guide
TDM Optional Modules

Version 6.1

Copyright 2016 Talend Inc. All rights reserved.

Information in this document is subject to change without notice. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of those agreements. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or any means electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of Talend Inc.

Talend Inc.
800 Bridge Parkway, Suite 200
Redwood City, CA 94065
United States
+1 (650) 539 3200

Welcome to Talend Training

Congratulations on choosing a Talend training module. Take a minute to review the following points to help you get the most from your experience.

Technical Difficulty

Instructor-Led

If you are following an instructor-led training (ILT) module, there will be periods for questions at regular intervals. However, if you need an answer in order to proceed with a particular lab, or if you encounter a situation with the software that prevents you from proceeding, don't hesitate to ask the instructor for assistance so it can be resolved quickly.

Self-Paced

If you are following a self-paced, on-demand training (ODT) module, and you need an answer in order to proceed with a particular lab, or you encounter a situation with the software that prevents you from proceeding with the training module, a Talend Support Engineer can provide assistance. Double-click the **Live Expert** icon on your desktop and follow the instructions to be placed in a queue. After a few minutes, a Support Engineer will contact you to determine your issue and help you on your way. Please be considerate of other students and only use this assistance if you are having difficulty with the training experience, not for general questions.

Exploring

Remember that you are interacting with an actual copy of the Talend software, not a simulation. Because of this, you may be tempted to perform tasks beyond the scope of the training module. Be aware that doing so can quickly derail your learning experience, leaving your project in a state that is not readily usable within the tutorial, or consuming your limited lab time before you have a chance to finish. For the best experience, stick to the tutorial steps! If you want to explore, feel free to do so with any time remaining after you've finished the tutorial (but note that you cannot receive assistance from Tech Support during such exploration).

Additional Resources

After completing this module, you may want to refer to the following additional resources to further clarify your understanding and refine and build upon the skills you have acquired:

- » Talend product documentation (help.talend.com)
- » Talend Forum (talendforge.org/)
- » Documentation for the underlying technologies that Talend uses (such as Apache) and third-party applications that complement Talend products (such as MySQL Workbench)

**This page intentionally left blank to ensure new chapters
start on right (odd number) pages.**

TALEND CONTE NT

LESSON 1 Working with SWIFT Structures

Overview	10
Lesson Overview	10
Objectives	10
Creating the SWIFT Structure	11
Overview	11
Starting Talend Studio	11
Creating the SWIFT Structure	12
Edit Properties	13
Create New Element	14
Show Document	16
Create New Elements	18
Next Step	19
Creating the F30 and F20 Fields	20
Overview	20
Creating the F30 Structure	20
Creating a New F30 Element in MT521-DOC	23
Creating the F20 Structure	25
Creating a New F20 Element in MT521-DOC	25
Next Step	27
Creating the F31P and F35B Fields	28
Creating the F31P Structure	28
Creating the F35B Structure	29
Creating New F31P and F35B Elements in MT521-DOC	32
Next Step	35
Creating the F35A and F83D Fields	36
Creating the F35A Structure	36
Creating the F83D Structure	37
Creating New F35A and F83D Elements in MT521-DOC	40
Next Step	41
Creating the F87D Fields	42
Creating the F87D Structure	42
Creating a New F87D Element in MT521-DOC	44
Next Step	45



Wrap-Up	46
Next Step	46

LESSON 2 Mapping SWIFT Files to MySQL

Overview	48
Lesson Overview	48
Objectives	48
Next Step	48
Setting Up the MySQL Connection	49
Next Step	52
Creating the DI Job	53
Creating the Mapping	60
Next Step	63
Performing the Mapping	64
Next Step	72
Running the Job	73
Checking the Results	74
Next Step	76
Wrap-Up	77
Next Step	77

LESSON 3 Mapping Multiple Record Types

Overview	80
Lesson Overview	80
Objectives	81
Next Step	81
Creating the Input Structure	82
Creating the A_Record Element	84
Creating the B_Record Element	85
Creating the C_Record Element	86
Creating the D_Record Element	88
Creating the Z_Record Record	89
Displaying a Sample Document	91
Next Step	93
Creating the Output Structure	94
Creating the AB_Record Element	96
Creating the AC_Record Element	97
Creating the AZ_Record Element	99
Next	101
Mapping the Elements	102
Adding Functions to Perform Computations	105
Test Run	113
Next Step	114
Creating the DI Job	115

Next Step	120
Wrap-Up	121
Next Step	121

LESSON 4 Performing a Database Lookup

Overview	124
Lesson Overview	124
Objectives	124
Next Step	124
Creating the Database Structure	125
Next Step	128
Creating the Map	129
Next Step	136
Wrap-Up	137
Next Step	137



**This page intentionally left blank to ensure new chapters
start on right (odd number) pages.**

LESSON

Working with SWIFT Structures

This chapter discusses the following.

Overview	10
Creating the SWIFT Structure	11
Creating the F30 and F20 Fields	20
Creating the F31P and F35B Fields	28
Creating the F35A and F83D Fields	36
Creating the F87D Fields	42
Wrap-Up	46



Overview

Lesson Overview

The objective of this lab is to create a SWIFT Structure and test it using an existing SWIFT file.

The example is based on the MT521 SWIFT part of the ISITC (Industry Standardisation for Institutional Trade Communication), an industry group that is developing a standard automated method for communicating trade instructions and reconciliation messages between Investment managers and Custodian banks. This eliminates the need for the Investment Management community to develop multiple proprietary linkages with the Custodian banks.

Objectives

After completing this lesson, you will be able to:

- » Create a SWIFT MT 521 Structure manually by defining message blocks and fields that match an available data file

Next Step

First, let's [create the SWIFT Structure itself](#) before creating each of its fields.

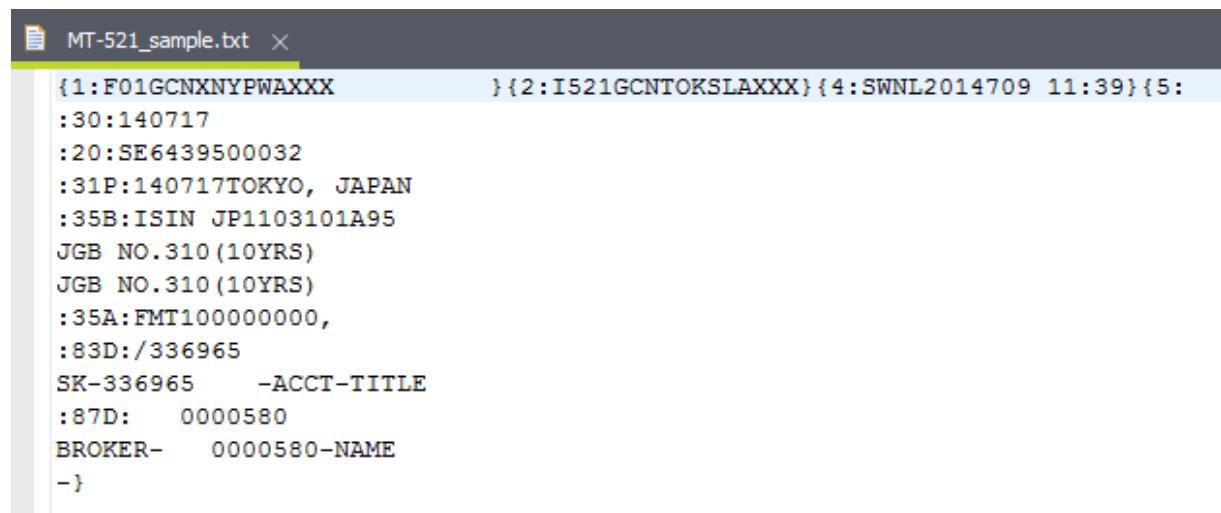
Creating the SWIFT Structure

Overview

The input file contains information stored within message blocks delimited by { }.

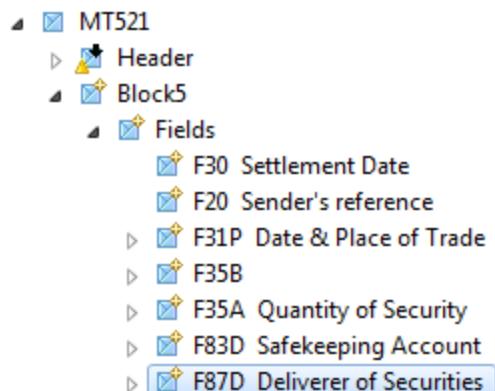
Each block can have fields starting with :nn: (where nn is some number). For example, block 5 below is the only one containing fields called 30, 20, 31P...

If a line does not start with :nn:, then it's a continuation of the previous field. For example, the two lines under :35B: that start with JGB are a continuation of the :35B: field.



```
MT-521_sample.txt X
{1:F01GCNXNYFWAXXX} {2:I521GCNTOKSLAXXX} {4:SWNL2014709 11:39} {5:
:30:140717
:20:SE6439500032
:31P:140717TOKYO, JAPAN
:35B:ISIN JP1103101A95
JGB NO.310 (10YRS)
JGB NO.310 (10YRS)
:35A:FMT100000000,
:83D:/336965
SK-336965 -ACCT-TITLE
:87D: 0000580
BROKER- 0000580-NAME
-}
```

The objective is to represent the input file in the following Structure, where *Header* is already prepared for you.

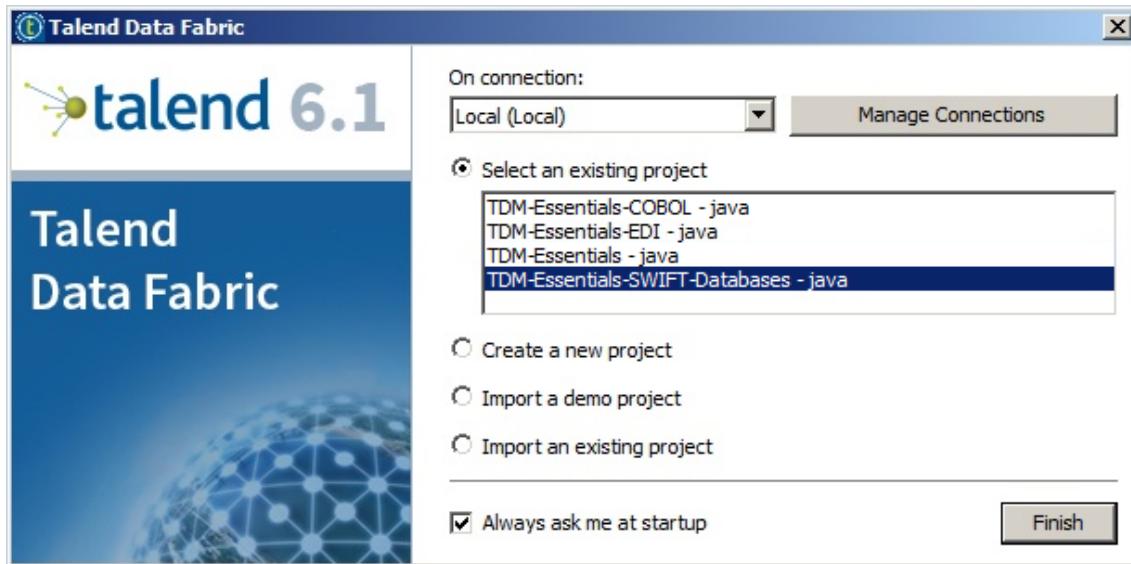


Starting Talend Studio

1. Click the **Talend Studio** link on the desktop.



2. Select the existing **TDM-Essentials-SWIFT-Databases** project and click **Finish**.

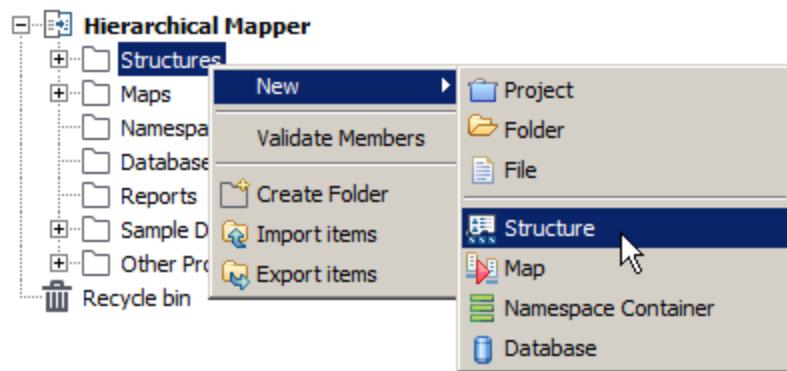


3. Click the **Start Now** button if the Welcome screen shows up. Otherwise, go to the next step.

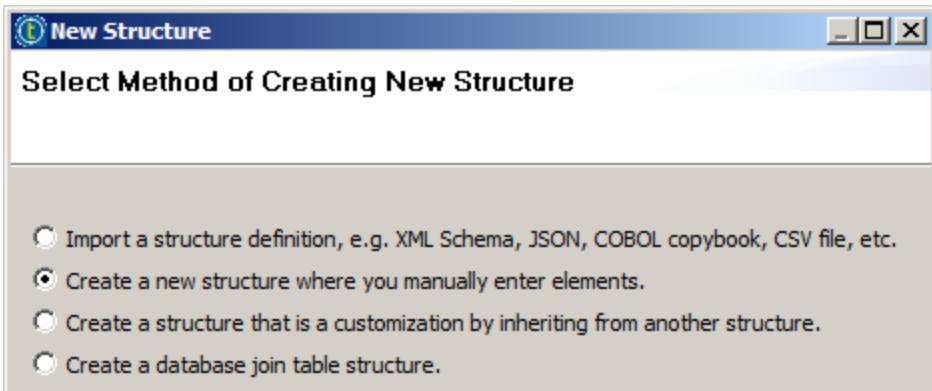
This is a screenshot of the "Getting Started" section of the Talend interface. It features a sidebar with links: Demos, Tutorials, Forums, and Training. Below the sidebar is a large button labeled "Start now!" which is highlighted with a red rectangular border. The background has a light blue gradient with a globe graphic.

Creating the SWIFT Structure

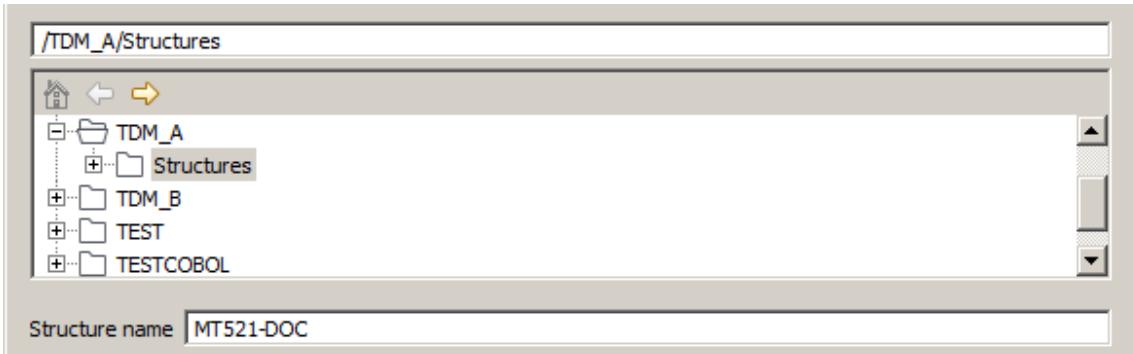
1. Right-click **Hierarchical Mapper > Structures** and select **New > Structure**.



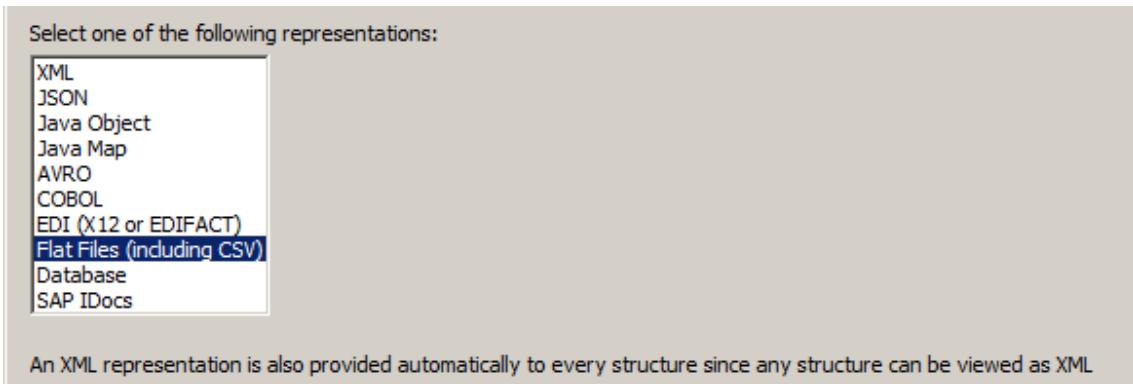
2. Select **Create a new structure where you manually enter elements** and click **Next >**



3. Name the new Structure **MT521-DOC**, keep the default location and click **Next**:

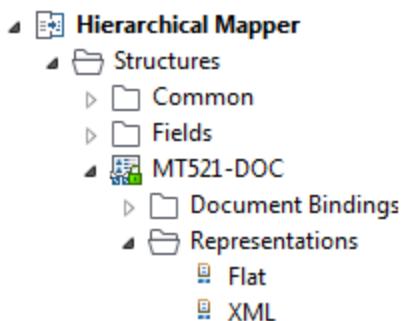


4. Select **Flat Files (including CSV)** for the representation and click **Finish**:

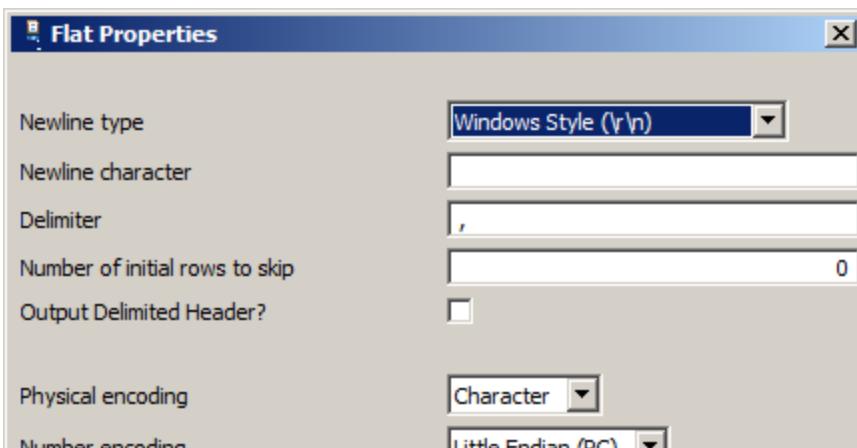


Edit Properties

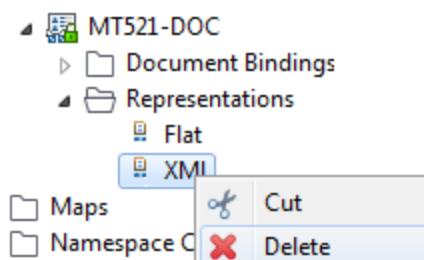
1. To edit the properties of the representation, double-click **Hierarchical Mapper > Structures > MT521-DOC > Representations > Flat**.



- Check that **Newline type** is set to **Windows Style (\r\n)**. If not, select it and click **OK**.

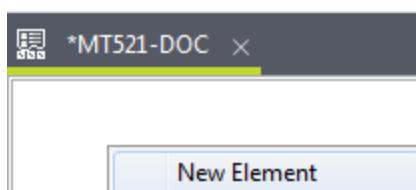


- Finally, delete the **XML** Representation by right-clicking it and selecting **Delete**, as it will not be used.

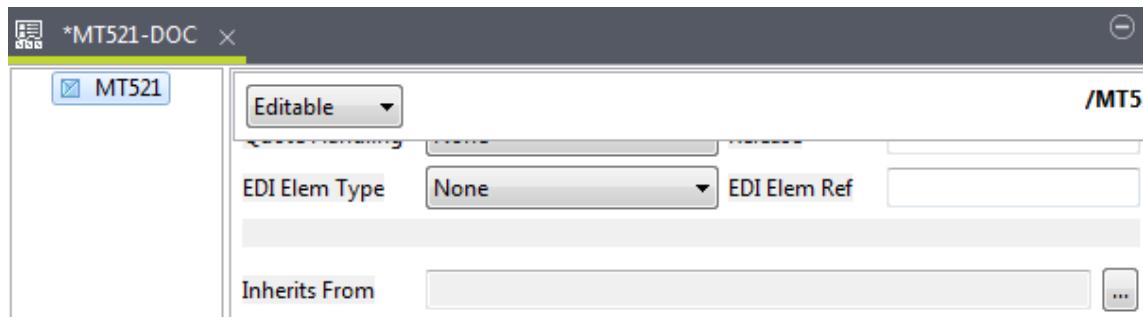


Create New Element

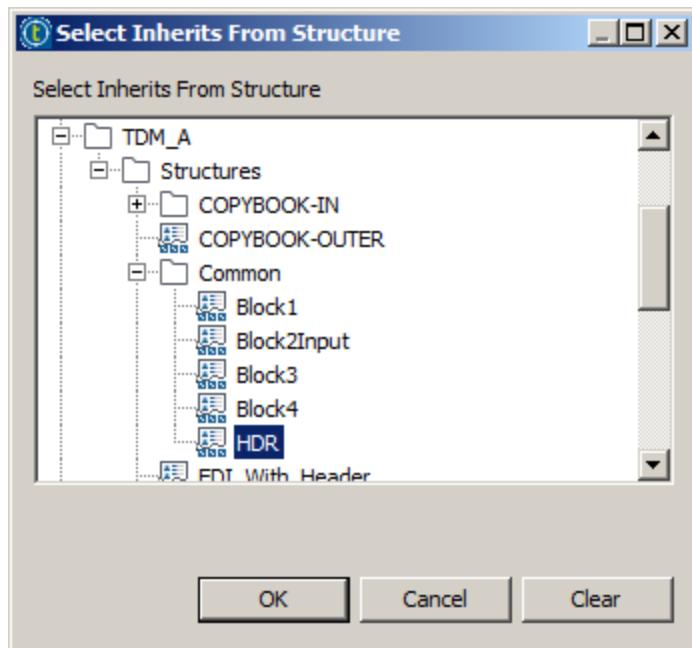
- In the new structure **MT521-DOC**, right-click the empty area and select **New Element**.



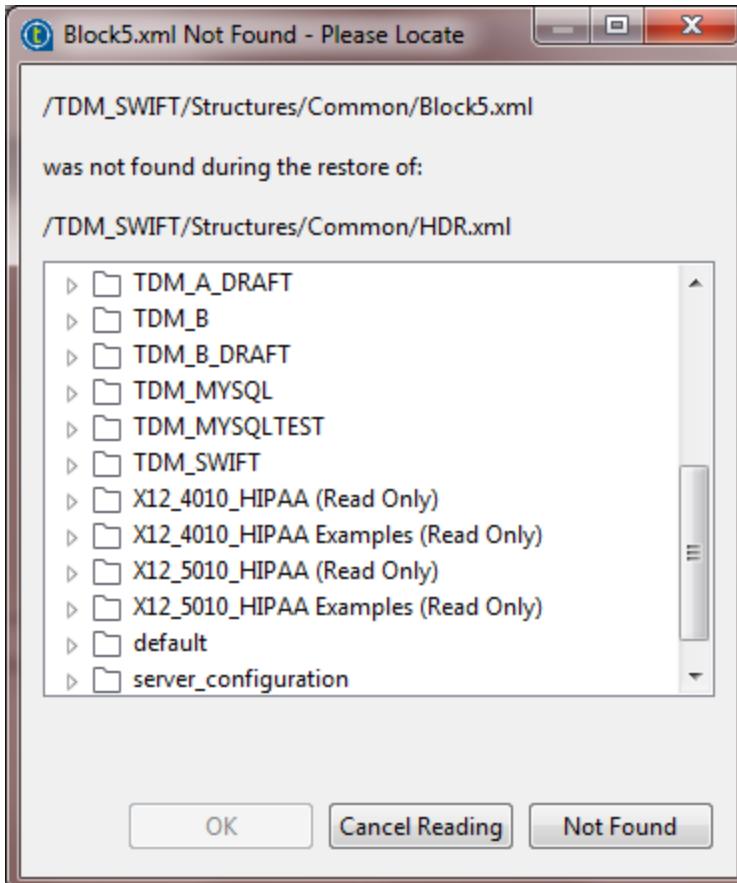
- Enter **MT521** for the name of the new element. Then scroll down to the **Inherits From** field and click the ... button next to it.



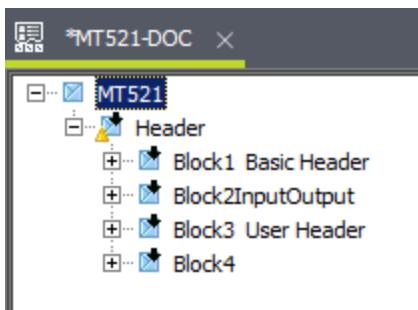
3. Select the **Structures > Common > HDR Structure** and click **OK**.



4. A message will indicate that **Block5** could not be found. Click the **Not found** button to proceed.

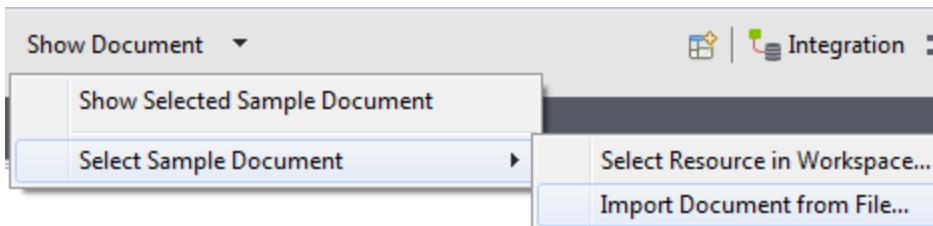


5. Expand the structure and check that it looks like the following:

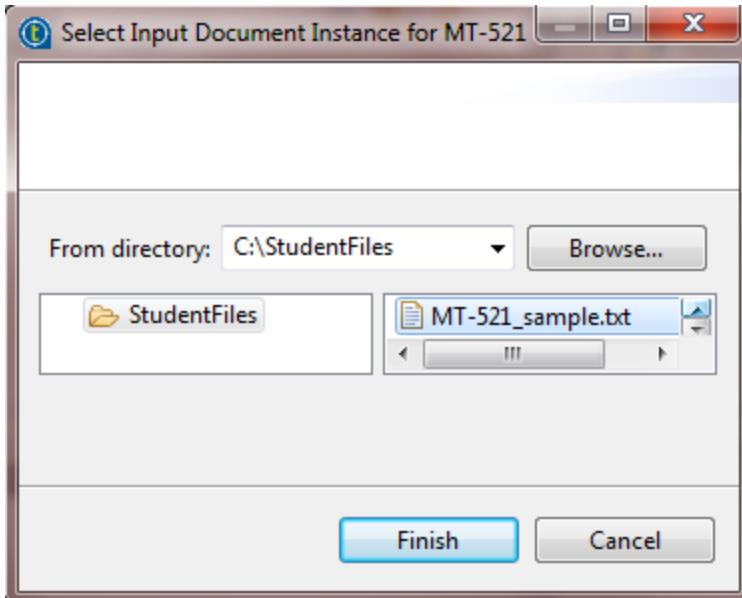


Show Document

1. Expand the **Show Document** menu and select **Select Sample Document > Import Document from File**.



2. Select *MT-521_sample.txt* from *C:\StudentFiles* and click **Finish**.



3. Select *Block1 Basic Header* in the Structure and check the following data is highlighted.

The screenshot shows the MT521-DOC editor interface. The title bar says '*MT521-DOC'. The left pane displays the structure of the document:

- MT521
 - Header
 - Block1 Basic Header
 - Block2InputOutput
 - Block3 User Header
 - Block4

The 'Block1 Basic Header' node is selected. The right pane shows its properties:

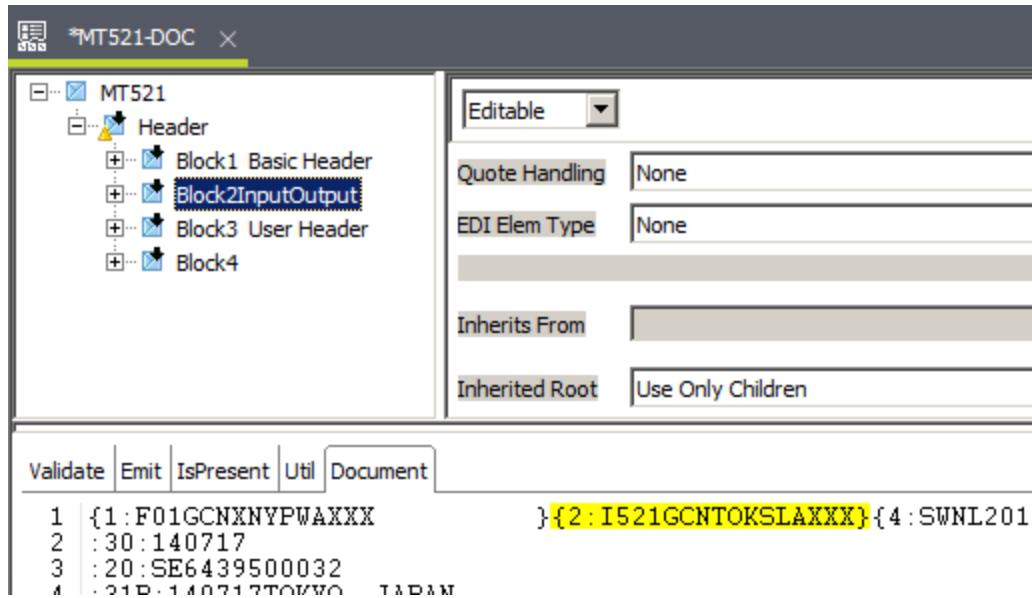
- Editable: None
- Quote Handling: None
- EDI Elem Type: None
- Inherits From: /TDM_A/Structures/Common/Block1
- Inherited Root: Use Everything

The bottom pane shows the EDI code:

```
1 {1:F01GCNXNYPWAXXX} {2:I521GCNTOKSLAXXX} {4:SWNL201
2 :30:140717
3 :20:SE6439500032
4 :21D:140717T00M00 TADAM
```

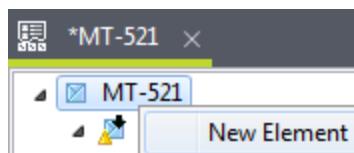
Line 1 of the code is highlighted in yellow.

4. Select *Block2InputOutput* in the Structure and check the following data is highlighted.

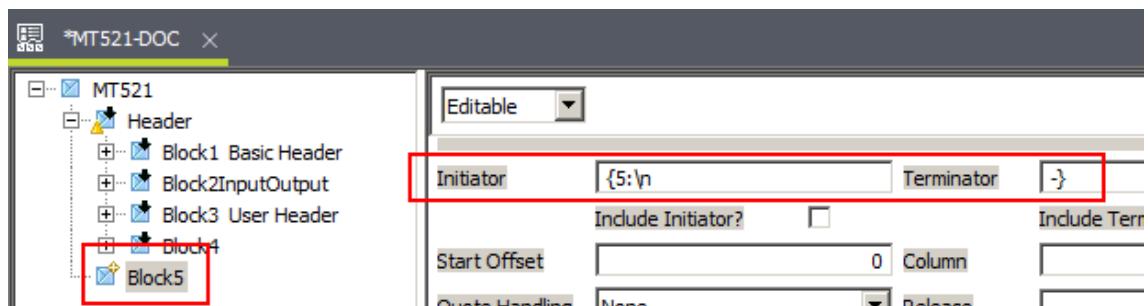


Create New Elements

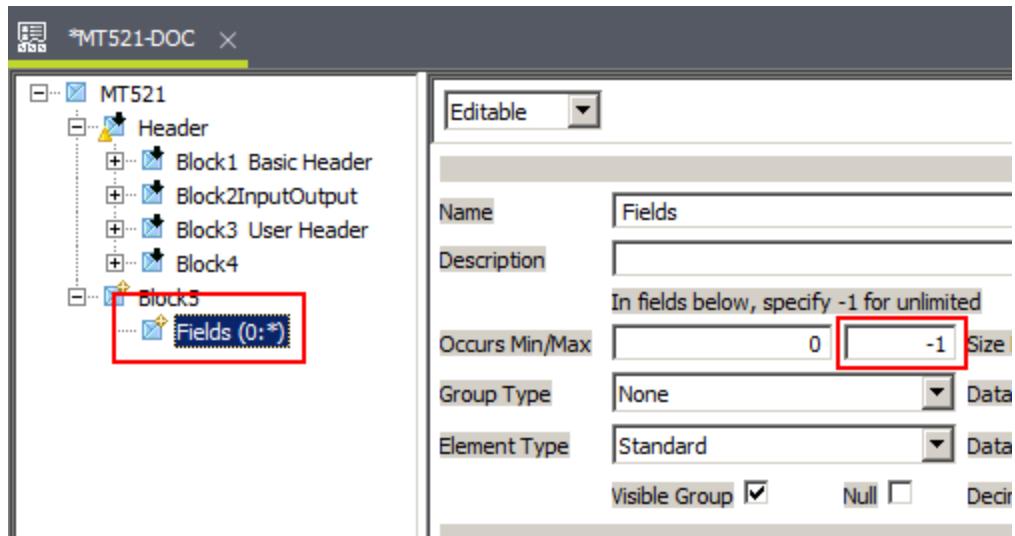
1. Right-click element **MT521** and click **New Element**.



2. Enter **Block5** for the name, **{5:\n}** in the **Initiator** field and **-}** in the **Terminator** field.



3. Finally, add a new *Fields* element under *Block5*. Enter 0 and -1 in **Occurs Min/Max** to make it a looping element.



Next Step

With the new Structure created, let's [add fields 30 and 20](#) under the *Fields* node.

Creating the F30 and F20 Fields

Overview

You will now create and prepare all the Sub Structures that will be inherited from in the MT521 Structure. They will be created in the **Hierarchical Mapper > Structures > Fields** folder. This helps in organizing your work.

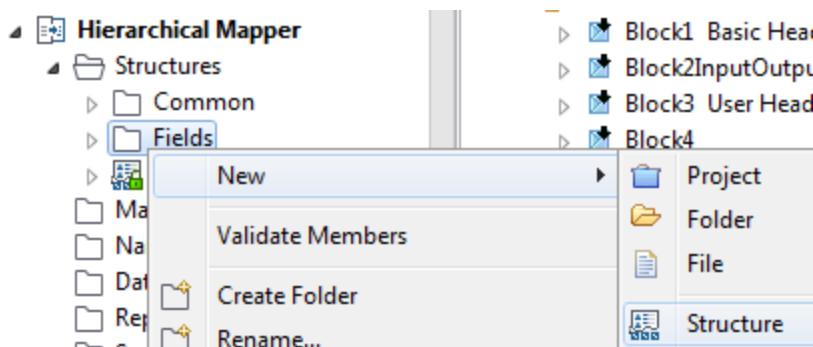
In this exercise, you will add one or two fields at a time to the *MT521-DOC* Structure and test it to make sure it is highlighting correctly. Due to the way the data mapper processes the data, you will need to create the next field before you can test the current one. For example, you will need to create both the *30D* field and the *20D* field before you can test the *30D* field.

Each of the fields will be set up as a separate Structure and then **inherited** into the *MT521-DOC* Structure. Note that a field can have one to many elements under it. It can also loop. All of the fields start with **:nn:**. Finally, if a field has multiple elements, you will need to create them under the main Structure (examples will be shown).

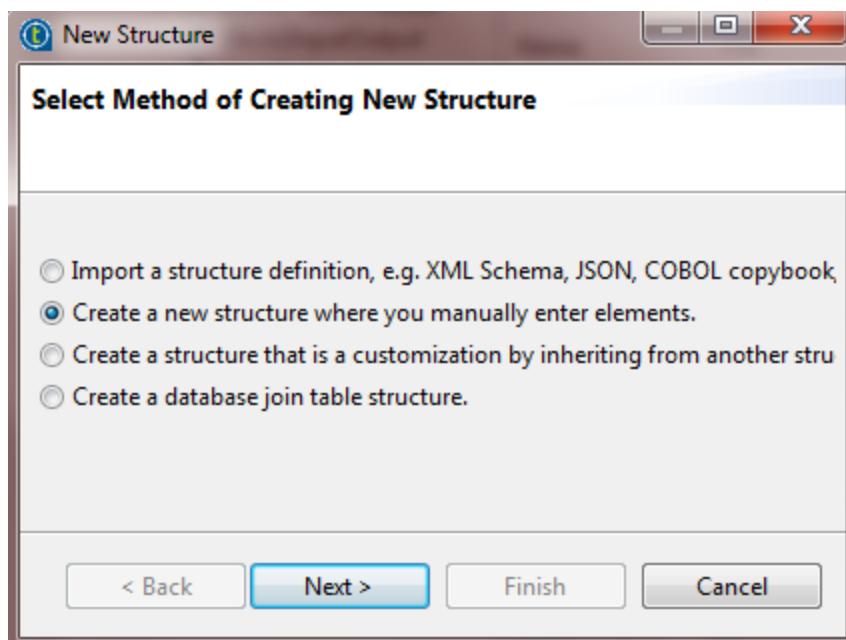
Field	Element	Type	Loop
F30	Settlement Date	String 6	0
F20	Sender's Reference	String 1-35	0

Creating the F30 Structure

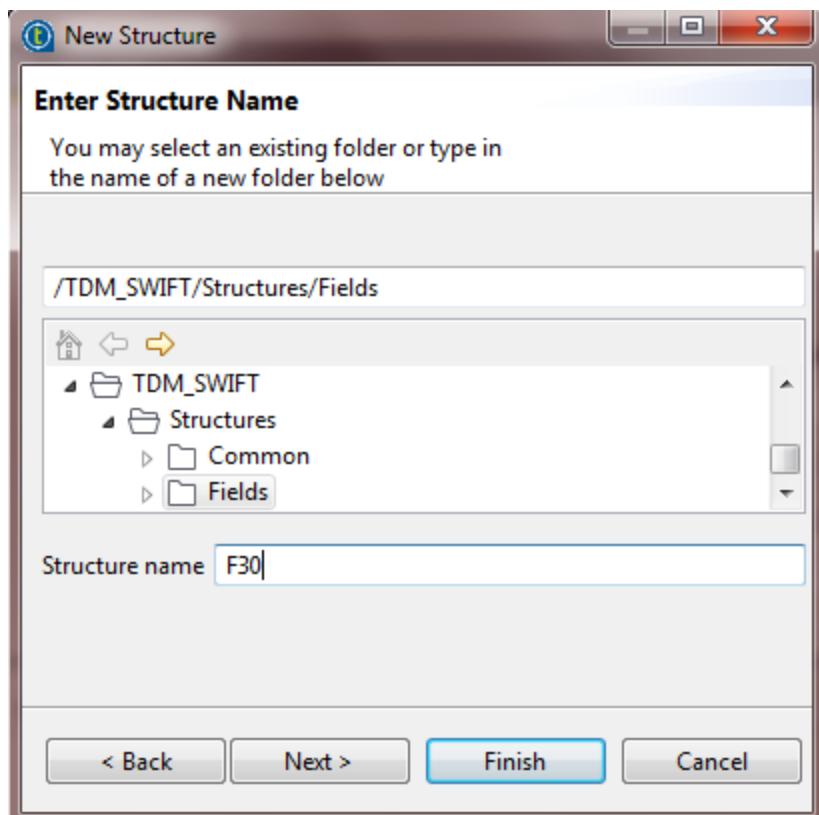
1. Right-click **Hierarchical Mapper > Structures > Fields** and select **New > Structure**.



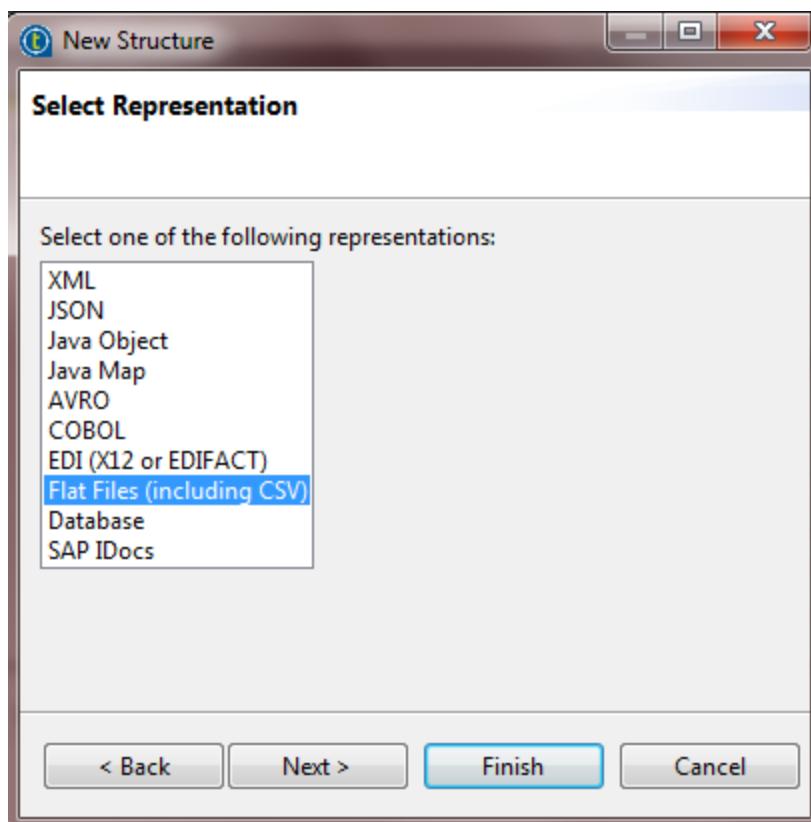
2. Select **Create a new structure where you manually enter elements** and click **Next >**.



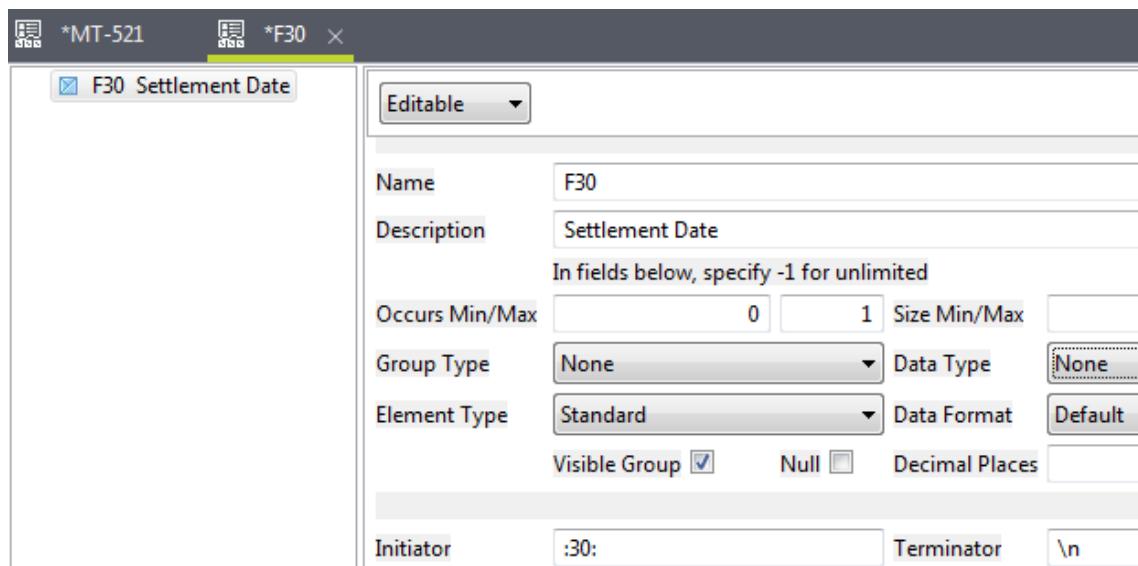
3. Name the new Structure F30 and click **Next >**:



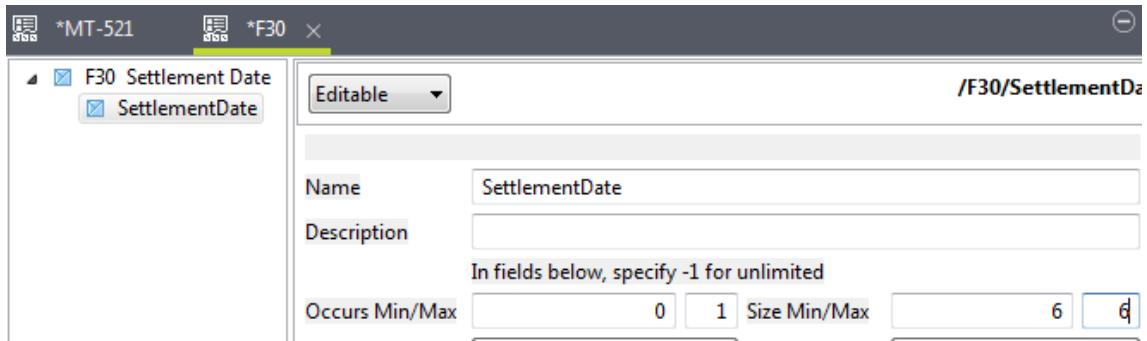
4. Select **Flat Files (including CSV)** for the representation and click **Finish**.



5. In the new Structure, right-click the empty area and select **New Element**. Enter **F30** for the **Name** and **Settlement Date** for the **Description**, then enter **:30:** in the field **Initiator** and **\n** in the field **Terminator**. **Data Type** should also be set to **None**.

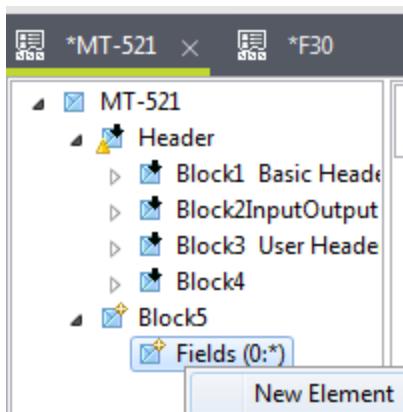


6. Right-click **F30 Settlement Date** and select **New Element**. Name it **SettlementDate** and enter **6** for **Size Min/Max** in both fields. Finally, set **Data Type** to **String**.

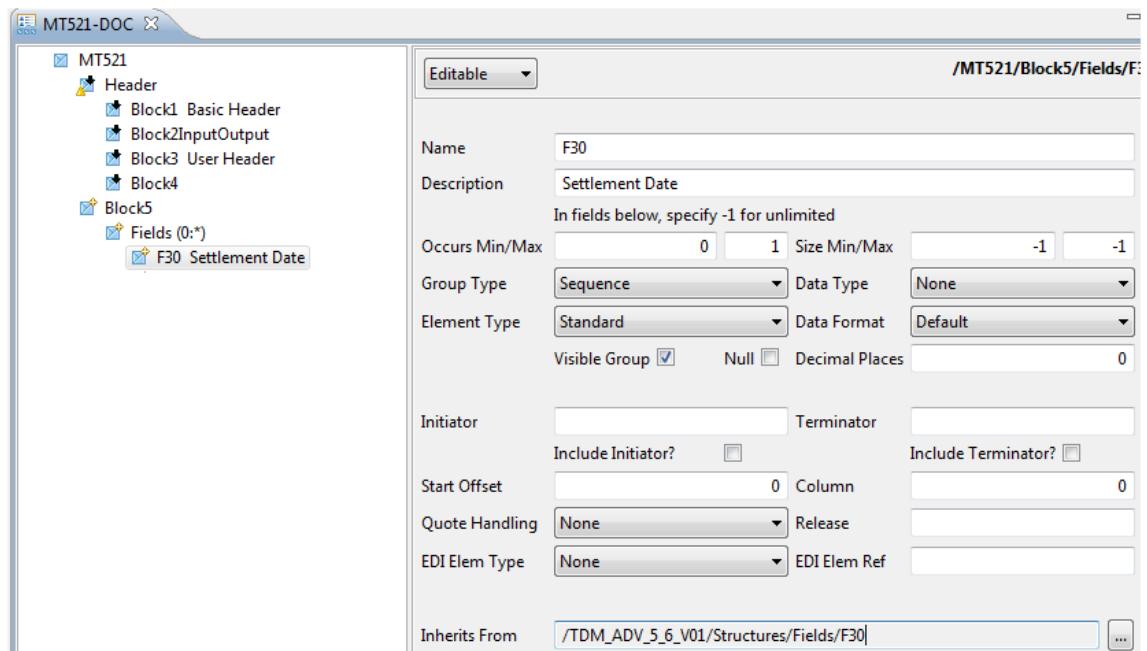


Creating a New F30 Element in MT521-DOC

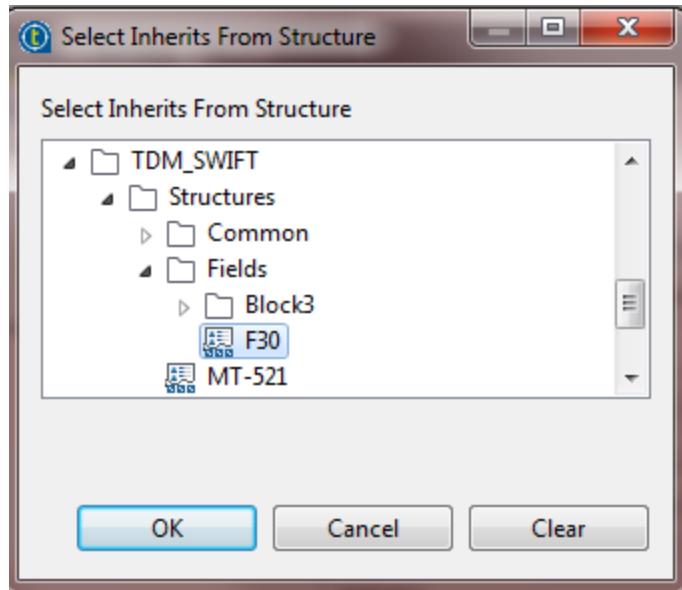
1. Switch back to the *MT521-DOC* Structure. Right-click the *Fields* element then select **New Element**.



2. Enter *F30* for the **Name** and *Settlement Date* for the **Description**. Finally, click the ... button next to the **Inherits From** field:



3. Select the **Structures > Fields > F30** Structure and click **OK**.



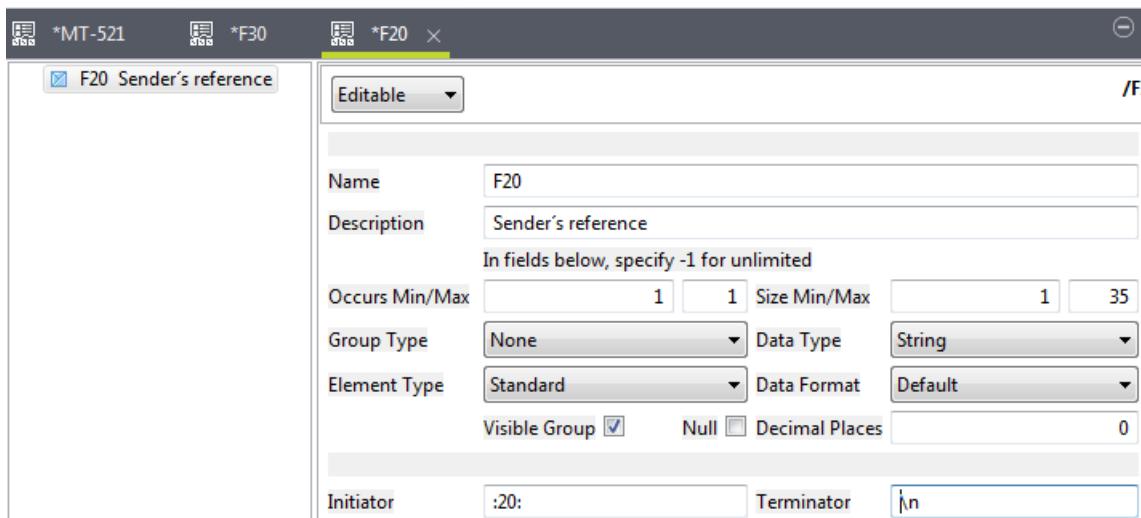
4. Finally, set **Inherited Root** to *Use Everything* for the *F30 Settlement Date* element and check the final screen looks like:

Editable		/MT-521/Block5/F
Name	F30	
Description	Settlement Date	
In fields below, specify -1 for unlimited		
Occurs Min/Max	0	1 Size Min/Max
Group Type	None	Data Type String
Element Type	Standard	Data Format Default
Visible Group	<input checked="" type="checkbox"/>	Null Decimal Places
Initiator		Terminator
Include Initiator?	<input type="checkbox"/>	Include Te
Start Offset	0	Column
Quote Handling	None	Release
EDI Elem Type	None	EDI Elem Ref
Inherits From	/TDM_SWIFT/Structures/Fields/F30	
Inherited Root	Use Everything	
Ignore Inherited Adds?	<input type="checkbox"/>	

Creating the F20 Structure

1. Right-click **Hierarchical Mapper > Structures > Fields** and select **New > Structure**.
2. Select **Create a new structure where you manually enter elements** and click **Next >**.
3. Name the new Structure *F20* and click **Finish**.
4. Right-click the empty area and select **New Element** to create a new element with the following properties:

Property	Value
Name	F20
Description	Sender's Reference
Occurs Min/Max	1 / 1
Size Min/Max	1 / 35
Data Type	String
Initiator	:20:
Terminator	\n



Creating a New F20 Element in MT521-DOC

1. Switch back to the *MT521-DOC* Structure.
2. Right-click **Fields** and select **New Element** to create a new element with the following properties.

Property	Value
Name	F20
Description	Sender's Reference
Inherits From	Structures/Fields/F20
Inherited Root	Use Everything

The screenshot shows the TDM interface for configuring fields in MT-521. The left pane displays the message structure with sections: Header, Block5, and Fields (0:*) containing F30 and F20. The right pane is the configuration panel for field F20, with the following settings:

- Name:** F20
- Description:** Sender's reference
- Occurs Min/Max:** 0..1
- Group Type:** None
- Data Type:** String
- Element Type:** Standard
- Data Format:** Default
- Visible Group:**
- Null:**
- Decimal Places:** 0
- Initiator:** [empty]
- Terminator:** [empty]
- Include Initiator?**
- Include Terminator?**
- Start Offset:** 0
- Column:** 0
- Quote Handling:** None
- Release:** [empty]
- EDI Elem Type:** None
- EDI Elem Ref:** [empty]
- Inherits From:** /TDM_SWIFT/Structures/Fields/F20
- Inherited Root:** Use Everything
- Ignore Inherited Adds?**

- Finally, select the *SettlementDate* element to verify that the data is highlighted correctly. Remember you had to specify both F30 and F20 to be able to test just F30 (the first one in the list).

The screenshot shows the TDM interface with the SettlementDate element selected in the tree view. The left pane displays the message structure with sections: Header, Block5, and Fields (0:*) containing F30 and F20. The right pane shows the selected SettlementDate element highlighted in yellow.

Next Step

Let's [add the next two fields, F31P and F35B.](#)

Creating the F31P and F35B Fields

This section adds two new fields to the list:

Field	Element	Type	Loop
F31P	Trade Date	String 6	0
	Settlement Location	String 1-29	0
F35B	Sender's Reference	String 1-35	0
	Security Info	String 1-35	0-2

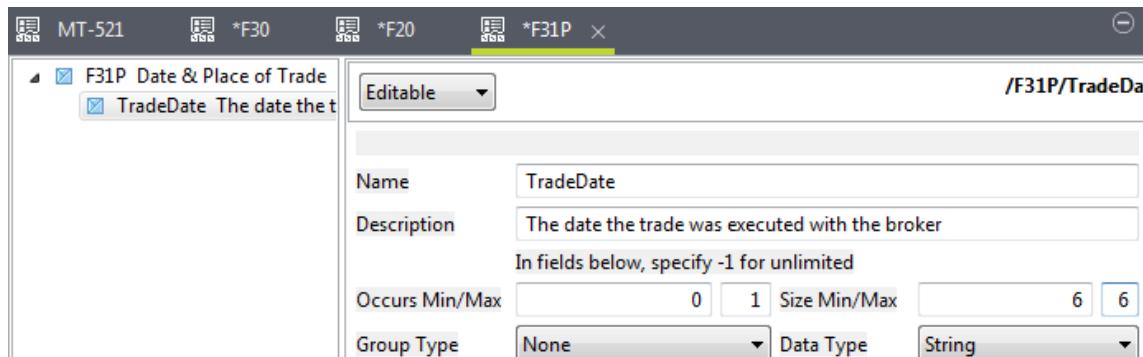
Creating the F31P Structure

1. Right-click **Hierarchical Mapper > Structures > Fields** and select **New > Structure**.
2. Select **Create a new structure where you manually enter elements** and click **Next >**.
3. Name the new Structure *F31P* and click **Finish**.
4. Right-click the empty area and select **New Element** to create a new element with the following properties:

Property	Value
Name	F31P
Description	Date & Place of Trade
Data Type	None
Initiator	:31P:
Terminator	\n

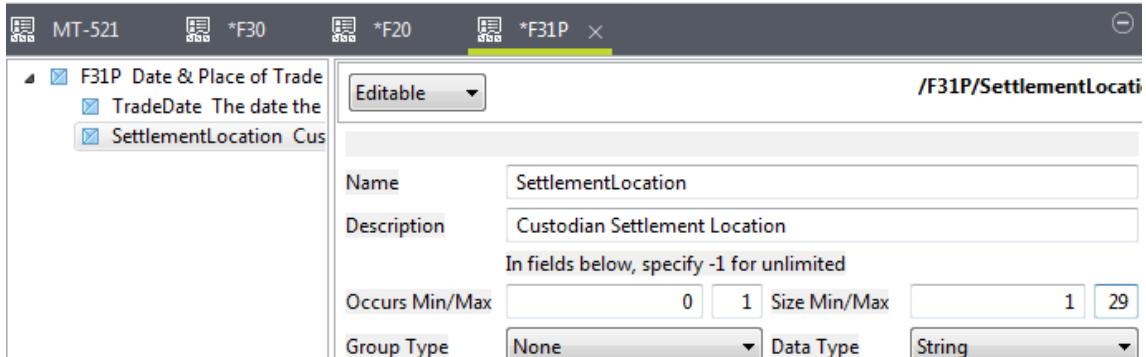
5. Right-click *F31P Date & Place of Trade* and select **New Element** to create a new child with the following properties:

Property	Value
Name	TradeDate
Description	The date the trade was executed with the broker
Size Min/Max	6/6
Data Type	String



6. Right-click *F31P Date & Place of Trade* again and select **New Element** to create another child with the following properties:

Property	Value
Name	SettlementLocation
Description	Custodian Settlement Location
Size Min/Max	1/29
Data Type	String



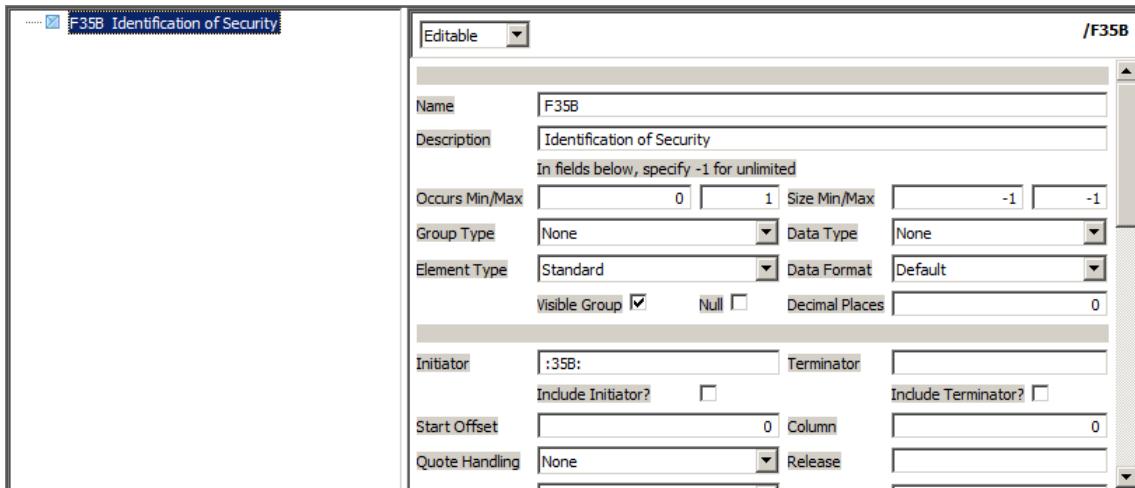
- Save your changes.

Creating the F35B Structure

- Right-click **Hierarchical Mapper > Structures > Fields** and select **New > Structure**.
- Select **Create a new structure where you manually enter elements** and click **Next >**.
- Name the new Structure **F35B** and click **Finish**.
- Right-click the empty area and select **New Element** to create a new element with the following properties:

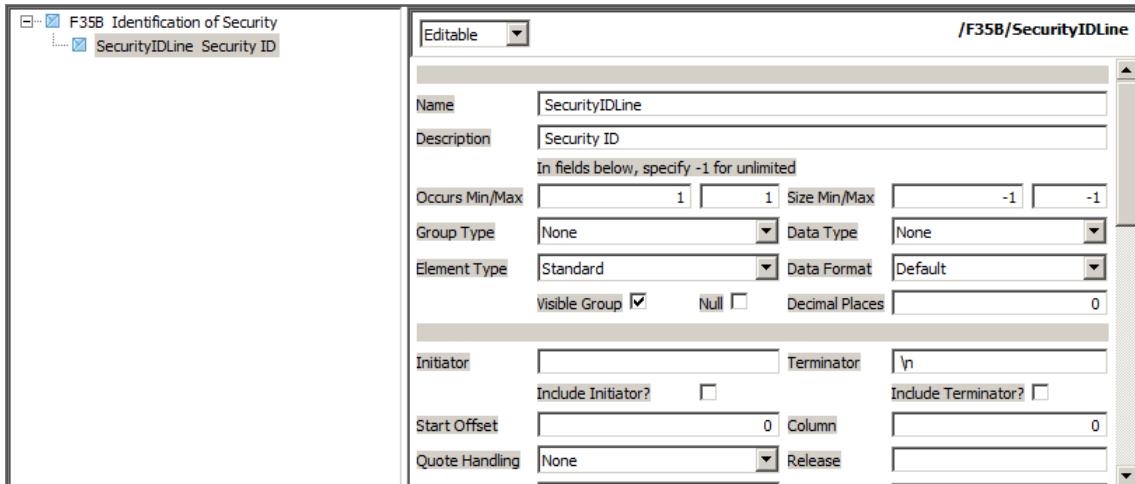
Property	Value
Name	F35B
Description	Identification of Security
Data Type	None
Initiator	:35B:

Note that, since this is a multiple line field, the terminator is not set to \n here. It will be set that way on the next sub element.



5. Right-click *F35B Identification of Security* and select **New Element** to create a new child with the following properties.

Property	Value
Name	SecurityIDLine
Description	Security ID
Occurs Min/Max	1/1
Data Type	None
Terminator	\n



6. Right-click *SecurityIDLine Security ID* and select **New Element** to create a new child with the following properties.

Property	Value
Name	ISIN_ID
Description	ISIN Security ID

Property	Value
Occurs Min/Max	1/1
Size Min/Max	1/35
Data Type	String

7. Right-click *F35B Identification of Security* and select **New Element** to create a new child with the following properties.

Property	Value
Name	SecurityInfoLine
Description	Security Info
Occurs Min/Max	0/2
Data Type	None
Terminator	\n

8. Right-click *SecurityInfoLine Security Info* and select **New Element** to create a new child with the following properties.

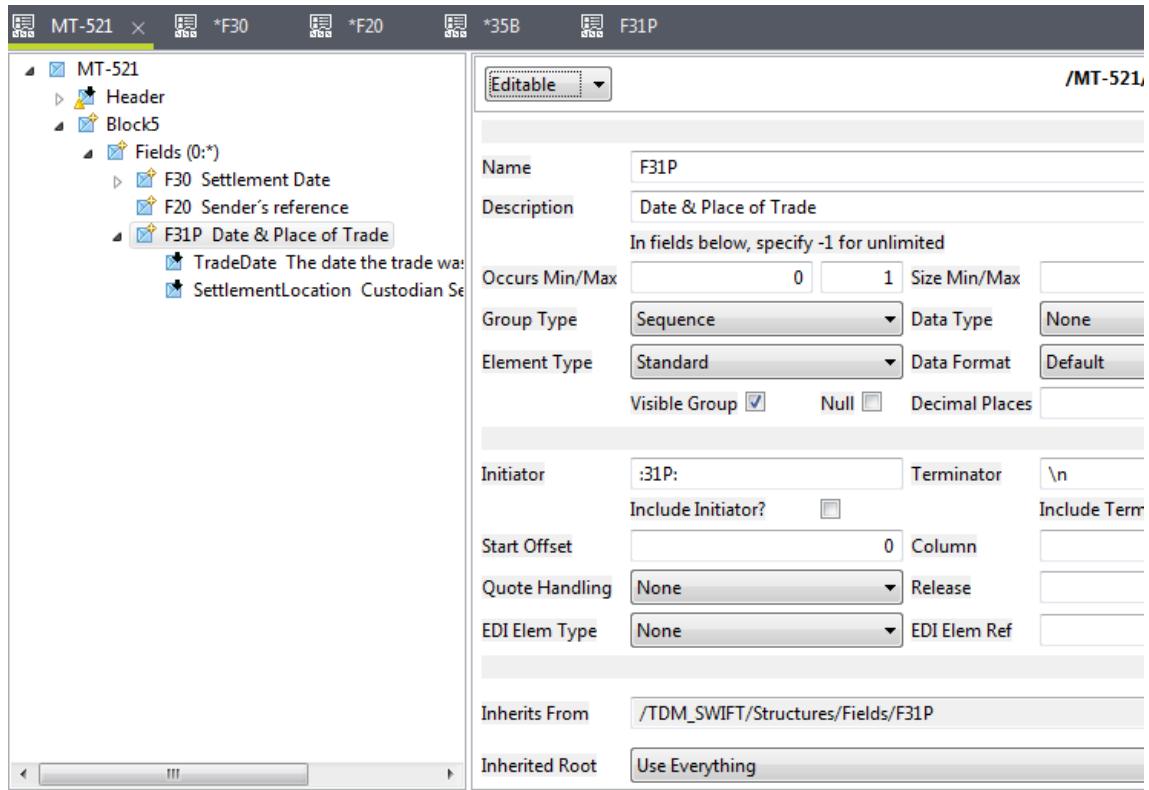
Property	Value
Name	Security_Info
Description	Name and description of the security
Occurs Min/Max	1/1
Size Min/Max	1/35
Data Type	String

The screenshot shows the MT521-DOC interface with the 'Fields' tool open. On the left, a tree view shows the structure under 'F35B Identification of Security'. On the right, a detailed configuration window for the 'Security_Info' element is displayed. The element has a name of 'Security_Info' and a description of 'Name and description of the security'. It is defined as occurring once (1) with a size of 1 to 35 characters. The data type is set to 'String'. The element type is 'Standard'. The 'Visible Group' checkbox is checked. There are options for initiators and terminators, but they are currently empty. The 'Quote Handling' is set to 'None'.

Creating New F31P and F35B Elements in MT521-DOC

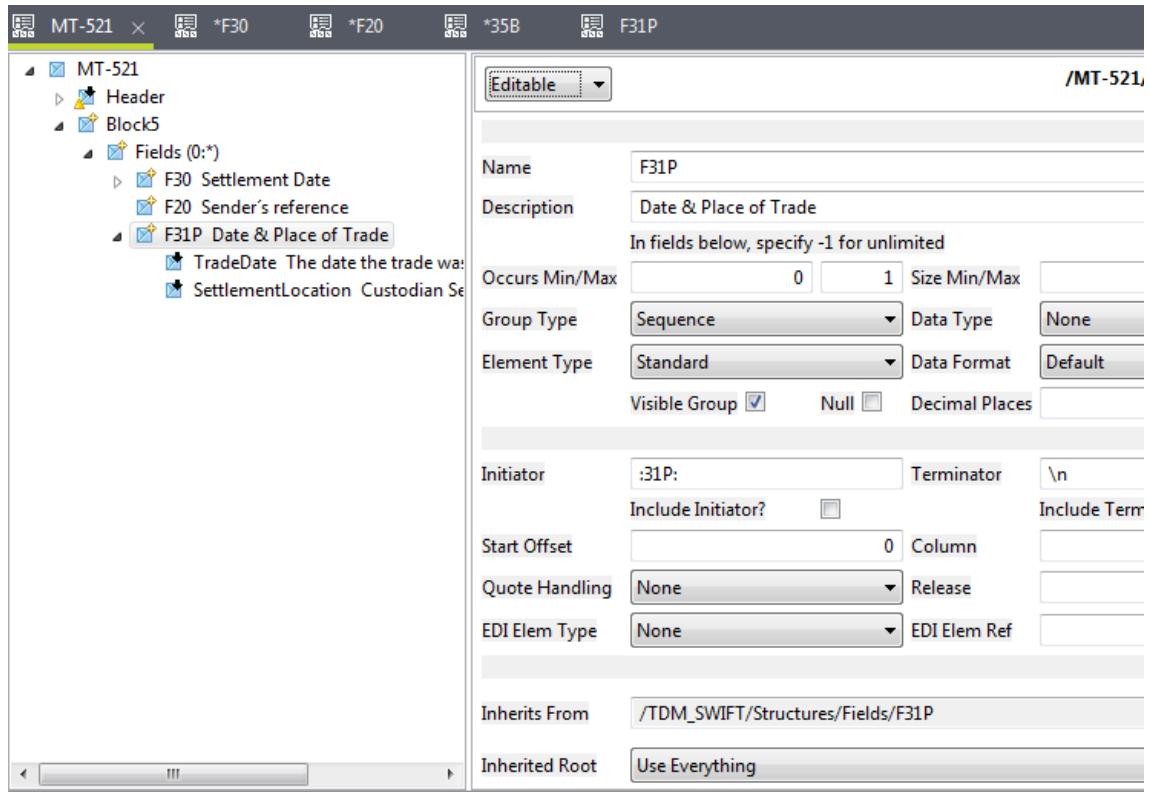
1. Switch back to the *MT521-DOC Structure*.
2. Right-click **Fields** and select **New Element** to create a new element with the following properties.

Property	Value
Name	F31P
Description	Date & Place of Trade
Inherits From	Structures/Fields/F31P
Inherited Root	Use Everything

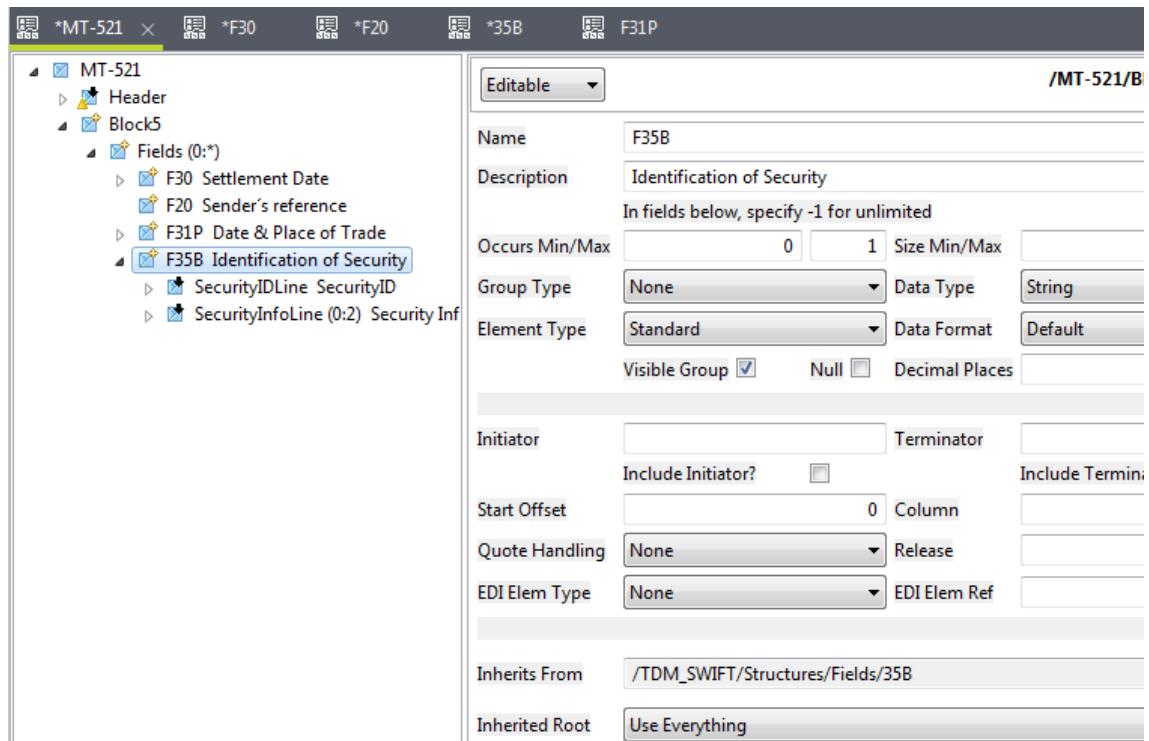


- Right-click **Fields** and select **New Element** again to create another new element with the following properties.

Property	Value
Name	F35B
Description	Identification of Security
Inherits From	Structures/Fields/F35B
Inherited Root	Use Everything



4. Check the final result looks like the following.



Next Step

Let's [add two more fields, F35A and F83D.](#)

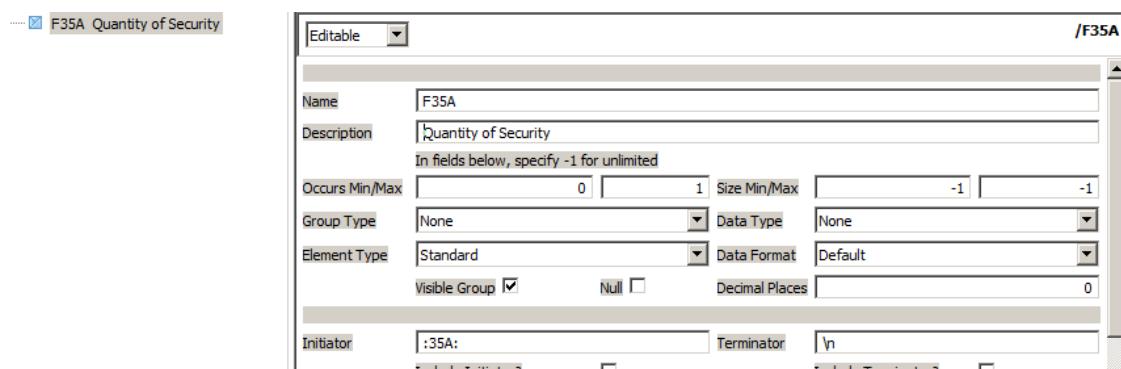
Creating the F35A and F83D Fields

Field	Element	Type	Loop
F35A	Prefix	String 1-3	0
	Quantity	String 1-15	0
F83D	Account Number	String 1-34	0
	Account Name	String 1-35	0

Creating the F35A Structure

1. Right-click **Hierarchical Mapper > Structures > Fields** and select **New > Structure**.
2. Select **Create a new structure where you manually enter elements** and click **Next >**.
3. Name the new Structure *F35A* and click **Finish**.
4. Right-click the empty area and select **New Element** to create a new element with the following properties:

Property	Value
Name	F35A
Description	Quantity of Security
Data Type	None
Initiator	:35A:
Terminator	\n



5. Right-click *F35A Quantity of Security* and select **New Element** to create a new child with the following properties:

Property	Value
Name	Prefix
Size Min/Max	1/3
Data Type	String

6. Right-click **F35A Quantity of Security** again and select **New Element** to create another child with the following properties:

Property	Value
Name	Quantity
Size Min/Max	1/15
Data Type	String

7. Save the changes to the Structure.

Creating the F83D Structure

1. Right-click **Hierarchical Mapper > Structures > Fields** and select **New > Structure**.
2. Select **Create a new structure where you manually enter elements** and click **Next >**.
3. Name the new Structure **F83D** and click **Finish**.
4. Right-click the empty area and select **New Element** to create a new element with the following properties:

Property	Value
Name	F83D
Description	Safekeeping Account
Occurs Min/Max	1/1
Data Type	None
Initiator	:83D:

..... F83D Safekeeping Account

Editable /F83D

Name	F83D				
Description	Safekeeping Account				
In fields below, specify -1 for unlimited					
Occurs Min/Max	1 1	Size Min/Max	-1 -1		
Group Type	None	Data Type	None		
Element Type	Standard	Data Format	Default		
Visible Group	<input checked="" type="checkbox"/>	Null	<input type="checkbox"/>	Decimal Places	0
Initiator	:83D:	Terminator			
<input type="checkbox"/> Include Terminator		<input type="checkbox"/> Include Terminator			

5. Right-click **F83D Safekeeping Account** and select **New Element** to create a new child with the following properties:

Property	Value
Name	Account_Number_Line
Occurs Min/Max	1/1
Data Type	None
Terminator	\n

..... F83D Safekeeping Account
..... Account_Number_Line

Editable /F83D/Account_Number_Line

Name	Account_Number_Line				
Description					
In fields below, specify -1 for unlimited					
Occurs Min/Max	1 1	Size Min/Max	-1 -1		
Group Type	Sequence	Data Type	None		
Element Type	Standard	Data Format	Default		
Visible Group	<input checked="" type="checkbox"/>	Null	<input type="checkbox"/>	Decimal Places	0
Initiator		Terminator	\n		
<input type="checkbox"/> Include Terminator		<input type="checkbox"/> Include Terminator			

6. Right-click **Account_Number_Line** and select **New Element** to create a new child with the following properties:

Property	Value
Name	Account_Number
Size Min/Max	1/34
Data Type	String
Initiator	/

File Structure:

- F83D Safekeeping Account
 - Account_Number_Line
 - Account_Number

Properties for Account_Number:

Name	Value
Name	Account_Number
Description	
Occurs Min/Max	0 1
Group Type	None
Element Type	Standard
Initiator	/
Terminator	

7. Right-click F83D Safekeeping Account and select **New Element** to create a new child with the following properties:

Property	Value
Name	Account_Name_Line
Occurs Min/Max	1/1
Data Type	None
Terminator	\n

File Structure:

- F83D Safekeeping Account
 - Account_Number_Line
 - Account_Number
 - Account_Name_Line

Properties for Account_Name_Line:

Name	Value
Name	Account_Name_Line
Description	
Occurs Min/Max	1 1
Group Type	Sequence
Element Type	Standard
Initiator	
Terminator	\n

8. Right-click Account_Name_Line and select **New Element** to create a new child with the following properties:

Property	Value
Name	Account_Name
Description	Custodian Account Name
Occurs Min/Max	1/1
Size Min/Max	1/35
Data Type	String

- Save the changes to the Structure.

Creating New F35A and F83D Elements in MT521-DOC

- Switch back to the *MT521-DOC* Structure.
- Right-click **Fields** and select **New Element** to create a new element with the following properties.

Property	Value
Name	F35A
Description	Quantity of Security
Inherits From	Structures/Fields/F35A
Inherited Root	Use Everything

- Right-click **Fields** and select **New Element** again to create another new element with the following properties.

Property	Value
Name	F83D

Property	Value
Description	Safekeeping Account
Occurs Min/Max	1/1
Inherits From	Structures/Fields/F35B
Inherited Root	Use Everything

The screenshot shows the MT521 structure editor interface. On the left, a tree view displays the structure hierarchy under MT521, including Header, Block1 Basic Header, Block2InputOutput, Block3 User Header, Block4, and Block5. Block5 contains fields F30 Settlement Date, F20 Sender's Reference, F31P Date & Place of Trade, F35B Identification of Secur, F35A Quantity of Security, and F83D Safekeeping Account. The F83D field is currently selected. The right side of the screen is a detailed configuration dialog for the F83D field, with tabs for 'Editable' and 'Advanced'. The 'Editable' tab shows the path /MT521/Block5/Fields/F83D. Configuration options include:

- Name:** F83D
- Description:** (empty)
- Occurs Min/Max:** 1..1
- Group Type:** None
- Data Type:** String
- Element Type:** Standard
- Visible Group:** checked
- Null:** unchecked
- Decimal Places:** 0
- Initiator:** (empty)
- Terminator:** (empty)
- Include Initiator?**: unchecked
- Include Terminator?**: unchecked
- Start Offset:** 0
- Column:** (empty)
- Quote Handling:** None
- Release:** (empty)
- EDI Elem Type:** None
- EDI Elem Ref:** (empty)
- Inherits From:** /TDM_A/Structures/Fields/F83D
- Inherited Root:** Use Everything

- Save the changes to the Structure.

Next Step

Let's [add the final F87D field](#).

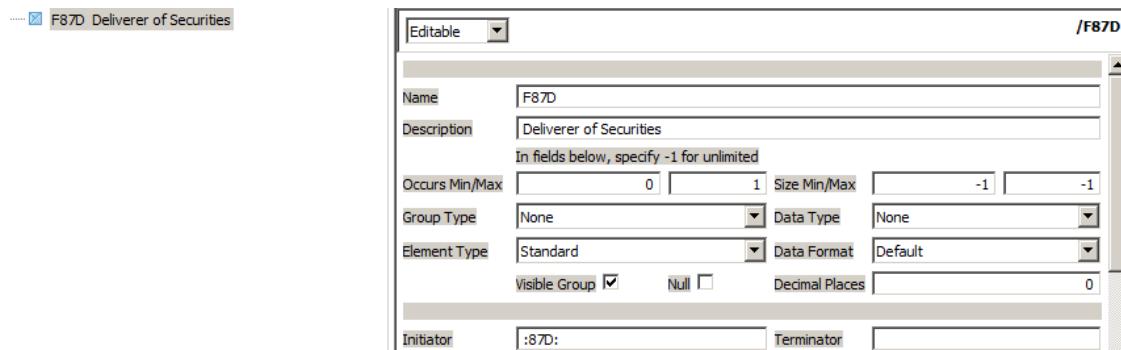
Creating the F87D Fields

Field	Element	Type	Loop
F87D	Code	String 1-34	0
	Broker Settlement Agent Name	String 1-35	0

Creating the F87D Structure

1. Right-click **Hierarchical Mapper > Structures > Fields** and select **New > Structure**.
2. Select **Create a new structure where you manually enter elements** and click **Next >**.
3. Name the new Structure **F87D** and click **Finish**.
4. Right-click the empty area and select **New Element** to create a new element with the following properties:

Property	Value
Name	F87D
Description	Deliverer of Securities
Data Type	None
Initiator	:87D:



5. Right-click **F87D Deliverer of Securities** and select **New Element** to create a new child with the following properties:

Property	Value
Name	Broker_Account_Number_Line
Occurs Min/Max	1/1
Data Type	None
Terminator	\n

Tree View:

- F87D Deliverer of Securities
 - Broker_Account_Number_Line

Properties Window:

Name	Broker_Account_Number_Line
Description	
Occurs Min/Max	1 1
Group Type	None
Element Type	Standard
Visible Group	<input checked="" type="checkbox"/>
Initiator	
Size Min/Max	-1 -1
Data Type	None
Data Format	Default
Null	<input type="checkbox"/>
Decimal Places	0
Terminator	\n

6. Right-click **Broker_Account_Number_Line** and select **New Element** to create a new child with the following properties:

Property	Value
Name	Code
Description	ISITC Code
Occurs Min/Max	1/1
Data Type	String

Tree View:

- F87D Deliverer of Securities
 - Broker_Account_Number_Line
 - Code ISITC Code

Properties Window:

Name	Code
Description	ISITC Code
Occurs Min/Max	1 1
Group Type	None
Element Type	Standard
Visible Group	<input checked="" type="checkbox"/>
Initiator	
Size Min/Max	-1 -1
Data Type	String
Data Format	Default
Null	<input type="checkbox"/>
Decimal Places	0
Terminator	

7. Right-click **F87D Deliverer of Securities** and select **New Element** to create a new child with the following properties:

Property	Value
Name	Broker_Agent_Name_Line
Occurs Min/Max	1/3
Data Type	None

The screenshot shows the MT521-DOC Structure Editor. On the left, there is a tree view of the structure:

- F87D Deliverer of Securities
 - Broker_Account_Number_Line
 - Code ISITC Code
 - Broker_Agent_Name_Line (1:3)** (highlighted in blue)

To the right is a properties panel for the selected element:

/F87D/Broker_Agent_Name_Line

Editable

Name	Broker_Agent_Name_Line		
Description			
Occurs Min/Max	1	3	Size Min/Max -1 -1
Group Type	None		
Element Type	Standard		
Visible Group	<input checked="" type="checkbox"/>	Null	<input type="checkbox"/>
Data Type	None		
Data Format	Default		
Decimal Places	0		
Initiator			
Terminator			

- Right-click **Broker_Agent_Name_Line** and select **New Element** to create a new child with the following properties:

Property	Value
Name	F87_Broker_Settlement_Agent_Name
Size Min/Max	1/35
Data Type	String
Terminator	\n

The screenshot shows the MT521-DOC Structure Editor. The tree view now includes the newly created element:

- F87D Deliverer of Securities
 - Broker_Account_Number_Line
 - Code ISITC Code
 - Broker_Agent_Name_Line (1:3)**
 - F87_Broker_Settlement_Agent_Name** (highlighted in blue)

The properties panel for the new element is shown:

/F87D/Broker_Agent_Name_Line/F87_Broker_Settlement_Agent_Name

Editable

Name	F87_Broker_Settlement_Agent_Name		
Description			
Occurs Min/Max	0	1	Size Min/Max 1 35
Group Type	None		
Element Type	Standard		
Visible Group	<input checked="" type="checkbox"/>	Null	<input type="checkbox"/>
Data Type	String		
Data Format	Default		
Decimal Places	0		
Initiator			
Terminator	\n		

- Save the changes to the Structure.

Creating a New F87D Element in MT521-DOC

- Switch back to the *MT521-DOC* Structure.
- Right-click **Fields** and select **New Element** to create a new element with the following properties.

Property	Value
Name	F87D
Description	Deliverer of Securities
Initiator	:87D:
Inherits From	Structures/Fields/F87D
Inherited Root	Use Everything

The screenshot shows the MT521 structure editor. On the left is a tree view of the message structure:

- MT521
 - Header
 - Block1 Basic Header
 - Block2InputOutput
 - Block3 User Header
 - Block4
 - Block5
 - Fields (0..*)
 - F30 Settlement Date
 - F20 Sender's Reference
 - F31P Date & Place of Trade
 - F35B Identification of Secur
 - F35A Quantity of Security
 - F83D Safekeeping Account
 - F87D Deliverer of Securities**

On the right is a detailed configuration window for the F87D field:

Editable /MT521/Block5/Fields/F87D

Name: F87D
Description: Deliverer of Securities
Occurs Min/Max: 0 1
Group Type: Sequence
Element Type: Standard
Visible Group: **Null:** **Decimal Places:** 0
Initiator: :87D:
Start Offset: 0
Quote Handling: None
EDI Elem Type: None
Inherits From: /TDM_A/Structures/Fields/F87D
Inherited Root: Use Everything

3. Save the changes to the Structure.
4. Select **F87_Broker_Settlement_Agent_Name** in the Structure and check that the data is highlighted correctly.

The screenshot shows the MT521 structure editor with the Broker_Agent_Name_Line element expanded. The F87_Broker_Settlement_Agent_Name field is highlighted in yellow.

Structure View:

- JGB Quantity of Security
 - F83D Safekeeping Account
 - F87D Deliverer of Securities**
 - Broker_Account_Number_Line
 - Broker_Agent_Name_Line (1:3)
 - F87_Broker_Settlement_Agent_Name**

Text View:

	Validate	Emit	IsPresent	Value	Util	Document
1	{1:F01GCNXNYPWAXXX					}
2	:30:140717					
3	:20:SE6439500032					
4	:31P:140717TOKYO, JAPAN					
5	:35B:ISIN JP1103101A95					
6	JGB NO.310(10YRS)					
7	JGB NO.310(10YRS)					
8	:35A:FMT1000000000,					
9	:83D:/336965					
10	SK-336965 -ACCT-TITLE					
11	:87D: 0000580					
12	BROKER- 0000580-NAME					
13	-}					

Next Step

This lesson is almost over. Head to the [Wrap-Up](#) section for a summary of the concepts reviewed in this lesson.

Wrap-Up

In this lesson, you learned how to:

- » Create a SWIFT MT 521 Structure manually by defining message blocks and fields that match an available data file

Next Step

Congratulations, you successfully completed this lesson. Click the **Check your status with this unit** button below in order to save your progress. Then click **Completed. Let's continue >** on the next screen to jump to the next lesson.

LESSON 2

Mapping SWIFT Files to MySQL

This chapter discusses the following.

Overview	48
Setting Up the MySQL Connection	49
Creating the DI Job	53
Performing the Mapping	64
Running the Job	73
Wrap-Up	77



Overview

Lesson Overview

The objective of this lab is to create a Job that reads from a SWIFT file and writes to a MySQL database.

Objectives

After completing this lesson, you will be able to:

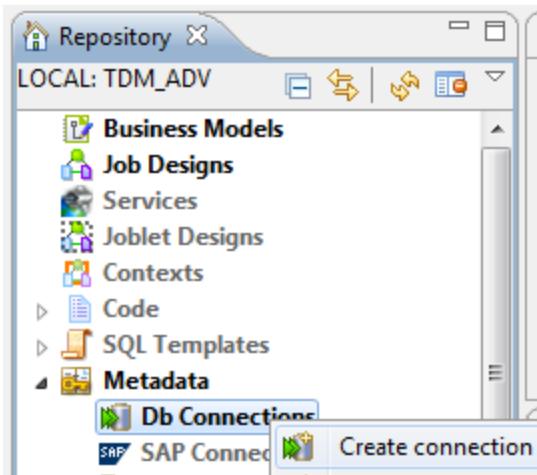
- » Set up a MySQL connection
- » Use a tHMap Component to read from a SWIFT file and write to a MySQL database

Next Step

First, let's [set up the MySQL connection.](#)

Setting Up the MySQL Connection

1. In the Integration perspective, right-click **Metadata > Db Connections** and select **Create connection**.



2. Provide details as illustrated then click **Next >**.

The dialog box is titled "Database Connection" and "New Database Connection on repository - Step 1/2". It contains the following fields:

Name	Training_SWIFT_MT521
Purpose	Connect to a MySQL Database
Description	(empty text area)
Author	user@talend.com
Locker	(empty text area)
Version	0.1
Status	(dropdown menu)
Path	(empty text area) <input type="button" value="Select"/>

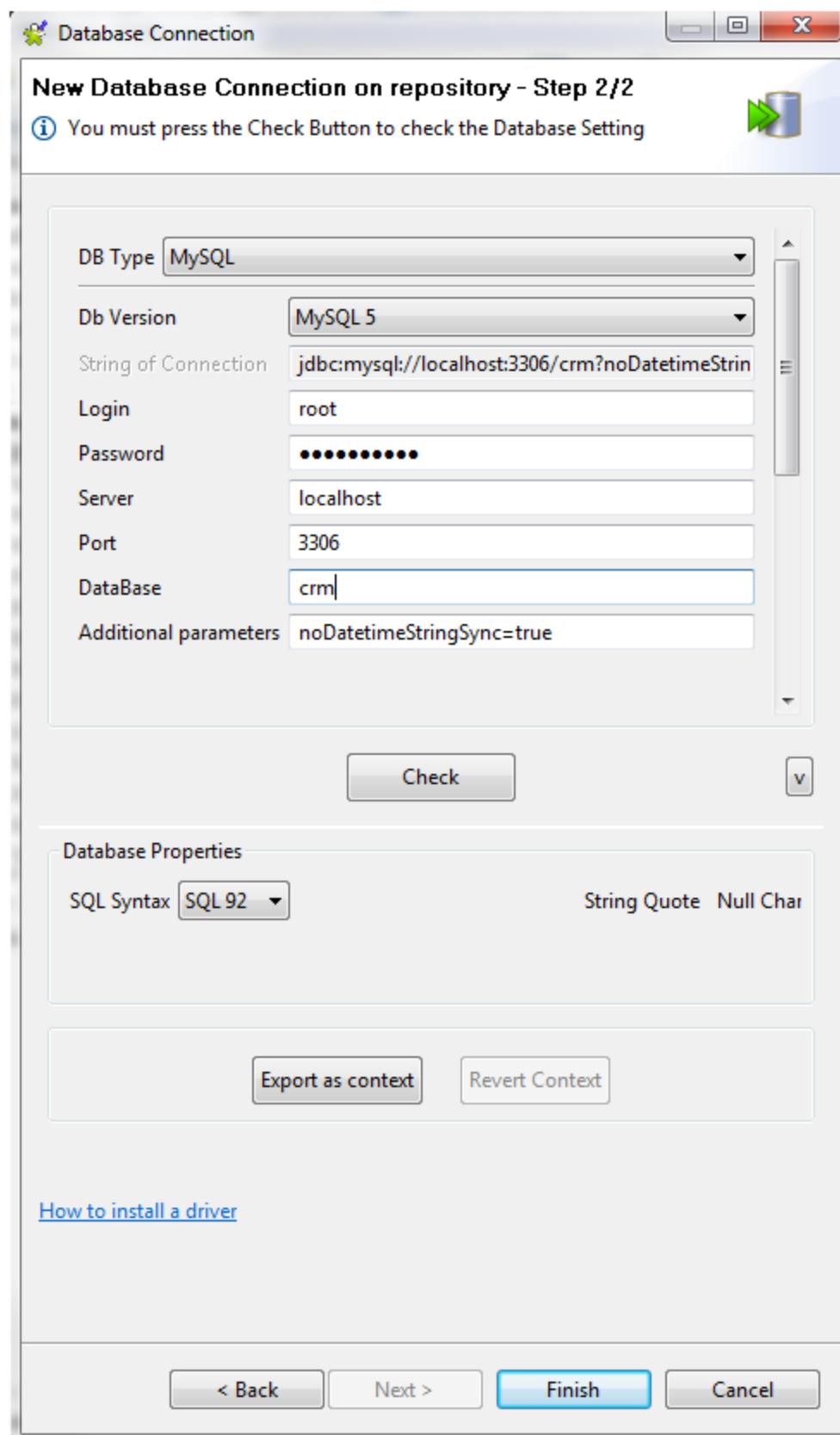
At the bottom, there are buttons: < Back, Next >, Finish, and Cancel.

3. Select **MySQL** for the **DB Type**.

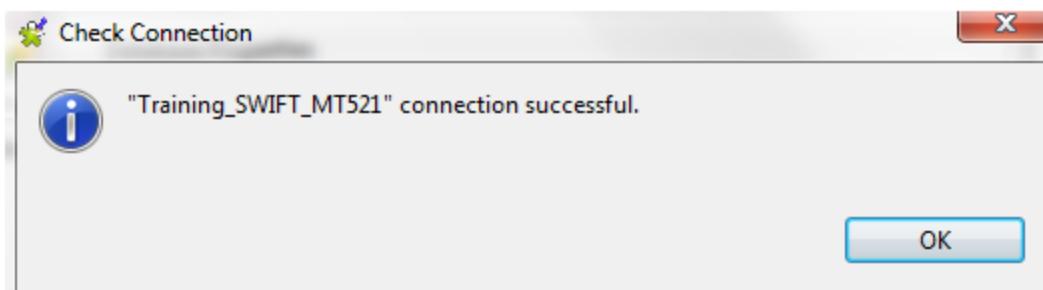
Use **root** for both the **Login** and the **Password**.

Enter **localhost** for the **Server** and **crm** for the **DataBase**.

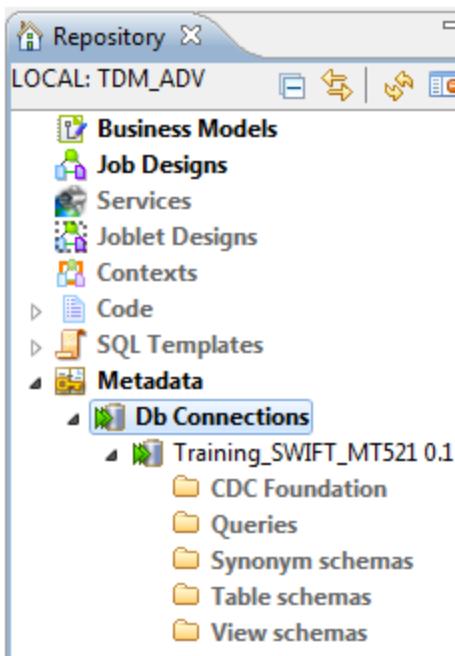
Finally, click the **Check** button to verify the connection.



4. Click **OK** in the dialog indicating success and click **Finish** to close the wizard.



5. Check that the MySQL connection appears under **DB Connections**.

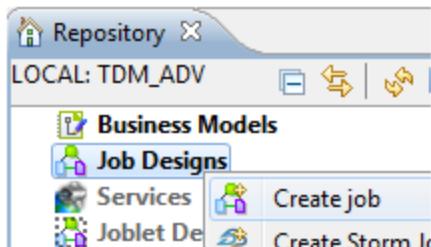


Next Step

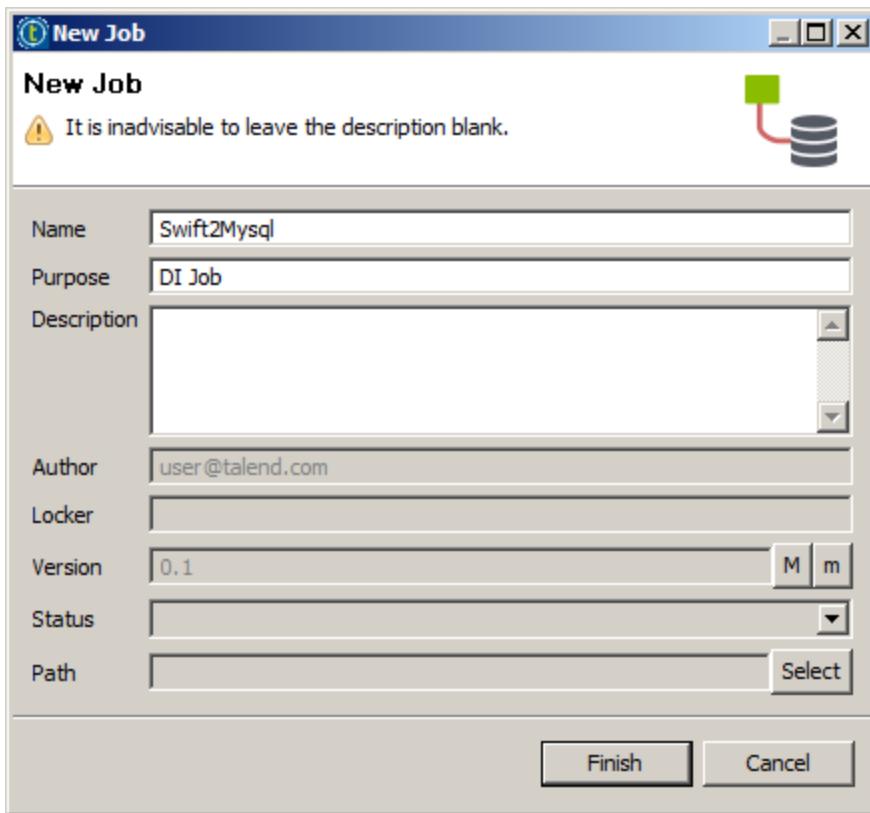
With the MySQL connection set up, let's [create a new DI Job](#).

Creating the DI Job

1. In the **Integration perspective**, right-click **Job Designs** and select **Create Standard Job**.

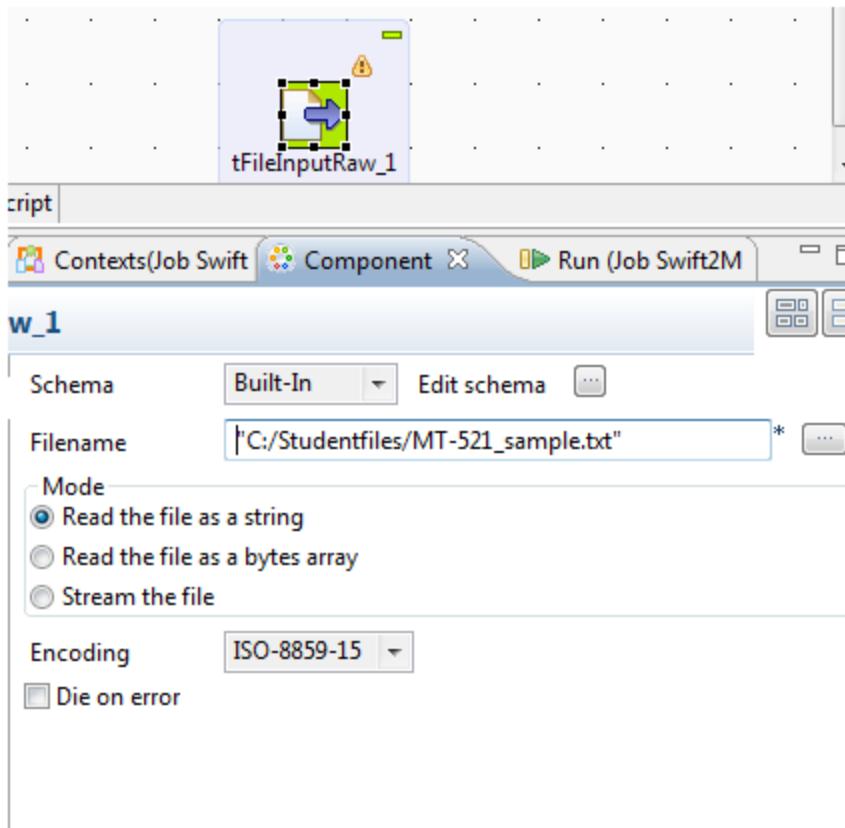


2. Provide details as illustrated below then click **Finish** to create the Job and open the Design area.

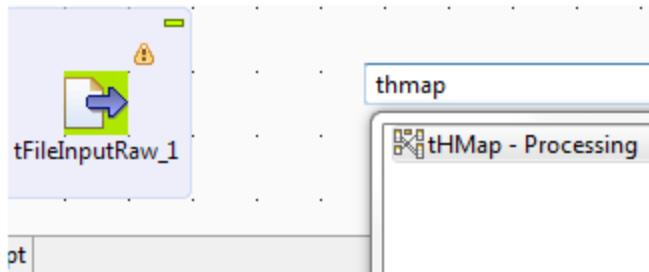


3. Drop a **tFileInputRaw** Component to the Design area and double-click it to display its **Component View**.

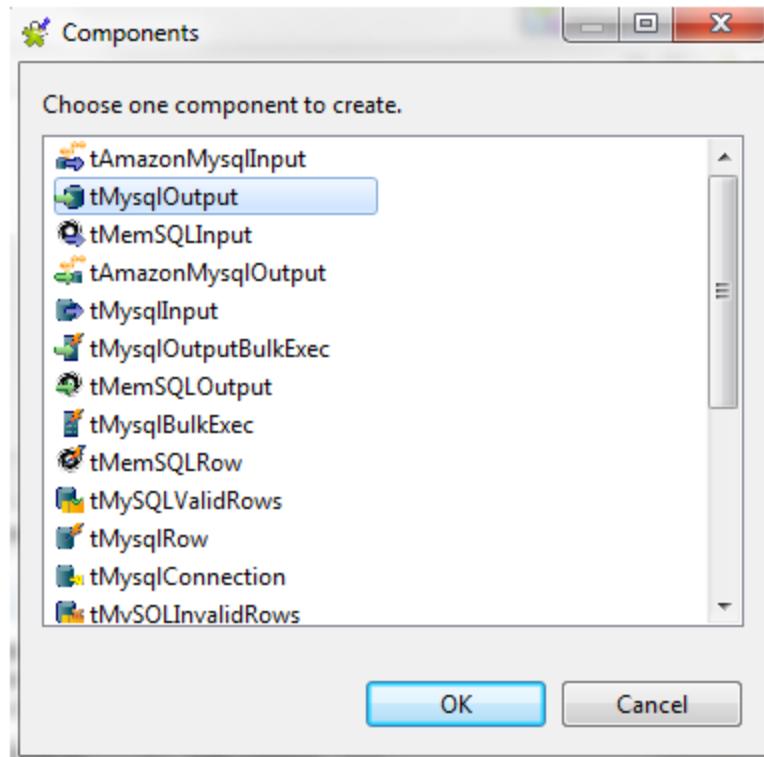
Click the ... button next to the **Filename** field and select `C:/StudentFiles/MT-521_sample.txt`.



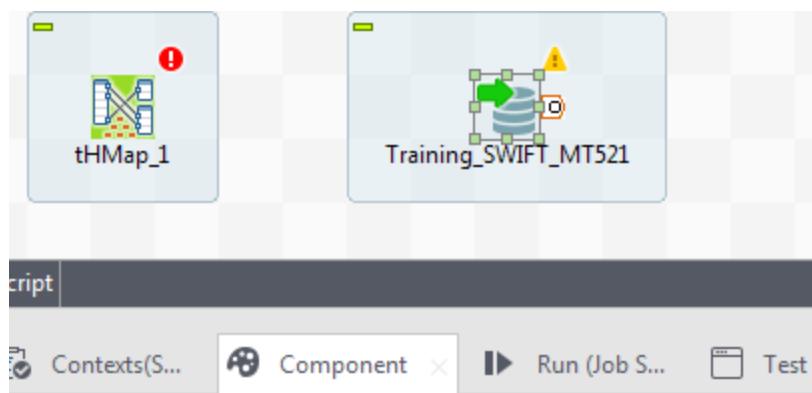
4. Add a **tHMap** Component to the Job.



5. Drag & drop **Metadata > Db Connections > Training_SWIFT_521** to the right side of **tHMap**, select a **tMysqlOutput** Component then click **OK**.



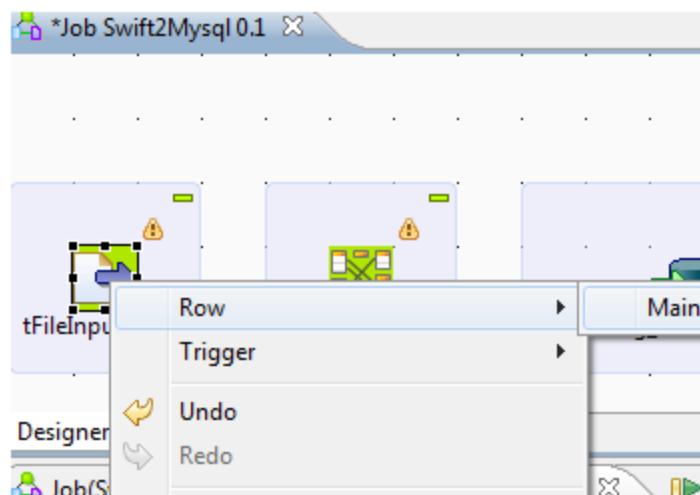
6. In the **Component** view, enter "*Training_SWIFT_MT521*" for the **Table** field and set **Action on table** to **Create table if it does not exist**.



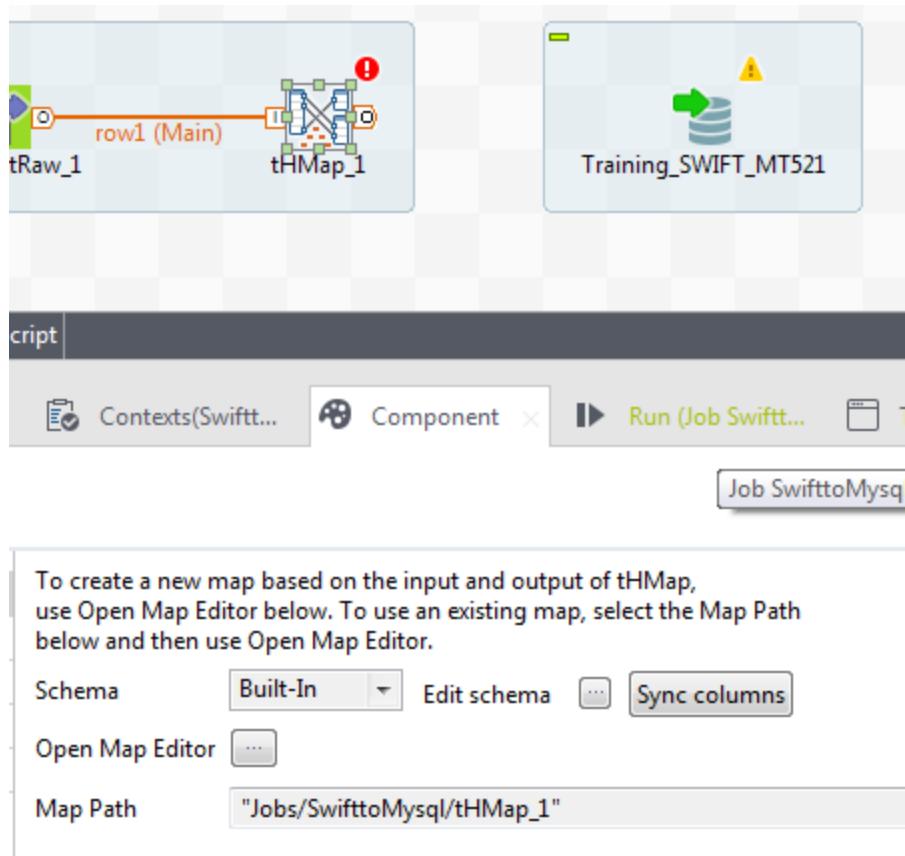
IFT_MT521(tMysqlOutput_1)

Property Type	Repository	DB (MYSQL):Training_SWIFT_MT521
DB Version	Mysql 5	
<input type="checkbox"/> Use an existing connection		
Host	"localhost"	* Port "3306"
Database	"crm"	
Username	"root"	* Password *****
Table	"Training_SWIFT_MT521"	
Action on table	Create table if does not exist	Action on data
Schema	Built-In	Edit schema ...

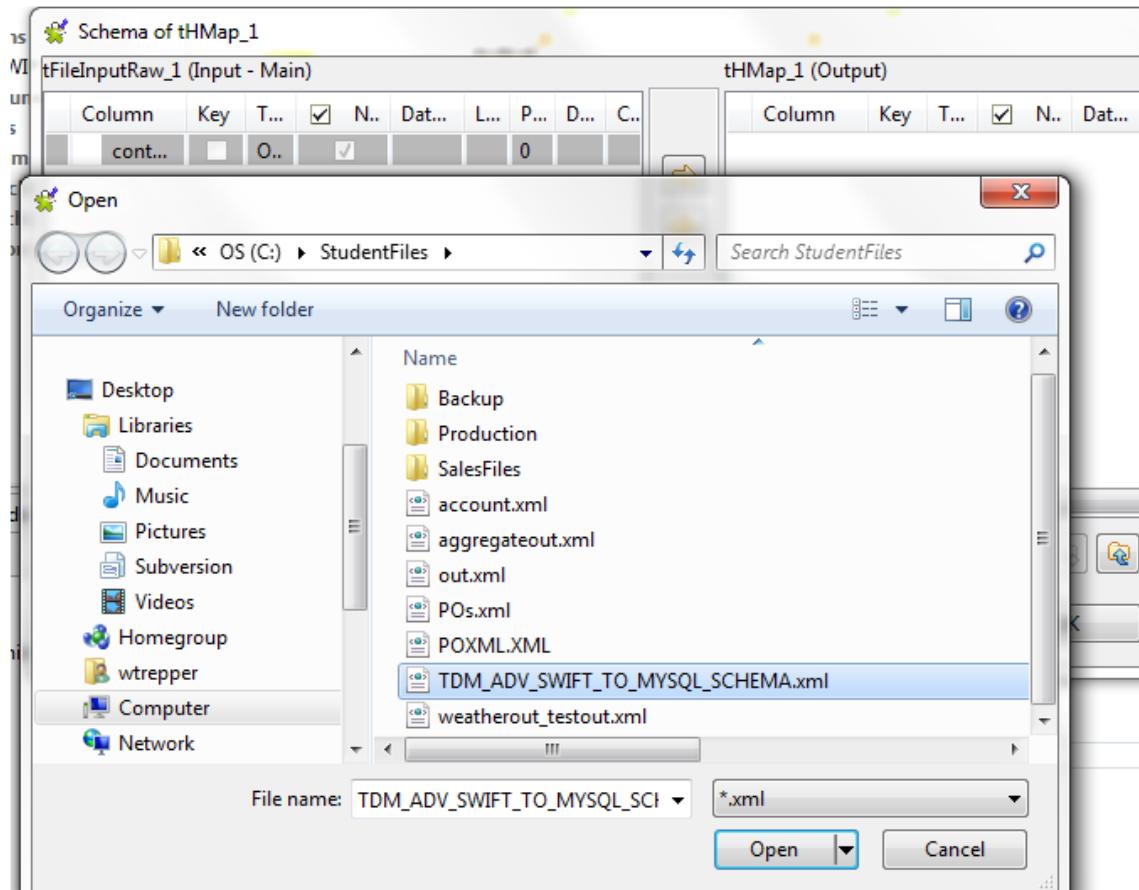
- Right-click **tFileInputRaw**, select **Row > Main** and connect it to **tHMap**.



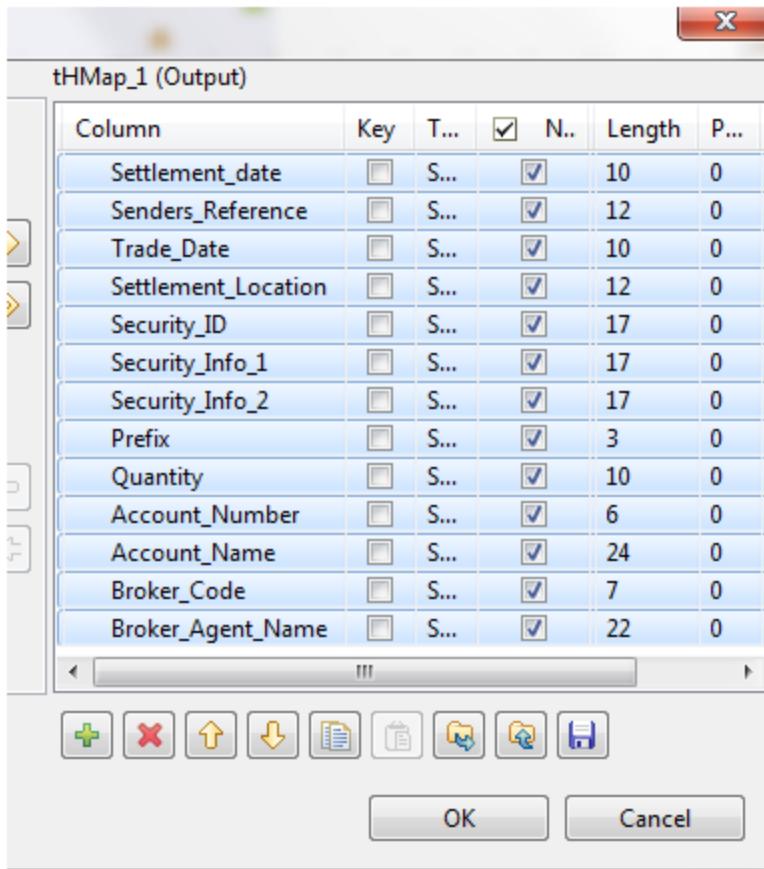
- Select the **tHMap** Component and click the ... button next to **Edit schema**.



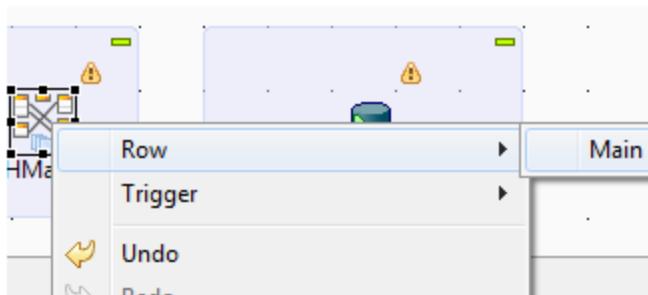
9. The output schema used to write in the MySQL database can be imported from an existing XML file. To do so, click the  button below the output table, select *C:/StudentFiles/TDM_ADV_SWIFT_TO_MYSQL_SCHEMA.xml* and click **Open**:



10. Check that the schema is actually copied to the **Output** then click **OK**.

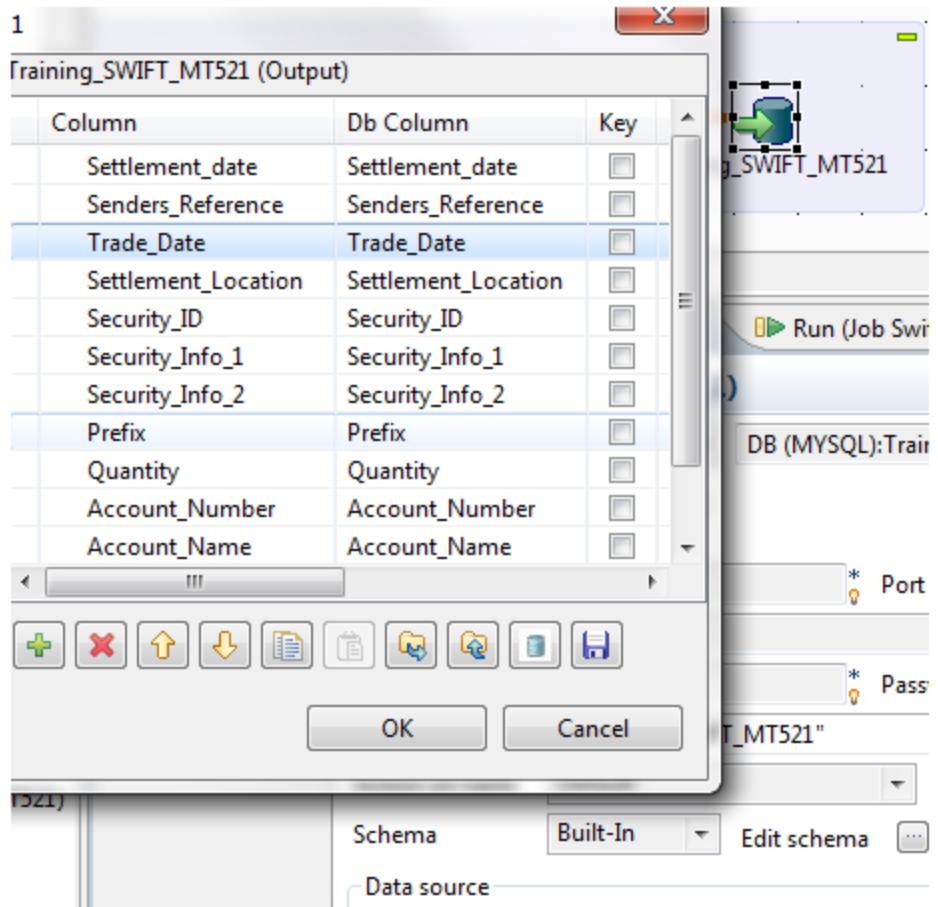


11. Right-click **tHMap**, select **Row > Main** and connect it to **Training_SWIFT_MT521**.



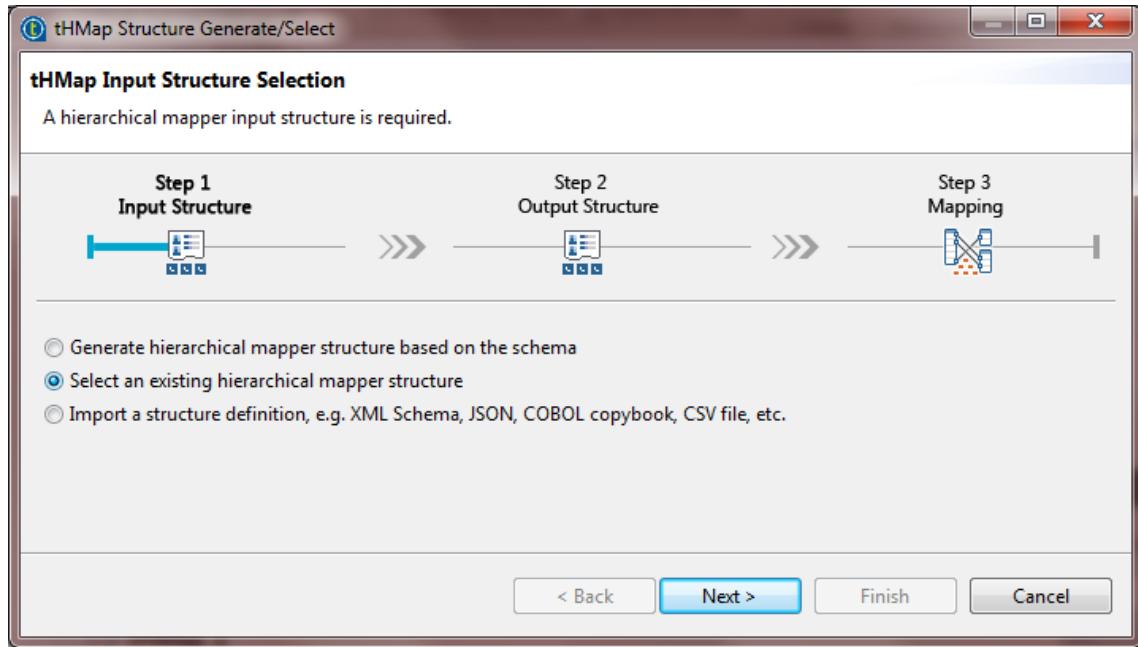
12. Select the **Training_SWIFT_MT521** Component then click the ... button next to **Edit schema**.

Check that the schema was copied from the **tHMap** Component, then click **OK**.

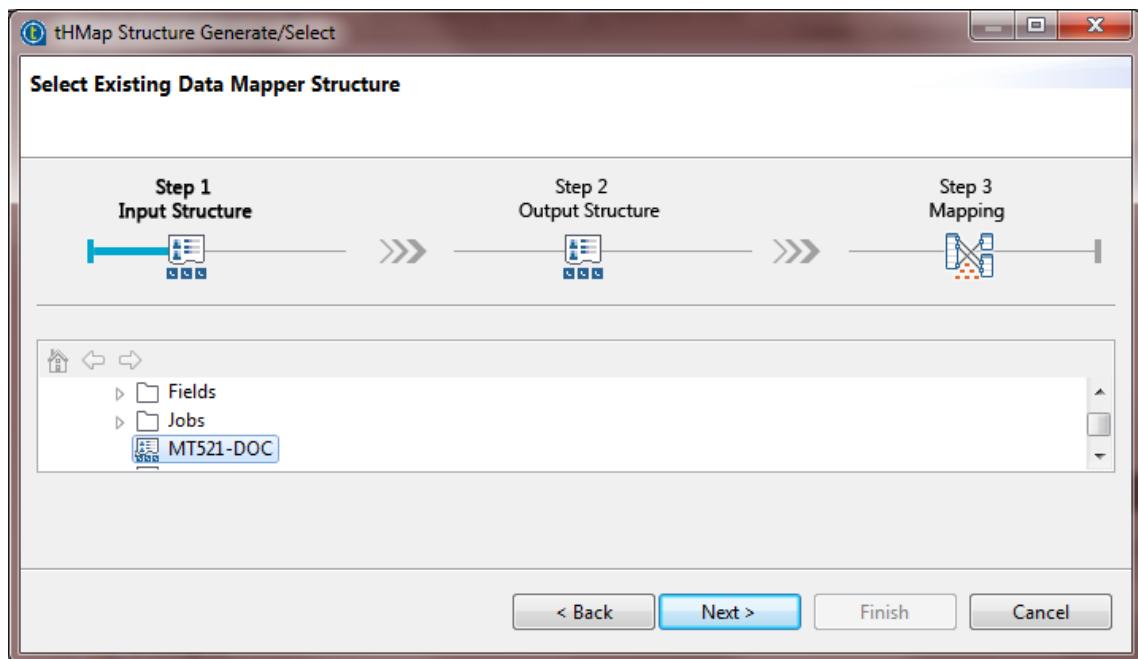


Creating the Mapping

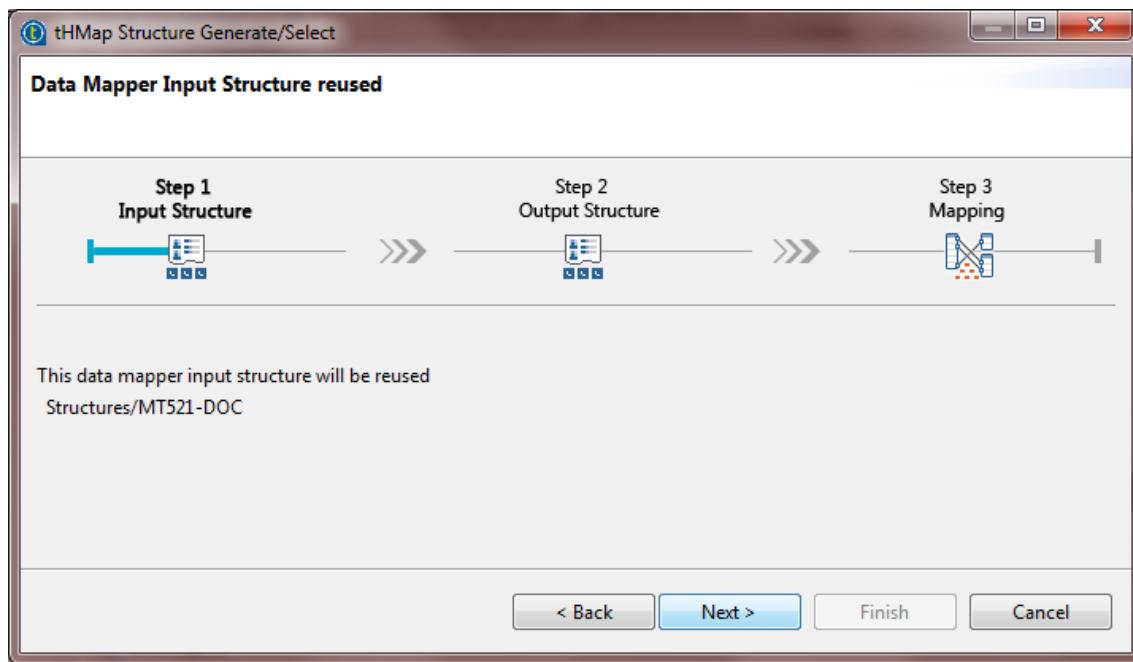
1. Double-click **tHMap** to start the mapping wizard. Pick the **Select an existing hierarchical mapper structure** option and click **Next >**.



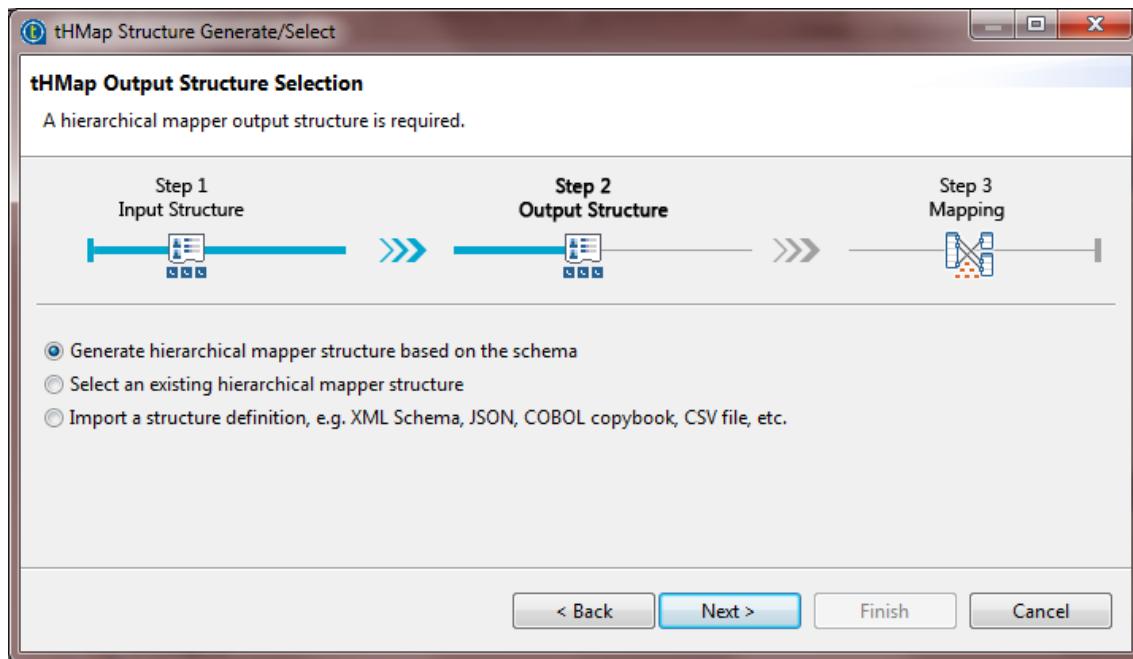
2. Select the **Structures > MT521-DOC** Structure then click **Next >**.



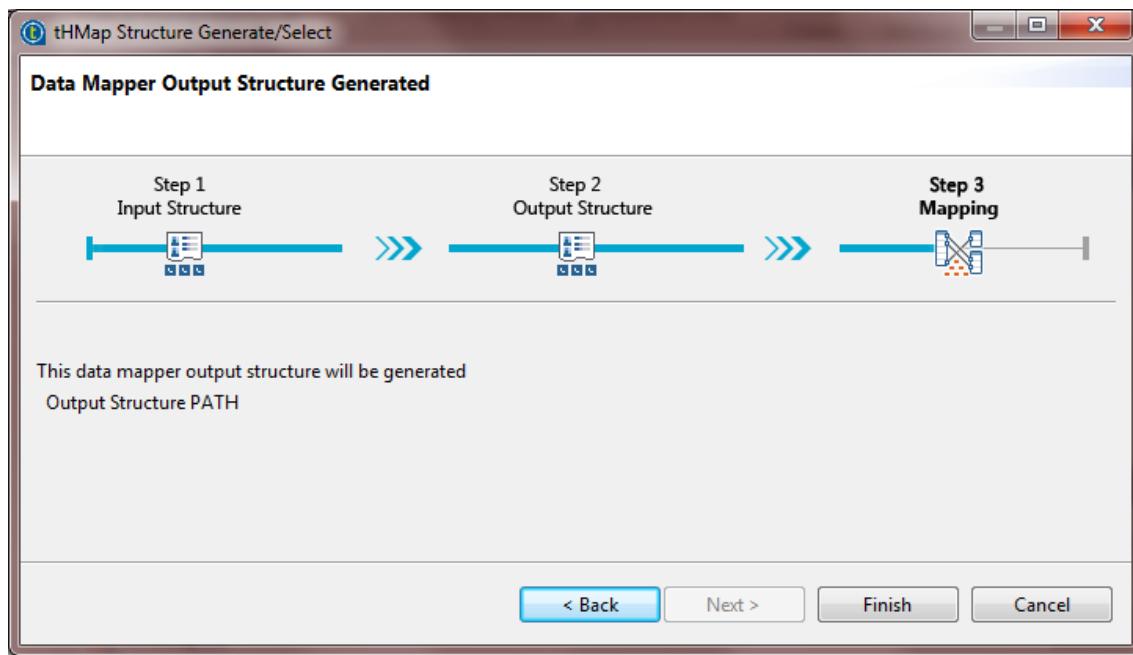
3. Click **Next >** in the confirmation screen.



4. Pick the **Generate hierarchical mapper structure based on the schema** option in Step 2 and click **Next >**.



5. Click **Finish** in the confirmation screen to open the Map Editor.

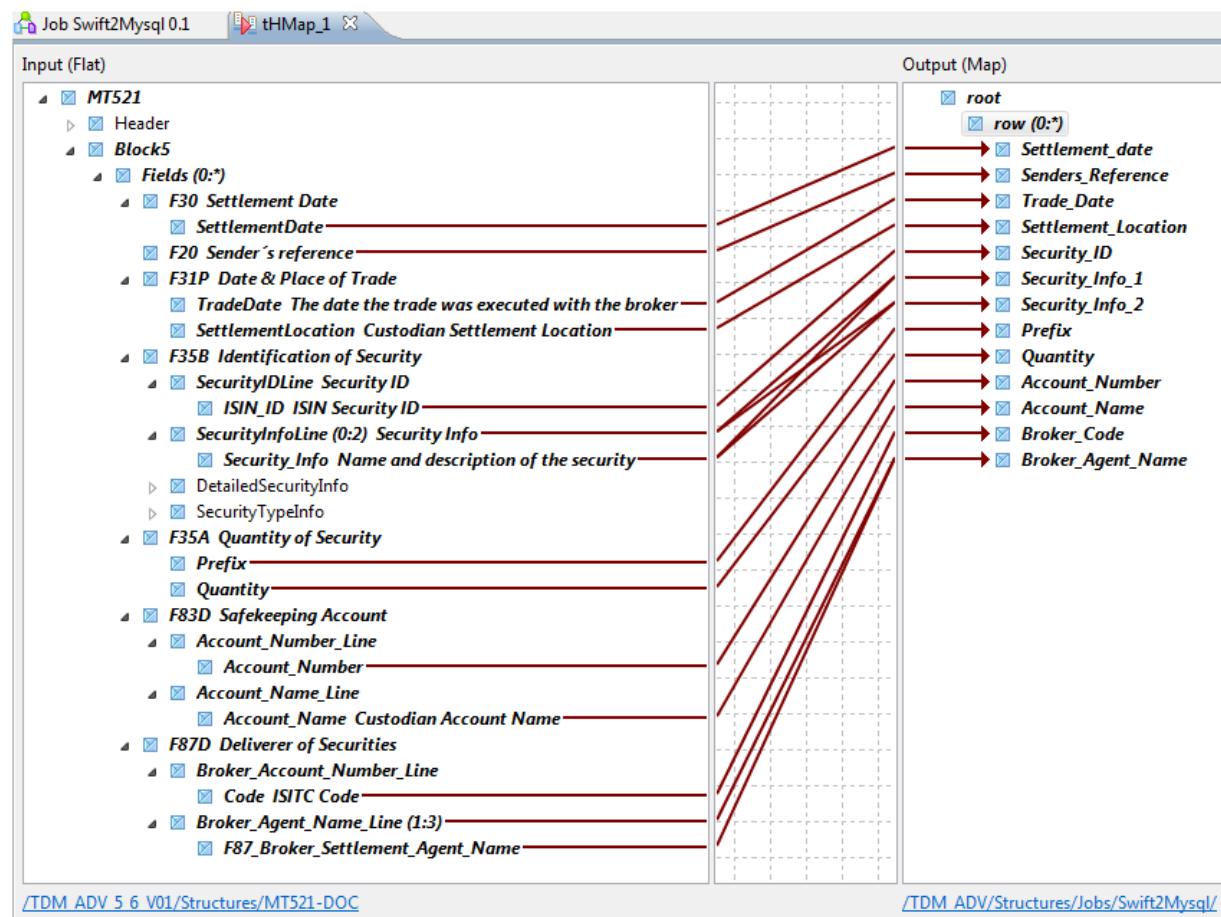


Next Step

Now that the input and output Structures are defined, let's [perform the actual mapping](#).

Performing the Mapping

The elements will be mapped like this:



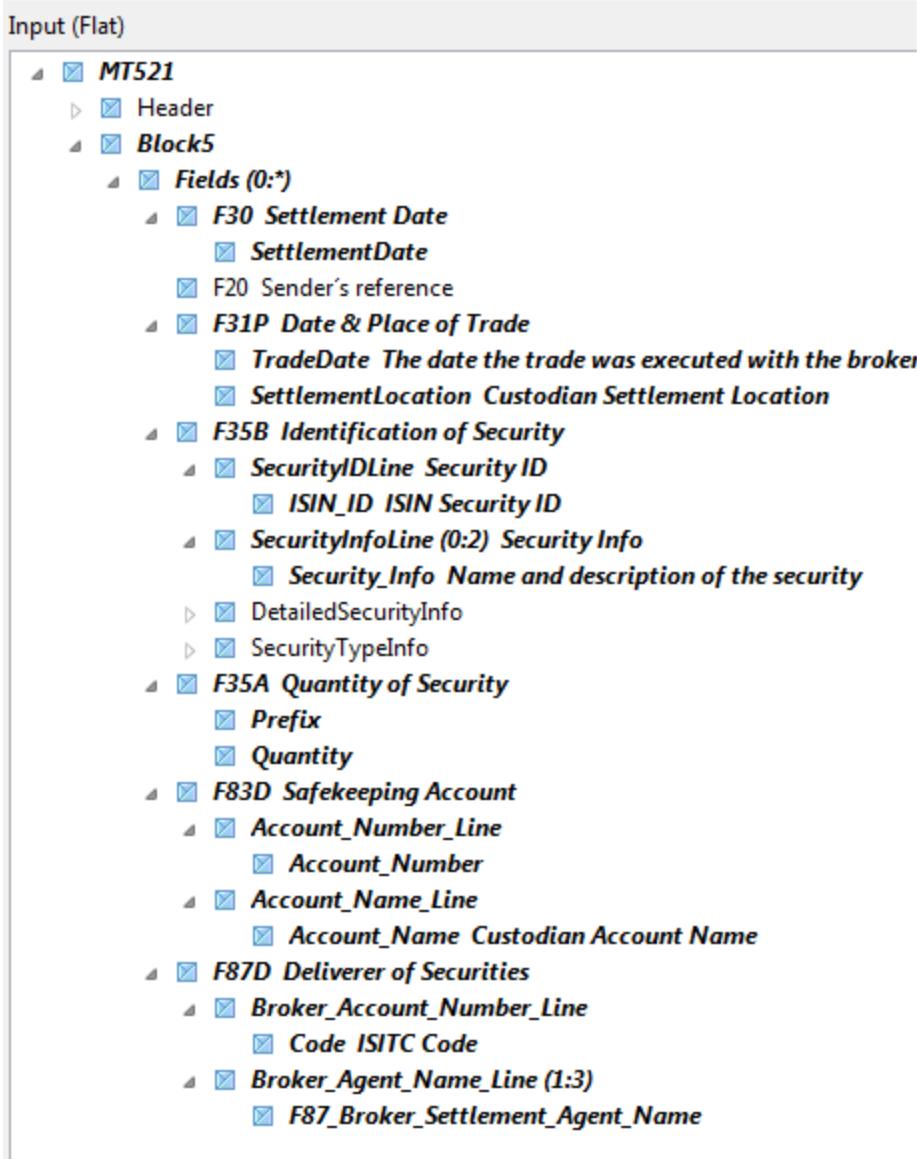
1. Switch to the **Mapping** perspective if it is not the current one already.

The screenshot shows the 'Mapping' perspective in a software interface. The top menu bar includes 'Run', 'Learn', 'Ask', 'Exchange', 'Mapping' (which is selected and highlighted in blue), and 'Component ...'. Below the menu is a toolbar with icons for saving, opening, and closing files. The main area is divided into two panels: 'Input (Flat)' on the left and 'Output (Map)' on the right.

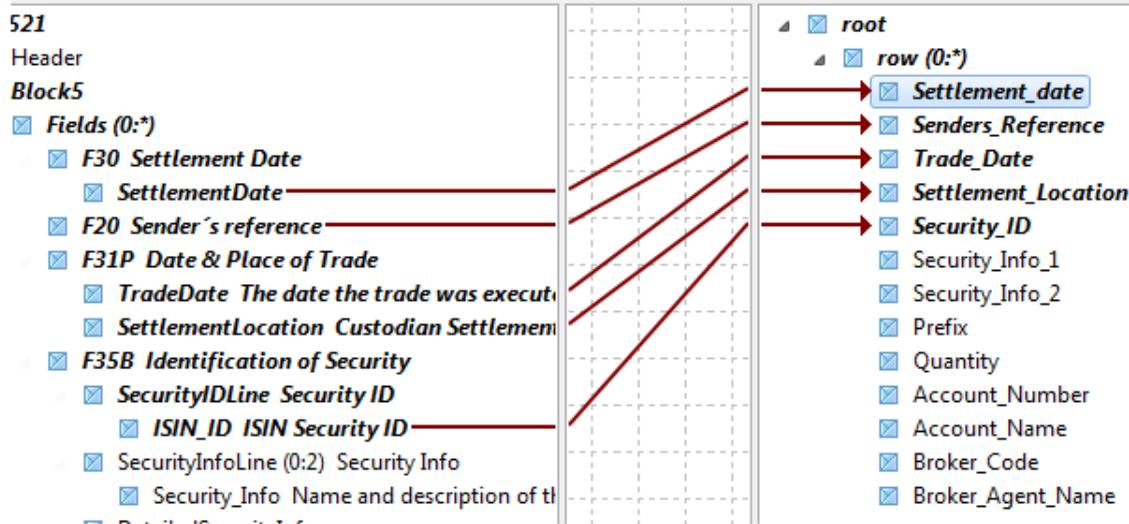
Input (Flat) Panel:

- MT521
 - Header
 - Block5
 - Fields (0:*)
 - F30 Settlement Date
 - F20 Sender's reference
 - F31P Date & Place of Trade
 - F35B Identification of Security
 - F35A Quantity of Security
 - F83D Safekeeping Account
 - F87D Deliverer of Securities

2. Expand **Block5** in the Input area.



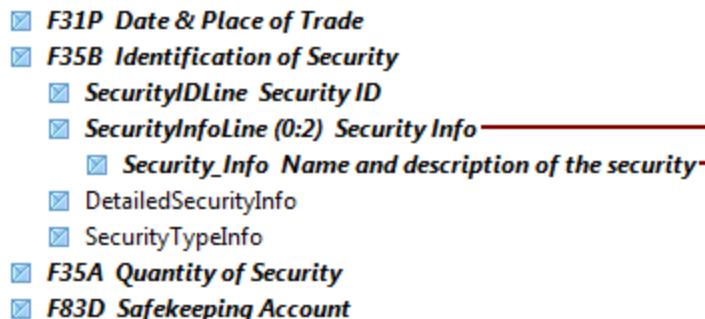
3. First, map **SettlementDate**, **Senders_Reference**, **Trade_Date**, **Settlement_Location** and **Security_ID** as illustrated below.



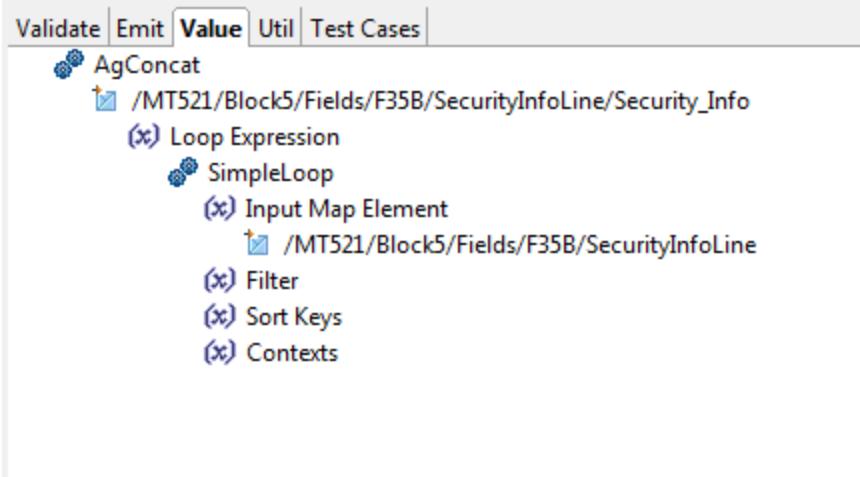
- Select **Security_Info_1** in the Output area and display its Value tab.

Drag & drop a Functions > Aggregate > AgConcat Function to the Value Tab.

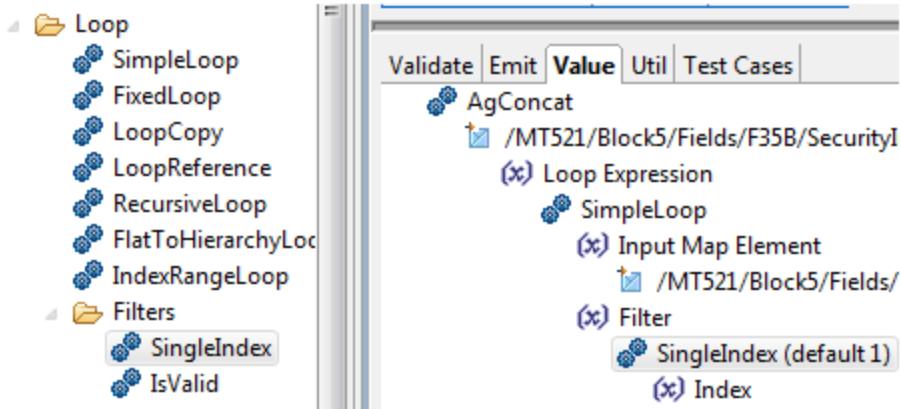
Finally, drag & drop **Security_Info Name and description of the security** from the Input area to AgConcat.



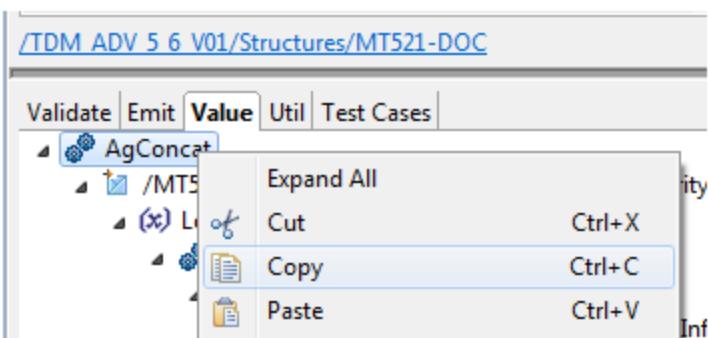
/TDM ADV 5 6 V01/Structures/MT521-DOC



- Drag & drop a Functions > Loop > Filters > SingleIndex Function to the empty Filter branch.



6. Right-click **AgConcat** and select **Copy**.

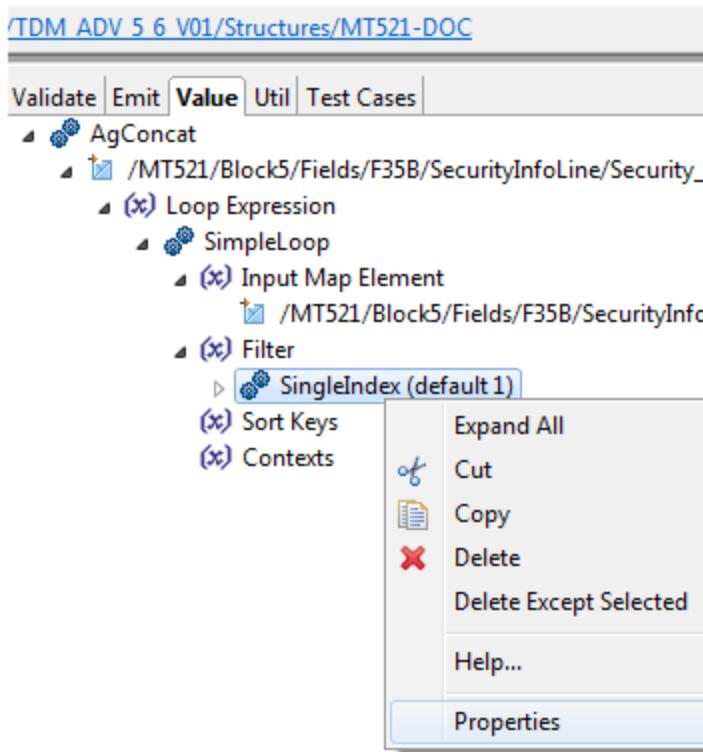


7. Select **Security_Info_2** in the Output area, right-click inside its **Value** tab and select **Paste** to insert a copy of the AgConcat Function.

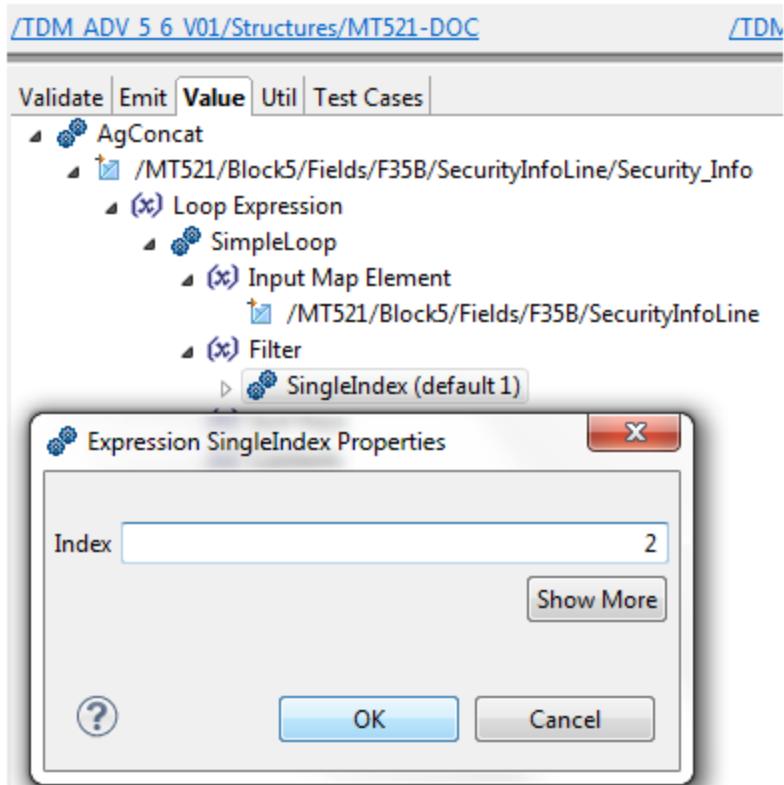
The screenshot shows the TDM ADV interface with the following components:

- Output Area:** Displays various fields such as **F2U Sender's reference**, **F31P Date & Place of Trade**, **TradeDate**, **SettlementLocation**, **F35B Identification of Security**, **SecurityIDLine Security**, and **ISIN_ID ISIN Security**.
- Value Tab:** For the node **Security_Info_2**. A context menu is open, with the **Paste** option highlighted. The menu also includes **Expand All**, **Cut**, and **Ctrl+C**.
- Paste Dialog:** A small dialog box is open at the bottom, containing the **Paste** button and the **Ctrl+V** keyboard shortcut.
- Right Panel:** Shows a list of available nodes or fields: **Security_ID**, **Security_Info_1**, **Security_Info_2**, **Prefix**, **Quantity**, **Account_Numb**, **Account_Name**, and **Broker_Code**.

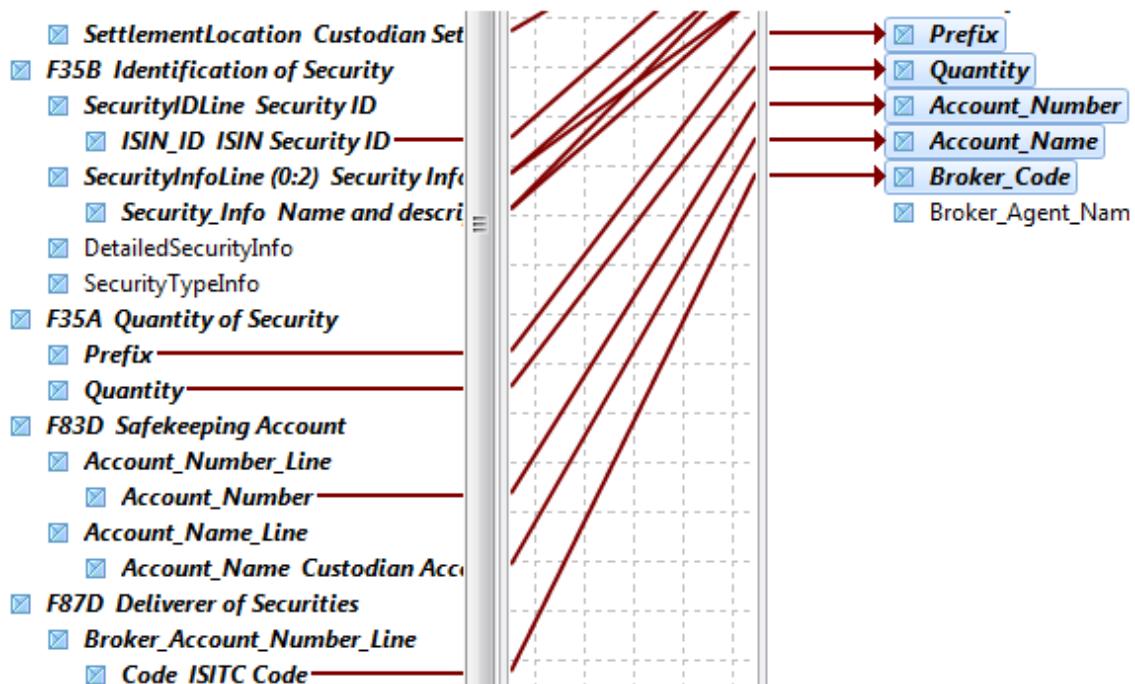
8. Expand the hierarchy and double-click **SingleIndex (default 1)** to display its properties.



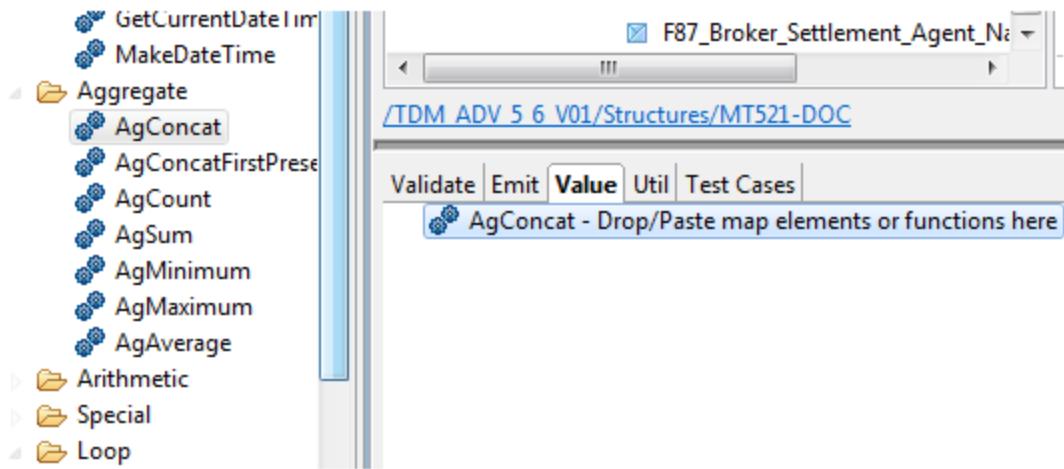
9. Set **Index** to 2 and click **OK**.



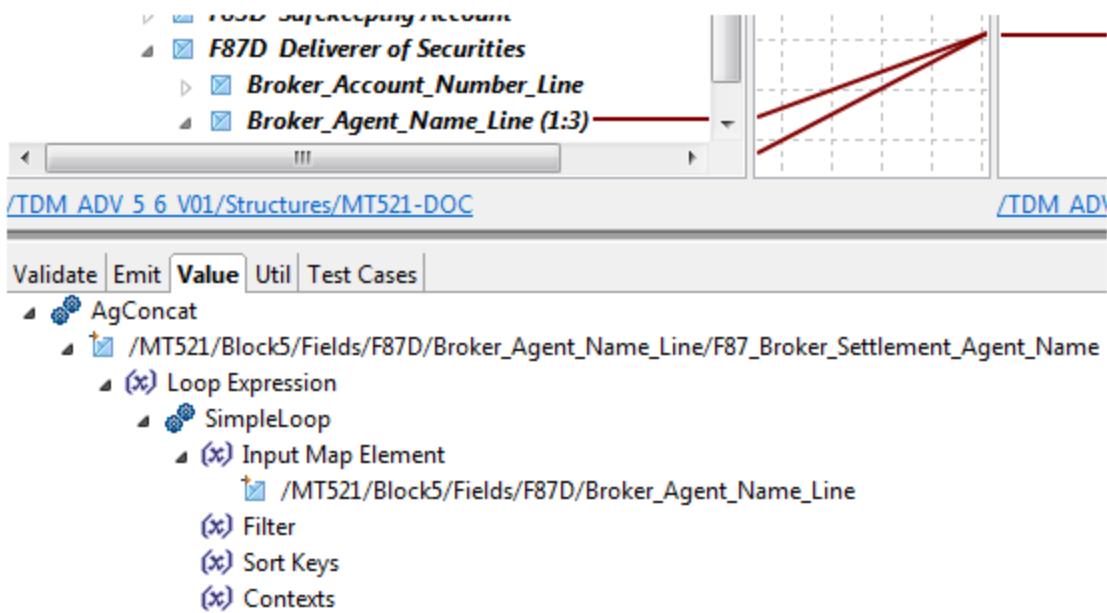
10. Map Prefix, Quantity, Account_Number, Account_Name and Broker_Code as illustrated below.



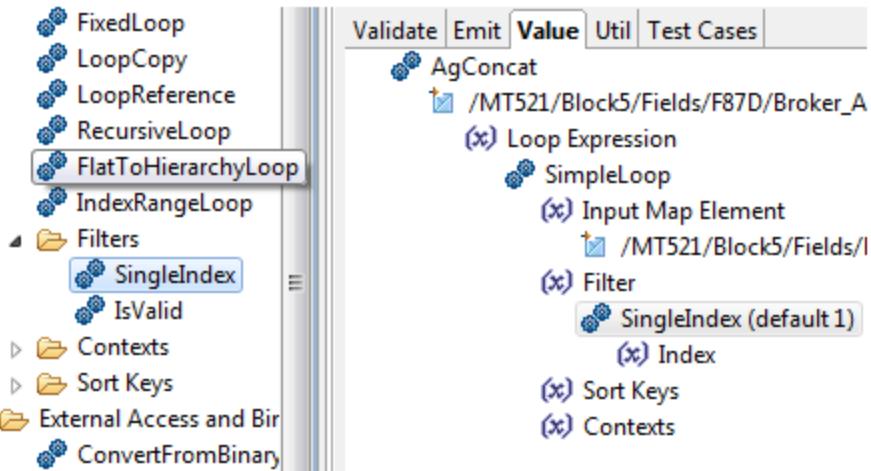
11. Finally, select Broker_Agent_Name in the Output area, display its Value tab and drag another Functions > Aggregate > AgConcat Function to it.



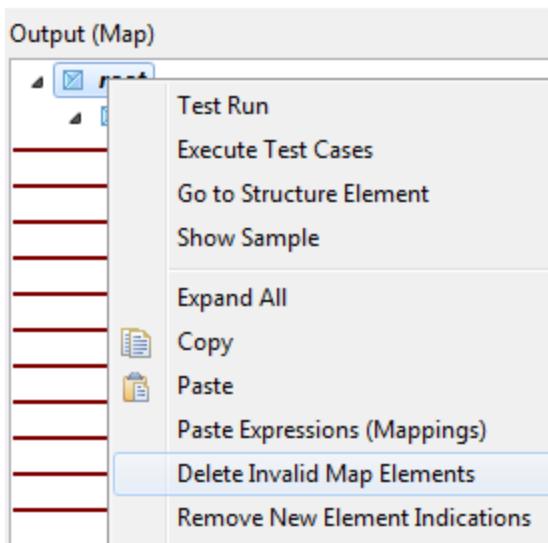
12. Drag & drop F87_Broker_settlement_Agent_Name from the Input area to the AgConcat function.



13. Drag & drop a Functions > Loop > Filter > SingleIndex Function to the Filter branch.



- Finally, right-click **root** in the **Output** area and select **Delete Invalid Map Elements**.

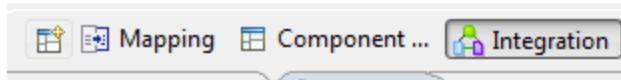


Next Step

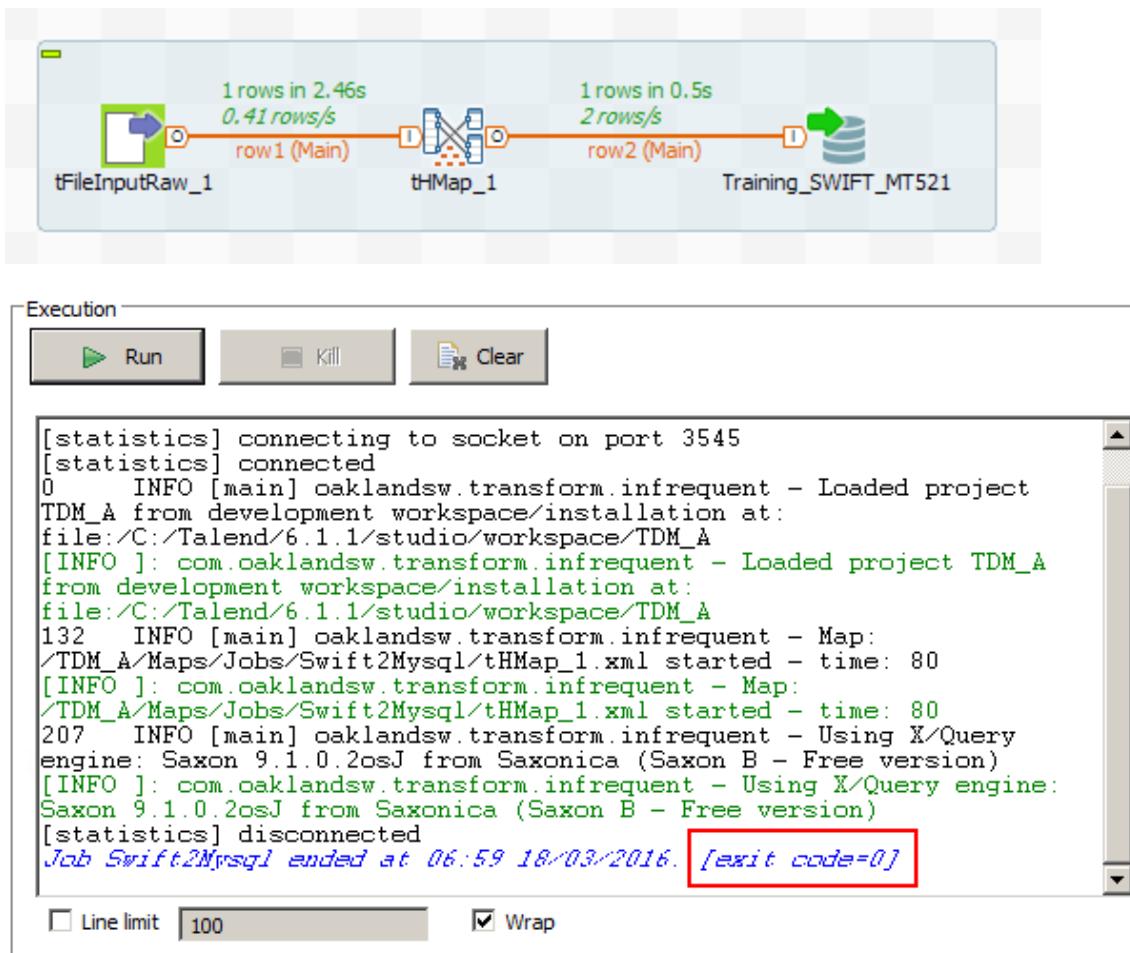
With the mapping performed, let's [run the DI Job](#) and check the results.

Running the Job

1. Switch back to the **Integration** perspective and the **Swift2Mysql** Job.

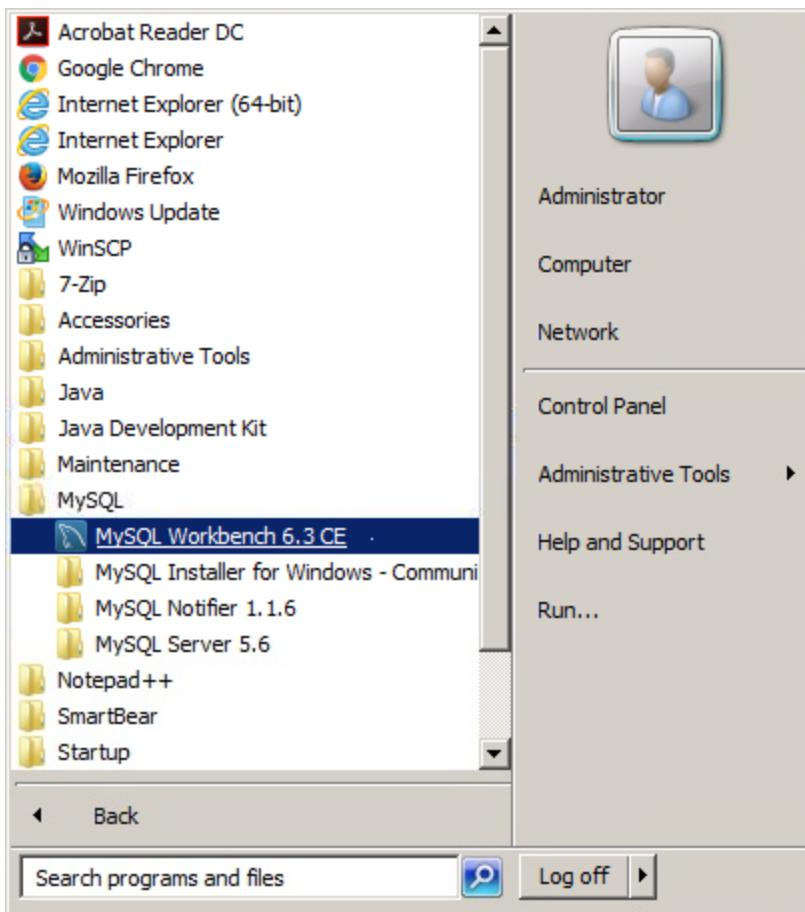


2. Run the job and check that it exits with a 0 code, indicating success.

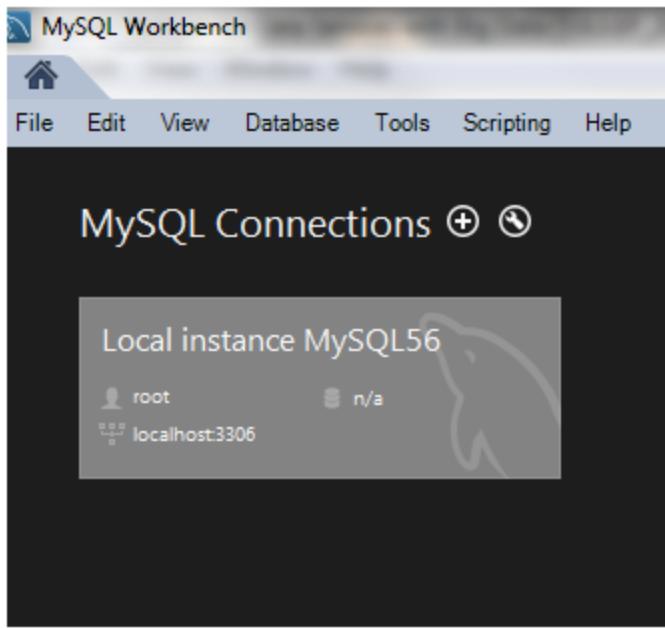


Checking the Results

1. Run All Programs > MySQL > MySQL Workbench 6.x CE from the Start Menu.



2. Double-click the */localhost* connection to open it.



3. Expand the **crm** branch on the left, right-click **crm > Tables > training_swift_mt521** and select **Select Rows - Limit 1000** to display the first thousand rows.

The screenshot shows the MySQL Workbench interface with the 'Performance Schema Setup' tab selected. On the left, the 'SCHEMAS' tree shows 'amc', 'cif', and 'crm'. Under 'crm', the 'Tables' branch is expanded, listing 'claim', 'clients', 'contract', 'contract_tp', 'cust', 'training_swift_mt521', and 'Views'. A context menu is open over the 'training_swift_mt521' table, with the 'Select Rows - Limit 1000' option highlighted.

4. There is only one record in the sample file, so one row should be added to the table every time the Job runs. Running the Job multiple times will only duplicate the first row.

The screenshot shows the 'Result Grid' interface in MySQL Workbench. The grid displays a single row of data from the 'training_swift_mt521' table. The columns are 'Settlement_date', 'Senders_Reference', 'Trade_Date', 'Settlement_Location', 'Security_ID', and 'Security_Info_1'. The data values are 140717, SE6439500032, 140717, TOKYO, JAPAN, ISIN JP1103101A95, and JGB NO.310(10YRS).

	Settlement_date	Senders_Reference	Trade_Date	Settlement_Location	Security_ID	Security_Info_1
▶	140717	SE6439500032	140717	TOKYO, JAPAN	ISIN JP1103101A95	JGB NO.310(10YRS)

Next Step

This lesson is almost over. Head to the [Wrap-Up](#) section for a summary of the concepts reviewed in this lesson.

Wrap-Up

In this lesson, you learned how to:

- » Set up a MySQL connection
- » Use a tHMap Component to read from a SWIFT file and write to a MySQL database

Next Step

Congratulations, you successfully completed this lesson. Click the **Check your status with this unit** button below in order to save your progress. Then click **Completed. Let's continue >** on the next screen to jump to the next lesson.

**This page intentionally left blank to ensure new chapters
start on right (odd number) pages.**

LESSON 3

Mapping Multiple Record Types

This chapter discusses the following.

Overview	80
Creating the Input Structure	82
Creating the Output Structure	94
Mapping the Elements	102
Creating the DI Job	115
Wrap-Up	121

Overview

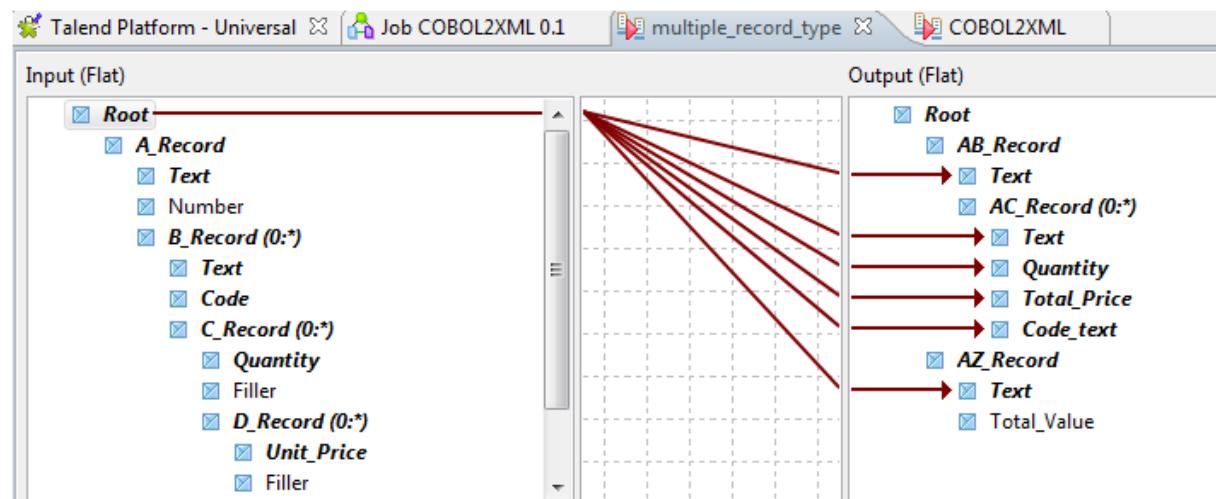
Lesson Overview

This exercise will guide you through the creation of a Map that converts a multiple record type flat file to another multiple record type flat file.

Here are the specifications you have been provided about the data conversion to be performed. The document contains all the configuration details, which elements to move, which elements to discard, which elements to compute...

A	B	C	D	E	F	G	H	I	J	K	L	M	N
Conversion Talend POC													
	Input file						Conversion			Output file			
	Header record									Header record			
	Label	Length	fix value							Label	Length	fix value	
	Type		1 "A"				Convert fix value "A" to "AB"			Type		1 "AB"	
	Text		10				Move without change			Text		10	
	Number		5				Don't move						
	Detail record (exists 2 times)									Detail record (exists 2 times)			
	Label	Length	fix value							Label	Length	fix value	
	Type		1 "B"				Convert fix value "B" to "AC"			Type		1 "AC"	
	Text		15				Move without change			Text		15	
	Code		2				Move from Quantity in sublevel "C"			Quantity		5	
							calculate Quantity in "C" x Unit price in "D"			Total price		5	
							Lookup text corresponding to "Code" in param file			Code text		20	
	Sublevel first record												
	Label	Length	fix value										
	Type		1 "C"										
	Quantity		5				Move without change to quantity in AC						
	Sublevel second record												
	Label	Length	fix value										
	Type		1 "D"										
	Unit price		7				Don't move						
	Footer record									Footer record			
	Label	Length	fix value							Label	Length	fix value	
	Type		1 "Z"				Convert fix value "Z" to "AZ"			Type		1 "AZ"	
	Text		10				Move without change			Text		10	
							Sum of "Total price" fields in output records "AC"			Total value		7	
Param file contents													
	Code	Text											
	01	Text 01											
	02	Text 02											
	03	Text 03											
	04	Text 04											

The final Map will look like the following, and will be called from a DI Job.



Objectives

After completing this lesson, you will be able to:

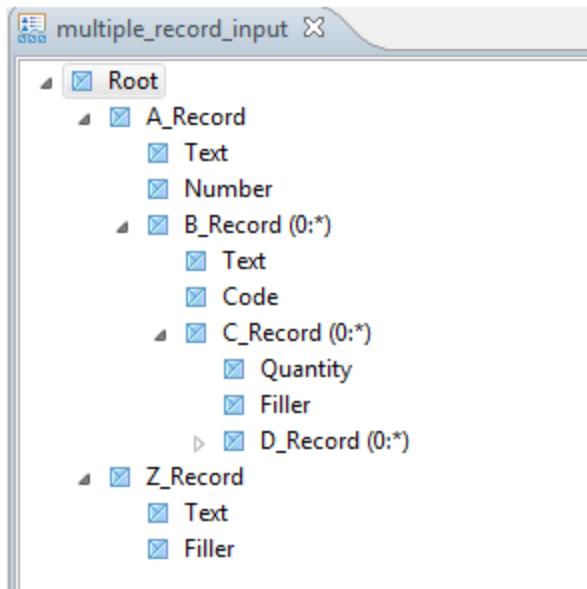
- » Convert a multiple record type Structure to another multiple record type Structure
- » Apply this transformation to flat files using a DI Job and a tHMap Component

Next Step

First, let's [create the input Structure](#) manually by adding and configuring elements.

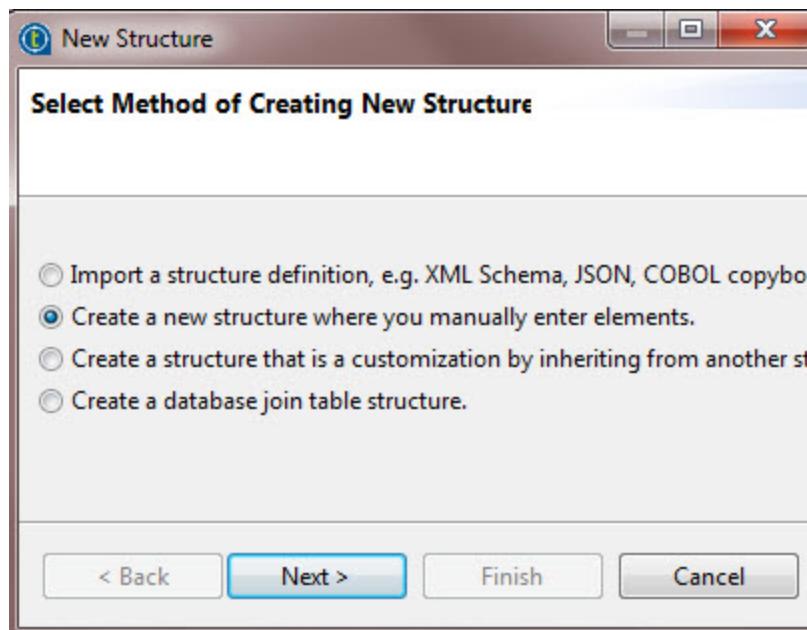
Creating the Input Structure

The objective of this section is to create the following input Structure.

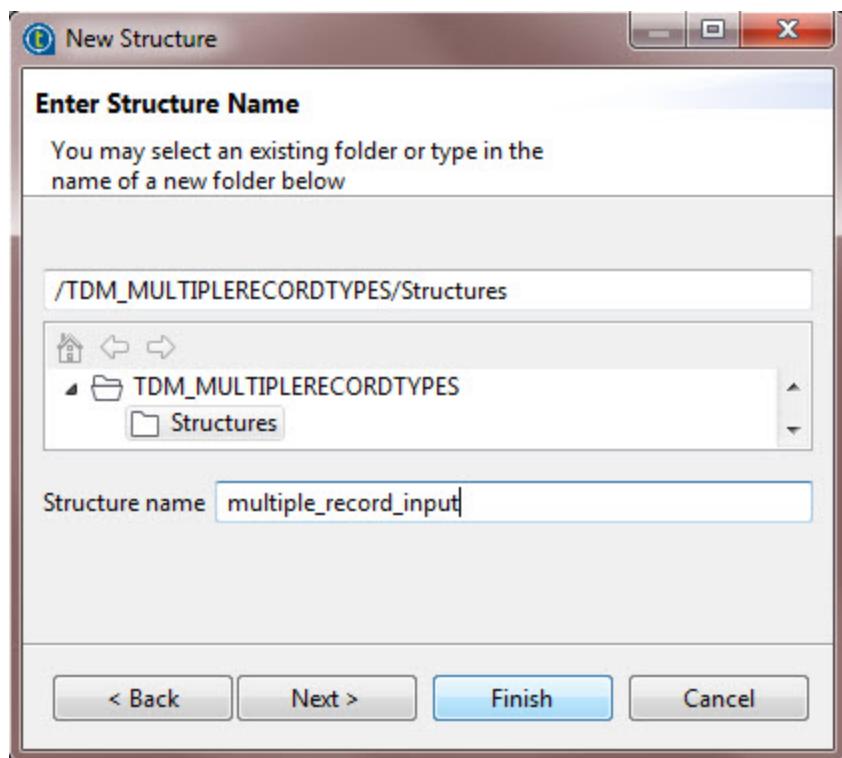


1. In the Mapping perspective, right-click **Hierarchical Mapper > Structures** and select **New > Structure**.

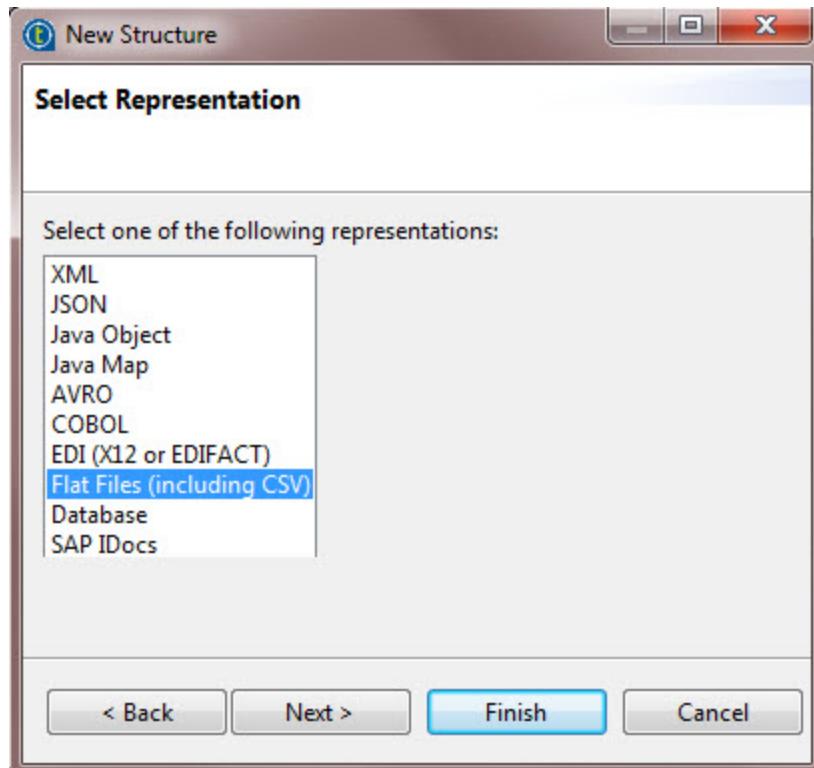
Pick the **Create a new structure where you manually enter elements**. option and click **Next >**:



2. Name the new Structure *multiple_record_input* and click **Next >**.



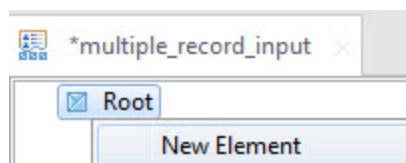
- Finally, select **Flat Files (including CSV)** for the representation and click **Finish**:



Creating the A_Record Element

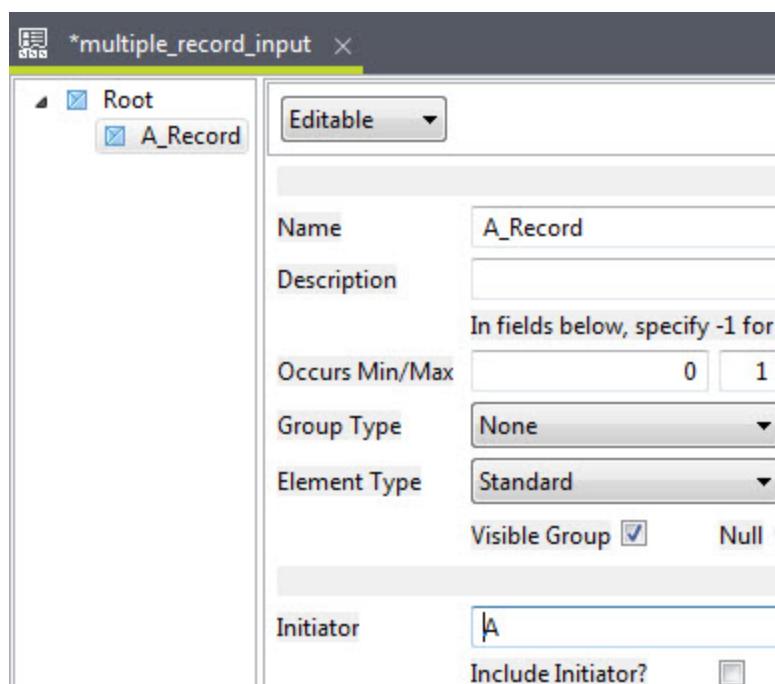
1. In the new Structure, right-click the empty area and select **New Element**.

Name it *Root*, then right-click it and select **New Element** again.



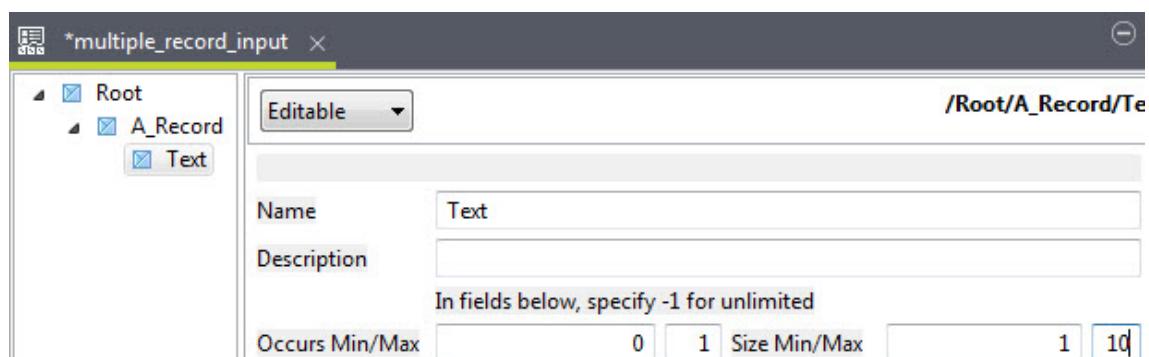
2. Name the new element **A_Record** and set the **Initiator** field to **A**.

Make sure the **Include Initiator** box is not checked.



3. Right-click **A_Record** and select **New Element** to create a new child element.

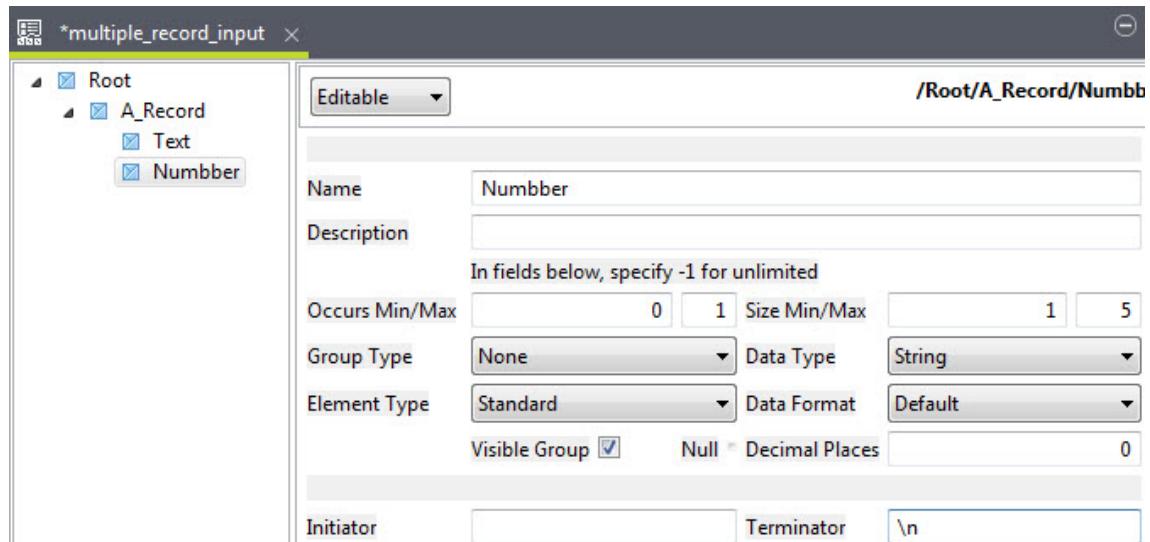
Name it *Text* and set the **Size Min/Max** fields to 1 (Min) and 10 (Max).



4. Right-click **A_Record** again and select **New Element** to create another child element.

Name it *Number* and set the **Size Min/Max** fields to 1 (Min) and 5 (Max).

Finally, enter `\n` in the **Terminator** field.

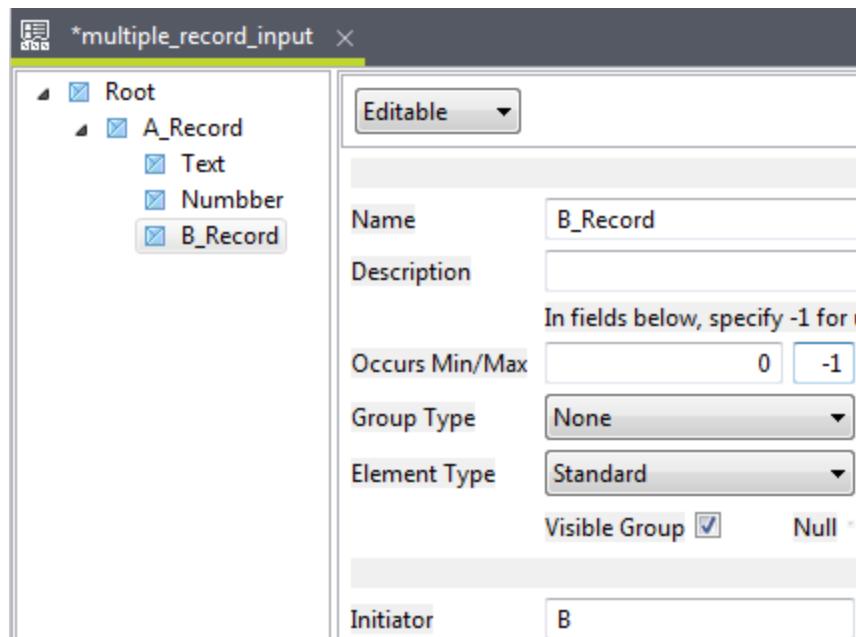


Creating the B_Record Element

1. Right-click **A_Record** and select **New Element**.

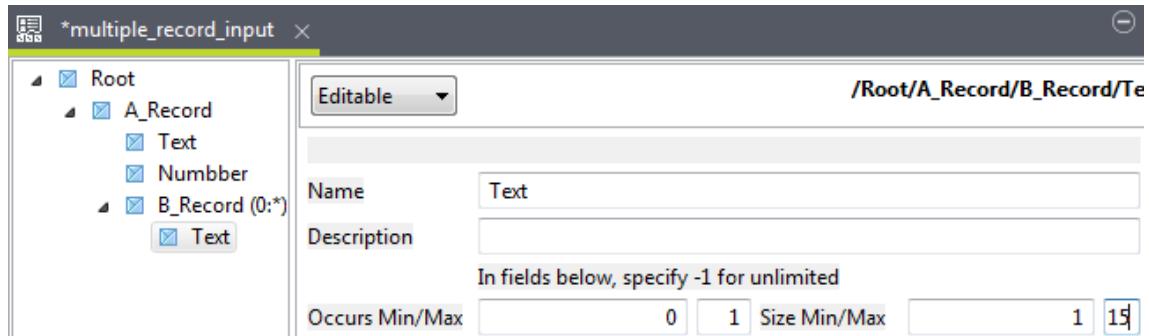
Name it **B_Record** and set the **Initiator** field to **B**.

Finally, set the **Occurs Min/Max** fields to 0 (Min) and -1 (Max) make it a looping element.



2. Right-click **B_Record** and select **New Element** to create a child element.

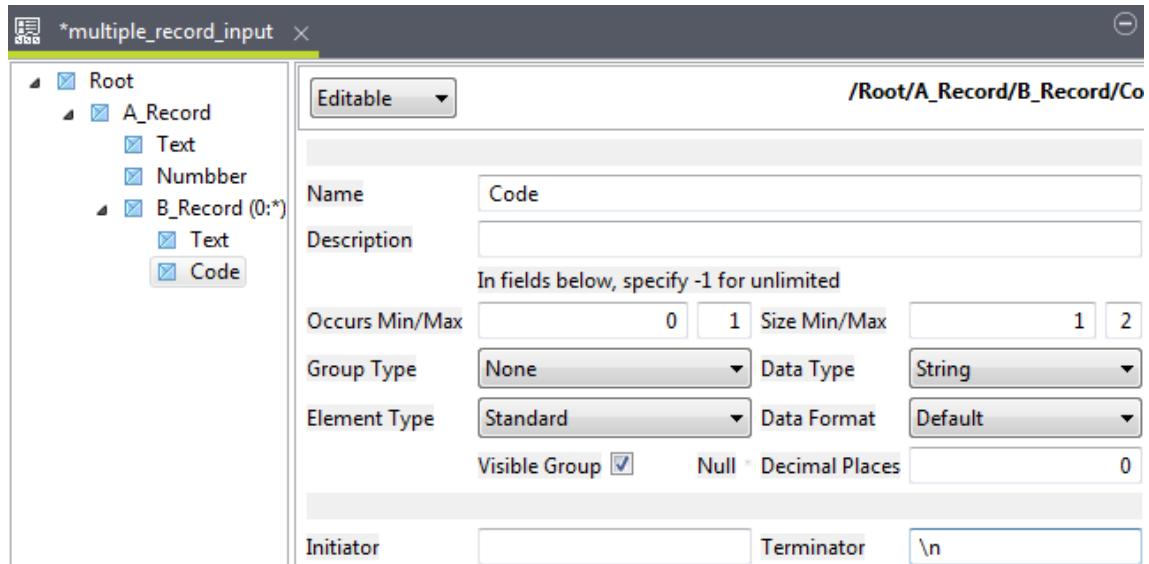
Name it **Text** and set the **Size Min/Max** fields to 1 (Min) and 15 (Max).



- Right-click **B_Record** again and select **New Element** to create another child element.

Name it *Code* and set **Size Min/Max** fields to 1 (Min) and 2 (Max).

Finally, set **Data Type** to *Integer (32)* and set **Terminator** to *\n*.

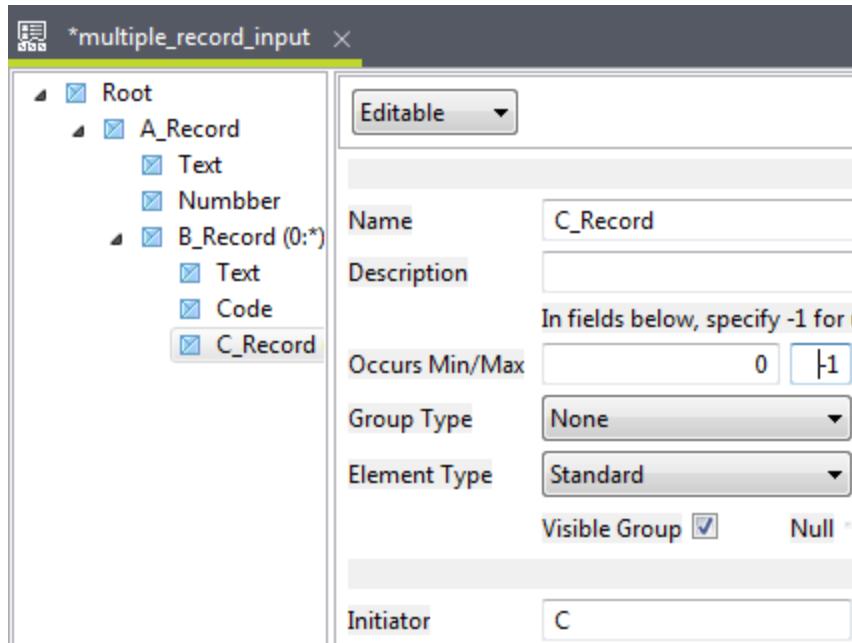


Creating the C_Record Element

- Right-click **B_Record** and select **New Element**.

Name it *C_Record* and set the **Initiator** field to *C*.

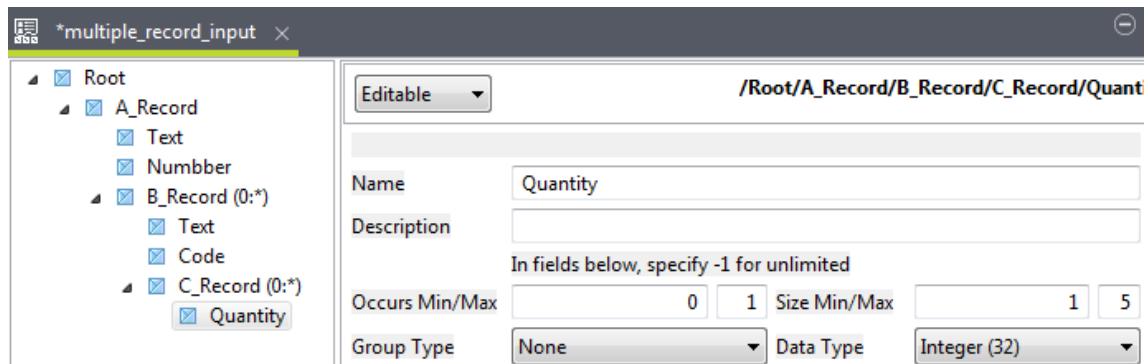
Finally, set the **Occurs Min/Max** fields to 0 (Min) and -1 (Max) to make it a looping element.



- Right-click **C_Record** and select **New Element**.

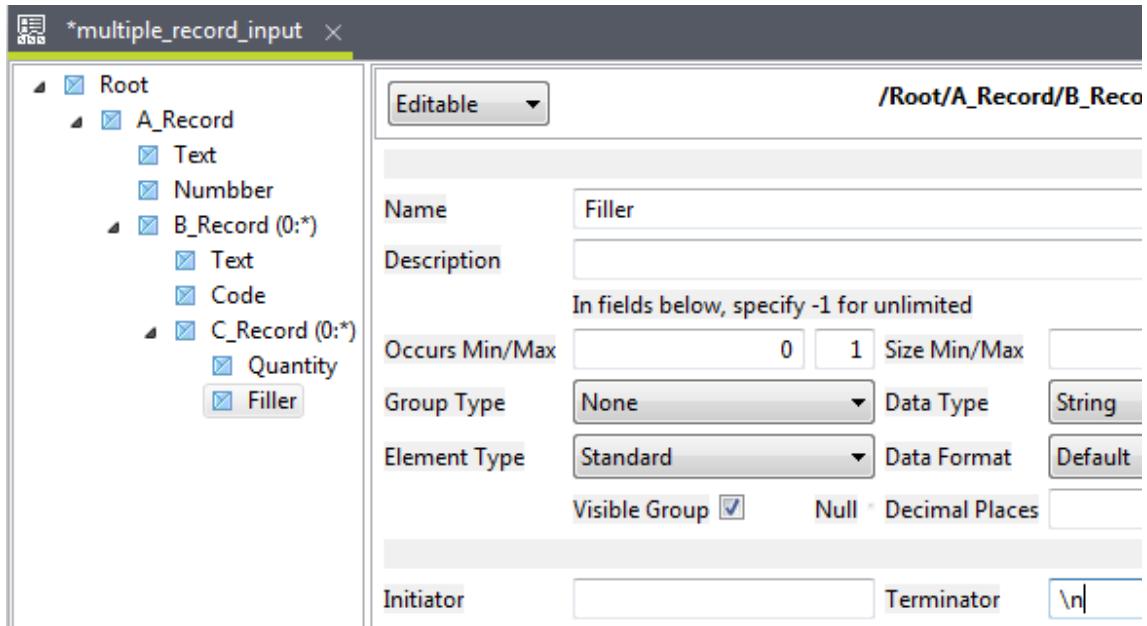
Name it **Quantity** and set the **Size Min/Max** fields to 1 (Min) and 5 (Max).

Finally, set **Data Type** to **Integer (32)**.



- Right-click **C_Record** and select **New Element**.

Name it **Filler** and set the **Terminator** field to **\n**.

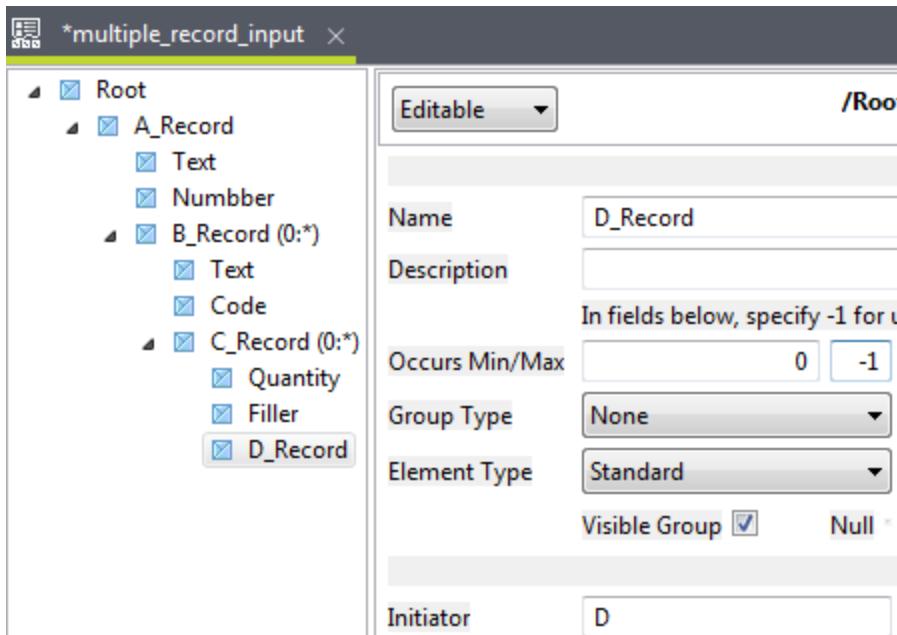


Creating the D_Record Element

1. Right-click **C_Record** and select **New Element**.

Name it **D_Record** and set the **Initiator** field to **D**.

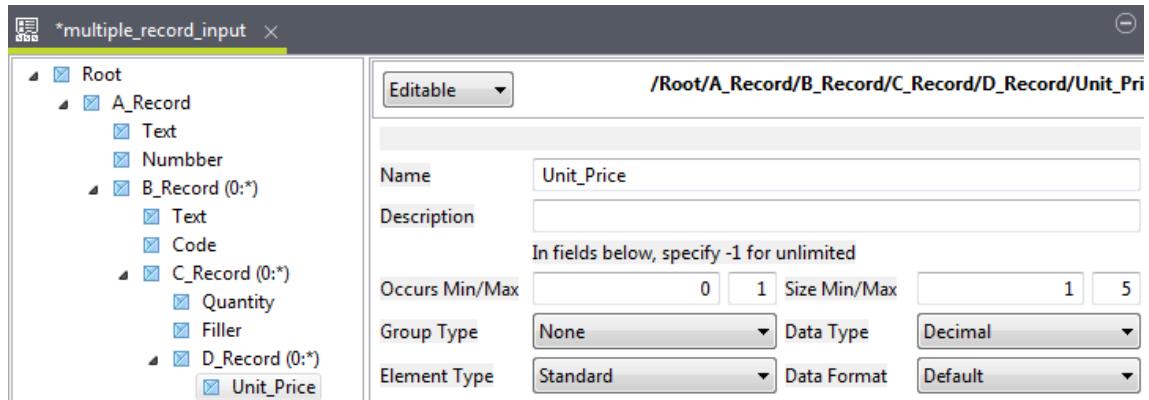
Finally, set the **Occurs Min/Max** fields to **0** (Min) and **-1** (Max) to make it a looping element.



2. Right-click **D_Record** and select **New Element**.

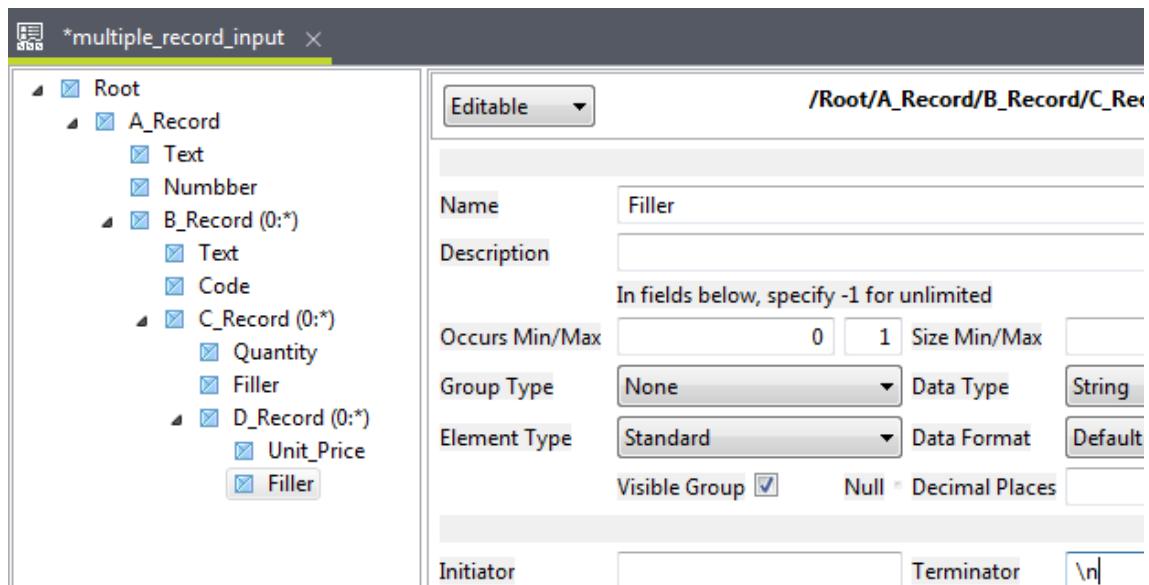
Name it **Unit_Price** and set the **Size Min/Max** fields to **1** (Min) and **5** (Max).

Finally, set the **Data Type** field to **Decimal**.



3. Right-click **D_Record** and select **New Element**.

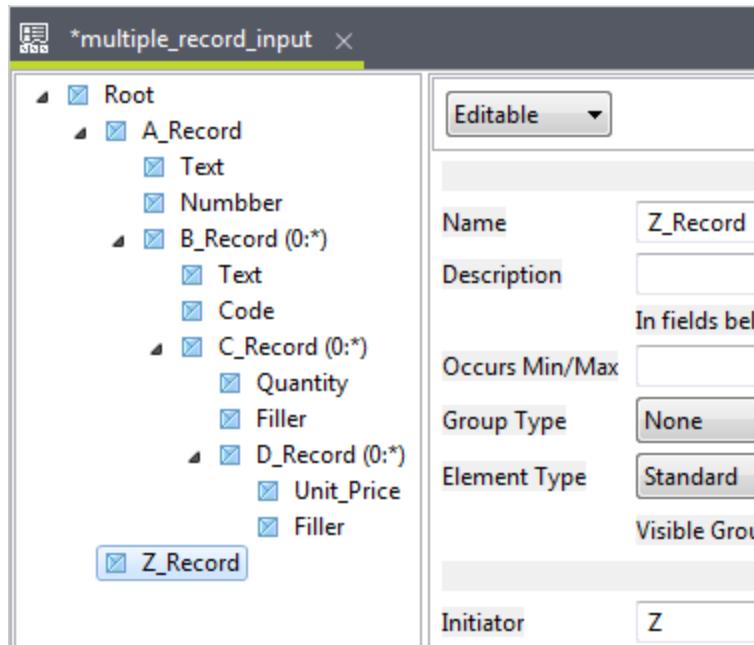
Name it **Filler** and set the **Terminator** field to **\n**.



Creating the Z_Record Record

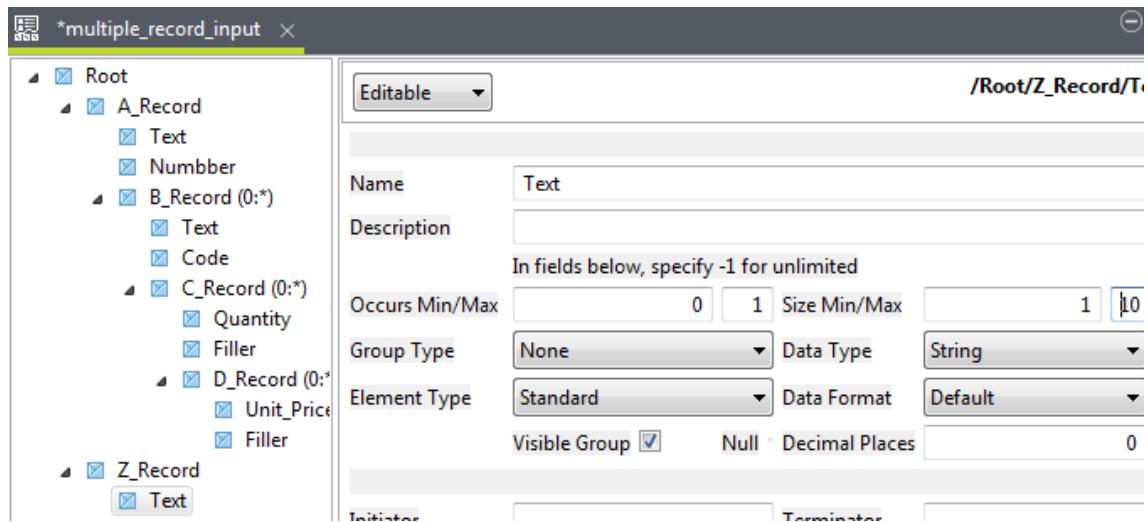
1. Right-click **Root** and select **New Element**.

Name it **Z_Record** and set the **Initiator** field to **Z**.



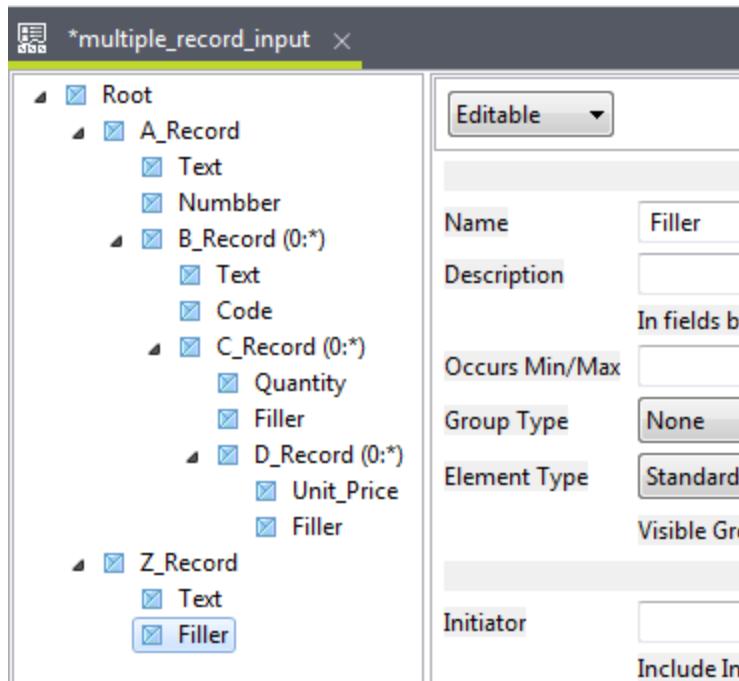
2. Right-click **Z_Record** and select **New Element**.

Name it **Text** and set the **Size Min/Max** fields to 1 (Min) and 10 (Max).



3. Right-click **Z_Record** and select **New Element**.

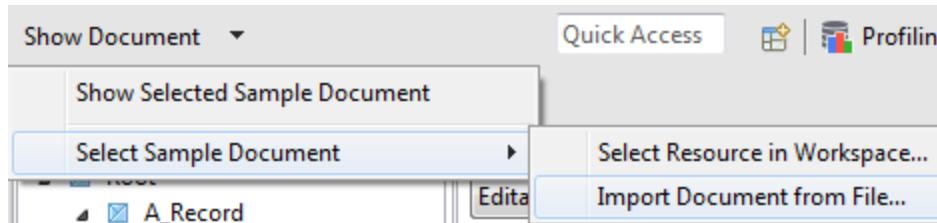
Name it **Filler**.



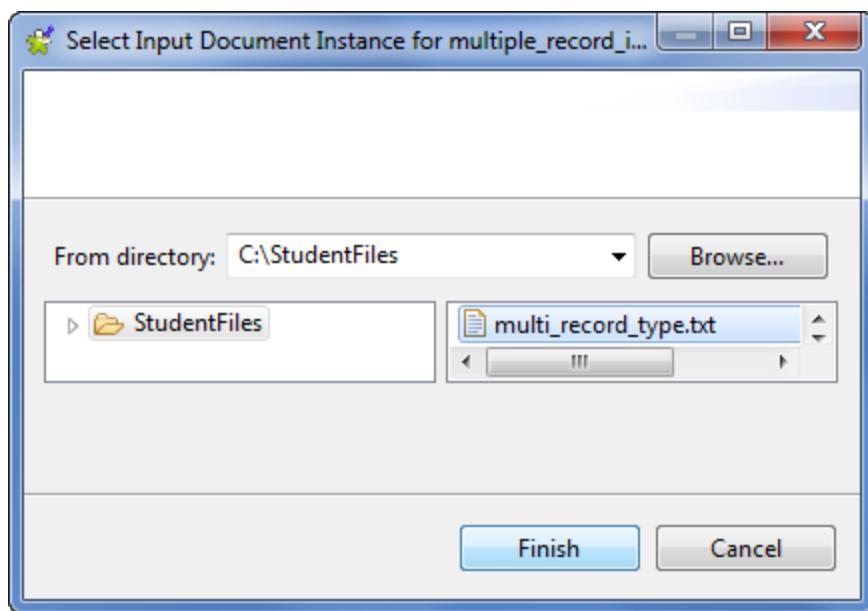
- Save the changes to the input Structure.

Displaying a Sample Document

- Expand the **Show Document** menu and go to **Select Sample Document > Import Document from File....**



- Click **Browse...** to select the *C:\StudentFiles* folder, then select *multiple_record_type.txt* and click **Finish**.



3. Select several elements from the input Structure and check that the right data is highlighted in the Document tab.

*multiple_record_input

The screenshot shows a software interface for defining an input structure. On the left is a tree view of the record hierarchy:

- Root
 - A_Record
 - Text
 - Number
 - B_Record (0:*)
 - Text
 - Code
 - C_Record (0:*)
 - Quantity
 - Filler
 - D_Record (0:*)
 - Unit_Price
 - Filler
 - Z_Record
 - Text
 - Filler

On the right, there are several property panels:

- Editable**: A dropdown menu.
- Name**: Text (selected)
- Description**: In field
- Occurs Min/Max**: []
- Group Type**: None
- Element Type**: Standard
- Visible**: []
- Initiator**: []
- Included**: []
- Start Offset**: []

Below the tree and properties are tabs: Validate, Emit, IsPresent, Value, Util, Document. The Document tab is selected, showing the following text:

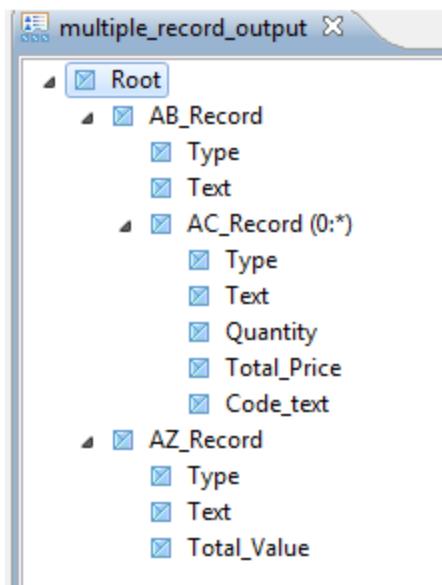
```
1 ATexthdr 00001
2 BTTextdetail1 01
3 C00010
4 D00999
5 BTTextdetail2 02
6 C00001
7 D00100
8 ZTextFooter
```

Next Step

With the input Structure created, let's [create the output Structure](#).

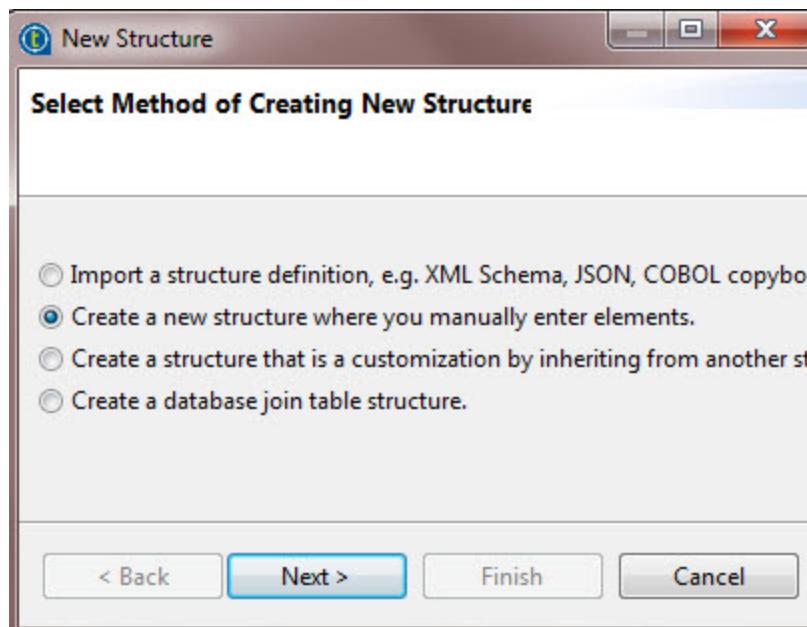
Creating the Output Structure

The objective of this section is to create the following output Structure.

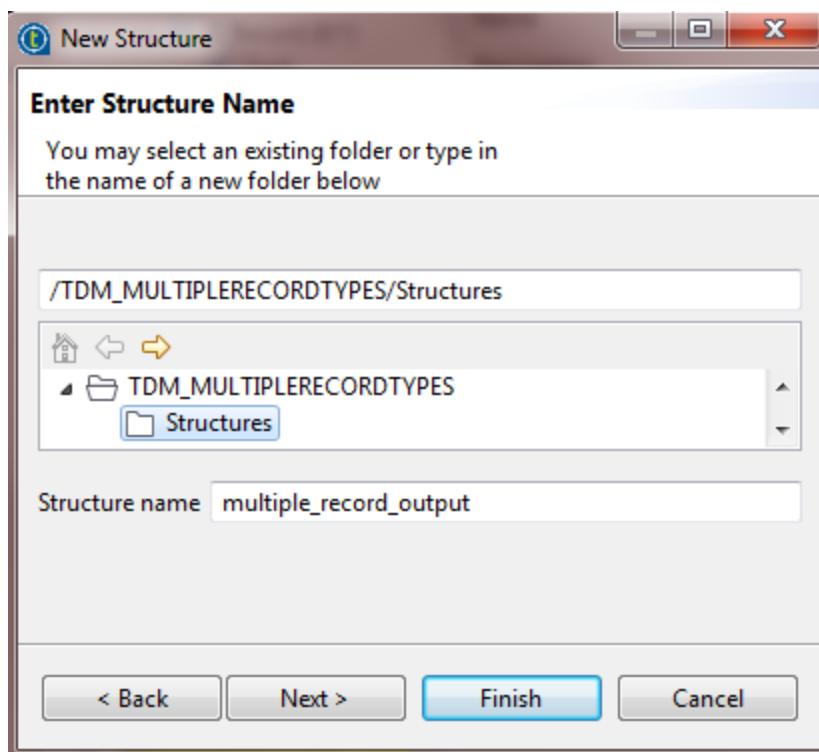


1. Right-click **Hierarchical Mapper > Structures** and select **New > Structure**.

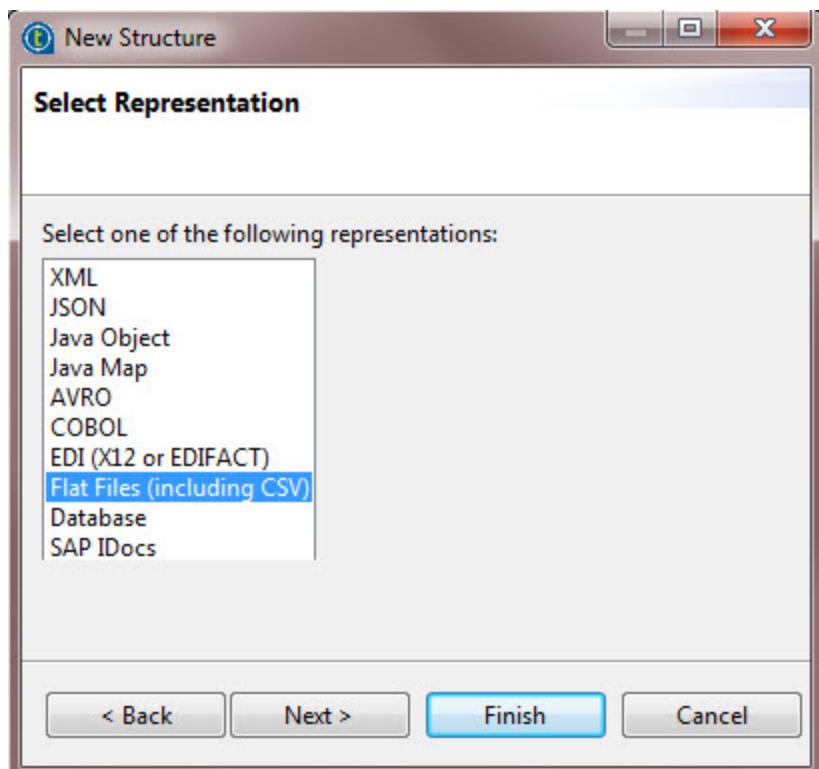
Pick the **Create a new structure where you manually enter elements**. option and click **Next >**:



2. Name the new Structure *multiple_record_output* and click **Next >**.



- Finally, select **Flat Files (including CSV)** for the representation and click **Finish**:

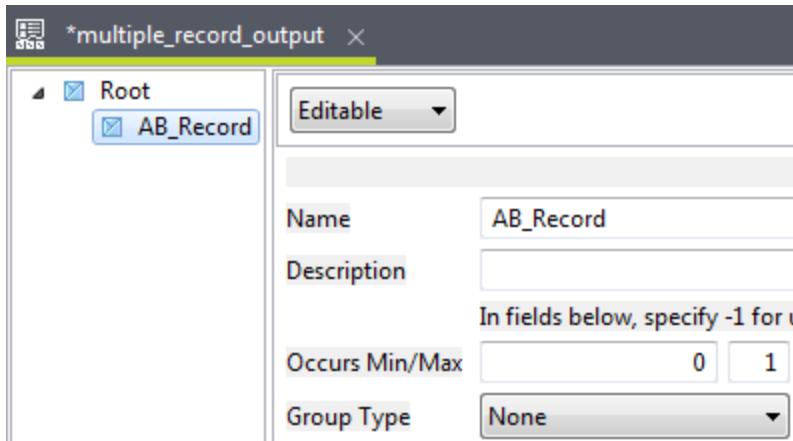


Creating the AB_Record Element

1. Right-click the empty area and select **New Element**.

Name it *Root*, then right-click it and select **New Element** again.

Name the child element *AB_Record*.



2. Right-click **AB_Record** and select **New Element**.

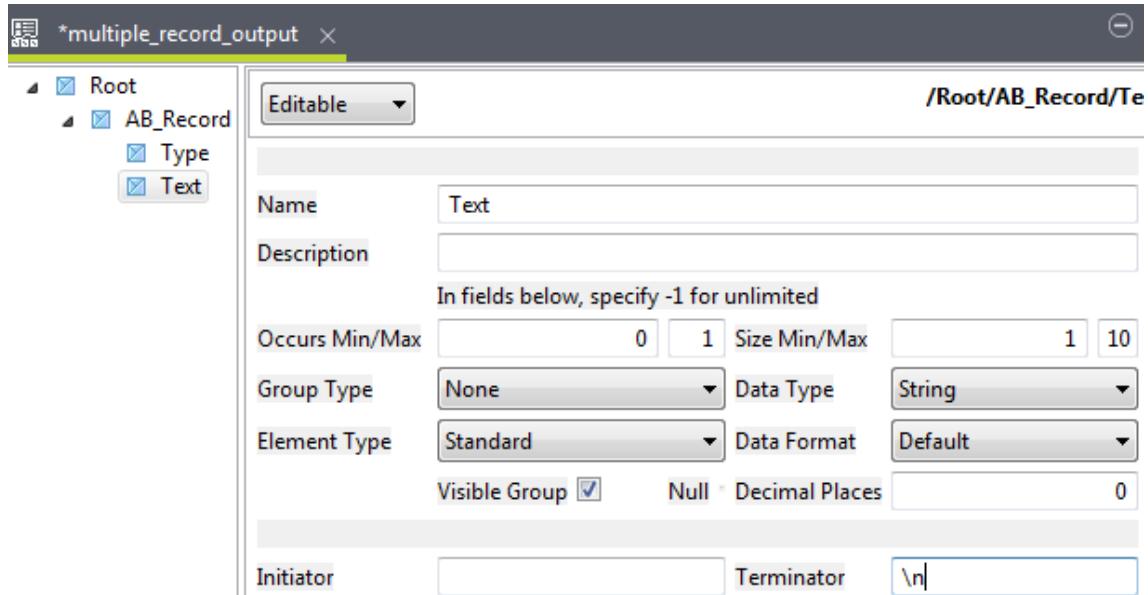
Name it *Type* and set the **Size Min/Max** fields to 2 (Min) and 2 (Max).



3. Right-click **AB_Record** and select **New Element**.

Name it *Text* and set the **Size Min/Max** fields to 1 (Min) and 10 (Max).

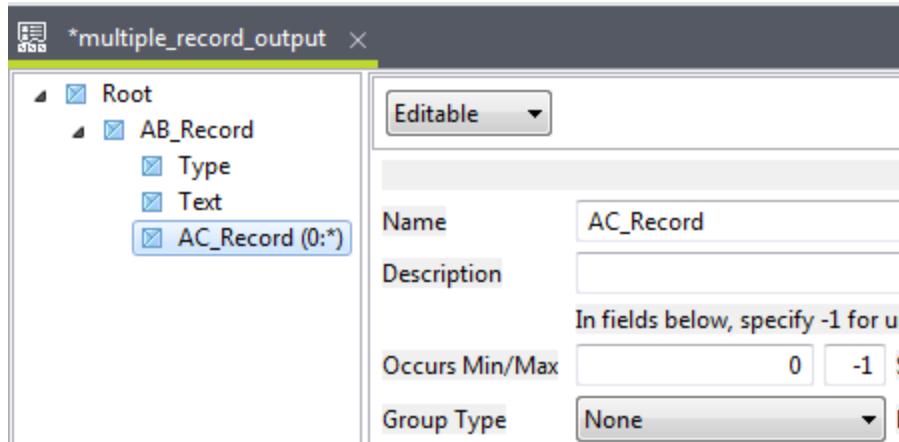
Finally, set the **Terminator** field to `\n`.



Creating the AC_Record Element

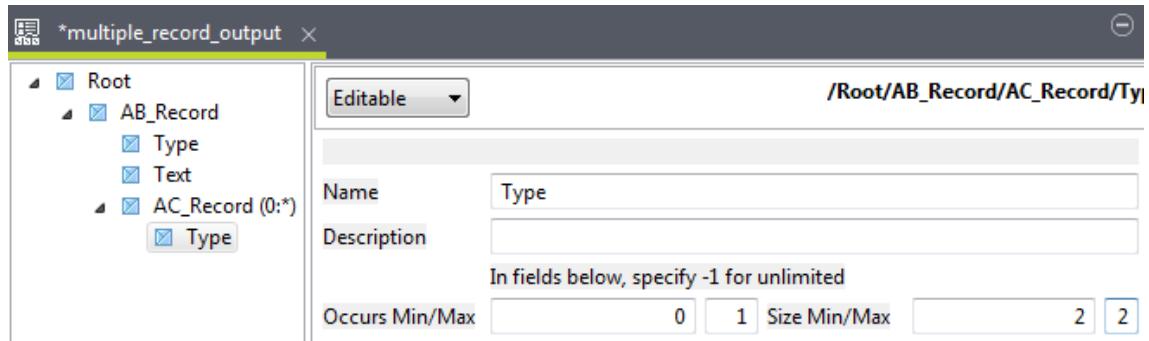
1. Right-click **AB_Record** and select **New Element**.

Name it **AC_Record** and set the **Occurs Min/Max** fields to 0 (Min) and -1 (Max) to make it a looping element.



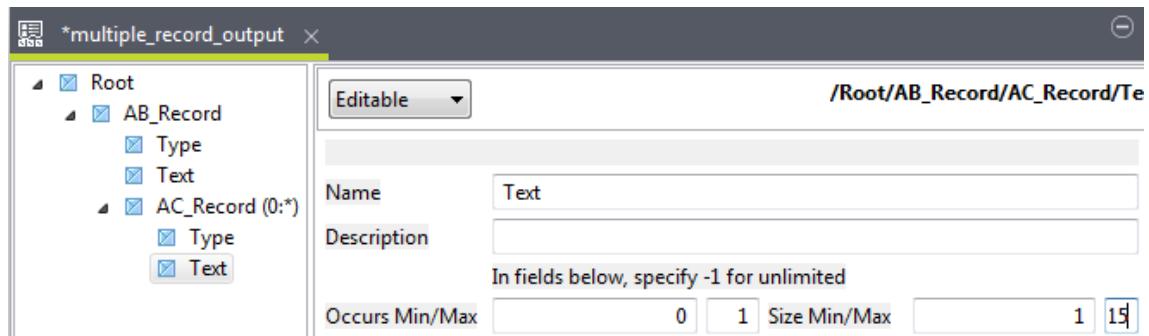
2. Right-click **AC_Record** and select **New Element**.

Name it **Type** and set the **Size Min/Max** fields to 2 (Min) and 2 (Max).



- Right-click **AC_Record** and select **New Element**.

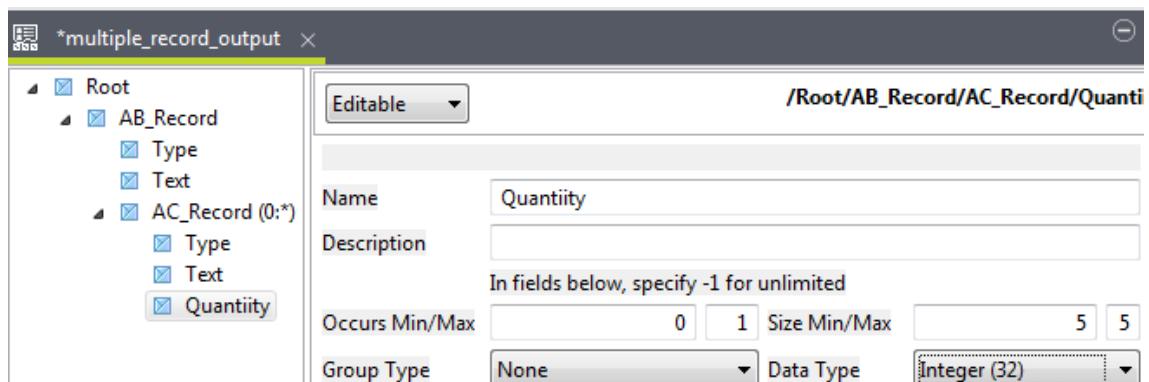
Name it *Text* and set the **Size Min/Max** fields to 1 (Min) and 15 (Max).



- Right-click **AC_Record** and select **New Element**.

Name it *Quantity* and set the **Size Min/Max** fields to 5 (Min) and 5 (Max).

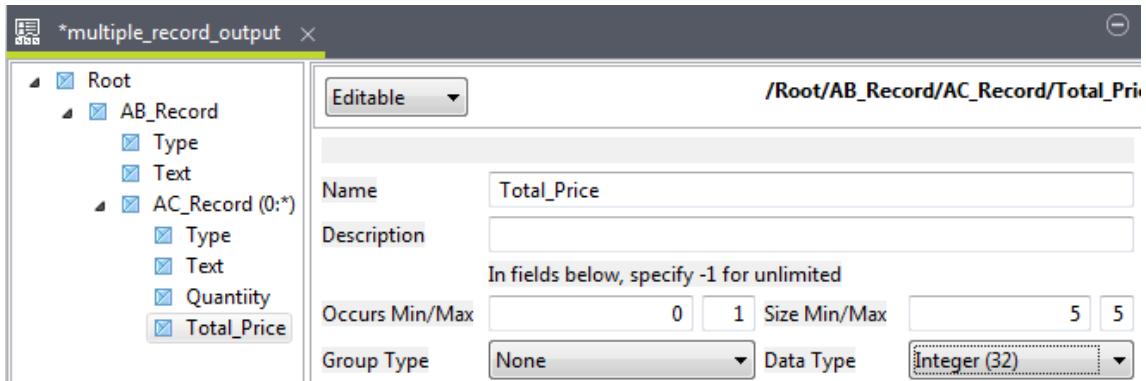
Finally, set **Data Type** to *Integer (32)*.



- Right-click **AC_Record** and select **New Element**.

Name it *Total_Price* and set the **Size Min/Max** fields to 5 (Min) and 5 (Max).

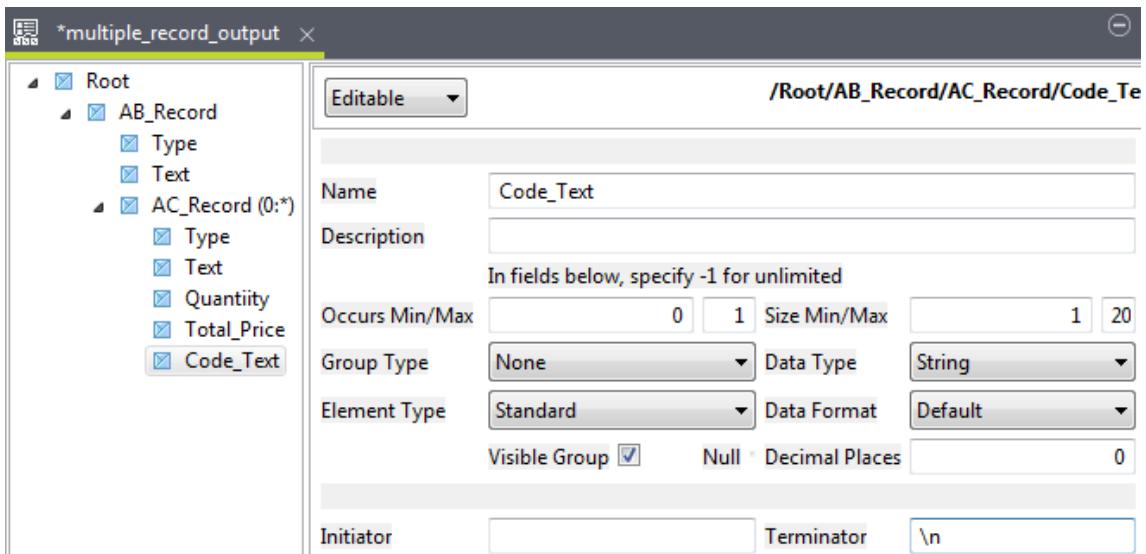
Finally, set **Data Type** to *Integer (32)*.



- Right-click **AC_Record** and select **New Element**.

Name it *Code_Text* and set the **Size Min/Max** fields to 1 (Min) and 20 (Max).

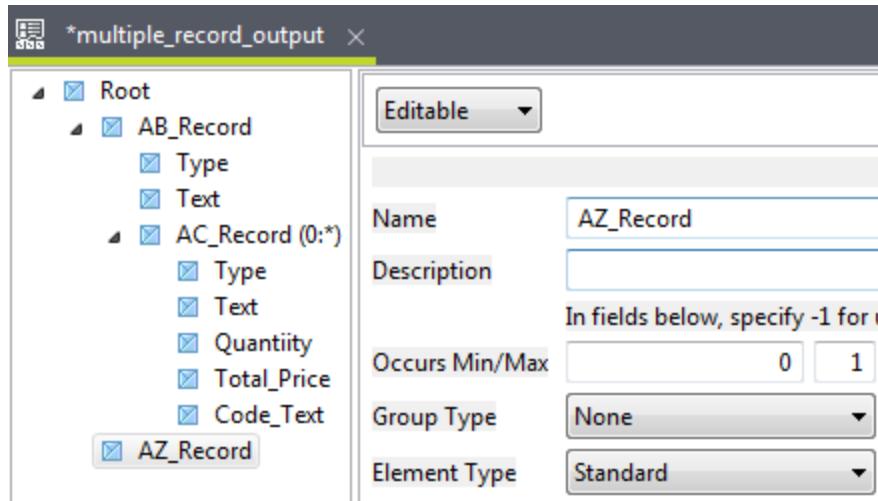
Finally, set the **Terminator** field to *\n*.



Creating the AZ_Record Element

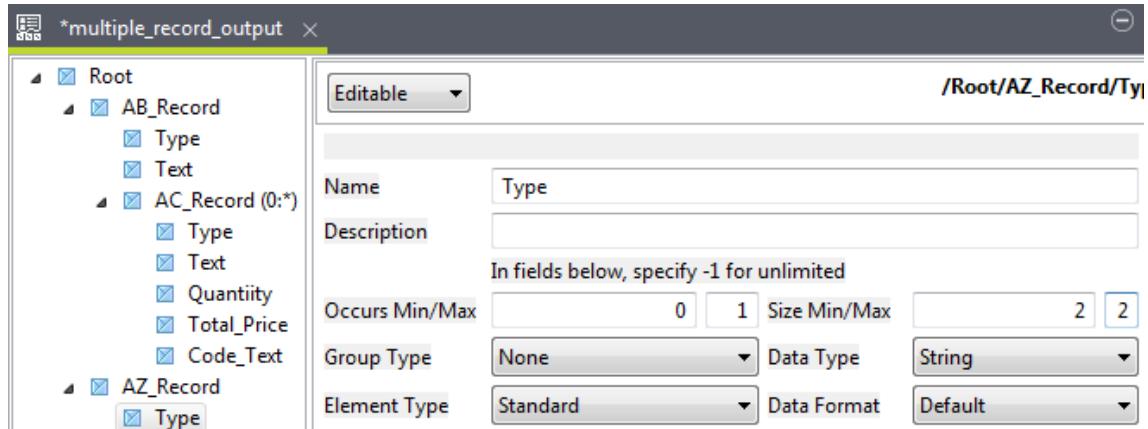
- Right-click **Root** and select **New Element**.

Name it *AZ_Record*.



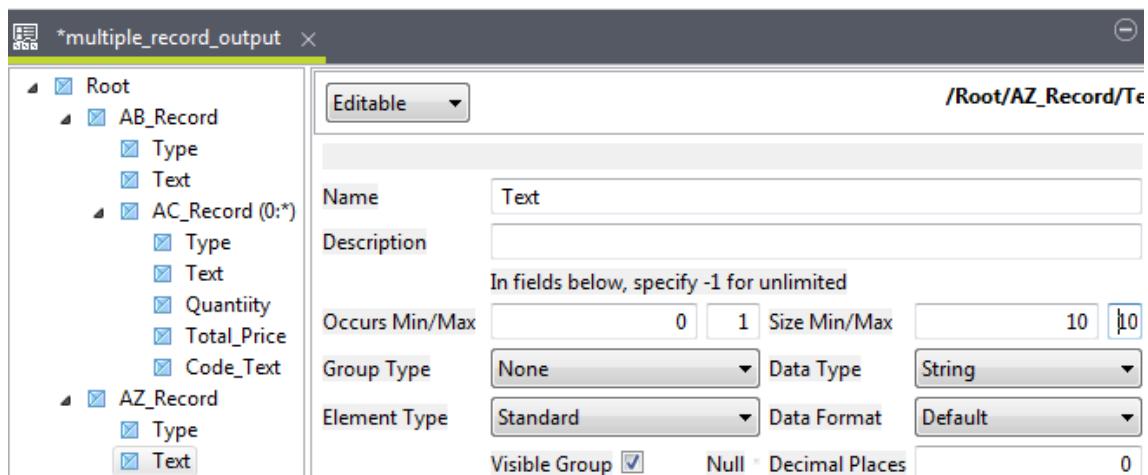
2. Right-click **AZ_Record** and select **New Element**.

Name it **Type** and set the **Size Min/Max** fields to 2 (Min) and 2 (Max):



3. Right-click **AZ_Record** and select **New Element**.

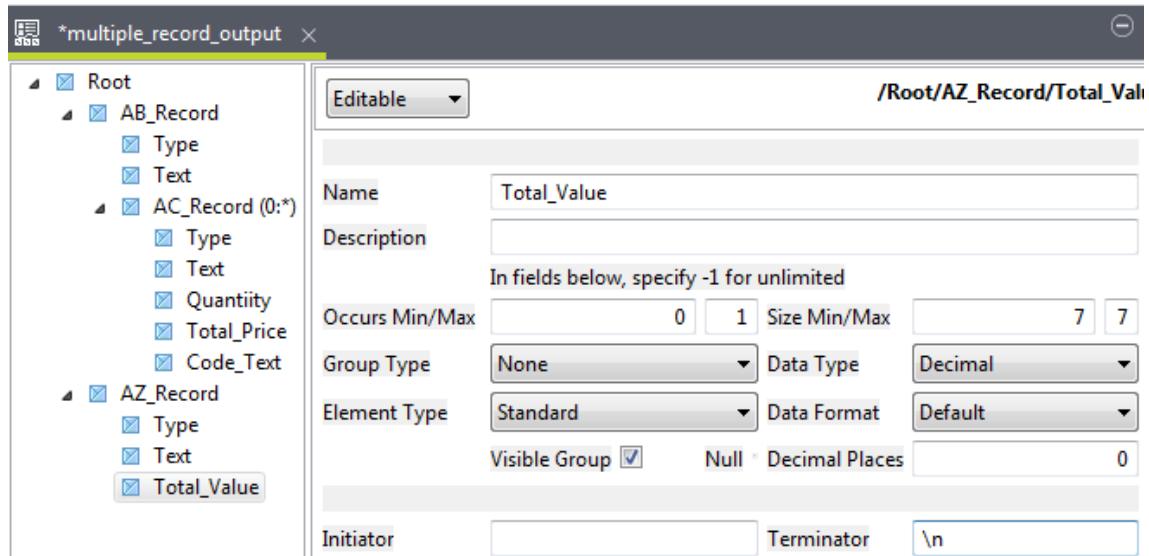
Name it **Text** and set the **Size Min/Max** fields to 10 (Min) and 10 (Max).



4. Right-click **AZ_Record** and select **New Element**.

Name it **Total_Value** and set the **Size Min/Max** fields to 7 for both Min and Max.

Finally, set **Data Type** to *Decimal* and **Terminator** to *\n*.



5. Save the changes to the new output Structure.

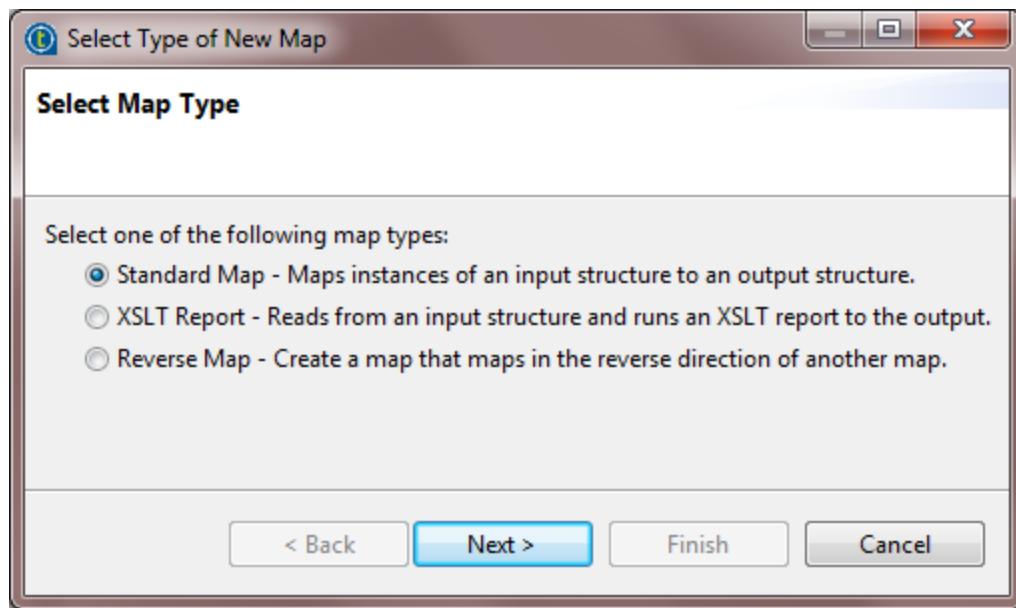
Next

With both the input and output Structures created, let's [map the elements](#).

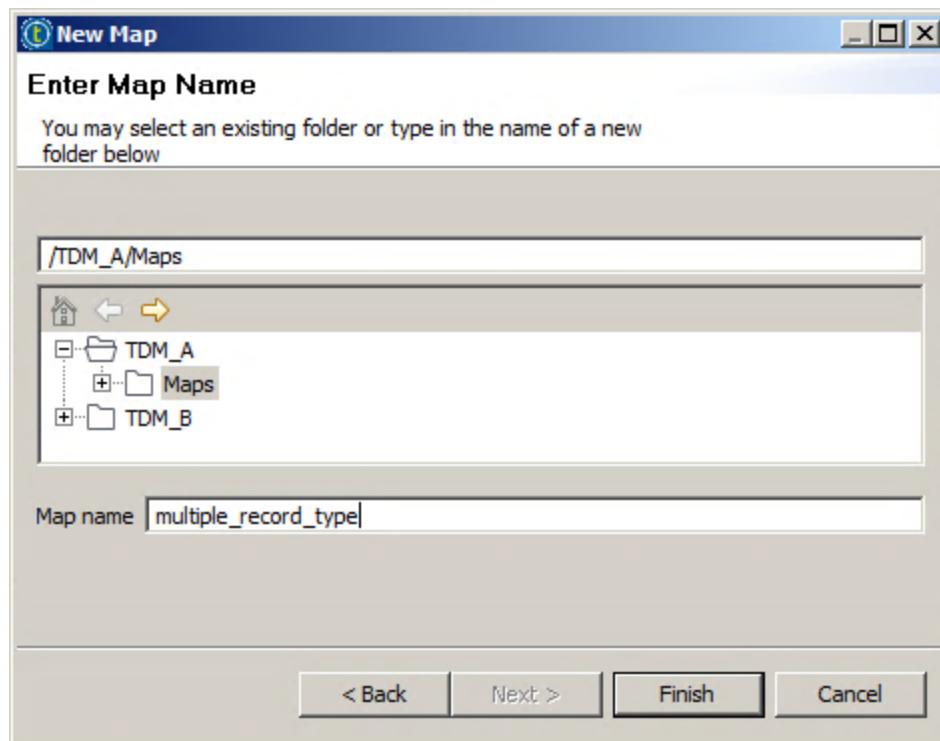
Mapping the Elements

1. Right-click **Hierarchical Mapper > Maps** and select **New > Map**.

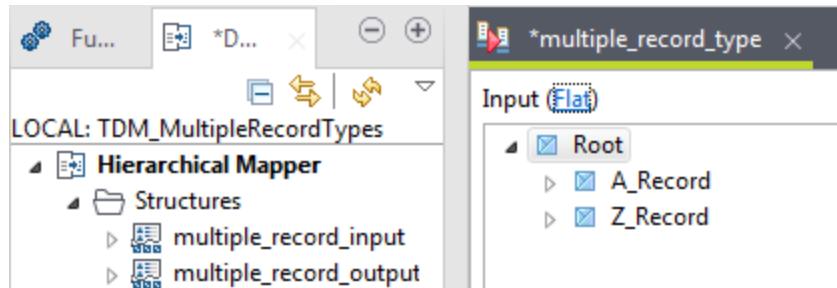
Pick the **Standard Map** option and click **Next >**.



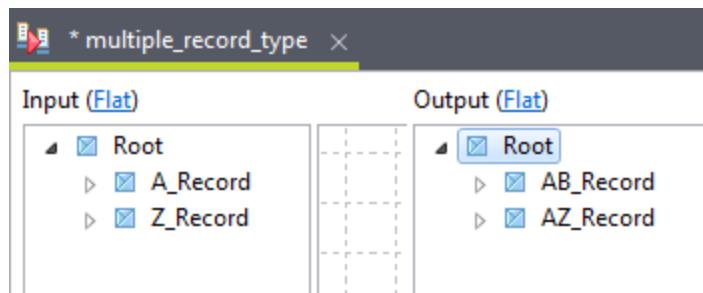
2. Name the new Map **multiple_record_type** and click **Finish**.



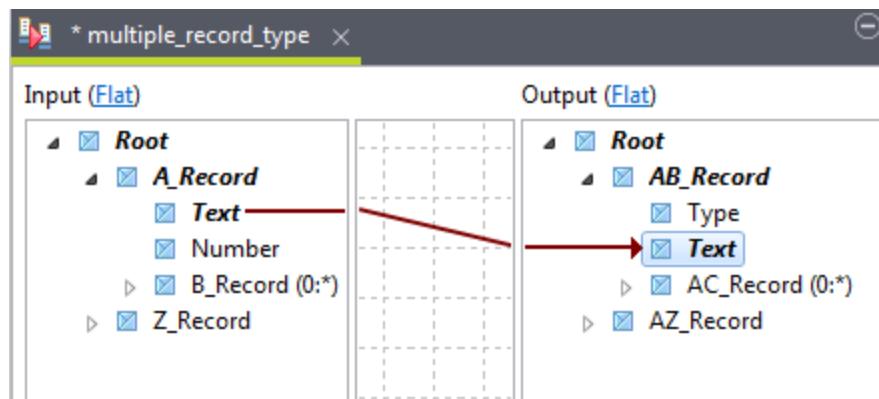
3. Drag & drop the **multiple_record_input** Structure to the **Input** area of the map.



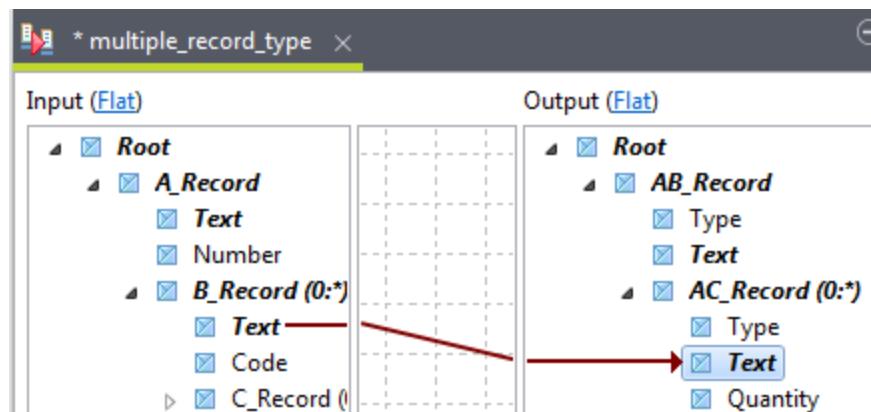
4. Drag & drop the **multiple_record_output** Structure to the **Output** area of the map.



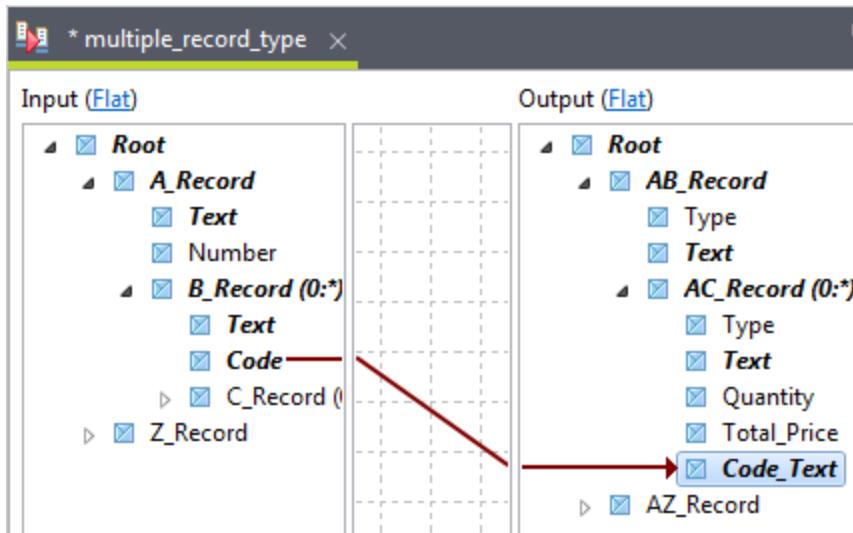
5. Map Root > A_Record > Text to Root > AB_Record > Text.



6. Map Root > A_Record > B_Record > Text to Root > AB_Record > AC_Record > Text.

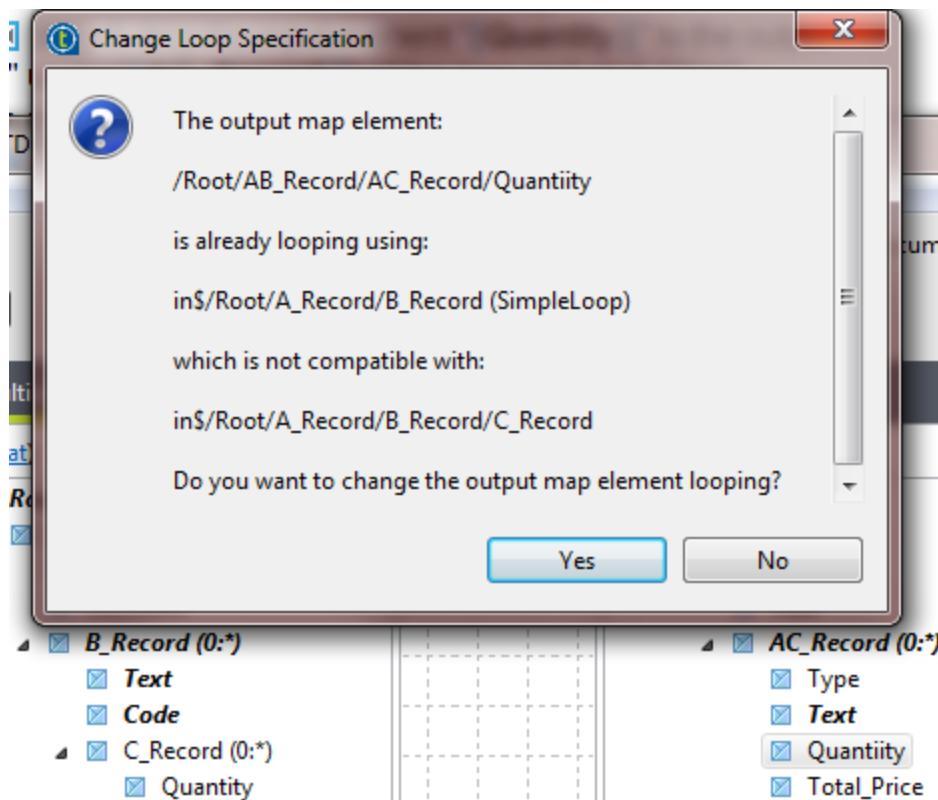


7. Map Root > A_Record > B_Record > Code to Root > AB_Record > AC_Record > Code_Text.

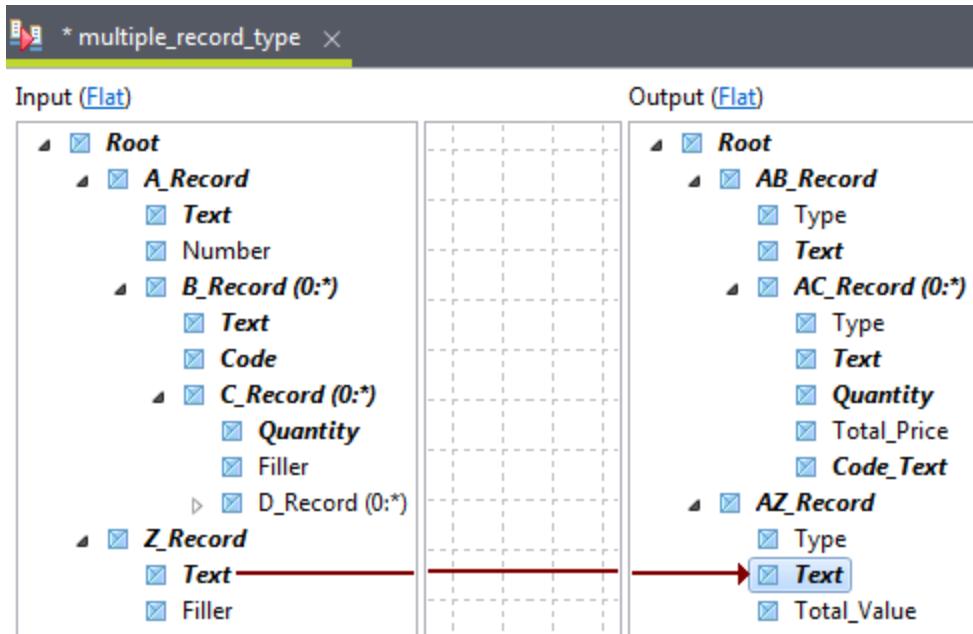


8. Map Root > A_Record > B_Record > C_Record > Quantity to Root > AB_Record > AC_Record > Quantity.

Click **Yes** when offered to change the output map element looping. *Talend Data Mapper* detects that a nested looping is necessary to reach the *Quantity* element in the **Input** area, and offers to add this nested looping automatically.



9. Map Root > Z_Record > Text to Root > AZ_Record > Text.



Adding Functions to Perform Computations

The objective of this section is to use Functions to perform simple computations, like the total price or the total value of the items.

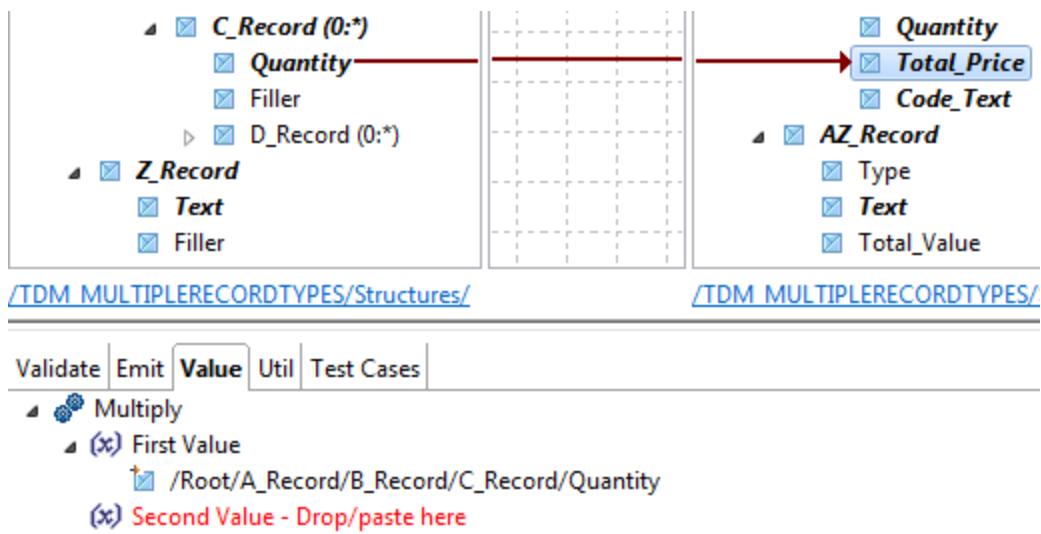
1. Select **Total_Price** in the **Output** area, then drag and drop a **Functions > Arithmetic > Multiply** Function to its **Value** tab.

The screenshot shows the TDM MULTIPLERECORDTYPES application interface. On the left, the **Functions** palette is open, showing categories like **Aggregate**, **Arithmetic**, **Special**, etc. Under **Arithmetic**, the **Multiply** function is selected.

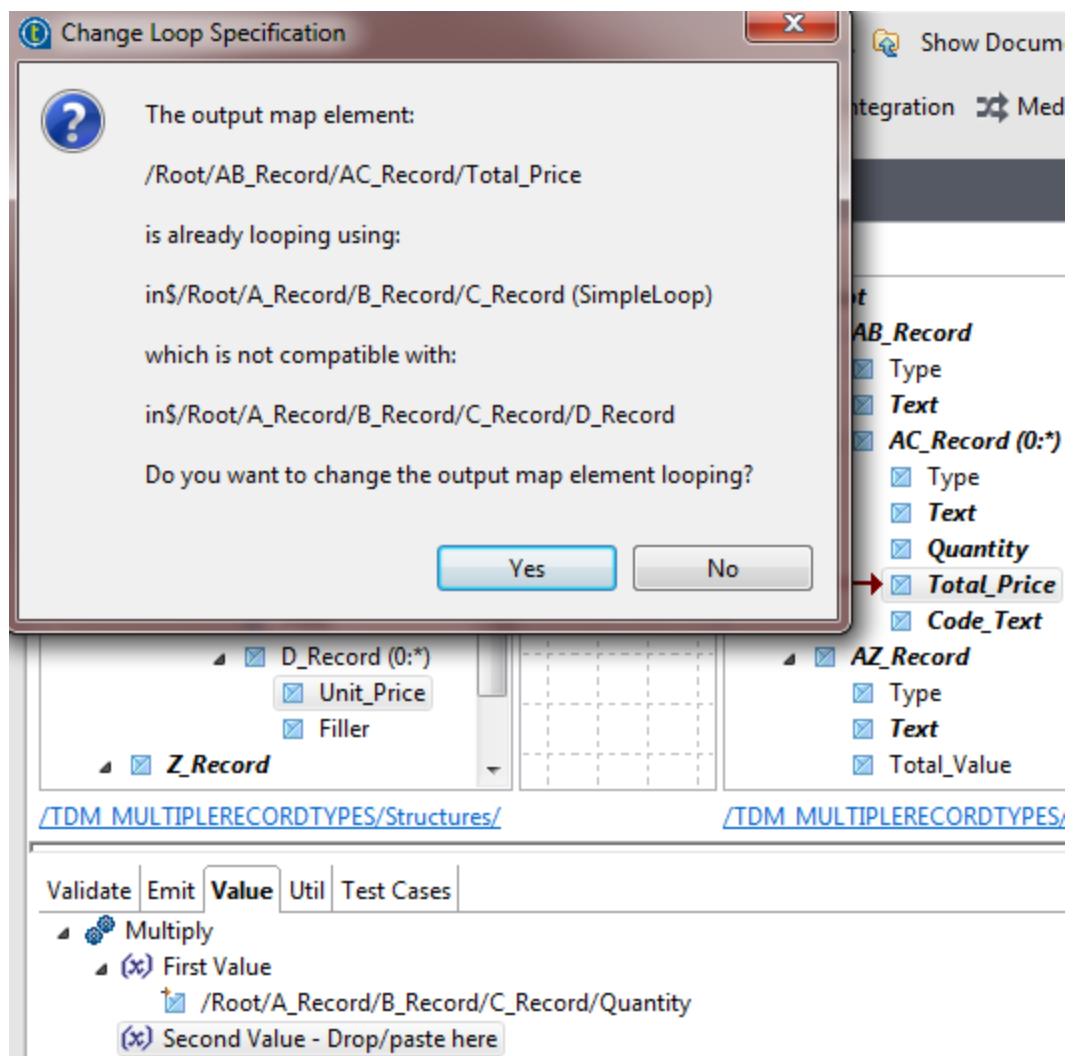
In the center, the **Input** and **Output** structures are visible. The **Output** structure includes the **AC_Record (0:*)** node with the **Total_Price** element selected.

At the bottom, a computation setup window is open, showing the **Multiply** function with two **Value** fields labeled **(x) First Value - Drop/paste here** and **(x) Second Value - Drop/paste here**.

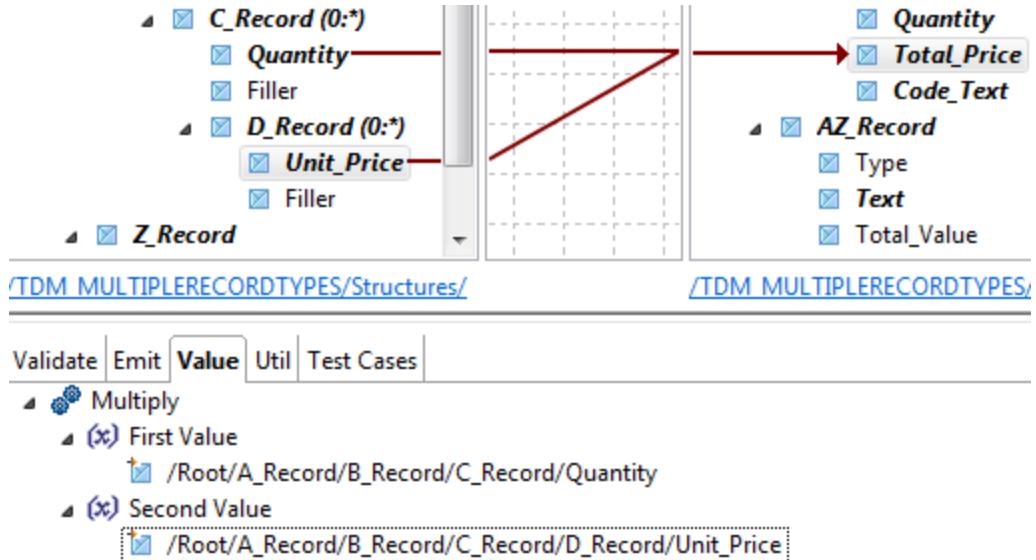
2. Drag and drop **C_Record > Quantity** from the **Input** area to the **First Value** field. Make sure not to select the element prior to dragging it, or the **Value** tab and the **First Value** field will disappear from view. You have to drag the **Quantity** element without selecting it first with a left click.



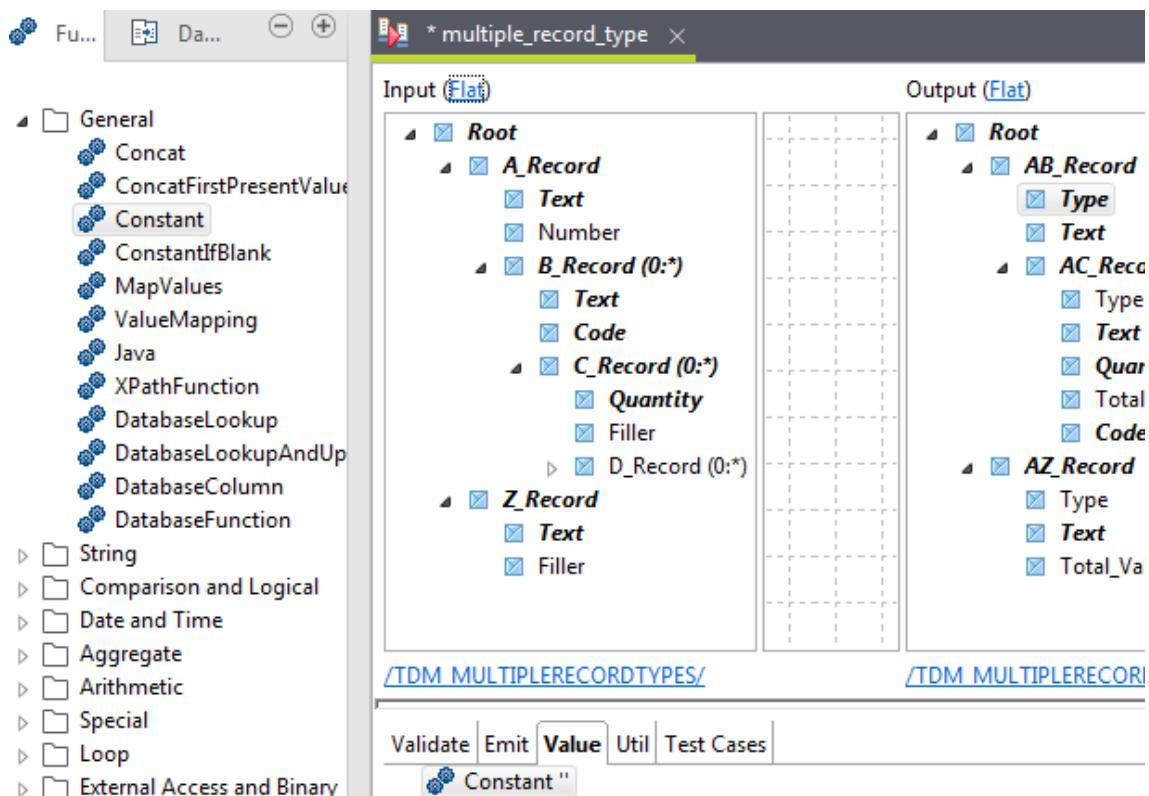
3. Drag and drop **D_Record > Unit_Price** from the **Input** area to the **Second Value** field and click **Yes** when offered to change the output map element looping.



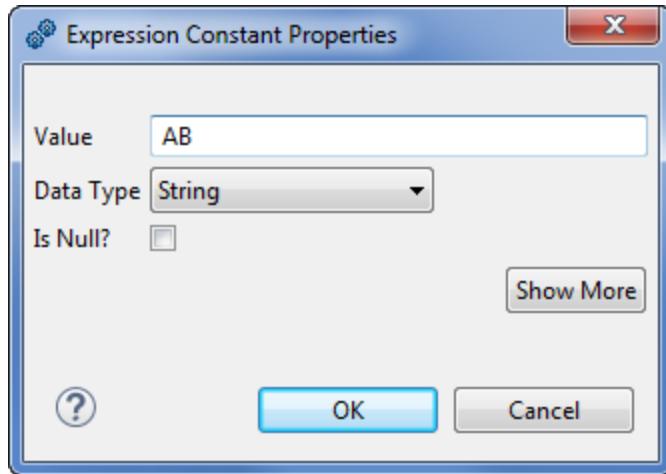
4. The final Multiply Function should look like the following.



- Now select Root > AB_Record > Type in the Output area, then drag and drop a Functions > General > Constant Function to its Value tab.



- Double-click the Constant element, set the Value field to AB and click OK.



7. Select Root > AB_Record > AC_Record > Type in the Output area, then drag and drop a Functions > General > Constant Function to its Value tab. Double-click the Constant element, set the Value field to AC and click OK.

8. Select Root > AZ_Record > Type in the Output area, then drag and drop a Functions > General > Constant Function to its Value tab. Double-click the Constant element, set the Value field to AZ and click OK

9. Select Root > AZ_Record > Total_Value in the Output area, then drag and drop a Functions > Aggregate > AgSum

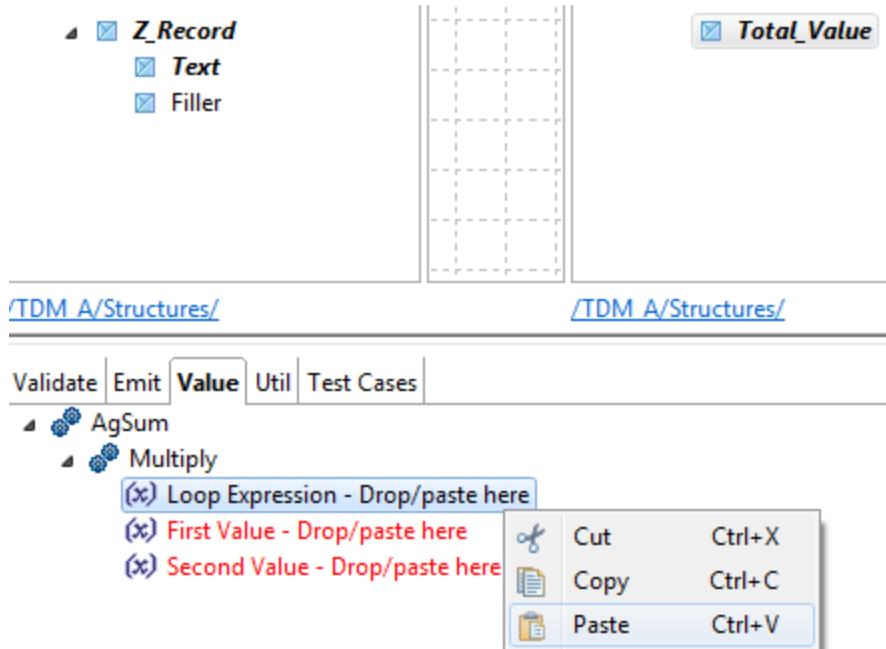
Function to its **Value** tab.

10. Drag and drop a **Functions > Arithmetic > Multiply** Function on top of the **AgSum** field.

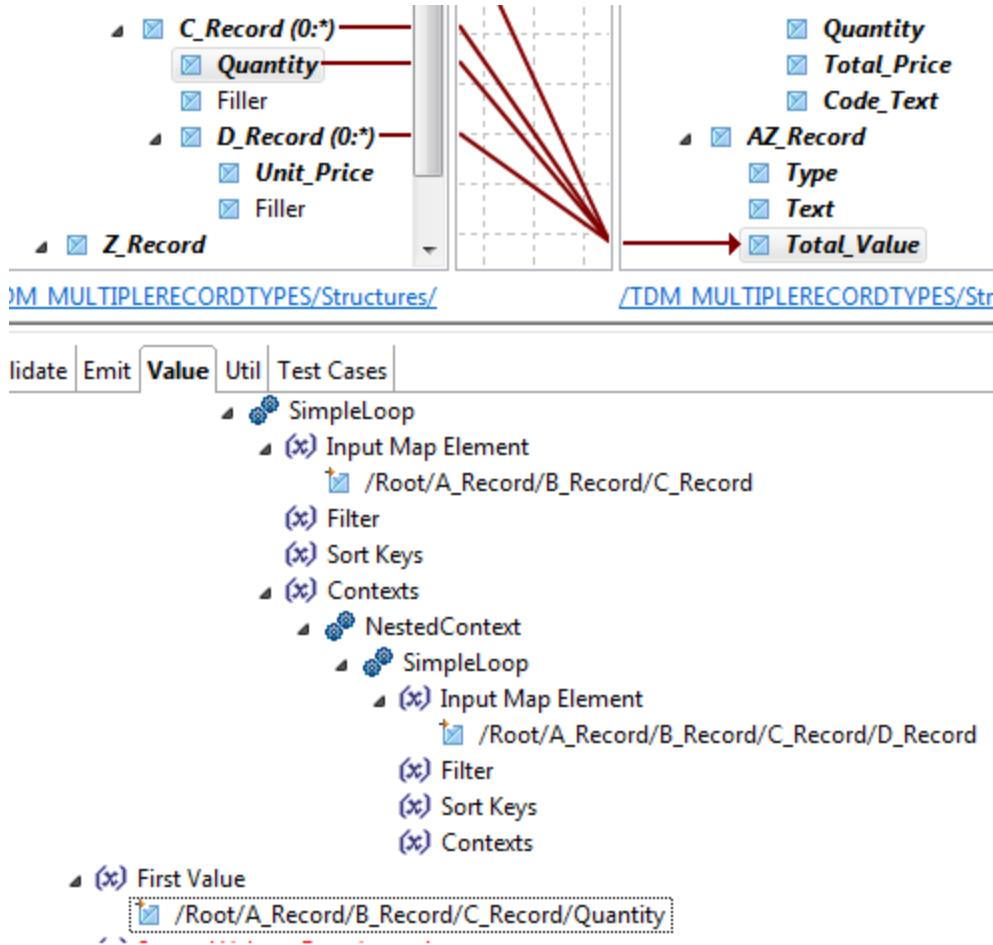
11. Select **Root > AB_Record > AC_Record** in the **Output** area, then switch to the **Loop** tab.

Right-click the **SimpleLoop** field and select **Copy** to reuse this formula.

12. Select **Root > AZ_Record > Total_Value** again in the **Output** area, right-click **Loop Expression** and select **Paste**.

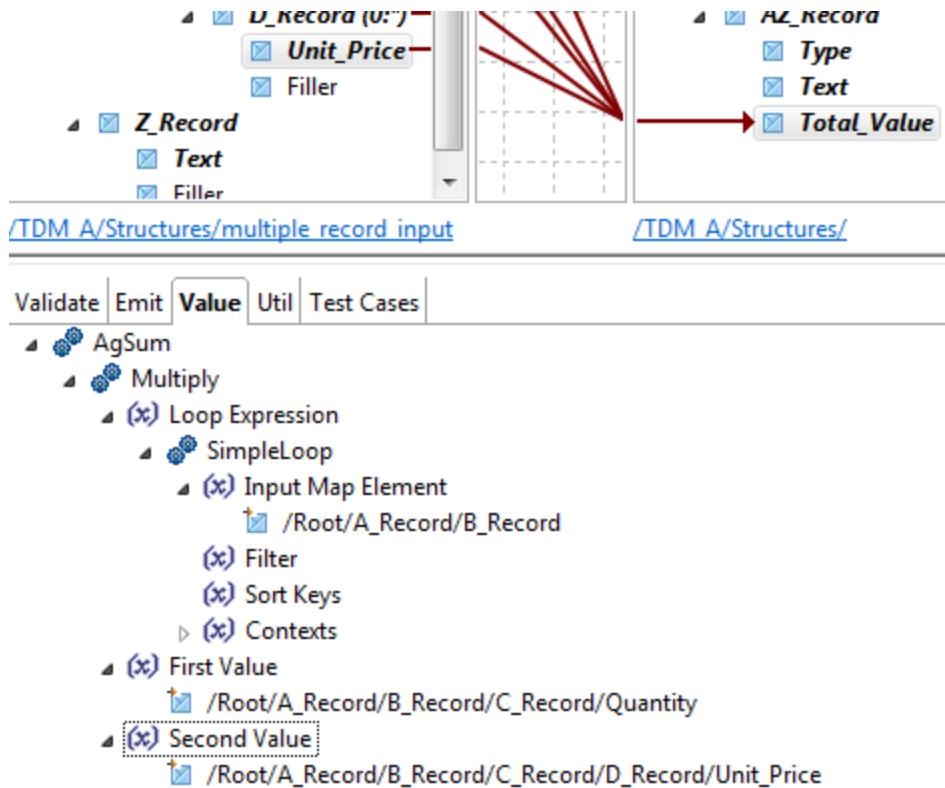


13. Drag and drop **C_Record > Quantity** from the **Input** area to the **First Value** field of the **AgSum** Function.



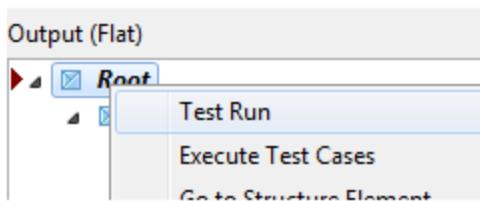
14. Drag and drop D_Record > Unit_Price from the **Input** area to the **Second Value** field of the **AgSum** Function.

The final **AgSum** Function should look like the following.

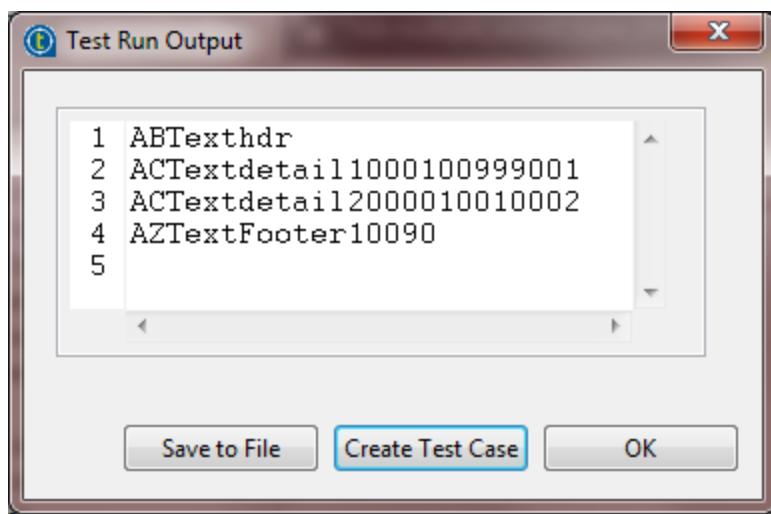


Test Run

1. Right-click **Root** in the **Output** area and select **Test Run**.



2. Check that the result looks like the following.

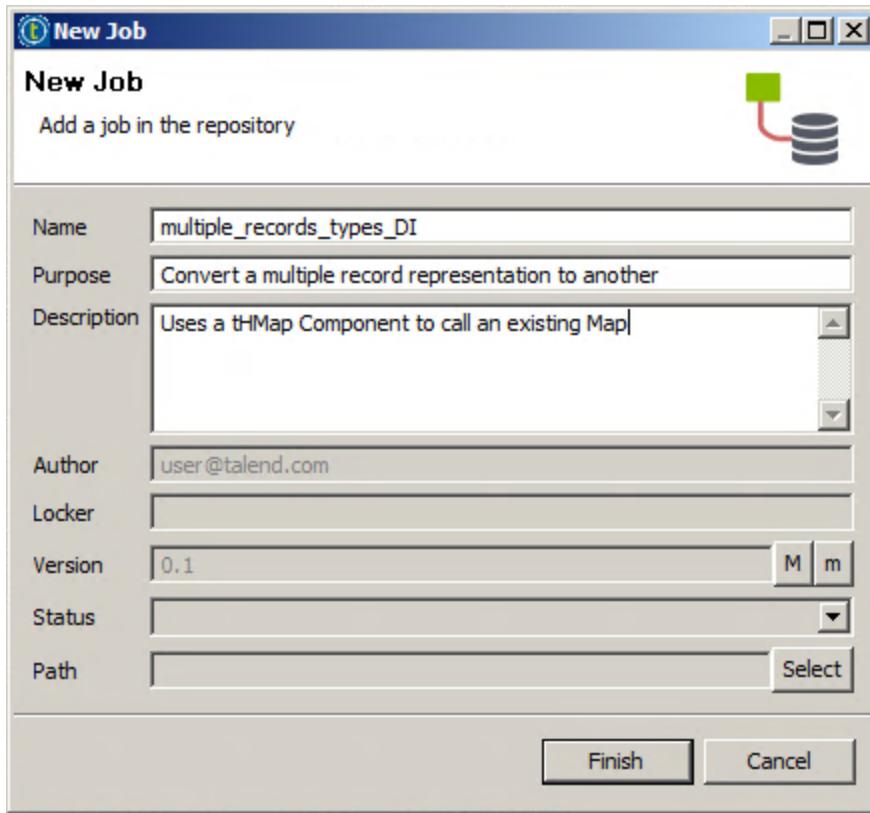


Next Step

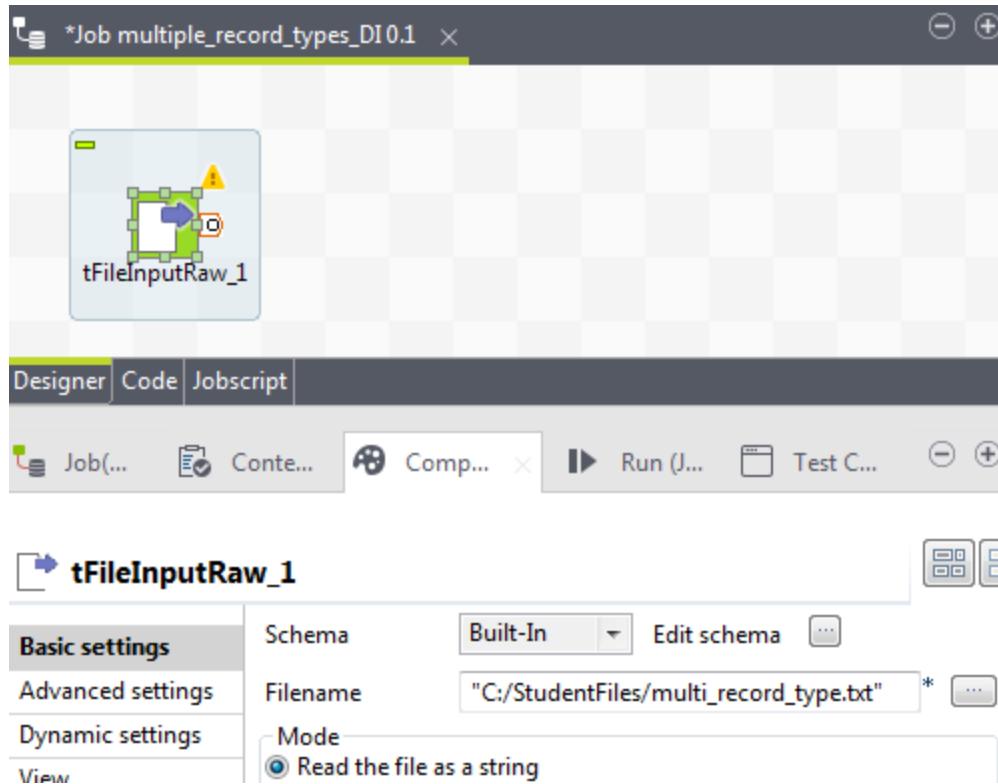
Now let's [create a DI Job](#) to call the mapping that was just completed.

Creating the DI Job

1. Switch to the **Integration** perspective and create a Job called *multiple_record_types_DI*.

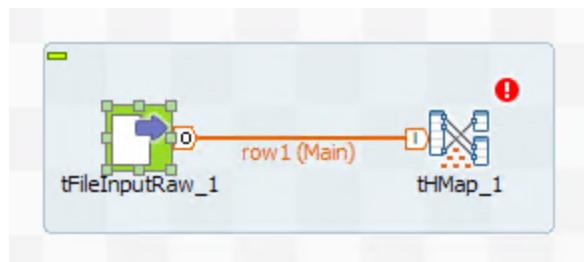


2. Place a **tFileInputRaw** Component in the canvas and double-click it to open its **Component** view.
Click the ... button next to the **Filename** field and select the *C:/StudenFiles/multiple_record_type.txt* file.



- Place a **tHMap** Component on the canvas, acknowledging the red error sign that indicates the Map could not be found because it has not been configured yet.

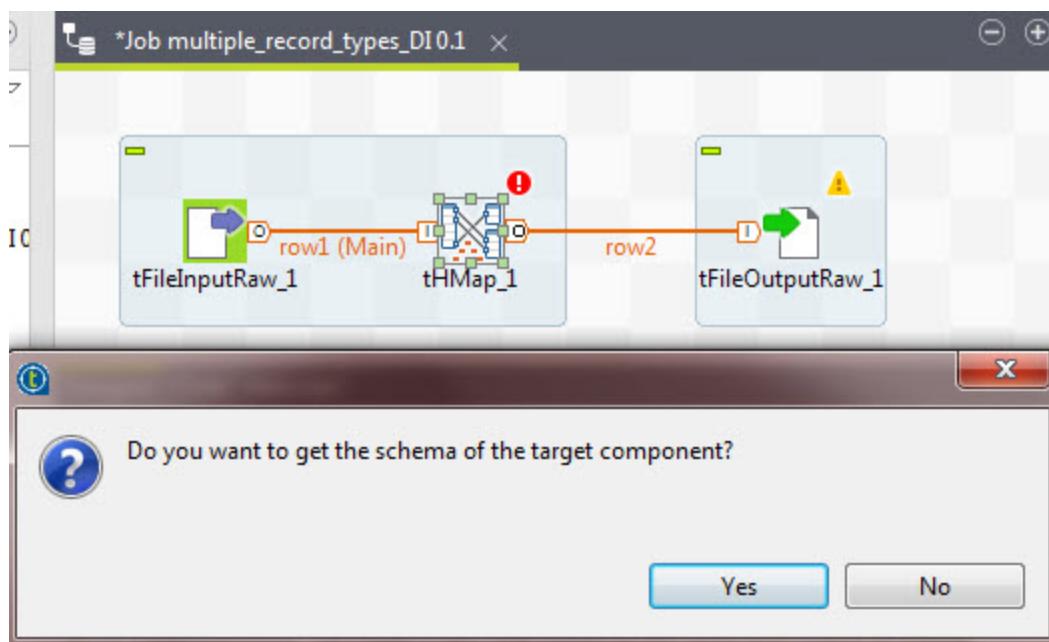
Right-click **tFileInputRaw**, select **Row > Main** and connect the Component to **tHMap**.



- Place a **tFileOutputRaw** Component on the canvas.

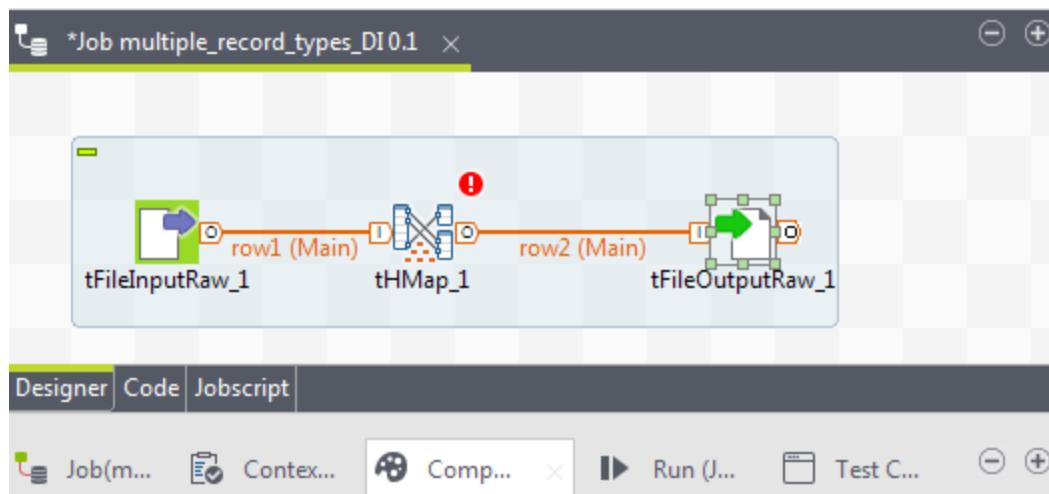
Right-click **tHMap**, select **Row > Main** and connect the Component to **tFileOutputRaw**.

Click **Yes** when asked to get the schema from the target Component.

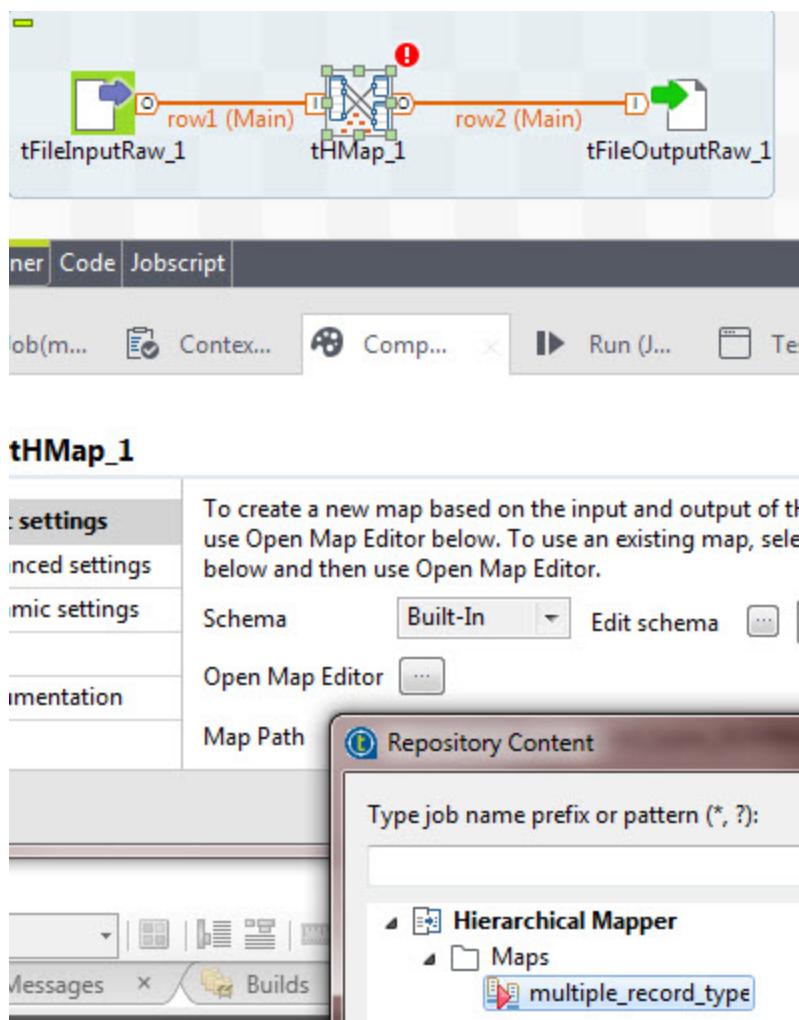


- Double-click **tFileOutputRaw** to open its Component view.

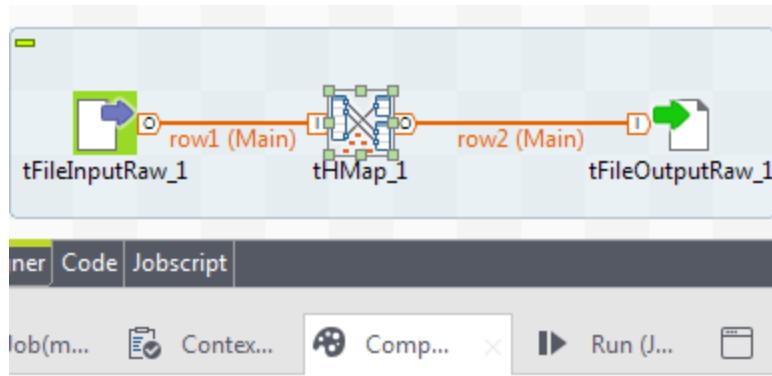
Click the ... button next to the **Filename** field, select **C:/StudentFiles/multiple_record_type.txt** and manually change the file name to **C:/StudentFiles/multiple_record_type_out.txt**.



- Select **tHMap** to display its **Component** view.
- Click the ... button to the right of the **Map Path** field.
- Select the **multiple_record_type** Map created in the previous section and click **OK**.



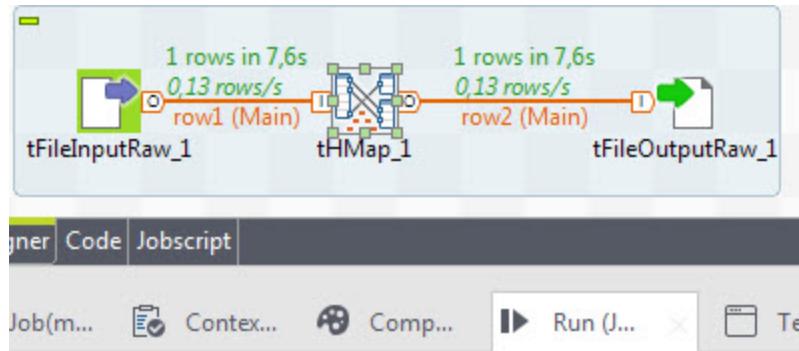
9. Note that **Read Input As** is automatically set to *Single column* and **Write Output As** is set to *String (single column)*.



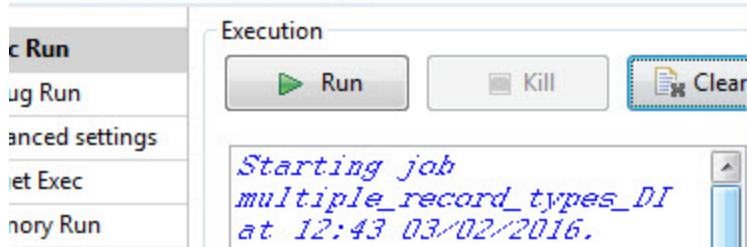
tHMap_1

settings	To create a new map based on the input and output of use Open Map Editor below. To use an existing map, se below and then use Open Map Editor.
inced settings	
imic settings	Schema <input type="button" value="Built-In"/> Edit schema <input type="button"/>
umentation	Open Map Editor <input type="button"/>
	Map Path "multiple_record_type"
	Read Input As <input checked="" type="radio"/> Single column
	Write Output As <input type="radio"/> Data Integration columns <input checked="" type="radio"/> String (single column) <input type="radio"/> Byte array (single column)

- Finally, save all the changes (Structures, Map and DI Job) and run the Job.



multiple_record_types_DI



- Open the output file C:\StudentFiles\multiple_record_type_out.txt and check it has the following content.

```

C:\StudentFiles\multiple_record_type_out.txt - Notepad
File Edit Search View Encoding Language Settings Macros
multiple_record_type_out.txt
1 ABTexthdr
2 ACTextdetail1000100999001
3 ACTextdetail2000010010002
4 AZTextFooter10090
5

```

Next Step

This lesson is almost over. Head to the [Wrap-Up](#) section for a summary of the concepts reviewed in this lesson.

Wrap-Up

In this lesson, you learned how to:

- » Convert a multiple record type Structure to another multiple record type Structure
- » Apply this transformation to flat files using a DI Job and a tHMap Component

Next Step

Congratulations, you successfully completed this lesson. Click the **Check your status with this unit** button below in order to save your progress. Then click **Completed. Let's continue >** on the next screen to jump to the next lesson.

**This page intentionally left blank to ensure new chapters
start on right (odd number) pages.**

LESSON 4

Performing a Database Lookup

This chapter discusses the following.

Overview	124
Creating the Database Structure	125
Creating the Map	129
Wrap-Up	137

Overview

Lesson Overview

The objective of this exercise is perform a database lookup directly from a Map, in order to retrieve the value of an Element from a database based on some condition.

The example will reuse the *multiple_record_type* Map from the previous exercise, and retrieve the value of *Total_Price* from a database.

Objectives

After completing this lesson, you will be able to:

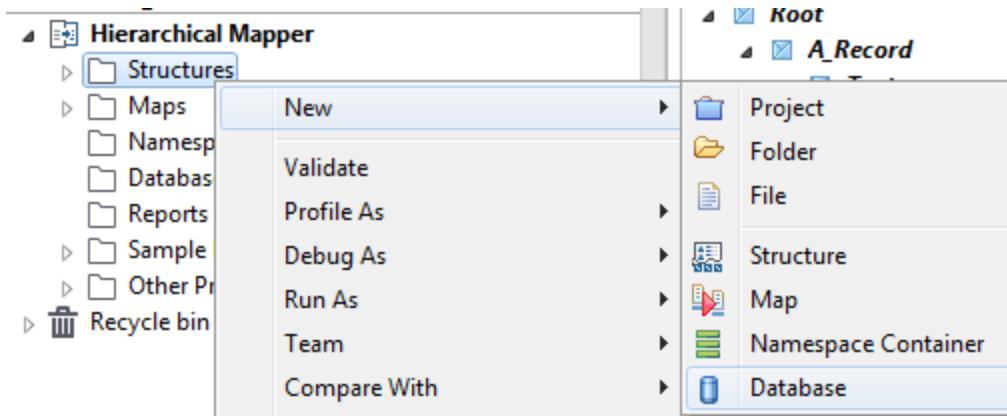
- » Create a database connection and the associated Structures that can be reused from within a Map
- » Compute the value of an Element at runtime, extracting it from a database based on a condition on other Elements

Next Step

First, let's [create the database connection](#) and the associated Structures that will be used to perform the lookup later on.

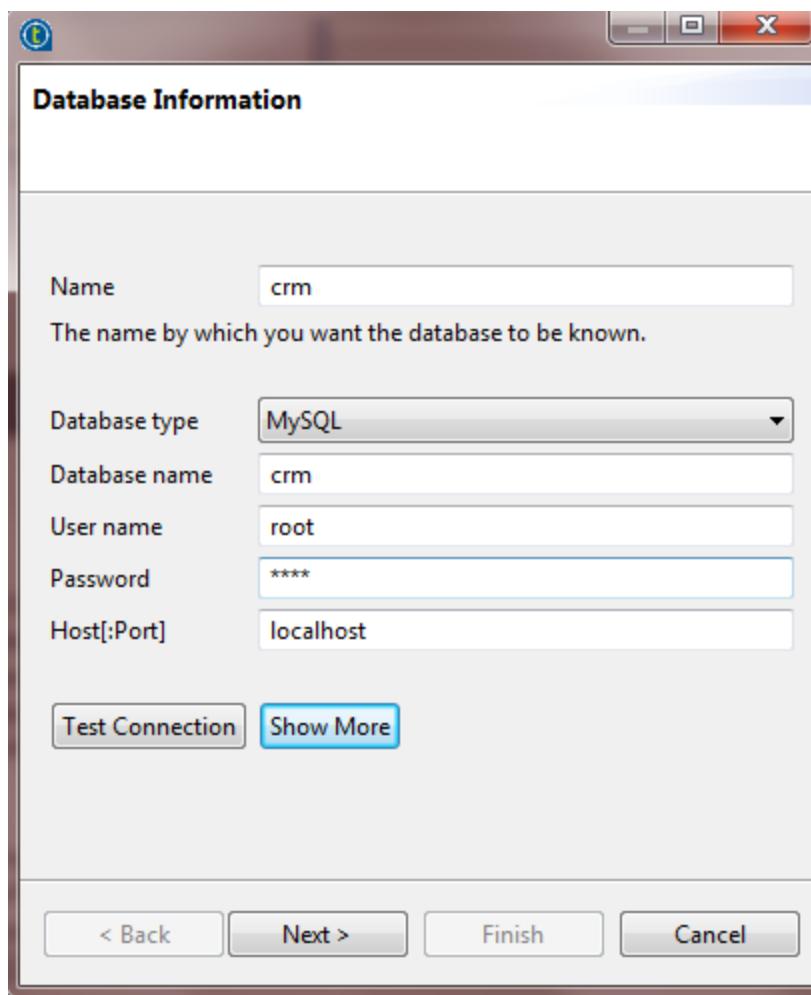
Creating the Database Structure

1. In the Mapping Perspective, right-click **Hierarchical Mapper > Structures** and select **New > Database**.

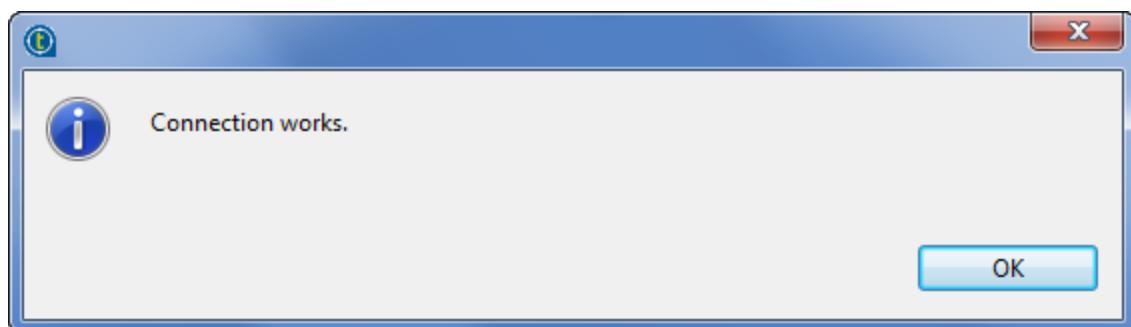


2. Enter the following details for the database connection and click **Test Connection**.

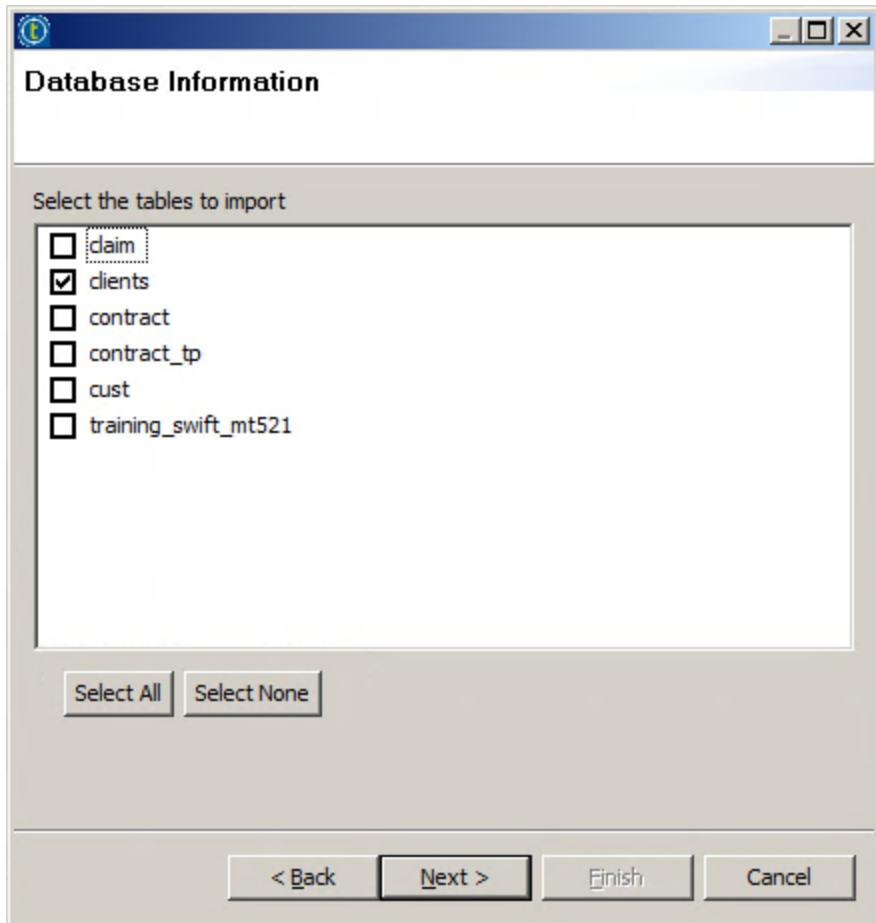
Field	Value
Name	crm
Database type	MySQL
Database name	crm
User name	root
Password	root
Host [:Port]	localhost



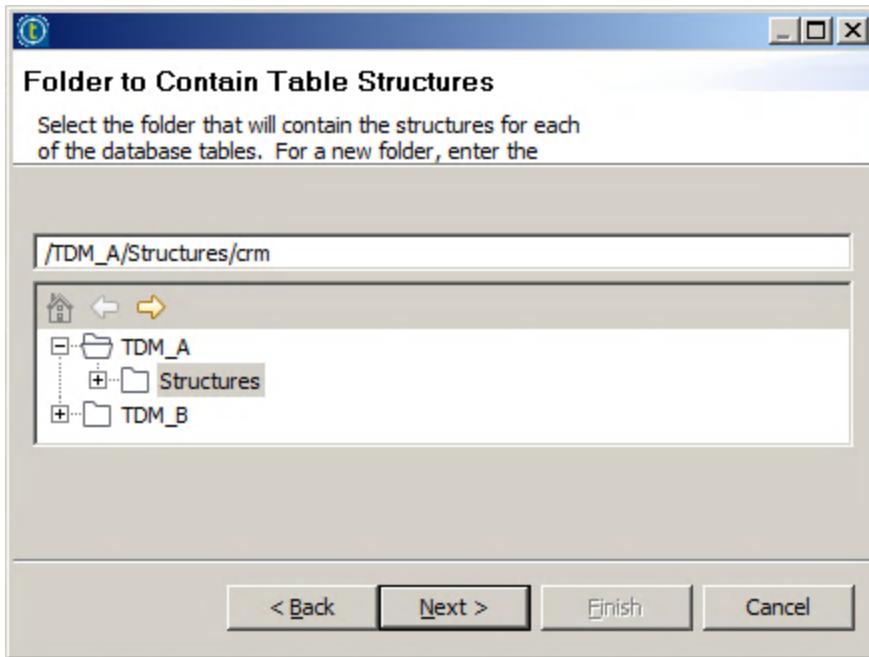
3. Click OK and then Next >.



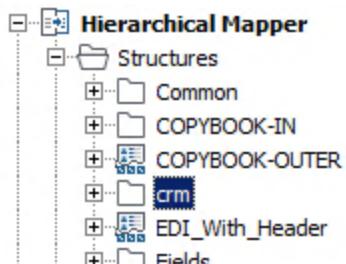
4. Select the **clients** table from the list and click Next >.



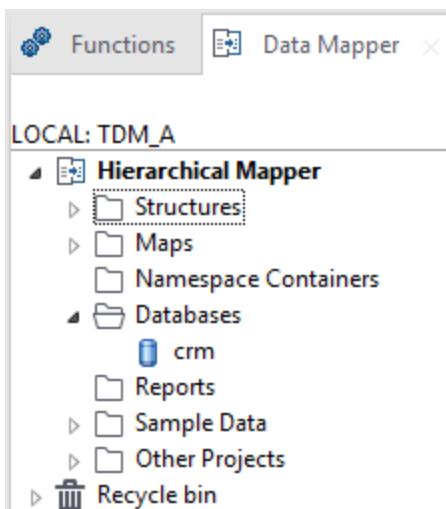
5. Keep the default saving location for the new Structure and click **Next >**.



6. Click **Finish** and check that the **crm** Structure was created under **Hierarchical Mapper > Structures**.



7. Also check that a **crm** connection was created under **Hierarchical Mapper > Databases**.

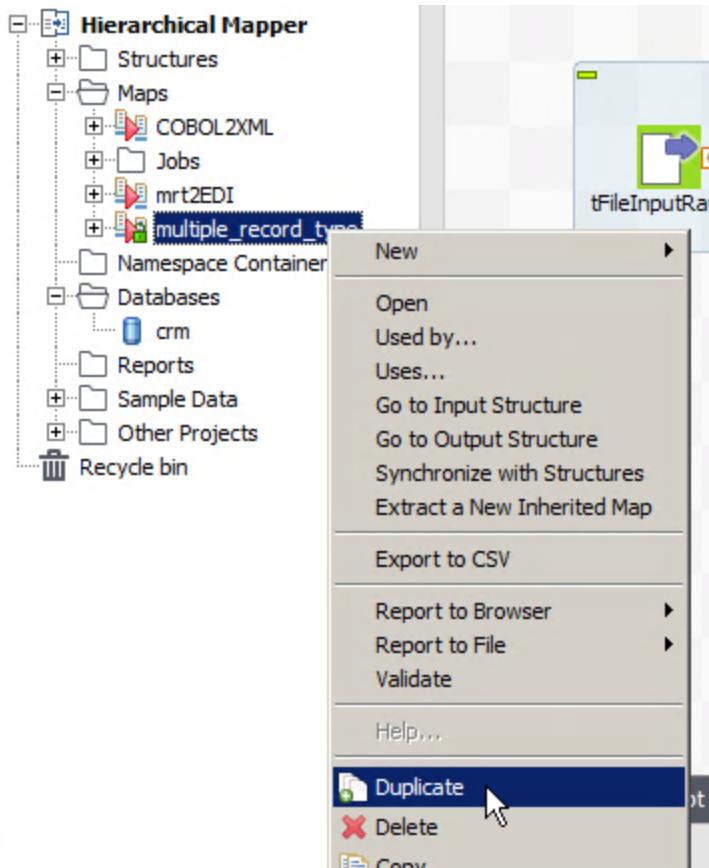


Next Step

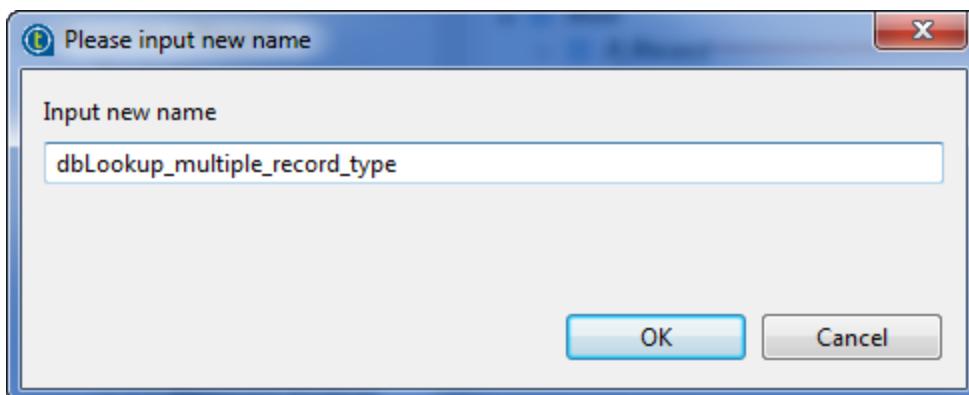
Now let's [create the Map](#) that will perform the actual lookup in the database.

Creating the Map

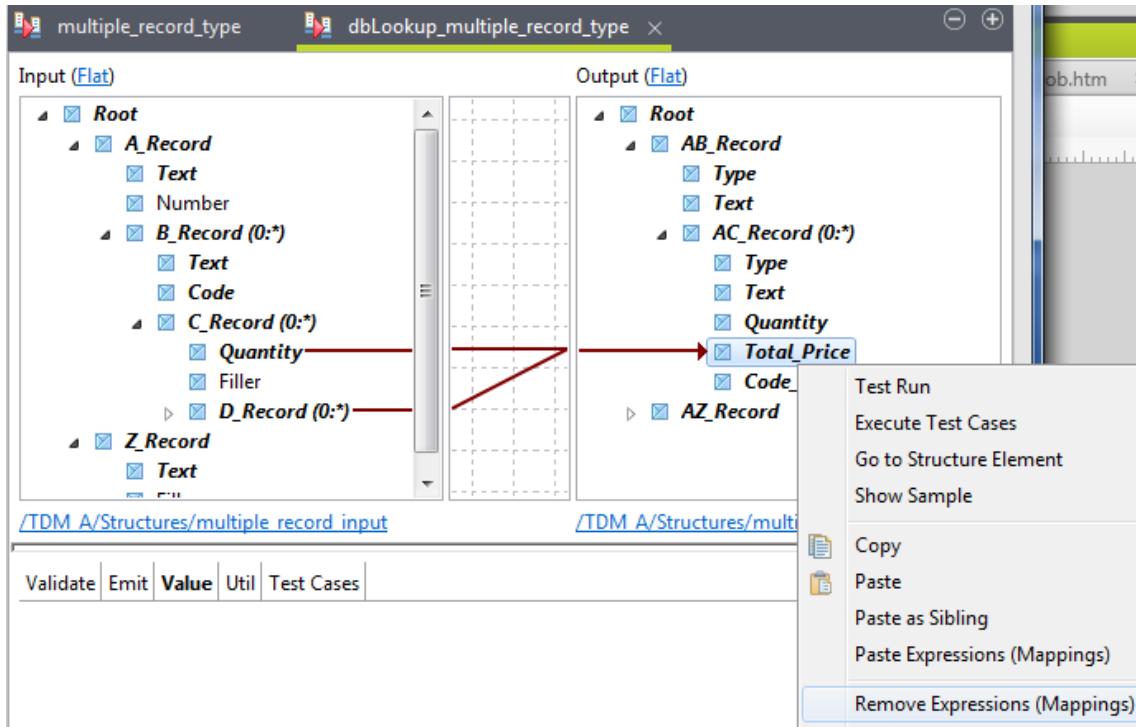
1. Right-click **Hierarchical Mapper > Maps > multiple_record_type** and select **Duplicate**.



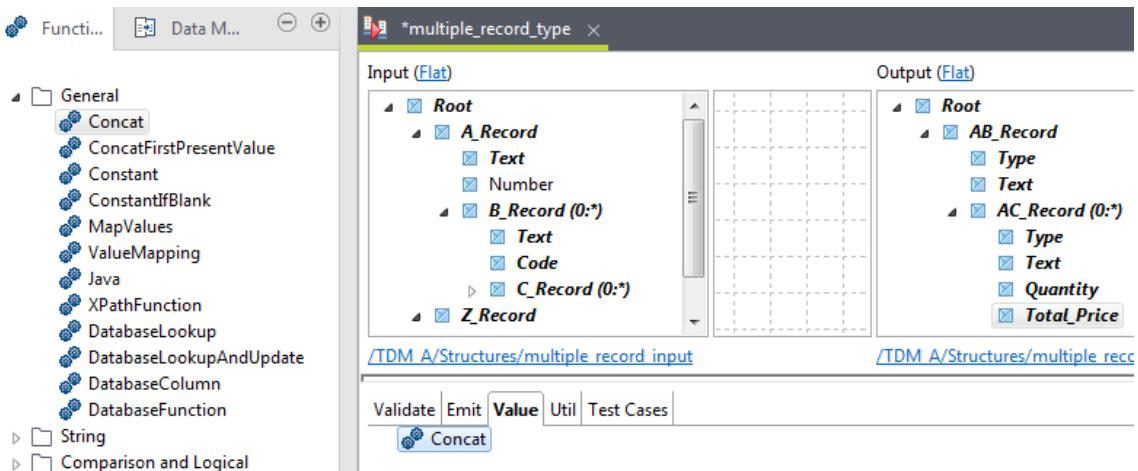
2. Change the name to **dbLookup_multiple_record_type** and click **OK**.



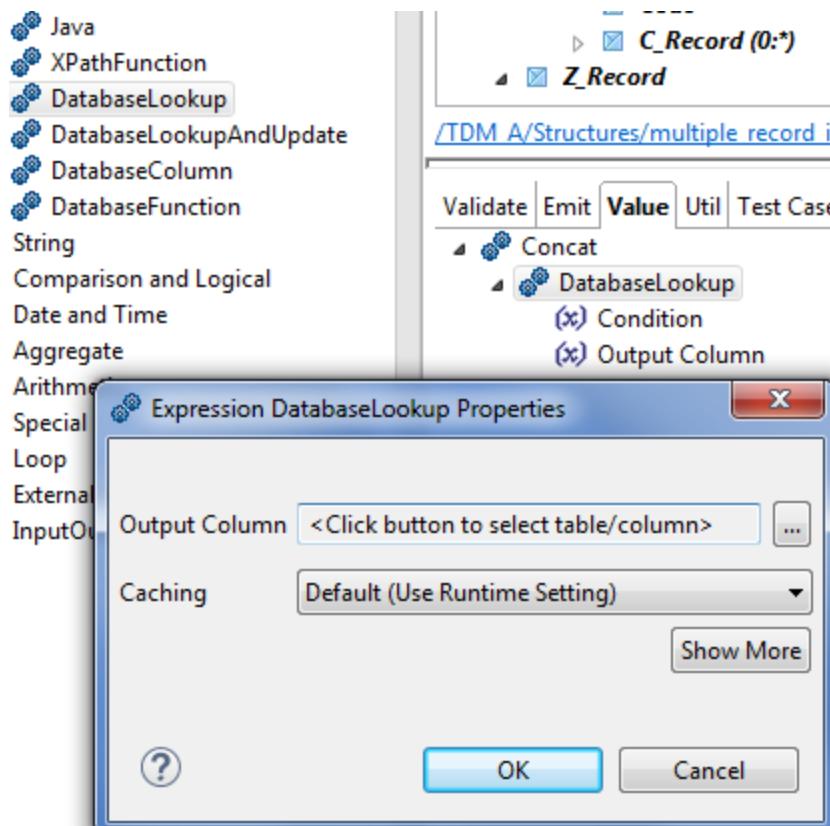
3. Open **dbLookup_multiple_record_type**, right-click **Root > AB_Record > AC_Record > Total_Price** in the **Output** area and select **Remove Expressions (Mappings)**.



- To set the value of **Total_Price** to the result of the database lookup, drag a **Functions > General > Concat** Function to the **Value** tab.

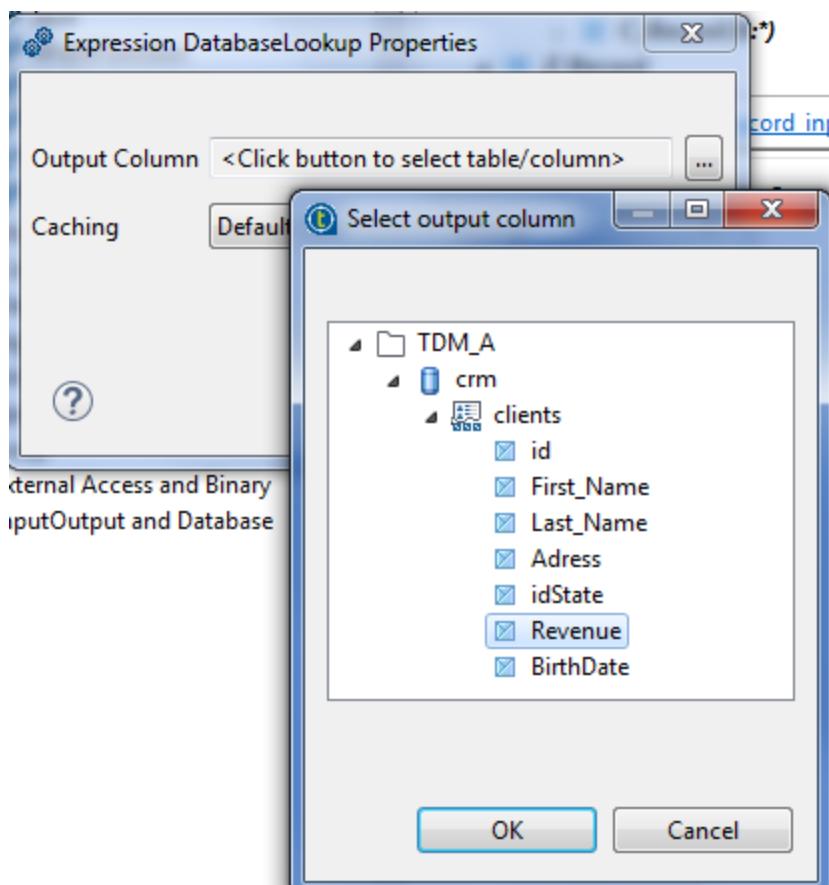


5. Drag and drop a Functions > General > DatabaseLookup Function on top of the Concat Function.

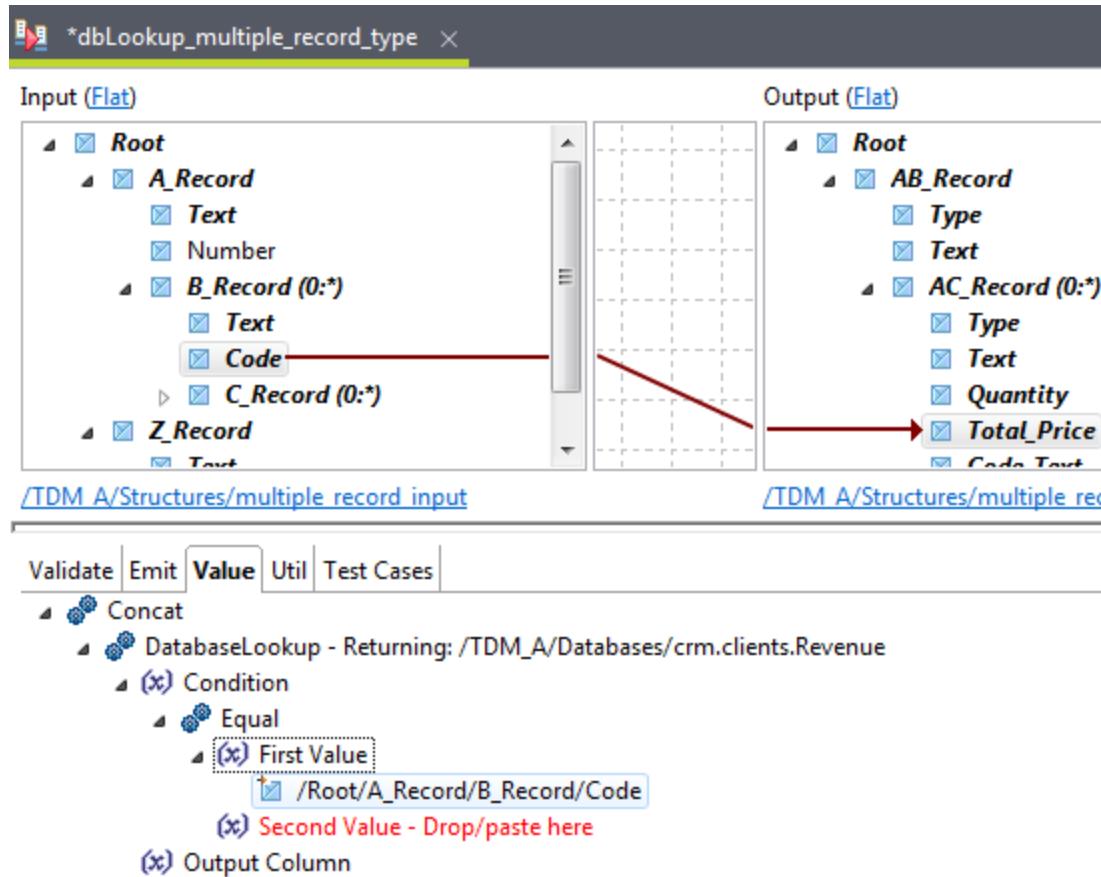


6. Click the ... button next to the **Output Column** field.

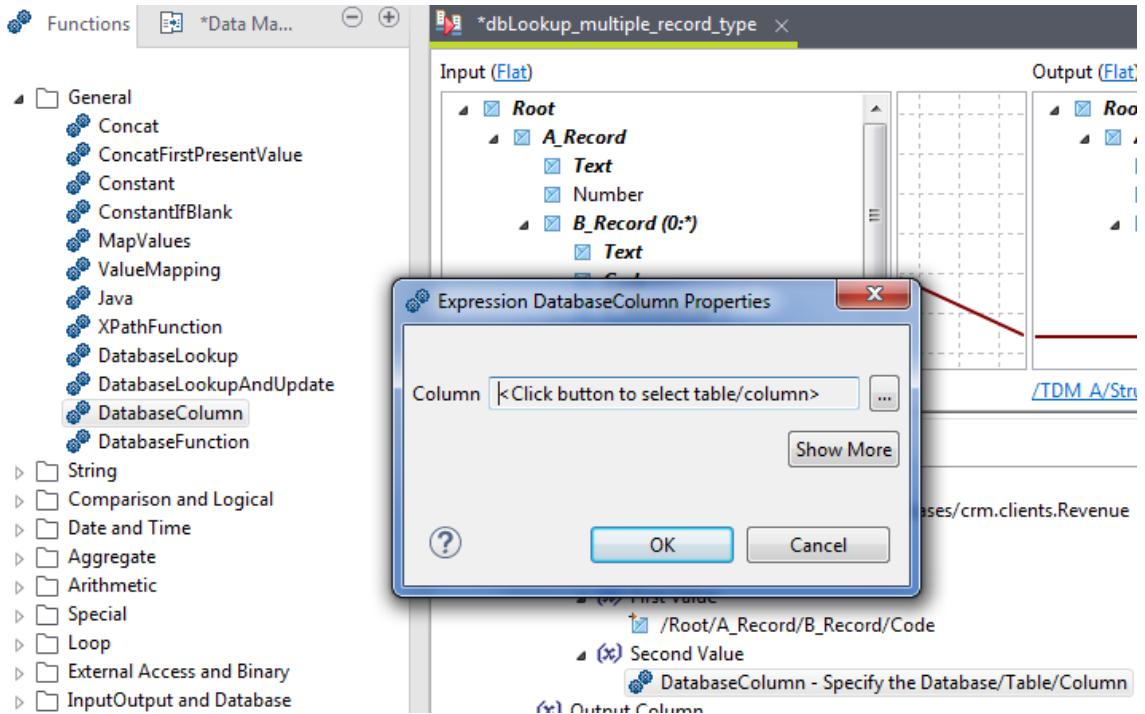
Select the *Revenue* column from the *clients* table in the *crm*, then click **OK** twice to save the configuration.



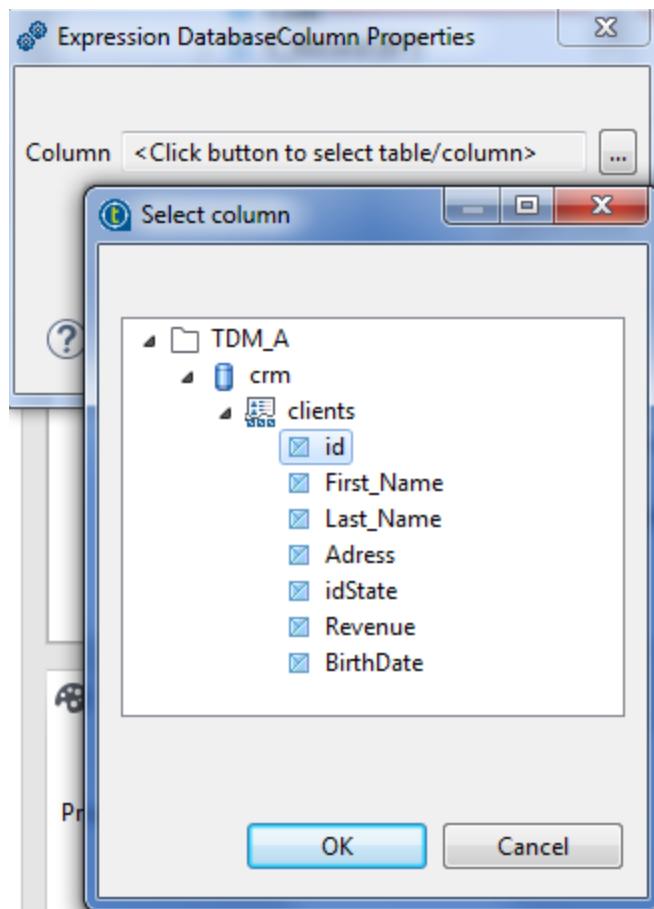
7. Drag and drop a **Functions > Comparison and Logical > Equal** Function to the **Condition** field in the **Value** tab, then drag and drop **Root > A_Record > B_Record > Code** to the **First Value** field of the condition.



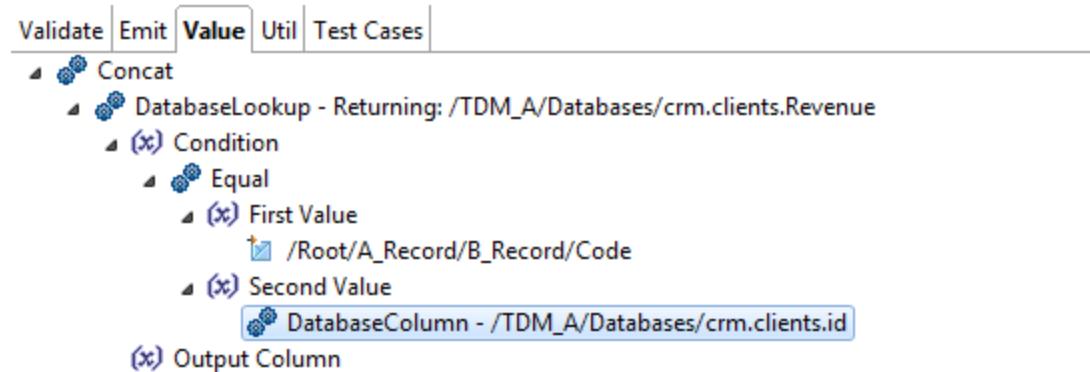
- Drag and drop a **Functions > General > DatabaseColumn** Function on top of the **Second Value** field, then click the ... button next to **Column** to select the actual lookup column.



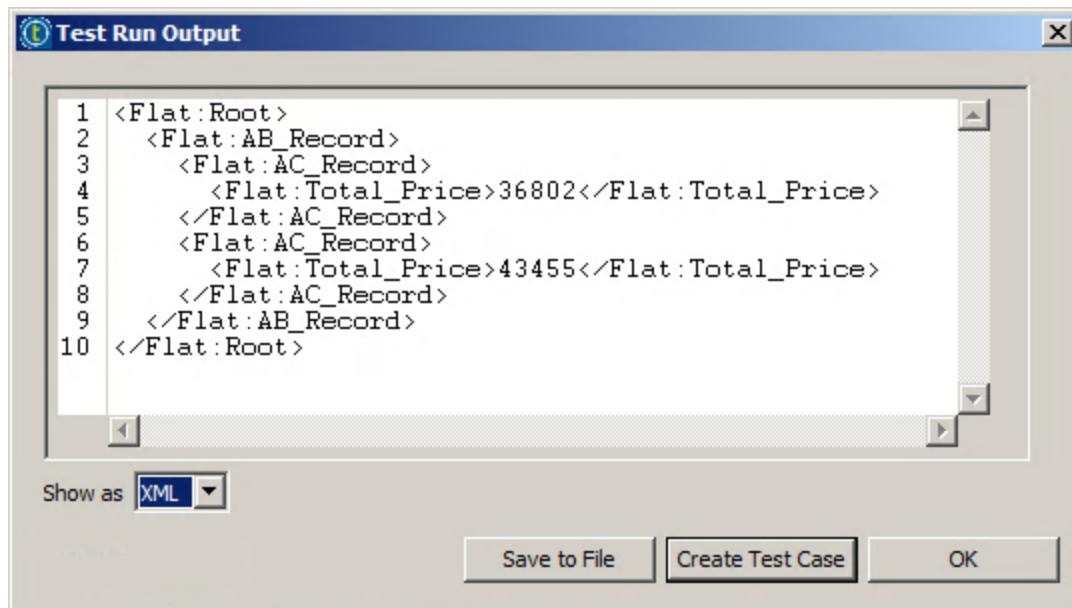
9. Select the *id* column for the lookup and click **OK** twice to save the configuration.



10. The final **Concat** Function should look like the following. This configuration will extract the value of the *Revenue* field where the value of the *id* field is equal to *Code*.



11. Right-click **Total_Price**, select **Test Run** and check that you get the following results. Note that you might have to select the XML representation at the bottom left of the dialog to get the same output.



The screenshot shows a Windows-style dialog box titled "Test Run Output". The main area contains the following XML code:

```
1 <Flat:Root>
2   <Flat:AB_Record>
3     <Flat:AC_Record>
4       <Flat:Total_Price>36802</Flat:Total_Price>
5     </Flat:AC_Record>
6     <Flat:AC_Record>
7       <Flat:Total_Price>43455</Flat:Total_Price>
8     </Flat:AC_Record>
9   </Flat:AB_Record>
10 </Flat:Root>
```

Below the code, there is a "Show as" dropdown menu set to "XML". At the bottom of the dialog are three buttons: "Save to File", "Create Test Case", and "OK".

Next Step

This lesson is almost over. Head to the [Wrap-Up](#) section for a summary of the concepts reviewed in this lesson.

Wrap-Up

In this lesson, you learned how to:

- » Create a database connection and the associated Structures that can be reused from within a Map
- » Compute the value of an Element at runtime, extracting it from a database based on a condition on other Elements

Next Step

Congratulations, you successfully completed this lesson. Click the **Check your status with this unit** button below in order to save your progress. Then click **Completed. Let's continue >** on the next screen to jump to the next lesson.

**This page intentionally left blank to ensure new chapters
start on right (odd number) pages.**