

User-Defined Functions (UDFs) in Informatica Cloud (IICS)

1. Introduction
2. What are User-Defined Functions (UDFs) in Informatica Cloud?
3. How to create User-Defined Functions (UDFs) in Informatica Cloud?
4. How to use User-Defined Functions (UDFs) in Informatica Cloud Mappings?
5. Practical scenarios for the application of User-Defined Functions (UDFs) in Informatica Cloud
 - 5.1. Setting Default Values for Nulls
 - 5.2. Creating a Master Data List for Data Mapping
 - 5.3. Advanced Pattern Matching with Regular Expressions
 - 5.4. Storing Custom Complex Formulae
6. Updating and Deleting User-Defined Functions (UDFs)
7. Difference between User-Defined Functions (UDFs) and Mapplets

1. Introduction

In the process of building mapping components within a project, we encounter scenarios where a complex expression logic is required for data transformation. Informatica provides a feature which allows saving this complex expression logic into a reusable object which can be used across several mappings. This approach not only streamlines development process but also mitigates the need for redundant implementation of the same logic.

In this article, let us understand how to create and utilize this powerful feature called User-Defined Functions (UDFs).

2. What are User-Defined Functions (UDFs) in Informatica Cloud?

User-Defined Functions in Informatica Cloud are reusable functions that can be used in field expressions. User-defined functions allows you to create and store complex transformation logic using built-in functions and reuse them across mappings.

User-Defined Functions can be used in following components:

- Mapping
- Mapplet
- In a Field Expression of a Mapping task
- In another User-defined function

User-Defined Functions are not supported in following components

- In an expression in a Synchronization task.
- Mappings in Advanced Mode.

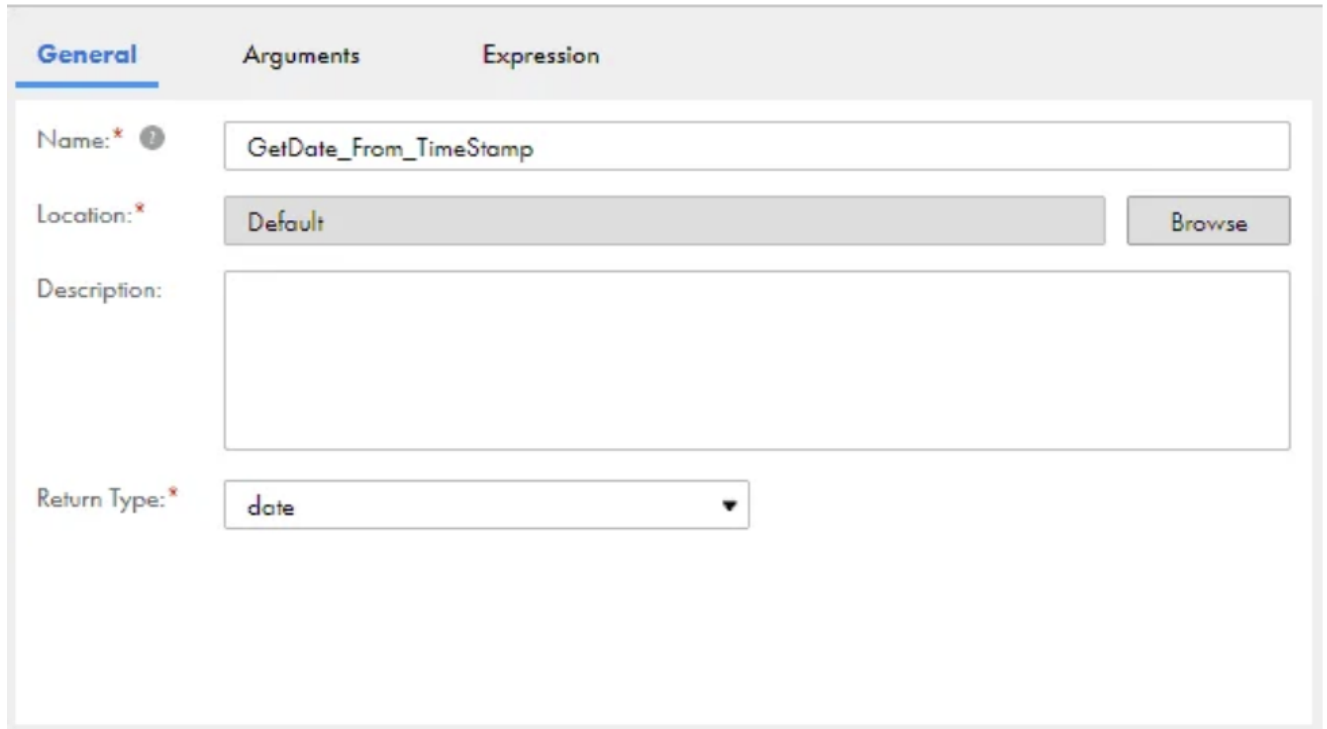
3. How to create User-Defined Functions (UDFs) in Informatica Cloud?

As an example, in this demonstration,
let us create a UDF which can extract the “date” value from a
“timestamp” field.

Follow below steps to create a User-defined function in Informatica Cloud.

1. Navigate to New > Components > User-Defined Function and then click Create.

2. In the **General** tab, enter details of UDF *Name*, *Location* and *Return Type* of value that UDF returns. The supported return types are binary, date, numeric and string.



The screenshot shows a configuration window for a User-defined function (UDF) with three tabs: **General**, **Arguments**, and **Expression**. The **General** tab is active. It contains the following fields:

- Name:** A text box containing "GetDate_From_TimeStamp".
- Location:** A dropdown menu showing "Default" and a "Browse" button.
- Description:** A large empty text area.
- Return Type:** A dropdown menu showing "date".

General tab in User-defined function

3. In the **Arguments** tab, create the *Arguments* to be used in the expression logic of UDF. The user-defined function supports creation of multiple arguments up to 10 in a single UDF component.

GeneralArgumentsExpression

Arguments (1)

+

↓

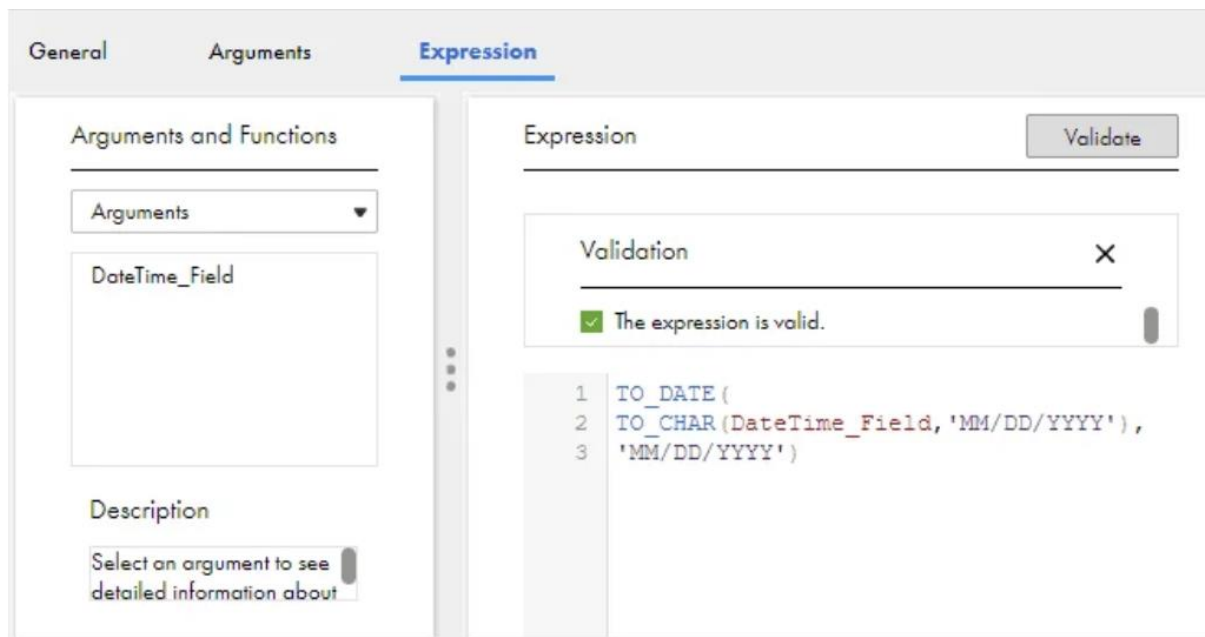
↑

Name	Type	Description
DateTime_Field	date	

Creating Arguments for User-defined function expression

Note that if multiple arguments are created in a UDF, they are passed to the function in the order in which they appear on the Arguments tab. To rearrange the order, select an argument and click Move up or Move down.

4. In the **Expression** tab, create the field expression logic using the built-in functions and arguments created.



Creating Expression logic in User-defined function

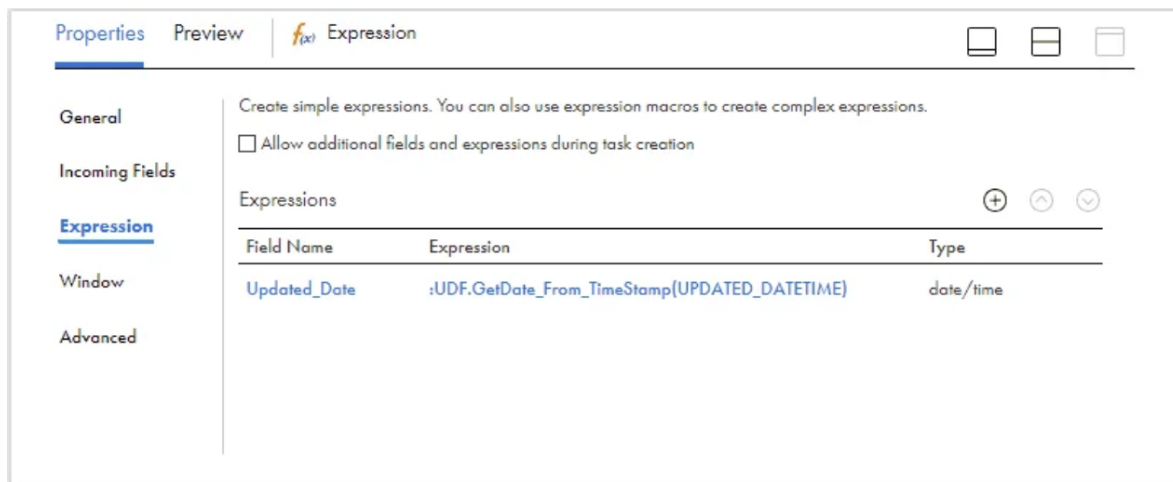
5. Validate and Save the User-defined Function.

4. How to use User-Defined Functions (UDFs) in Informatica Cloud Mappings?

The User-defined function can be used in any field expression of a transformation in the mapping.

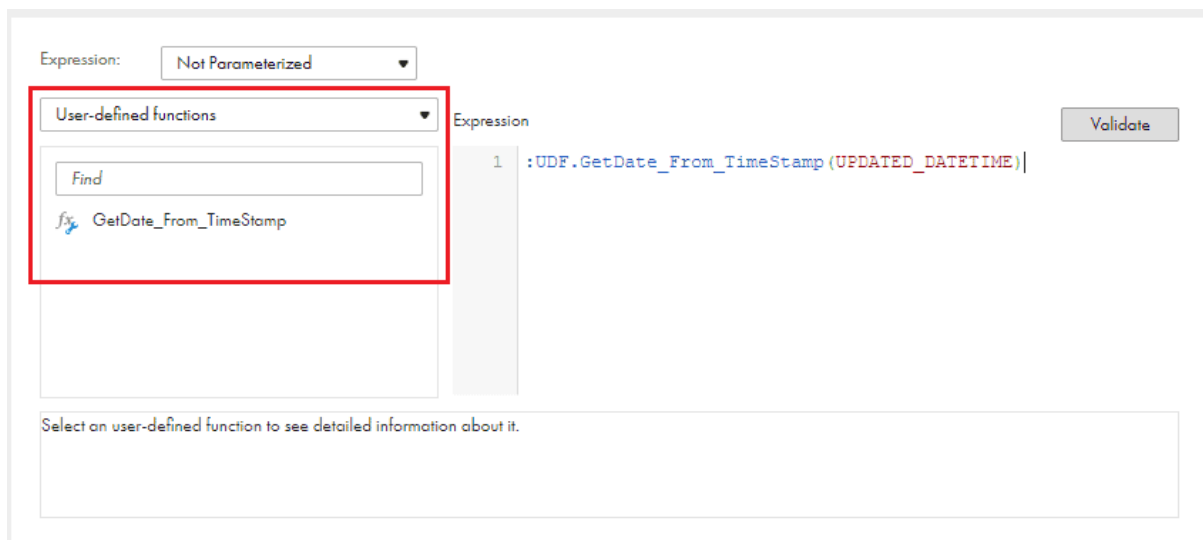
Follow below steps to use User-defined function in a mapping.

1. Create an output/variable field with a data type identical to the return type of the user-defined function.



Creating an output field in Expression to read the output value of UDF

2. In the field expression editor of the transformation, under *User-defined functions*, list of all the valid UDFs will be displayed.



Expression editor showing the list of all available valid User-defined functions

3. Select the required UDF into the expression editor and pass the input fields from upstream transformation as argument to the UDF.


The User-defined function selected in the expression editor should be prefixed with “:UDF.” as shown below.





The below image shows the output of the field *Updated_Date* using the *GetDate_From_Timestamp* user-defined function.

Properties

Preview

 Expression

Run Preview...



Row	EMPLOYEE_ID	NAME	SALARY	UPDATED_DATETIME	Updated_Date
	100	Jennifer	4400.00	10/07/2023 14:43:14	10/07/2023 00:00:00
	101	Michael	13000.00	10/07/2023 14:43:14	10/07/2023 00:00:00
	102	Pat	6000.00	10/07/2023 14:43:14	10/07/2023 00:00:00
	103	Den	11000.00	10/07/2023 14:43:14	10/07/2023 00:00:00

1 - 4 of 4

<

1

>

Items per Page

100

5. Practical scenarios for the application of User-Defined Functions (UDFs) in Informatica Cloud

Below are some of the real-time use cases of UDFs in Informatica Cloud

5.1. Setting Default Values for Nulls

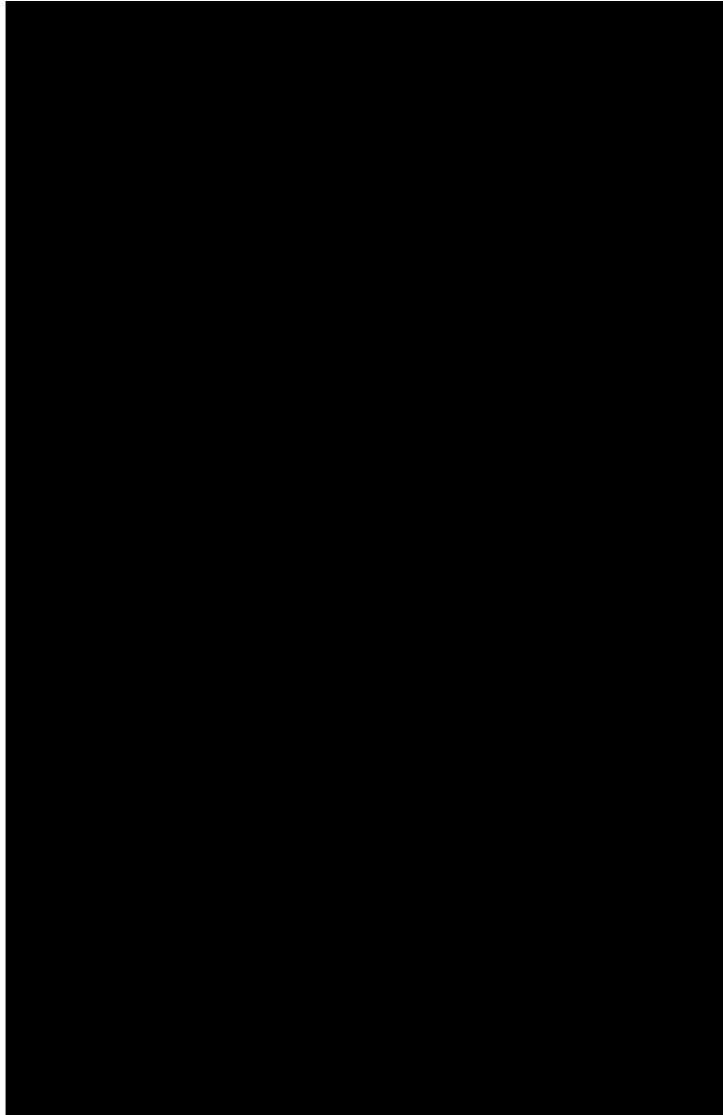
In a real-time data integration scenarios, UDFs can be employed to set default values for missing or null data. This ensures consistency across not null fields as they all are assigned with same default values.



5.2. Creating a Master Data List for Data Mapping

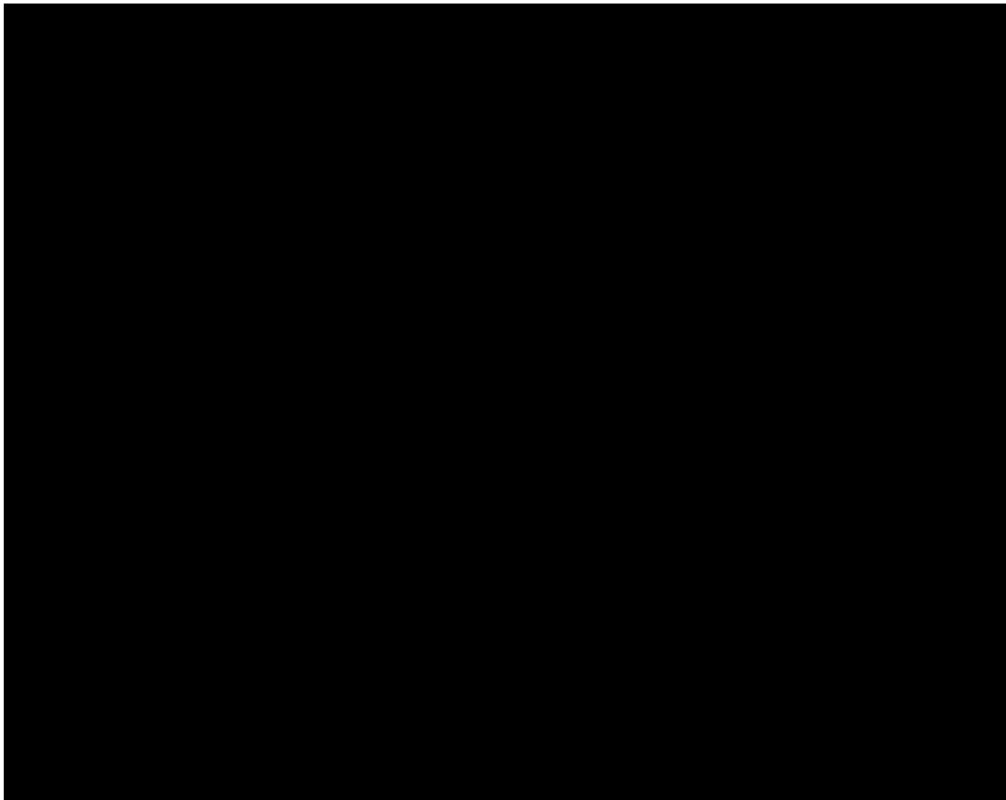
UDFs help in building a Reference Data Repository. For instance, you can construct a repository designed to retrieve currency information by supplying a country code as an argument to UDF. Any addition of new country information into the UDF is easily propagated to all mappings using UDF without the need to edit the mapping.

UDFName: GetCurrency



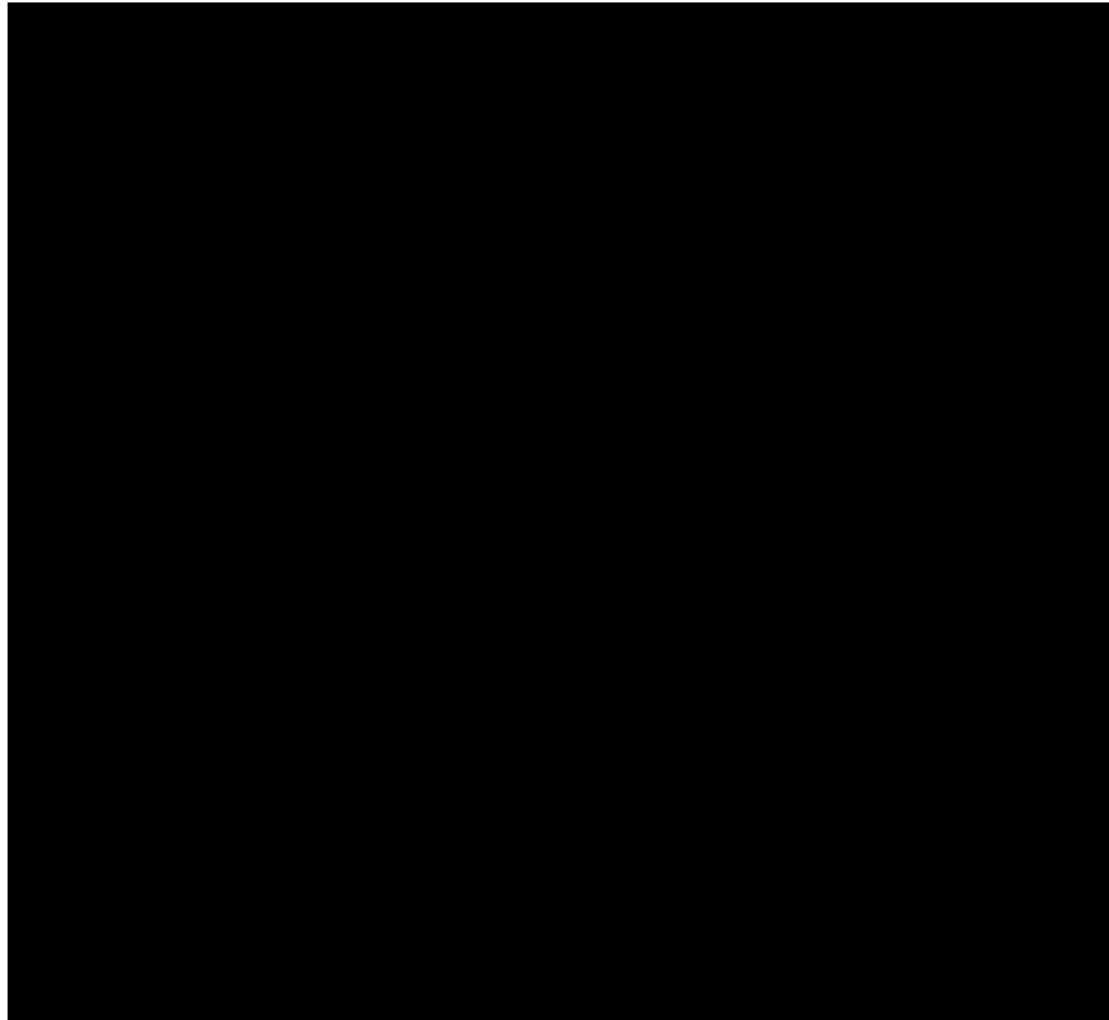
5.3. Advanced Pattern Matching with Regular Expressions

UDFs help in storing complex transformation logic involving Advanced Pattern Matching with Regular expressions. For instance, you can construct a UDF which accepts any field as an argument and removes all special characters from the field value and replace them with null using regular expressions.



5.4. Storing Custom Complex Formulae

UDFs allow you to store and apply unique business rules involving complex custom formulae for calculations. For instance, you might have a unique business rule that calculates monthly mortgage payments based on the principal loan amount, annual interest rate, and loan term. You can encapsulate this complex mortgage calculation logic in a UDF for consistent and real-time application.



6. Updating and Deleting User-Defined Functions (UDFs)

Updating a user-defined function can affect the expressions and functions that use it. Follow below guidelines for updating and deleting User-defined functions.

- If a user-defined function is edited and the updated function is valid, Data Integration propagates the changes to all expressions and user-defined functions that use the function.
- If a user-defined function is edited but the updated function is not valid, Data Integration does not propagate the changes to the expressions and user-defined functions that use it.
- A User-defined function cannot be renamed after it is saved.
- A User-defined function can only be deleted after removing it from all expressions and functions that use it.

7. Difference between User-Defined Functions (UDFs) and Mapplets

Both User-Defined Functions (UDFs) and Mapplets are reusable components in Informatica Cloud for data transformation and processing. But they serve a different purpose and have own distinct characteristics.

- UDFs are primarily used for defining custom data transformation logic in field expressions. They allow you to encapsulate specific data manipulation logic in a reusable function that can be applied within a mapping or across different mappings.
- [Mapplets](#) contain reusable transformation logic that can contain multiple transformations, source definitions, and target definitions. They are designed to encapsulate a set of transformations for reuse across multiple mappings.