

The background of the slide features a dark blue field filled with glowing green circuit traces and white dots, resembling a complex digital network. Overlaid on this are various sequences of binary code (0s and 1s) in a lighter blue, semi-transparent font, some of which are slightly blurred to create a sense of depth and motion.

Semarchy

xDI DEV

E Hierarchical Files



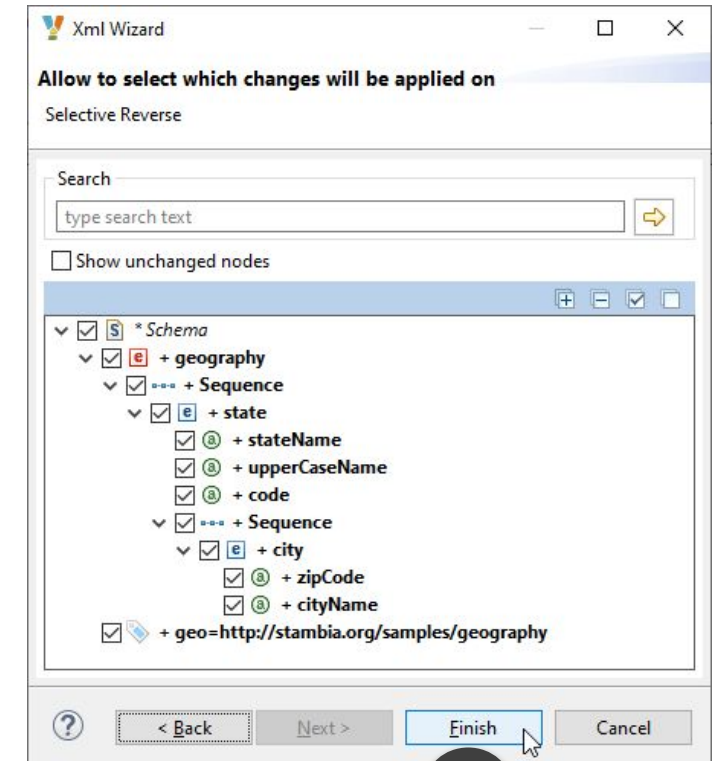
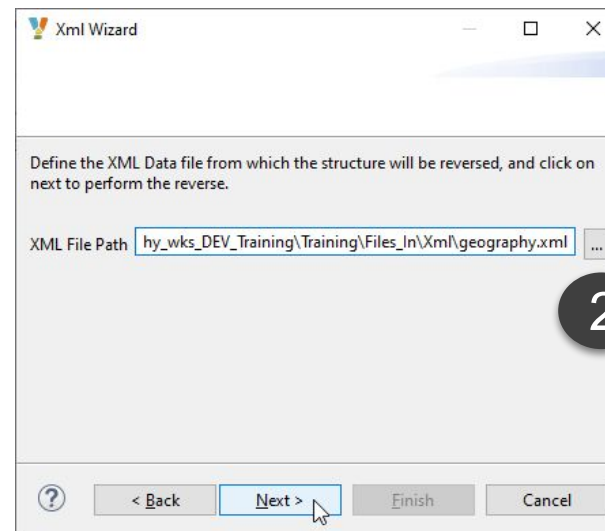
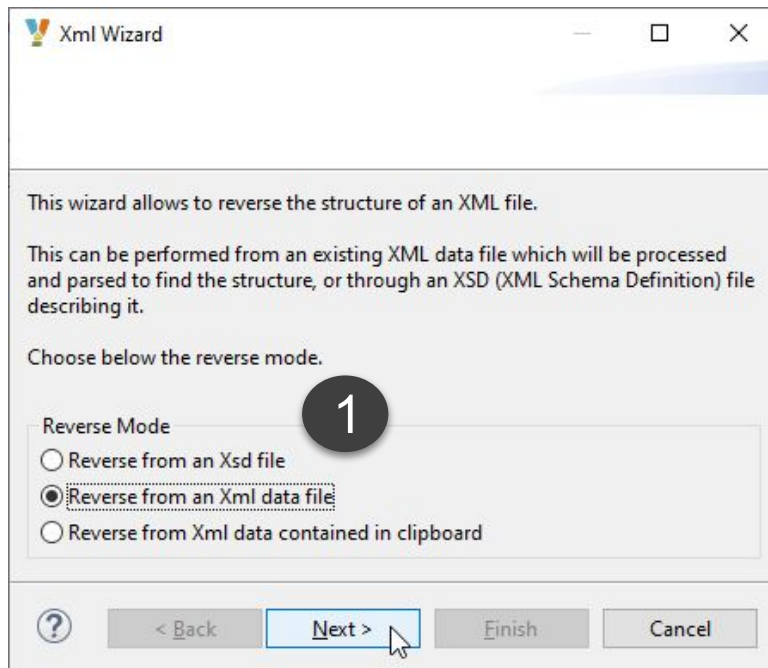
xDI DEV

E Hierarchical Files

E1 - Flat file, Xml & Json Metadatas

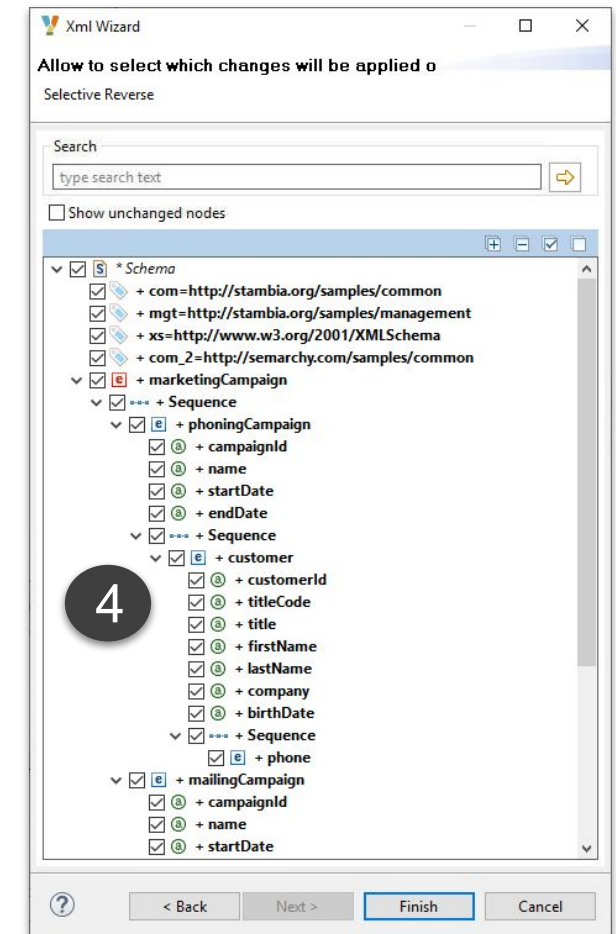
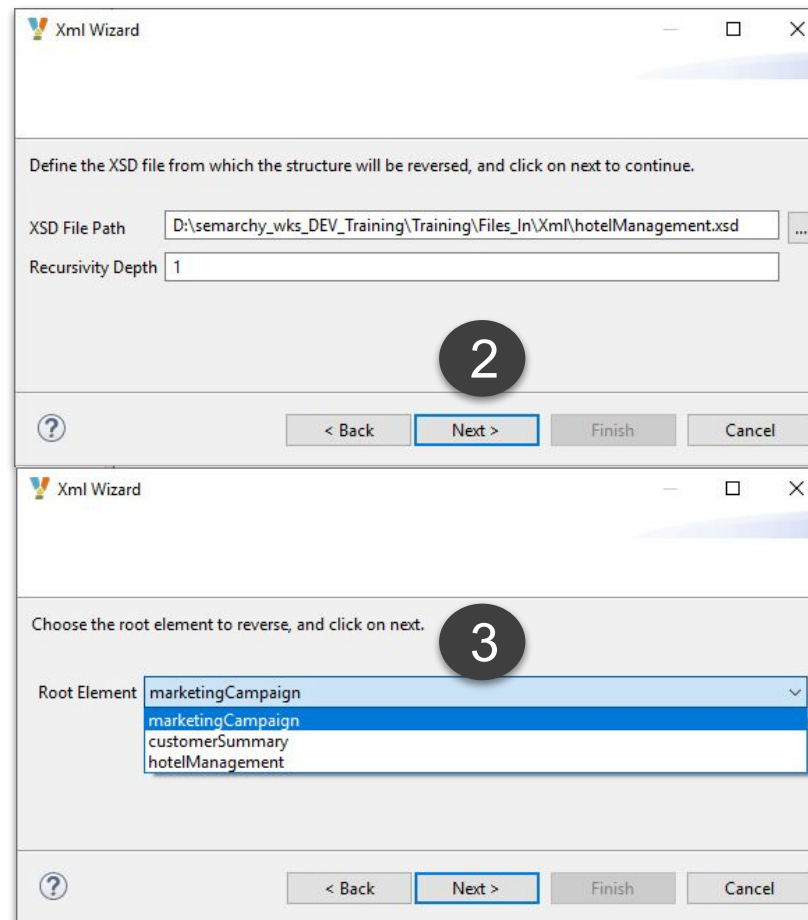
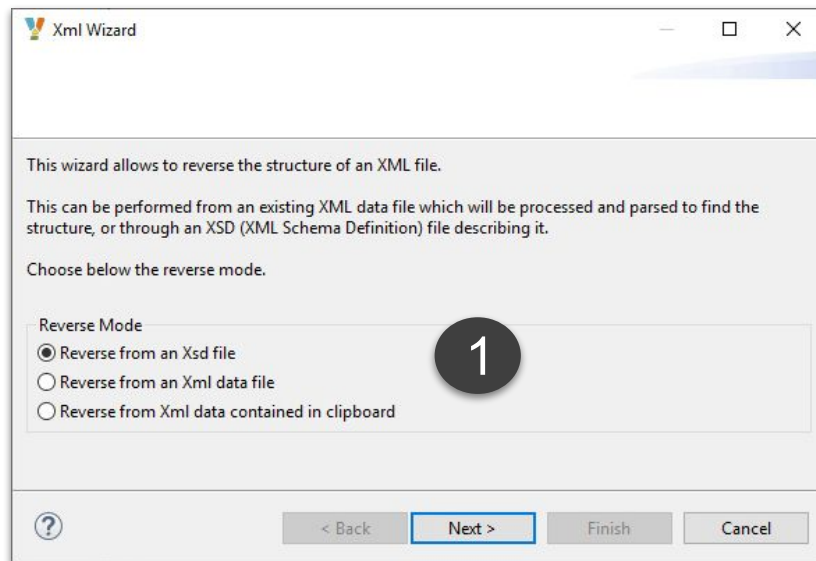
Xml & Xsd

The reverse of an XML file description can be based on a sample XML file

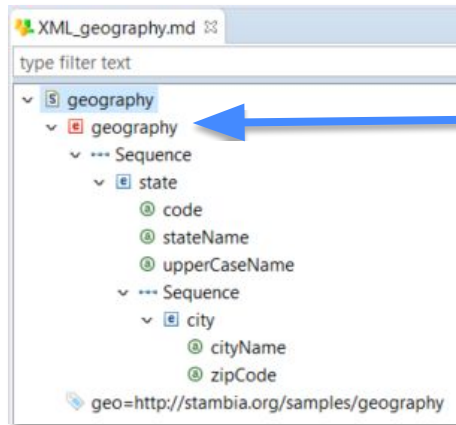


Xml & Xsd

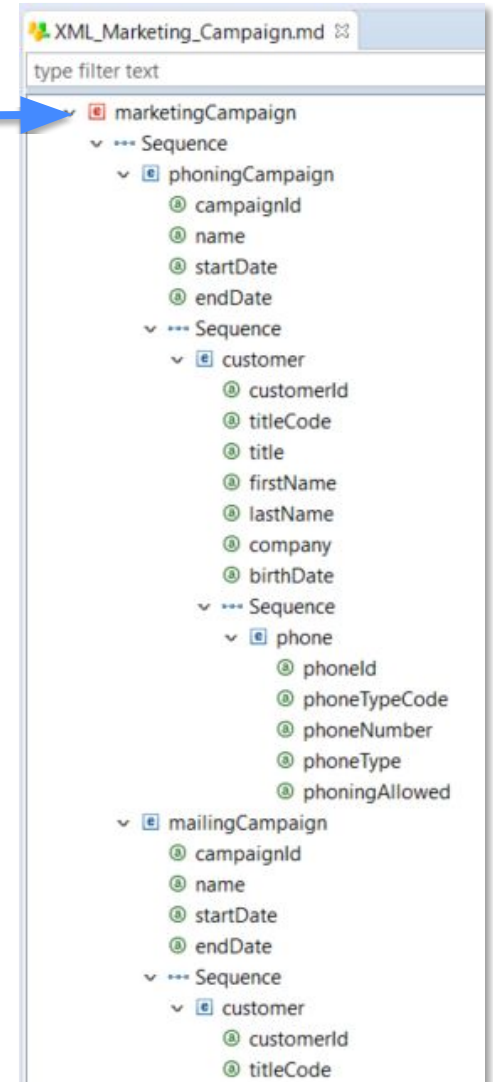
The reverse of an XML file description can be based on an XSD file (XML Schema Definition)



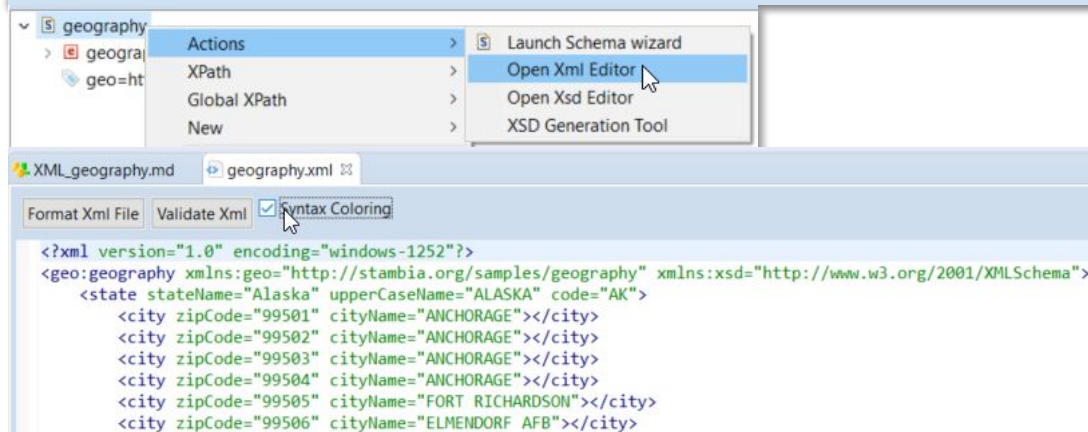
Xml & Xsd - The generated Xml Metadata



The root element (in red) must be drag & dropped in a mapping to be used



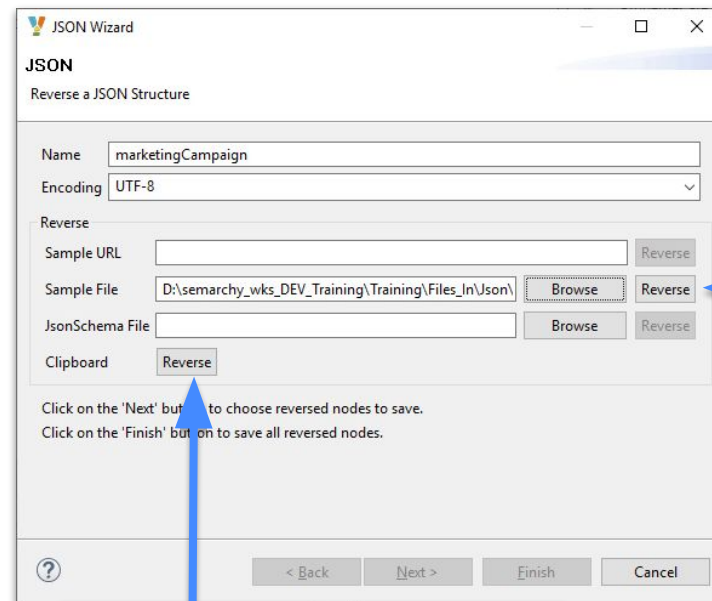
It's also possible to consult the content of a Xml file



JSON

The reverse of a JSON file is always based on a sample JSON file

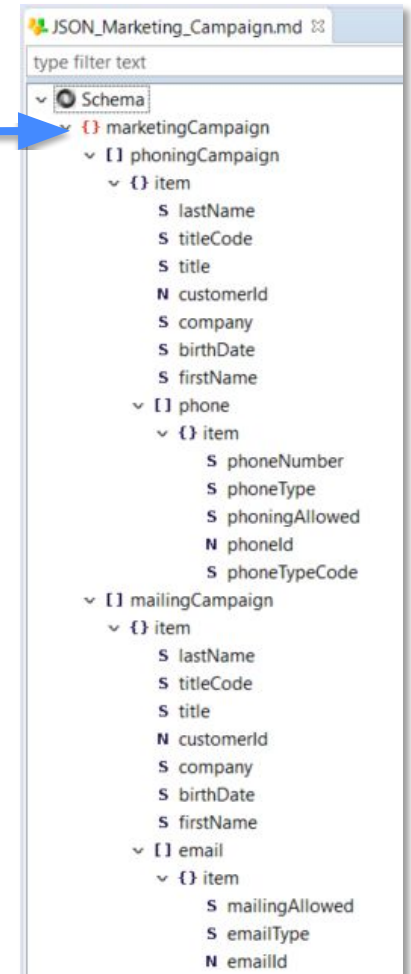
The generated metadata



The main bracket (in red) must be drag & dropped in a mapping to be used

In the other case, use the « Browse » button

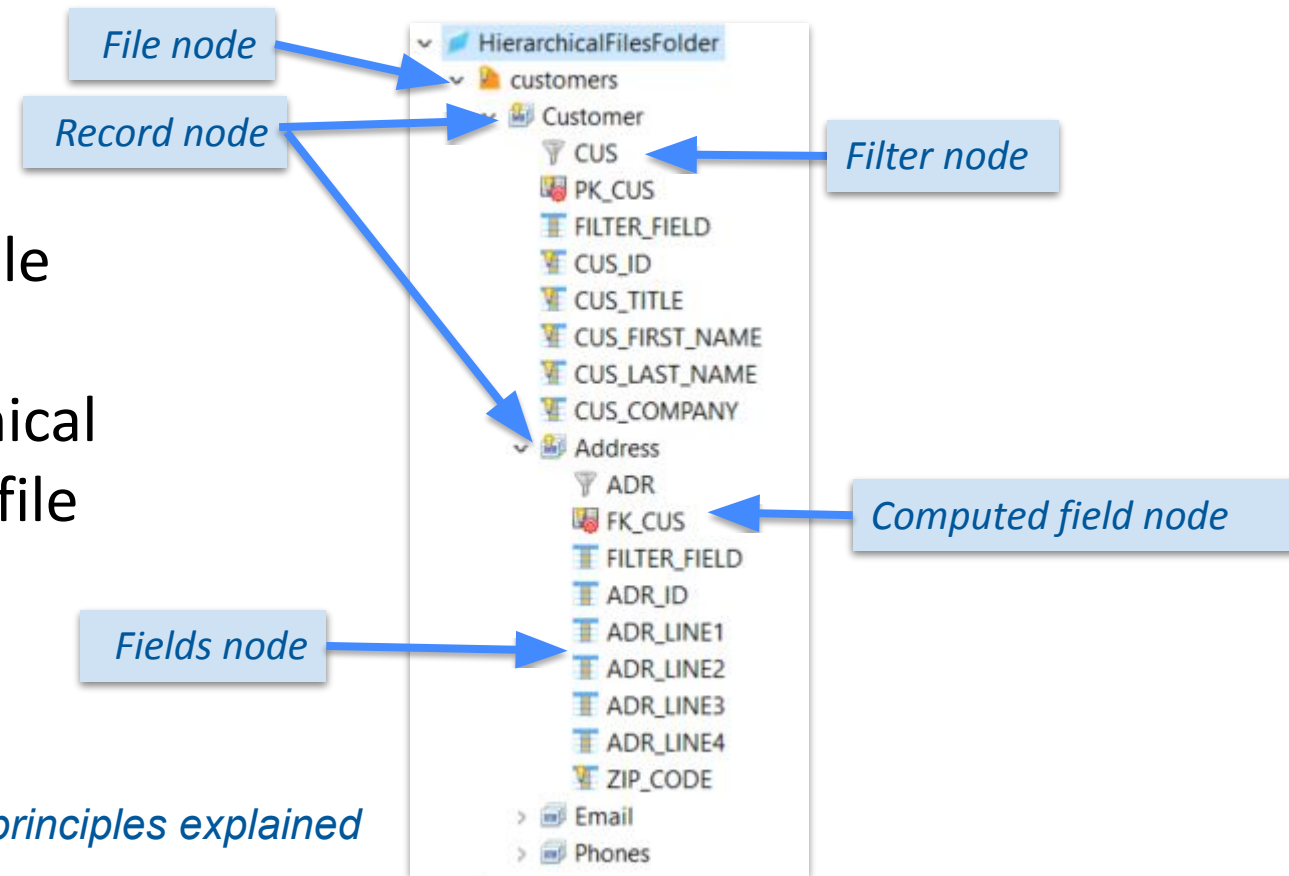
If the sample json is in clipboard, use directly the « Reverse » button



Flat hierarchical file

A hierarchical flat file is a flat file with records nodes between the file and the fields nodes

- A hierarchy between the records is possible
- Filters and Computed fields can be added
- A simple flat file can be used as a hierarchical file if there is a record node between the file node and the field nodes



In the slides hold in the “Write in hierarchical files” chapter, the principles explained can be applied to Json and Flat hierarchical files



xDI DEV

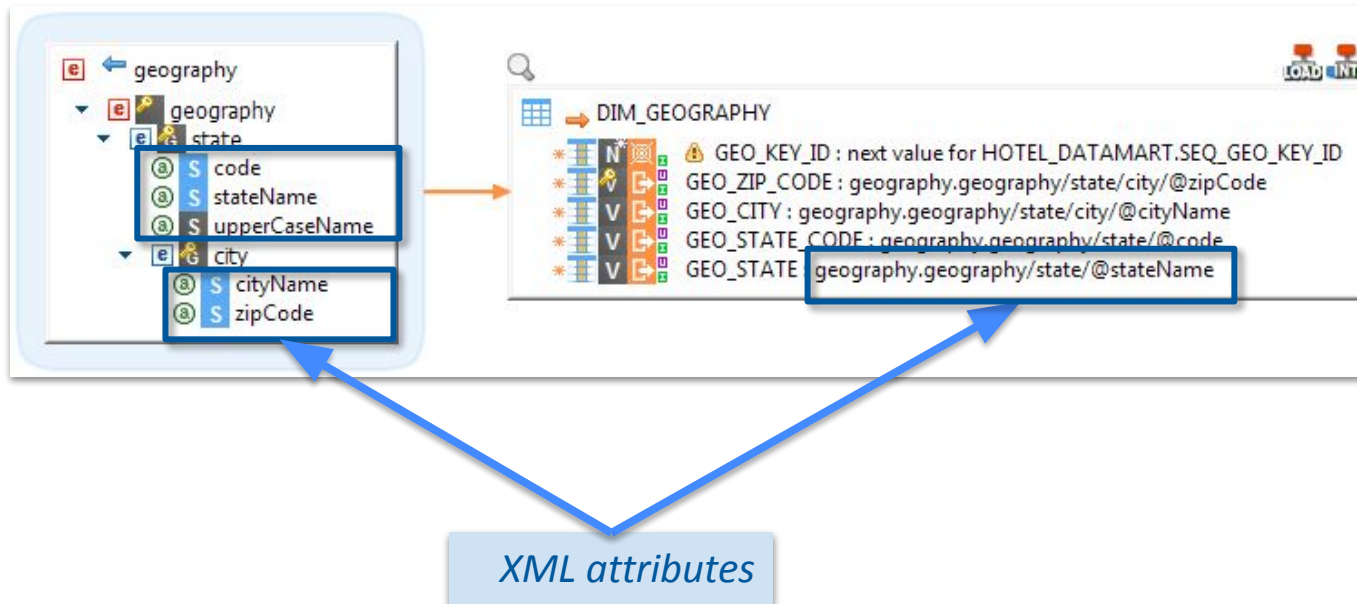
E Hierarchical Files

E2 - Read hierarchical files

Reading XML files

You can read an XML file in a mapping in the same way as you read a normal flat file

- Just drag & drop XML attributes





Practice exercise

Reverse an XML metadata
Create a first mapping with an XML source

T ogether
E veryone
A chieves
M ore





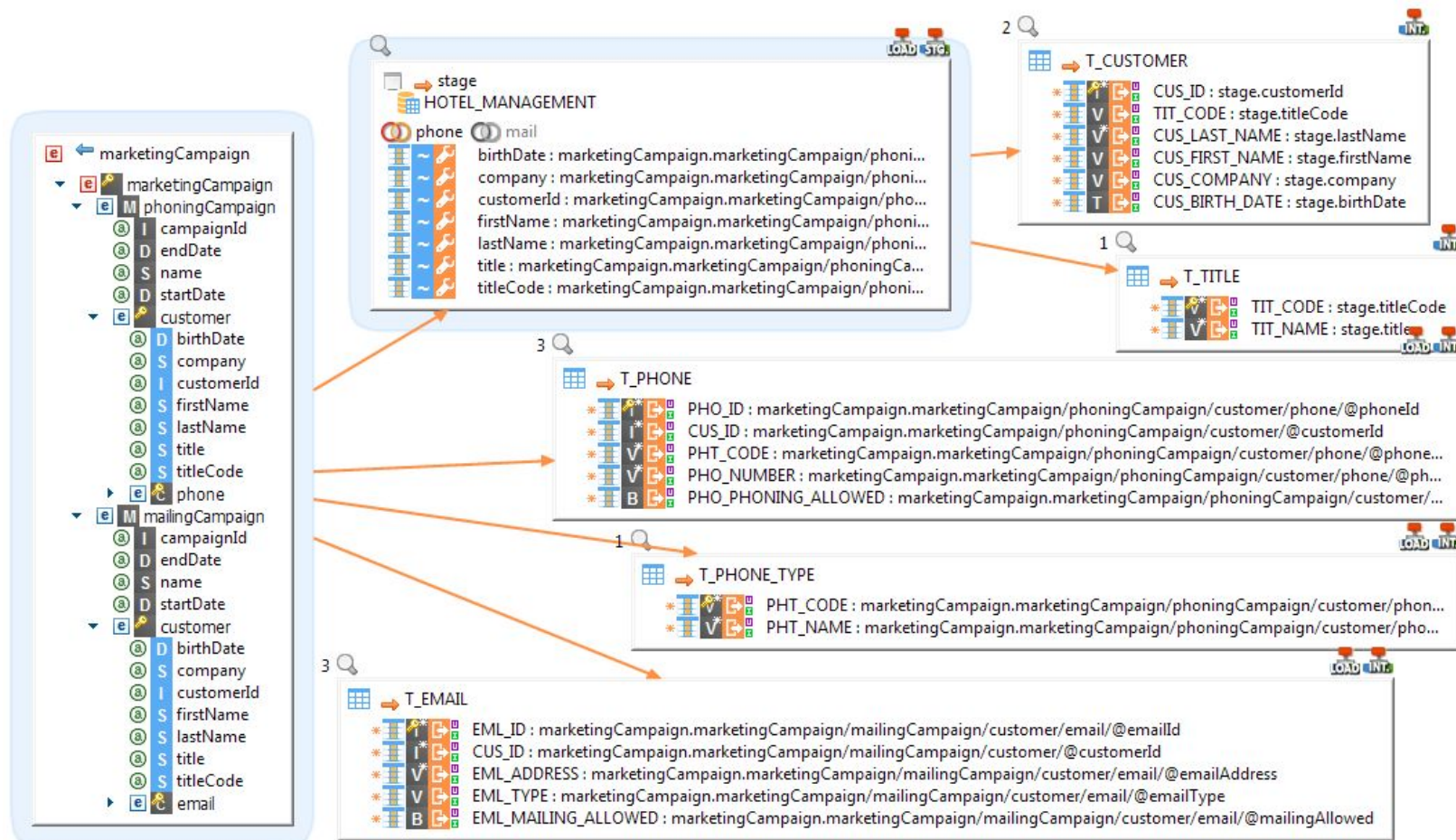
Demo

Reverse engineering Json & hierarchical file
Create a mapping with an XML source



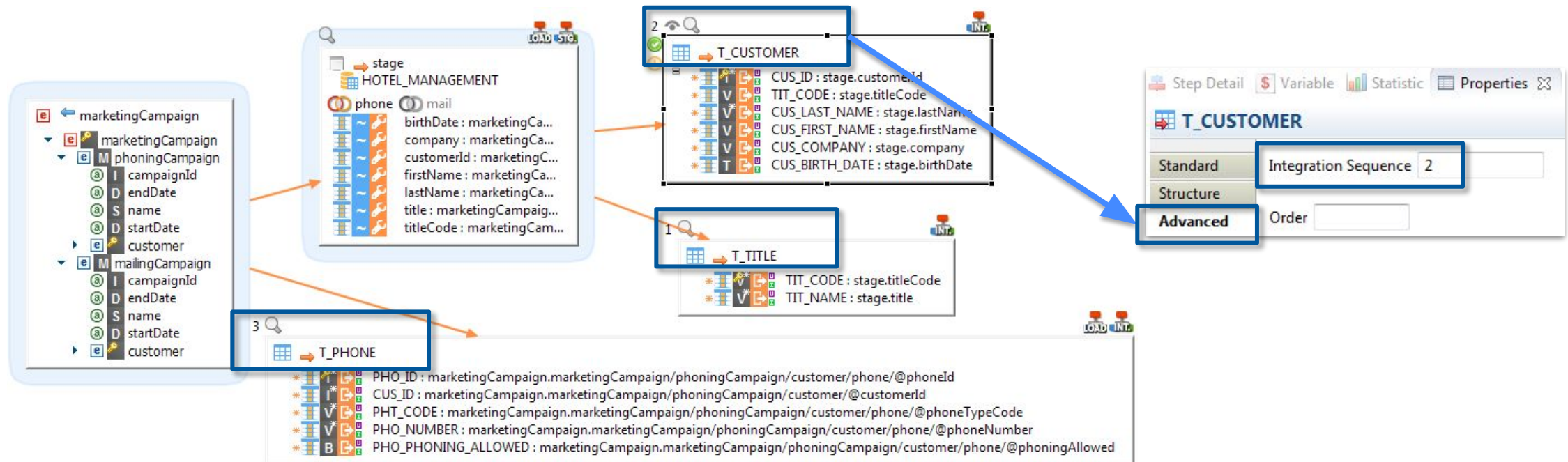
Load XML files in multiple tables

The load template will load the data into tables and keep the relationships



Load XML files in multiple tables

To be able to load the multiple tables in an particular order (foreign keys), you will have to complete an “integration sequence” property in the “Advanced” tab



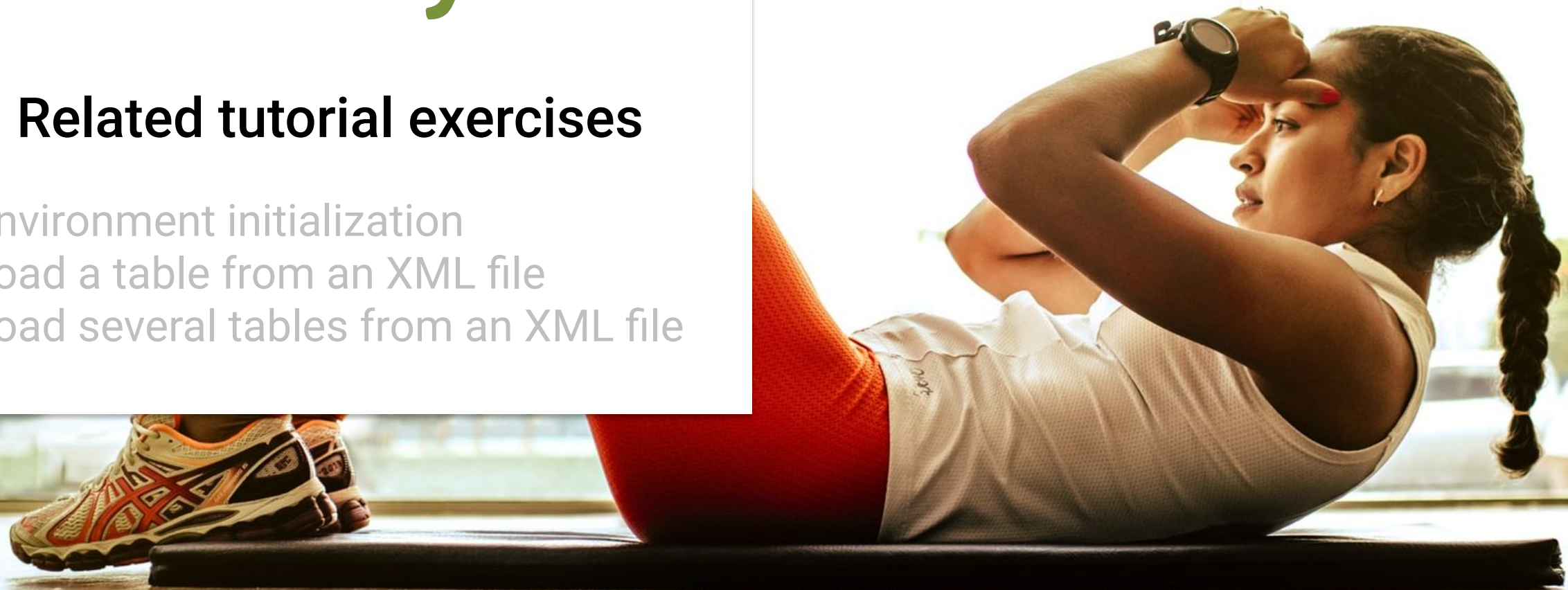


Related tutorial exercises

Environment initialization

Load a table from an XML file

Load several tables from an XML file





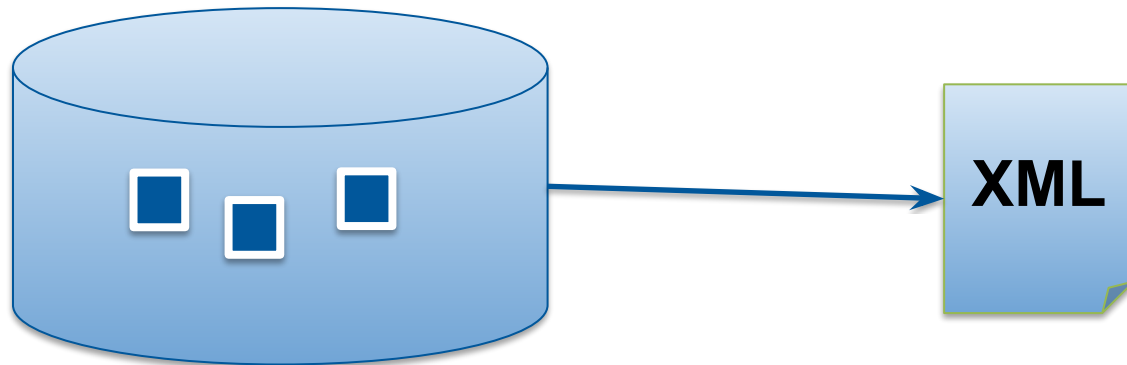
xDI DEV

E Hierarchical Files

E3 - Write hierarchical files

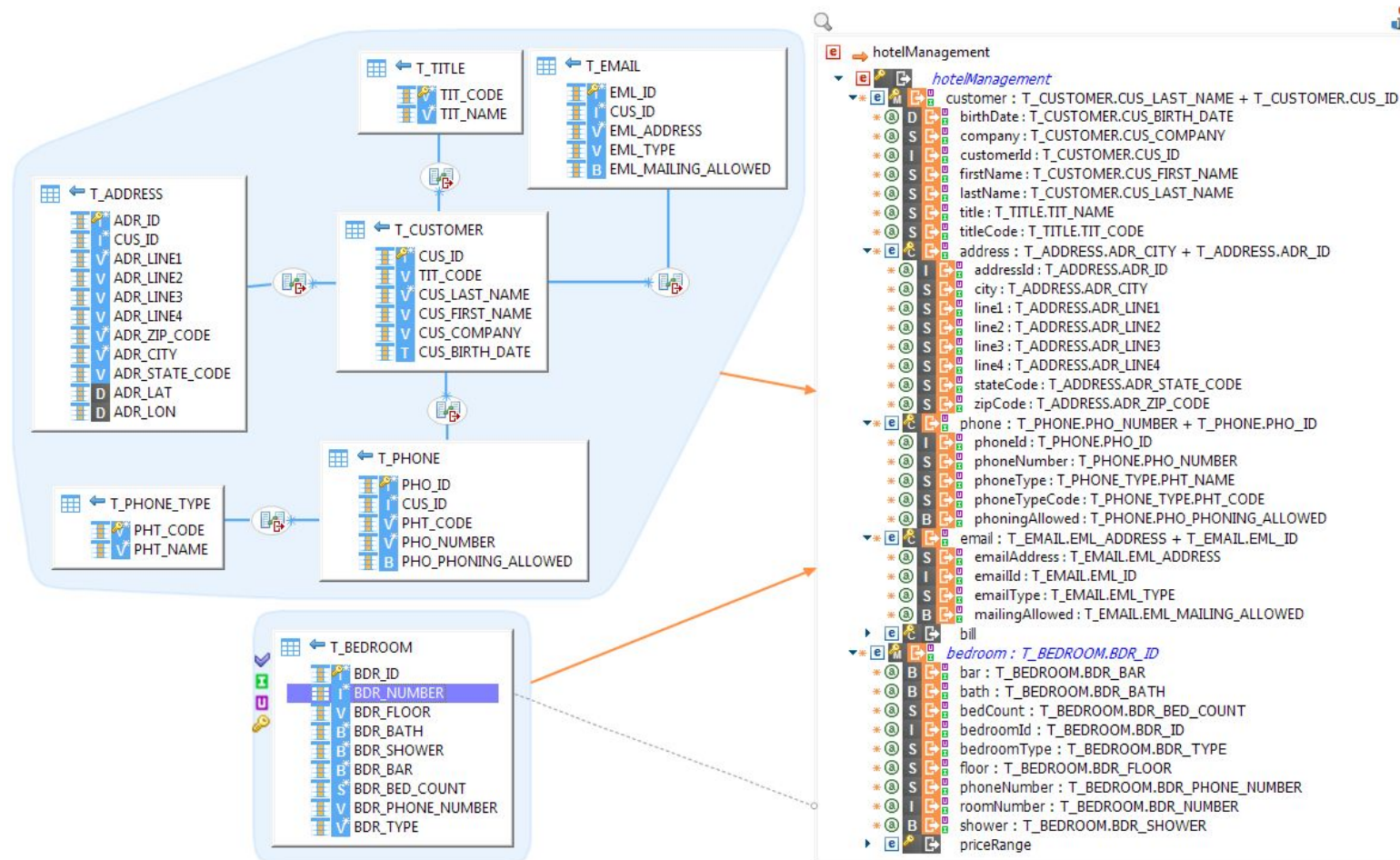
Writing XML files

You can write data into XML files in the same way as you write in a table
The Staging Area (thus the transformations) will be on the source



XML Mappings

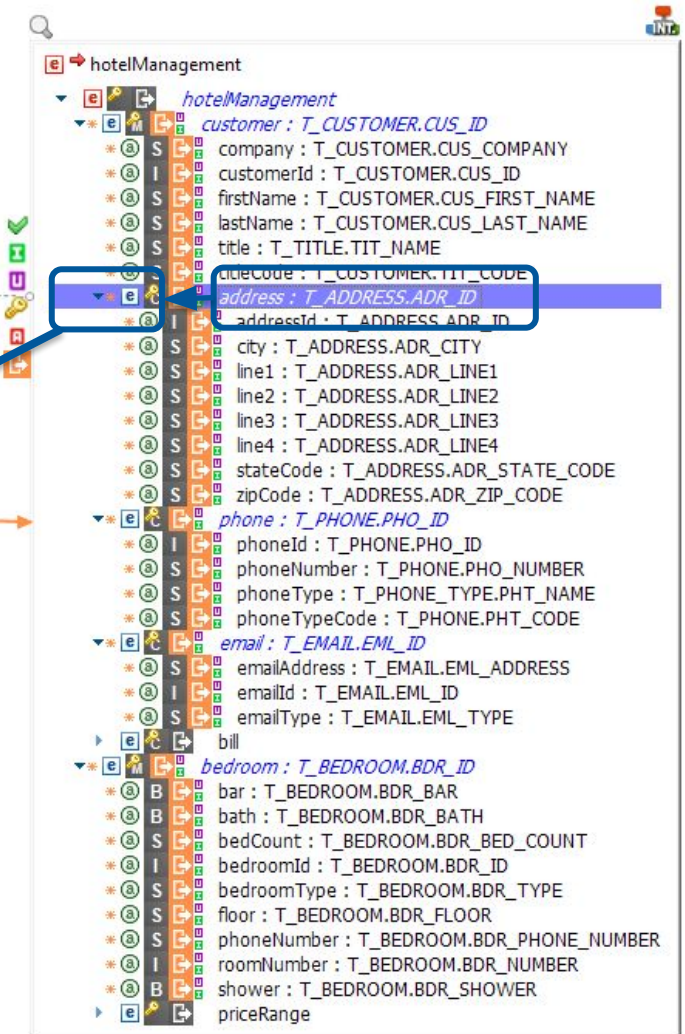
An XML Mapping is as easy to realize as the Mapping of a table



Particularities

In hierarchical files (XML, JSON, flat files), repetition blocks (occurrences on elements) are symbolized by key icons

- The expression used on a key will define the repetition (group) key for the data
- It will also define the sort





Demo

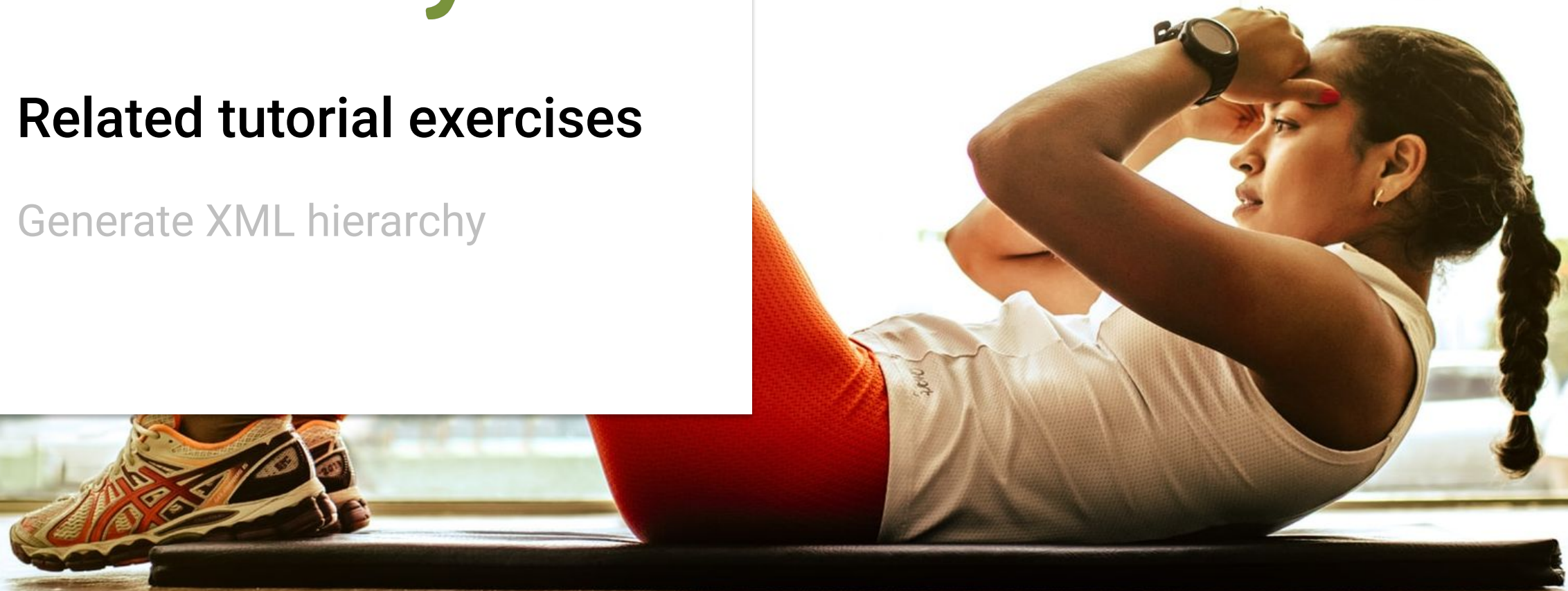
Create a mapping to generate XML file





Related tutorial exercises

Generate XML hierarchy



Writing multiple files

In 3 steps, it is possible to write several files using a single mapping

The image shows three screenshots from the Semarchy software interface, illustrating the steps to write multiple files using a single mapping.

Step 1: The screenshot shows the mapping editor for the 'hotelManagement' mapping. The root element is 'hotelManagement : T_ADDRESS.ADR_CITY'. The 'iterate' expression is specified on the root element, as indicated by the callout: *Specify the iterate expression on the root element*.

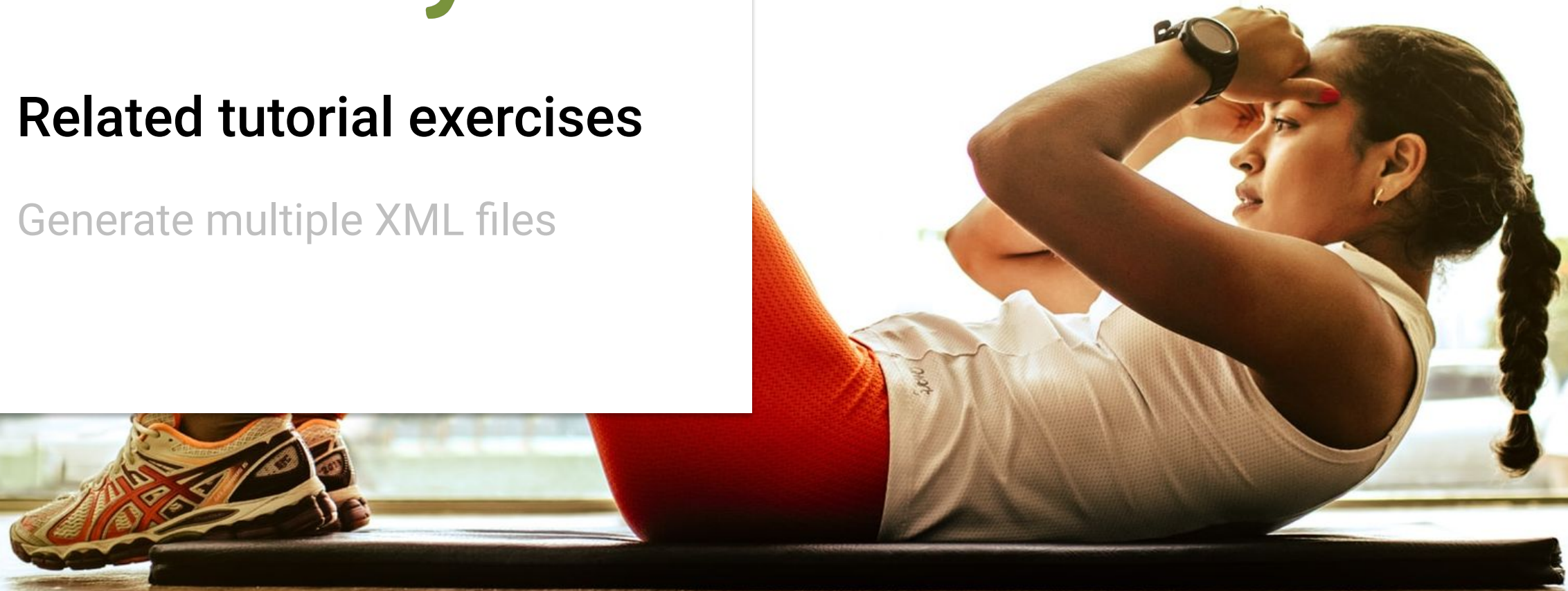
Step 2: The screenshot shows the 'Field city' configuration panel. The 'Tags' section is highlighted, showing the 'CITY_NAME' tag. The callout states: *Define the "Tags" to be used in the integration template (Out File Mask)*.

Step 3: The screenshot shows the 'Action Process INTEGRATION Rdbms to Xml' configuration panel. The 'Out File Mask' property is defined as 'States/[STATE]/customerDetails_[CITY].xml', as indicated by the callout: *Define the Out File Mask property of the Integration Template*.



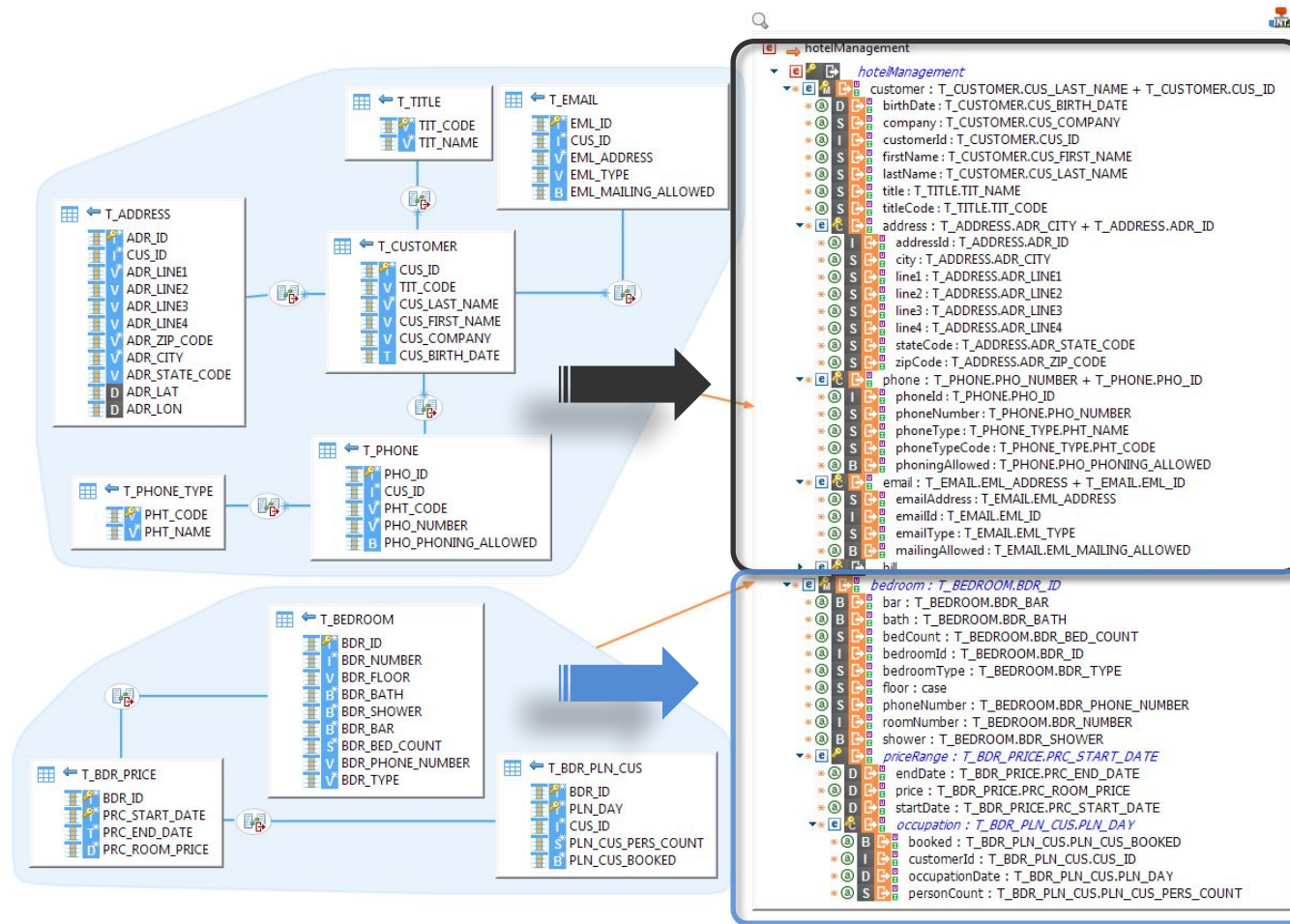
Related tutorial exercises

Generate multiple XML files



Writing into multiple hierarchies

It is possible to write into multiple and distinct Xml hierarchies

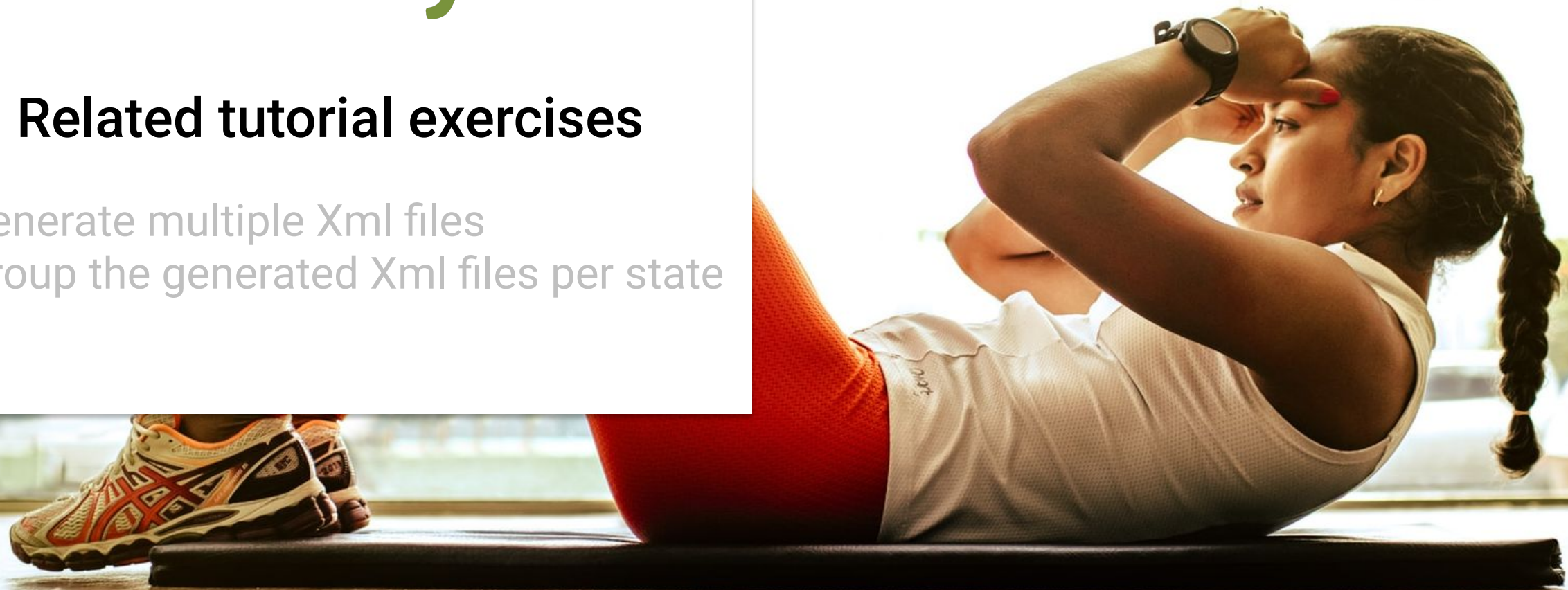




Related tutorial exercises

Generate multiple Xml files

Group the generated Xml files per state



To go further

Document Type	Link
English video Read and write hierarchical files	https://www.youtube.com/watch?v=SIEJSO9780I
Stambia.org article Using XML property fields	https://stambia.org/doc/66-technology-articles/xml/425-using-xml-property-fields
Stambia.org article Reading XML nodes with varying names	https://stambia.org/doc/66-technology-articles/xml/427-reading-xml-nodes-with-varying-names
Stambia.org article Reading multiple XML files	https://stambia.org/doc/66-technology-articles/xml/429-reading-multiple-xml-files
Stambia.org article Validating an XML file with an XSD	https://stambia.org/doc/66-technology-articles/xml/187-validating-an-xml-file-with-an-xsd
Stambia.org article Indenting an XML file using an XSL transformation	https://stambia.org/doc/66-technology-articles/xml/411-indenting-an-xml-file-using-an-xsl-transformation
Stambia.org article Sorting data when writing an XML file	https://stambia.org/doc/66-technology-articles/xml/449-sorting-data-when-writing-an-xml-file

A large crowd of people is shown from behind, with their hands raised in the air, suggesting a concert or a large gathering. The scene is dimly lit, with some light reflecting off the raised hands and the crowd's hair. The background is dark, and the overall atmosphere is one of excitement and participation.

Semarchy

Questions?