



# Leading the **Integration** Revolution

---

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



ADEPTIA

Use Case: Creating an output with Header and Detail records

# Use Case: Creating an output with Header and Detail records

Input

PO1, Detail1  
PO1, Detail2  
PO2, Detail1  
PO3, Detail1  
PO3, Detail2  
PO3, Detail3



Convert →

Output

PO1  
- Detail1  
- Detail2  
PO2  
- Detail1  
PO3  
- Detail1  
- Detail2  
- Detail3

# Go to Data Interface

Go to Solutions > Data Interfaces and setup a new data interface.  
Example shown below.

**Edit Data Interfaces: PurchaseOrder\_Data\_Conversion**  

Name \* PurchaseOrder\_Data\_Conversion

Description \* Convert CSV into XML with Header and Detail records

Source

Trigger\* PurchaseOrder\_Data\_Trigger (poll for a file in a certain ...) New Edit

My source layout is Text Schema

My source layout details PurchaseOrder\_Source\_Schema (CSV or a tab delimited file) New Edit

Target

Convert source to this target layout XML Schema

My target layout details PurchaseOrder\_Target\_Schema (Target schema with header and. New Edit

My target location details PurchaseOrder\_Target (generated XML file placement ...) New Edit

Mapping

Use this mapping for data conversion PurchaseOrder\_map (apply mapping rules to conver...) New Edit

Save Save As

# Data Mapping

Follow the steps as explained in the video

The screenshot displays the Data Mapper application interface. The top toolbar includes icons for Data Mapper, XSL, and Debugger. The current element is identified as 'Transaction'. The main workspace is divided into two panels: 'Structure\_of\_source\_XSD' and 'Structure\_of\_destination\_XSD'. The source structure is a 'PurchaseOrder\_Source\_Schema' with a 'Record (F)' element containing various fields like 'recordNumber', '\_OrderNumber (string) (M)', 'Document\_Date (date) (M)', 'CustomerID (string) (M)', 'CustomerDUNS (string) (M)', 'SupplierID (string) (M)', 'SupplierDUNS (string) (M)', 'TotalAmount (integer) (M)', 'PaymentDetails (string) (M)', 'LineNumber (integer)', 'Action\_Status (string)', 'Material (string)', 'TotalLineQTY (integer)', 'UOM (string)', 'ScheduleID (integer)', and 'OpenQTY (integer)'. The destination structure is a 'PurchaseOrder\_Target\_Schema' with an 'Exercise2' element containing a 'Transaction (F, LV)' element. This 'Transaction' element has three sub-elements: 'HeaderLevel' (containing 'OrderNumber (M)', 'DocumentDate (M)', 'CustomerID (M)', 'CustomerDUNS (M)', 'SupplierID (M)', 'SupplierDUNS (M)', 'TotalAmount (M)', and 'PaymentDetails (M)'), 'LineLevel (F)' (containing 'LineNumber (M)', 'ActionStatus (M)', 'Material (M)', 'TotalLineQTY (M)', and 'UOM (M)'), and 'ScheduleLevel'. Blue arrows indicate the mapping from the source fields to the corresponding fields in the destination structure. Below the workspace, a menu bar includes 'Math', 'String', 'Date', 'Aggregation', 'Conditional', 'Axis', 'Boolean', 'Context', 'XSL Template', and 'DB'. The bottom panel shows 'Mapping Rules' with 'XPath' set to '/Exercise2/Transaction' and 'ForEach' set to '\$Input\_PurchaseOrder\_Source\_Schema/Root/Record[ generate-id( ) = generate-id(key ('orderKey', \_OrderNumber))]'.



Thank you!