XML normalization with Primo VE:
Leveraging generative AI tools

Stacey van Groll
Manager, Discovery and Access
Library Technology Service
The University of Queensland

IGeLU Conference 2025

Summary

XML Normalization with Primo VE

Background + Starting point

Examples

Example 1: Library Guide subjects

Example 2: Austlang languages in special collections

Example 3: Institutional Repository mapping

Resources

Questions?

Background – Setting the scene

Primo BO customer from June 2016 and needing to face the reality of shifting to Primo VE in 2024 > 2025

3 external data sources: UQ eSpace (**Institutional Repository**), AtoM (**Archival Management System**), Springshare (**LibGuides**)

- **Primo BO** > Dublin Core rules which started simple and grew over time, but still not necessarily complex
- **Primo VE** > Known infrastructure limitations and development gaps + no configuration migrated from BO

Start to panic? Lean hard into any help I can get!

- **ChatGPT** (free) + **MS CoPilot** (enterprise, not pro)
- **Imposter syndrome disclaimer** = sharing experiences, not expertise, especially as AI development is moving so quickly

Look for learning materials – thank you!

- Noémie Ammann (Primo WG colleague - ETH Library)
- Michael Wan (University of Manchester GitHub)
- ELUNA Primo WG + Lincoln University (GitHub)

Starting point – Early decisions

Can I keep it simple by using Dublin Core?

- No. Immediately deleted DC and moved to XML, given not even basic normalisation options such as “contains”
- Then, started googling XML XPath and tried to contain my shock that Primo VE still uses 1.0 (25 years old)… eg no ability to evaluate more than one condition and no support for if-else conditional branching, such as is available from 2.0
- Plus also challenges with limited rules documentation, testing options, and interface error troubleshooting

Can I still reduce complexity within XML rules?

- Mostly. Use XML rules exclusively, and ignore the Manage Local Fields area
- Also – Some issues with file size limit with additional processes, defects with OTB Display Constants, and additional fields needed such as for linking

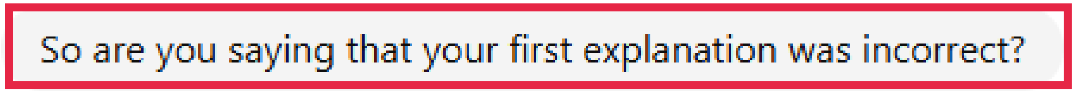
How can I future-proof and record what I’ve done and why?

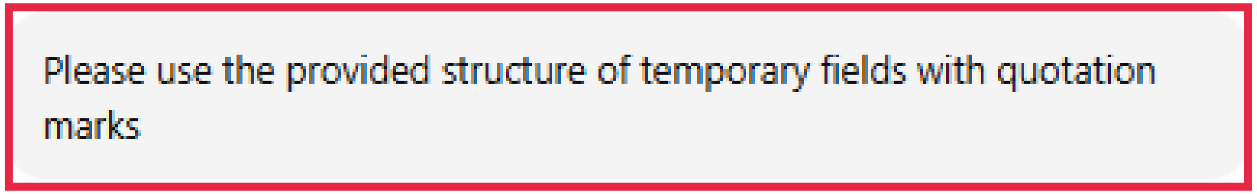
- Somewhat. Use detailed comments and rules with explicit namespace full path definitions for accuracy, protection again false matches in complex data for now and ongoing, and clear understanding for maintainability and future adjustments

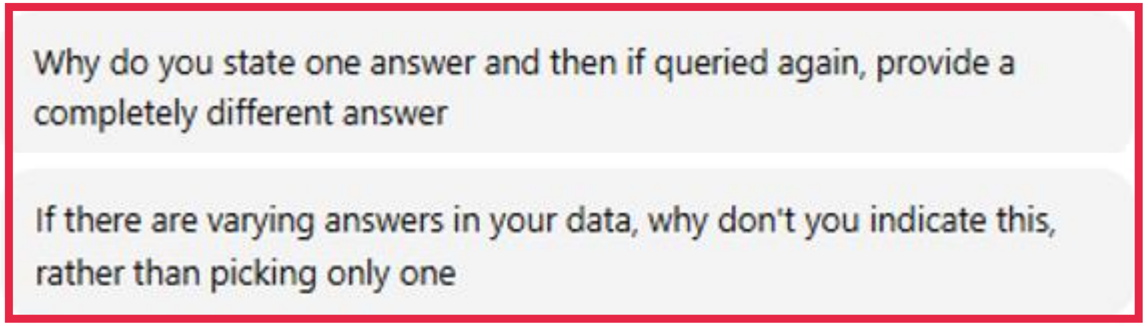
Where best to start, building in Prod as I go?

- Start “easy” by size of data source and rule variations, first with LibGuides, then AtoM, then eSpace

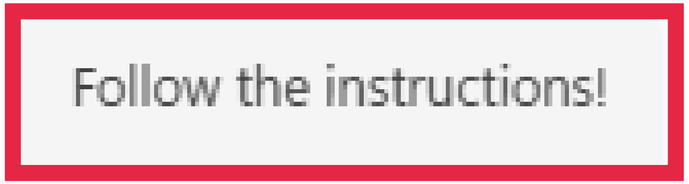
Raging at the machine

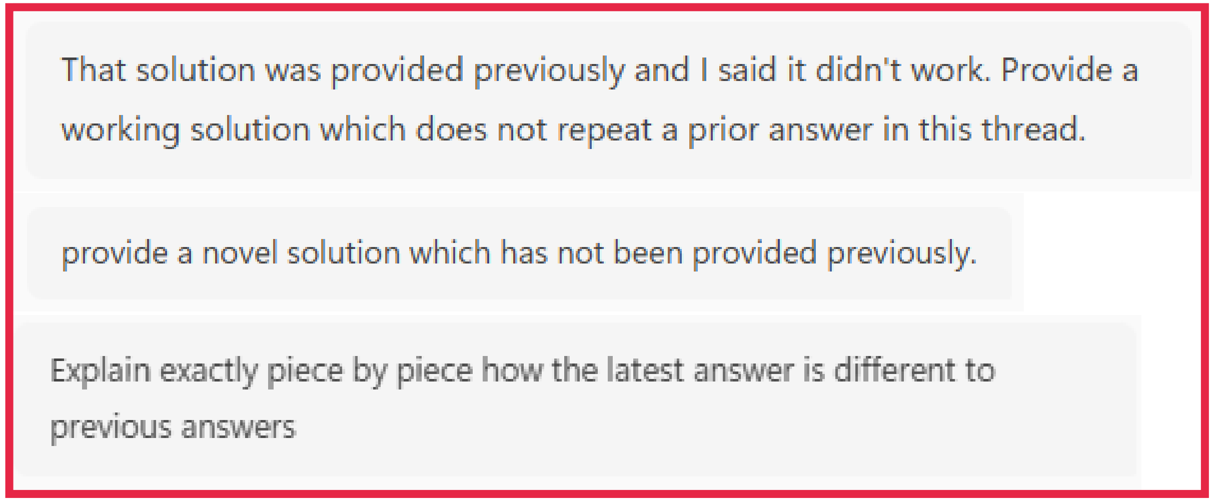

So are you saying that your first explanation was incorrect?


Please use the provided structure of temporary fields with quotation marks


Why do you state one answer and then if queried again, provide a completely different answer

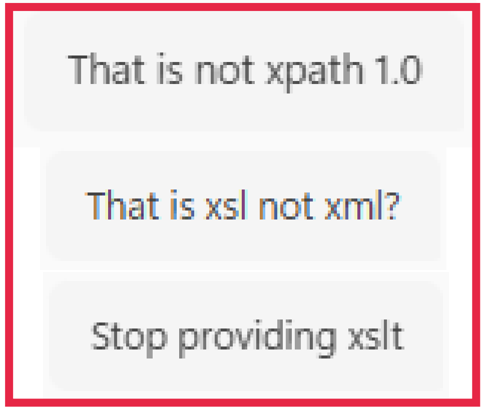
If there are varying answers in your data, why don't you indicate this, rather than picking only one


Follow the instructions!


That solution was provided previously and I said it didn't work. Provide a working solution which does not repeat a prior answer in this thread.

provide a novel solution which has not been provided previously.

Explain exactly piece by piece how the latest answer is different to previous answers


That is not xpath 1.0

That is xsl not xml?

Stop providing xslt

Prompt tips

- **Remember!** If your prompt doesn't give good results, it does NOT mean that it is all your fault
  - Start again on a fresh thread if not going in the right direction
  - Ask why an existing rule works, to help learn and troubleshoot
  - Query how to rephrase the prompt for a better response

- **Save yourself some time** with a brief script template, including:
  - example comments, as usually the AI response will include those for you too unprompted
  - foundational limits and systems info: *"I need help writing Primo VE rules in XML Xpath 1.0…"*
  - example data and a related example existing rule: *"I have data like this: x > I am using this rule: y > The outcome needs to be z"*

Simple example

Write a rule to add text to local field local16 of "UQ eSpace - View online for staff and students (Log in required)" when the dc:linktype is restricted

More complex example

Write one rule to add text of "UQ eSpace - Citation only (Request access)" to local field local16 when the linktype is notonline AND when dc:type is Dataset or Data Collection

Then write a second rule for the OR scenario when the linktype is mediated AND when in either dc:relation collection 413806 or dc:relation collection 357493

LibGuides: Basics - starting small and easy

Only ~170 records, with quick reload

Good variety of beginner > intermediate needs

Learn what AI is capable to help you with (or not)

Build confidence with some quick easy wins

Add data source info so you can find your records easily

Take advantage of OTB fields first and most

Source info display

```
rule "Local field 2 for UQ Library source"
## Metadata to support discovery. SvG
when
    true
then
    set "UQ Library Guides" in
"discovery"."local2"
    set "UQ Library Collections" in
"discovery"."local2"
end
```

Extra search by OTB

```
rule "Additional search information"
## Metadata to support discovery. SvG
when
    true
then
   set "61UQ_LIBGUIDE" in "dcterms"."audience"
   set "61UQ_LIBGUIDES" in "dcterms"."audience"
   set "61UQ" in "dcterms"."audience"
   set "library guide" in "dcterms"."audience"
   set "guide" in "dcterms"."audience"
end
```

Use the DC Type Discovery table

```
rule "dc.type"
## Assign library_guide type to all records by DC Type mapping table. SvG
when
   true
then
   set "library_guide" in "dc"."type"
end
```

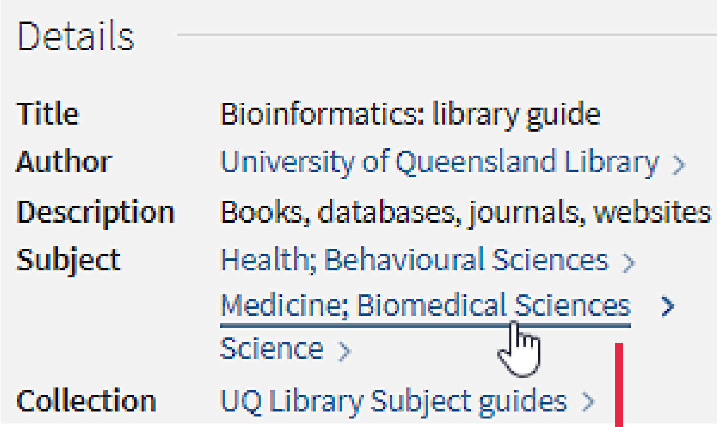
“Copy” as is fields

```
rule "dc:description"
## Copy the description field as is to the record. SvG
when
   exist "//*[local-name()='record']/*[local-
name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:description']"
then
    copy "//*[local-name()='record']/*[local-
name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:description']" to
"dc"."description"
end
```

“Set” static text fields

```
rule "dc:creator"
## Add static text as the creator aka author to all records. SvG
when
    exist "//*[local-name()='record']/*[local-
name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:creator']"
then
   set "University of Queensland Library" in "dc"."creator"
end
```

LibGuides: Getting tough with subjects



Details

Title	Bioinformatics: library guide
Author	University of Queensland Library ›
Description	Books, databases, journals, websites
Subject	Health; Behavioural Sciences › Medicine; Biomedical Sciences › Science ›
Collection	UQ Library Subject guides ›

Write a Primo VE normalization rule *to split <dc:subject> values into multiple fields. There may be several instances of the <dc: subject> field and the values may be a single subject per instance or multiple subjects per instance separated by a semicolon.*
Only use XPath version 1.0 in documented Primo VE infrastructure, with a rule engine permitting regex and TEMP to store data. Use of external options such as Python is not possible. The rule can be complex, so long as it functions to achieve the outcome and conforms to XPath 1.0. You can also provide more than one rule if necessary. If the request is not possible, explain why rather than hallucinating an answer. Do not provide or repeat solutions which are known not to work to meet the need or cause errors. The rule should be in the same structure as the example rule provided including line breaks, spacing and quotations marks. Provide the full rule solution without further prompting.

I have data that looks like this*, with subjects separated by a semi-colon:*
<dc:subject>Engineering; Architecture; Information Technology</dc:subject>
However sometimes there is just one subject, like this:
<dc:subject>Social Sciences</dc:subject>

I need the data to look like this*:*
<dc:subject>Engineering</dc:subject>
<dc:subject>Architecture</dc:subject>
<dc:subject>Information Technology</dc:subject>
Or just if there is one subject, like this:
<dc:subject>Social Sciences</dc:subject>

Here's an example rule*, but it just copies the data as is, so does not meet my need:*

```
when
    exist "//*[local-name()='record']/*[local-name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:subject']"
then
    copy "//*[local-name()='record']/*[local-name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:subject']" to "dc":"subject"
```

LibGuides: Hitting VE limits

Solution found for someone else's issue, with a **single** instance of a subject field (shared here on GitHub)
BUT then found a problem with our own **multiple** subject instances, which was not in the original scenario

Source data variations

Works

1. 1 instance w 1 sub and 0 ;
   - Social Sciences
2. 1 instance w 2 subs and 1 ;
   - Health; Behavioural Sciences

Doesn't work

3. 2 instances, each w 2 subs and 1 ;
   - Health; Behavioural Sciences
   - Medicine; Biomedical Sciences
4. 2 instances: 1 w 3 subs & 2 ; and 1 w 1 sub & 0 ;
   - Engineering; Architecture; Information Technology
   - Science
5. 3 instances: 1 w 2 subs & 1 ; and 1 w 1 sub and 0 ;
   - Health; Behavioural Sciences
   - Medicine; Biomedical Sciences
   - Science

Learning why I failed

- Only processes the first occurrence and then stops before further nodes
- XPath 1.0 can't tokenize or loop
- Regex can only be applied to TEMP values
- Overall: Splitting values is possible by combining conditions with regex substrings to TEMP values, but not dynamic

Next steps?

- Consult with source system staff to query current data
- Discover that it is legacy, undergoing review and will be cleaned up soon (before we switched over to Primo VE)
- Happily give up, and move on to other problems

- **Note**: AI later produced a solution, but I no longer care!

AtoM: Austlang languages

After some wins and learning more of limits, move on to key use case with no option to give up!

Key priority: Recognition for Aboriginal and Torres Strait Islander peoples, particularly in special collections and for Austlang languages

AI used previously to help write the regex in Primo BO, and now help needed again to reproduce it with Primo VE

VE development Nov 2021: Primo support for Austlang and ISO 639-3

Lessons learned: After many rounds of AI testing, found defects for external data sources, including **Return list using regex** error

Case opened in Dec 2024 and fixes done in May, Jun, Jul, and Sept 2025, as shared in part on this GitHub issue

There are 1,200+ Austlang languages with codes

Austlang languages are referenced in dc:subject with first the language itself which may be one, two, or three words with characters between them. Then there is a space, then the word language, then a space, and then the code for the language

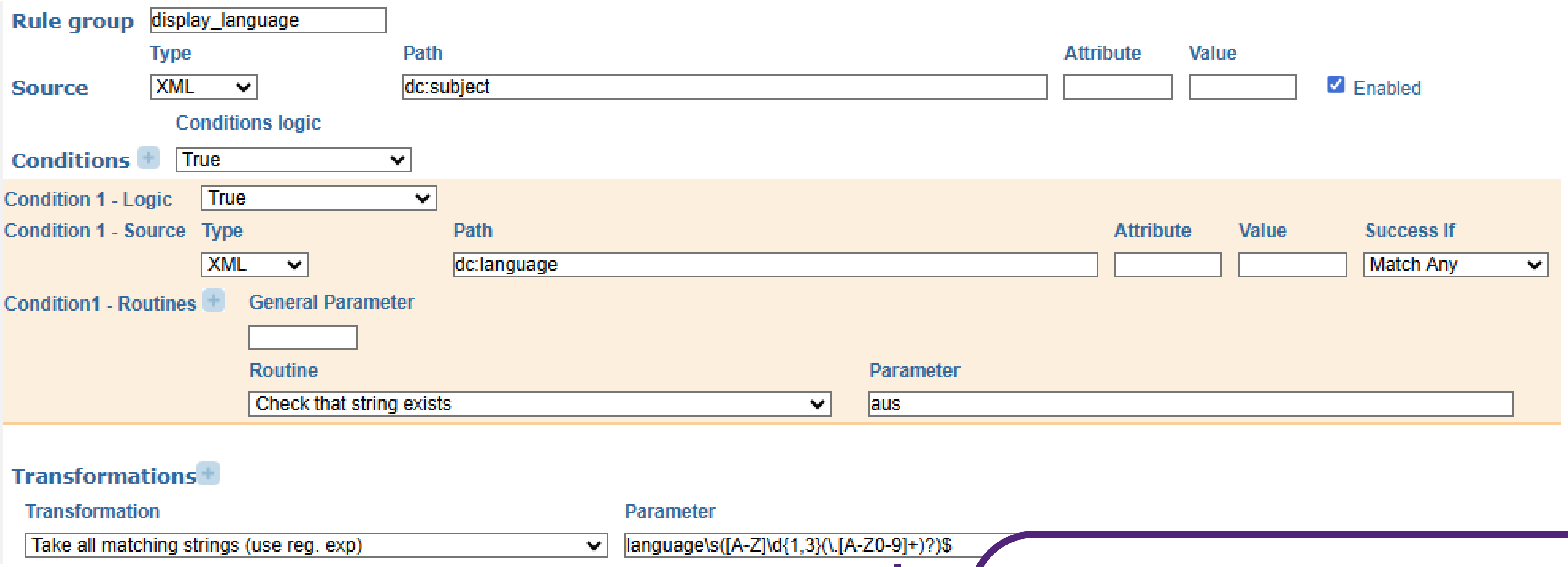
The language code always has an upper case character first, then may have one, two or three digits, then may have a full stop, and then the full stop may be followed by a digit or an upper case character

A record may have many dc:subjects with Austlang language codes

The final data must be in an exact form in dc:language to map against the Customised Languages table

Ideally, the record must also include dc:language field of aus, and may also have other data such as an additional entry of eng for English

AtoM: BO to VE regex transformation



Rule group: display_language

Source — Type: XML; Path: dc:subject; Attribute: ; Value: ; ☑ Enabled

Conditions — Conditions logic: True

Condition 1 - Logic: True

Condition 1 - Source — Type: XML; Path: dc:language; Attribute: ; Value: ; Success If: Match Any

Condition1 - Routines — General Parameter:

Routine: Check that string exists; Parameter: aus

Transformations

Transformation: Take all matching strings (use reg. exp)

Parameter: language\s([A-Z]\d{1,3}(\.[A-Z0-9]+)?)$

Primo BO

Take all matching strings (use reg. exp)

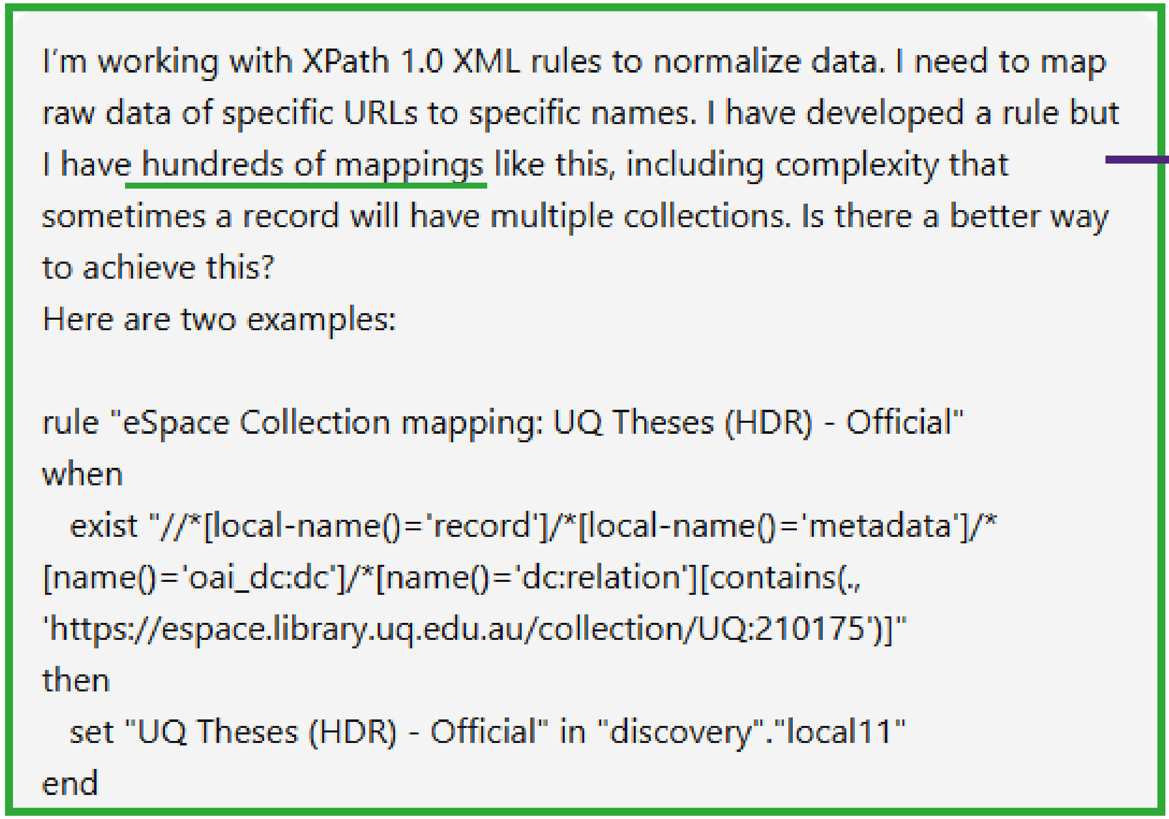
language\s([A-Z]\d{1,3}(\.[A-Z0-9]+)?)$

Primo VE

```
rule "dc:language for Austlang languages"
when
  exist "//*[local-name()='subject'][contains(text(),' language ')]"
then
  set TEMP"2" to xpath "//*[local-name()='subject'][contains(text(),' language ')]" concat values with ";"
  lower case (TEMP"2")
  return list using regex (TEMP"1",TEMP"2","(?<=language\\s+)([a-z]\\d{1,3})(?=\\D|$)")
  set list TEMP"1" in "dc"."language"
end
```

eSpace: Local data mapping

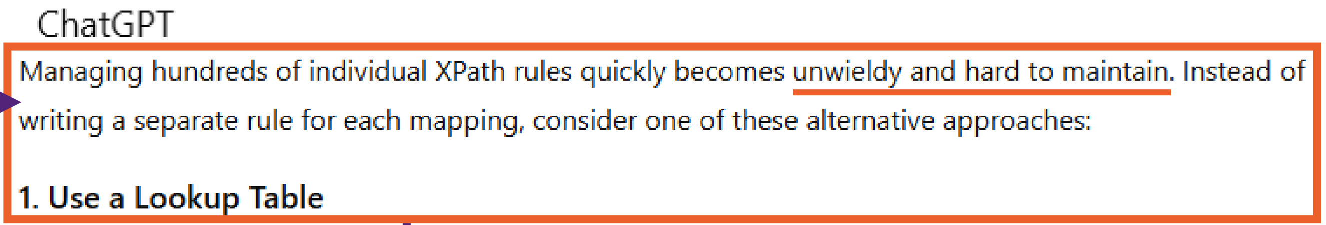
Feature Capability gap: No ability to create custom mapping tables, for use case such as org units and local collections



I'm working with XPath 1.0 XML rules to normalize data. I need to map raw data of specific URLs to specific names. I have developed a rule but I have hundreds of mappings like this, including complexity that sometimes a record will have multiple collections. Is there a better way to achieve this?

Here are two examples:

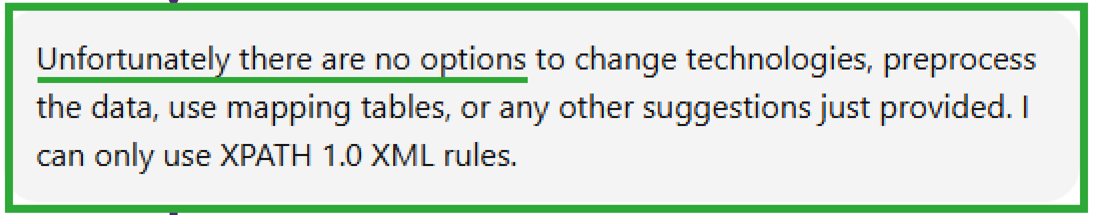
```
rule "eSpace Collection mapping: UQ Theses (HDR) - Official"
when
  exist "//*[local-name()='record']/*[local-name()='metadata']/*
[name()='oai_dc:dc']/*[name()='dc:relation'][contains(.,
'https://espace.library.uq.edu.au/collection/UQ:210175')]"
then
  set "UQ Theses (HDR) - Official" in "discovery"."local11"
end
```



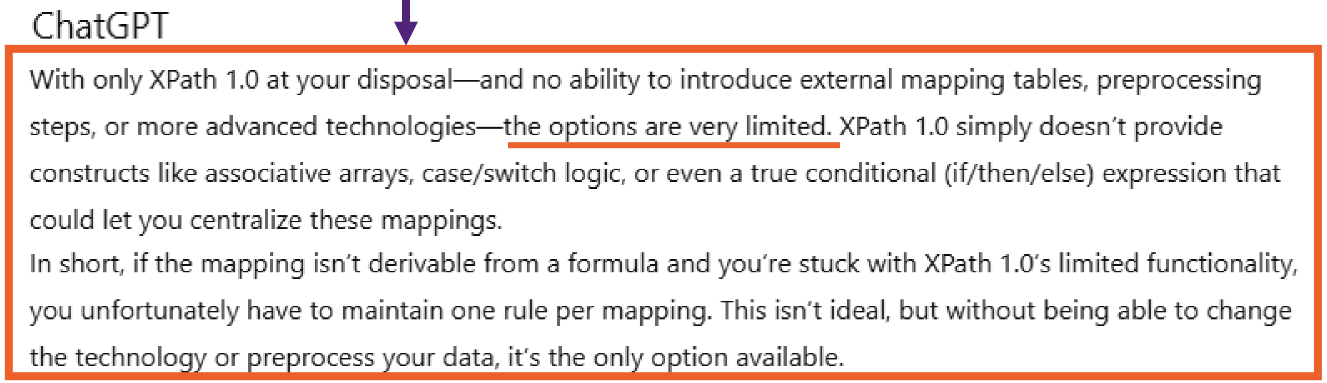
ChatGPT

Managing hundreds of individual XPath rules quickly becomes unwieldy and hard to maintain. Instead of writing a separate rule for each mapping, consider one of these alternative approaches:

1. Use a Lookup Table



Unfortunately there are no options to change technologies, preprocess the data, use mapping tables, or any other suggestions just provided. I can only use XPATH 1.0 XML rules.



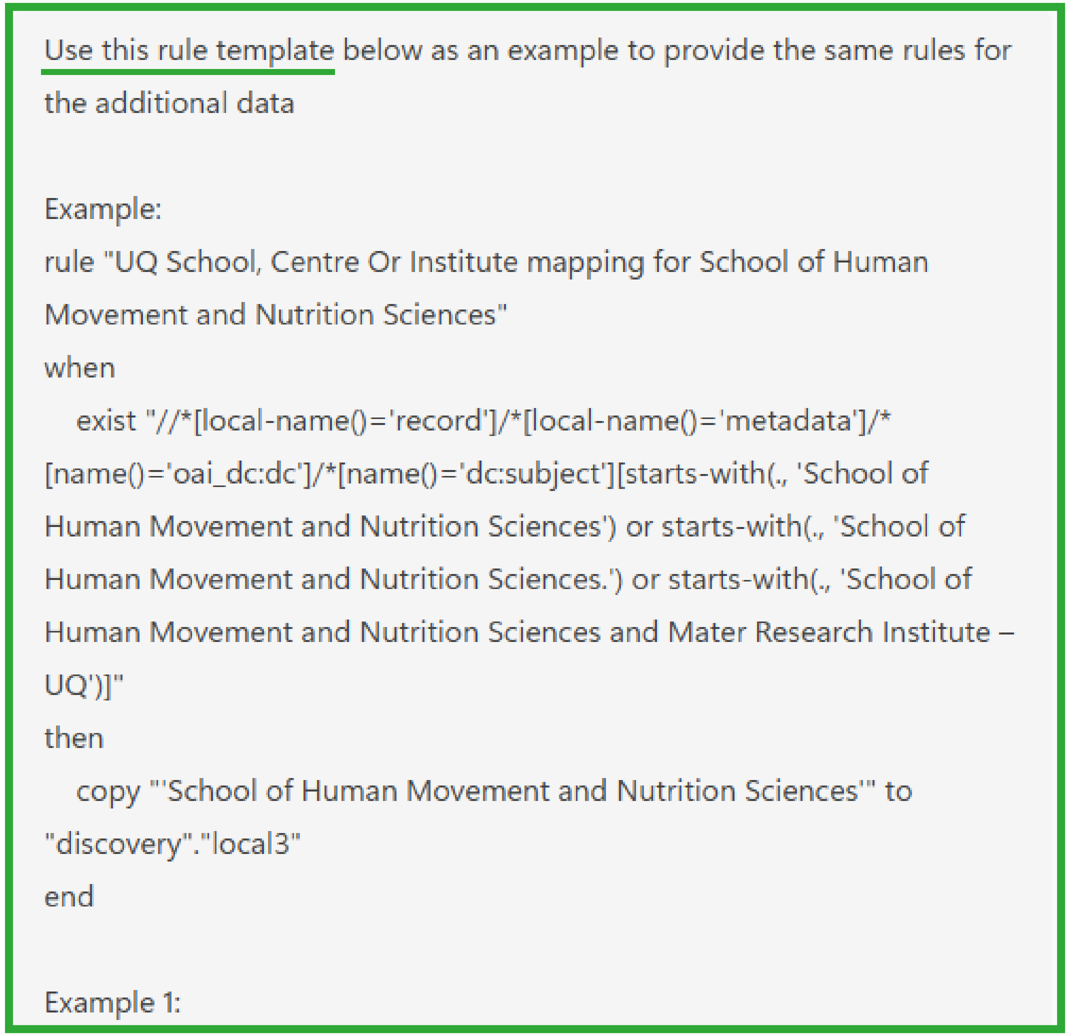
ChatGPT

With only XPath 1.0 at your disposal—and no ability to introduce external mapping tables, preprocessing steps, or more advanced technologies—the options are very limited. XPath 1.0 simply doesn't provide constructs like associative arrays, case/switch logic, or even a true conditional (if/then/else) expression that could let you centralize these mappings.

In short, if the mapping isn't derivable from a formula and you're stuck with XPath 1.0's limited functionality, you unfortunately have to maintain one rule per mapping. This isn't ideal, but without being able to change the technology or preprocess your data, it's the only option available.

eSpace: Local data mapping cont...

Accept the fact that I'm going to have hundreds of rules

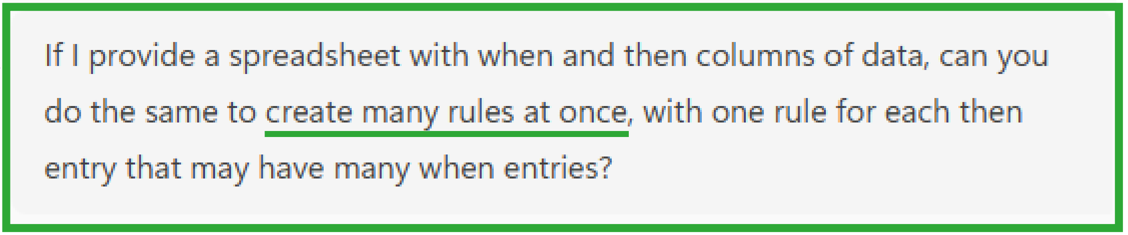


Use this rule template below as an example to provide the same rules for the additional data

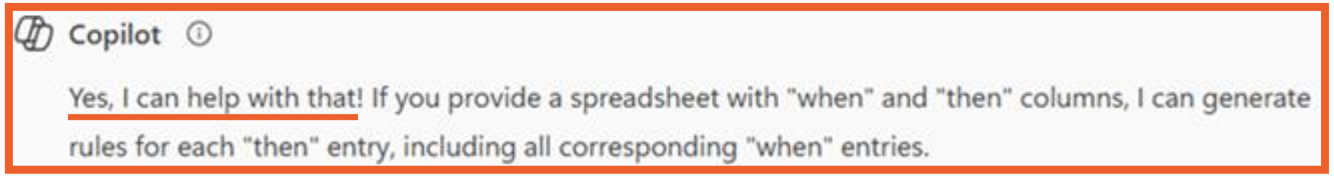
Example:

```
rule "UQ School, Centre Or Institute mapping for School of Human Movement and Nutrition Sciences"
when
    exist "//*[local-name()='record']/*[local-name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:subject'][starts-with(., 'School of Human Movement and Nutrition Sciences') or starts-with(., 'School of Human Movement and Nutrition Sciences.') or starts-with(., 'School of Human Movement and Nutrition Sciences and Mater Research Institute – UQ')]"
then
    copy "'School of Human Movement and Nutrition Sciences'" to "discovery"."local3"
end
```

Example 1:

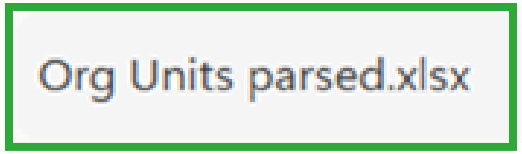


If I provide a spreadsheet with when and then columns of data, can you do the same to create many rules at once, with one rule for each then entry that may have many when entries?



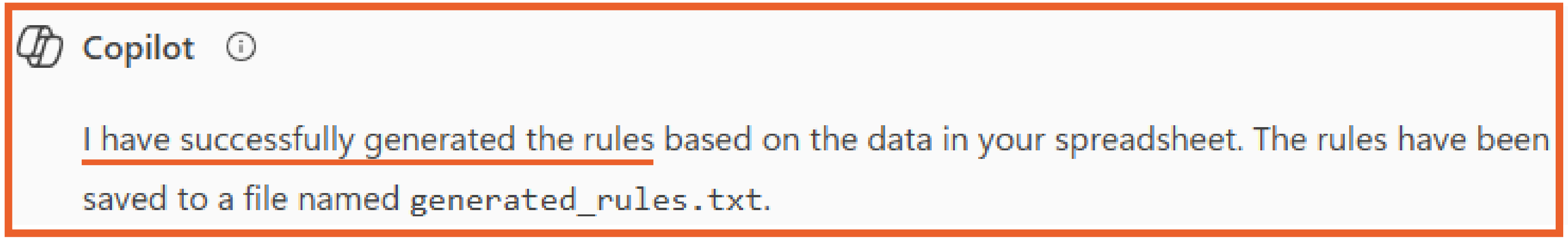
Yes, I can help with that! If you provide a spreadsheet with "when" and "then" columns, I can generate rules for each "then" entry, including all corresponding "when" entries.

- Export the mapping table from Primo BO
- Separate the delimited entries
- Then use AI to write the rules in bulk to a template…



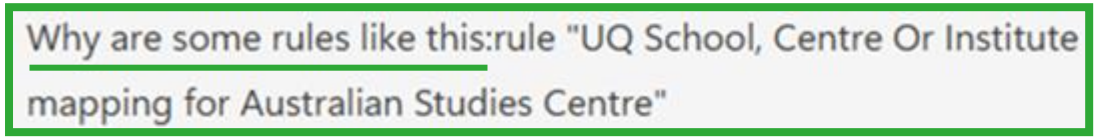
Org Units parsed.xlsx

eSpace: Local data mapping cont...

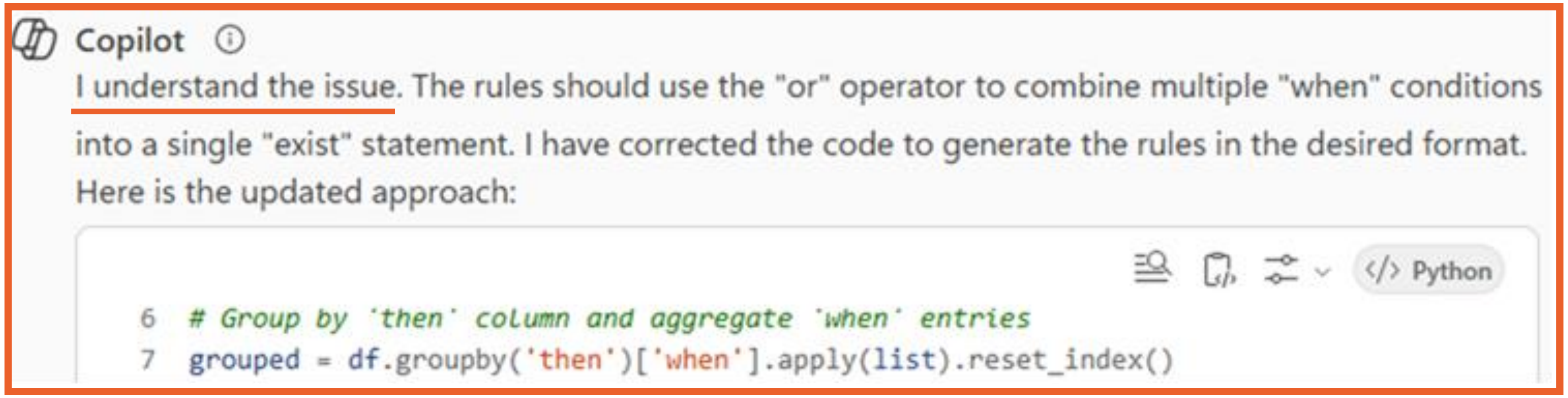


Copilot

I have successfully generated the rules based on the data in your spreadsheet. The rules have been saved to a file named `generated_rules.txt`.



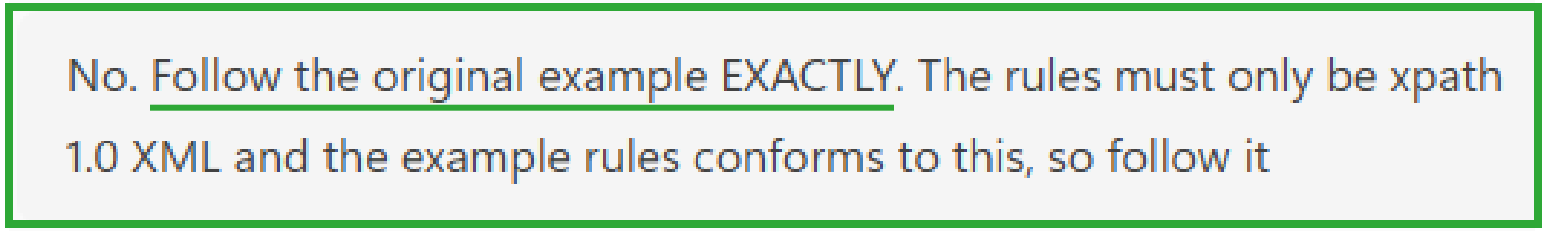
Why are some rules like this:rule "UQ School, Centre Or Institute mapping for Australian Studies Centre"



Copilot

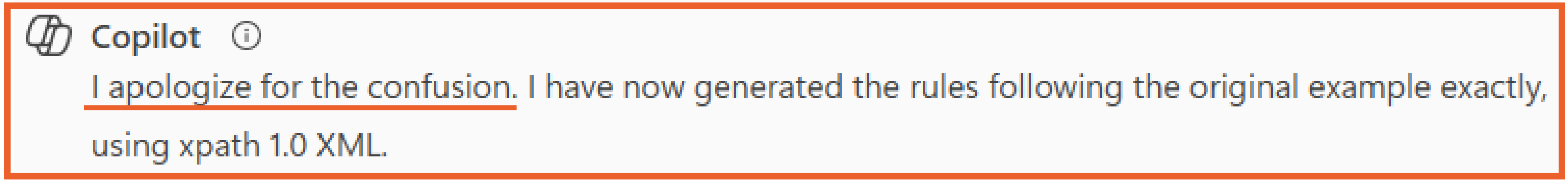
I understand the issue. The rules should use the "or" operator to combine multiple "when" conditions into a single "exist" statement. I have corrected the code to generate the rules in the desired format. Here is the updated approach:

```python
# Group by 'then' column and aggregate 'when' entries
grouped = df.groupby('then')['when'].apply(list).reset_index()
```



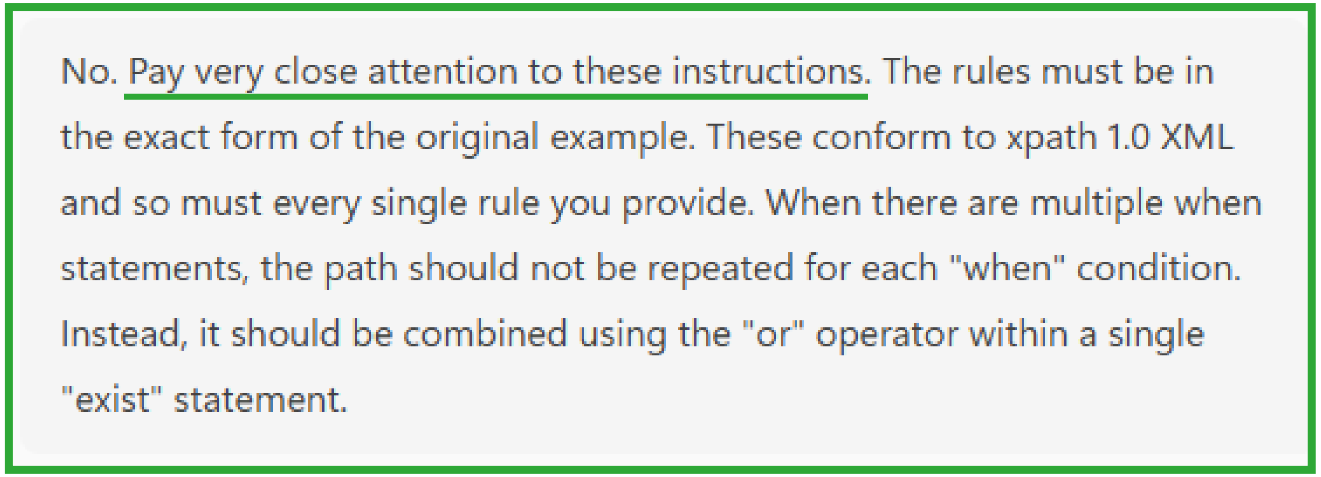
No. Follow the original example EXACTLY. The rules must only be xpath 1.0 XML and the example rules conforms to this, so follow it

eSpace: Local data mapping cont...

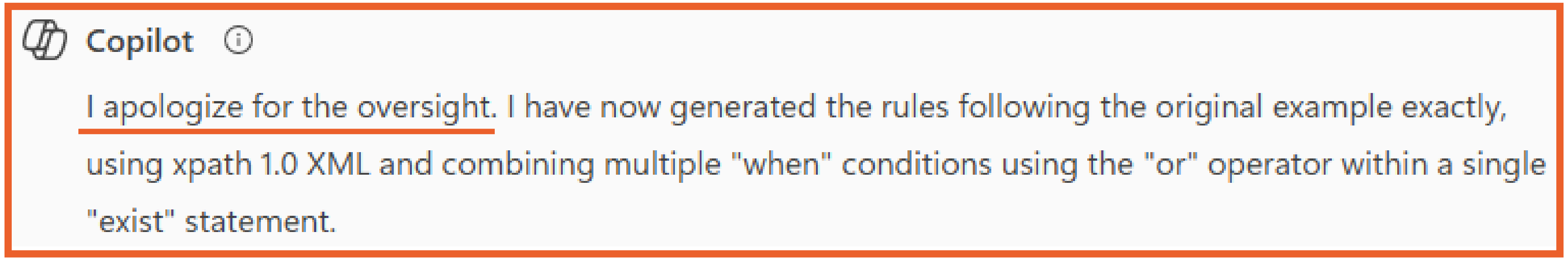


Copilot

I apologize for the confusion. I have now generated the rules following the original example exactly, using xpath 1.0 XML.



No. Pay very close attention to these instructions. The rules must be in the exact form of the original example. These conform to xpath 1.0 XML and so must every single rule you provide. When there are multiple when statements, the path should not be repeated for each "when" condition. Instead, it should be combined using the "or" operator within a single "exist" statement.



Copilot

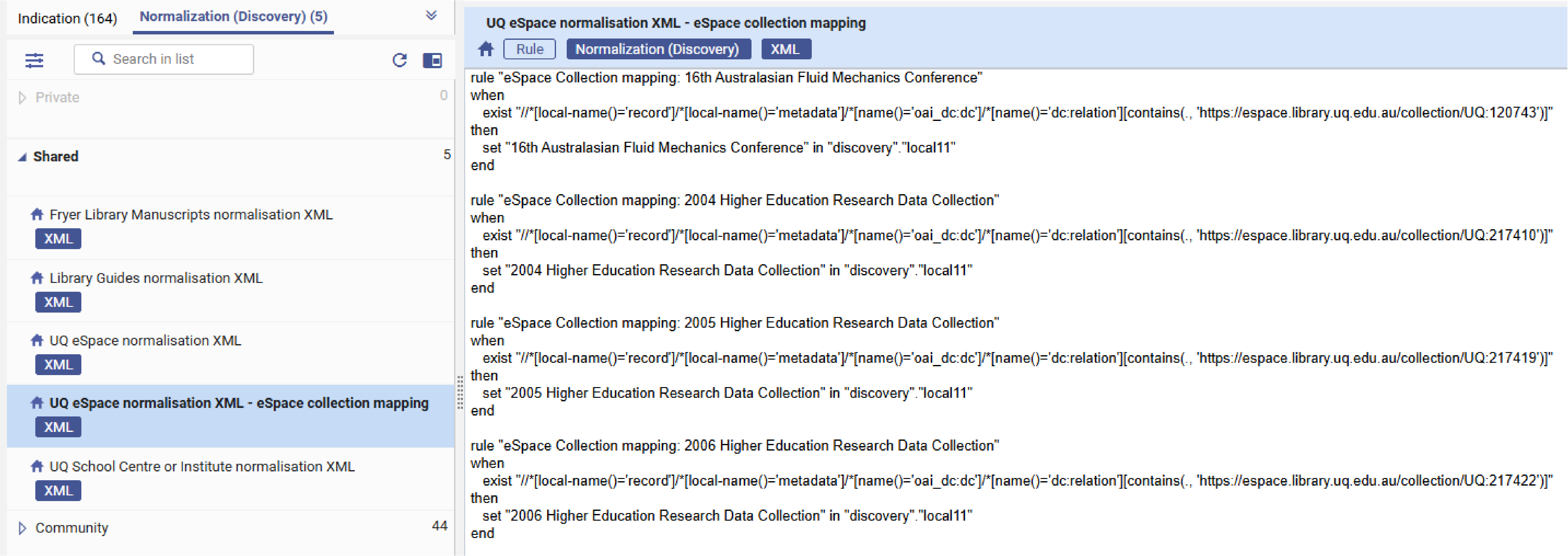
I apologize for the oversight. I have now generated the rules following the original example exactly, using xpath 1.0 XML and combining multiple "when" conditions using the "or" operator within a single "exist" statement.

eSpace: Local data mapping cont...

eSpace: Local data mapping cont...

Issue in the middle… Hit undocumented file size limit, so new XML processes created

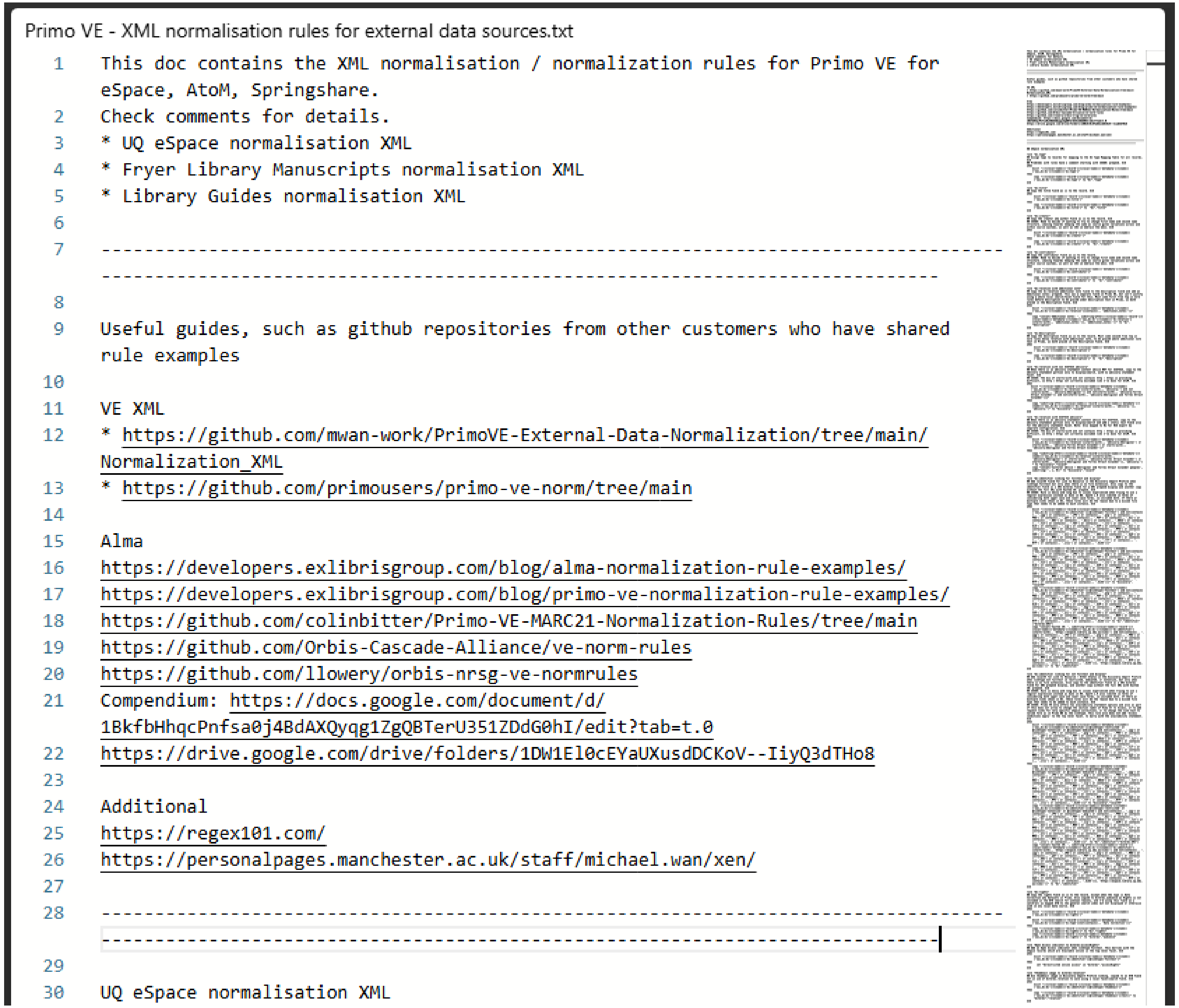
Plus – Feature Capability gap: No ability to remove the data from OTB Subject field for org unit anyway!


Indication (164)
Normalization (Discovery) (5)
Search in list
Private
0
Shared
5
Fryer Library Manuscripts normalisation XML
XML
Library Guides normalisation XML
XML
UQ eSpace normalisation XML
XML
UQ eSpace normalisation XML - eSpace collection mapping
XML
UQ School Centre or Institute normalisation XML
XML
Community
44
UQ eSpace normalisation XML - eSpace collection mapping
Rule
Normalization (Discovery)
XML

```
rule "eSpace Collection mapping: 16th Australasian Fluid Mechanics Conference"
when
   exist "//*[local-name()='record']/*[local-name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:relation'][contains(., 'https://espace.library.uq.edu.au/collection/UQ:120743')]"
then
   set "16th Australasian Fluid Mechanics Conference" in "discovery"."local11"
end

rule "eSpace Collection mapping: 2004 Higher Education Research Data Collection"
when
   exist "//*[local-name()='record']/*[local-name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:relation'][contains(., 'https://espace.library.uq.edu.au/collection/UQ:217410')]"
then
   set "2004 Higher Education Research Data Collection" in "discovery"."local11"
end

rule "eSpace Collection mapping: 2005 Higher Education Research Data Collection"
when
   exist "//*[local-name()='record']/*[local-name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:relation'][contains(., 'https://espace.library.uq.edu.au/collection/UQ:217419')]"
then
   set "2005 Higher Education Research Data Collection" in "discovery"."local11"
end

rule "eSpace Collection mapping: 2006 Higher Education Research Data Collection"
when
   exist "//*[local-name()='record']/*[local-name()='metadata']/*[name()='oai_dc:dc']/*[name()='dc:relation'][contains(., 'https://espace.library.uq.edu.au/collection/UQ:217422')]"
then
   set "2006 Higher Education Research Data Collection" in "discovery"."local11"
end
```

Extra tips

- Create sets of example records for testing
- Keep a copy of your rules externally
- Test & write-up a reload process



Primo VE - XML normalisation rules for external data sources.txt

```
This doc contains the XML normalisation / normalization rules for Primo VE for eSpace, AtoM, Springshare.
Check comments for details.
* UQ eSpace normalisation XML
* Fryer Library Manuscripts normalisation XML
* Library Guides normalisation XML


-------------------------------------------------------------------------------------
--------------------------------------------------------------------------------

Useful guides, such as github repositories from other customers who have shared rule examples

VE XML
* https://github.com/mwan-work/PrimoVE-External-Data-Normalization/tree/main/Normalization_XML
* https://github.com/primousers/primo-ve-norm/tree/main

Alma
https://developers.exlibrisgroup.com/blog/alma-normalization-rule-examples/
https://developers.exlibrisgroup.com/blog/primo-ve-normalization-rule-examples/
https://github.com/colinbitter/Primo-VE-MARC21-Normalization-Rules/tree/main
https://github.com/Orbis-Cascade-Alliance/ve-norm-rules
https://github.com/llowery/orbis-nrsg-ve-normrules
Compendium: https://docs.google.com/document/d/1BkfbHhqcPnfsa0j4BdAXQyqg1ZgQBTerU351ZDdG0hI/edit?tab=t.0
https://drive.google.com/drive/folders/1DW1El0cEYaUXusdDCKoV--IiyQ3dTHo8

Additional
https://regex101.com/
https://personalpages.manchester.ac.uk/staff/michael.wan/xen/


-------------------------------------------------------------------------------------
--------------------------------------------------------------------------------

UQ eSpace normalisation XML
```

Key Ex Libris documentation

- **Primary resources hub page**
  - Loading records from external sources into Primo VE – *8 pages BUT still not everything you need*

- **Reference also**
  - Working with normalization rules – *Alma focused, but related information*
  - Mapping to the Display, Facets, and Search sections in the Primo VE record – *the OTB for Dublin Core form*
  - Managing Display and Local Fields for Primo VE – *foundational details for local fields and options*
  - Managing Local Search and Facet Fields for Primo VE – ***Tip!** Request extension from 10 to 20 fields by Support case*
  - Configuring normalization rules for display and local fields – *OTB infrastructure eg display constants, syntax, options, limitations*

Recap

XML Normalization with Primo VE

Background + Starting point

Examples

Example 1: Library Guide subjects

Example 2: Austlang languages in special collections

Example 3: Institutional Repository mapping

Resources

Questions?

Questions?

Stacey van Groll

Manager, Discovery and Access

s.vangroll@uq.edu.au