Function 2: Clear All dc:relation Fields

Overview

Function 2 removes specific Dublin Core relation fields from bibliographic records in Alma. This data-modifying function targets `dc:relation` elements containing URLs to legacy Digital Grinnell collections, cleaning up obsolete metadata from migrated records. The function includes safety confirmations to prevent accidental data deletion.

What It Does

This function searches for and deletes `dc:relation` fields that contain URLs starting with `https://digital.grinnell.edu/islandora/object/`, which represent links to the old Digital Grinnell repository. These fields become obsolete after migration to Alma Digital.

Key Features

- **Targeted deletion**: Only removes dc:relation fields with specific URL pattern
- **Safe filtering**: Preserves dc:relation fields with other content
- **Batch processing**: Works on single records or entire sets
- **Confirmation dialog**: Requires user approval before deletion
- **Progress tracking**: Real-time progress bar during batch operations
- **Kill switch**: Can stop batch processing mid-operation
- **Detailed results**: Reports count of cleaned, skipped, and failed records

The Problem It Solves

Legacy Metadata Cleanup

When records were migrated from Digital Grinnell (Islandora) to Alma Digital, many retained `dc:relation` fields pointing back to the old repository:

```
<dc:relation>https://digital.grinnell.edu/islandora/object/grinnell:12345</dc:relation>
```

Issues with keeping these fields:

- Links point to deprecated system
- May confuse users about authoritative source
- Clutter metadata with obsolete information
- Don't follow current metadata best practices
- Can break if old system goes offline

Solution: Function 2 removes these obsolete relation fields, cleaning up migrated records and ensuring metadata reflects current repository architecture.

How It Works

Step-by-Step Process

1. **User Input**:
    - Enter MMS ID for single record, or
    - Load set ID for batch processing
2. **Confirmation Dialog**:
    - Shows record count (single or batch)
    - Warns about permanent deletion
    - Requires user to click "Proceed" or "Cancel"
3. **For Each Record**:
    - Fetch full bibliographic record XML from Alma
    - Parse XML to find `<anies><record>` section
    - Locate all `dc:relation` elements
    - Check each one for target URL pattern
    - Remove matching elements from XML
    - Update record back to Alma (only if changes made)
4. **Result Tracking**:
    - **Cleaned**: Records where dc:relation fields were removed
    - **Skipped**: Records with no matching dc:relation fields
    - **Failed**: Records that encountered errors
5. **User Feedback**:
    - Progress bar updates in real-time (batch mode)
    - Final summary with counts for each category
    - Detailed logging for troubleshooting

The Deletion Logic

Target Pattern:

```
url_starts_with = "https://digital.grinnell.edu/islandora/object/"
```

Matching Process:

```
# Find all dc:relation elements in the Dublin Core section
relation_elements = record_element.findall('.//dc:relation', namespaces)

# Track which ones to delete
elements_to_delete = []
```

```
# Check each dc:relation
for rel_elem in relation_elements:
    if rel_elem.text and rel_elem.text.startswith(url_starts_with):
        elements_to_delete.append(rel_elem)

# Remove matched elements
for elem in elements_to_delete:
    parent = record_element
    parent.remove(elem)
```

What Gets Kept:

- `dc:relation` fields with different URLs
- `dc:relation` fields with non-URL content (titles, series, etc.)
- All other Dublin Core fields (dc:title, dc:creator, etc.)
- All MARC21 data
- All other parts of the bibliographic record

Usage

Single Record Mode

Step-by-Step:

1. **Enter MMS ID**:
    - Type or paste MMS ID in the "MMS ID" input field
    - Example: `991234567890104641`
2. **Select Function**:
    - Choose "Clear All dc:relation Fields" from dropdown
3. **Click Execute**:
    - Click the function button
    - Confirmation dialog appears
4. **Confirmation Dialog**:

    ```
    Warning: Clear dc:relation Fields

    This will modify 1 record in Alma by removing dc:relation fields.

    This action cannot be undone.

    Are you sure you want to proceed?

    [Cancel]  [Proceed]
    ```

5. **Review and Approve**:
    - Click "Cancel" to abort
    - Click "Proceed" (red button) to continue
6. **Processing**:
    - Function fetches record
    - Removes matching dc:relation fields
    - Updates record in Alma
    - Shows result message
7. **Result**:

```
Function 2 completed:
- Cleaned: 1
- Skipped: 0
- Failed: 0
```

Batch Processing Mode

Step-by-Step:

1. **Load Set**:
    - Enter set ID in "Set ID" field (e.g., `7071087320004641`)
    - Or click the DCAP01 set ID link to auto-populate
    - Click "Load Set"
    - Wait for set members to load
2. **Select Function**:
    - Choose "Clear All dc:relation Fields" from dropdown
3. **Click Execute**:
    - Click the function button
    - Confirmation dialog appears with total count
4. **Confirmation Dialog**:

```
Warning: Clear dc:relation Fields

This will modify 2,847 records in Alma by removing dc:relation fields.

This action cannot be undone.

Are you sure you want to proceed?
```

```
[Cancel]  [Proceed]
```

5. **Review and Approve**:
    - **IMPORTANT**: Note the record count carefully
    - Click "Cancel" if the count seems wrong
    - Click "Proceed" (red button) to start batch processing
6. **Monitor Progress**:
    - Progress bar shows current record number
    - Percentage completion updates in real-time
    - Can click "Kill" button to stop processing
7. **Kill Switch** (if needed):
    - Click "Kill" button to stop processing
    - Current record completes
    - No further records processed
    - Partial results displayed
8. **Final Results**:

    ```
    Function 2 completed:
    - Cleaned: 2,156
    - Skipped: 678
    - Failed: 13
    ```

Confirmation Dialog Details

Single Record Confirmation

Title: "Warning: Clear dc:relation Fields"

Message:

```
This will modify 1 record in Alma by removing dc:relation fields.

This action cannot be undone.

Are you sure you want to proceed?
```

Buttons:

- **Cancel**: Aborts operation, no changes made
- **Proceed** (red): Continues with deletion

Batch Confirmation

Title: "Warning: Clear dc:relation Fields"

Message:

```
This will modify 2,847 records in Alma by removing dc:relation fields.

This action cannot be undone.

Are you sure you want to proceed?
```

Key Differences:

- Shows actual count from loaded set
- Emphasizes scale of batch operation
- Same warning about permanence

XML Transformation

Before Function 2

```
<bib>
  <mms_id>991234567890104641</mms_id>
  <title>Grinnell Historical Photo</title>
  <anies>
    <record xmlns="http://alma.exlibrisgroup.com/dc/01GCL_INST"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:dcterms="http://purl.org/dc/terms/">
      <dc:title>Grinnell Historical Photo</dc:title>
      <dc:creator>Smith, John</dc:creator>
      <dc:date>1925</dc:date>

<dc:relation>https://digital.grinnell.edu/islandora/object/grinnell:12345</
dc:relation>
      <dc:relation>Part of Grinnell College Photographs
Collection</dc:relation>
      <dc:identifier>dg_12345</dc:identifier>
      <dc:rights>Public Domain</dc:rights>
    </record>
  </anies>
</bib>
```

After Function 2

```
<bib>
  <mms_id>991234567890104641</mms_id>
  <title>Grinnell Historical Photo</title>
  <anies>
    <record xmlns="http://alma.exlibrisgroup.com/dc/01GCL_INST"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:dcterms="http://purl.org/dc/terms/">
      <dc:title>Grinnell Historical Photo</dc:title>
      <dc:creator>Smith, John</dc:creator>
      <dc:date>1925</dc:date>
      <dc:relation>Part of Grinnell College Photographs Collection</dc:relation>
      <dc:identifier>dg_12345</dc:identifier>
      <dc:rights>Public Domain</dc:rights>
    </record>
  </anies>
</bib>
```

What Changed:

- ❌ Removed: `<dc:relation>https://digital.grinnell.edu/islandora/object/grinnell:12345</dc:relation>`
- ✓ Kept: `<dc:relation>Part of Grinnell College Photographs Collection</dc:relation>`
- ✓ Kept: All other Dublin Core fields unchanged

Result Categories

Cleaned Records

Definition: Records where at least one dc:relation field was removed

Criteria:

- Record had one or more dc:relation elements
- At least one matched the deletion pattern
- Element(s) successfully removed
- Record successfully updated in Alma

Example Scenarios:

- Record had 1 Islandora URL → removed
- Record had 2 Islandora URLs → both removed
- Record had 1 Islandora URL + 1 other relation → Islandora URL removed, other kept

Skipped Records

Definition: Records where no matching dc:relation fields were found

Criteria:

- No dc:relation elements in record, OR
- dc:relation elements present but don't match deletion pattern

- No changes made to record
- No API update call needed

Example Scenarios:

- Record has no dc:relation fields at all
- Record has dc:relation with different URL (e.g., external website)
- Record has dc:relation with text content (e.g., "Part of Series Name")
- Record already processed by previous run

Why Skipping Is Important:

- Avoids unnecessary API calls
- Prevents timestamp updates on unchanged records
- Improves batch processing performance
- Idempotent operation (safe to run multiple times)

Failed Records

Definition: Records that encountered errors during processing

Common Failure Causes:

Error Type	Cause	Example
404 Not Found	Invalid MMS ID	Typo in MMS ID or record deleted
401 Unauthorized	API key expired	Need to regenerate key
403 Forbidden	Insufficient permissions	API key lacks "Bibs" write access
500 Server Error	Alma internal error	Temporary Alma service issue
Network timeout	Slow connection	Large record or network issue
XML parse error	Malformed XML	Corrupted record data

Error Handling:

- Error logged with full details
- Record counted as "failed"
- Processing continues to next record (batch mode)
- Error message stored for review

Use Cases

1. Post-Migration Cleanup

Scenario: Records migrated from Digital Grinnell to Alma, need cleanup

Workflow:

1. Identify set of migrated records (e.g., DCAP01 set)
2. Load set in CABB

3. Run Function 2 in batch mode
4. Review results (cleaned vs. skipped)
5. Investigate any failed records

Benefits:

- Removes all obsolete Islandora links in one operation
- Ensures clean metadata in Alma
- Prepares records for public discovery
- Documents migration completion

2. Incremental Cleanup

Scenario: New records added to set, need to clean them up

Workflow:

1. Reload set to get updated member list
2. Run Function 2 again
3. Most records skipped (already processed)
4. Only new records cleaned

Benefits:

- Safe to run repeatedly
- Idempotent operation
- Catches newly added records
- No risk of duplicate processing

3. Single Record Fix

Scenario: Individual record has Islandora link that needs removal

Workflow:

1. Get MMS ID from Alma or search
2. Enter MMS ID in CABB
3. Run Function 2 in single mode
4. Verify removal with Function 1

Benefits:

- Quick fix for individual cases
- Doesn't require full batch run
- Immediate confirmation
- Easy to verify result

4. Quality Assurance Check

Scenario: Verify all migrated records have been cleaned

Workflow:

1. Load entire migration set
2. Run Function 2
3. Check results:
    - **Cleaned > 0**: Some records still had Islandora links
    - **Cleaned = 0**: All records already clean
4. Document completion

Benefits:

- Confirms cleanup completion
- Identifies any missed records
- Provides documentation for audit
- Safe to run for verification

Technical Details

API Operations

Read Record:

```
GET /almaws/v1/bibs/{mms_id}?view=full&expand=None
Accept: application/xml
```

Update Record:

```
PUT /almaws/v1/bibs/{mms_id}
Content-Type: application/xml
Body: <bib>...</bib>
```

Namespace Handling

Dublin Core Namespaces:

```
namespaces = {
    'dc': 'http://purl.org/dc/elements/1.1/',
    'dcterms': 'http://purl.org/dc/terms/'
}
```

Finding Elements:

```
# Find the Dublin Core record section
record_element = root.find('.//record[@xmlns]')
```

```
# Find all dc:relation elements
relation_elements = record_element.findall('.//dc:relation', namespaces)
```

Performance Considerations

Single Record:

- API calls: 1 GET + 1 PUT (if changes made)
- Time: 2-3 seconds typical

Batch Processing:

- API calls per record: 1 GET + 0-1 PUT
- Skipped records: Only 1 GET (no PUT)
- Time per record: ~2-3 seconds
- Total time: (record_count × 2.5 seconds) average
- Example: 2,847 records ≈ 2 hours

Optimization:

- Skip logic avoids unnecessary updates
- Only PUTs when changes actually made
- Progress bar provides time estimates
- Kill switch for long-running operations

Error Recovery

Network Errors:

- Each record processed independently
- Network failure on one record doesn't stop batch
- Failed record logged and counted
- Processing continues to next record

API Errors:

- 4xx/5xx errors logged with details
- Error message includes MMS ID and status code
- Record marked as failed
- Batch processing continues

Kill Switch:

- User can stop batch mid-process
- Current record completes
- Partial results displayed
- Can resume later (skipped records won't be reprocessed)

Safety Features

Confirmation Dialog

Purpose: Prevent accidental deletion of metadata

Design:

- Modal dialog (must respond to proceed)
- Clear warning message
- Record count prominently displayed
- "Cannot be undone" warning
- Red "Proceed" button (danger color)
- "Cancel" button (safe option)

User Protection:

- Forces conscious decision
- Shows scale of operation (especially important for batch)
- Provides chance to verify settings
- Requires deliberate click on red button

Idempotent Operation

Meaning: Safe to run multiple times on same records

Implementation:

- Only removes fields matching specific pattern
- Records without matching fields are skipped
- No changes made to already-clean records
- Running twice produces same result as running once

Benefits:

- Can verify completion by re-running
- Safe to run on overlapping sets
- No risk of "over-processing"
- Easy to catch newly added records

Logging

What Gets Logged:

- Function start with mode (single/batch) and count
- Each record processing start (MMS ID)
- Deletion details (how many dc:relation fields removed)
- API responses and errors
- Final results summary

Log Location: `logfiles/cabb_YYYYMMDD_HHMMSS.log`

Example Log Entries:

```
Function 2: Clear dc:relation Fields - Batch mode - 2,847 records
Processing MMS ID: 991234567890104641
Found 2 dc:relation fields to remove
Removed 2 dc:relation elements
Successfully updated record 991234567890104641
...
Function 2 completed: Cleaned: 2,156, Skipped: 678, Failed: 13
```

Best Practices

Before Running

1. **Test on single record first**: Verify function works as expected
2. **Use Function 1 to inspect**: View XML before and after for sample record
3. **Verify set membership**: Ensure set contains only intended records
4. **Check API key**: Confirm valid key with write permissions
5. **Note record count**: Be aware of batch size before proceeding
6. **Schedule appropriately**: Large batches best run during off-peak hours

During Execution

1. **Monitor progress**: Watch progress bar for stalls or errors
2. **Check results periodically**: Look at cleaned/skipped/failed counts
3. **Use kill switch if needed**: Stop if results seem unexpected
4. **Keep application open**: Don't close browser during batch processing
5. **Note any patterns**: If many failures, may indicate systematic issue

After Completion

1. **Review results**: Check cleaned/skipped/failed counts
2. **Investigate failures**: Look at logs for failed records
3. **Verify sample records**: Use Function 1 to confirm changes
4. **Document completion**: Note date and results for records
5. **Handle failures**: Process failed records individually if needed
6. **Update tracking**: Mark set or project as completed

Quality Assurance

1. **Run twice**: Second run should show all skipped (confirms completion)
2. **Random sampling**: Check 10-20 random records with Function 1
3. **Compare before/after**: Use XML exports for documentation
4. **Check for unintended deletions**: Verify other dc:relation fields intact
5. **Test discovery**: Confirm records display correctly in Primo

Limitations

- **Pattern-specific**: Only removes dc:relation fields with exact URL pattern
- **Not customizable**: Cannot change target URL pattern without code modification

- **Permanent deletion**: No built-in undo capability
- **Single pattern**: Cannot specify multiple patterns in one run
- **No backup**: Doesn't create backup before modification (manual backup recommended)
- **Batch only**: Cannot selectively process subset of loaded set

Troubleshooting

No Records Cleaned (All Skipped)

Possible Causes:

- Records already processed previously
- Records don't have dc:relation fields
- dc:relation fields don't match pattern
- Wrong set loaded

Solutions:

- Check sample record with Function 1
- Verify URL pattern in code matches actual data
- Confirm correct set loaded
- This is normal if re-running after completion

High Failure Rate

Possible Causes:

- API key expired or invalid
- Network connectivity issues
- Alma service degradation
- Rate limiting

Solutions:

- Regenerate API key
- Check network connection
- Wait and retry during off-peak hours
- Contact Alma support if persistent

Confirmation Dialog Doesn't Appear

Possible Causes:

- JavaScript error
- Browser blocking popup
- Page not fully loaded

Solutions:

- Check browser console for errors
- Disable popup blockers for application

- Refresh page and try again
- Try different browser

Wrong Records Being Modified

Possible Causes:

- Wrong set loaded
- Set membership changed
- MMS ID typo in single mode

Solutions:

- Use kill switch immediately
- Verify set ID before proceeding
- Check set membership in Alma
- Always test single record first

Integration with Other Functions

Before Function 2

Function 1: Inspect sample records

- Verify dc:relation fields exist
- Check URL format matches pattern
- Confirm records should be modified

Function 3: Export current state

- Document before state
- Create backup of metadata
- Useful for comparison later

After Function 2

Function 1: Verify changes

- Check dc:relation fields removed
- Confirm other fields intact
- Document after state

Function 3: Export cleaned records

- Document completion
- Compare before/after exports
- Audit changes made

Alongside Other Functions

Can be run in sequence with Functions 6 and 7 for comprehensive cleanup:

1. Function 2: Clear dc:relation
2. Function 6: Replace dc:rights
3. Function 7: Add Grinnell identifiers

All three can work on same set with independent confirmation dialogs.

Related Documentation

- **Alma Bibs API**: https://developers.exlibrisgroup.com/alma/apis/bibs/
- **Dublin Core dc:relation**: https://www.dublincore.org/specifications/dublin-core/dcmi-terms/#relation
- **Digital Grinnell Migration**: See project documentation

Version History

- **Initial Implementation**: Original cleanup function
- **Confirmation Dialog Added**: Enhanced safety feature
- **Purpose**: Post-migration metadata cleanup
- **Status**: Active, production-ready