Function 7: Add Grinnell: dc:identifier Field As Needed

Overview

Function 7 automatically adds standardized Grinnell identifiers to bibliographic records that have legacy Digital Grinnell identifiers. This function ensures that records migrated from the old Digital Grinnell system maintain consistent identifier formats by creating `Grinnell:` identifiers based on existing `dg_` identifiers.

What It Does

This function searches bibliographic records for `dc:identifier` fields containing legacy Digital Grinnell identifiers (starting with `dg_`) and creates corresponding standardized `Grinnell:` identifiers using the same numeric value.

Identifier Transformation

Pattern Recognition:

- **Input identifier**: `dg_<number>` (e.g., `dg_12345`)
- **Output identifier**: `Grinnell:<number>` (e.g., `Grinnell:12345`)

Example:

```
<!-- Before -->
<dc:identifier>dg_12345</dc:identifier>

<!-- After -->
<dc:identifier>dg_12345</dc:identifier>
<dc:identifier>Grinnell:12345</dc:identifier>
```

Smart Processing Logic

The function implements intelligent record filtering to avoid unnecessary updates:

Records That Get Updated

✅ Records with a `dg_` identifier AND no existing `Grinnell:` identifier

Example:

- Has: `dc:identifier` = `dg_98765`
- Missing: Any `Grinnell:` identifier
- **Action**: Adds `dc:identifier` = `Grinnell:98765`

Records That Get Skipped

⊘ **No dg_ identifier**: Record has no `dc:identifier` starting with `dg_`

- **Message**: "No dg_ identifier found"
- **Reason**: Nothing to transform

⊘ **Grinnell: identifier already exists**: Record already has a `Grinnell:` identifier

- **Message**: "Grinnell: identifier already exists"
- **Reason**: Update already applied previously

Processing Flow

Step-by-Step Process

1. **Fetch Record**: Retrieves the bibliographic record from Alma as XML
2. **Parse XML**: Parses the XML and locates all `dc:identifier` elements
3. **Scan Identifiers**: Examines each identifier to find:
   - Any starting with `dg_` (source identifier)
   - Any starting with `Grinnell:` (target identifier)
4. **Evaluate Action**:
   - If no `dg_` found → Skip (nothing to transform)
   - If `Grinnell:` already exists → Skip (already done)
   - If `dg_` found and no `Grinnell:` → Proceed with update
5. **Extract Number**: Removes `dg_` prefix to get the numeric portion
6. **Create New Identifier**: Builds new `Grinnell:<number>` identifier
7. **Add to Record**: Appends new `dc:identifier` element to the record
8. **Update Alma**: Sends the modified XML back to Alma

XML Manipulation Details

Namespace Handling: The function properly handles Dublin Core namespaces:

- `dc`: http://purl.org/dc/elements/1.1/

Element Creation:

```
new_identifier = ET.Element('{http://purl.org/dc/elements/1.1/}identifier')
new_identifier.text = f"Grinnell:{dg_number}"
parent_element.append(new_identifier)
```

Parent Element Detection: The function locates the parent element that contains existing `dc:identifier` elements and adds the new identifier as a sibling.

Operation Modes

Single Record Mode

Process a single bibliographic record by MMS ID:

1. Enter an MMS ID in the input field
2. Select "Add Grinnell: dc:identifier Field As Needed" from the function dropdown
3. Click the function button
4. **Confirm the warning dialog** before proceeding
5. View result in the status area and log

Possible Outcomes:

- ✓ **Success**: "Added Grinnell:12345 to record 991234567890104641"
- ⊘ **Skipped**: "No dg_ identifier found" or "Grinnell: identifier already exists"
- × **Error**: "Failed to fetch record: 404" or other error message

Batch Mode

Process multiple records from a loaded set:

1. Load a set using "Load Set by ID" or "Load MMS IDs from CSV"
2. (Optional) Set a limit in the "Limit" field to process only the first N records
3. Select "Add Grinnell: dc:identifier Field As Needed" from the function dropdown
4. Click the function button
5. **Confirm the warning dialog** showing the number of records to be modified
6. Monitor progress via the progress bar
7. View summary results when complete

Batch Processing Features:

- Progress bar shows current record being processed
- Kill switch available to stop processing mid-batch
- Three-category summary: added, skipped, failed
- Individual record results logged with status symbols:
  - ✓ = Successfully added identifier
  - ⊘ = Skipped (no action needed)
  - × = Failed with error

Example Summary:

```
Batch complete: 45 added, 23 skipped, 2 failed out of 70 records
```

Safety Features

Confirmation Dialog

Before any modification occurs, a warning dialog appears with:

- ⚠️ Clear warning that Alma data will be PERMANENTLY modified
- Function name and description
- Number of records affected (batch mode) or MMS ID (single mode)
- Red "Proceed" button to continue

- "Cancel" button to abort

Example Warnings:

Single Record:

```
⚠️ WARNING: This will modify the bibliographic record in Alma.

MMS ID: 991234567890104641
Function: Add Grinnell: dc:identifier Field As Needed

This action will PERMANENTLY add a dc:identifier field if a dg_ identifier
exists.

Do you want to continue?
```

Batch Mode:

```
⚠️ WARNING: This will modify 150 bibliographic record(s) in Alma.

Function: Add Grinnell: dc:identifier Field As Needed

This action will PERMANENTLY add dc:identifier fields to records with dg_
identifiers.

Do you want to continue?
```

Additional Safeguards

- **No changes without confirmation**: Operation cannot proceed without explicit user approval
- **Idempotent**: Running multiple times on the same record won't create duplicates
- **Non-destructive**: Original `dg_` identifier is preserved
- **API validation**: Alma validates all XML before accepting changes
- **Detailed logging**: Every decision (add/skip/fail) is logged with MMS ID and reason
- **Error handling**: Failed updates are caught and reported without stopping batch processing
- **Kill switch**: Emergency stop button available during batch operations

Use Cases

1. Digital Grinnell Migration

After migrating records from the legacy Digital Grinnell system to Alma:

Scenario:

- 500 records imported from Digital Grinnell
- All have `dg_` identifiers (e.g., `dg_001`, `dg_002`, etc.)
- Need to add standardized `Grinnell:` identifiers for linking and discovery

Workflow:

1. Export records to CSV to identify those with `dg_` identifiers
2. Load the set of records
3. Run Function 7 on the entire set
4. Review log to confirm all identifiers added
5. Verify in Alma that records now have both identifier formats

2. Incremental Updates

Processing records added to Digital Grinnell collections over time:

Scenario:

- New batch of 50 digital objects added weekly
- Each gets a `dg_` identifier during initial cataloging
- Need to add `Grinnell:` identifiers for consistency

Workflow:

1. Load the weekly set of new records
2. Run Function 7 on the set
3. Review summary (should show 50 added, 0 skipped)

3. Cleanup After Manual Edits

Fixing records where `Grinnell:` identifiers were accidentally removed:

Scenario:

- Batch edit removed some `Grinnell:` identifiers
- `dg_` identifiers still intact
- Need to restore the `Grinnell:` identifiers

Workflow:

1. Identify affected records and create a set
2. Run Function 7 on the set
3. Function re-adds the `Grinnell:` identifiers based on existing `dg_` values

4. Quality Assurance

Verifying identifier consistency across a collection:

Scenario:

- Want to ensure all Digital Grinnell records have both identifier types
- Thousands of records to check

Workflow:

1. Load the entire Digital Grinnell collection set
2. Run Function 7 with a limit (e.g., 100 records) as a test

3. Review skipped count (high = good coverage, low = many missing)
4. Run on full set if needed
5. Expected result: High skip count indicates good identifier coverage

Technical Details

API Endpoints Used

GET Bibliographic Record:

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

PUT Updated Record:

```
PUT /almaws/v1/bibs/{mms_id}?
validate=true&override_warning=true&override_lock=true&stale_version_check=
false&check_match=false
```

XML Parsing Strategy

Finding Identifiers:

```
identifier_elements = root.findall('.//dc:identifier', search_namespaces)
for identifier_elem in identifier_elements:
    if identifier_elem.text.startswith("dg_"):
        # Found source identifier
    elif identifier_elem.text.startswith("Grinnell:"):
        # Found target identifier - skip record
```

Number Extraction:

```
dg_number = dg_identifier.replace("dg_", "")
# Example: "dg_12345" → "12345"
```

Creating New Element:

```
new_identifier = ET.Element('{http://purl.org/dc/elements/1.1/}identifier')
new_identifier.text = f"Grinnell:{dg_number}"
parent_element.append(new_identifier)
```

Error Handling

Common errors and handling:

Error Condition	Error Message	Handling
API Key not configured	"API Key not configured"	Returns error, no processing occurs
Record fetch failure	"Failed to fetch record: {status_code}"	Logs HTTP status and error details, continues to next record in batch
No parent element found	"Could not find parent element for dc:identifier"	Logs error, skips record
XML parsing error	"Error processing record {mms_id}: {error}"	Logs error with traceback, continues to next record
Update failure	"Failed to update record: {status_code}"	Logs full error response from Alma, continues to next record

Performance Considerations

Processing Speed:

- Single record: ~1-2 seconds per record
- Batch processing: Limited by API rate limits
- Progress updates: Every record in batch mode

API Rate Limits:

- Standard Alma limit: 25 requests per second
- Function uses 2 API calls per record (GET + PUT)
- Effective throughput: ~12 records per second maximum
- Large batches: May take several minutes

Network Optimization:

- Records processed sequentially (no parallel processing)
- Full record fetched each time (no caching)
- XML transmitted twice per record (GET + PUT)

Output and Logging

Success Messages

Single Record (Added):

```
Added Grinnell:12345 to record 991234567890104641
```

Single Record (Skipped):

```
No dg_ identifier found
```

or

```
Grinnell: identifier already exists
```

Batch Mode:

```
Batch complete: 45 added, 23 skipped, 2 failed out of 70 records
```

Log Entries

The function logs detailed information:

Successful Addition:

```
✓ 991234567890104641: Added Grinnell:12345 to record 991234567890104641
```

Skipped Records:

```
⊘ 991234567890104641: No dg_ identifier found
⊘ 991234567890204641: Grinnell: identifier already exists
```

Failed Records:

```
✗ 991234567890304641: Failed to fetch record: 404
```

Detailed Processing Steps:

- Start of operation with MMS ID
- Number of dc:identifier elements found
- Each identifier discovered (dg_ and Grinnell:)
- Decision made (add, skip, error)
- XML sample (first 500 chars) before sending to Alma
- API request/response details
- Final result for each record

Status Updates

Real-time status updates show:

- Current operation in progress
- Record being processed (batch mode)
- Progress percentage (batch mode)
- Final result summary

Identifier Format Standards

Legacy Digital Grinnell Format

Pattern: `dg_<number>`

Characteristics:

- Prefix: `dg_` (lowercase)
- Number: Variable length numeric value
- No leading zeros enforced
- Case-sensitive

Examples:

- `dg_1`
- `dg_42`
- `dg_12345`
- `dg_999999`

Standardized Grinnell Format

Pattern: `Grinnell:<number>`

Characteristics:

- Prefix: `Grinnell:` (mixed case, with colon)
- Number: Exact numeric value from dg_ identifier
- No modification of the number portion
- Case-sensitive

Examples:

- `dg_1` → `Grinnell:1`
- `dg_42` → `Grinnell:42`
- `dg_12345` → `Grinnell:12345`
- `dg_999999` → `Grinnell:999999`

Why Both Formats?

Preservation of Legacy Data:

- Original `dg_` identifiers maintained for historical reference
- Existing links and citations continue to work
- Audit trail of original system identifiers

Standardization Benefits:

- `Grinnell:` format follows institutional naming conventions
- Easier to identify and search for Grinnell-specific records
- Aligns with other institutional identifier schemes
- Supports future discovery layer enhancements

Best Practices

1. **Test First**: Run on 5-10 records before processing large batches
2. **Review Logs**: Check for unexpected skips or errors before proceeding
3. **Use Limits**: For large sets (>1000 records), process in batches with limits
4. **Export Before**: Use Function 3 to export records before bulk updates
5. **Verify Sample**: Spot-check updated records in Alma to confirm expected results
6. **Monitor Progress**: Watch the progress bar and log for patterns
7. **Document Changes**: Note which sets were processed and when

Limitations

- **Sequential Processing**: Records processed one at a time (not parallel)
- **No Batch API**: Uses individual GET/PUT calls for each record
- **Exact Match Required**: Only processes identifiers starting with exactly `dg_`
- **Number Preservation**: No normalization or padding of numeric values
- **Single dg_ Identifier**: If multiple `dg_` identifiers exist, uses the first one found
- **No Validation**: Does not verify that the number portion is valid or exists in legacy system
- **No Reverse Operation**: Cannot automatically remove `Grinnell:` identifiers

Integration with Other Functions

Function 1: Fetch and Display XML

- Use before Function 7 to examine record structure
- Verify that `dc:identifier` fields are present
- Check current identifier values

Function 2: Clear dc:relation Collections

- Can be run before or after Function 7
- Independent operations on different metadata fields

Function 3: Export Set to DCAP01 CSV

- Export records before running Function 7 for documentation
- Use CSV to identify records with `dg_` identifiers
- Export after Function 7 to verify additions

Function 4: Filter CSV for Pre-1931 Dates

- Combine with Function 3 to identify subset of records
- Apply Function 7 to filtered records

Function 6: Replace dc:rights

- Can be combined in a metadata cleanup workflow
- Run both functions on the same set
- Independent operations on different fields

Troubleshooting

Common Issues

No records updated in batch:

- **Cause**: All records either lack `dg_` identifiers or already have `Grinnell:` identifiers
- **Solution**: Export CSV and verify identifier fields, may not need Function 7

High failure rate:

- **Cause**: Network issues, API problems, or invalid MMS IDs
- **Solution**: Check log for specific error messages, verify API key permissions

"Could not find parent element":

- **Cause**: Unexpected XML structure in the record
- **Solution**: Use Function 1 to examine the record's XML, may need manual intervention

Identifier added with wrong number:

- **Cause**: Multiple `dg_` identifiers in record, function uses first found
- **Solution**: Manually verify and correct if needed

Recovery

If processing fails mid-batch:

- Check log file in `logfiles/` directory for details
- Identify last successfully processed record
- Create new set starting after that record
- Resume processing with new set

Related Documentation

- **Alma Bibs API**: https://developers.exlibrisgroup.com/alma/apis/bibs/
- **Dublin Core Identifier Element**: https://www.dublincore.org/specifications/dublin-core/dcmi-terms/#http://purl.org/dc/elements/1.1/identifier
- **Digital Grinnell Documentation**: [Internal documentation for legacy system]

Version History

- **Initial Implementation**: Added as Function 7 in December 2025
- **Purpose**: Support Digital Grinnell to Alma migration project
- **Status**: Active, production-ready