Is It Roy E. Harrington or Roy S. Harrington?: How to Make Technology Work for You In an ArchivesSpace Data Cleanup Project

July 8, 2020 – Webinar
Presenters

Amy Berish
Rockefeller Archive Center

Katie Martin
Rockefeller Archive Center

Darren Young
Rockefeller Archive Center
The Rockefeller Archive Center

“The Archives Program of the Rockefeller Archive Center fosters and supports a broad community of users examining the history of philanthropy and its related endeavors.”

- Opened in 1974
- Located in Sleepy Hollow, NY
- Independent operating foundation
- Makes available the papers of the Rockefeller Family, the records of the philanthropic institutions they founded, and the records of other philanthropic organizations
- Collections include: Rockefeller Foundation, Rockefeller Brothers Fund, Rockefeller University, Ford Foundation, Russell Sage Foundation, General Education Board, Henry Luce Foundation, Commonwealth Fund, Hewlett Foundation, etc.
Tools We Use

ArchivesSpace
A community served by LYRASIS

RockefellerArchiveCenter / scripts

dimes.rockarch.org
The Online Collections and Catalog of Rockefeller Archive Center
Context for ASpace Data Cleanup at RAC

- Moving to new discovery and delivery interface
- Known data issues inhibiting staff workflows
- Legacy data inherited from other content management systems
- Processing archivists’ collaboration with Digital Strategies Team on automated approaches to working with data in ASpace
Data Cleanup as 3 Projects

1. Agents

2. Legacy Access Notes

3. Dates

Read more about these projects on the RAC Blog: Bits and Bytes
Cleaning up Agent Records
How did we want to use our agents data?
ArchivesSpace: Agent in Resource Record

<table>
<thead>
<tr>
<th>Role</th>
<th>Relator</th>
<th>Agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creator</td>
<td>Author</td>
<td>Boggs, Nancy</td>
</tr>
<tr>
<td>Creator</td>
<td>Author</td>
<td>Ford Foundation</td>
</tr>
</tbody>
</table>
ArchivesSpace: Agent Record

Boggs, Nancy

Basic Information

Agent Type: Person
Publish: True


Names

Boggs, Nancy

Linked Records

<table>
<thead>
<tr>
<th>Record Type</th>
<th>Title</th>
<th>Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource</td>
<td>Ford Foundation records, Office of Reports, Office Files of Nancy Boggs</td>
<td>FA1533</td>
</tr>
</tbody>
</table>

Created by battlb 2018-07-18 15:30:07 -0400
Last Modified by k.martin 2020-03-24 09:16:45 -0400
DIMES: Agent in Resource Record

Ford Foundation records, Office of Reports, Office Files of Nancy Boggs (FA1533)

Contents

Overview
Access and Use
Arrangement

Overview

Creator
Boggs, Nancy
Ford Foundation

Title
Ford Foundation records, Office of Reports, Office Files of Nancy Boggs
What Prevented Us From Using Our Agents as Access Points?

- Duplicate agent records representing the same entity
- Inaccurate data in agent records
- No standard, consistent approach to the data in agent records
- Massive amounts of agent records assigned at the file level in the Ford Foundation grants and catalogued reports collections
What We Needed to Accomplish

1. Remove all duplicate agents from ArchivesSpace
2. Remove all file level agents in the Ford Foundation grants and catalogued reports collections
How We Hoped to Do It

Develop a Python script (or scripts) to automate the process of removing agent records we wanted gone
<table>
<thead>
<tr>
<th>Institution Name</th>
<th>Location</th>
<th>Job Title</th>
<th>Contact</th>
<th>Admin 1</th>
<th>Admin 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Columbia Basin Institute</td>
<td>NULL</td>
<td>NULL</td>
<td>237</td>
<td>pgalligan</td>
<td>pgalligan</td>
</tr>
<tr>
<td>Columbia Books, Inc. (Washington, D.C.)</td>
<td>NULL</td>
<td>NULL</td>
<td>237</td>
<td>admin</td>
<td>admin</td>
</tr>
<tr>
<td>Columbia College of Chicago</td>
<td>NULL</td>
<td>NULL</td>
<td>237</td>
<td>pgalligan</td>
<td>pgalligan</td>
</tr>
<tr>
<td>Columbia University</td>
<td>NULL</td>
<td>NULL</td>
<td>237</td>
<td>admin</td>
<td>admin</td>
</tr>
<tr>
<td>Columbia University. Institute of Human Nutrition</td>
<td>NULL</td>
<td>Institute of Human Nutrition</td>
<td>237</td>
<td>admin</td>
<td>admin</td>
</tr>
<tr>
<td>Columbia University. Nutrition</td>
<td>NULL</td>
<td>Nutrition</td>
<td>237</td>
<td>admin</td>
<td>admin</td>
</tr>
<tr>
<td>Columbia University</td>
<td>NULL</td>
<td>NULL</td>
<td>237</td>
<td>pgalligan</td>
<td>pgalligan</td>
</tr>
<tr>
<td>Columbia University Press</td>
<td>NULL</td>
<td>NULL</td>
<td>237</td>
<td>pgalligan</td>
<td>pgalligan</td>
</tr>
<tr>
<td>Columbia University. Academy of Political Science</td>
<td>NULL</td>
<td>NULL</td>
<td>242</td>
<td>admin</td>
<td>admin</td>
</tr>
<tr>
<td>Columbia University. Bureau of Applied Social Research</td>
<td>NULL</td>
<td>NULL</td>
<td>237</td>
<td>admin</td>
<td>admin</td>
</tr>
<tr>
<td>Columbia University. Bureau of Applied Social Research</td>
<td>NULL</td>
<td>NULL</td>
<td>237</td>
<td>pgalligan</td>
<td>pgalligan</td>
</tr>
<tr>
<td>Columbia University. Center for American Culture Studies</td>
<td>NULL</td>
<td>NULL</td>
<td>237</td>
<td>admin</td>
<td>admin</td>
</tr>
</tbody>
</table>
Harrington, Roy E.; Harrington, Roy L.; Harrington, Roy S.

- Duplicate names
- Misspelled names
- Inverted names
- Names with different middle initials
- Shoehorned LOC subject headings
- Subjects of some library books as agents
- Inconsistency in name formatting (‘Primary Part of Name’; ‘Rest of Name’)
- Incorrect agent types (Corporate used as Person)
- Inconsistent use of dates in names
- Inconsistent name source and rules
No Pattern We Could Identify

- The issues we discovered were too complex and too varied
- The script we had planned to write to unlink agents with no source would not solve them
New Approach: Keep, Merge, or Delete
ArchivesSpace Enhanced Agent Merging Function
### Merging Plan in Action

<table>
<thead>
<tr>
<th>Action</th>
<th>Gwin</th>
<th>Catherine</th>
<th>237</th>
<th>NULL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merge</td>
<td>Gwin, Catherine</td>
<td>NULL</td>
<td>237</td>
<td>NULL</td>
</tr>
<tr>
<td>Keep</td>
<td>Gwinn</td>
<td>Nancy E.</td>
<td>237</td>
<td>NULL</td>
</tr>
<tr>
<td>Merge</td>
<td>Gyftopoulos</td>
<td>E. P.</td>
<td>237</td>
<td>NULL</td>
</tr>
<tr>
<td>Merge</td>
<td>Gyftopoulos, Elias P. et al.</td>
<td>NULL</td>
<td>236</td>
<td>NULL</td>
</tr>
<tr>
<td>Keep</td>
<td>Gyorgy</td>
<td>Paul</td>
<td>NULL</td>
<td>242</td>
</tr>
<tr>
<td>Keep</td>
<td>Gyulavari, Antal</td>
<td>NULL</td>
<td>236</td>
<td>NULL</td>
</tr>
<tr>
<td>Keep</td>
<td>Gzesh, Susan</td>
<td>NULL</td>
<td>237</td>
<td>NULL</td>
</tr>
<tr>
<td>Keep</td>
<td>Haas</td>
<td>Joan K.</td>
<td>237</td>
<td>NULL</td>
</tr>
<tr>
<td>Merge</td>
<td>Haas</td>
<td>William J.</td>
<td>237</td>
<td>NULL</td>
</tr>
<tr>
<td>Keep</td>
<td>Haas</td>
<td>Ernest B.</td>
<td>237</td>
<td>NULL</td>
</tr>
<tr>
<td>Keep</td>
<td>Haas, Ellen</td>
<td>NULL</td>
<td>237</td>
<td>NULL</td>
</tr>
<tr>
<td>Keep</td>
<td>Haas, Toni, 1942-</td>
<td>NULL</td>
<td>237</td>
<td>NULL</td>
</tr>
<tr>
<td>Merge</td>
<td>Haas, W. J., 1924-</td>
<td>NULL</td>
<td>237</td>
<td>NULL</td>
</tr>
</tbody>
</table>
Agents Cleanup

Objective 1

- 6,704 agent records merged or deleted
- 18% of total agents in ArchivesSpace

October-December, 2019
Some drawbacks to our approach

- Merging agent records was slow and slowed down performance across ArchivesSpace for all users.
- Also some ArchivesSpace performance issues caused by merging agent records that were not valid.
Ford Foundation
Grants and Catalogued Reports

- Imported from Ford Foundation’s systems
- File level agents not part of RAC processing practices
- Agents not useful because the agents are named in the grant/report record
We Can Automate It!

- Clear aim: Remove all file level agents from a select group of resource records
- We were able to develop a Python script to unlink all agents records from file level archival objects within an indicated resource record
Running the script

1. Provide the corresponding resource ID for the collection guide on which you want to run the script
2. The script iterates through the files in the finding aid and unlinks all the agent records
# Remove Agents Script in Action

<table>
<thead>
<tr>
<th>Collection</th>
<th>Resource ID (script input)</th>
<th>Completed?</th>
<th>Number of Agents Deleted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ford Foundation records, Grants A-B (FA732A)</td>
<td>11460</td>
<td>✔</td>
<td>4,682</td>
</tr>
<tr>
<td>Ford Foundation records, Grants C-D (FA732B)</td>
<td>11381</td>
<td>✔</td>
<td>5,195</td>
</tr>
<tr>
<td>Ford Foundation records, Grants E-G (FA732C)</td>
<td>11389</td>
<td>✔</td>
<td>2,732</td>
</tr>
<tr>
<td>Ford Foundation records, Grants H-K (FA732D)</td>
<td>11396</td>
<td>✔</td>
<td>5,153</td>
</tr>
</tbody>
</table>
Agents Cleanup

Objective 2

82,041 file level agents unlinked from across 18 resource records

January-March, 2020
Cleaning Up Legacy Access Notes
Original Problem

- Unnecessary restriction notes appeared thousands of times at the file level of more than 40 finding aids.
- Extra work for reference staff
- Needed an automated solution
Getting Started

A script that can perform the following actions with ArchivesSpace data:

Universe

- An individual finding aid resource record.
- User enters the Resource ID Number/Finding Aid Number.

Find and Delete specified Conditions of Access Notes

- User enters the text of a Conditions of Access Note.
- Script finds the specified note, and deletes/eliminates the given note from the resource record.
Process

- Learning Python and ArchivesSpace API
- Standup meetings to move project forward
Changes and Improvements

- **ArchivesSnake** client library
- **Fuzzy string matching**

```python
def contains_match(content, search_string):
    #"""Returns True if user-provided note input matches the corresponding note within a given ratio (CONFIDENCE_RATIO)."""
    ratio = fuzz.token_sort_ratio(content.lower(), search_string.lower())
    return True if ratio > CONFIDENCE_RATIO else False
```
Changes and Improvements (continued)

- Logging top container information
- **Argparsing** Python module
- Expanded scope beyond access restriction notes

```python
def get_parser():
    parser = argparse.ArgumentParser()
    parser.add_argument("note_type", choices=NOTE_TYPE_CHOICES, help="The type of note within a finding aid you wish to modify or delete (ex. bioghist)")
    parser.add_argument("action", choices=ACTION_CHOICES, help="The action you wish to perform against matched notes")
    parser.add_argument("resource_id", type=int, help="The identifier of the resource record in which you want to search. Found in the URL.")
    parser.add_argument("search_string", help="A string to be matched against in resource record notes.")
    parser.add_argument("level", choices=LEVEL, help="The level within the resource hierarchy you would like to change (collection, series, file, or item.")
    parser.add_argument("-r", "--replace_string", help="The new note content to replace the old note content. (Only relevant if you are modifying note(s))")
    return parser
```
Running the Script

- Script can be found within the scripts repository of the Rockefeller Archive Center GitHub page: edit_notes.py
Using the Script for Data Cleanup
Using the Script for Data Cleanup (continued)

- “Prior archival review” notes appeared more than 20,000 times
- Removed fourteen different types of access notes that appeared over 27,000 times across 679 finding aids.
Lessons from the Access Project

- Learning takes time!
- Quality code requires input from more than one person
- Limiting the input requirements will save you time when you are running the script over and over again in the data cleanup process
Adding Structured Dates to Our Entire Repository
Dates in ArchivesSpace
What We Needed to Accomplish

- Use date expression field data to add begin/end dates to all/most archival objects in ArchivesSpace

Why?
- Facilitate faceted date searching
- Improved searching within our discovery system (DIMES)
The Original Plan

- Use **Calculate Dates** feature in Archivesspace
- Add structured (Begin/End) dates to all series-level components
Calculate Dates… Needs Dates!

- In order to use “Calculate Dates” you need actual dates!
- Calculate Dates relies on the existence of structured dates on archival objects below it
- 195,000 out of 650,000 archival objects were missing structured dates.
Finding A Solution:  

- Simple to use/install  
- Ability to parse formats other than YYYY/MM/DD  
- High confidence in that data we were changing  
- Not erase dates it cannot understand

Searching for Tools:  

Tools we considered:  

- DateUtil python module  
- OpenRefine  
- Timewalk plug-in  
- Timetwister gem
Our Choice: Timewalk Plug-In

- Automated date parser for ArchivesSpace
- Parse any values in the Date Expression field into ISO8601-compliant Begin and End values.
- Parses out date certainties and sets the calendar/era values automatically

https://github.com/alexduryee/timewalk
Implementing and Testing Timewalk

- Install Timewalk on development
- Test using examples from our repository

What does Timewalk do?

What doesn’t it do?
<table>
<thead>
<tr>
<th>Expression</th>
<th>Type</th>
<th>Begin</th>
<th>End</th>
<th>Certainty</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/2/1972</td>
<td>Single</td>
<td>1972-10-02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>June 3, 1958</td>
<td>Single</td>
<td>1978-06-03</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring 1996</td>
<td>Inclusive</td>
<td>1996-03-20</td>
<td>1996-06-21</td>
<td></td>
</tr>
<tr>
<td>Early 1950s</td>
<td>Inclusive</td>
<td>1950</td>
<td>1955</td>
<td></td>
</tr>
<tr>
<td>Jan-Nov 1917</td>
<td>Inclusive</td>
<td>1917-01</td>
<td>1917-11</td>
<td></td>
</tr>
<tr>
<td>undated</td>
<td>[blank]</td>
<td>[blank]</td>
<td>[blank]</td>
<td></td>
</tr>
<tr>
<td>Circa 1950</td>
<td>Single</td>
<td>1950</td>
<td></td>
<td>Approximate</td>
</tr>
<tr>
<td>C. 1950</td>
<td>Single</td>
<td>1950</td>
<td></td>
<td>Approximate</td>
</tr>
</tbody>
</table>
Timewalk Can Not Parse:

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Date</td>
<td>Does nothing</td>
</tr>
<tr>
<td>N.D</td>
<td>Does nothing</td>
</tr>
<tr>
<td>n/d</td>
<td>Does nothing</td>
</tr>
<tr>
<td>d.1913</td>
<td>Does nothing</td>
</tr>
<tr>
<td>Dec. 13, 1979</td>
<td>Does nothing</td>
</tr>
<tr>
<td>1979 Jan. 12</td>
<td>Does nothing</td>
</tr>
<tr>
<td>Probably 1938</td>
<td>Does nothing</td>
</tr>
<tr>
<td>Exhibited: 1960</td>
<td>Does nothing</td>
</tr>
</tbody>
</table>
“This Vehicle Stops for Quality Control”

- Manual work to address dates we knew Timewalk would not be able to understand:
  - “160” to “1960”
“This Vehicle Stops for Quality Control”

Taking advantage of patterns:

“Jan.” to “January”

“No date” to “undated”

“d. 1910” to “1910”

“Exhibited: 1960” to “1960”
Solution: Script that Triggers Timewalk

- List of expressions to “find and replace”
  - Standardized language used in date expressions:
    - Unknown dates = undated
    - Months should always be fully spelled out

- How do we want to run script?
  - On each finding aid vs the entire repository?
  - Per finding aid since working on production
Solution: Script that Triggers Timewalk (cont.)

- **Replace_date_expressions.py**
- “Walks” a resource tree
- Replaces date expressions that conform to list of “find and replace” patterns
- “Touches” (opens and saves) archival objects to trigger Timewalk
Solution: Script that Triggers Timewalk (cont.)

- Run script on all 1,700+ of our finding aids.
- 200 finding aids/week
- Assistance from Katie and Darren to divide and conquer
  - Divide list of finding aids into 3 sections
  - Project completed in 3-4 weeks
  - Updated hundreds of thousands of archival object records over 1700 finding aids
Thinking Long Term

● Timewalk has a permanently home on our production server of ArchivesSpace where it continues to be used, without running the script.
● As long as a date expression is entered, Timewalk will parse upon saving.
● Saves Processing Archivists from entering dates twice in a record.
Project Reflections

- Automation = delegate more tasks to machines
- Take advantage of patterns!
- Sometimes human intervention is needed

What is a job for a machine?
What is a job for a human?
Thank You!

Blog Series:

- [ArchivesSpace Clean Up: An Outline](#)
- [ArchivesSpace Clean Up: Wrangling Agents in the Wild](#)
- [ArchivesSpace Clean Up: Removing Legacy Access Restriction Notes](#)
- [ArchivesSpace Clean Up: Adding Structured Dates to an Entire Repository](#)

GitHub: [https://github.com/rockefellerarchivecenter](https://github.com/rockefellerarchivecenter)