A toolset for combining legacy collections data with data from modern digitization efforts

Maricela Abarca
California Academy of Sciences

mabarca@calacademy.org
github.com/myabarca

BACKGROUND

Some natural history collections have already implemented large digitization efforts while others have just begun to undertake specimen digitization or fall somewhere along this spectrum. In all scenarios, newly created data needs to be integrated into a database, often a relational database. A challenge may be presented by combining legacy data with newly generated data that is structured and standardized. This was the case for our fossil collections at the California Academy of Sciences (CAS). During and after the conclusion of the Eastern Pacific Invertebrate Communities of the Cenozoic (EPICC) thematic collections network (PI Peter Roopnarine, NSF Award 1503628), **we were operating out of two unconnected databases with an urgent need to integrate all of our assets into one database that would facilitate internal collections tasks and streamline serving data to aggregators like GBIF and iDigBio.**

THE PROBLEM

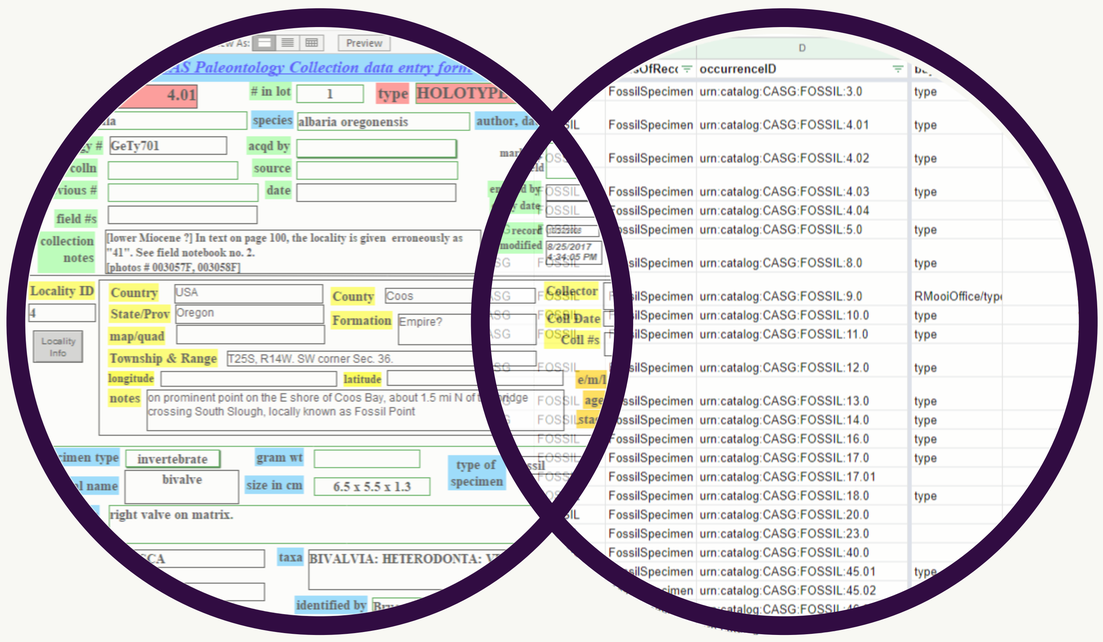
- Filemaker database (in use since the late 1990's) was not configured well for EPICC's data workflow.
- EPICC workers entered data into spreadsheets, the end goal being to ingest into a new database later.
- **Working in two separate databases at the same time led to duplication of catalog numbers and negatively affected regular collection tasks (e.g. loan tracking and record verification).**

THE PLAN

- Redesign EPICC MariaDB database into a full-service database that holds all digitized assets.
- Merge Filemaker records with EPICC data entry spreadsheets using the Python Pandas library and SQL queries.
  - Reconcile Filemaker fields with Darwin Core fields.
  - Clean original EPICC spreadsheets and exported Filemaker records for consistency with OpenRefine.

Old Data

- Filemaker relational database
- >15k specimen records
  - Some in EPICC's scope
  - All of our type specimens
- **Loan records**
- **Publication records**



New Data

- Spreadsheets of EPICC records
- >27k specimen records
- Specimen images
- **Darwin Core standards**
- **Georeferencing data**

THE OUTCOME

- **Better data usability, >37k lot records centralized for collection tasks.**
- **Ability to map occurrences and identify strengths and gaps in our data.**
- **Data added to legacy records (Filemaker specimens inherited coordinates if their locality was georeferenced as part of EPICC).**
- **An open-source solution.**

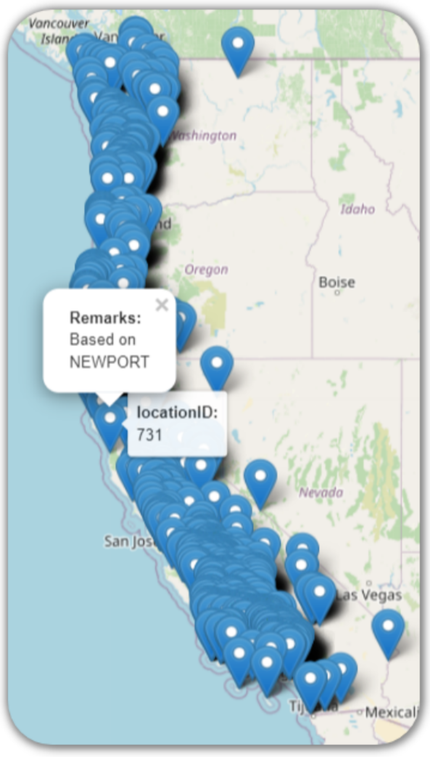
QR code to access resources and tools used in this project

Name	Rows	Size
georeference	1,724	1.5 MiB
loan	139	80.0 KiB
loan_lots	1,460	288.0 KiB
locality	8,652	4.5 MiB
lot	37,335	17.1 MiB
lot_is_publishe...	8,147	2.5 MiB
lot_with_loc_w...		
person	208	64.0 KiB
reference	1,237	416.0 KiB

Tables in MariaDB database

```
# need to install folium and pandas if you don't have them already installed
import folium
import pandas as pd

# replace filename in "" with your file's name.
# also make sure it's in the same folder as this jupyter notebook,
# or wherever your code is stored
df = pd.read_csv("localityDataWithGeorefData-samplePublic.csv")

# peek into a couple rows of your data
df.head()

# need strings for content to appear in tooltip in map
df['locationID'] = df['locationID'].apply(str)
df['georeferenceRemarks']=df['georeferenceRemarks'].apply(str)

# plotting a single point based on our csv data

# store the first row as your point

#point= df.loc[0]

# folium.Marker(
#     location=[point['decimalLatitude'], point['decimalLongitude']]

# ).add_to(map)

# map
# works for single points, but want to iterate over all columns to make markers for all points

# mapping all the points in your csv through iterration

# location=[40.43148077979194, -121.6627849541342],
# zoom_start=5 just sets a default map view and zoom. this can be customized

map = folium.Map(
location=[40.43148077979194, -121.6627849541342],
zoom_start=5
)

for i, point in df.iterrows():
    #iterrows stores the row index in i, and stores the row in point
    folium.Marker(
    location=[point['decimalLatitude_x'], point['decimalLongitude_x']],
    popup='<strong>Remarks: </strong><br>'+ point['georeferenceRemarks'],
    tooltip='<strong>locationID: </strong><br>'+ point['locationID']
    ).add_to(map)

map

# save your map with 'map name'.save('filename.html')

# map.save('CAS-records-georeferenced.html')
```

Jupyter notebook for mapping, available at QR code at left



All CAS localities georeferenced, 2021

LESSONS LEARNED

1. We all want our specimens digitized, and **spreadsheet data is better than none** in the absence of a formal content management system!
   a. This is a good start. Sheet data can be manipulated and **ingested into a database later.**
2. Solve as many downstream data integrity issues as possible by **setting data standards and controlled vocabularies at the beginning of digitization efforts.**

FUTURE DIRECTIONS

1. Revisit the **automation of taxonomic and lithostratigraphic backfilling** from data fragments entered in specimen and locality records.
2. Better integrate **specimen images** and scans of **locality ledger documents.**
3. Set future digitization priorities and implement **standardized vocabularies** developed during this process**.**

ACKNOWLEDGEMENTS
Big thanks to Peter Roopnarine & Chrissy Garcia for their support and mentorship; to Marie Angel, Rose De Guzman, Ruigie Arevalo, Cassie Ettinger, & Jasmine Nguyen for feedback; and to Alice Chang & Jason Konrad for their help when I was very stuck on programming problems.