

# Open-Source Email Curation Software Designed for Reusability

---

**Christopher (Cal) Lee and Kam Woods**

UNC Chapel Hill School of Information and Library Science

17th International Digital Curation Conference

June 14, 2022



UNC  
SCHOOL OF INFORMATION  
AND LIBRARY SCIENCE

VIEW THESE SLIDES AT: <https://bit.ly/idcc-ratom-2022>

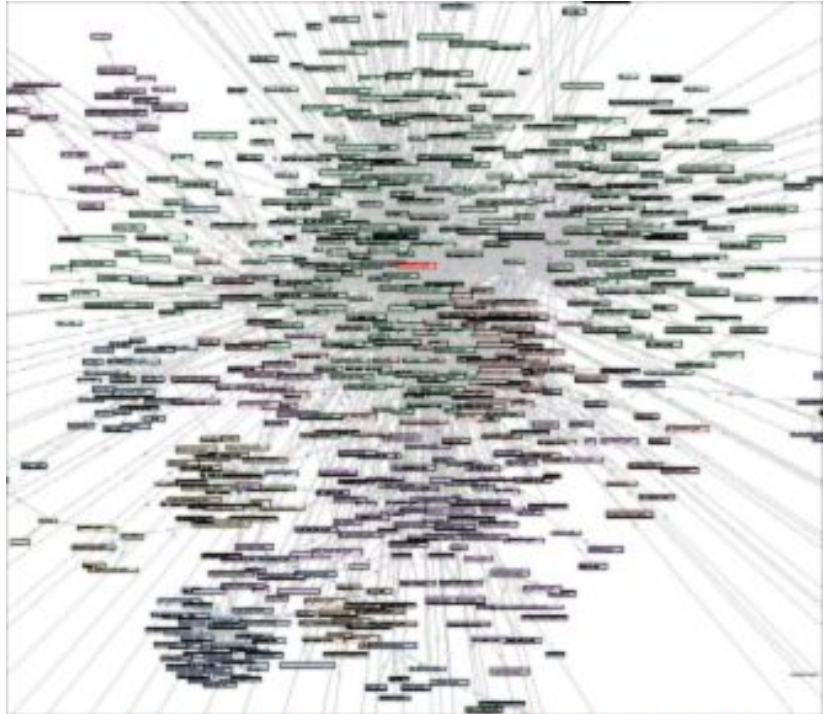


# Motivation - Selection/Appraisal

- ❖ Despite progress on various technologies to support data management and digital preservation of email, software support for the core activities of selection and appraisal remains limited
- ❖ Selection/appraisal decisions are based on various patterns
- ❖ When patterns can be identified algorithmically, software can assist the process
- ❖ LAMs frequently want to take actions that reflect contextual relationships
- ❖ Timeline representations and visualizations can also provide useful, high-level views of materials

# Motivation - Email

- ❖ About 50 years of email creation
- ❖ Hundreds of billions of messages generated every day
- ❖ Most has little long-term retention value, but some absolutely does
- ❖ Despite presence of numerous other modalities, email still deeply embedded in activities, serving as massive source of evidence and information
- ❖ Often found in collections and acquisitions with other types of materials



# Review, Appraisal, and Triage of Mail (RATOM)

- ❖ Funded by Andrew W. Mellon Foundation (2019-2021)
- ❖ Developing and repurposing software (including NLP and machine learning) for selection/appraisal in BitCurator environment with hooks and enhancements to TOMES output
- ❖ Support iterative processing - information discovered at various points in the processing workflow can support further selection, redaction or description actions
- ❖ Mapping of timestamp, entity, sensitive features and other elements across the tools



**Ray Tomlinson**

Implemented first email program on ARPANET.  
Credited with invention of first email system.

# Team Members



**Cal Lee**  
PI



**Antoine De Torcy**  
Software Engineer



**Eliscia Kinder**  
Project Manager



**Kam Woods**  
Technical Lead (UNC)



**Camille Tyndall Watson**  
Co-PI



**Jamie Patrick-Burns**  
Investigator



**Sangeeta Desai**  
Technical Lead (NC DAR)



**Cactus Group**  
Software Development

# Scope of the project

The RATOM project has had specific development goals designed to serve the needs of collecting institutions tasked with preparing email collections for public access:

- Development of an integrated Python library to simplify parsing and processing **PST**, **OST**, and **mbox** email formats
- Building utilities to support entity identification and export reports suitable for conducting automated and human-directed redaction actions at scale
- Creating an interface allowing processing archivists to browse email collections and mark messages as suitable for retention
- Exploring methods apply machine learning techniques (by training on annotated message collections) to recognize candidate materials for retention

# RATOM tools - libratom

Python library to parse and analyze **PST, OST,** and **mbox** email formats

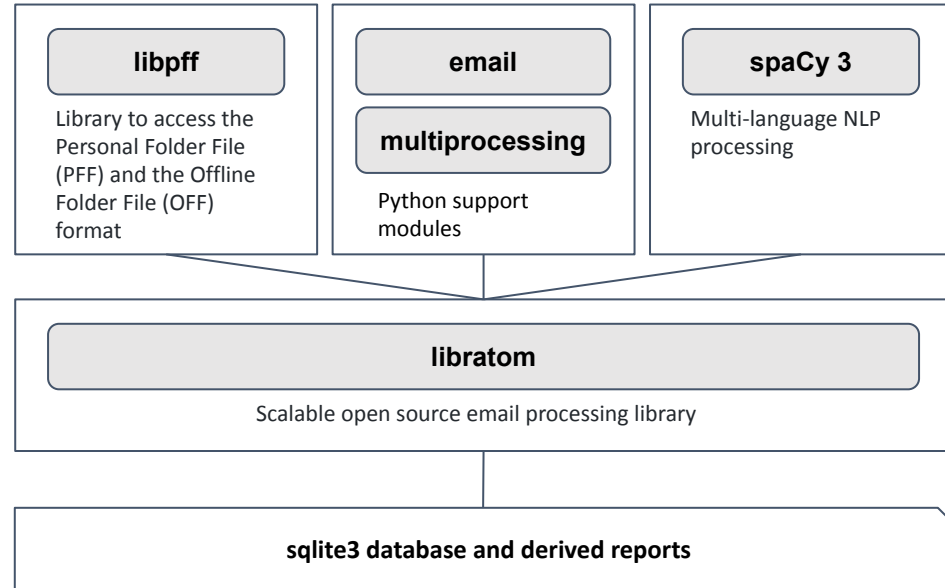
Wraps functionality from **libpff**, Python **mailbox**, and **spaCy 3**

Extracts and analyzes email message content and headers; **identifies and classifies entities** identified in email bodies via **spaCy**

Performance scales with processor core count; per-core memory usage never exceeds 1.6GB in default configuration, irrespective of input size

Current release: 0.6.0

<https://www.github.com/libratom/libratom>



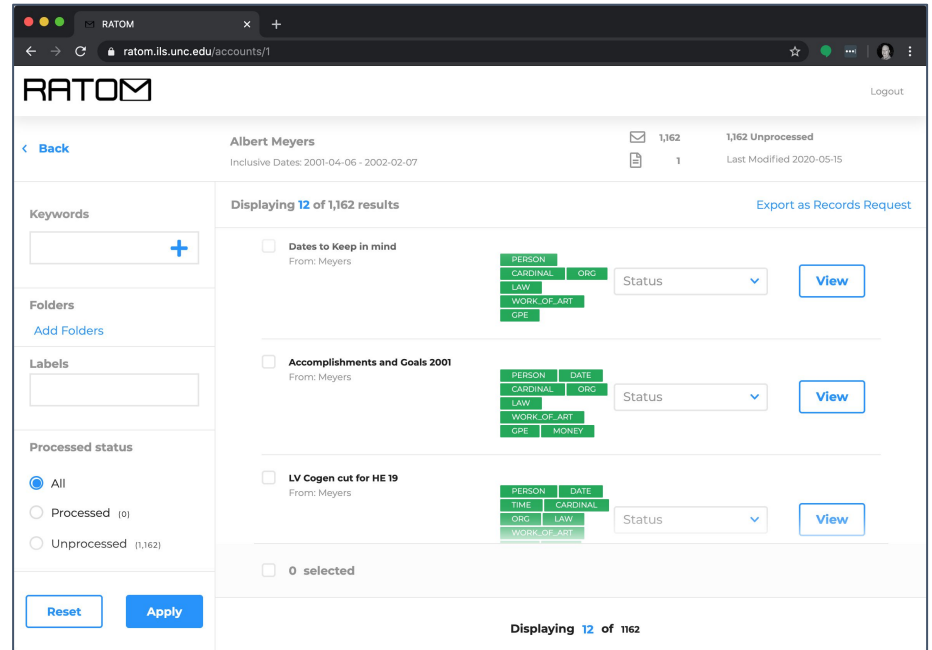
Features generated and stored are **VERIFIABLE, REPRODUCIBLE, AND REUSABLE**

This facilitates understanding changes in assessments of materials over time.

# RATOM tools - Iterative Processing Interface

Assist archivists in reviewing email materials for retention and/or release.

- ❖ Import of email accounts from PSTs and entity identification via libratom
- ❖ Creation of processing accounts associated with individual email users
- ❖ Interactive review and tagging of email messages within these accounts (e.g. “record”, “non-record”, “redact”)
- ❖ Export of selected messages as EML for retention or release



<https://github.com/StateArchivesOfNorthCarolina/ratom-deploy>



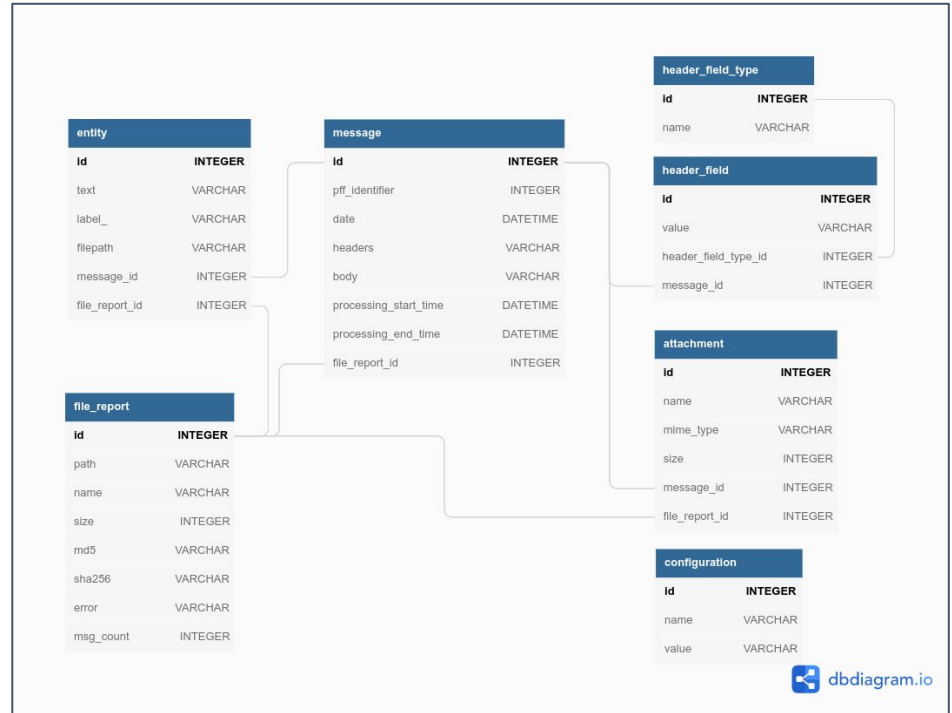
# Data analysis and comparison across time and corpora versions

## Engineering for extensibility and ease of maintenance

**libratom** uses SQLAlchemy for SQL / ORM handling, decoupling the underlying data model from the schema and sqlite3 databases generated by the CLI tool

## A schema to support common use cases

The current schema focuses on initial triage and assessment, providing a map of 18 unique entity types to their locations (message and file) within a corpus.



entity	
id	INTEGER
text	VARCHAR
label_	VARCHAR
filepath	VARCHAR

Extracted entity instances (18 categories via spaCy)

message	
id	INTEGER
pff_identifier	INTEGER
date	DATETIME
headers	VARCHAR
body	VARCHAR
processing_start_time	DATETIME
pro	
file	

Message metadata (optionally contains message body)

header_field_type	
id	INTEGER
name	
header	
id	INTEGER
value	VARCHAR
header_field_type_id	INTEGER
message_id	INTEGER

Extracted header fields and values

file_report	
id	INTEGER
path	VARCHAR
name	VARCHAR
size	INTEGER
md5	
sha25	
error	
msg_count	INTEGER

Originating file metadata

attachment	
id	INTEGER
name	VARCHAR
mim	
size	
mes	
file_	
con	
id	
name	VARCHAR
value	VARCHAR

Attachment metadata, MIME types validated against IANA content types

883381 | **the Department of Environmental Protection** | ORG | david\_delainey\_000\_1\_2.pst | 2325380  
 883382 | **Cellucci** | PERSON | david\_delainey\_000\_1\_2.pst | 2325380  
 883383 | **the United States** | GPE | david\_delainey\_000\_1\_2.pst | 2325380  
 883384 | **five** | CARDINAL | david\_delainey\_000\_1\_2.pst | 2325380  
 883385 | **six** | CARDINAL | david\_delainey\_000\_1\_2.pst | 2325380  
 883386 | **Jane Swift** | PERSON | david\_delainey\_000\_1\_2.pst | 2325380  
 883387 | **the Department of Environmental Protection** | ORG | david\_delainey\_000\_1\_2.pst | 2325380  
 883388 | **six** | CARDINAL | david\_delainey\_000\_1\_2.pst | 2325380  
 883389 | **the next few months** | DATE | david\_delainey\_000\_1\_2.pst | 2325380  
 883390 | **1.5** | CARDINAL | david\_delainey\_000\_1\_2.pst | 2325380  
 883391 | **3 pounds** | QUANTITY | david\_delainey\_000\_1\_2.pst | 2325380  
 883392 | **megawatt-hour** | TIME | david\_delainey\_000\_1\_2.pst | 2325380  
 883393 | **five** | CARDINAL | david\_delainey\_000\_1\_2.pst | 2325380  
 883394 | **Sithe Energies, Inc.** | ORG | david\_delainey\_000\_1\_2.pst | 2325380

Results with the smallest pre-trained model available.  
 Accuracy is closer to 90% when using the most recent  
 transformer-based models

Model: Spacy  
 en\_core\_web\_sm, trained on  
 OntoNotes 5, accuracy  
 evaluation:

ENTS_P	Named entities (precision)	0.84
ENTS_R	Named entities (recall)	0.83
ENTS_F	Named entities (F-score)	0.84

With the libratom CLI, we can **load different models** (including user trained models) **on demand** for tasks / languages

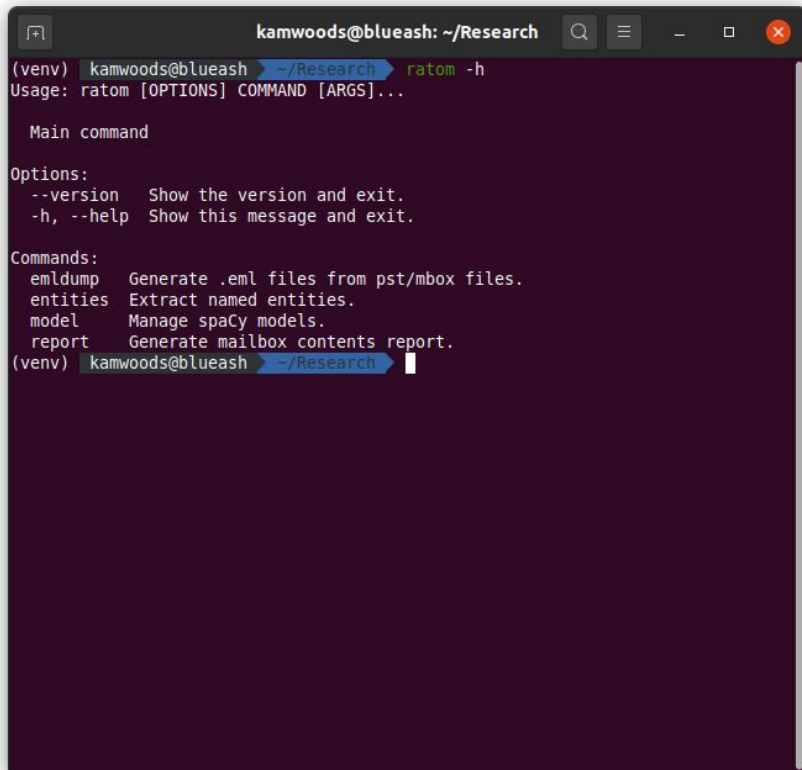
# libratom commands

**entities** command generates structured db output from mbox, PST, and OST sources, performing a full source scan and NER processing

**model** command provides granular control of entity identification model(s) in use, including access to previously released models

**report** command generates a fast report, populating the sqlite3 schema without entities (but optionally including message text and headers)

**emldump** provides a mechanism for generating EML files using JSON structured message id lists produced by the web app (may also be used standalone)



```
kamwoods@blueash: ~/Research
(venv) kamwoods@blueash ~/Research ratom -h
Usage: ratom [OPTIONS] COMMAND [ARGS]...

Main command

Options:
  --version  Show the version and exit.
  -h, --help Show this message and exit.

Commands:
  emldump  Generate .eml files from pst/mbox files.
  entities Extract named entities.
  model    Manage spaCy models.
  report   Generate mailbox contents report.
(venv) kamwoods@blueash ~/Research
```

```
kamwoods@blueash: ~/Research
(venv) kamwoods@blueash ~/Research ratom entities -h
Usage: ratom entities [OPTIONS] [SOURCE]

Extract named entities from a PST or mbox file, or a directory of one or
more PST and mbox files.

If SOURCE is a directory it will be walked recursively. Non-PST and non-
mbox files will be skipped.

Upon success the result will be a new .sqlite3 database file. If an output
path is provided it will be either the output file's parent directory or
the file itself.

If no output path is provided the file will be written in the current
working directory.

Options:
  -o, --out <path>          Write the output to <path>.
  --spacy-model <model>     Use a given spaCy model to extract entities.
  -m, --include-message-contents Also extract message headers and bodies.
  -j, --jobs <n>            Use <n> concurrent jobs.
  -v, --verbose              Increase verbosity (can be repeated).
  -p, --progress            Show progress.
  -h, --help                Show this message and exit.
(venv) kamwoods@blueash ~/Research
```

```
kamwoods@blueash: ~/Research
(venv) kamwoods@blueash ~/Research ratom model -h
Usage: ratom model [OPTIONS]

Manage spaCy models.

List, install, or upgrade currently installed models. Use the -i and
--version flags together to install a previously released version of a
specific model.

Options:
  -l, --list           List spaCy models.
  -i, --install <model> Install <model>.
  -u, --upgrade <model> Upgrade <model>.
  --version <version> If used alongside -i/--install, install a given model
                       version. Otherwise this has no effect.

  -h, --help          Show this message and exit.
(venv) kamwoods@blueash ~/Research
```

Note: This functionality has also been factored out as a standalone tool for use with other spaCy-dependent projects. See <https://github.com/carascap/spacy-model-manager>

```
(venv) kamwoods@azalea:~/Research$ ratom model -l
spaCy model          installed version  available versions
-----
da_core_news_lg      3.0.0, 2.3.0
da_core_news_md      3.0.0, 2.3.0
da_core_news_sm      3.0.0, 2.3.0
de_core_news_lg      3.0.0, 2.3.0
de_core_news_md      3.0.0, 2.3.0, 2.2.5, 2.2.1, 2.2.0, 2.1.0, 1.0.0
de_core_news_sm      3.0.0, 2.3.0, 2.2.5, 2.2.1, 2.2.0, 2.1.0, 2.0.0
de_dep_news_trf      3.0.0
de_pytt_bertbasecased_lg  2.1.1, 2.1.0
de_trf_bertbasecased_lg  2.3.0, 2.2.0
el_core_news_lg      3.0.0, 2.3.0
el_core_news_md      3.0.0, 2.3.0, 2.2.5, 2.2.3, 2.2.2, 2.2.0, 2.1.0
el_core_news_sm      3.0.0, 2.3.0, 2.2.5, 2.2.3, 2.2.2, 2.2.0, 2.1.0
en_core_web_lg       3.0.0, 2.3.1, 2.3.0, 2.2.5, 2.2.0, 2.1.0, 2.0.0
en_core_web_md       3.0.0, 2.3.1, 2.3.0, 2.2.5, 2.2.0, 2.1.0, 2.0.0, 1.2.1, 1.2.0
en_core_web_sm       3.0.0, 2.3.1, 2.3.0, 2.2.5, 2.2.0, 2.1.0, 2.0.0, 1.2.0
en_core_web_trf      3.0.0
en_depvec_web_md     2.2.1, 1.2.0
en_pytt_bertbaseuncased_lg  2.1.1, 2.1.0
en_pytt_distilbertbaseuncased_lg  2.1.0
en_pytt_robertabase_lg  2.1.0
en_pytt_xlnetbasecased_lg  2.1.1
en_trf_bertbaseuncased_lg  2.3.0
en_trf_distilbertbaseuncased_lg  2.3.0
en_trf_robertabase_lg  2.3.0
en_trf_xlnetbasecased_lg  2.3.0
en_vectors_glove_md  1.0.0
en_vectors_web_lg    2.3.0
es_core_news_lg      3.0.0
es_core_news_md      3.0.0
es_core_news_sm      3.0.0
es_core_web_md       1.0.0
es_dep_news_trf      3.0.0
fr_core_news_lg      3.0.0
fr_core_news_md      3.0.0
fr_core_news_sm      3.0.0, 2.3.0, 2.2.5, 2.2.0, 2.1.0, 2.0.0
fr_dep_news_trf      3.0.0
fr_depvec_web_lg    1.0.0
it_core_news_lg      3.0.0, 2.3.0
it_core_news_md      3.0.0, 2.3.0
it_core_news_sm      3.0.0, 2.3.0, 2.2.5, 2.2.0, 2.1.0, 2.0.0
```

Transformer-based models in spaCy 3.x provide better NER accuracy. The `en_core_web_sm` model is loaded by default when running the CLI provided by `libratom 0.6.x` (for minimum download overhead) but any other available model can be loaded on demand.

[https://spacy.io/models/en#en\\_core\\_web\\_trf](https://spacy.io/models/en#en_core_web_trf)

# libratom 0.6.0 processing the Gov. Kaine (Library of Virginia) sample corpus

Kaine sample corpus: Approximately **12.3GB**, includes **1 PST file** containing **79,538 messages**

**Entity extraction from all 79.5K messages (spaCy en\_core\_web\_sm model):**

16-core Threadripper 2950X: **~24 minutes**

32-core Threadripper 3970X: **~13 minutes**

Memory usage is bounded for the **spaCy** configuration and number of processes. For 32 processes, accessible memory is **~1.6GB/process**, resident memory is **~500MB/process** on average.

In libratom 0.6.0 , this run yields a **547MB sqlite3** file (including plaintext message bodies and parsed headers), containing **3,260,781** entity instances.

<https://www.virginiamemory.com/collections/kaine/>

```
Processing messages 100% | ██████████ | 79538/79538 [13:16<00:00, 99.94 msg/s]
Generating message reports 100% | ██████████ | 79538/79538 [13:16<00:00, 99.94 msg/s]
```



# Exploring the output (LVA Kaine corpus sample)

## Example 1:

Entity groups by type, ordered by count.

```
sqlite> select count(*), label_
from entity group by label_
order by count(*) DESC;
1288056|PERSON
686345|ORG
444841|DATE
362347|CARDINAL
257576|GPE
173187|TIME
60158|MONEY
32378|NORP
31311|ORDINAL
27683|FAC
27122|LOC
24327|PERCENT
23698|WORK_OF_ART
22738|PRODUCT
14511|LAW
11194|EVENT
8163|QUANTITY
586|LANGUAGE
sqlite>
```

## Example 2:

Individual text elements identified by spaCy as "PERSON" that appear more than 10,000 times

Note that the NER processor **will** return full names as entities when they are encountered.

This particular group of entities was simply mentioned only by first or last name a large number of times in the collection.

Entity  $\neq$  Identity!

```
sqlite> select count(*), text from
entity where label_ = 'PERSON'
group by text having count(*) >
10000 order by count(*) DESC;
41480|Gail
35978|Marilyn
29690|Jaspen
19033|Tavanner
18462|Bill
17264|Barbara
17077|Mark
13858|Wayne
13434|Brian
12723|Rubin
10933|Craig
10767|Leighty
10761|Burns
10371|Heidi
sqlite>
```

# Exploring the output (LVA Kaine corpus sample)

## Example 3:

We can use this database to explore data that might pose issues for processing.. For example:

All attachments with identical names that appear in the collection more than 60 times

```
sqlite> select count(*), name
from attachment group by name
having count(*) > 60 order by
count(*) DESC;
1005|image001.jpg
729|image001.gif
303|Document.pdf
161|Scan001.PDF
157|image002.jpg
152|image002.gif
108|IQFormatFile.txt
85|Blank Bkgrd.gif
82|Jane Woods.vcf
79|image003.gif
74|Karen Remley
(karen.remley@vdh.virginia.gov
).vcf
68|Robert A Nebiker.vcf
64|image003.jpg
63|sg-0.gif
61|janderson.vcf
```

## Example 4:

Duplicate names don't always mean duplicate files.

Refining the previous query, we get a better sense of the degree of (possible) duplication.

Of the 1005 “image001.jpg” files identified in the original query, 262 have an identical size of 2,950 bytes. 61 have an identical size of 3,126 bytes.

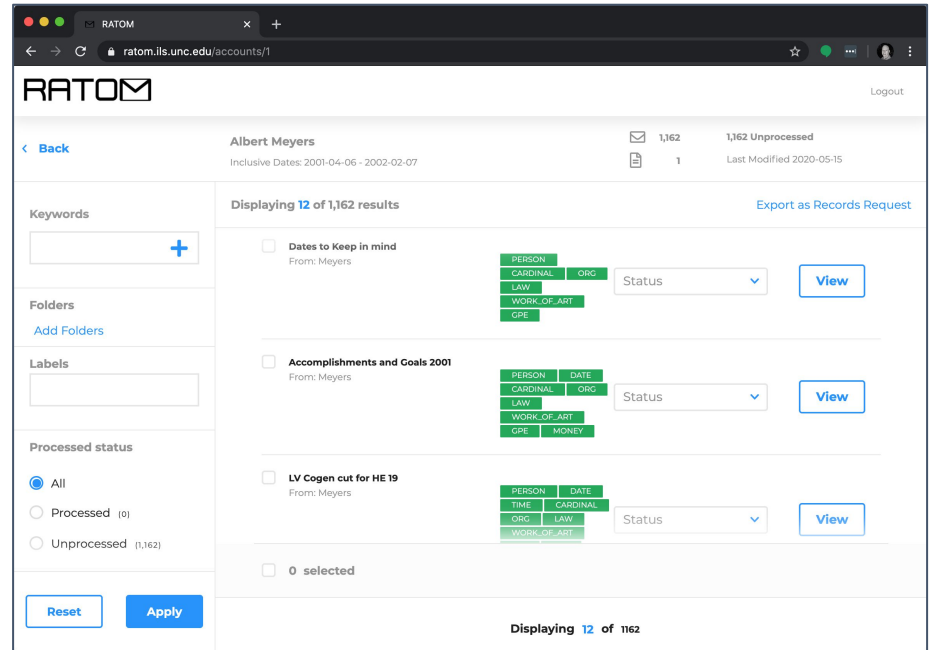
Of the 303 “Document.pdf” files identified in the original query, 27 have an identical size of 710,730 bytes.

```
sqlite> select count(*), name, size
from attachment group by name, size
having count(*) > 25 order by count(*)
DESC;
262|image001.jpg|2950
134|image001.gif|1564
106|IQFormatFile.txt|199
85|Blank Bkgrd.gif|145
82|Jane Woods.vcf|163
80|image001.gif|3725
79|image001.jpg|2743
74|Karen Remley
(karen.remley@vdh.virginia.gov).vcf|51
9
68|Robert A Nebiker.vcf|574
...
35|image002.gif|73
34|jim.burns.vcf|348
27|Document.pdf|710730
27|Glazer, Melinda.vcf|294
```

# RATOM tools - Iterative Processing Interface

Assist archivists in reviewing email materials for retention and/or release.

- Import of email accounts from PSTs and entity identification via libratom
- Creation of processing accounts associated with individual email users
- Interactive review and tagging of email messages within these accounts (e.g. “record”, “non-record”, “redact”)
- Export of selected messages as EML for retention or release



<https://github.com/StateArchivesOfNorthCarolina/ratom-deploy>

## Accounts View

- Allows users to view all accounts loaded into the interface
- Multiple .psts can be associated with an individual account
- Allows view of inclusive dates, number of messages (including number processed), number of .psts associated with the account, and last modified date on one screen

The screenshot shows a web browser window with the URL `ratom.ils.unc.edu`. The page title is "RATOM" and there is a "Logout" link in the top right corner. The main heading is "My Accounts" with a "New Account" button. Below this is a table listing six accounts with their details.

Account Name	Inclusive Dates	Complete	Messages	Unprocessed	Last Modified	More
Albert Meyers	2001-04-06 - 2002-02-07	Complete	1,162	1,162 Unprocessed	2020-05-15	...
Andrea Ring	2000-06-06 - 2002-11-30	Complete	1,013	1,013 Unprocessed	2020-05-15	...
Andrew Lewis	1980-01-01 - 2002-02-07	Complete	2,668	2,668 Unprocessed	2020-05-15	...
Diana Scholtes	2001-04-04 - 2002-02-07	Complete	707	707 Unprocessed	2020-05-15	...
Don Baughman	1980-01-01 - 2002-11-30	Complete	4,099	4,099 Unprocessed	2020-05-15	...
Drew Fossum	1980-01-01 - 2002-11-30	Complete	9,297	9,297 Unprocessed	2020-05-15	...

## Individual Account

- Selecting an account displays an infinite-scroll view of individual messages associated with that account.
- Green tags indicate entity classes identified during processing.
- Blue tags indicate user-generated classes
- Status dropdown allows messages to be marked for retention or redaction (also appears in individual message view).

The screenshot shows the RATOM web application interface. At the top, the browser address bar displays 'ratom.iils.unc.edu/accounts/2'. The RATOM logo is in the top left, and a 'Logout' link is in the top right. The main header for the account 'Andrea Ring' shows '1,013' messages and '1,013 Unprocessed' items, with a 'Last Modified' date of 2020-05-15. A 'Back' link is on the left, and an 'Export as Records Request' link is on the right. The left sidebar contains filters: 'Keywords' (with a search box and a plus icon), 'Folders' (with an 'Add Folders' link), 'Labels' (with a search box), and 'Processed status' (with radio buttons for 'All', 'Processed (0)', and 'Unprocessed (1,013)', and 'Reset' and 'Apply' buttons). The main content area shows 'Displaying 96 of 1,013 results'. It lists four messages, each with a checkbox, a subject line, a 'From' field, a table of green tags, a 'Status' dropdown, and a 'View' button. The messages are: 1) 'Re: 9/00 Purchase from Koch Energy Trading at Sabine Hub' with tags DATE, CARDINAL, ORG, LAW, WORK\_OF\_ART; 2) 'Re: 9/00 Purchase from Koch Energy Trading at Sabine Hub' with the same tags; 3) 'Re: Happy Hour for Jennifer Fraser & Sarah Mulholland' with tags PERSON, DATE, ORG, LAW, WORK\_OF\_ART; 4) 'Re: Happy Hour for Jennifer Fraser & Sarah Mulholland' with the same tags. At the bottom, it says '0 selected' and 'Displaying 96 of 1013'.

## Message View

- Header information
- Libratom generated tags (green)
- User created tags (blue)
- Message body (can be viewed as plain text or with HTML markup)
- Attachments are indicated by file name

The screenshot shows a web browser window with the address bar displaying `ratom.ils.unc.edu/accounts/2/messages/686`. The page title is "RATOM" and the logo is visible in the top left. The interface includes a "Back" button, a "37 of 1013" indicator, a "View as plain-text" checkbox (checked), and a "Status" dropdown menu. The email header shows the subject "Re: 8/00 Purchase from Koch Energy Trading at Sabine (Henry Hub) - Sitara Deal" and the date "Sep 19, 2000 11:04 PM". The "To" field is "Michael Mousteiko" and the "From" field is "Andrea Ring". Below the header, there are green tags: "PERSON", "CARDINAL", "ORG", "LAW", "WORK\_OF\_ART", "GPE", and "MONEY", along with a "+ Add Label" link. A breadcrumb trail reads "> Top of Personal Folders > ring-a > Andrea\_Ring\_Jun2001 > Notes Folders > Sent". The message body is separated from the header by a dashed line labeled "START MESSAGE BODY". The body text reads: "I do not show I did any deals outside of EOL with Koch at the Sabine Hub during this time frame. Sitara deal #369260 refers to EOL deal ID 369298." This is followed by a block of text enclosed in asterisks: "\*\*\*\*\*  
EDRM Enron Email Data Set has been produced in EML, PST and NSF format by ZL Technologies, Inc. This Data Set is licensed under a Creative Commons Attribution 3.0 United States License <http://creativecommons.org/licenses/by/3.0/us/>. To provide attribution, please cite to 'ZL Technologies, Inc. (http://www.zlti.com).'  
\*\*\*\*\*". The message body is separated from the footer by a dashed line labeled "END MESSAGE BODY". The footer shows "37 of 1013".

## Tagging and Search

- Fields Searched:
- Keyword
- Folder
- Label
- Processing status (processed v. unprocessed)
- Record status (open, restricted, needs redaction, non-record)
- Email addresses
- Date range

The screenshot shows the RATOM web interface. The browser address bar displays 'ratom.ils.unc.edu/accounts/2'. The page header includes the RATOM logo and a 'Logout' link. The main content area is titled 'Andrea Ring' and shows search statistics: 1,013 total records, 1,013 unprocessed, and 1 document. A sidebar on the left contains search filters, with the 'Record status' section circled in red. The 'Record status' section includes radio buttons for 'All' (selected), 'Open (0)', 'Restricted (0)', 'Needs redaction (0)', and 'Non-record (0)'. Below this are fields for 'Email addresses' and 'From:' (with a date range 'YYYY-MM-DD'). At the bottom of the sidebar are 'Reset' and 'Apply' buttons. The main results area displays 'Displaying 24 of 1,013 results' and an 'Export as Records Request' link. The results list shows four email entries, each with a checkbox, subject line, sender, tags (PERSON, ORG, LAW, WORK\_OF\_ART, DATE), a 'Status' dropdown, and a 'View' button. The bottom of the results area shows '0 selected' and 'Displaying 24 of 1013'.

# Audit History

- Audit histories for individual messages are retained.
- This ensures a clear record of initial processing actions and potential changes over time.
- Includes action taken and who performed the action

The screenshot shows a web browser window displaying the Django administration interface for message audits. The browser's address bar shows the URL `ratom.ils.unc.edu/admin/core/messageaudit/`. The page header includes the Django logo and the text "Django administration" on the left, and a welcome message "WELCOME, RATOM@PROTONMAIL.COM" with links for "VIEW SITE / CHANGE PASSWORD / LOG OUT" on the right. Below the header, a breadcrumb trail reads "Home > Core > Message audits".

The main content area is titled "Select message audit to change" and features a search bar with a magnifying glass icon and a "Search" button. Below the search bar, there is an "Action:" dropdown menu set to "-----" and a "Go" button, with the text "0 of 100 selected" to its right.

The central part of the page is a table with the following columns: "PK", "MESSAGE", "PROCESSED", "IS RECORD", "DATE PROCESSED", and "UPDATED BY". The table contains 15 rows of data, each with a checkbox in the "PK" column. The "PROCESSED" column contains red 'x' icons, and the "IS RECORD" column contains green checkmarks. The "DATE PROCESSED" and "UPDATED BY" columns are empty for all rows.

On the right side of the page, there is a "FILTER" sidebar with three sections: "By is record", "By processed", and "By account". Each section has a dropdown menu with "All", "Yes", and "No" options. The "By account" section is currently expanded, showing a list of names: Albert Meyers, Andrea Ring, Andrew Lewis, Diana Scholtes, Don Baughman, Drew Fossum, and Dutch Quigley.

In the top right corner of the main content area, there is a button labeled "ADD MESSAGE AUDIT +" with a plus sign icon.





Project info, news, and blog posts:

<https://ratom.web.unc.edu/>

Core library:

<https://github.com/libratom/libratom>

Sample Jupyter notebooks:

<https://github.com/libratom/ratom-notebooks>

Web Service + Interface:

<https://github.com/StateArchivesOfNorthCarolina/ratom-deploy>



@RATOM\_Project

A screenshot of the RATOM website. The header features the RATOM logo and the tagline "Review, Appraisal, and Triage of Mail". A navigation menu includes links for Home, About, Tools and Code, Presentations, ML4ARC, Hackathon, Project Personnel, and Advisory Board. The main content area is divided into two columns. The left column features a "Latest release: libratom 0.5.4" section with a profile picture of karnwoods, the date September 22, 2021, and an edit link. The text describes the release, including testing with spaCy 3.1 and the use of GitHub Actions. The right column has a "SOFTWARE AND SOCIAL" section with social media icons for GitHub and Twitter, and a "RECENT POSTS" section listing recent updates and events like the RATOM Webinar and Hackathon.

# Releases and Code Quality

## Updates and improvements as of 0.6.0:

Releases have been generated in tandem with tags on GitHub main branch, tracking all minor and patch updates (currently 0.6.0).

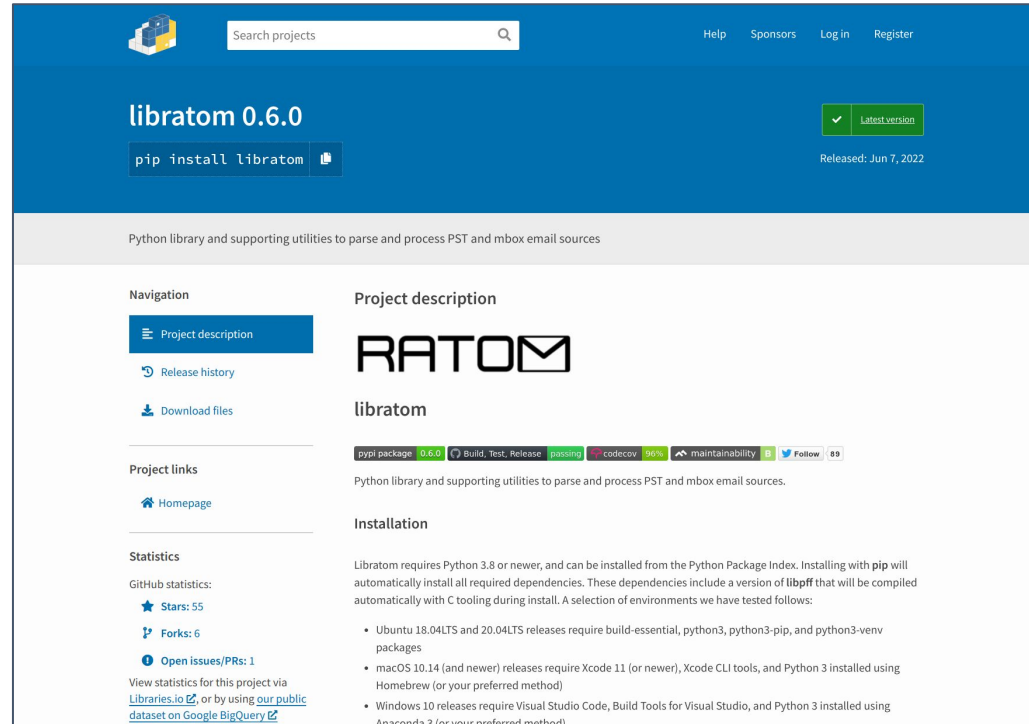
All releases automatically pushed to PyPI.

CI via GitHub Actions, multi-platform support, Python 3.8, 3.9, and 3.10.

Codebase tracked with codeclimate to assess maintainability

Code coverage tracked via codecov (currently 96%) - effectively all core code is exercised by the test suite

Routine dependency tree checks via dependabot



The screenshot shows the PyPI page for the 'libratom' package. At the top, there's a search bar and navigation links for Help, Sponsors, Log in, and Register. The package name 'libratom 0.6.0' is prominently displayed, along with a 'pip install libratom' button and a 'Latest version' badge. Below this, the package description reads: 'Python library and supporting utilities to parse and process PST and mbox email sources'. The page is divided into several sections: 'Navigation' with links for Project description, Release history, and Download files; 'Project links' with a link to the Homepage; 'Statistics' showing GitHub stats (Stars: 55, Forks: 6, Open issues/PRs: 1) and a link to view statistics on Libraries.io or Google BigQuery; 'Project description' featuring the 'RATOM' logo and a detailed description of the library's purpose and dependencies; and 'Installation' providing instructions for installing the package on various operating systems (Ubuntu, macOS, Windows) and environments (pip, Homebrew, Visual Studio Code, Anaconda).