Tidy Humanities Data

Matthew Lincoln

May 28, 2019

Making tidy humanities data

The problem

- Text, network, and other quantitative analyses need data supplied in one neat table

The problem

- Text, network, and other quantitative analyses need data supplied in one neat table
- All real-life historical data are more complicated than that.

What we'll learn

- Structuring our sources as “tidy” data for future analysis
  - Categorization
  - Multiple values
  - Handling dates
  - Missing or uncertain data
- Example queries
- Practical exercise

What is tidy data?

1. One variable per column
2. One observation per row
3. Consistent data types
4. If you can't do 1 and 2, that means you need an extra table

Messy (but well-intentioned!) data

acq_no	museum	artist	date	medium	tags
1999.32	Rijksmuseum	Studio of Rembrandt, Govaert Flinck	after 1636	oil on canvas	religious, portrait
1908.54	Victoria & Albert	Jan Vermeer	c. 1650	oil paint on panel	domestic scene and women
1955.32	British Museum	Possibly Vermeer, Jan	c. 1655	oil on canvas	woman at window, portrait
1955.33	Rijksmuseum	Hals, Frans	16220	oil on canvas, relined	portraiture

Categorization

What concepts matter to you?

What bits of your free text fields could be systematized?

Figure out the *grammar* of your data (how it fits together) and define a *vocabulary* (what the individual pieces are)

It's easy to combine things back together after collecting data; it's hard to split them out.

Example 1

acq_no	medium
1999.32	oil on canvas
1908.54	oil paint on panel
1955.32	oil on canvas
1955.33	oil on canvas, relined

Here, `medium` contains info on:

- painting medium (what it's painted *with*)

Example 1

acq_no	medium
1999.32	oil on canvas
1908.54	oil paint on panel
1955.32	oil on canvas
1955.33	oil on canvas, relined

Here, `medium` contains info on:

- painting medium (what it's painted *with*)
- painting support (what it's painted *on*)

Example 1

acq_no	medium
1999.32	oil on canvas
1908.54	oil paint on panel
1955.32	oil on canvas
1955.33	oil on canvas, relined

Here, `medium` contains info on:

- painting medium (what it's painted *with*)
- painting support (what it's painted *on*)
- conservation techniques

Example 1

acq_no	medium	support	cons_note
1999.32	oil	canvas	
1908.54	oil	panel	
1955.32	oil	canvas	
1955.33	oil	canvas	relined

- separated different concepts into columns
- standardized vocabulary for each concept
    - keep the differences that are important, get rid of the ones that aren't

Example 2

acq_no	artist
1999.32	Studio of Rembrandt, Govaert Flinck
1908.54	Jan Vermeer
1955.32	Possibly Vermeer, Jan
1955.33	Hals, Frans

- artist column tells us more than just a name - it also has qualifiers.

Example 2

One possibility:

acq_no	artist_1_name	artist_1_qual	artist_2_name	artist_2_qual
1999.32	Rembrandt	studio	Govaert Flinck	
1908.54	Jan Vermeer			
1955.32	Jan Vermeer	possibly		
1955.33	Frans Hals			

- separate the qualifiers from the artist name
- standardize the names so the same person is spelled consistently

Example 2

One possibility:

acq_no	artist_1_name	artist_1_qual	artist_2_name	artist_2_qual
1999.32	Rembrandt	studio	Govaert Flinck	
1908.54	Jan Vermeer			
1955.32	Jan Vermeer	possibly		
1955.33	Frans Hals			

- Should we split out the first / last names too?
    - only if you need to for your research!
- Now we're dealing with complicated multiple values. . .

Multiple Values

Multiple Values

Spreadsheets look like they just hold one value per cell. Often our variables have a many-to-one or many-to-many relationship.

2 strategies for this:

- use a delimiter (`;`, `,`, `|`) to put together quick small labels and tags into one cell together

Multiple Values

Spreadsheets look like they just hold one value per cell. Often our variables have a many-to-one or many-to-many relationship.

2 strategies for this:

- use a delimiter (`;`, `,`, `|`) to put together quick small labels and tags into one cell together
- for complicated info, you need a *related table*

Example 1: delimiters

acq_no	tags
1999.32	religious, portrait
1908.54	domestic scene and women
1955.32	woman at window, portrait
1955.33	portraiture

These tags are self-contained (they don't have lots of related info - the term IS the data).

Example 1: delimiters

acq_no	tags
1999.32	religious;portrait
1908.54	domestic scene;woman
1955.32	woman;window;portrait
1955.33	portrait

- Standardize each individual tag
- Use a common delimiter to keep them separate

Example 1: delimiters

If we encode this data correctly, then once we start processing it, we can easily "pivot" those data into the format we need to e.g. retrieve the paintings with the tag `woman`.

```
clean_data_tags %>%
  separate_rows(tags, sep = ";")
```

acq_no	tags
1999.32	religious
1999.32	portrait
1908.54	domestic scene
1908.54	woman
1955.32	woman
1955.32	window
1955.32	portrait
1955.33	portrait

Example 1: delimiters

Once it's in this format, we can now filter to get just the paintings we want based on tag.

```
clean_data_tags %>%
  separate_rows(tags, sep = ";") %>%
  filter(tags == "woman")
```

acq_no	tags
1908.54	woman
1955.32	woman

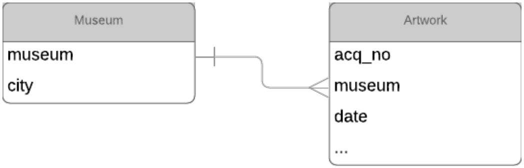
Example 2: Related table

Sometimes our objects make reference to things like people or places that, themselves, have many attributes.

In these cases, it's not enough to just use delimiters - we actually need to have a *related table* that can hold this additional information without needing to repeat it again and again.

Many-to-one

One museum can own many objects, but one object belongs to only one museum



museum	city
Rijksmuseum	Amsterdam
Victoria & Albert	London
British Museum	London

Many-to-one

We perform a *left join* (you'll see this in R, python, SQL, pretty much everywhere) to copy attributes from the museums onto the objects.

```r
combined_data <- messy_data %>%
  left_join(museums, by = "museum")
```

acq_no	museum	artist	date	medium	tags	city
1999.32	Rijksmuseum	Studio of Rembrandt, Govaert Flinck	after 1636	oil on canvas	religious, portrait	Amsterdam
1908.54	Victoria & Albert	Jan Vermeer	c. 1650	oil paint on panel	domestic scene and women	London
1955.32	British Museum	Possibly Vermeer, Jan	c. 1655	oil on canvas	woman at window, portrait	London
1955.33	Rijksmuseum	Hals, Frans	16220	oil on canvas, relined	portraiture	Amsterdam

Many-to-one

```
london_paintings <- combined_data %>%
  filter(city == "London")
```

acq_no	museum	artist	date	medium	tags	city
1908.54	Victoria & Albert	Jan Vermeer	c. 1650	oil paint on panel	domestic scene and women	London
1955.32	British Museum	Possibly Vermeer, Jan	c. 1655	oil on canvas	woman at window, portrait	London

Many-to-many

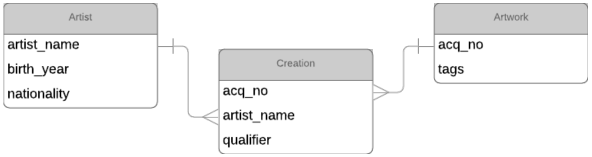
- Each object can have many artists
- Each artist can have many objects

We may well have biographical information about our artists

name	birth_year
Rembrandt	1606
Govaert Flinck	1615
Jan Vermeer	1632
Frans Hals	1582

Many-to-many

This requires an intermediate table where we get to encode the relationship, and also encode variables *about* that relationship.



Many-to-many

acq_no	name	qualification
1999.32	Rembrandt	studio of
1999.32	Govaert Flinck	
1908.54	Jan Vermeer	
1955.32	Jan Vermeer	possibly
1955.33	Frans Hals	

Note that painting `1999.32`, which has two artists, is repeated twice. And we can relate the qualifications (`studio of`, `possibly`) to specific artwork-artist pairs.

Many-to-many

This lets us filter paintings based on their artists' biographical info

```
temp_table <- artworks %>%
  left_join(creations) %>%
  left_join(artists)
```

acq_no	medium	support	cons_note	name	qualification	birth_year
1999.32	oil	canvas		Rembrandt	studio of	1606
1999.32	oil	canvas		Govaert Flinck		1615
1908.54	oil	panel		Jan Vermeer		1632
1955.32	oil	canvas		Jan Vermeer	possibly	1632
1955.33	oil	canvas	relined	Frans Hals		1582

Querying on related tables

```r
date_filtered <- temp_table %>%
  filter(birth_year <= 1615)
```

acq_no	medium	support	cons_note	name	qualification	birth_year
1999.32	oil	canvas		Rembrandt	studio of	1606
1999.32	oil	canvas		Govaert Flinck		1615
1955.33	oil	canvas	relined	Frans Hals		1582

Dates

There's no one true way to encode date information. It depends on your use-case.

1. "point" events? Or durations?
2. How precise are your sources?
3. How varied is that precision?

Point vs. duration events

date
1660-03-01
1661-05-20
1661-12-05

Point data specifies exactly one point in time; durations have a beginning and an end.

For point data, when in doubt, use `YYYY-MM-DD` format. It's an international standard, everything reads it.

beginning	end
1660-03-01	1660-03-03
1661-05-19	1665-05-25
1661-12-05	1661-12-05

For ranges, you'll want to have a start and end point specified using the same `YYYY-MM-DD` format.

start_by	end_by	note
1660-03-01	1660-04-01	Month precision
1661-01-01	1665-12-31	Year precision
1661-12-05	1661-12-05	Day precision
1661-12-05	1661-12-07	Day ranges

Depending on your context, it might be more accurate to think in terms of “no sooner than” and “no later than” rather than “beginning” and “end”. This can be useful when you have varying precision of dates:

But don't overdo it

untidy	tidy_start	tidy_end
19th century	1800	1899
17th-18th c.	1600	1799
1670s	1670	1679
mid-1800s	1830	1870

If your sources never have information down to the month or day, or you know that such precision isn't important, then just use a year or century marker. As long as you are *consistent*.

untidy	tidy_century
12th c.	11
10th century	9
200s	2

The precision that's useful to you will be totally context-dependent. Don't give yourself more work than you need to.

Applying to our original data

We need to make an executive decision about how we want to express “circa” or unbounded claims like “after”.

acq_no	orig_date	year_early	year_late
1999.32	after 1636	1636	1680
1908.54	c. 1650	1645	1655
1955.32	c. 1655	1650	1660
1955.33	16220	1622	1622

Here, I’ve expanded “circa” to mean around 5 years before or after. “after 1636” could have a number of different meanings depending on the context - maybe we can limit it based on the last year of the studio’s activity.

Gotchas

- If you're dealing with times, not just dates... then watch out for time zones. Python and R both have specialized libraries for these.
- When hand entering dates, make sure to validate the dates! You will inevitably enter `YYYY-02-31`, which doesn't exist.

Uncertainty

Uncertainty

There's a lot of uncertainty and missing information in historical sources.

What we can mostly handle are the known unknowns.

Uncertainty DON'Ts

- Adding [?] into records won't tell you much
  - Was the info totally missing from the document?
  - Was that info there, but illegible?
  - Did the document literally say [?]?
- Mixing uncertainty across different fields
  - i.e. having a check mark to say a record is "done" isn't very informative
  - Which part of the record is uncertain? The date? The artist?

Uncertainty DOs

- Make an uncertainty vocabulary if appropriate
  - missing
  - illegible
  - approximated
- Put boundaries on uncertainty
  - Dates aren't usually *totally* unknown - what are the realistic early/late dates given context?
- Use separate columns liberally, e.g. `date`, `date_uncertainty`

You can't document everything! If some tricky field is just not relevant enough to your research, then don't kill yourself trying to capture it with perfect specificity.

Be context-specific

acq_no	name	qualification
1999.32	Rembrandt	studio of
1999.32	Govaert Flinck	
1908.54	Jan Vermeer	
1955.32	Jan Vermeer	possibly
1955.33	Frans Hals	

- standardize your terms if you can
- only expend energy on it if it will meaningfully connect to your research question

The “notes” column

There will always be info that doesn’t fit into your schema. A “notes” column can be helpful here.

But as soon as you notice repeatedly putting a certain type of info in there, consider going back and making a dedicated column for that info.

Notes should usually have unique values. If they’re often the same value, that means some of them should be moved to their own column.

Only go as far as you need to

Only go as far as you need to

Data modeling can get infinitely complicated if you want to accommodate every possible use case. If you're not a museum or library, don't do that.

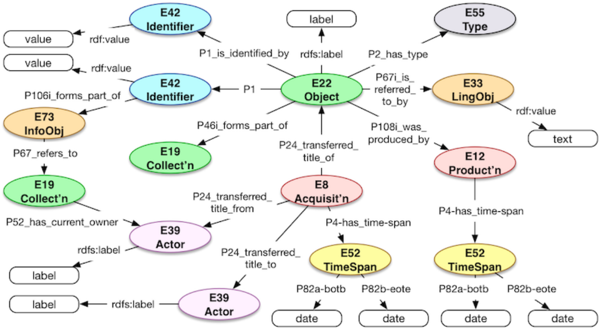


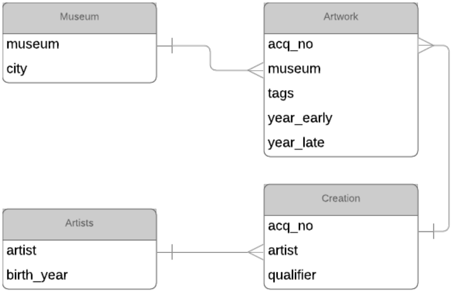
Figure 1: Photo archives data model from https://linked.art

Bringing it all back together

Bringing it all back together

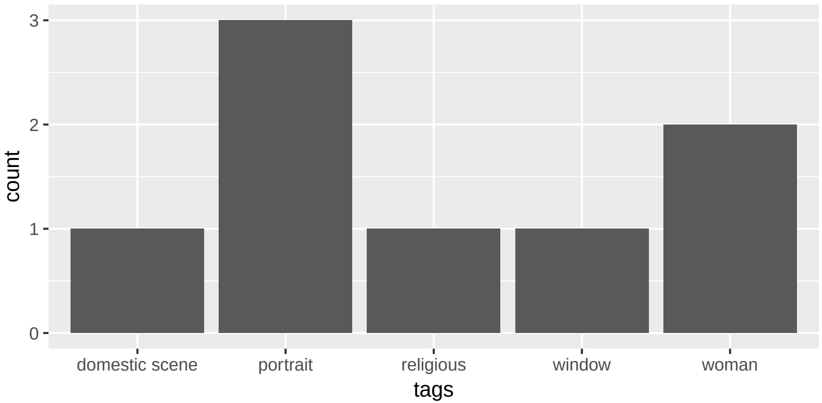
Most visualization and analysis software works with just one table.

By separating out tables first, we now have the flexibility to produce the one table we need for a given question.



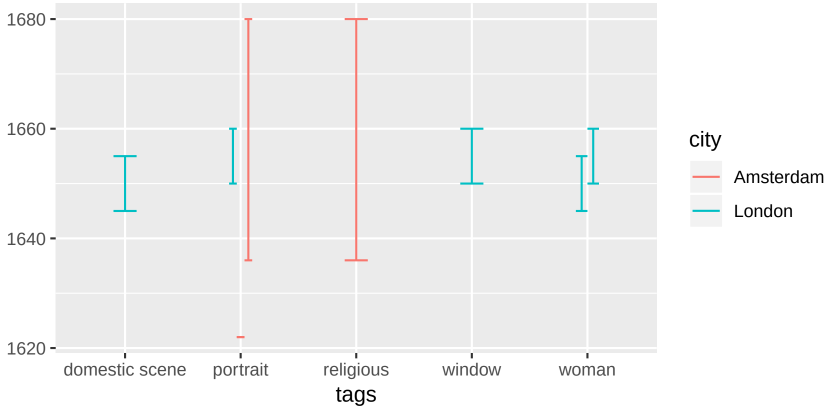
Count up the different tags used

```r
artworks %>%
  separate_rows(tags, sep=";") %>%
  ggplot(aes(x = tags)) +
  geom_bar()
```



Tags based on year of creation

```r
artworks %>%
  separate_rows(tags, sep=";") %>%
  left_join(museums, by = "museum") %>%
  ggplot(aes(ymin = year_early, ymax = year_late,
             x = tags, color = city, group = acq_no)) +
  geom_errorbar(position="dodge", width = 0.2)
```



Network of artists who worked together and when

Joining a table to itself can give us combinations of artists who worked together on the same artwork. This could be used to create a network data set to analyze.

```r
creations %>%
  left_join(creations, by = "acq_no") %>%
  filter(name.x != name.y) %>%
  left_join(artworks, by = "acq_no") %>%
  select(acq_no, name.x, name.y, year_early, year_late)
```

acq_no	name.x	name.y	year_early	year_late
1999.32	Rembrandt	Govaert Flinck	1636	1680
1999.32	Govaert Flinck	Rembrandt	1636	1680

Practical exercise

Practical exercise

Source: https://tinyurl.com/cmudh-2019-artcatalog

Group 1: https://tinyurl.com/cmudh-2019-tidy1

Group 2: https://tinyurl.com/cmudh-2019-tidy2

Working in groups, draft a data scheme for encoding this auction catalog. Aside from the obvious, think about:

- can we encode who's owned the artwork before?
- what are different ways to categorize the content of the descriptions?

Linking data

Linking data

Using shared vocabularies between data sets

- https://vocab.getty.edu
- https://programminghistorian.org/en/lessons/intro-to-linked-data

Documenting tidy data

Do it for future-you & for others

- You **will** forget what you did in a few months. Or even a few days. Docs will remind you.
- Docs make writing reports/articles easier
- Docs make your data reusable:
  - others won't have to guess at what a certain column means
  - or what decisions you made when recording it
  - or how to cite it
  - or if/how they may reuse it

Show your work

- Describe what you made:
  - Keep a plain text doc in the same directory as your tables
  - Have a heading for each table
  - List every column name and describe what it means
    - Incl. list of possible values, relation to other tables as appropriate
- Document the process
  - Did you adapt this from another data set? (incl. original data, or link)
  - Describe the transformations you made, including what software you used

Documentation format

A plain `.txt` file with a column name / definition list:

```
Table 1
-------
col 1 - definition
col 2 - joining id with table 2 - col 3
col 3 - definition


Table 2
-------
col 1 - definition
col 2 - definition
col 3 - joining id with table 1 - col 2
```

Plain text

- Use plain text file types for tables and docs (`.txt`, `.csv`, not `.xslx`)
  - Free
  - Somewhat more future-proof
  - Track-able
- Creating in Excel/Google Sheets is fine, you can export it
  - When saving in Excel, use `UTF-8` so that accents & special characters are preserved
  - Don't rely on meaningful formatting (colored cells, bold, italics, borders), because that won't be preserved
  - Save multiple versions

Archive it

- Bundle data and documentation in the same directory and zip them.
- Distribute
  - Institutional repository (upload it with your dissertation)
  - Journal websites
  - Zenodo
  - Git (works great with all-text files - more and more libraries and journals will be moving towards this method for tracking file versions)

Resources

Resources

- Building out tidy data using Google Sheets
- AirTable is a decent, Google-sheets-like option for building out multi-table relational databases.
- See a tidy data example
- Database management
    - UCLA DH101: Data and Databases
    - Designing Databases for Historical Research (great intro to relational DBs)
- Data cleaning with OpenRefine