

Data Processing, Code Documentation and Beyond (Emacs and org-mode)

JONATHAN A. HARTMAN | LUKAS C. BOSSERT

September 19, 2023

Contents

1	Overview	1
2	Introduce	2
3	Prepare	2
3.1	Data retrieval using SPARQL	2
3.2	Data cleaning using shell	4
4	Process	5
4.1	Data Aggregation with Python	5
4.2	Counting Elements with awk	6
4.3	Network Disply with R	6
5	Preserve	6
5.1	Manual export	9
5.2	Automatic batch process	9

1 Overview

This document provides insights into an efficient way handling data. We show not only how to retrieve data from an publicly accesible webpge but also how the data can be processed afterwards. We admit that in the examples shown below we definetly drawing from the full, but we consider this as a proof of concept for how in our modern technological world plain text is still a great way of processing and documenting data workflow and analyses.

The paper is divided into three main steps, focussing on first preparing, second processing and last presevering the data and its documentation (fig. 1).

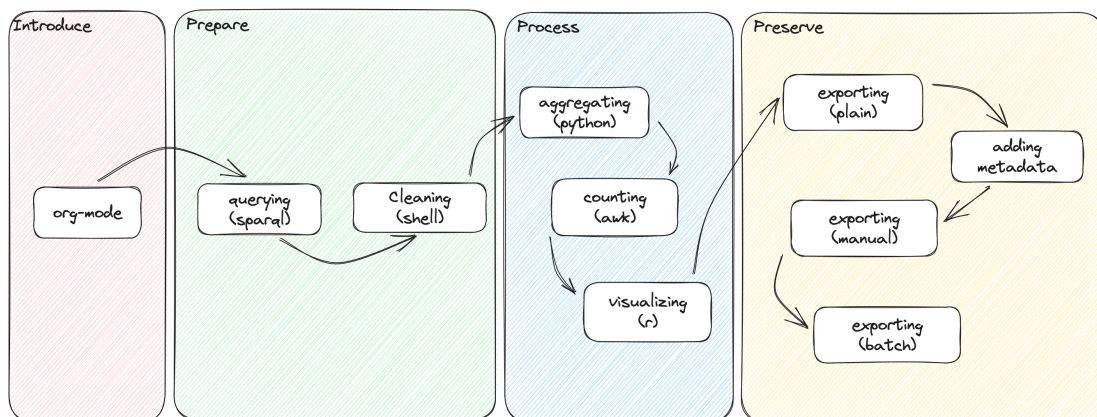


Figure 1: Workflow of the document. Source Excalidraw.

2 Introduce

What is Emacs and **org-mode**? Well, where to start? You may not have heard of Emacs or org-mode, yet. Usually it is considered to a tool for geeks, this might be kind of true, but once you noticed the myriad ways of using Emacs¹ and espeically its module org-mode you never ever won't to use anything else.² Emacs has been around for decades (no kidding) and is free software.

Org-mode is quite younger but the killing feature in Emacs. Or let's express it with the words of the original creator Carsten Dominik:

Org-mode does outlining, note-taking, hyperlinks, spreadsheets, TODO lists, project planning, GTD, HTML and L^AT_EX authoring, all with plain text files in Emacs.

or in a nutshell:

Back to the future for plain text
(Carsten Dominik)

Let's make an executive summary of org-mode:

- Module for Emacs
- Plain text based
- Around since 2003
- Meant for (scientific) text production and organisation
 - project management
 - agenda, diary, journaling
 - personal knowledge management
 - presentation
 - single-source-publishing
 - literate programming
- Extensible and fully customizable

Org-mode is a magnificent tool when it comes to reproducible research,³ since this combines a well documented way of analysing a data set.

3 Prepare

For our demonstration, we are going to create a dataset from openly available data on the German National Research Data Infrastructure (NFDI) and perform some simple analysis tasks on it.

3.1 Data retrieval using SPARQL

The data we are interested in exists on Wikidata. Wikidata is similar to Wikipedia, but rather than long form articles, the data is stored as structured data. This allows machines to easily access and traverse these pages with query langauges. Here, we are going to submit a SPARQL query to the Wikidata query endpoint.

¹Harley Hahn. *Harley Hahn's Emacs Field Guide*. Apress, 2016. doi: 10.1007/978-1-4842-1703-0; John R. Kitchin, Ana E. Van Gulick, and Lisa D. Zilinski. "Automating data sharing through authoring tools". In: *International Journal on Digital Libraries* 18.2 (June 2016), pp. 93–98. doi: 10.1007/s00799-016-0173-7; Stefan Strobel and Thomas Uhl. "GNU Emacs". In: *Linux Unleashing the Workstation in Your PC*. Springer US, 1996, pp. 287–324. doi: 10.1007/978-1-4684-0247-6_13.

²There might be people having a different opinion.

³Luka Stanisic and Arnaud Legrand. "Effective Reproducible Research with Org-Mode and Git". In: *Euro-Par 2014: Parallel Processing Workshops*. Ed. by Luís Lopes et al. Cham: Springer International Publishing, 2014, pp. 475–486. ISBN: 978-3-319-14325-5.

SPARQL will look familiar to anyone familiar with SQL, however it is slightly more cryptic at first glance. Take a look at the below query – things like “Q98270496” refer to specific items in wikidata, where things like “P31” are more akin to concepts. In English, this query translates to something like

Give me the Names for items that has a property (P31) of NFDI Consortia (Q98270496), and return all items you find on each of those entries under the property “affiliations” (P1416).

If you like how to do this in more detail, have a look at.⁴

```

1 SELECT ?wLabel ?pLabel
2 WHERE
3 {
4   ?p wdt:P31 wd:Q98270496 .
5   ?p wdt:P1416 ?w .
6   SERVICE wikibase:label { bd:serviceParam wikibase:language "en" . }
7 }
8 ORDER BY ASC(?wLabel) ASC(?pLabel)
9 LIMIT 50

```

Listing 1: Retrieving the dataset from wikidata

Table 1: Result of the query for NFDI consortia and their institutions.

wLabel	pLabel
Q105775472	NFDI4Health
Q1117007	NFDI4Health
Q115254989	NFDI4Objects
Q1205424	NFDI4Objects
Q17575706	NFDI4Objects
Academy of Sciences and Humanities in Hamburg	Text+
Academy of Sciences and Literature Mainz	NFDI4Culture
Academy of Sciences and Literature Mainz	NFDI4Memory
Academy of Sciences and Literature Mainz	NFDI4Objects
Academy of Sciences and Literature Mainz	Text+
Alfred Wegener Institute for Polar and Marine Research	NFDI4Biodiversity
Alfred Wegener Institute for Polar and Marine Research	NFDI4DataScience
Alfred Wegener Institute for Polar and Marine Research	NFDI4Earth
Anthropological Society (Munich)	NFDI4Objects
Arachnologische Gesellschaft	NFDI4Biodiversity
Arbeitskreis Provenienzforschung e.V.	NFDI4Memory
Archivschule Marburg	NFDI4Memory
Archäologische Kommission für Niedersachsen	NFDI4Objects
Archäologisches Museum Hamburg und Stadtmuseum Harburg	NFDI4Objects
Arthistoricum	NFDI4Culture
Association for Data-Intensive Radio Astronomy	PUNCH4NFDI
Association for Technology and Construction in Agriculture	FAIRAgro
Association of German Architects	NFDI4Culture
Association of Population Based Cancer Registries in Germany	NFDI4Health
Association of states archaeologists	NFDI4Objects
BERD@NFDI	Base4NFDI
Bach-Archiv Leipzig	NFDI4Culture
Bauhaus-Universität Weimar	NFDI4Ing
Bavarian Academy of Sciences and Humanities	BERD@NFDI

Continued on next page

⁴Lukas C. Bossert et al. “Das muss noch in Wikidata rein”. In: *Bausteine FDM* (Sept. 11, 2023), pp. 2–18. doi: 10.17192/bfdm.2023.5.8580.

Continued from previous page

wLabel	pLabel
Bavarian Academy of Sciences and Humanities	NFDI4Earth
Bavarian Academy of Sciences and Humanities	NFDI4Memory
Bavarian Academy of Sciences and Humanities	NFDI4Objects
Bavarian Academy of Sciences and Humanities	NFDI4CS
Bavarian Academy of Sciences and Humanities	PUNCH4NFDI
Bavarian Academy of Sciences and Humanities	Text+
Bavarian Forest National Park	NFDI4Biodiversity
Bavarian Natural History Collections	NFDI4Biodiversity
Bavarian Natural History Collections	NFDI4Objects
Bavarian State Archaeological Collection	NFDI4Objects
Bavarian State Archives	FAIRAgro
Bavarian State Archives	NFDI4Biodiversity
Bavarian State Archives	NFDI4Earth
Bavarian State Archives	NFDI4Objects
Bavarian State Library	NFDI4Culture
Bavarian State Library	NFDI4Memory
Bavarian State Research Center for Agriculture	FAIRAgro
Beethoven House	NFDI4Culture
Beilstein Institute for the Advancement of Chemical Sciences	NFDI4Chem
Berlin State Library	Base4NFDI
Berlin State Library	NFDI4Memory

3.2 Data cleaning using shell

The data we got from listing 1 is good but it needs further cleaning.

We can see several entries in our data that look like "Q1234567" - These are Q Ids for items which no label has been defined. Let's remove those from our dataset.

We're going to include the output from the previous cell, where we executed the SPARQL query, as an input variable to this cell (:var input=raw-dataset).

```
echo "$input" | sed -E '/Q[0-9]+/d'
```

Listing 2: Cleaning the raw data using good old sed and a regex pattern.

Table 2: Cleaned data set which will be used for further processing.

wLabel	pLabel
Academy of Sciences and Humanities in Hamburg	Text+
Academy of Sciences and Literature Mainz	NFDI4Culture
Academy of Sciences and Literature Mainz	NFDI4Memory
Academy of Sciences and Literature Mainz	NFDI4Objects
Academy of Sciences and Literature Mainz	Text+
Alfred Wegener Institute for Polar and Marine Research	NFDI4Biodiversity
Alfred Wegener Institute for Polar and Marine Research	NFDI4DataScience
Alfred Wegener Institute for Polar and Marine Research	NFDI4Earth
Anthropological Society (Munich)	NFDI4Objects
Arachnologische Gesellschaft	NFDI4Biodiversity
Arbeitskreis Provenienzforschung e.V.	NFDI4Memory
Archivschule Marburg	NFDI4Memory
Archäologische Kommission für Niedersachsen	NFDI4Objects
Archäologisches Museum Hamburg und Stadtmuseum Harburg	NFDI4Objects
Arthistoricum	NFDI4Culture
Association for Data-Intensive Radio Astronomy	PUNCH4NFDI
Association for Technology and Construction in Agriculture	FAIRAgro
Association of German Architects	NFDI4Culture

Continued on next page

Continued from previous page

wLabel	pLabel
Association of Population Based Cancer Registries in Germany	NFDI4Health
Association of states archaeologists	NFDI4Objects
BERD@NFDI	Base4NFDI
Bach-Archiv Leipzig	NFDI4Culture
Bauhaus-Universität Weimar	NFDI4Ing
Bavarian Academy of Sciences and Humanities	BERD@NFDI
Bavarian Academy of Sciences and Humanities	NFDI4Earth
Bavarian Academy of Sciences and Humanities	NFDI4Memory
Bavarian Academy of Sciences and Humanities	NFDI4Objects
Bavarian Academy of Sciences and Humanities	NFDIxCs
Bavarian Academy of Sciences and Humanities	PUNCH4NFDI
Bavarian Academy of Sciences and Humanities	Text+
Bavarian Forest National Park	NFDI4Biodiversity
Bavarian Natural History Collections	NFDI4Biodiversity
Bavarian Natural History Collections	NFDI4Objects
Bavarian State Archaeological Collection	NFDI4Objects
Bavarian State Archives	FAIRAgro
Bavarian State Archives	NFDI4Biodiversity
Bavarian State Archives	NFDI4Earth
Bavarian State Archives	NFDI4Objects
Bavarian State Library	NFDI4Culture
Bavarian State Library	NFDI4Memory
Bavarian State Research Center for Agriculture	FAIRAgro
Beethoven House	NFDI4Culture
Beilstein Institute for the Advancement of Chemical Sciences	NFDI4Chem
Berlin State Library	Base4NFDI
Berlin State Library	NFDI4Memory

4 Process

4.1 Data Aggregation with Python

The great thing about org mode is that we can seamlessly switch between languages! Our original query (listing 1) was written in SPARQL, which returned a kind of table (tab. 1). We then took that table and ran a shell command on it. Now, we're going to take the output of that shell command (cf. tab. 2) and run some python code on it.

```
python -m pip install pandas --user
```

Table 3: Overview of institutions and the count of their associated consortia.

wLabel	Count
Bavarian Academy of Sciences and Humanities	7
Bavarian State Archives	4
Academy of Sciences and Literature Mainz	4
Alfred Wegener Institute for Polar and Marine Research	3
Berlin State Library	2
Bavarian State Library	2
Bavarian Natural History Collections	2
BERD@NFDI	1
Beilstein Institute for the Advancement of Chemical Sciences	1
Beethoven House	1
Bavarian State Research Center for Agriculture	1
Bavarian State Archaeological Collection	1
Bavarian Forest National Park	1
Bauhaus-Universität Weimar	1

Continued on next page

Continued from previous page

wLabel	Count
Bach-Archiv Leipzig	1
Academy of Sciences and Humanities in Hamburg	1
Association of Population Based Cancer Registries in Germany	1
Association of German Architects	1
Association for Technology and Construction in Agriculture	1
Association for Data-Intensive Radio Astronomy	1
Arthistoricum	1
Archäologisches Museum Hamburg und Stadtmuseum Harburg	1
Archäologische Kommission für Niedersachsen	1
Archivschule Marburg	1
Arbeitskreis Provenienzforschung e.V.	1
Arachnologische Gesellschaft	1
Anthropological Society (Munich)	1
Association of states archaeologists	1

There is also a “native way” getting the counting done by using the package `org-aggregate`⁵.

4.2 Counting Elements with awk

We’re not limited to python though. Here we’re going to perform a very similar aggregation, but grouping by consortia to get the number of institutes at each. Like the listing 3 above, we are going to use the output of listing 2 (cf. tab. 2) to perform this operation. Instead of python, we’re going to use `awk` for our data processing.

As an additional bonus, we’re going to parameterize this cell by defining a variable called `consortium`. With this we could reuse the code in this cell over and over, changing the desired consortium name to show only the desired results.

Having created the source block we can also use it in our text with executing the the function `call_institutions-count('NFDI40bjects')`. The result will be blended in smoothly in the text and if there are any changes to the initial data set updated automatically.

Back to our example: So, now we know of many institutions are involved in `NFDI40bjects` (9 institutions) or in `NFDI4Earth` (3 institutions).

4.3 Network Disply with R

How about something a little more visual than some tables? We can also create plots and visuals, generating them with the code contained in the document and embedding the results in the output.

And while we’re at it, how about another language? This time we’ll use R to make a simple network plot of our data. Again, we’re still using the output from listing 2 (which is tab. 2) to do this.

The result is a nice visualization of a network (fig. 2). Such a visualization can help to detect outliers faster.

5 Preserve

There are two ways exporting this document in multiple documents. The concept of this is called “single-source-publishing”. This means we have on document, our `org-file`, and we will export it into different formats, which are more suitable for different occasions.

⁵<https://github.com/tbanel/orgaggregate>

```

1 import pandas as pd
2
3 # The data comes into the cell as a list of lists.
4 # We can pick it apart into a DataFrame object
5 df = pd.DataFrame(clean_df[1:], columns=clean_df[0])
6
7 # Perform a groupby operation on wLabel and
8 # rename the resulting new column "Count"
9 institutions_by_consortia = (
10     df
11     .groupby("wLabel")
12     .size()
13     .sort_values(ascending=False)
14     .reset_index(name="Count"))
15
16 # Return our dataframe in a way that org will
17 # display it as an org table
18 return [list(institutions_by_consortia.columns),
19         None, *map(list, institutions_by_consortia.values)]

```

Listing 3: Counting the number of consortia involved in one institution.

```

1 BEGIN {
2 # before the evaluating process of the data begins
3 # this block is taken in account
4 # set the separator to tab
5 FS = "\t"
6 }
7 # MAIN section of the evaluating process
8 #-----
9 # while going through the rows of the input
10 # check only for the second column
11 # step a counter for equal values and store it in 'counts'
12 $2 == consortium { ++counts[$2] }
13 END {
14 # final part where no evaluation is done anymore
15 # only collecting and printing results
16 # going through the counts from above
17 for (k in counts)
18 # check for the amount of associated institutions
19     if (counts[k] == 1)                                     (singular)
20 # if only one institution, then use the singular version
21     print consortium " (" counts[k] " institution)";
22 # otherwise we need the plural form.
23     else print consortium " (" counts[k] " institutions)"
24 }

```

Listing 4: Calculating the number of involved institions in one specific consortium.

```

1  # making sure the required package is installed
2  if (!require("igraph")) install.packages("igraph")
3  library("igraph")
4  # making a more robust outcome by stating a seed number
5  set.seed(123456789)
6  # convert the tabular data into a data frame which is required
7  # for creating a network
8  NFDI_network <- graph_from_data_frame(NFDI_edges,
9                                     directed = FALSE)
10 plot(NFDI_network,                    # loading data frame
11      main = "NFDI Network",           # adding a title
12      # adding a color to all nodes from the second column.
13      vertex.color = c("blue", "red"),
14      [1 + names(V(NFDI_network)) %in% NFDI_edges[,2]],
15      vertex.size = 4,                  # size of the node
16      vertex.frame.color = NA,          # no frame for nodes
17      vertex.label = NA,                # no color of the description
18      edge.curved = 0.2,                # factor of "curvity"
19      )

```

Listing 5: Network of all institutions and their related consortia.

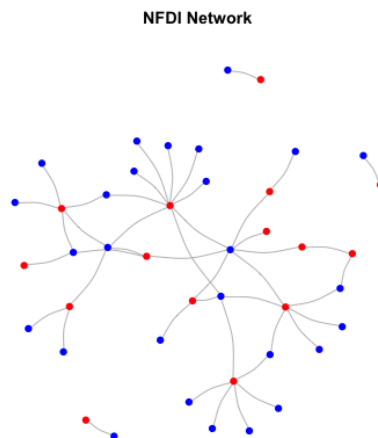


Figure 2: Network of NFDI consortia (red) and institutions (blue).

5.1 Manual export

The common approach is to invoke the commands for exporting into a certain format individually and by hand. Org-mode has a great build in exporting mechanism which converts the document into all mainly used formats. You get to the menu by calling `SPC m e` or `C-c C-e` and then select which export format you would like to have.

In tab. 4 you find a quick overview of some basic formats.

Table 4: Overview of various individual export functions.

	evil	normal
PDF	<code>SPC m e l o</code>	<code>C-c C-e l o</code>
HTML	<code>SPC m e h o</code>	<code>C-c C-e h o</code>
ASCII	<code>SPC m e t a</code>	<code>C-c C-e t a</code>

5.2 Automatic batch process

In a batch process the file is opened with a clean and neutral version of emacs and will be exported (see listing 6).

```
1 (let ((org-file (find-file-noselect filename)))
2   (with-current-buffer org-file
3     (org-html-export-to-html)
4     (message "HTML export successful.")
5   )
6   (with-current-buffer org-file
7     (org-ascii-export-to-ascii)
8     (message "ASCII export successful.")
9   )
10  (with-current-buffer org-file
11    (org-latex-export-to-pdf)
12    (message "PDF export successful.")
13  ))
```

Listing 6: Exporting file into various formats

References

- Bossert, Lukas C. et al. "Das muss noch in Wikidata rein". In: *Bausteine FDM* (Sept. 11, 2023), pp. 2–18. DOI: 10.17192/bfdm.2023.5.8580.
- Hahn, Harley. *Harley Hahn's Emacs Field Guide*. Apress, 2016. DOI: 10.1007/978-1-4842-1703-0.
- Kitchin, John R., Ana E. Van Gulick, and Lisa D. Zilinski. "Automating data sharing through authoring tools". In: *International Journal on Digital Libraries* 18.2 (June 2016), pp. 93–98. DOI: 10.1007/s00799-016-0173-7.
- Stanisic, Luka and Arnaud Legrand. "Effective Reproducible Research with Org-Mode and Git". In: *Euro-Par 2014: Parallel Processing Workshops*. Ed. by Luís Lopes et al. Cham: Springer International Publishing, 2014, pp. 475–486. ISBN: 978-3-319-14325-5.
- Strobel, Stefan and Thomas Uhl. "GNU Emacs". In: *Linux Unleashing the Workstation in Your PC*. Springer US, 1996, pp. 287–324. DOI: 10.1007/978-1-4684-0247-6_13.