Preserving Survey Data

Lars Vilhuber Laurel Krovetz

2026-06-14

Preserving Survey Data

Lars Vilhuber Laurel Krovetz

2026-06-14

labordynamicsinstitute.github.io/tutorial-preserving-survey/presentation/

The problem of credibility

How can we know that a data source is reliably obtained?

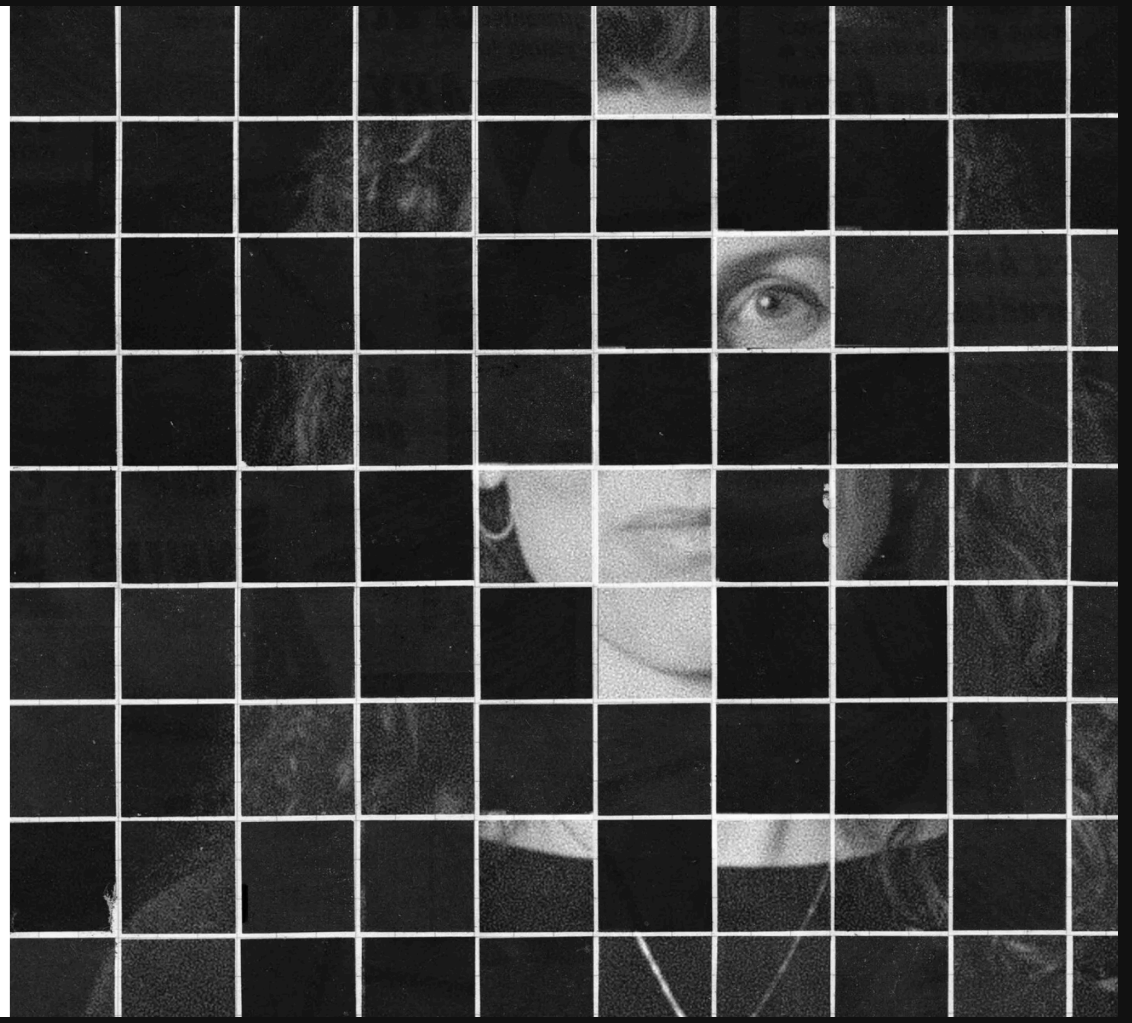
Consider the case of Gino

The New York Times

Account

The Harvard Professor and the Bloggers

When Francesca Gino, a rising academic star, was accused of falsifying data — about how to stop dishonesty — it didn't just torch her career. It inflamed a crisis in behavioral science.



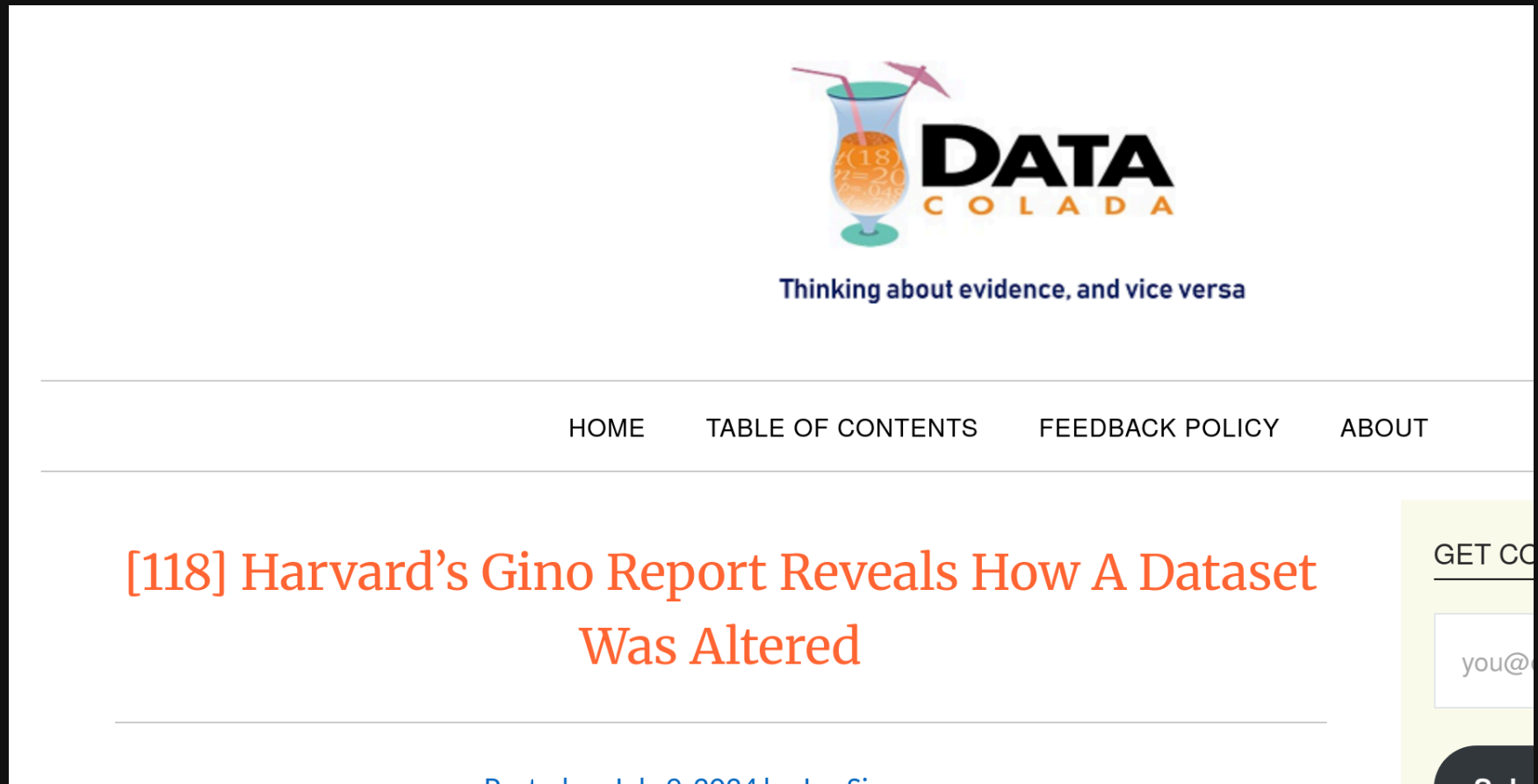
Francesca Gino

The case of Gino

- Francesca Gino was a tenured professor at Harvard Business School, writing on honesty (!)

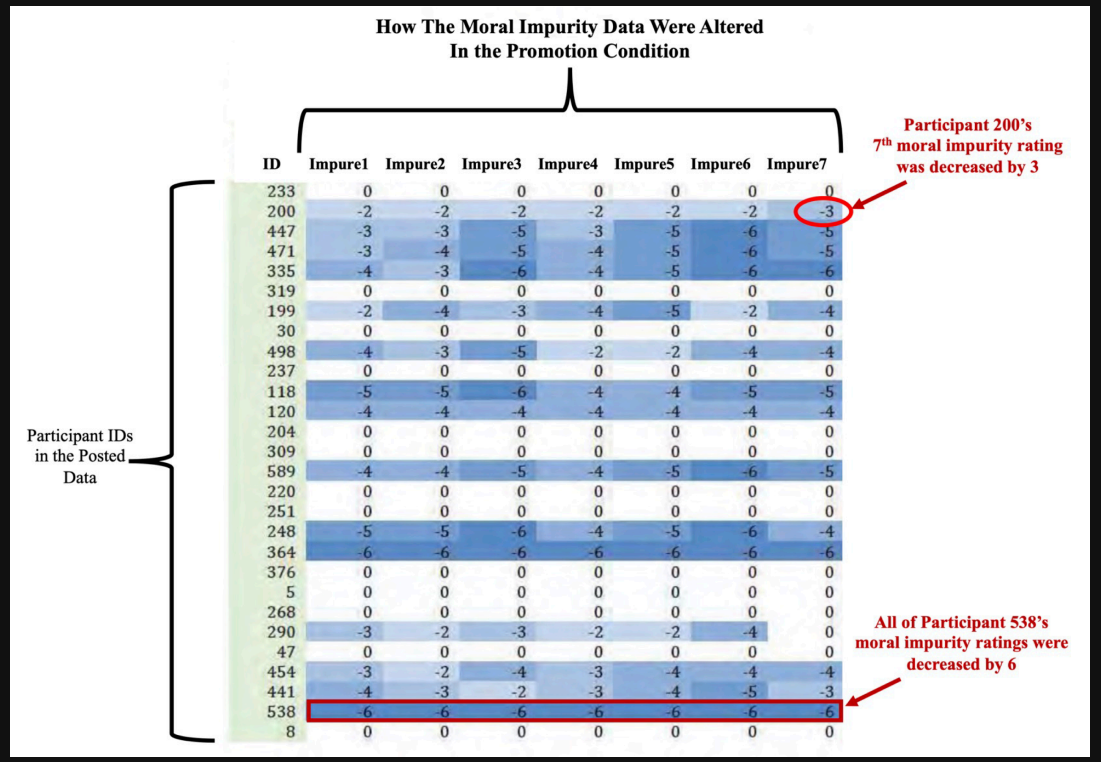
The case of Gino

- Several articles were investigated by third parties (Data Colada, in particular [1]), and found to be problematic

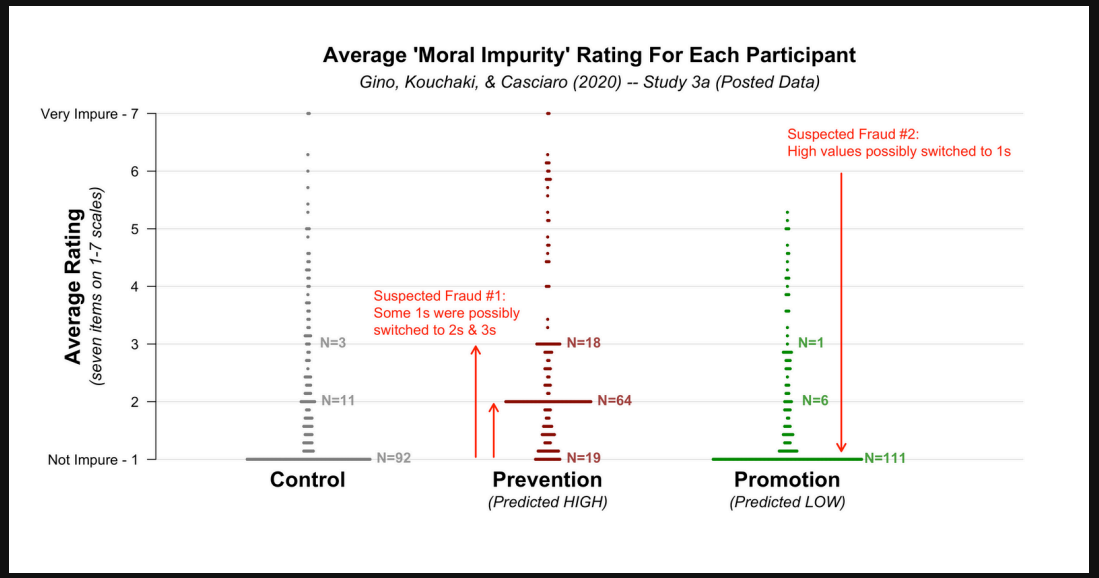


The case of Gino

- At least one of them had manipulated data **AFTER** it had been collected, **BEFORE** it had been analyzed.



ID	Impure1	Impure2	Impure3	Impure4	Impure5	Impure6	Impure7
233	0	0	0	0	0	0	0
200	-2	-2	-2	-2	-2	-2	-3
447	-3	-3	-5	-3	-5	-6	-5
471	-3	-4	-5	-4	-5	-6	-5
335	-4	-3	-6	-4	-5	-6	-6
319	0	0	0	0	0	0	0
199	-2	-4	-3	-4	-5	-2	-4
30	0	0	0	0	0	0	0
498	-4	-3	-5	-2	-2	-4	-4
237	0	0	0	0	0	0	0
118	-5	-5	-6	-4	-4	-5	-5
120	-4	-4	-4	-4	-4	-4	-4
204	0	0	0	0	0	0	0
309	0	0	0	0	0	0	0
589	-4	-4	-5	-4	-5	-6	-5
220	0	0	0	0	0	0	0
251	0	0	0	0	0	0	0
248	-5	-5	-6	-4	-5	-6	-4
364	-6	-6	-6	-6	-6	-6	-6
376	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0
268	0	0	0	0	0	0	0
290	-3	-2	-3	-2	-2	-4	0
47	0	0	0	0	0	0	0
454	-3	-2	-4	-3	-4	-4	-4
441	-4	-3	-2	-3	-4	-5	-3
538	-6	-6	-6	-6	-6	-6	-6
8	0	0	0	0	0	0	0

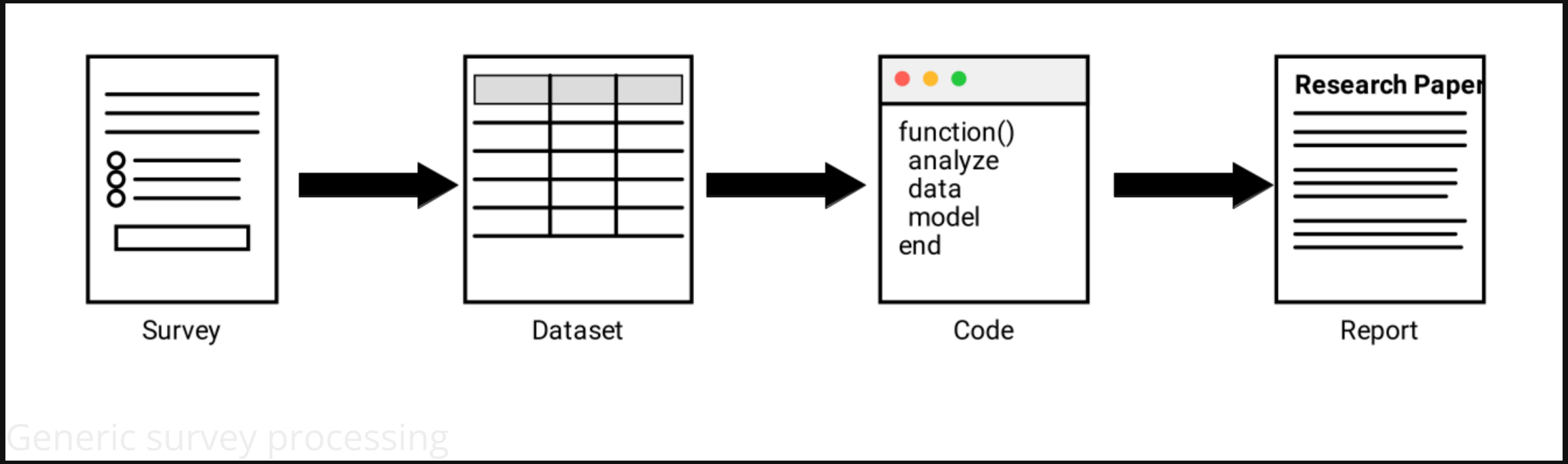


Results of manipulation

Data manipulation

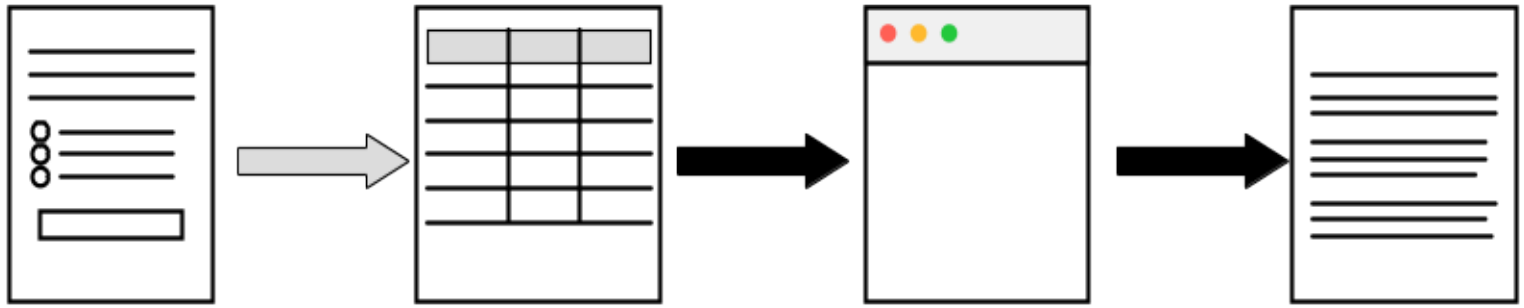
A generic survey workflow

Generic survey processing



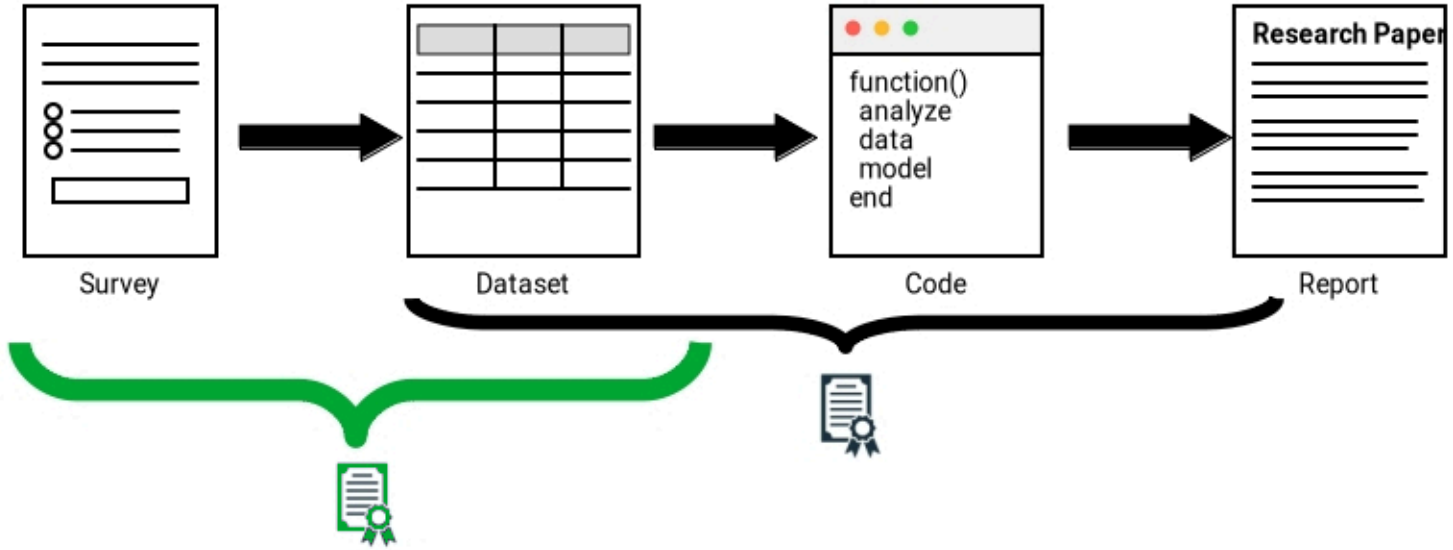
Generic survey processing

Requiring transparency in academia



Generic survey processing

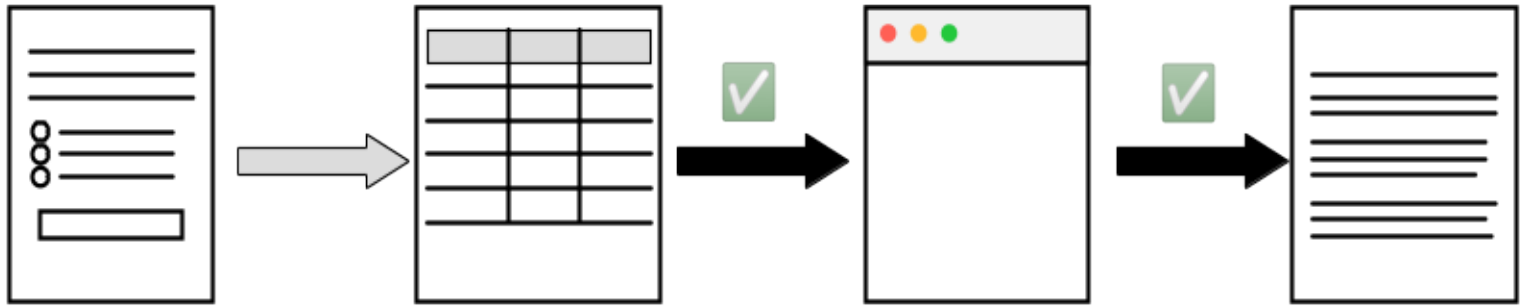
Where we are headed



Certified survey processing

Modern verification processes

Verifying transparency in academia



Generic survey processing

Verification by journals

- **Provision** (publication of materials) provides transparency
- **Verification** (running the analysis again - computational reproducibility) compensates for ***mistrust***/***absence of trust***

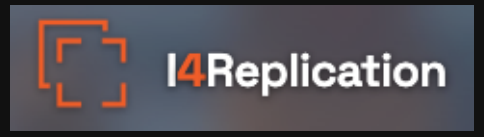
Which journals

- American Economic Association (8)
- Econometric Society (3)
- Canadian Journal of Economics (1)
- Royal Economic Society (2)
- Western Economic Association International (1)
- European Economic Association (1)
- Review of Economic Studies (1)
- Journal of the European Economic Association (1)
- Journal of Political Economy (3)
- American Journal of Political Science (1)
- American Political Science Review (1)

Verification by others

- Pre-publication: cascad

- Post-publication: Data Colada, Institute for Replication

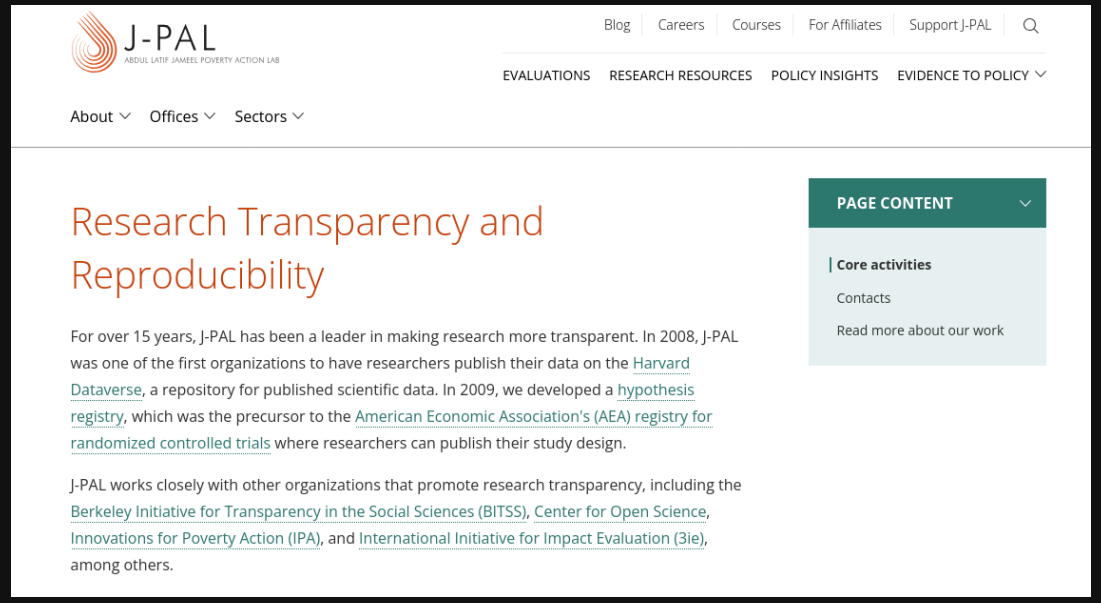


Verification by institutions

- World Bank

THE WORLD BANK
IBRD • IDA
Reproducible Research Repository

World Bank RRR[2]



J-PAL again?

Outline of the tutorial

Basic

- how to process,
- de-identify, and
- analyze survey data
- publish data

Expansion

- how to download automatically
- how to preserve automatically
- how to do so in a credible and transparent fashion.

In a nutshell

We'll use an API to retrieve the data, show you how to clean and strip the data of confidential information and non-consenting responses, and use another API to preserve the data.

Goals

- [ ] **Create a survey** (in Qualtrics) for data collection.
- [ ] Load the latest responses from the server (**using an API**)
- [ ] Clean and process the data to **remove non-public data automatically**.
- [ ] **Preserve** shareable data in a trusted repository
- [ ] Later, **publish those data** with a credible record of when it was first preserved!

Some notes on Qualtrics

- [ ] **Create a survey** (in Qualtrics) for data collection.
- [ ] Load the latest responses from the server **using an API**

There is not much in this tutorial that requires Qualtrics.

- You could do this with SurveyCTO, LimeSurvey, or any other system that has an API.
- You could do this with Google Forms, if you have linked it to a Google Sheet.
- You could do this with a **lab experiment** system that stores data in an SQL database

Some notes on removal of PII

- Clean and process the data to **remove non-public data automatically**.

It is important to remove any PII or confidential information as soon as possible.

Some notes on removal of PII

- That **may not always be feasible**. For instance, if you need geolocation to merge in contextual data, or compute distances, then some data processing may unavoidably require access to sensitive data.
- But any data that is ***not needed should be removed*** early on.
- This is **not irreversible**: if you later find that you need more data elements, you can always re-process the raw data, stored on Qualtrics, until your IRB requires that you delete those data.

Some notes on Preservation vs. Sharing

- [ ] **Preserve** shareable data in a trusted repository
- [ ] Later, **publish those data** with a credible record of when it was first preserved!

It is important to distinguish

- **preserving data** from
- **publishing** data, and possibly
- **sharing data with collaborators**

Preservation

- Preservation != publication, != sharing
- In fact, preservation may mean: not very accessible at all!
- Preservation is intended to maintain data for tens, even hundreds of years
  - Preservation may involve curation: active transformation of the data for improved accessibility

Sharing data

- Shared on a personal website
- Sharing a Dropbox link
- Posting it on OSF as a project

All useful for sharing, but do not preserve the data

Test

- Who has a Github account?

Test

- Who has a Github account?
- How long does it take you to delete your entire Github repository, forever?

Demonstrating the core steps in R

The core steps

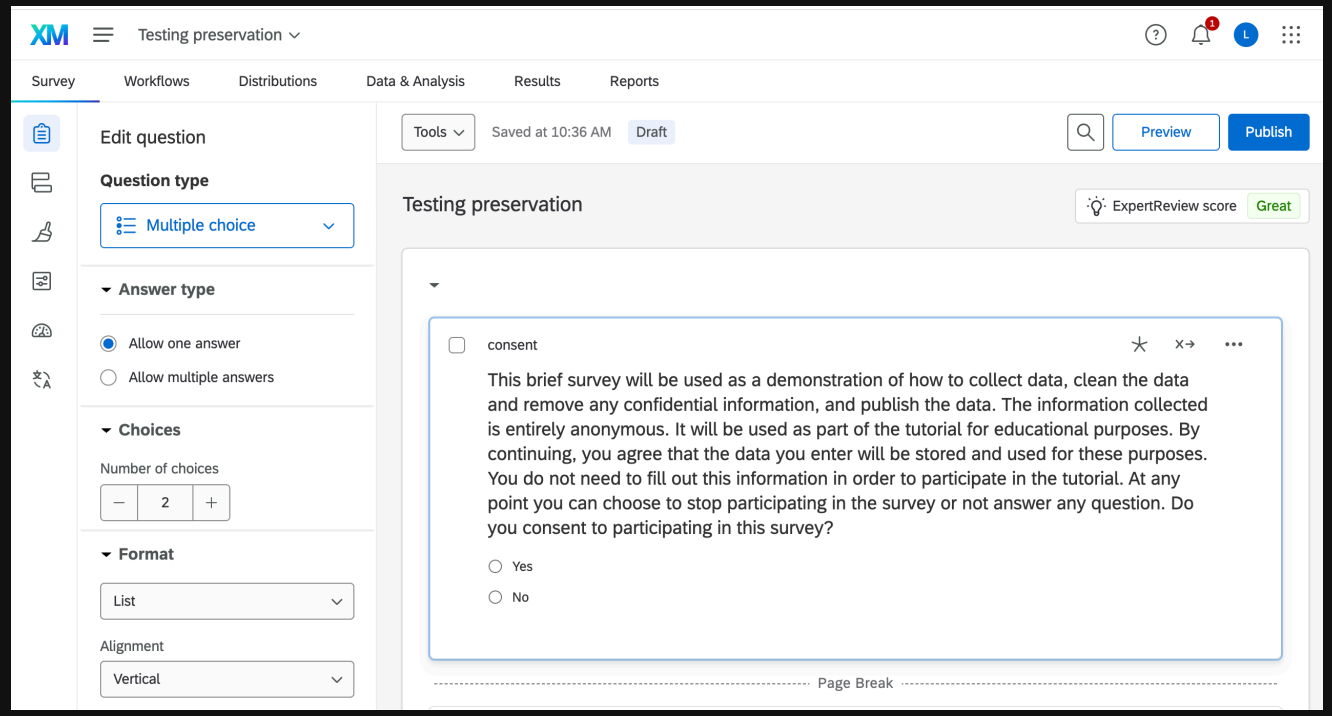
We will first walk through the core steps you can do **by hand**, in R:

1. **Collect** — create the survey and gather responses (in Qualtrics)
2. **Download** — export the responses from the web interface
3. **Analyze** — load, clean, and process the data in R
4. **Publish** — save the cleaned data, ready to share

This is not a full Qualtrics tutorial!

Creating a survey in Qualtrics

You'll typically have access to a Qualtrics account through your university or organization. Then it is easy to construct a survey using the web tool.



Side-note: Survey definition from Qualtrics

You should not forget to preserve your survey definition!

- Download a `qsf` file to save and transfer survey structure to have a backup survey template. Export as `.qsf` in `Tools` in Qualtrics.
- Can also export survey as Word doc in `Tools` in Qualtrics. Choose this option to get a well-formatted document.

Side-note to side-note: Confidential data in survey definitions!

It is possible that your survey definition itself contains information that you are not allowed to publish:

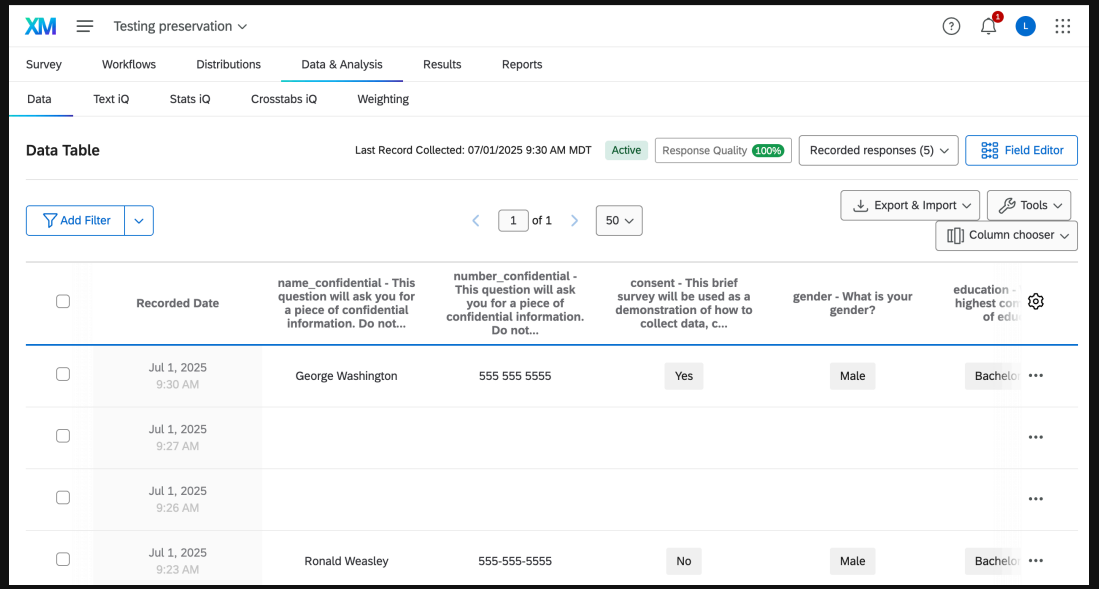
- You might be running the survey with a firm, and the firm does not want to be identified
- You are asking questions about specific products, and the product names are confidential

> It is actually **hard** to de-identify a `qsf` file. We will not try to do this here, but you should be aware of this issue.

Take our survey

Survey responses in Qualtrics

Responses can be easily checked at a glance in the `Data and Analytics` tab. 🔒

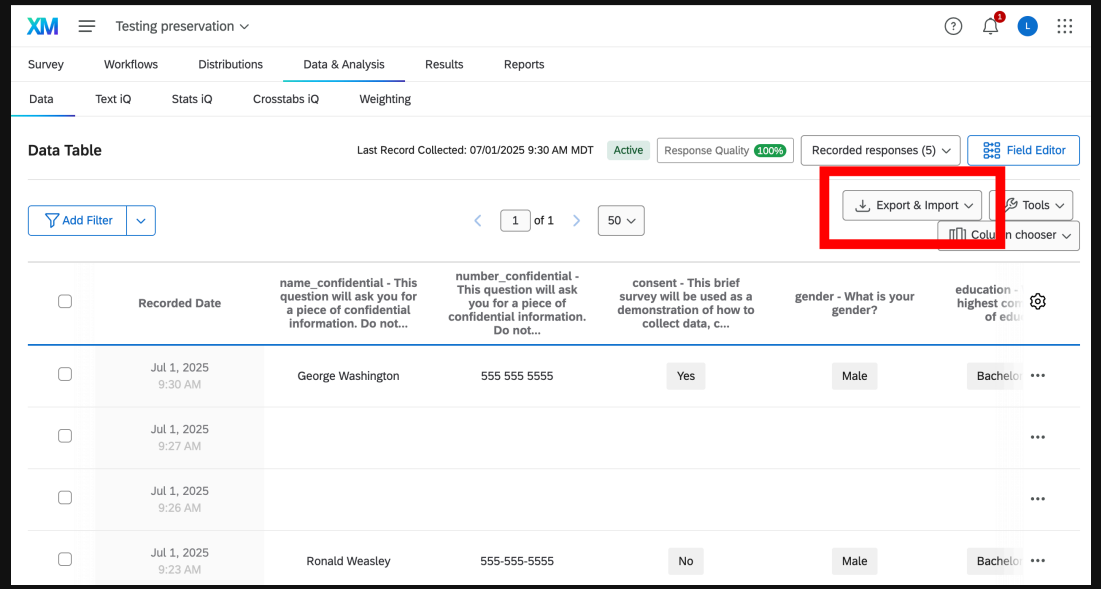


Qualtrics data interface

Downloading data

You can download data directly from this page

- If you do this only once, downloading manually is fine.
- Do it 2-3 times, you ***may*** want to program it!

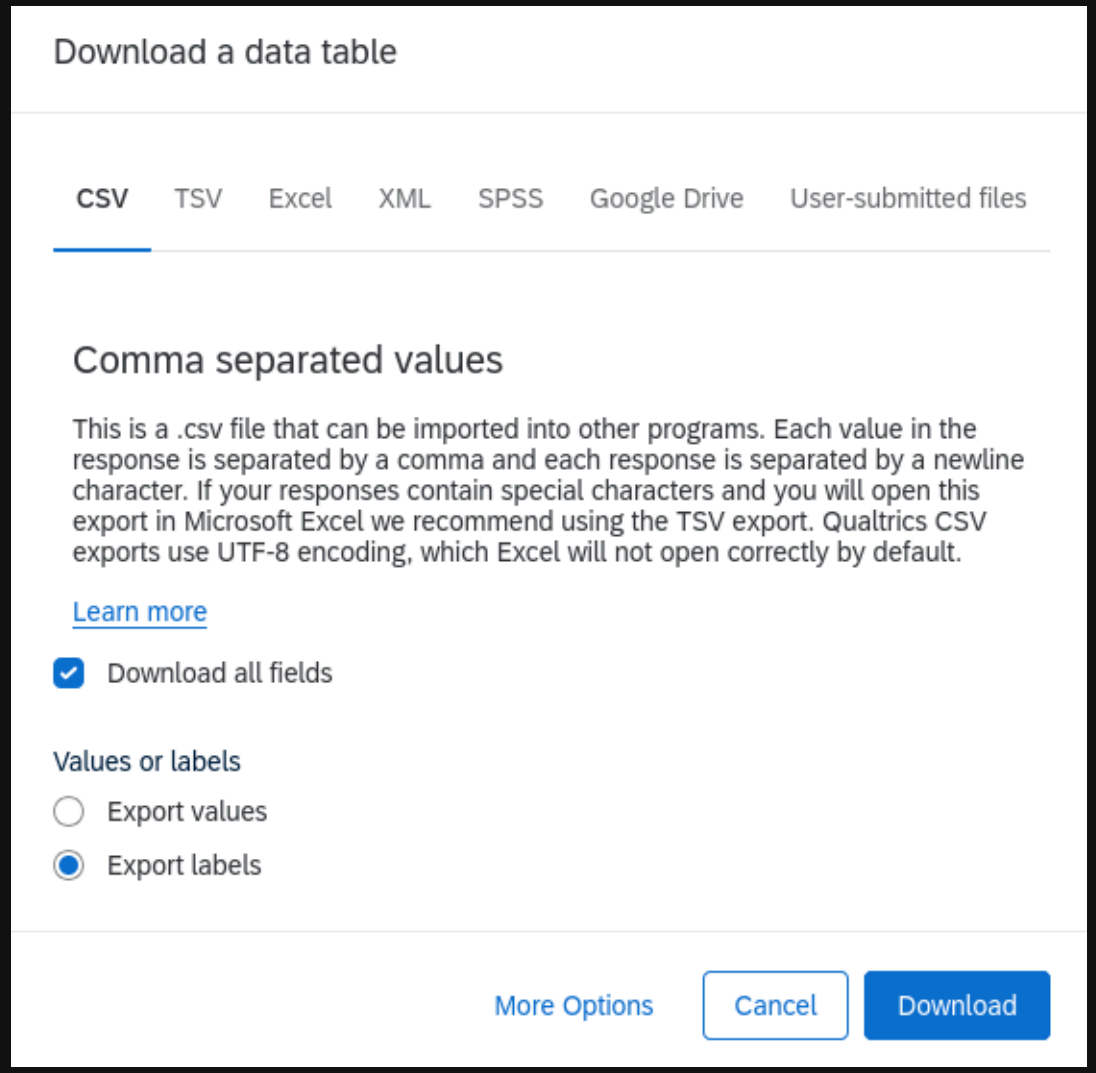


Qualtrics data download

Download options

You can download data directly from this page

- Do it 2-3 times, you ***may*** want to program it!



Qualtrics data download options

Downloaded data

The data downloaded depends on parameters chosen. For instance, downloading as CSV with default settings yields

```
1 StartDate,EndDate,Status,Progress,Duration (in seconds),Finished,RecordedDate,Response
2 Start Date,End Date,Response Type,Progress,Duration (in seconds),Finished,Recorded Date
3 "{""ImportId"":""startDate"",""timeZone"":""America/New_York""}","{""ImportId"":""endDa
4 2025-07-01 11:13:44,2025-07-01 11:14:18,IP Address,100,34,True,2025-07-01 11:14:19,R_5
5 2025-07-01 11:23:01,2025-07-01 11:23:28,IP Address,100,26,True,2025-07-01 11:23:28,R_5
```

Loading the downloaded data into R

You downloaded the responses from the Qualtrics web interface (previous slide). Now read that exported file into R.

```
datesuffix <- "June+16,+2026_15.35"
fileprefix <- "Testing+preservation_"
filename <- paste0(fileprefix, datesuffix, ".csv")
```

Some data org hygiene

We want to be careful about managing our data structure:[3]

```
# Path to the file you downloaded from Qualtrics
datapath <- here::here("data")
rawdatapath <- file.path(datapath, "raw-confidential")
confdatapath <- file.path(datapath, "confidential")
cleandatapath <- file.path(datapath, "clean")
metadatapath <- file.path(datapath, "metadata")
```

Minor thing

Let’s ensure that these paths all exist!

```
for (path in list(rawdatapath, confdatapath, cleandatapath, metadatapath)) {
  if (!dir.exists(path)) {
    dir.create(path, recursive = TRUE)
    message("Created directory: ", path)
  } else {
    message("Directory already exists: ", path)
  }
}
```

```
Directory already exists: /home/runner/work/tutorial-preserving-survey/tutorial-preserving-survey/data/raw-confidential
```

```
Created directory: /home/runner/work/tutorial-preserving-survey/tutorial-preserving-survey/data/confidential
```

```
Created directory: /home/runner/work/tutorial-preserving-survey/tutorial-preserving-survey/data/clean
```

```
Created directory: /home/runner/work/tutorial-preserving-survey/tutorial-preserving-survey/data/metadata
```

Loading the downloaded data into R

- Any CSV reader works too, with some adjustments.

```
library(readr)
# discard the two Qualtrics metadata rows
data.raw <- read_csv(file.path(rawdatapath, filename), skip = 3,
                     col_names = FALSE)
# read the header separately to get column names
header <- read.csv(file.path(rawdatapath, filename),
                     nrows=0)
colnames(data.raw) = colnames(header)
```

Loading the downloaded data into R

```
1 head(data.raw)
```

```
# A tibble: 6 × 17
  StartDate           EndDate             Status  Progress Duration..in.seconds.
  <dttm>              <dttm>              <chr>      <dbl>                  <dbl>
1 2025-07-01 11:13:44 2025-07-01 11:14:18 IP Add…      100                     34
2 2025-07-01 11:23:01 2025-07-01 11:23:28 IP Add…      100                     26
3 2025-07-01 11:26:40 2025-07-01 11:26:40 Survey…      100                      0
4 2025-07-01 11:27:12 2025-07-01 11:27:12 Survey…      100                      0
5 2025-07-01 11:30:26 2025-07-01 11:30:47 IP Add…      100                     21
6 2025-12-19 12:20:44 2025-12-19 12:21:19 IP Add…      100                     35
# i 12 more variables: Finished <lgl>, RecordedDate <dttm>, ResponseId <chr>,
#   DistributionChannel <chr>, UserLanguage <chr>, consent <chr>, age_1 <dbl>,
#   gender <chr>, education <chr>, num_tabs_1 <dbl>, name_confidential <chr>,
#   number_confidential <chr>
```

Loading with `qualtRics` package

- The `qualtRics`[4] package can read a Qualtrics CSV export directly with `read_survey()`:

```
1 library(qualtRics)
2
3 data.raw <- read_survey(file.path(rawdatapath, filename))
```

Loading with `qualTRics` package

```
1 head(data.raw)
```

```
# A tibble: 6 × 17
  StartDate           EndDate             Status Progress Duration (in seconds…[1]
  <dttm>              <dttm>              <chr>     <dbl>                    <dbl>
1 2025-07-01 11:13:44 2025-07-01 11:14:18 IP Ad…      100                       34
2 2025-07-01 11:23:01 2025-07-01 11:23:28 IP Ad…      100                       26
3 2025-07-01 11:26:40 2025-07-01 11:26:40 Surve…      100                        0
4 2025-07-01 11:27:12 2025-07-01 11:27:12 Surve…      100                        0
5 2025-07-01 11:30:26 2025-07-01 11:30:47 IP Ad…      100                       21
6 2025-12-19 12:20:44 2025-12-19 12:21:19 IP Ad…      100                       35
# i abbreviated name: [1]`Duration (in seconds)`
# i 12 more variables: Finished <lgl>, RecordedDate <dttm>, ResponseId <chr>,
#   DistributionChannel <chr>, UserLanguage <chr>, consent <chr>, age_1 <dbl>,
#   gender <chr>, education <chr>, num_tabs_1 <dbl>, name_confidential <chr>,
#   number_confidential <chr>
```

Cleaning data

- We filter the data to only include those who consented
- We remove survey preview responses
- (Optionally) remove responses that took place outside the relevant window.
- Remove confidential data (variables `name_confidential` and `number_confidential` in our survey, for example).

```
data.confidential <- data.raw |>
    filter(consent == "Yes") |>
    filter(Status != "Survey Preview") |>
    filter(StartDate > QUALTRICS_STIME & EndDate < QUALTRICS_ETIME) |>
    select(StartDate,EndDate,Status,Finished,RecordedDate,
           ResponseId,consent,age_1,gender,education,
           num_tabs_1,name_confidential,number_confidential)
data.clean <- data.confidential %>%
  select(-name_confidential, -number_confidential)
```

Cleaning data by selection

We could also simply **not select** the confidential data if we don't actually need it.

```
data.clean <- data.raw |>
   filter(consent == "Yes") |>
   filter(Status != "Survey Preview") |>
   filter(StartDate > QUALTRICS_STIME & EndDate < QUALTRICS_ETIME) |>
   select(StartDate,EndDate,Status,Finished,RecordedDate,
          ResponseId,consent,age_1,gender,education,num_tabs_1)
```

Using confidential data

We could also (hypothetically) immediately compute variables that rely on confidential data.

```
# not run
 data.clean <- data.raw |>
    filter(consent == "Yes") |>
    filter(Status != "Survey Preview") |>
    filter(StartDate > QUALTRICS_STIME & EndDate < QUALTRICS_ETIME) |>
    select(StartDate,EndDate,Status,Finished,RecordedDate,
    ResponseId,consent,age_1,gender,education,num_tabs_1,
           gps_lat, gps_lon) |>
    mutate(distance = compute_distance_from_cornell(
    gps_lat,gps_lon,precision="100m")) |>
    select(-gps_lat, -gps_lon)
```

Saving confidential and clean data

- save the confidential data to a **clearly marked** folder (J-PAL policy: an encrypted volume)
- save the cleaned publishable data to a **well-defined** folder.

```
# save confidential data NOT for publishing, if needed
write.csv(data, file.path(confdatapath,"confidential_data.csv"),
          row.names = FALSE)
saveRDS(data, file.path(confdatapath,"confidential_data.rds"))
# saving clean data for publishing
write.csv(data, file.path(cleandatapath,"clean_data.csv"),
           row.names = FALSE)
```

Descriptive statistics

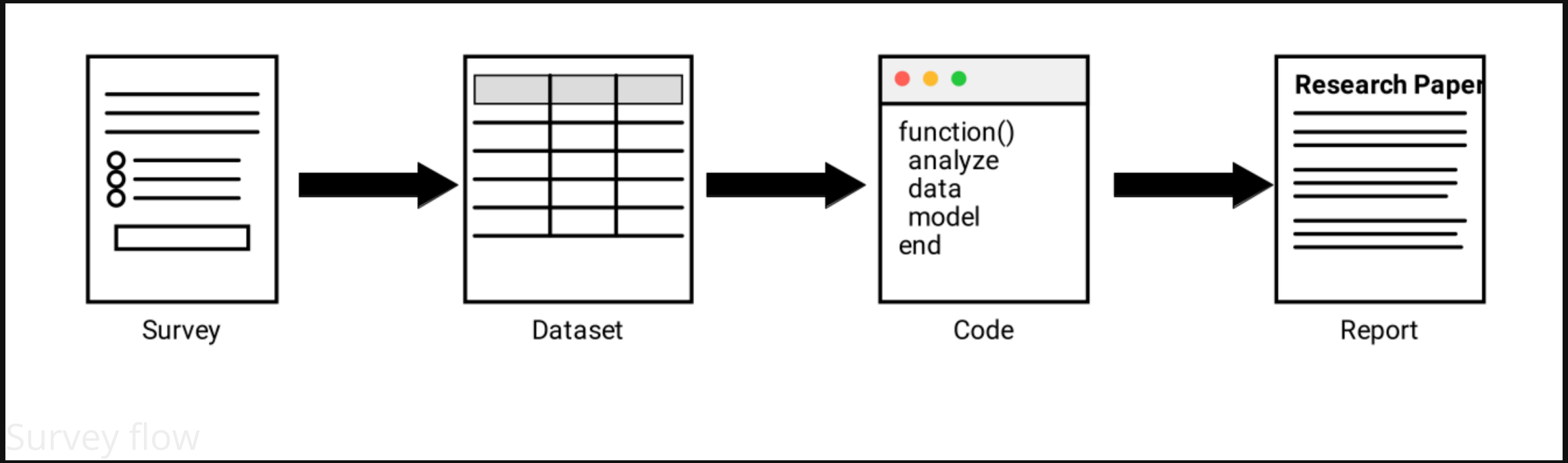
Now you are ready use your cleaned data for reproducible analyses!

Stepping it up

Stepping up the process

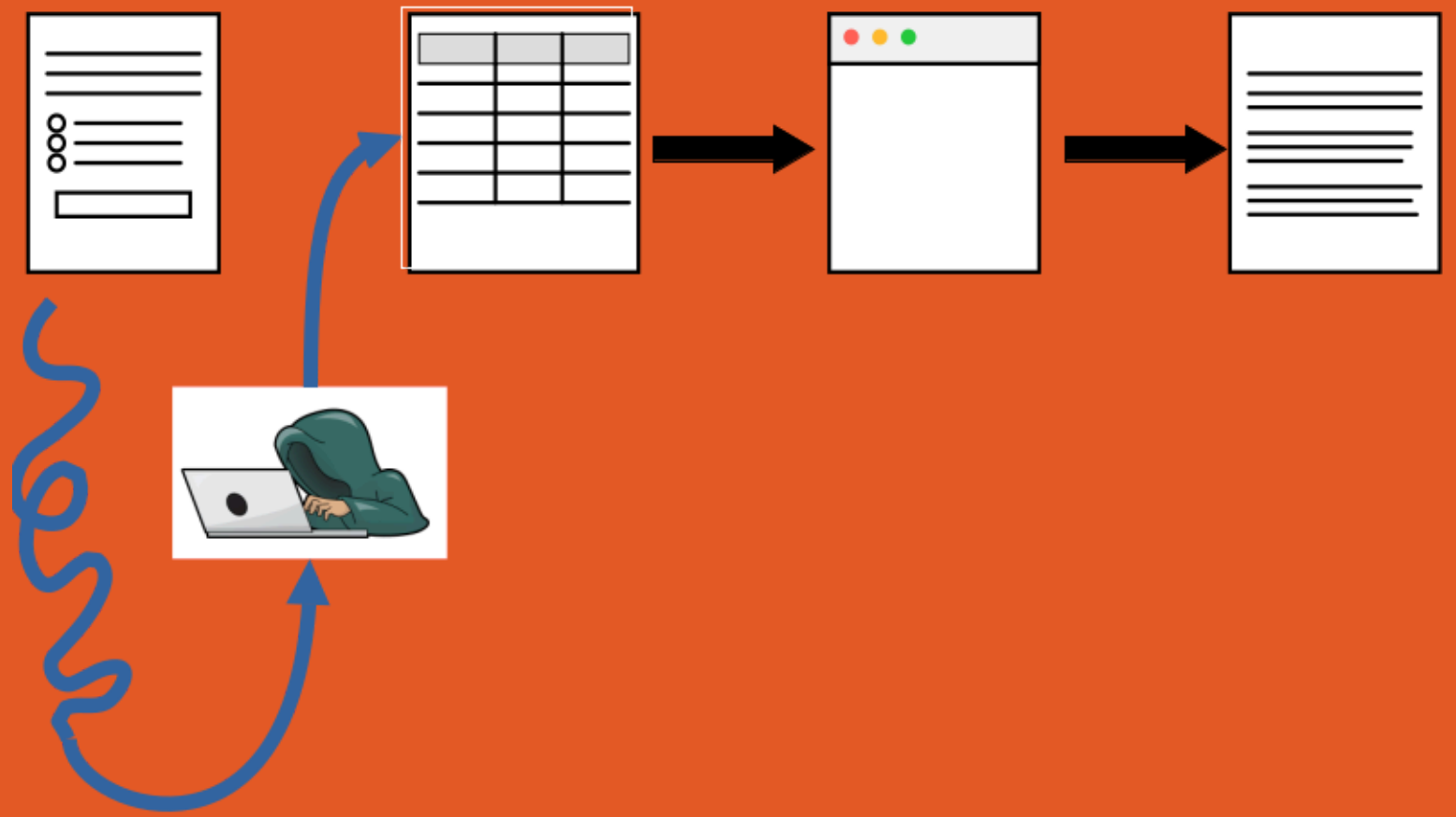
- API,
- checksums,
- trusted systems
- preservation

Credibility of the data flow

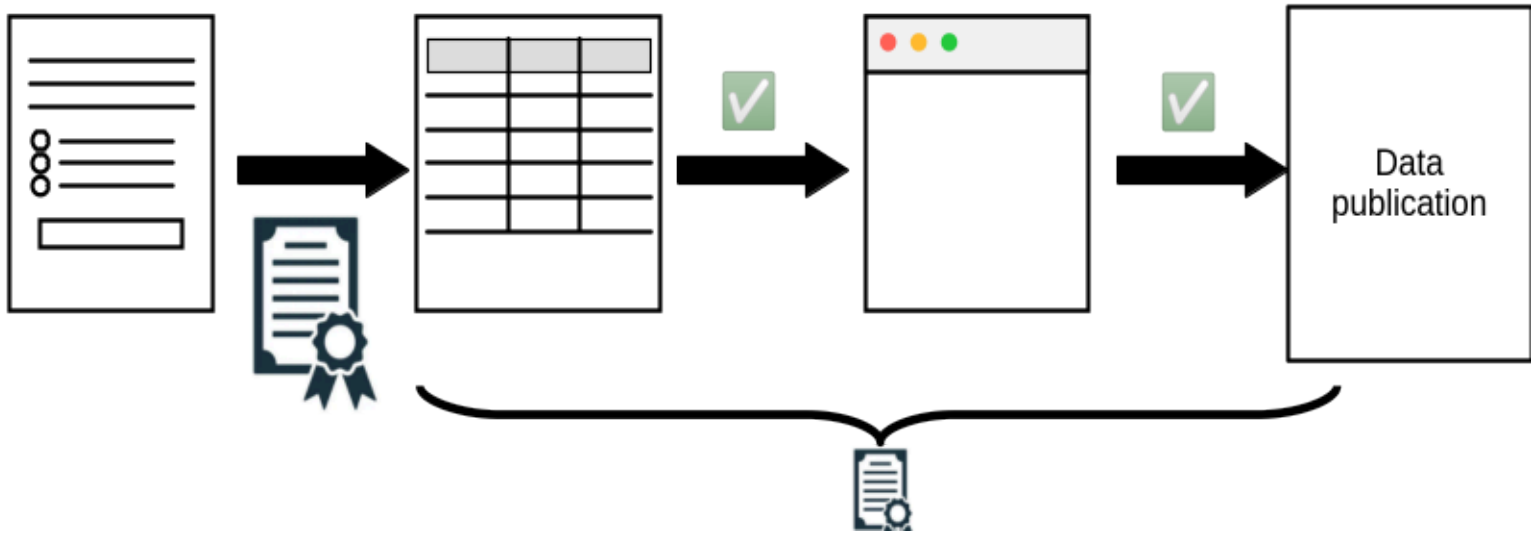


Survey flow

What could go wrong?



How to CERTIFY the full process?



Survey flow

Taking it a step further

- Survey tool provider (Qualtrics, etc.) exports data, posts checksum
- Survey tool provider exports data only to institution directly into trusted repository, researchers obtain data from there (with privacy protections)
- Has been discussed by authors behind Data Colada
- Don’t hold your breath…

Using automation

APIs

- An API (***Application Programming Interface***) is a mechanism that enables two software components to communicate with each other
- APIs can be used to request data or services and get responses without needing to know how the other program works internally
- We will use APIs to streamline and **automate** the processing

Loading data from Qualtrics using an API

We need to know a few things:

- the URL we want to use, defined by a generic part, and a survey specific part
- these are **public** - no need for secrecy

```
1  # qualtrics URL components
2  QUALTRICS_FULL_URL <- "first part of survey URL"
3
4  QUALTRICS_SURVEY <- "second part of survey URL, usually starts with SV"
```

Loading data from Qualtrics using an API

We may want to limit the responses we download programmatically. This is not part of API, but of good programming practices.

```
1 # Keep only responses in the desired window of time
2 QUALTRICS_STIME <- ymd_hms("2025-07-01 00:00:01")
3 QUALTRICS_ETIME <- ymd_hms("2025-08-26 23:59:00")
```

Fetching the data with the API

The API call ***replaces the manual download*** from before:

```
data.raw <- fetch_survey(surveyID = QUALTRICS_SURVEY,
                         verbose = TRUE)
```

BUT: Privacy! Confidentiality!

Can anybody just download these data?

> NO!

We need to authenticate, but not by entering a password manually.

> That’s where the **API token** comes in.

Fetching the data with the API

We need to set an **API token**, then we can download this.

```
1 if (Sys.getenv("QUALTRICS_API_KEY") != "") {
2   data.raw <- fetch_survey(surveyID = QUALTRICS_SURVEY, verbose = TRUE)
3 } else {
4   stop("Please set the QUALTRICS_API_KEY environment
5   variable to your API key.")
6 }
```

The rest of the pipeline is unchanged

`data.raw` now comes from the API instead of a downloaded file — but the cleaning and saving steps from before are **exactly the same**:

```
clean_data <- data.raw |>
    filter(consent == "Yes") |>
    filter(Status != "Survey Preview") |>
    filter(StartDate > QUALTRICS_STIME & EndDate < QUALTRICS_ETIME) |>
    select(StartDate,EndDate,Status,Finished,RecordedDate,
    ResponseId,consent,age_1,gender,education,num_tabs_1)

write.csv(clean_data, file.path(publicdata,"clean_data.csv"),
row.names = FALSE)
```

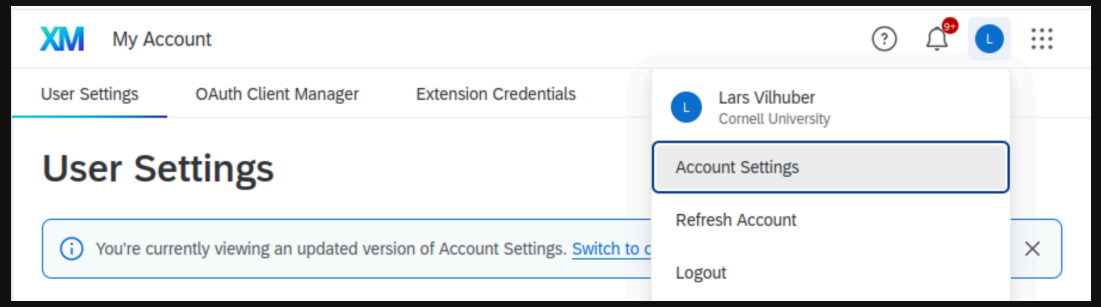
Not unique to Qualtrics

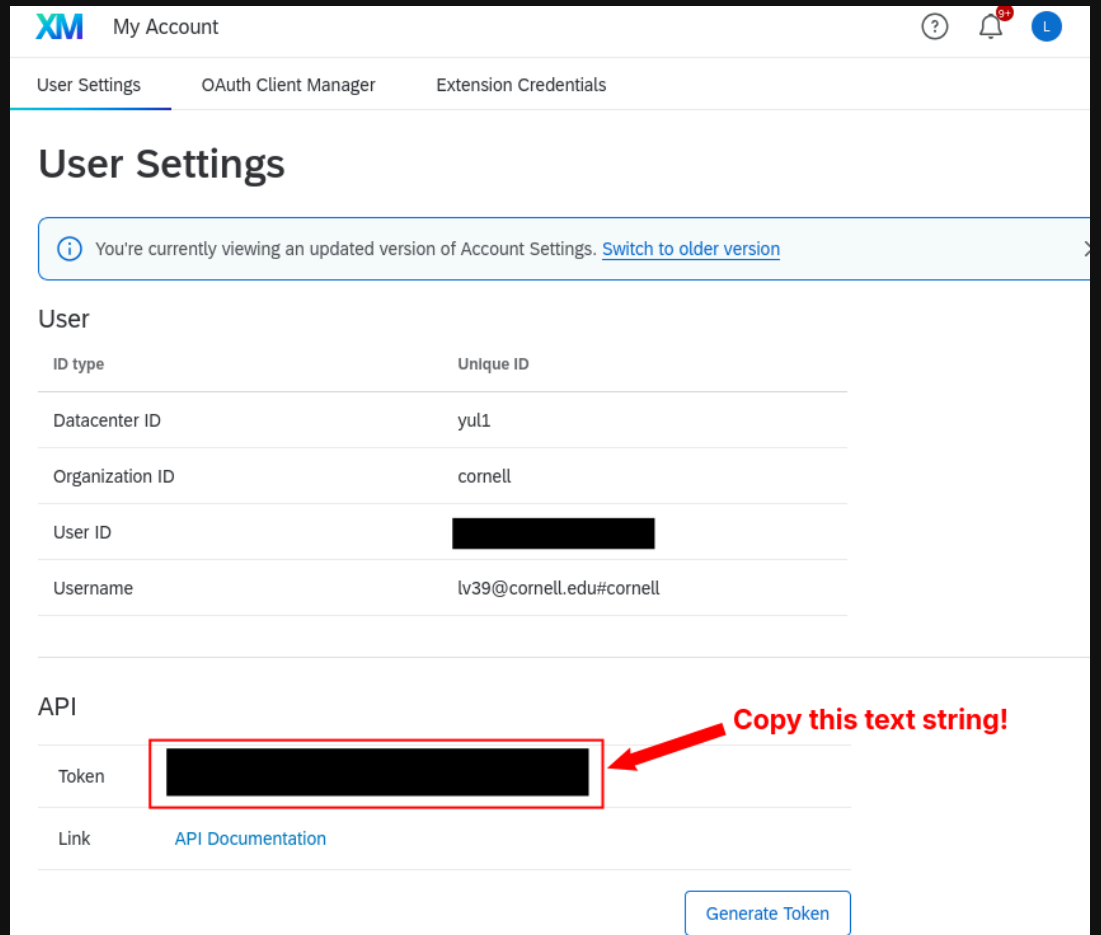
- LimeSurvey -> `limer`
- SurveyCTO -> `rsurveycto`
- Google Forms -> `googlesheets4`

And of course works just fine in Python (and via Python, could use Stata!)

Qualtrics and API tokens.

An API token is assigned to your **Qualtrics account**. Where do you find it?





Setting API tokens

> Not specific to the Qualtrics API!

- Set it manually:

```
1  Sys.setenv(QUALTRICS_API_KEY = "ab7ece8b")
```

- Set it using environment variables stored outside your code (e.g., in `.Renviron` file) - ***good for testing***

```
1  # This is .Renviron
2  QUALTRICS_API_KEY="ab7ece8b"
```

Setting API tokens

We want to automate on cloud servers!

- Push these “***secrets***” to `GitHub Secrets` and load it in `GitHub Actions` [link]

Using API tokens

Now we need to make it available to our code (regardless of where it comes from)

```
# Here environment variables are read from .Renviron
QUALTRICS_API_KEY <- Sys.getenv("QUALTRICS_API_KEY")
```

Full code

Now this works both ***locally*** and on ***cloud servers*** without any manual interaction!

```
QUALTRICS_FULL_URL <- "first part of survey URL"
QUALTRICS_SURVEY <- "second part of survey URL, usually starts with SV"

if (Sys.getenv("QUALTRICS_API_KEY") != "") {
  data.raw <- fetch_survey(surveyID = QUALTRICS_SURVEY, verbose = TRUE)
} else {
  stop("Please set the QUALTRICS_API_KEY environment
  variable to your API key.")
}
```

Side-note: Qualtrics API credentials

Qualtrics API credentials cannot be restricted to a single survey.

skip

Traditional Static API Tokens (`X-API-TOKEN`)

- **The Limit:** You can only have **one** active static API token per user account at a time.
- **The Catch:** If you go to your account settings and generate a new token for a second application, it will **immediately overwrite and invalidate** the old token, breaking your first application.

OAuth 2.0 Client Credentials

Separate, independent credentials for different applications:

- **Account Settings** > **Qualtrics IDs** > **OAuth Client Manager**.
- Click **Create Client** to generate unique sets of `Client ID` and `Client Secret` credentials.
- You can create **multiple clients** for different applications
- To revoke access for one app, you can delete its specific client without affecting the others.

BUT: API Key access ALL your surveys

- Regardless of method, the API key can access ALL of your surveys!
- There is **NO** way to restrict that natively.

That’s a problem.

Workaround: “Service Account”

- You can create a specific user, say `jpal-survey-user`
- Requires support from system administrator
- Once the user is created, proceed as before, but when logged in as `jpal-survey-user`!
- As `YOU`, share only the specific survey with `jpal-survey-user` and give it only the permissions it needs

It’s a bit more complicated…

Secrets

Secrets (Github version)

- You will want to keep APIs key safe using **GitHub Secrets**.
- Secrets allow you to store sensitive information in the repository environment.
- Use the secret as an environment variable in the GitHub workflow file.

Storing secrets in `.Renviron` locally

You already have a `.Renviron` for local development:

```
1 QUALTRICS_API_KEY='something here'
```

- Do not publish this file!
- Do not commit it to Github![5]

Storing secrets in Github

- Enter them manually in the GitHub web interface
- Use the `.Renviron` file to set the GitHub Actions secrets with the Github CLI:

```
1
2 ✓ Set Actions secret DATAVERSE_TOKEN for
3 ✓ Set Actions secret QUALTRICS_BASE_URL
4 ✓ Set Actions secret DATAVERSE_SERVER fo
5 ✓ Set Actions secret QUALTRICS_API_KEY f
6 ✓ Set Actions secret DATAVERSE_DATASET_D
```

```
1 gh secret set -f .Renviron
```

Using secrets in GitHub Actions

In GitHub workflows, set your environment variables:

```
1  echo "QUALTRICS_API_KEY=${{ secrets.QUALTRICS_API_KEY }}" >> $GITHUB_ENV
```

Using secrets in Github Actions

- R code does **not** need to be adapted!

```r
# Same R code as before!
if (Sys.getenv("QUALTRICS_API_KEY") != "") {
  data.raw <- fetch_survey(surveyID = QUALTRICS_SURVEY, verbose = TRUE)
} else {
  stop("Please set the QUALTRICS_API_KEY environment
  variable to your API key.")
}
```

Checksums

Checksums can be used to demonstrate consistency

- A **checksum** is a single value calculated from a data file that can be used to verify the integrity of the file.
- The `sha256` algorithm is commonly used for this purpose.

Checksums can be used to demonstrate consistency

- In R, the package `digest`[6] is available.

```
1 library(digest)
2 # Calculate the checksum of a file
3 digest(trees)
```

```
[1] "370a7132861fb520bd721d9bcbe008a4"
```

```
1 digest(trees,algo="sha256")
```

```
[1] "70823d6c0cebdc582b388f3ec56930bd2ec5dd176272032d949b93319d74d17b"
```

Adding checksums to the data download

```
if (Sys.getenv("QUALTRICS_API_KEY") != "") {
  data.raw <- suppressMessages(fetch_survey(surveyID = QUALTRICS_SURVEY, verbose = FAL
  data.raw.sha256 <- digest::digest(data.raw, algo = "sha256")
  message("Checksum of the downloaded data: ", data.raw.sha256)
  # Write checksum to a file
  writeLines(data.raw.sha256, file.path(metadatapath, "data.raw.sha256"))
} else {
  stop("Please set the QUALTRICS_API_KEY environment
  variable to your API key.")
}
```

```
Checksum of the downloaded data:
a0e2146acc752debff66a670fe654c7618c45bacf6b8c634e58a21d5999fd222
```

How does that help?

- Subsequent downloads can verify that the download is the same as originally downloaded!

```
1  # Read the original checksum from file
2  original.sha256 <- readLines(file.path(metadatapath, "data.raw.sha256"))
3  message("Original checksum from file: ", original.sha256)
```

```
Original checksum from file:
a0e2146acc752debff66a670fe654c7618c45bacf6b8c634e58a21d5999fd222
```

```
1  # Redownload data
2  if (Sys.getenv("QUALTRICS_API_KEY") != "") {
3    data.raw <- suppressMessages(fetch_survey(surveyID = QUALTRICS_SURVEY, verbose = FAL
4    data.raw.sha256 <- digest::digest(data.raw, algo = "sha256")
5    message("Checksum of the downloaded data: ", data.raw.sha256)
6  }
```

```
Checksum of the downloaded data:
a0e2146acc752debff66a670fe654c7618c45bacf6b8c634e58a21d5999fd222
```

How does that help?

- Subsequent downloads can verify that the download is the same as originally downloaded!

```
# Compare the checksums
if (original.sha256 == data.raw.sha256) {
  message("Checksums match! Data integrity verified.")
} else {
  warning("Checksums do NOT match! Data may have changed/ been altered.")
}
```

```
Checksums match! Data integrity verified.
```

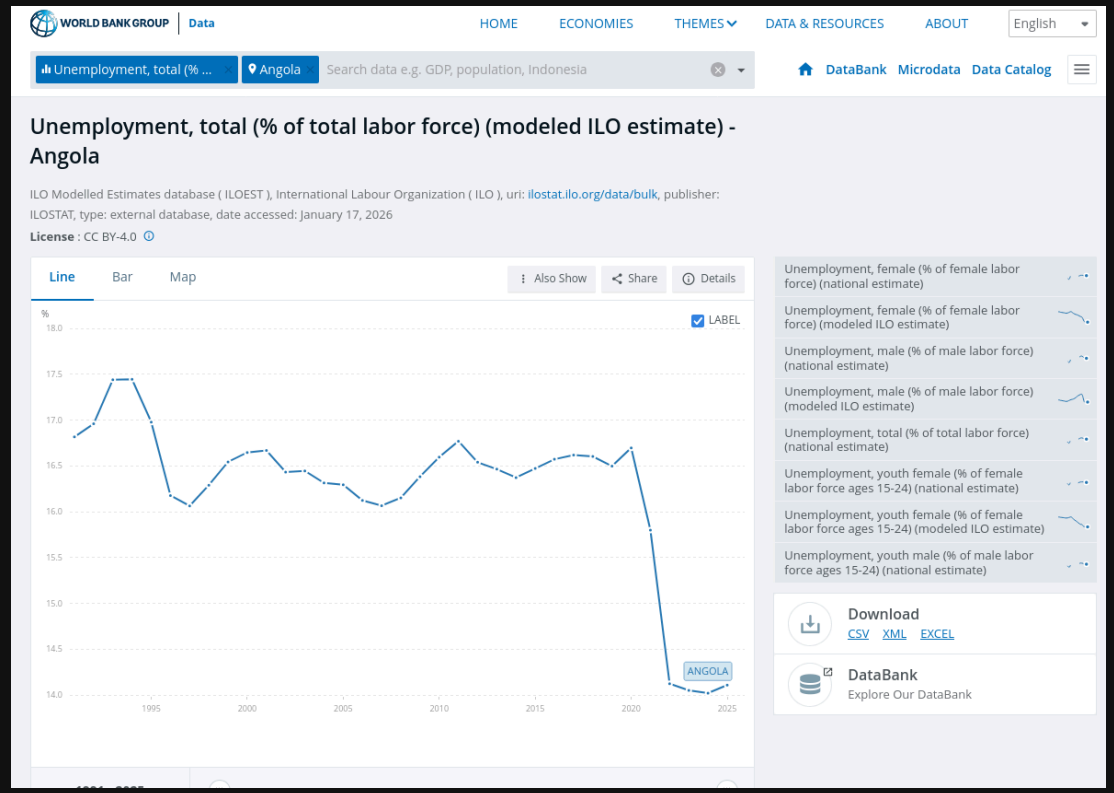
Preservation

Credibility of Survey Data

- You run a study using the ***PSID***. Do you **trust** the downloaded data?

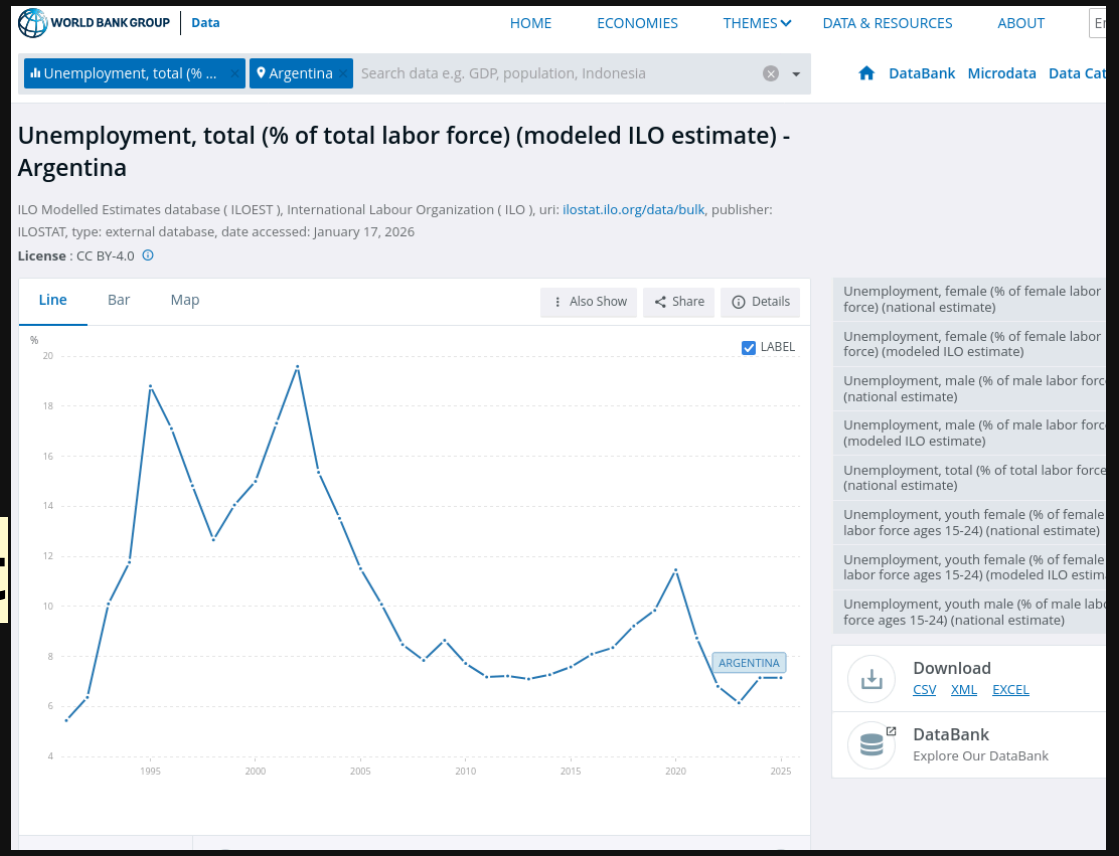
Credibility of Government Data

- You run a study using the PSID. Do you trust the downloaded data?
- You use unemployment data for **Angola** through ***World Bank Data Bank***. Do you **trust** the downloaded data?



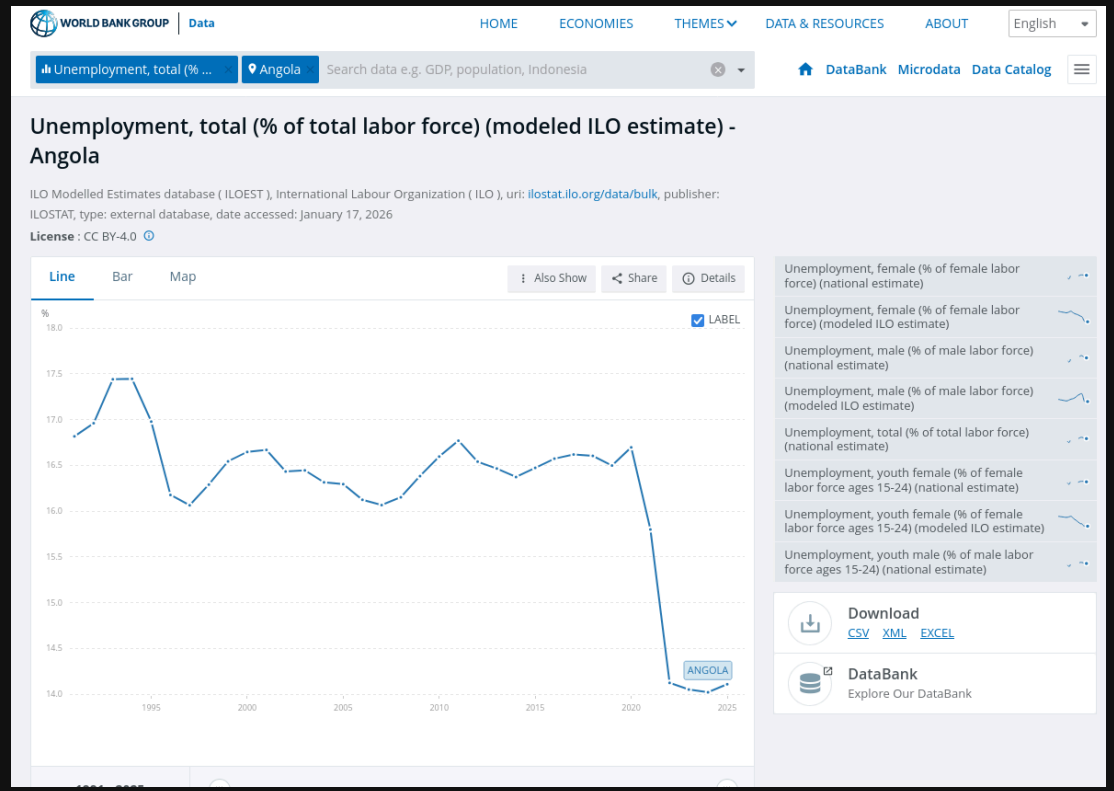
Credibility of Government Data

- You run a study using the PSID. Do you trust the downloaded data?
- You use unemployment data for **Argentina** through World Bank Data Bank. Do you **trust** the downloaded data?



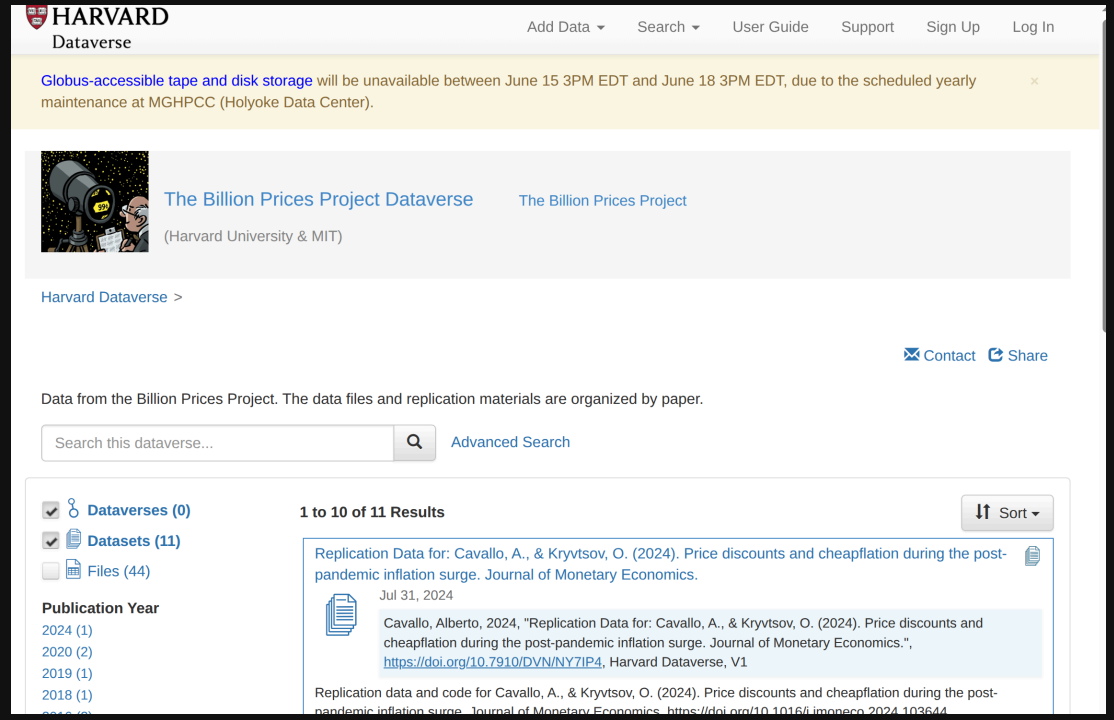
Credibility of Government Data

- You run a study using the PSID. Do you trust the downloaded data?
- You use unemployment data for **United States** through World Bank Data Bank. Do you **trust** the downloaded data?



Credibility of Researcher provided data

- You run a study using the PSID. Do you trust the downloaded data?
- You use inflation data for ***Argentina*** through a **research deposit on Dataverse**. Do you **trust** the downloaded data?

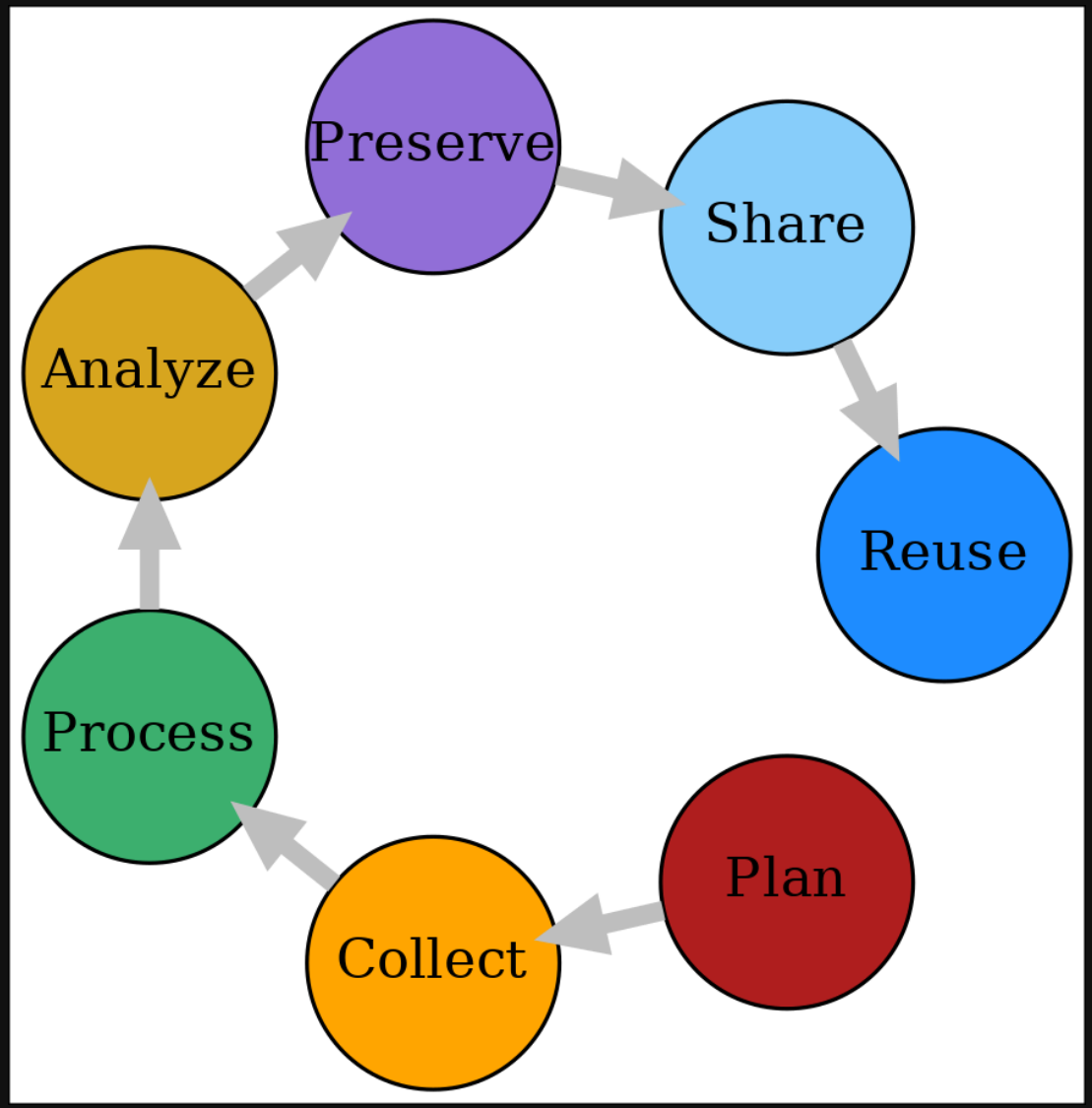


Let’s make you become a mini-PSID

Timing

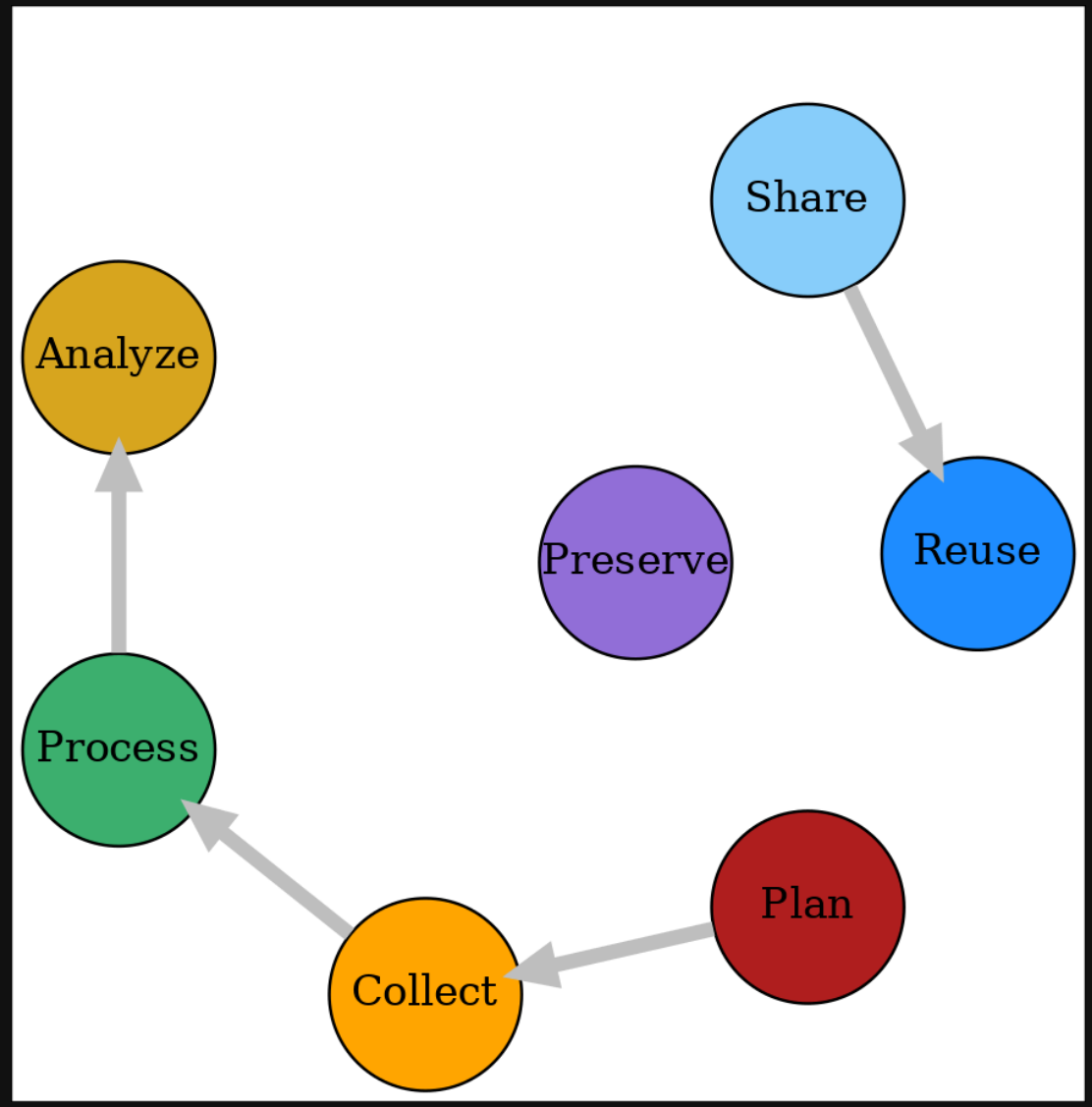
Once you have registered your analysis plan - should the **processing and analysis** really change?

Once you have **collected** the data - is it really going to change?



Modified Data and Workflow

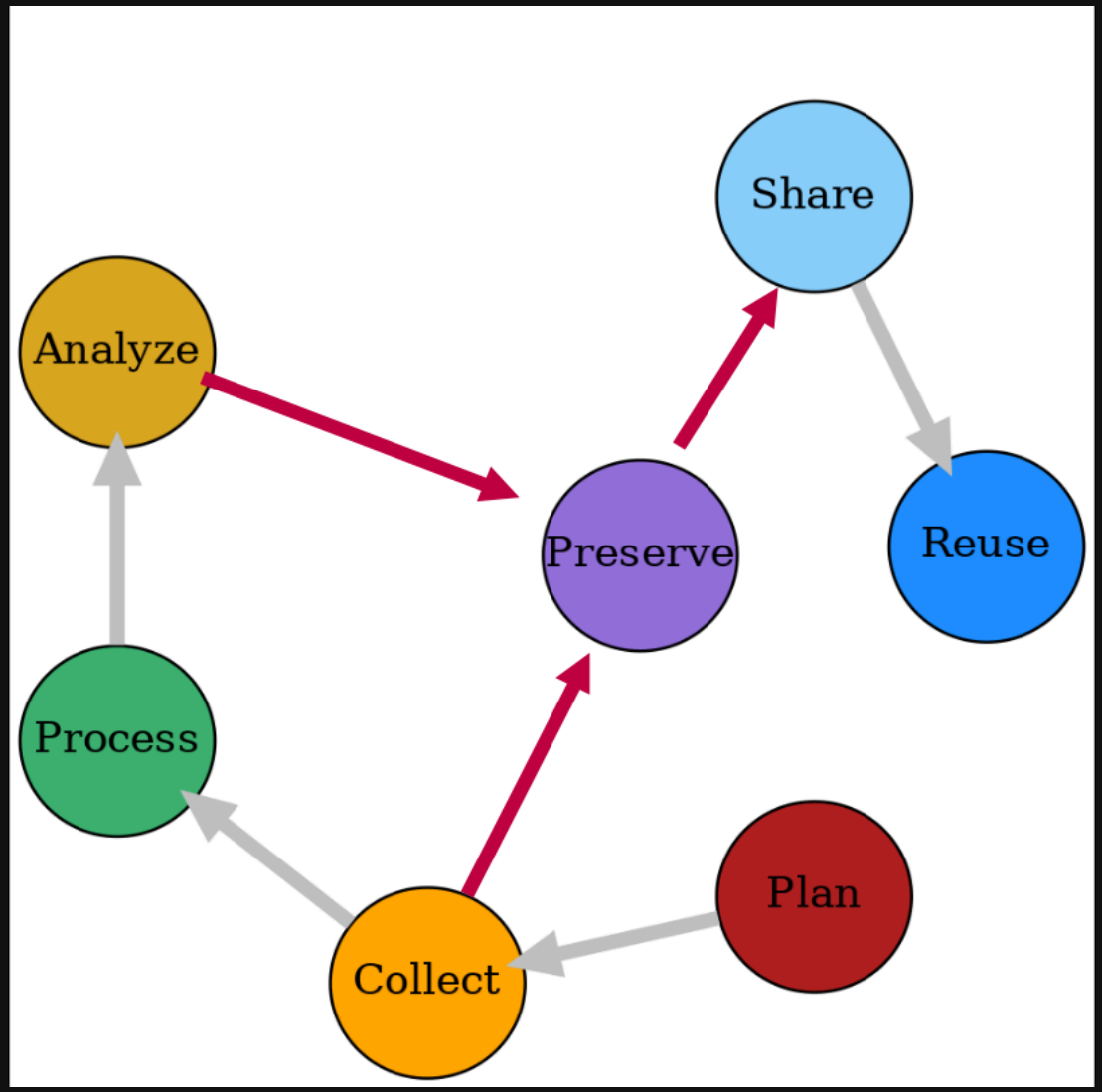
Let’s consider the preservation part separately:



Modified Data and Workflow

Proposal:

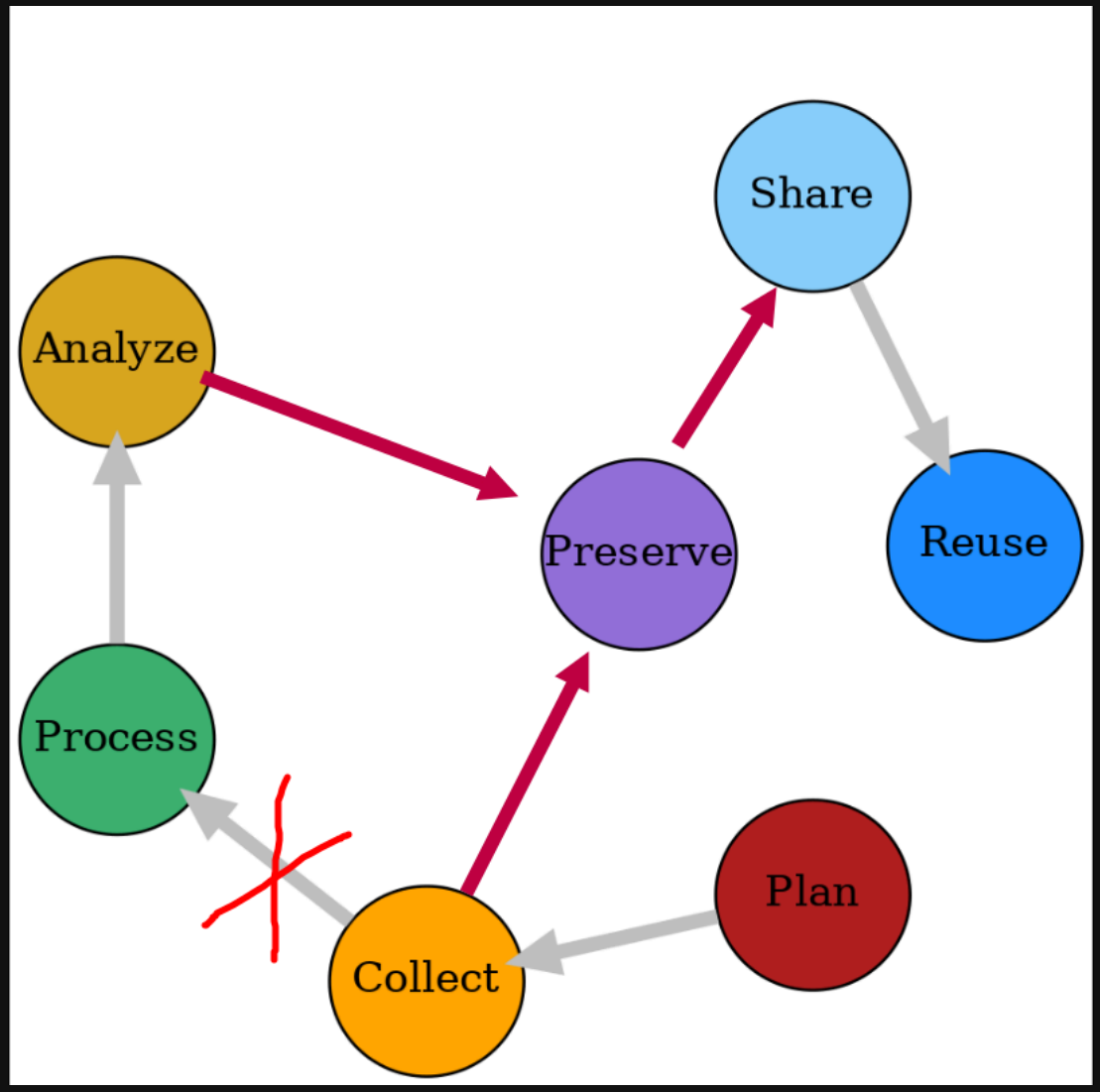
- Preserve as you go



Modified Data and Workflow

Proposal:

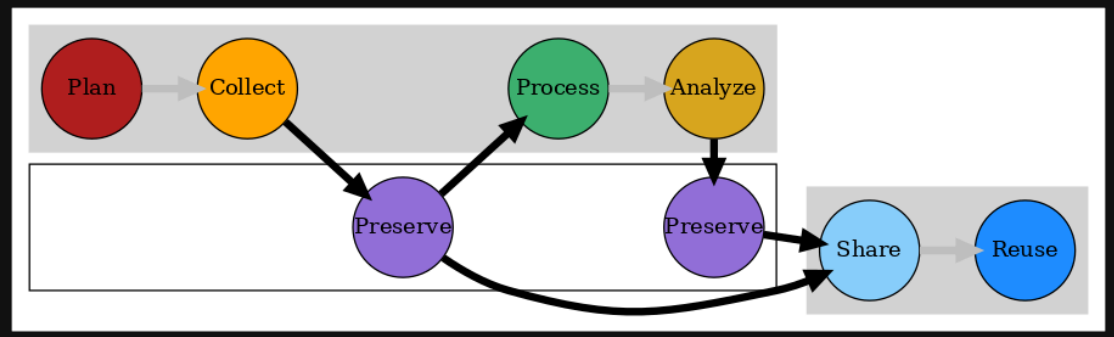
- Preserve as you go
- Use what you preserve



Modified Data and Workflow

Proposal:

- Preserve as you go
- Use what you preserve



Note: Doubtful ethics of others...

I don’t want to be scooped!

Thus, I’m not going to publish my raw data just yet!

What is preservation

Preservation

- Preservation != **publication**, != ***sharing***
- In fact, preservation may mean: ***not very accessible*** at all!
- Preservation is intended to maintain data for tens, even hundreds of years
  - Preservation may involve curation: active transformation of the data for improved accessibility

What is publication

Publication typically involves making information about the data, as well as the data themselves, available to others.

- Publication can initially mean that only **metadata** (information about the data) is published
- In some cases, it may be that **only** metadata is ever published
- But the metadata will point to how to access the data, how long the data will be preserved, and other salient facts

This all seems so complicated

- I need to preserve my data for decades!
- I need to manage the application process for decades!
- Where do I get that DOI thing?
- How to I get Google to index my data?

Options for Preservation (1)

Trusted Repositories

Journals and institutions have assessed a number of trusted repositories:

- CoreTrustSeal has a certification process
- re3data.org lists research data repositories
- Nature, F1000Research, and PLOS have lists of trusted repositories.
- Always check with your journal for specific restrictions or suggestions.

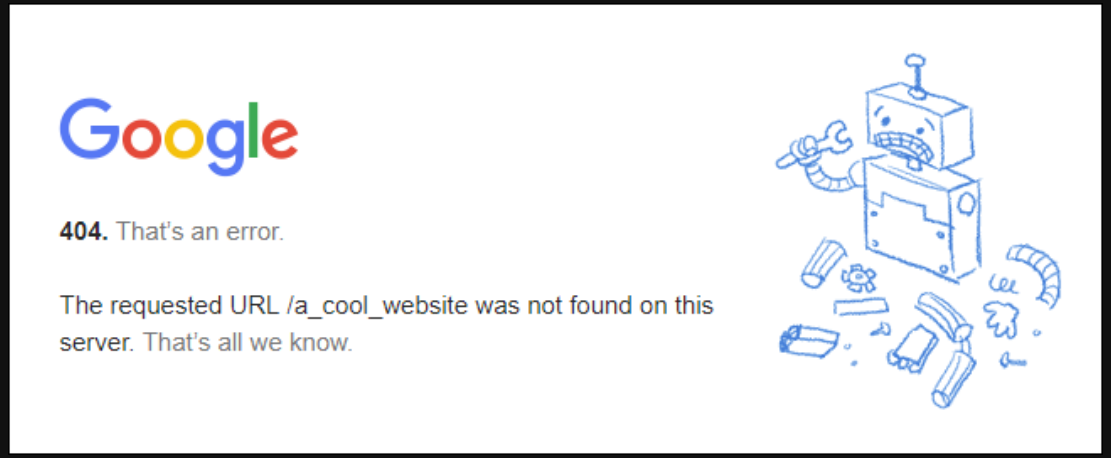
Options for Preservation (2)

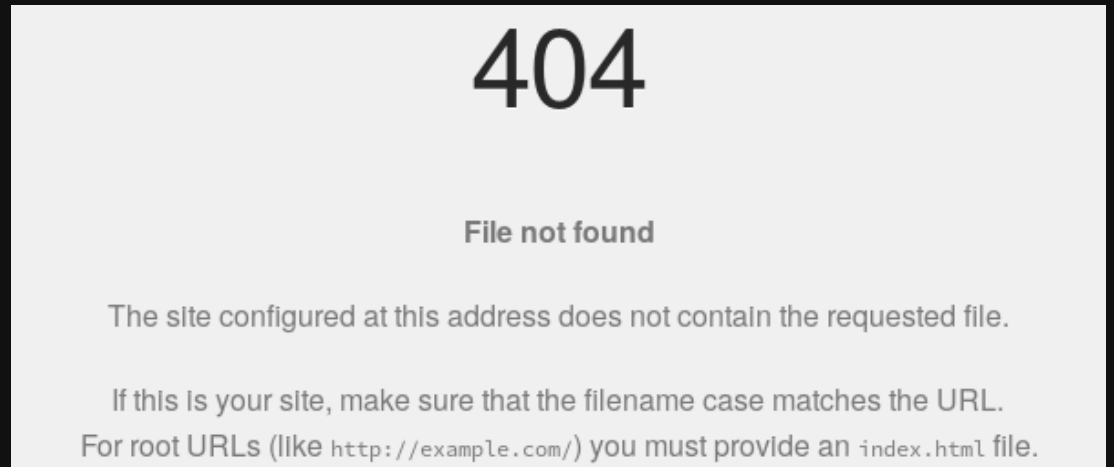
Trusted Repositories

- These generally include at least the following:
  - Dryad Digital Repository
  - figshare
  - Harvard Dataverse
  - ICPSR and OPENICPSR
  - Open Science Framework
  - Zenodo
  - Country or region-specific repositories (that nevertheless generally accept depositors from anywhere): GESIS (Germany), Swedish National Data Service (SND), EASY (Netherlands), CSIRO (Australia), etc.
- Many universities have formal document repositories that may be able to assume such a role; talk to your (data) librarian

What are NOT options for preservation

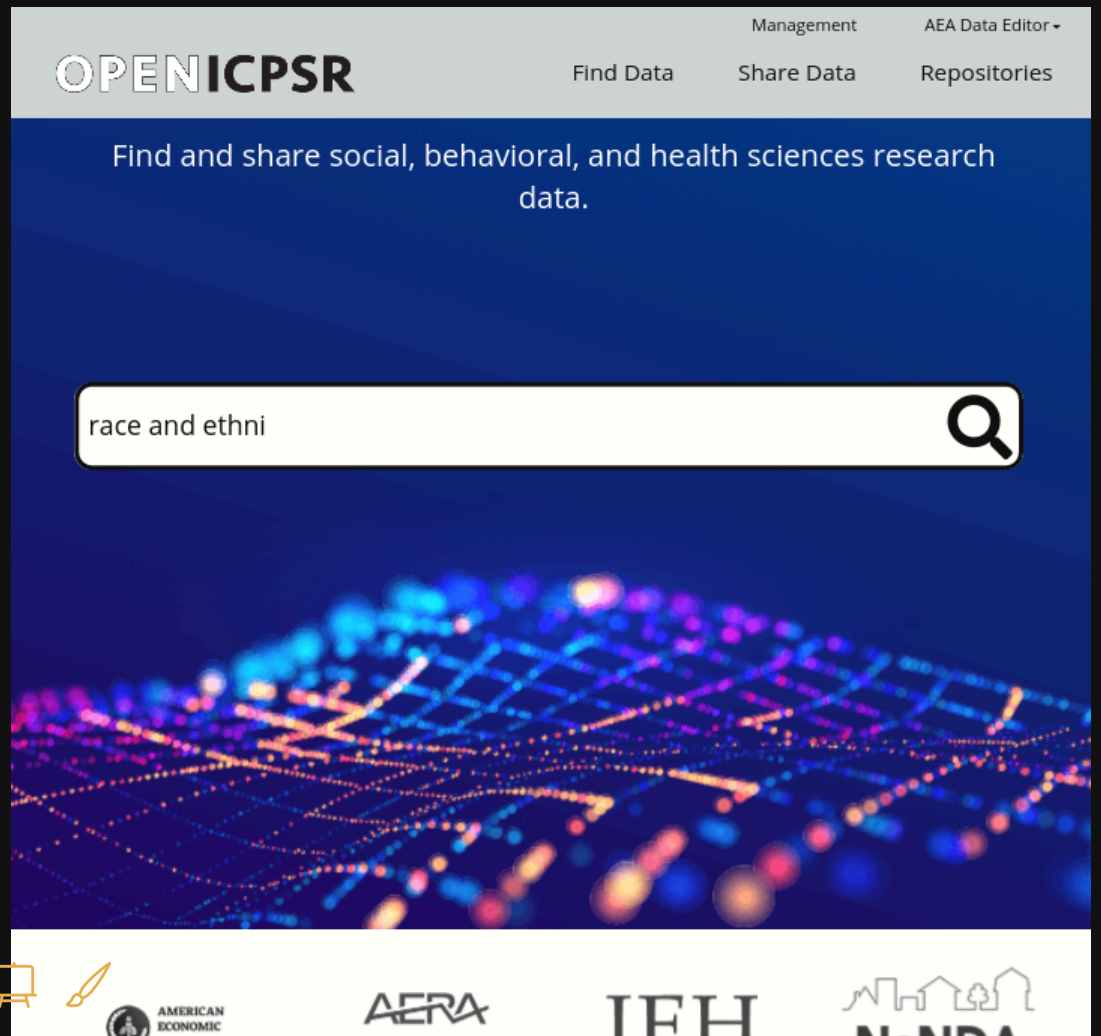
- Github, Gitlab, Bitbucket, etc.
- Dropbox, Box.com, Google Drive, etc.
- Your personal website
- Your university's departmental website





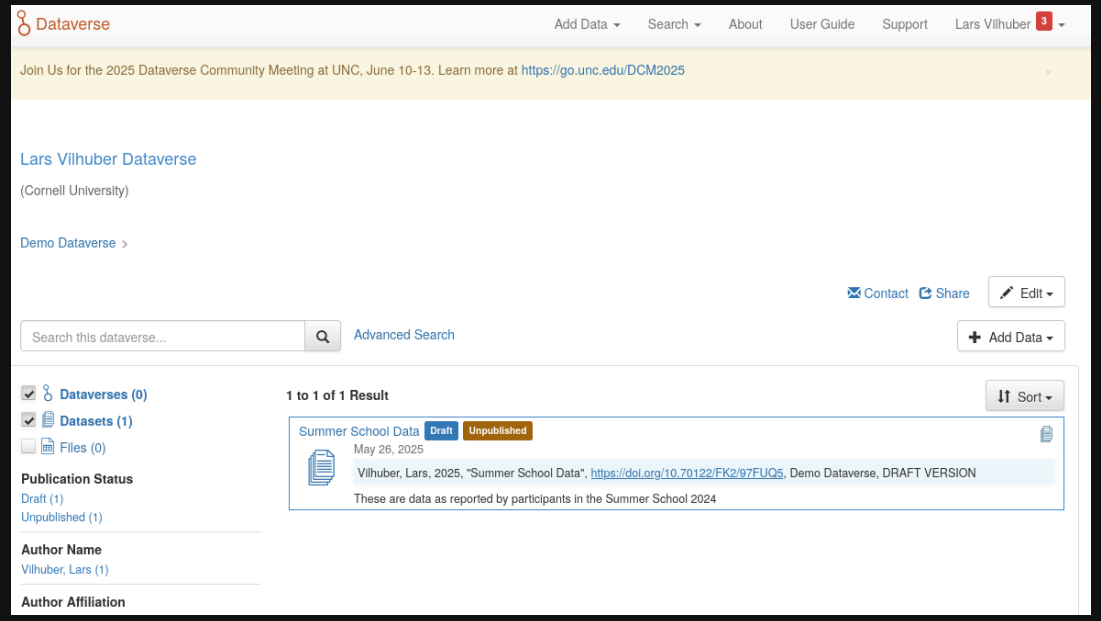
Options for Preservation

In one of my day jobs:

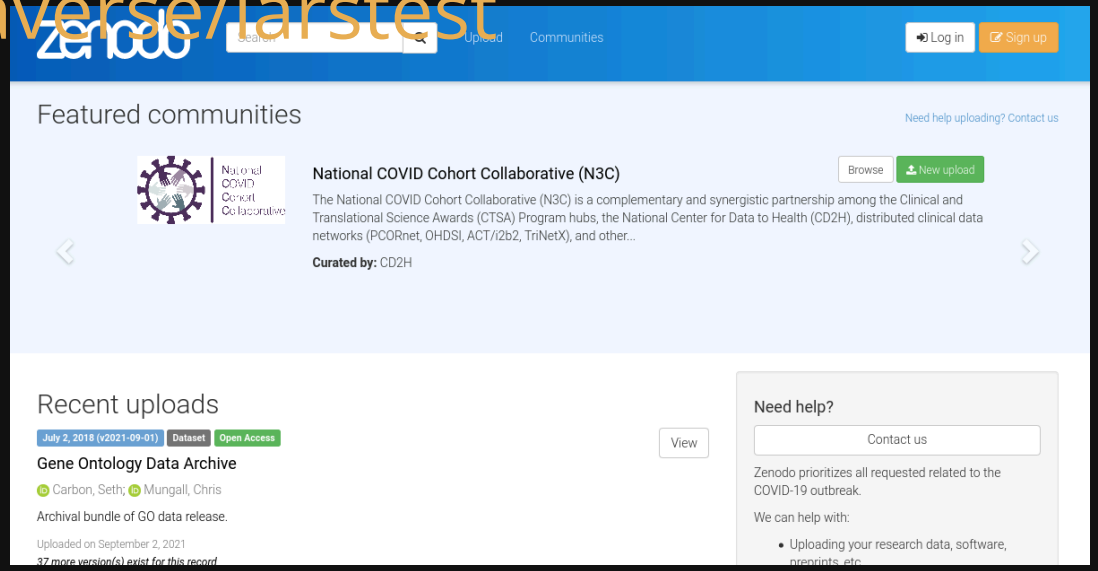


Options for Preservation with API

Dataverse https://demo.dataverse.org/dataverse/larstest



Also Zenodo https://zenodo.org



Getting started on Dataverse

We will NOT use the regular Dataverse; rather, we will test on the demo site.

- This also works with Zenodo: https://sandbox.zenodo.org/
- Check your URL bar! There's often no other indication that this is not the real Zenodo or Dataverse!

A tutorial of sorts

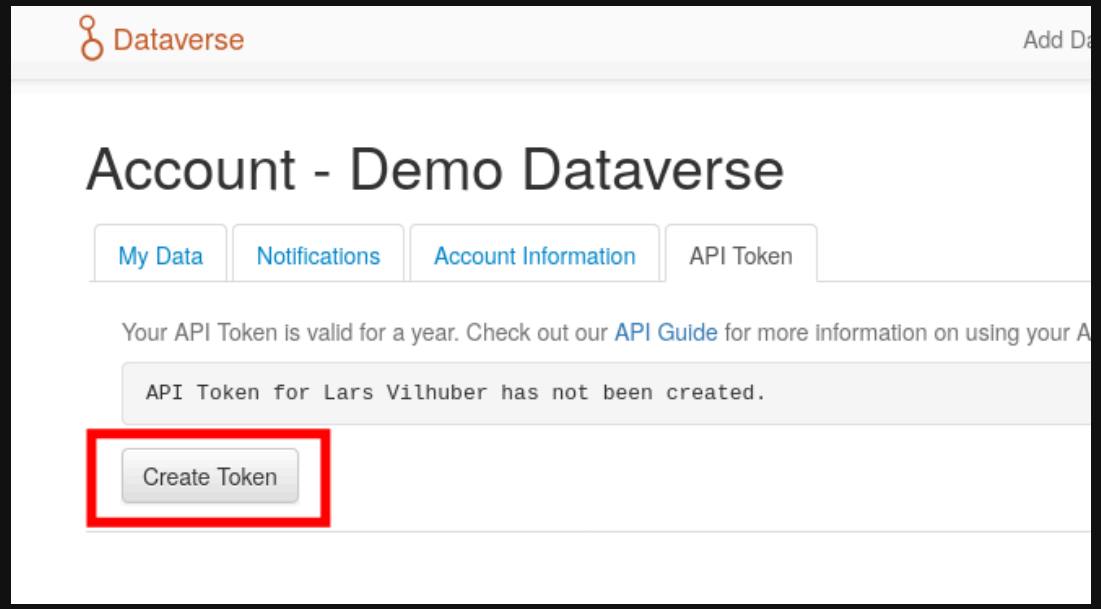
- Demo Dataverse for Lars
  https://demo.dataverse.org/dataverse/larstest

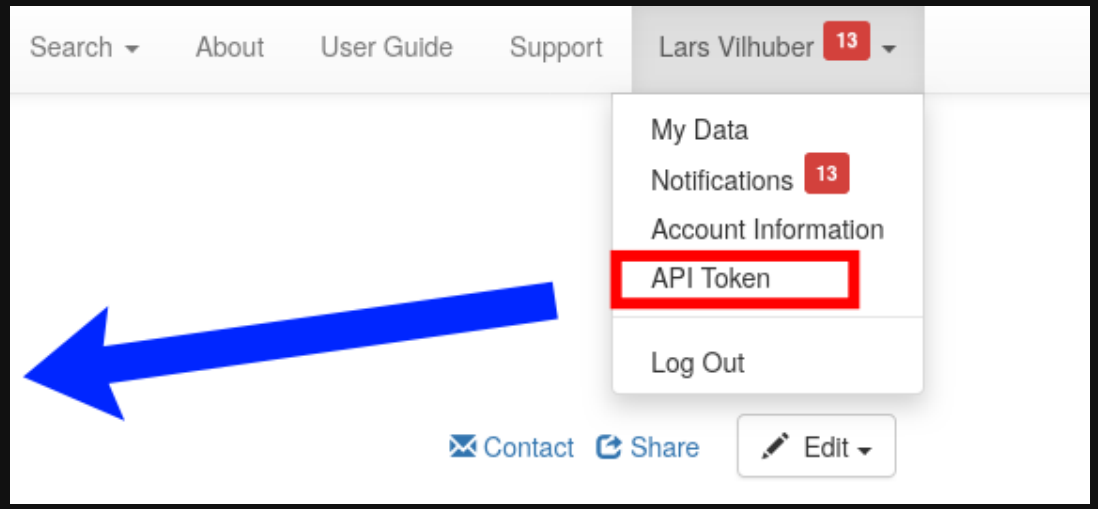
Remember the API tokens?

```
QUALTRICS_API_KEY='something here'
QUALTRICS_BASE_URL='url goes here'
DATAVERSE_TOKEN='token goes here'
DATAVERSE_SERVER='https://demo.dataverse.org'
DATAVERSE_DATASET_DOI='doi goes here'
```

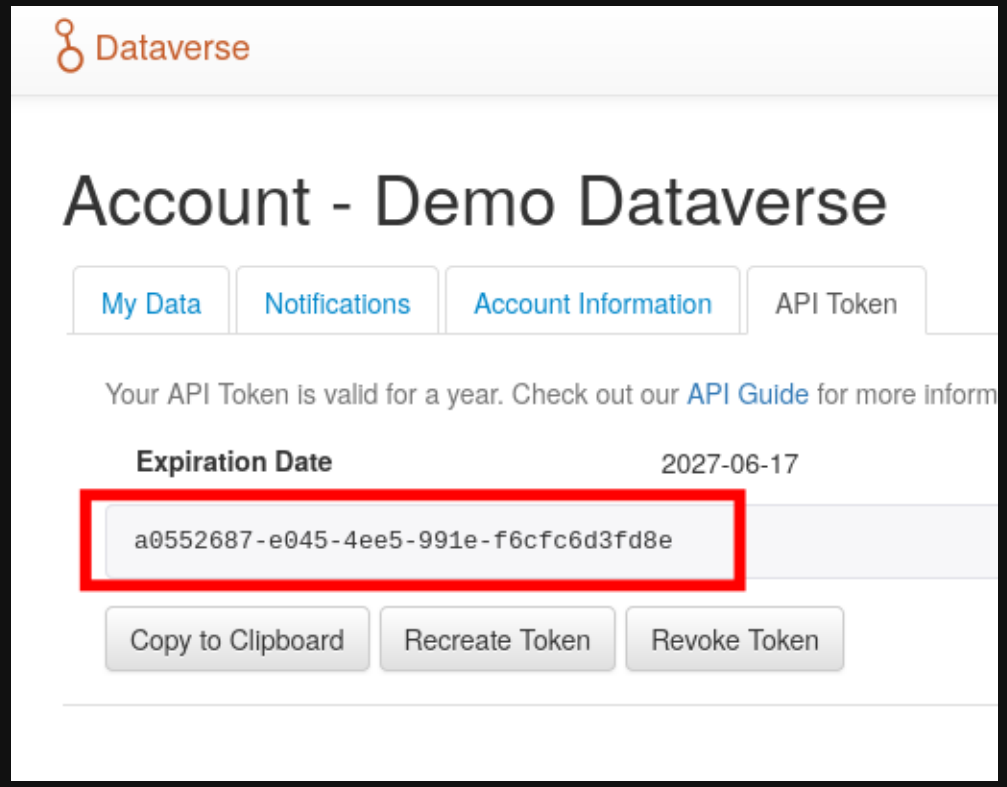
We’re going to need the last three here!

Getting your API keys from Dataverse





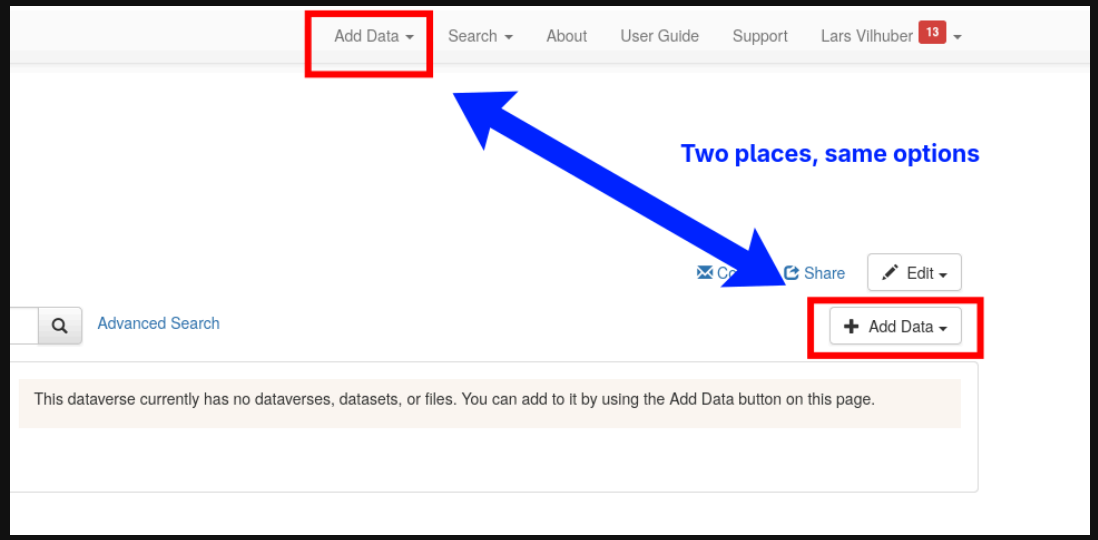
Adding your API key to your .Renviron



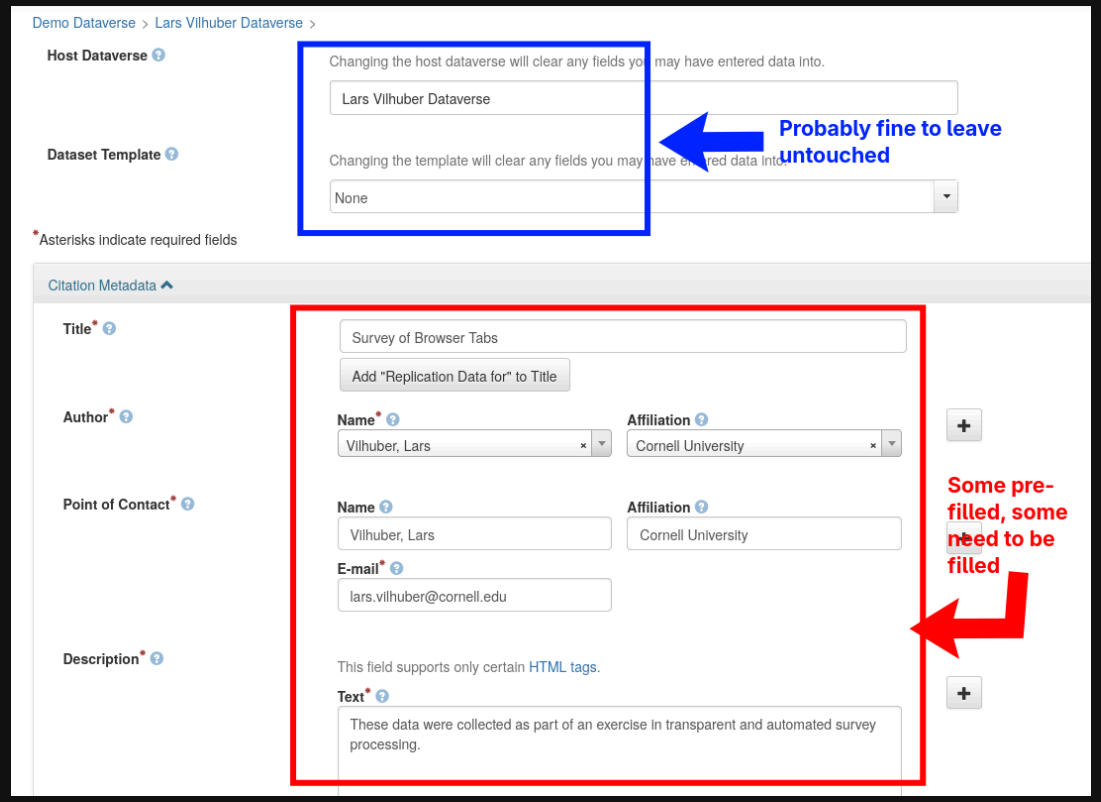

```
QUALTRICS_API_KEY='something here'
QUALTRICS_BASE_URL='url goes here'
DATAVERSE_TOKEN='token goes here'
DATAVERSE_SERVER='https://demo.dataverse
DATAVERSE_DATASET_DOI='doi goes here'
```

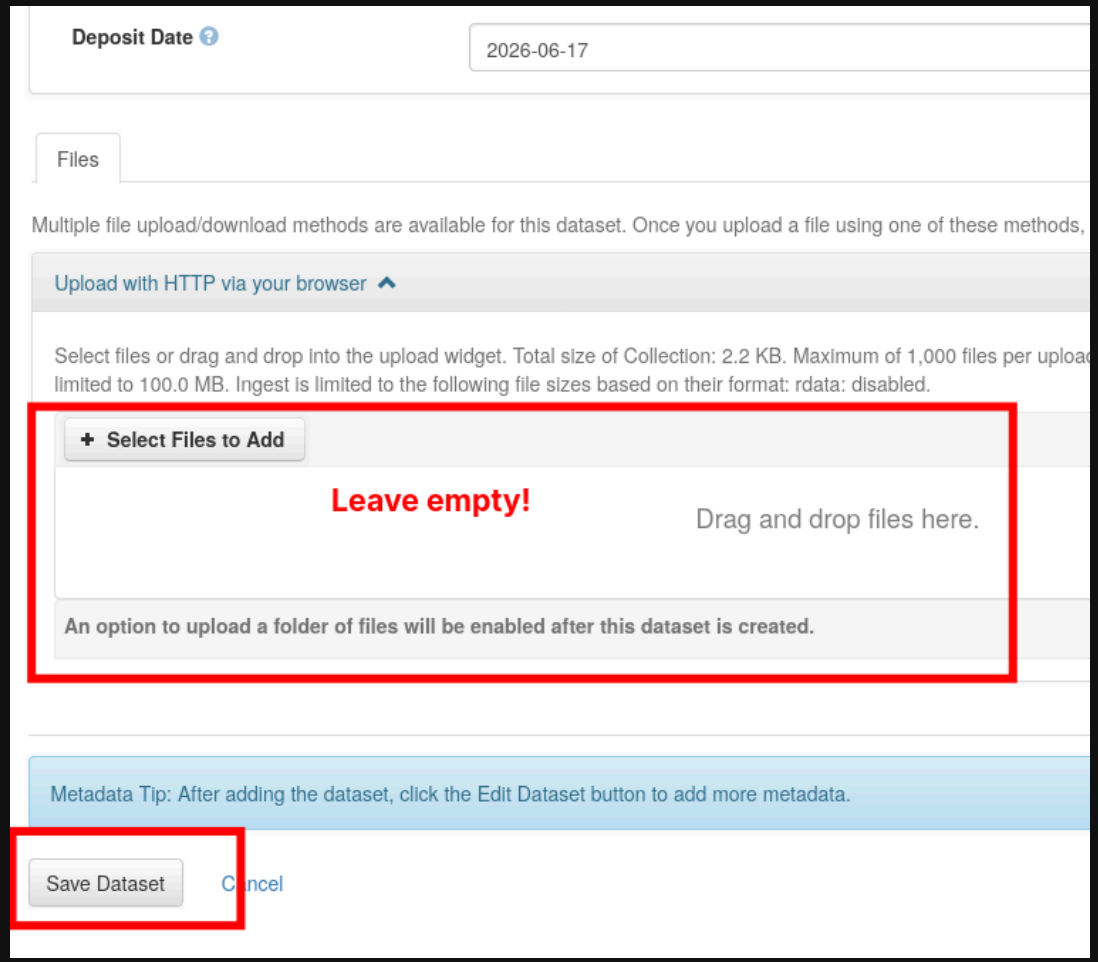
We need a “container”

- Dataverse calls this a “dataset”.
- A “dataset” can hold multiple files.
- While this ***can be created*** via the API, I suggest doing it **manually** (once per project)



Fill in metadata



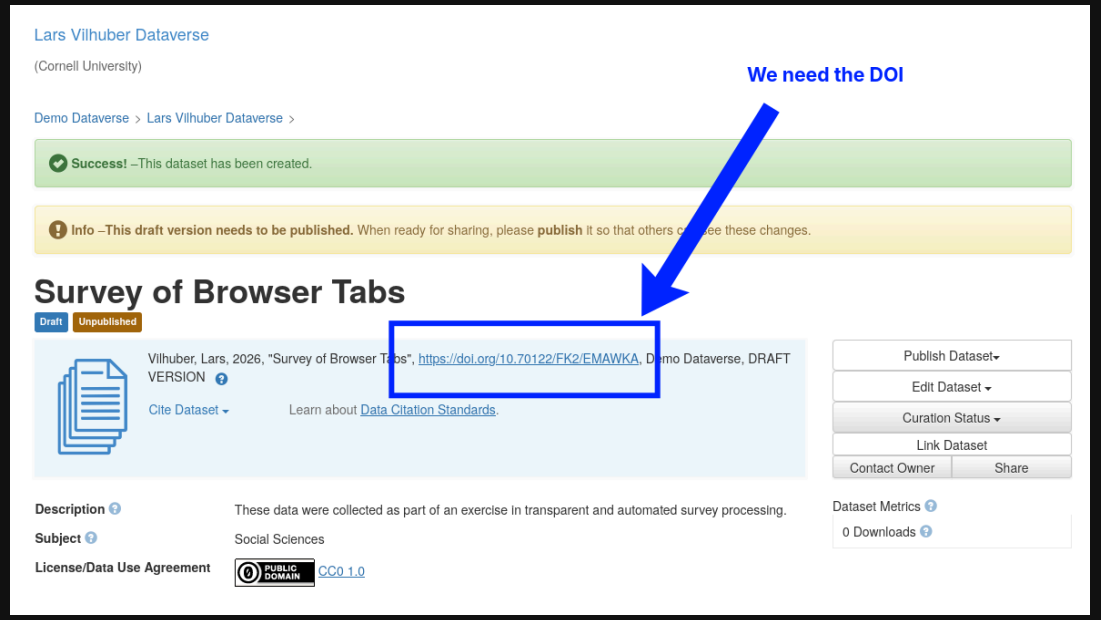


Uploading data to Dataverse

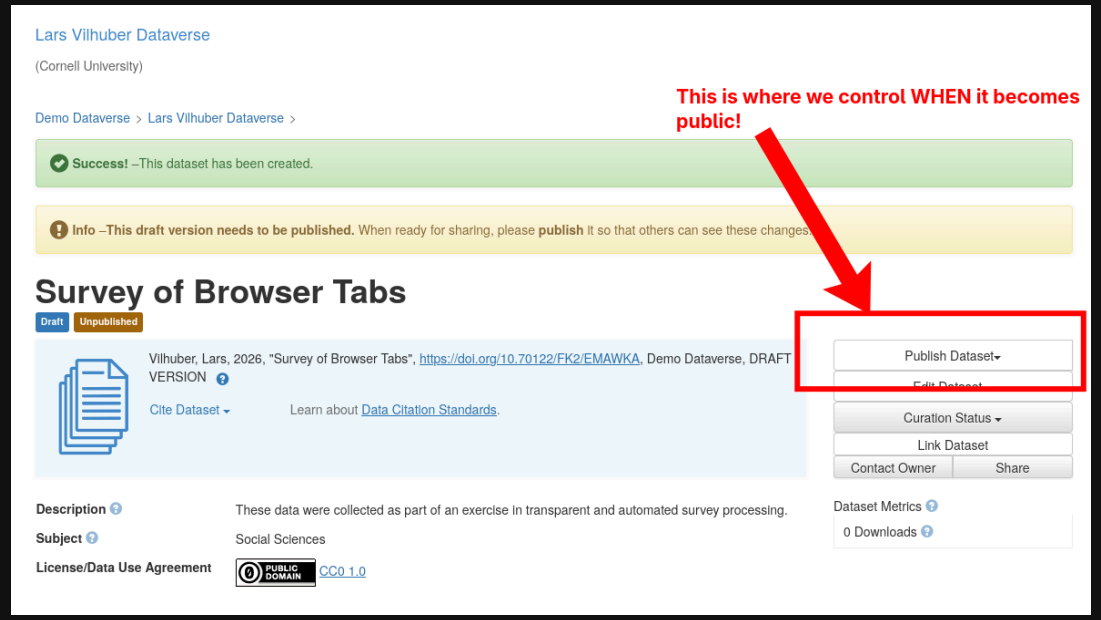
- You ***could*** upload data manually, but this is about automation!
- Now that the “container” is ready, we can upload data to it via the API.

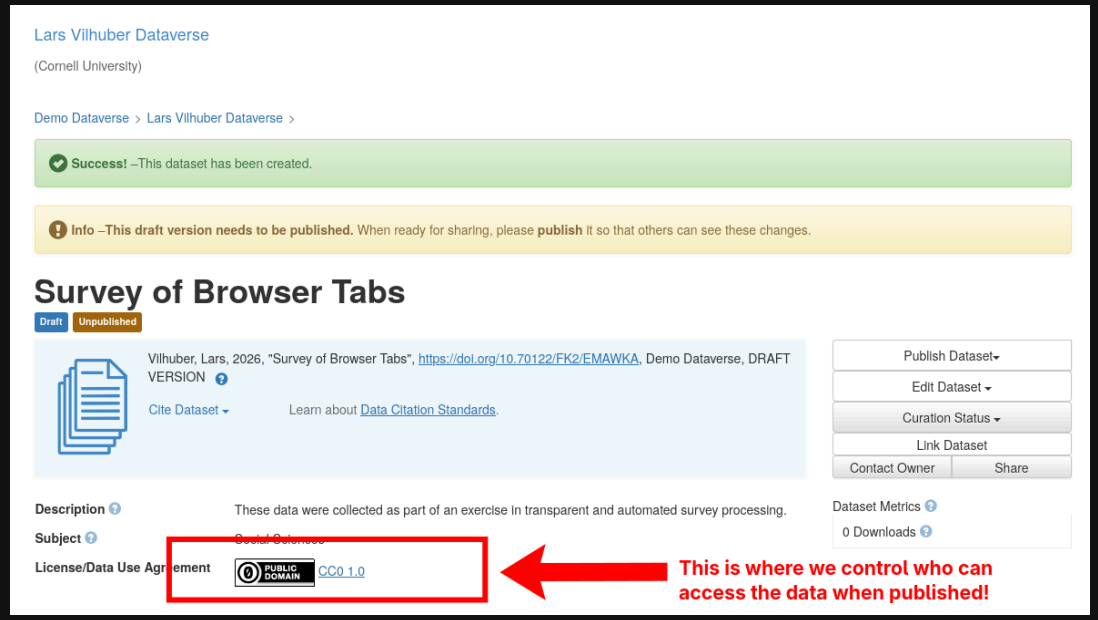
Getting the Identifiers

```
1 QUALTRICS_API_KEY='something here'
2 QUALTRICS_BASE_URL='url goes here'
3 DATAVERSE_TOKEN='token goes here'
4 DATAVERSE_SERVER='https://demo.dataverse
5 DATAVERSE_DATASET_DOI='doi goes here'
```

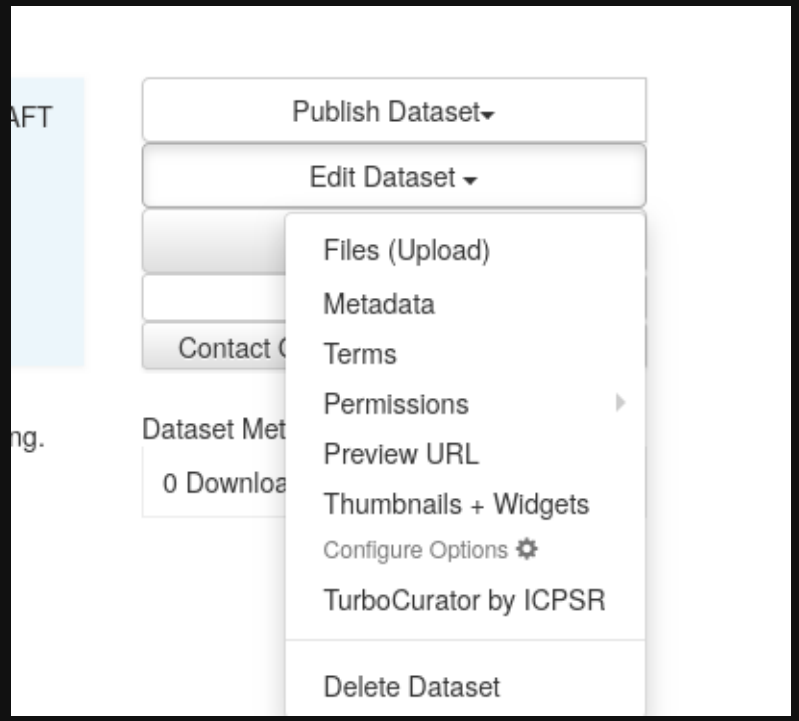


Additional controls



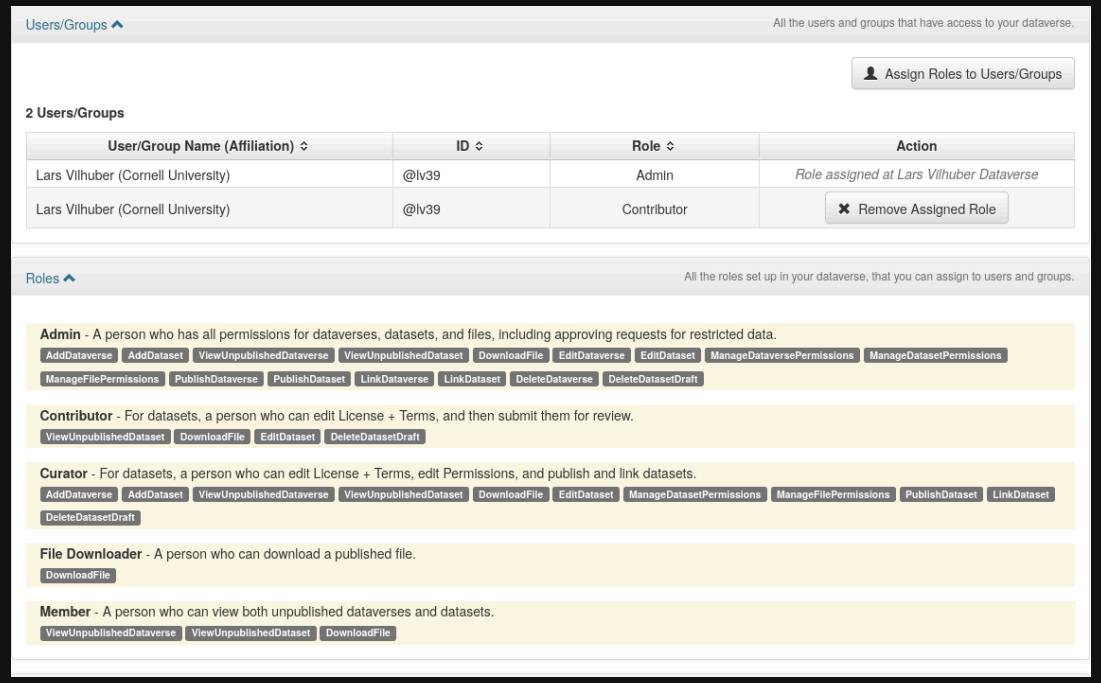


Licensing and permissions



- **Terms**: For people who wish to download published files
- ***Permissions***: For fine-grained access over who can do what before publication

Permissions



You can designate

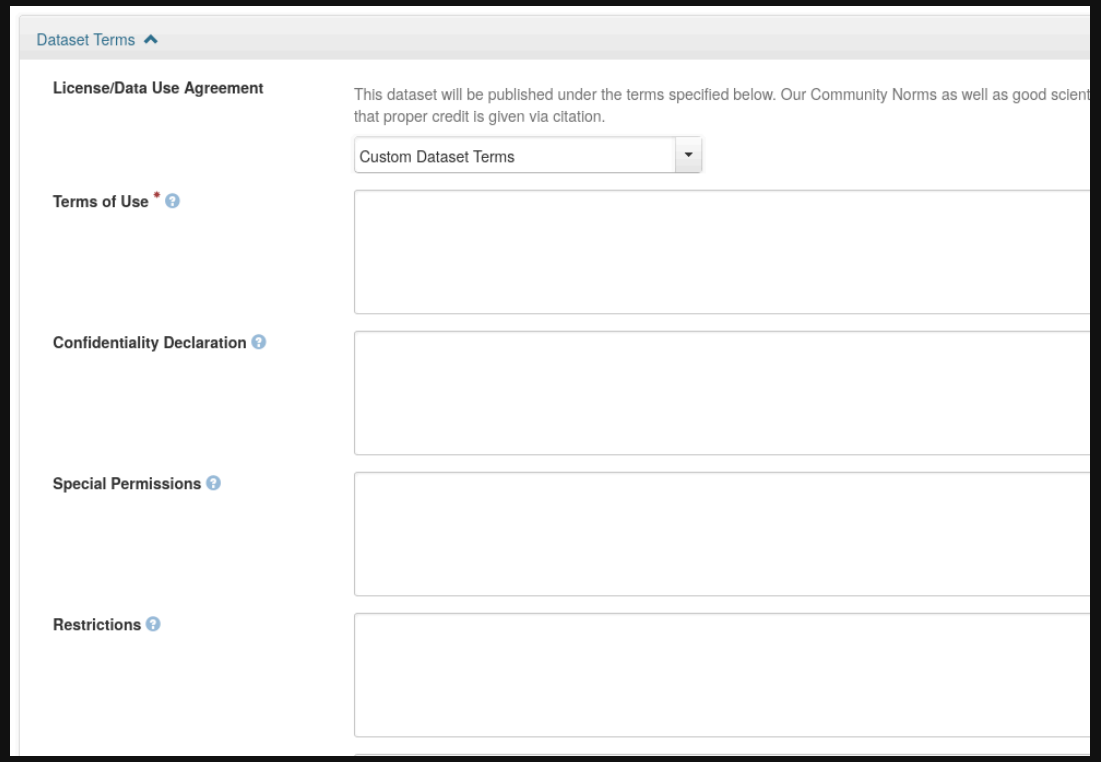
- who can upload
- who can edit metadata
- who can publish

Terms and Licenses

- **Licenses** are broad permissions on how to re-use
- Often CC-BY, see https://creativecommons.org/cc-licenses/
- ***Terms*** are more restrictive. Do third party data users
  - need to contact somebody
  - need to sign a data use agreement

Terms and Licenses

- You can define custom terms (instead of a standard license)
- Strongly suggest talking with University Counsel!



Back to practical matters

Uploading data to Dataverse via API

- From terminal, with ***Python***

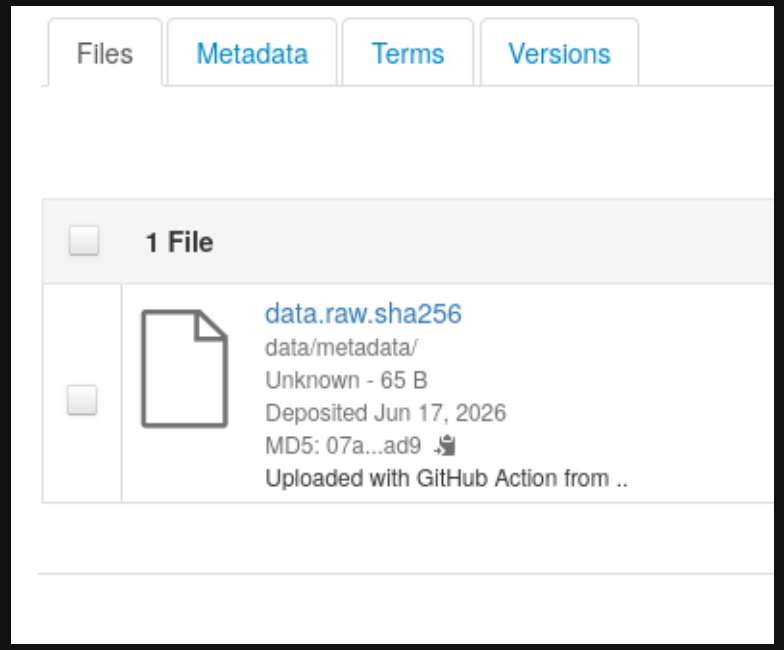
```
python3 -m venv venv-dv
source venv-dv/bin/activate
source .Renviron
git clone https://github.com/larsvilhuber/dataverse-uploader
pip install -r dataverse-uploader/requirements.txt
python3 dataverse-uploader/dataverse.py \
    $DATAVERSE_TOKEN $DATAVERSE_SERVER \
    $DATAVERSE_DATASET_DOI . -d data/metadata
```

Results locally

```
Connecting to Dataverse server: https://
Dataset DOI: doi:10.70122/FK2/EMAWKA
Dataset ID: 2695588
Found 0 existing files in dataset
Deleting 0 existing files...

Paths to upload: ['./data/metadata']
```

```
Scanning path: ./data/metadata
  Directory: ./data/metadata (contains 1
    Uploading [1]: ./data/metadata/data.
      Filename: data.raw.sha256
      Directory label: './data/metadata'
      Response status: 200

Total files uploaded: 1

Done!
```

Results remotely



- Filename is preserved
- Pathname is preserved! `data/metadata`
- MD5 checksum is also present! (less useful for the checksum file!)

Automatically from Github Actions

- Code for uploading automatically

Voilà!

We have a workflow that can automatically download from Qualtrics, and in the same move, upload to Dataverse!

Possible improvements:

- immediately publish (no human intervention)
- upload the download logs and checksums together with the data
- run on a schedule

Putting it all together

Putting it all together

- We already downloaded from the API, and have created checksum for raw data.
- Let’s clean the data, and save local copies
- Then upload the publishable data to Dataverse, and add metadata.

Cleaning the data

```
data.confidential <- data.raw |>
    filter(consent == "Yes") |>
    filter(Status != "Survey Preview") |>
    filter(StartDate > QUALTRICS_STIME & EndDate < QUALTRICS_ETIME) |>
    select(StartDate,EndDate,Status,Finished,RecordedDate,
           ResponseId,consent,age_1,gender,education,
           num_tabs_1,name_confidential,number_confidential)
data.clean <- data.confidential %>%
  select(-name_confidential, -number_confidential)
```

Save files

```
# save files in their locations
data.confidential.file <-
    file.path(confdatapath,"confidential_data.rds")
data.clean.file <-
    file.path(cleandatapath,"clean_data.rds")
saveRDS(data.confidential,
        data.confidential.file)
saveRDS(data.clean,
        data.clean.file)
```

... and create checksums

```
# Calculate checksums for the saved files
confidential_checksum <-
     digest::digest(data.confidential.file,
                    algo = "sha256",
                    file = TRUE)
clean_checksum <-
     digest::digest(data.clean.file,
                    algo = "sha256",
                    file = TRUE)
# Write checksums to files
writeLines(confidential_checksum,
           file.path(metadatapath, "data.confidential.sha256"))
writeLines(clean_checksum,
           file.path(metadatapath, "data.clean.sha256"))
```

Analysis

So here are the results so far (2026-06-18):

gender	Frequency	Percent
Male	5	45.45
Female	5	45.45
NA	1	9.09

By Education

education	Frequency	Percent
Secondary or less	1	9.09
Master’s degree	5	45.45
Professional or doctoral degree	4	36.36
NA	1	9.09

Age

Statistic	Value
Count	10.00
Mean	33.40
Median	28.50
Min	25.00
Max	62.00
Std. Dev.	11.48

Number of tabs open

Statistic	Value
Count	10.00
Mean	15.80
Median	16.00
Min	2.00
Max	27.00
Std. Dev.	9.19

State of the data directory

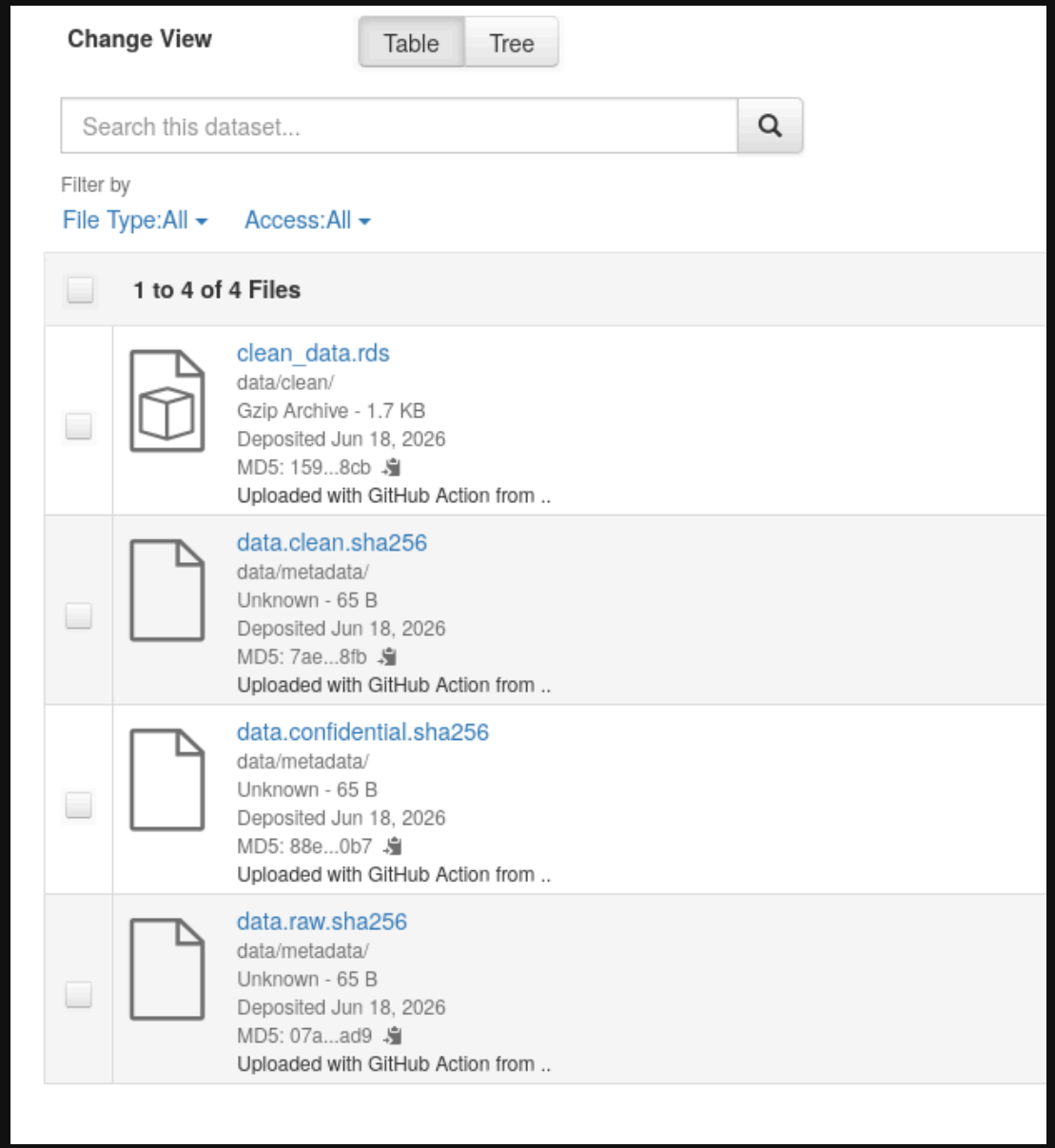
```
1 fs::dir_tree(datapath)
```

```
/home/runner/work/tutorial-preserving-survey/tutorial-preserving-survey/data
├── clean
│   └── clean_data.rds
├── confidential
│   └── confidential_data.rds
├── metadata
│   ├── data.clean.sha256
│   ├── data.confidential.sha256
│   └── data.raw.sha256
├── raw-confidential
│   ├── README.md
│   └── Testing+preservation_June+16,+2026_15.35.csv
├── tutorial-survey.csv
└── tutorial-survey.rds
```

Uploading to Dataverse

```
# System setup
# Need: apt install python3.10-venv
python3 -m venv venv-dv
source venv-dv/bin/activate
git clone https://github.com/larsvilhube
pip install -r dataverse-uploader/requir
```

```
# Do the uploads
python3 dataverse-uploader/dataverse.py
    $DATAVERSE_TOKEN $DATAVERSE_SERVER \
    $DATAVERSE_DATASET_DOI . -d data/meta
python3 dataverse-uploader/dataverse.py
    $DATAVERSE_TOKEN $DATAVERSE_SERVER \
    $DATAVERSE_DATASET_DOI . \
    -d data/clean \
    --remove false
```

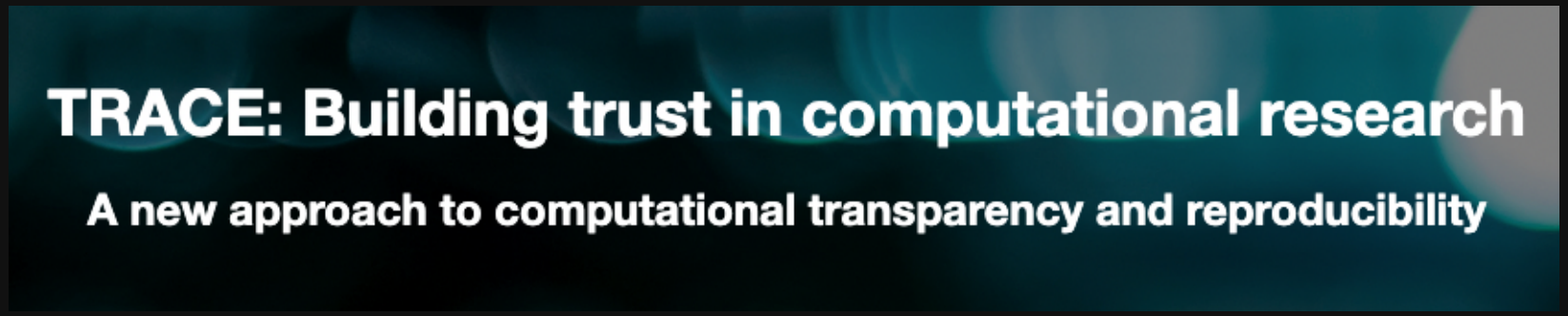

Change View
Table
Tree
Search this dataset...
Filter by
File Type:All
Access:All
1 to 4 of 4 Files
clean_data.rds
data/clean/
Gzip Archive - 1.7 KB
Deposited Jun 18, 2026
MD5: 159...8cb
Uploaded with GitHub Action from ..
data.clean.sha256
data/metadata/
Unknown - 65 B
Deposited Jun 18, 2026
MD5: 7ae...8fb
Uploaded with GitHub Action from ..
data.confidential.sha256
data/metadata/
Unknown - 65 B
Deposited Jun 18, 2026
MD5: 88e...0b7
Uploaded with GitHub Action from ..
data.raw.sha256
data/metadata/
Unknown - 65 B
Deposited Jun 18, 2026
MD5: 07a...ad9
Uploaded with GitHub Action from ..

What is next in this space?

Using 3rd-party trusted systems

A sketch: Transparency Certified

https://transparency-certified.github.io/



Transparency Certified

Work in progress

- Working with cascad, several INEXDA members, World Bank, various RDCs
- Relying on external certification of data inputs (data catalogs with metadata, checksums)

Work in progress

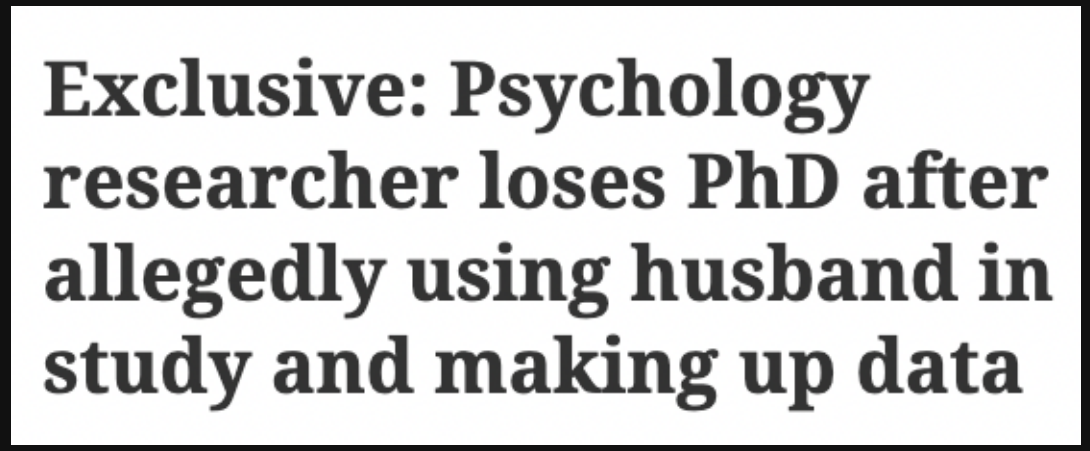
- SIVACOR: Scalable Infrastructure for Validation of Computational Social Science Research[7]

SIVACOR

Does it prevent all fraud?

Does not prevent all fraud

Toronto researcher loses Ph.D.


Exclusive: Psychology researcher loses PhD after allegedly using husband in study and making up data

Toronto case

MIT student makes up firm data


Home | News | Assuring An Accurate Research Record

Assuring an accurate research record

May 16th, 2025

MIT case

Back to Gino

- A transparent, automated pipeline makes it ***much harder*** to manipulate data **after collection, before analysis** — exactly the Gino failure mode.
- But it does **not** prevent fabricating data at the source, or other forms of misconduct.
- Transparency and preservation raise the cost of fraud and the odds of detection — they are not a silver bullet.

The end! Thanks for your attention.

Footnotes

1.

https://datacolada.org/109, https://datacolada.org/110, https://datacolada.org/111, https://datacolada.org/112, https://datacolada.org/114, https://datacolada.org/118

2.

Jones, M. (2024). Introducing Reproducible Research Standards at the World Bank. Harvard Data Science Review, 6(4). https://doi.org/10.1162/99608f92.21328ce3

3. See my tutorial on handling of confidential data and reproducibility

4.

Ginn J, O'Brien J, Silge J (2024). ***qualtRics: Download 'Qualtrics' Survey Data***. R package version 3.2.1, https://github.com/ropensci/qualtRics, https://docs.ropensci.org/qualtRics/.

5.

Add `.Renviron` to your `.gitignore` file to prevent it from being tracked by Git and accidentally pushed to GitHub.