SEPIALOG INSIGHTS · ISSUE 02 · AUGUST 2026

A trustworthy dataset family tree

Why every derived dataset needs a source, an exact version and a research purpose.

THIS MONTH

A note from SepiaLog

Research data rarely moves in a straight line. We clean it, anonymize it, reshape it and create different versions for different analyses or collaborators. That is normal. The risk begins when those relationships survive only in filenames.

This month's SepiaLog Insight looks at dataset branching: how to preserve the source, exact version and research purpose behind every derived file - without treating one file as the mythical universal “final.”

ONE QUESTION TO CARRY FORWARD

What is this file a branch of, and what research purpose makes the branch necessary?

03 · DATA LINEAGE

When One Dataset Becomes Five, Which One Is the Truth?

Research rarely produces one final dataset. It produces a family of related datasets - and the relationships between them are part of the evidence.

SepiaLog Insights · editorial photography

You begin with one source file.

Then you remove duplicate records. You exclude ineligible cases. You create an anonymized version for a collaborator. You reshape the data for one analysis and calculate new variables for another.

Before long, the folder contains:

- `survey_clean.csv`
- `survey_clean_final.csv`
- `survey_clean_final_revised.csv`
- `survey_anonymized.csv`
- `survey_analysis_v2.csv`

Every file had a reason to exist when it was created. A few months later, the names are doing more work than they can handle.

Which file came directly from the source? Does the anonymized version include the later corrections? Which exclusions were used for the paper? Can a collaborator safely update one branch without changing another?

The problem is not that there are too many datasets. The problem is that their relationships have become invisible.

A derived dataset is a research decision

Creating a new dataset is often described as a technical step: copy the file, run the script, save the output.

But every derivation can contain an analytical decision.

Removing rows defines who remains in the study. Recoding a value changes how a response is interpreted. Selecting variables determines what a collaborator can see. Aggregating observations changes the unit of analysis. Anonymizing a file may reduce disclosure risk while also reducing analytical detail.

These are not merely file operations. They shape the evidence and the conclusions that can be drawn from it.

That means a useful record should preserve more than the name of the new file. It should answer:

1. **What was the source?**
2. **Which exact version of the source was used?**
3. **Why was the new dataset created?**
4. **What transformation produced it?**
5. **Which analysis, collaborator or output is it intended for?**

Without those answers, a folder shows what exists but not how the pieces are related.

Copying is not the same as branching

A copy gives you another file. A branch gives the file an identity and a parent.

That distinction matters because research datasets rarely move along a single straight line. One cleaned dataset may lead to a teaching version, a restricted collaborator version, a public-use version and several analysis-specific files. None of these is necessarily the universal “final” dataset. Each has a different purpose.

Thinking in branches replaces the fragile question “Which file is final?” with better questions:

- Final for which analysis?
- Derived from which source state?
- Prepared under which inclusion rules?
- Safe for which audience?
- Used in which publication?

The goal is not to force every project into a complex technical system. It is to acknowledge the structure that already exists in the work.

Keep the original stable

A clear branching practice begins with one rule: do not overwrite the original data.

Treat the source as a stable reference point. When cleaning, anonymizing, reshaping or subsetting changes the material, create a derived dataset and record the relationship.

A simple branch note might read:

Derived `interviews_public_v1` from the corrected transcript set dated 14 May. Removed direct identifiers and generalized workplace locations for external sharing. Full-detail transcripts remain restricted.

Another might say:

Derived `panel_analysis_complete_cases` from cleaned panel version 6. Restricted to participants observed in all four waves for the longitudinal model in Paper 2.

These notes do not need to reproduce the complete method. Their job is to make the branch understandable and point to the transformation that created it.

A branch preserves the relationship between a source dataset and each purposeful derivative.

The family tree is often more useful than the file list

A file list is chronological only by accident. A dataset family tree shows structure.

At the centre is the source. From it, arrows lead to cleaned, anonymized, harmonized or analysis-specific descendants. Each child can have its own later versions without obscuring its origin.

This makes several common problems easier to detect:

- an analysis file was created from an outdated cleaned dataset;
- a public-use file does not include a later correction;
- two collaborators created similar branches for different purposes;
- the dataset cited in a paper cannot be connected to the transformation script;
- a derived file exists, but no one can explain why it was needed.

The visual relationship does not replace documentation. It makes missing documentation visible.

Connect code, data and explanation

For computational work, lineage becomes stronger when the derived dataset is connected to the code that produced it.

Ideally, you should be able to move from a result backwards through three layers:

1. the file used in the analysis;
2. the script and exact code version that produced or transformed it; and
3. the note that explains the research reason for the transformation.

Code can show that 312 rows were removed. The note can explain that they failed the preregistered eligibility criteria. Both are needed. One preserves execution; the other preserves intent.

This is especially important when a project is reviewed, handed to a colleague, revisited after publication or extended into a new study.

How SepiaLog makes branches visible

SepiaLog lets a researcher derive a new dataset from a selected source without changing the original. It records the source file and its exact version, places the new file in the project's derived-data area and creates a note about the branch.

The Dataset family tree then shows how the datasets are related. When analysis code is linked in research notes, the Reproducibility map can also connect files to the relevant script and code revision.

The point is not to create a diagram for its own sake. It is to make a research claim easier to trace.

A question for the next file you create

The next time you save a dataset under a new name, pause before adding `final`, `new` or `v2`.

Ask:

What is this file a branch of, and what research purpose makes this branch necessary?

If that answer stays connected to the file, the dataset can multiply without the project losing its history.

Research does not need one mythical final file. It needs a trustworthy family tree.

Explore dataset branching and research lineage at sepialog.com.

#DataLineage #ReproducibleResearch #ResearchDataManagement #DataCleaning #OpenScience

Research data, documented.

SepiaLog is a private, local-first journal for research data. Connect files, versions, decisions, metadata and delivery - on your own computer.

EXPLORE SEPIALOG.COM

Free · No forced account · No silent upload · Local AI ready

ABOUT THIS ISSUE

The full web edition, accessible image descriptions and related research-data resources are available at sepialog.com/insights. Editorial photography was created for SepiaLog Insights. Any application view shown in the final publication uses a genuine SepiaLog capture.