Fold DB: Compute Without Exposure

Explained Simply

Tom Tang

March 4, 2026

1 What's the Problem?

Imagine you have a diary. You want your doctor to read your health pages, but not your personal secrets. You want a scientist to see your test results, but not your name. And you definitely don't want strangers reading any of it.

Most databases work like a filing cabinet with one lock. Either someone has the key and sees everything, or they don't and see nothing. There's no way to say "you can see page 3 but not page 7" or "you can see page 7, but only a summary."

Fold DB solves this.

2 The Big Idea: Folds

A **fold** is like a magic window you put in front of your data. Different windows show different things.

- Window 1 might show everything—your name, your diagnosis, your test results.
- Window 2 might hide your name (replace it with a random code) but still show your test results.
- Window 3 might only show a single number—like a health score—computed from your data.

You choose who gets to look through which window. The actual data never leaves the room. People only see what the window lets through.

3 An Example

You're a patient at a hospital. You have three pieces of information stored:

1. Your name
2. Your diagnosis
3. Your lab results

You create three windows (folds):

Window 1: "Doctor View" — Shows everything. Only people you really trust can look *and write* through it (you and your doctor). Your doctor can update your diagnosis or add new lab results. Every change is signed with the writer's digital signature, and the old version is kept forever—nothing is ever deleted.

Window 2: "Researcher View" — Shows your diagnosis and lab results, but scrambles your name into a code that can't be unscrambled. Researchers further from your inner circle can use this window.

Window 3: "Score View" — Looks through Window 2 first, then calculates a health score from your diagnosis and lab results. Anyone with permission to both windows sees just one number.

Now when people come asking:

- **Your doctor** looks through Window 1 and sees everything. Fine—you trust her.
- **A researcher** tries Window 1 and sees nothing (not trusted enough). Tries Window 2 and sees the scrambled name plus diagnosis and lab results. Can't figure out who you are.
- **A stranger** tries any window and sees nothing. They don't even learn what windows exist.

4 Who Gets to Look—and Write? Three Checks

Windows control both reading *and* writing. It's not just the owner who can change data—anyone with a close enough trust distance (and the right permissions) can write too. But every write must be signed, and the old version is always kept.

Every time someone tries to read or write through a window, three things are checked. *All three* must pass.

4.1 Check 1: How Close Are You?

You give people a number called their **trust distance**.

- 0 = you (the owner)
- 1 = people you really trust (your doctor)
- 2 = one step further out
- 5 = distant acquaintances
- 10 = basically a stranger

Each window has a rule like "readable if trust distance is 1 or less." If your number is too high, you see nothing.

Trust **adds up automatically**. If you trust Alice (distance 1) and Alice trusts Bob (distance 1), then Bob is distance 2 to you—without you having to do anything. Alice's friend of a friend would be distance 3, and so on. This means you don't have to assign a number to every single person. Trust flows through the chain.

But you can always override. If you don't like the number the chain gives someone, you can assign them a different one directly. For example, even though Bob would be distance 2 through Alice, you could set him to distance 5 (less trusted) or distance 1 (more trusted) yourself.

4.2 Check 2: Do You Have the Right Key?

Some windows require a special digital key to look through. The key might be limited: "you can look 10 times, then it stops working." This is for extra-sensitive data where trust distance alone isn't enough.

If a window requires both a trust distance check *and* a key, you need to pass both. One doesn't excuse the other.

4.3 Check 3: Have You Paid?

Some windows cost money to look through. People you trust more pay less. Strangers pay more (if they're even allowed at all). If you haven't paid, you see nothing.

5 Transforms: Changing What You See

Windows don't just show or hide data—they can **transform** it.

- **One-way transforms** (like scrambling a name into a code): you can read the result, but you can never get the original back. This is how the "Researcher View" hides your name.
- **Two-way transforms** (like converting dollars to euros): you can read *and* write. If you write in euros, it converts back to dollars in the original data.

6 Windows Can Stack

A window can look through *another* window first, then do something extra. Like Window 3 in the example: it looks through Window 2 (which already scrambles your name), then computes a health score.

When windows stack:

- Each window checks its own rules independently.
- If *any* window in the chain says no, the whole thing returns nothing.
- You never get a partial result. It's all or nothing.

7 Security Labels

Every piece of data has a sensitivity level, like:

$$\text{Public} \rightarrow \text{Internal} \rightarrow \text{Confidential} \rightarrow \text{Secret}$$

A transform can move data to an equal or higher level, but never lower. You can't take Secret data, run it through a transform, and call the result Public. The system won't allow it.

8 Everything Is Logged

Every version of every piece of data is kept forever. If someone changes a value, the old value is still there. Nothing is ever deleted or overwritten. You can always go back and see what the data looked like at any point in time, and who changed it.

9 Everything Is Encrypted

Even the data sitting on disk is encrypted. Each dataset has its own key. If someone steals the hard drive, they get encrypted gibberish.

10 Universal Fold Service

So far, each person creates their own windows for their own data. The **Universal Fold Service** lets someone publish a *template*—a standard set of window shapes that anyone can adopt.

How it works:

1. Someone publishes a template (e.g., "health record" with fields for age, BMI, blood pressure).
2. Thousands of people adopt the template, each connecting it to their own data.
3. A program written for that template works on *everyone's* data—even though everyone's actual numbers are different.

Example: A health agency publishes a "health record" template. Patients everywhere adopt it. The agency writes a risk calculator. The system runs it across all patients. Each patient's window independently decides whether the agency is allowed in. Some say yes, some say no. The agency gets a set of risk scores back—without seeing anyone's raw data, and without knowing who said no.

The same program, the same template, thousands of different people's data. Write once, run everywhere.

11 Universal Transform Registry

Transforms—the operations that change what a window shows—live inside folds. The **Universal Transform Registry** is a shared library of pre-approved transforms that any fold can use.

Why this helps:

- Without it, every person who wants to scramble a name writes their own scrambling code. Some do it well, some do it badly. No one can compare results.
- With it, there's *one* official "scramble a name" transform. It's tested, audited, and everyone who uses it gets the exact same behavior.

How it works:

1. A developer writes a transform and submits it.
2. The registry checks it: does it do the same thing every time? If it claims to be reversible, does the reverse actually work? Does it respect security levels?
3. If it passes, it gets a permanent ID (a fingerprint of the code). That ID never changes.
4. Any fold, anywhere, can use the transform by referencing its ID.

Example: A medical standards group submits a transform that replaces exact ages with 5-year ranges (25 becomes "25–29"). It's marked as one-way (can't get the exact age back). Every hospital, every researcher, every insurance company that needs age anonymization uses the same transform, referenced by the same ID. Auditors can verify privacy compliance just by checking which transform IDs a fold uses.

12 The Core Guarantee

Fold DB makes one promise:

A query never returns more than what the caller is allowed to see.

Every window checks who you are, what keys you hold, and whether you've paid. Every transform either hides information (one-way) or preserves it exactly (two-way). Stacking windows can only *reduce* what you see, never expand it. And if anything goes wrong, you get nothing.

13 Summary

- Your data lives behind **windows** (folds). No one ever touches the data directly.
- Different windows show different things to different people.
- Three checks gate every access: trust distance, cryptographic key, and payment.
- Transforms can hide, summarize, or convert data before it reaches the caller.
- Windows can stack—each one enforces its own rules independently.
- If any check fails, the result is **nothing**. No partial data. No hints.
- The **Universal Fold Service** lets one template work across many people's data.
- The **Universal Transform Registry** ensures everyone uses the same vetted transforms.
- Everything is logged, everything is encrypted, and you never get more than you're allowed to see.