RJL SOLUTIONS

PERSPECTIVE AND PRACTITIONER GUIDE / DATA, AI AND ENGINEERING

Data Is Not Just Code

Why reliable AI answers require context, evidence and disciplined delivery.

CONTEXT · AUTHORITY · EVIDENCE

Richard White
RJL Solutions · rjlsol.com

VERSION 3 · PUBLISHED EDITION
OCTOBER 2026

01 / THE EXECUTIVE CASE

A precise number can answer the wrong question.

PART ONE · FOR SENIOR LEADERS

> Ask an AI assistant how many claims were open at quarter-end. It can find a status column, write valid SQL and return a precise number. That number can still answer the wrong question. Does open include reopened claims? Does quarter-end mean the state of the business then, or the information available when the report was filed? Which acquired system's codes apply?

Better models can discover these questions, retrieve relevant policies and calculate the alternatives. Reliable answers also need the organization to make the applicable meaning available, and to establish which sources and decisions are authoritative. Connecting a model to data gives it access. Access alone doesn't give it that context.

This isn't a claim about model intelligence. Take two organizations with identical tables and different approved definitions. A system given only the tables has no evidence of which policy applies. A capable analyst would need the same context.

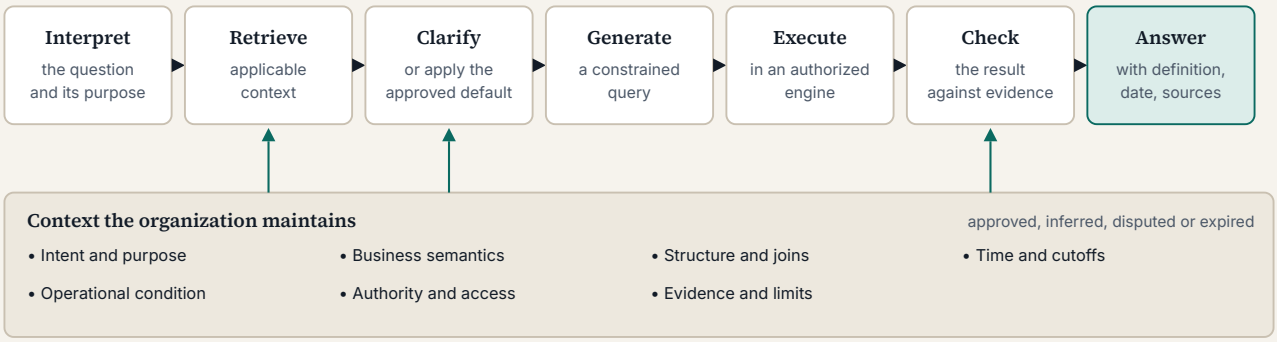


Exhibit: Connecting a model to data provides access. Context, authority and checks turn access into a reliable answer.

THE THESIS An LLM can infer possible meanings and generate working queries. Reliable business answers need enough context, explicit authority, controlled execution and evidence that the result is fit for its use. Software delivery practices supply much of the machinery for keeping those conditions in place. The organization stays accountable for the decisions.

02 / SHARED DISCIPLINES

Shared engineering disciplines, additional evidence obligations.

Experienced software teams will recognize these problems. Applications carry state too. They encode business meaning and produce consequences no rollback can erase. Data products concentrate these challenges: records outlive individual applications, definitions span business functions and results get copied into many downstream decisions. So the lesson is to apply software discipline to the whole data product, its meaning and evidence included.

SHARED PRACTICE	DATA-PRODUCT CONCERN	EVIDENCE REQUIRED
Domain modeling and API contracts	Definitions span functions and change with policy	Named definitions with owners, scope and versions
Database migrations and expand, migrate, contract	Years of history move with every change to meaning	Bounded backfill, retained prior outputs, reconciled differences
Consumer-driven tests	Copies and extracts create consumers no one registered	Lineage and access logs that find consumers before a change
Progressive delivery	Staging contains new reads; it cannot recall a filed report	Versioned interfaces and an agreed cutover per consumer
Observability	Plausible but wrong numbers pass every uptime check	Freshness, distribution and reconciliation monitors with owners
Incident response	A defect may already sit in decisions people made	A recovery plan that includes exposed consumers and corrections

Software already has answers to most of these problems, and they apply here: parallel change for interfaces and schemas, evolutionary database design, contract tests, staged rollout and incident response [6][7]. The data product adds evidence that its meaning, history and consumers get the same care as its code.

Google Cloud DORA (DevOps Research and Assessment) 2025 research describes AI as an amplifier of the strengths and weaknesses an organization already has. It associates better outcomes with practices such as healthy data ecosystems, AI-accessible internal data, strong version control and small batches [1]. These are survey-based associations, not guarantees. They still point the same way. AI accelerates whatever system you put it in.

03 / CONTEXT

Context is an operational asset, not a longer prompt.

At answer time, an AI system needs more than a glossary. It needs the context that settles what the question means for this purpose, now, under this authority.

CONTEXT CATEGORY	QUESTIONS IT MUST RESOLVE
User intent and purpose	Who is asking, what decision follows, and which interpretation fits that purpose?
Business semantics	What is the metric: population, grain, inclusions, exclusions and unit?
Structure	Which sources and keys apply; what are the join cardinalities, deduplication and aggregation rules?
Time	Which event date, effective date, reporting cutoff, timezone and historical definition apply?
Operational condition	Is the feed fresh and complete; are there known incidents, missing periods or exceptions?
Authority and access	Which definition and source are approved for this purpose; which data and actions are authorized?
Evidence and limitations	Which approved examples, reconciliations, provenance and known limits support the answer?

Treat context like other production assets: versioned, owned, discoverable, testable and scoped to the question. Mark each item as approved, inferred, disputed or expired. An agent may discover a likely join or definition. Promoting it to an authoritative default follows the product's agreed policy.

None of this means curating everything by hand before the first useful query. Start with the relevant sources and questions. Reuse existing code and documentation, let AI propose missing context and escalate material ambiguity. Industry tools already work this way. Snowflake's Cortex Analyst, for example, uses semantic models and a repository of verified question and SQL pairs to improve answers; that's an implementation pattern, not a guarantee of accuracy [8]. Anthropic describes context as a finite resource with diminishing returns, which argues for relevant, bounded context over everything in the prompt [9].

04 / THREE USES OF AI

Build, answer and act are different risks.

Discussions of AI in data work often slide between three uses without separating them. Each needs different controls.

USE	WHAT AI DOES	WHAT GOVERNS IT
Build-time assistance	Proposes pipeline code, tests, mappings and documentation	Review and CI/CD, as for any change to persistent logic
Runtime analytics	Interprets each question, selects context and generates or invokes a query	Runtime checks and answer-level evidence; release tests cannot enumerate every future question
Action-taking agents	Uses an answer to change a price, approve a payment or modify a record	Separately bounded action permissions and controls matched to the consequence

Keep computation and authority in the engine

Use the database, or another computation engine, for exact aggregations. The model plans, generates and explains the computation. Enforce access controls and query limits in the execution layer. Retrieved records and documents must never gain authority to override tool permissions or policy just because they appear in context.

A correct aggregate answers what happened. Why it happened and what to do next need their own assumptions and validation. Context is necessary for many reliable answers. It isn't enough for every analytical claim.

Calibrate the expected gain locally

Evidence on AI productivity is still unsettled. METR's 2025 randomized trial found experienced developers in large, familiar codebases took 19 percent longer with AI tools while believing they were faster [2]. Its February 2026 follow-up called its new data an unreliable signal because many developers declined to work without AI. It said current speedups may be larger, and redesigned the study [3]. On benchmarks, the best system in the 2024 Spider 2.0 paper solved its original enterprise text-to-SQL setting 21.3 percent of the time [4]. The current leaderboard reports far higher scores on its Snow and Lite settings; those are different configurations, so it isn't a like-for-like improvement [5]. Here's the lasting point. Benchmarks specify the task and supply the context. They don't show that an agent can choose an organization's unstated policy.

IMPLICATION Measure accepted results end to end, in your own setting. Expand AI's autonomy where that evidence supports it.

05 / LEADERSHIP DECISIONS

Fund the full path from question to trusted answer.

Gartner predicts that through 2026, organizations will abandon 60 percent of AI projects unsupported by AI-ready data. In its July 2024 survey of 1,203 data management leaders, 63 percent said they lack, or aren't sure they have, the right data management practices for AI [10]. That's a prediction and a self-assessment, not an observed failure rate. It does match what practitioners report.

Start or strengthen

- **One valuable question first.** Pick a consequential question, name its owner and consumers, and make its definitions and examples explicit.
- **Owners who can decide.** A business owner for each material definition, with a route to settle disputes.
- **Context as funded work.** Discovery, profiling and context curation, estimated after profiling rather than before.
- **Evidence matched to consequence.** Tiers: exploratory read-only work, routine approved queries, and material changes to consequential decisions.
- **Review capacity.** More AI-written changes need qualified review; plan for it.

Stop

- Treating an AI coding or query tool as the data strategy.
- Measuring data teams only by pipelines shipped; measure accepted answers, their quality and total cost.
- Requiring the same ceremony for every change: a small read-only change does not need two month-ends of parallel running.

> FROM THE FIELD · COMPOSITE
>
> A technology executive funded AI assistants for a regulatory reporting rebuild and planned on halving the schedule. The pipelines got written quickly. The program still ran long. Finance and risk disagreed about two definitions, and the new figures had to be explained against filed numbers before anyone would sign.

DECISION Before approving a data program, ask three questions. When will the definitions be agreed? How will the new figures be shown fit for use? Which consumers will change, and when? A plan without those answers is a build plan. It isn't a delivery plan.

06 / DISCOVER AND PROFILE

Find out what the data actually holds.

PART TWO · FOR ENGINEERING AND DATA LEADS

Estimates made before profiling can miss the exceptions that drive effort. Before designing anything, find out what exists and what it holds.

ACTIVITY	EVIDENCE TO PRODUCE	AI'S ROLE
Inventory sources	Systems, tables, files and owners in scope, with access confirmed	Scan catalogs, query logs and code for candidates
Trace current lineage	How today's reports are produced, including spreadsheets and manual steps	Read SQL, jobs and scripts; draft lineage for review
Profile the data	Counts, keys, nulls, ranges, duplicates and late arrivals by period	Run and summarize profiles; flag outliers
Find the exceptions	Reopened cases, reversals, legacy codes, migrated records	Cluster unusual patterns for a person to explain
Classify sensitivity	Personal, confidential and regulated fields, before anything is copied	Propose classifications; a person confirms

Profile production data, or an approved, protected copy. The exceptions are the work: a reopen code used differently in one region, a field repurposed after an acquisition, a system that stores local time. AI finds these patterns fast. The business decides which are correct.

Never give an AI tool broader access than the person using it. Classify and mask sensitive fields before profiling, and keep profiling outputs under the same controls as the data.

EXIT TEST The team can show, from the data, the volumes, defects and exceptions that will drive the design, and an estimate that reflects them.

07 / AGREE MEANING

Name each definition, its owner and its purpose.

Agreeing meaning is the decisive step, and executable examples help expose the choices. Different functions can legitimately need different measures. You aren't after one canonical definition for everything. You want explicit, named definitions with owners, scopes and defaults, reconciled instead of suppressed.

- Name the accountable owner for each material definition, and who must be consulted.
- Write it with grain, period, inclusions, exclusions and known limitations.
- Test it against known cases the business agrees on, including awkward ones.
- Where purposes differ, keep separate named measures, such as gross and net claim cost, and state which is the default for which purpose.
- Version the definition; treat a change as a product change with consumer impact.

AI helps by drafting definitions from policies, code and usage, and by showing where existing reports already disagree. It can shorten the evidence-gathering. The decision still needs someone with authority. A model that silently picks one definition has made a business decision nobody approved.

LAYER	EXAMPLE	OWNER
Business meaning	What counts as an open claim, and on which date	Business owner and steward
Executable logic	The SQL or semantic model that implements it	Data engineering, reviewed against the definition
Evidence	Examples, reconciliations and approval records	Product owner, retained for audit

EXIT TEST Every material term in scope has an owner, a written definition, agreed test cases and a recorded decision on each known conflict.

08 / THE RELEASE PIPELINE

Use one delivery platform; match the gates to the consequences.

Data and AI changes should flow through the same delivery platform as application changes. What differs is the evidence each gate collects.

STAGE	SHARED SOFTWARE PRACTICE	ADDITIONAL EVIDENCE FOR THIS DATA OR AI PRODUCT
Propose	Issue, specification, pull request	Purpose, changed definition, affected periods, sources, consumers, owner, risk tier
Build	Versioned artifacts, reproducible builds	Transformations, schemas and contracts, semantic definitions, prompts and tool configuration where relevant, migration plan
Verify	Unit, integration, security, performance	Grain and join checks; temporal cases; freshness and quality; business examples; segment reconciliation
Evaluate AI	Regression tests of changed behavior	Held-out questions, ambiguity handling, permission boundaries, evidence quality, unsupported questions, latency and cost
Stage	Isolated environment, compatibility, progressive release	Protected representative records, replay or bounded backfill, restart behavior, consumer tests, shadow comparison where warranted
Authorize	Risk-based release policy	Recorded approval for material definition changes; accepted explanation of material differences
Release	Promotion, routing, monitoring	Versioned views or interfaces, staged consumer cutover, preserved prior outputs where needed, retirement criteria
Recover	Rollback or roll-forward	Restore or recompute affected state, identify exposed consumers, correct outputs, communicate consequences

Version or record the code, definition, schema, source snapshot or offsets where feasible, the relevant context, the model version when available, configuration and evaluation results. Deterministic transformations should be reproducible. Model outputs may not be, so keep the actual query, result references and decision evidence under the retention rules.

Test end to end whenever the model, prompt, retrieval corpus, semantic definition or tool configuration changes. The SQL can stay identical while an AI interface starts picking a different metric.

Test at several levels, not only the total.

Software delivery practices provide the foundation. Data-specific evidence stays part of the build and test process. AI drafts transformations, tests and documentation; engineers review them.

Correctness needs more than one reconciliation

Reconciliation helps. It can't be the only definition of correct. Two segment errors can cancel out in the total. An old report can be wrong. A new output and its supposedly independent reference can share an upstream defect. Check at several levels: business examples, record identity, cohort counts, joins, segment totals, time periods, independent control totals and selected manual traces. Explain differences you expect because a definition changed on purpose.

A ledger works for reconciling compatible accounting amounts. A claim cost measure may need a bridge for scope, timing and gross or net treatment. An average needs its numerator and denominator checked separately. Matching a ledger total doesn't validate a claim cohort or a closure definition.

Make AI-written code and tests safe to accept

- Give the assistant the approved definitions and contracts, so it implements decisions rather than inventing them.
- AI-generated tests can repeat the assumptions baked into AI-generated code. Set expected outcomes for critical cases independently.
- A model reviewing its own output adds scrutiny. It isn't independent ground truth.
- Review generated SQL for joins, filters and grain. The common errors produce plausible numbers, not failures.
- Keep the assistant's access within the engineer's own permissions.

EXIT TEST Critical cases have independently established expected results, and every change passes data, contract and multi-level checks in the pipeline.

Plan for history and consumers, not only code.

When a release changes stored history or existing consumers, include a migration and acceptance plan in the deployment.

STEP	WHAT GOOD LOOKS LIKE
Version the interface	Breaking changes publish a new version; the old one stays available through an announced window
Backfill deliberately	Recompute history in bounded batches with checkpoints; record which version produced each period
Compare	Parallel runs, historical replay or shadow comparison, chosen for the risk they expose
Explain differences	Each difference is a defect to fix or an approved change to explain to consumers
Accept	The owner accepts material changes to consequential figures before cutover
Cut over and retire	Route consumers on agreed dates; retire unmanaged duplicates, keep intentionally different named measures

Keep valid time apart from recorded time. A backfill may correct when something was true without changing what was known when a report was filed. Keep both, and restatements become explainable.

Staged rollout contains what consumers read next. It can’t undo a filing, a price or a payment already made on the old or new numbers. That’s why acceptance sits before cutover for consequential data.

> FROM THE FIELD · COMPOSITE
>
> A platform migration rewrote several hundred pipelines and passed its tests. In the first week after cutover, a quarterly planning model failed. It read an extract from the old warehouse that no inventory had recorded. The fix took a day. Finding it cost a planning deadline. Consumer discovery now runs through lineage and access logs before any cutover.

Watch for plausible but wrong, and plan for people.

Plausible but wrong outputs are an important failure mode in both software and data, and neither uptime nor execution success detects them. A late feed, a changed source code or a duplicating join leaves the numbers believable.

SIGNAL	WHAT TO MONITOR	RESPONSE OWNER
Freshness and volume	Arrival, completeness and counts against the contract and recent history	Product owner, with the source owner
Schema and codes	Upstream changes to columns, types and code values	Source owner and product owner
Quality and reconciliation	Contracted tests and control totals each cycle	Product owner; business owner for breaks
AI answers	Clarification and refusal rates, evidence quality, sampled correctness	AI product owner

Change lands on people

A new definition reaches people who built processes and explanations around the old one. Name a change lead for each affected function. Explain material differences before people see them. Give consumers early access to test their own work. Retire unmanaged duplicates on an agreed date, while keeping intentionally different measures clearly named. And measure adoption, including old outputs still running. Adoption by Design covers the organizational patterns in more depth.

Claim cost to close, end to end.

An insurer wants a trusted figure for claim cost to close, defined here as cumulative paid cost for claims closed in the period. Two acquisitions left it with three claims systems. This example is illustrative. The figures are invented to show the shape of the work.

CLAIM	PAID	RECOVERIES	STATUS
A	$100	$20	Closed
B	$200	$0	Closed
C	$300	$100	Reopened

Gross paid cost for currently closed claims is $300; net of recoveries, it's $280. Include the reopened claim and you get $600 gross and $480 net. Every amount is calculated correctly from the same rows. The definition decides which one is relevant. A quarter-end answer adds two more requirements: historical status and the reporting cutoff. A trustworthy system retrieves the approved definition, or asks. It never silently picks among four totals.

STAGE	WHAT HAPPENED	AI'S PART	WHO DECIDED
Discover	Three claims systems and a payments feed; 11 reports compute the figure	Scanned schemas and report SQL; drafted lineage	Data lead confirmed sources
Agree	Finance reported gross; Claims reported net of recoveries	Showed which reports used each definition	Finance owns the gross measure; Claims keeps a named net measure
Build and test	Transformations, tests and a bridge from claim payments to the ledger	Drafted code and tests; engineers wrote critical expected cases	Engineers reviewed; Finance accepted the bridge
Migrate	Five years backfilled; two month-ends compared before cutover	Drafted backfill and difference reports	Owner accepted before cutover
Change	8 reports moved to the product; 3 duplicates retired	Found consumers through lineage and logs	Change lead set the retirement date

WHAT MADE IT WORK Explicit, named definitions with owners; expected results set before the build; and a change lead who knew every report that would move.

Coding agents change implementation, not the lifecycle.

Coding agents can explore a repository, draft a plan, write code and tests and operate tools fast enough to change how work gets organized. The accepted outcome still depends on purpose, current context, independent challenge, release authority and observed operation. The useful unit of delivery is a governed change package: intent, context, plan, artifacts, evidence, authorization and the result seen in production.

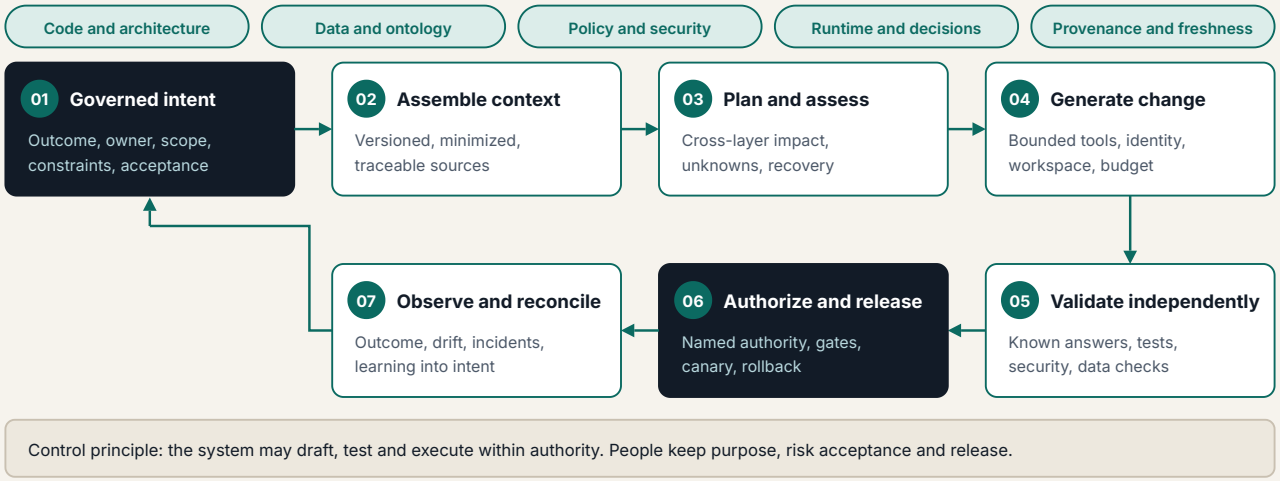


Exhibit: Agentic delivery is a controlled evidence loop. A faster implementation step does not remove the need for authority, independent validation or operational learning.

Keep each lifecycle function when a ceremony changes. A planning meeting, design review, code review or change board can shrink once a better artifact, service or control serves its purpose. NIST's notional DevSecOps reference model keeps plan, develop, build, test, release, deploy and operate, connected by feedback and control gates [11]. Combine phases or automate transitions. Just keep the ability to show what entered each transition, which checks ran and who had authority.

INDEPENDENT VALIDATION

The expected answer must never come only from the path that produced the change. Use approved examples, contract tests, negative authorization cases, migration reconciliation and canary measures. A second prompt asking the same agent to critique its work can improve the draft. It isn't an independent check.

Expand authority as the evidence allows.

STAGE	HOW THE WORK CHANGES	CONTROL AND EVIDENCE THAT REMAIN
Governed intent	Agents structure requests, find precedents and expose missing decisions	Named owner, approved outcome, scope, examples, authority and economics
Context and impact	Tools assemble code, data, policy, runtime and decision context	Source, version, freshness, access boundary, conflicts and affected consumers
Implementation	Agents draft code, contracts, migrations, configuration and tests	Task-scoped identity and tools, approved plan, traceable changes
Independent validation	Automated evaluators run more cases and prepare failure evidence	Maintained expected answers, negative cases, reconciliation, security limits
Authorization and release	Release services enforce preconditions and run a bounded rollout	Named authority, immutable candidate, canary, rollback, evidence of execution
Operation and learning	Agents correlate telemetry, investigate incidents and propose fixes	Observed outcome, reconciliation, incident command, updates to intent

Measure accepted change, not generated code

Measure from complete intent to accepted operating result: lead time, first-pass acceptance, rework, escaped defects, reviewer time, recovery time, total cost and the business measure attached to the intent. A team that generates ten times more code and accepts no more valuable change has grown inventory, not throughput. METR’s results, earlier in this paper, are a reminder to measure the whole workflow in its setting [2, 3].

Grant authority in steps. First assistance and inspectable plans. Then proposals in an isolated workspace. Then bounded build and test actions. Then preauthorized low-risk releases with current preconditions, independent tests and rollback. Narrow it again when the context service, tests or target system can’t hold the boundary.

Measure accepted answers, and start with one question.

ROLE	ACCOUNTABLE FOR
Data product owner	Scope, consumers, the contract, context policy and retirement of old outputs
Business owner and steward	Definitions, defaults, exceptions and acceptance of material changes
Data, analytics and AI engineers	Pipelines, context, tests, evaluations and review of AI-written changes
Platform team	Shared delivery, environments, policy as code and observability
Change lead	Consumer mapping, communication, training and adoption

MEASURE	WHY IT MATTERS
Time to trusted data	Request to accepted product, paired with answer quality and total cost; split working time from waiting time
Accepted-answer rate	Share of answers accepted without correction, by question type
Clarification and refusal	Whether the system asks or stops when meaning or evidence is insufficient
Change failure and restatement	Releases that cause an incident or a correction
Adoption	Intended consumers on the product; unmanaged duplicates retired

A bounded first step

Start with one important question and the decision it supports. Make its definitions explicit, connect the relevant sources, establish known cases and test the complete path from request to answer. Use AI to discover context, write and check transformations and explain results. Expand its autonomy where the evidence supports it. To measure the value of context itself, compare the same model and tool budget four ways: with schema only, with descriptions, with approved context and examples, and with clarification and validation. Score them independently on held-out questions.

THE GOAL A system that answers quickly when the meaning is settled, clarifies when it’s ambiguous and stops when the evidence falls short. That takes engineering the context and the controls alongside the code.

PRODUCT CONSTRUCTION / SOURCE CORRECTION

A data rollback is not simply a code rollback.

Carry the same synthetic invoice correction through specifications, implementation and consumer recovery.

CHANGE STEP	ENGINEERING WORK AND DECISION
Identify	Bind the event to invoice/entity/currency and source version. Separate an identical replay from a revised fact or backdated event.
Assess	Trace affected products, snapshots, measures, reports and agent tools. Determine whether a closed period or consumer contract is affected.
Propose	An engineer or bounded coding agent prepares mapping changes, tests, policy bindings and migration/recompute steps as a reviewable change.
Prove	Test join fan-out, duplicate events, late arrival, missing credits, access denial and reconciliation at consistent cutoffs. Preserve results and independent approval where required.
Release	Promote compatible code, contract, rules and data versions. Backfill or restate only within approved authority; never replay invoice commands as a data correction.
Recover	If release fails, reconcile published versions and downstream effects. Restoring code may leave modified records, extracts or completed external actions to resolve.

Development and production services need separate credentials and permitted data. IDE access to catalog descriptions does not grant source access. Give the coding assistant approved specifications and synthetic examples; require CI checks and accountable review of generated changes.

See Data Products in Practice for the values and product relationships. Rewiring Data Management with AI describes how the approved intent and change package reduce repeated handoffs without transferring authority to the model.

Design references reviewed 29 October 2026: Databricks: medallion architecture; Designing data products; Google SRE: service objectives. The worked records, controls and targets are illustrative RJL recommendations.

Sources and how they were used.

Prepared by Richard White. Version 3, October 2026. The field notes are composites drawn from the author's experience in large institutions, with details changed. The worked example and its figures are illustrative. Study and survey figures are reported as their sources state them; benchmark results change, so check current figures. Links accessed 12 October 2026.

[1] DORA (Google Cloud), *State of AI-assisted Software Development*, September 2025, and the DORA AI Capabilities Model. Survey-based associations. cloud.google.com

[2] Becker, J., Rush, N. et al., METR, "Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity," July 2025. 16 developers, randomized by task.

[3] METR, update on its developer productivity study, 24 February 2026: new estimates treated as an unreliable signal because of task selection; study redesigned. metr.org

[4] Lei, F. et al., "Spider 2.0: Evaluating Language Models on Real-World Enterprise Text-to-SQL Workflows," ICLR 2025. 21.3 percent for the best system in the original setting. arxiv.org/abs/2411.07763

[5] Spider 2.0 official site and leaderboard. Snow and Lite settings; the original setting was replaced by Spider2-DBT in May 2025. spider2-sql.github.io

[6] Fowler, M. and Sadalage, P., "Evolutionary Database Design," and Sato, D., "Parallel Change." martinfowler.com/articles/evodb.html; martinfowler.com/bliki/ParallelChange.html

[7] Jones, A., *Driving Data Quality with Data Contracts*, Packt, 2023.

[8] Snowflake documentation, Cortex Analyst Verified Query Repository. docs.snowflake.com

[9] Anthropic, "Effective context engineering for AI agents," 29 September 2025. anthropic.com/engineering

[10] Gartner, "Lack of AI-Ready Data Puts AI Projects at Risk," 26 February 2025. Prediction through 2026; July 2024 survey of 1,203 data management leaders. gartner.com

[11] NIST National Cybersecurity Center of Excellence, "Notional Reference Model for DevSecOps for Demonstration of NIST SSDF." Lifecycle phases, feedback and control gates. pages.nist.gov

Background reading: Dehghani, Z., *Data Mesh*, O'Reilly, 2022; DAMA-DMBOK, 2nd edition, 2017; BCBS 239, 2013.

Companion papers at rjlsol.com/whitepapers: Data Products in Practice, Context Before Query, Adoption by Design.
Proposed operating guidance; not legal advice.

ABOUT

Richard White

I was global firm chief data officer at McKinsey & Company, owning data, AI and platforms end to end, and before that chief data officer at New York Life, where I built the chief data office and reported data risk to risk committees at multiple levels. Earlier I led data at the Federal Reserve Bank of New York and enterprise data governance at Citigroup. I founded RJL Solutions in 2024. My book, *The Day After the Demo*, is free at rjlsol.com/book.

RJL Solutions connects strategy, architecture, engineering, governance and operation through Frame, Prove, Build, and Run & Transfer. Each phase is separately scoped and priced in a written proposal before you commit. The first sixty-minute working session is free.

MEANING **Context Before Query**
Business meaning before the call; policy at the data.

PRODUCTS **Data Products in Practice**
Boundaries, contracts, quality and certification.

ADOPTION **Adoption by Design**
The people work that decides whether a capability is used.

RJL SOLUTIONS

WEB rjlsol.com

EMAIL rwhite@rjlsol.com

Examples are illustrative and describe no client engagement or client result. Field notes are composites with details changed. Study, survey and benchmark figures are reported as their sources state them.