AgenticObjects

TECHNICAL WHITE PAPER · SEPTEMBER 2026

From Answers to **Records**

A technical guide to agentic analytics you can stand behind

Written for the people who will take it apart: data, platform and architecture leads, and the technical evaluator on a buying team.

CATEGORY
Technical White Paper

PUBLISHED
September 2026

AUTHOR
Ali Yıldırım
Founder & CEO, D-CAT Group

A D-CAT Group product

CONTENTS

> Capability statements in this paper are marked against the product's own verified status register. Roadmap items are labelled as roadmap, every time.

Foreword

The answer disappeared.

Three years ago the cost of producing an analytical finding collapsed. An agent connected to a governed data model now generates observations, comparisons, anomaly reports and recommended actions continuously, overnight, at a marginal cost approaching zero.

Nothing happened to the side that consumes them.

That side still runs on human working memory, and that capacity is a research finding rather than a product parameter. It has not moved. It will not move.

So the industry solved generation and left everything after it untouched. And in the gap, something quietly went wrong that nobody has named.

The output disappears.

You ask, the answer arrives, you close the window. Next week the same question returns a different number. Only the person who asked ever saw it. Six months later, in an audit, "what did we base that decision on?" has no answer at all.

This paper is about what has to replace that. Not a better answer. A different kind of thing entirely.

Cowan's 2001 review of the evidence put short-term capacity at *"a single, central capacity limit averaging about four chunks."*

No estimate in that literature lands anywhere near what an agent produces before a coffee refill.

Agents execute. Records remain.

1 Four ways your AI output fails

It is hard to leave what you have paid for. You have invested in dashboards, in a BI team, in a copilot licence, in the processes around all three. But the honest position is this: **every form your AI output currently takes fails in the same place, and none of them fail for reasons that formatting can fix.**

01

The chat answer

Fluent, immediate, and gone when the window closes.

02

The generated document

Looks like a record. Is a snapshot severed from its source.

03

The emailed digest

Arrives where the reader is, without its permission context.

04

The dashboard

Persistent, governed, live, and carrying no claim.

The chat answer

Fluent, immediate, contextual, and gone when the window closes.

It has no identity you can cite next quarter. It cannot be found again by anyone who was not in the conversation. Its numbers carry no address you can follow back to a query. It cannot be linked to the recommendation it produced or the briefing that covered it.

And there is a second problem, which is worse and better documented.

That finding, from a study across three datasets, is the one every vendor in this category should have on the wall. Adding an explanation to a machine recommendation did not improve team accuracy. It improved *acceptance*. Showing your reasoning, by itself, buys agreement rather than correctness.

Vasconcelos and colleagues later supplied the reason: an explanation only helps when it actually lowers the cost of checking. Their conclusion, *"the explanation not sufficiently reducing the costs of verifying the AI's prediction"*, is the design brief for everything in this paper.

> *"explanations increased the chance that humans will accept the AI's recommendation, **regardless of its correctness**."*
>
> Source: Bansal et al., CHI 2021

The generated document

The PDF, the Word file, the exported deck. This is the case most often mistaken for a solution.

A generated document *feels* like a record. Filename, date, a place on a shared drive, a signature block. But it is a snapshot severed from its source. No figure in it traces back to the query that produced it. Nothing revises it when the number is corrected upstream. It keeps circulating by email long after it stopped being true, and every forward makes it look more official.

It has the appearance of authority without any of the properties that would justify it, which, in front of an auditor, makes it **worse** than the chat answer it replaced.

The emailed digest

It reaches the reader where they already are. That is its whole advantage, and it is a real one.

But the reader has no way to tell what data scope the sender was entitled to see. A summary forwarded twice has lost its permission context entirely, and nobody in the chain knows it.

The dashboard

The incumbent, and it fails differently from the other three.

It is persistent. It is governed. It is connected to live data. What it does not do is carry a **claim**.

Turning a chart into a statement (*this fell, in this segment, it matters this much, here is what we should do*) still happens in a human head. Nothing in the system records that it happened, on what basis, or by whom. The dashboard shows data. The decision was made somewhere else, off the record.

What all four are missing

The axis that matters is not ephemeral versus persistent, and not chat versus document. It is whether the delivered thing carries five properties:

It stays complete and wrong.

	PROPERTY	THE QUESTION IT ANSWERS
1	**Identity**	Can I cite this in six months?
2	**Address**	Where did each number come from?
3	**Authority**	Who was entitled to see this, when it was made?
4	**Type**	What kind of claim is this, and what does that oblige?
5	**Lifecycle**	What happens when it is superseded?

Read honestly, the four are not empty. If you run a governed BI platform you already have some of this, and it was expensive to build. Row-level security enforces authority every time a dashboard is opened. Column-level lineage traces a measure back through the semantic layer to its tables. A document in a records system has an identifier, a version history and a retention class. Some assistants now show the source they consulted.

None of that is nothing. But look at **where the property sits**, and **when it is applied**.

In every case it belongs to the container: the dashboard, the file, the mailbox. And it is applied live, at read. Lineage tells you how a metric is built; it does not tell you which value was on the screen when somebody decided something. Row-level security tells you what a user may see now; it does not tell you what the author was entitled to see then, and it is left behind the moment anyone exports or screenshots. A retention schedule governs the document; the claim inside it has no lifecycle of its own.

> **So the gap is not that these properties are missing from your stack. It is that none of them travels with the number.**

And no amount of formatting makes them travel, because not one of them is a formatting property. Auditability cannot be bolted onto a rendered artifact after the fact. It has to be written onto the thing at the moment it is created.

Where each property actually sits

Five properties decide whether a delivered thing can be relied on later. Most stacks have some of them, on the container, applied at read. The question is whether any of it travels with the number.

○ absent ◐ present on the container, applied at read, not carried by the claim ● frozen onto the record at creation

	Identity cite it later	Address per number	Authority who could see it	Type enforced rules	Lifecycle revision
Chat answer gone when the window closes	○	◐	◐	○	○
Generated document a copy severed from its source	◐	○	○	◐	◐
Emailed digest permission context lost in the forward	◐	○	○	○	○
Dashboard governed and live, and it carries no claim	◐	◐	◐	○	◐
Agentic object a record, not a rendering	●	●	●	●	●

Half-filled is not a criticism of your stack. It marks a property that really is there, on the container, applied at read. It is not carried by the number you acted on.

Figure 1 The five properties, read against the four forms AI analytics output takes today. Half-filled marks a property a governed stack really does have, on the container and applied at read, but that is not carried by the delivered claim.

The gap is not that agents produce too little.

It is that what they produce cannot be relied on later. Section 2 defines what “relied on” means, using a standard that has existed for decades.

2 What a record has to be

"Record" is not a marketing word. It has a definition in the international standard for records management, **ISO 15489-1:2016**, which defines a record as:

"information created, received and maintained as evidence and as an asset by an organization or person, in pursuit of legal obligations or in the transaction of business."

Source: ISO 15489-1:2016, clause 3.14

The same standard names what an authoritative record must be able to demonstrate. Read the four properties against the two forms most AI analytics output takes today.

ISO 15489-1:2016	WHAT IT REQUIRES	A CHAT ANSWER	A GENERATED DOCUMENT
Authenticity	it can be proven to be what it says it is: made by whoever it says made it, at the time it says	No stable identity, no sealed producer context	Filename and date, but nothing provable about which run, whose identity, what access
Reliability	its contents can be trusted as a complete and accurate account, and relied on for what comes next	Nothing binds the sentence to the query	Whatever binding existed was lost at export
Integrity	*"complete and unaltered"*	Regenerable, different every time	Unaltered, and uncorrectable
Useability	*"can be located, retrieved, presented and interpreted within a time period deemed reasonable by stakeholders"*	Retrievable only by whoever held the session	Retrievable only by whoever holds a copy

Both fail all four **as delivered**, and they fail structurally rather than accidentally. A records system can supply authenticity and useability for the *file*. That is exactly what records management was built to do. Nothing in the chain supplies them for the *claim* inside it.

Observability, auditability, defensibility

There is a vocabulary problem in this market, and it costs buyers money. Three different properties get sold under one word.

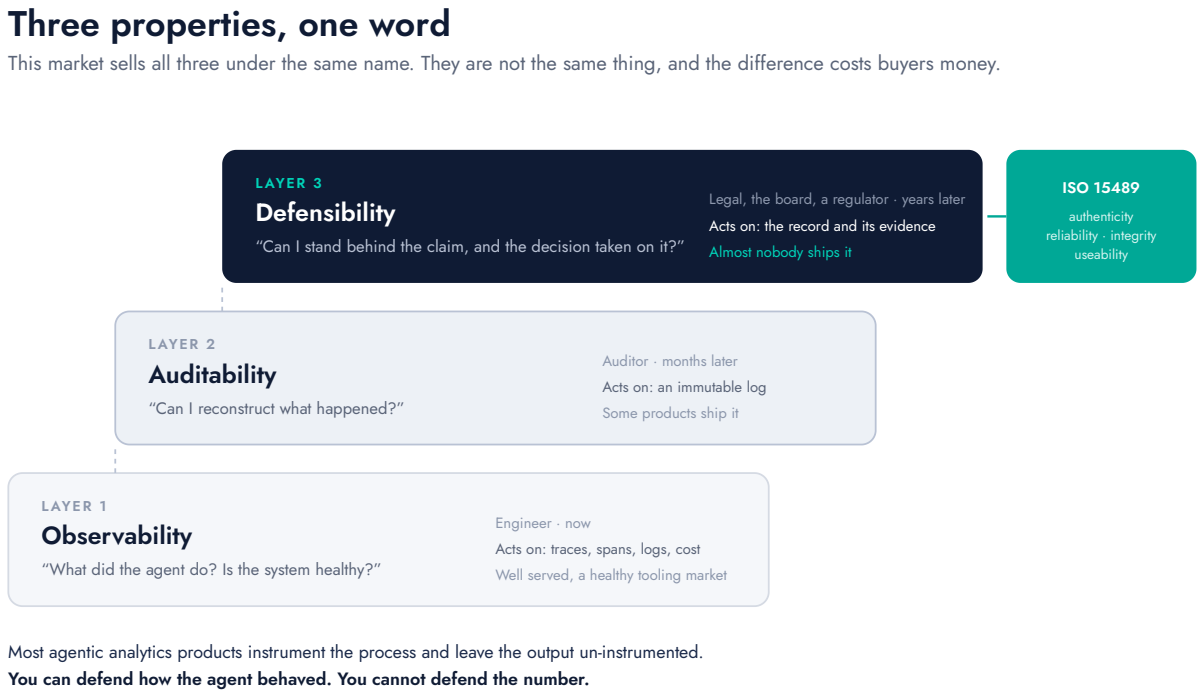


Figure 2 Three properties sold under one word. Observability, auditability and defensibility answer different questions, for different people, on different timescales.

Observability came from control theory into DevOps and then into LLM operations. It is real, it is necessary, and it is now well served, with a healthy market of tracing tools. Its object is **the agent's behaviour**.

Defensibility is a different thing. It comes from records management and e-discovery, and its object is **the claim the agent produced**.

ISO 15489's four properties are, in effect, the operational definition of defensibility. The standard says nothing about observability; the axes are orthogonal.

Most agentic analytics products instrument the process and leave the output un-instrumented. You can defend how the agent behaved. You cannot defend the number.

That is the gap AgenticObjects was built to close.

A note on what we are not claiming: ISO 15489-1 is a concepts-and-principles standard, not a certifiable one. The certifiable records management standard is ISO 30301. We do not hold a certification. We designed against the definition.

Observability tells you what the agent did. Auditability lets you reconstruct it. Neither tells you whether you can stand behind the claim it produced. That is what a board asks about, years later.

3 AgenticObjects

An **agentic object** is a persistent, evidence-bound, auditable unit of business knowledge produced by an agent run. AgenticObjects is the platform that produces them, and that refuses to publish the ones it cannot support.

Not a message. Not a cached answer. It lives in an archive, appears in a feed, links to other objects, carries the authority context it was born with, and is never physically deleted, only archived. Where a chat answer is an event, an agentic object is a thing.

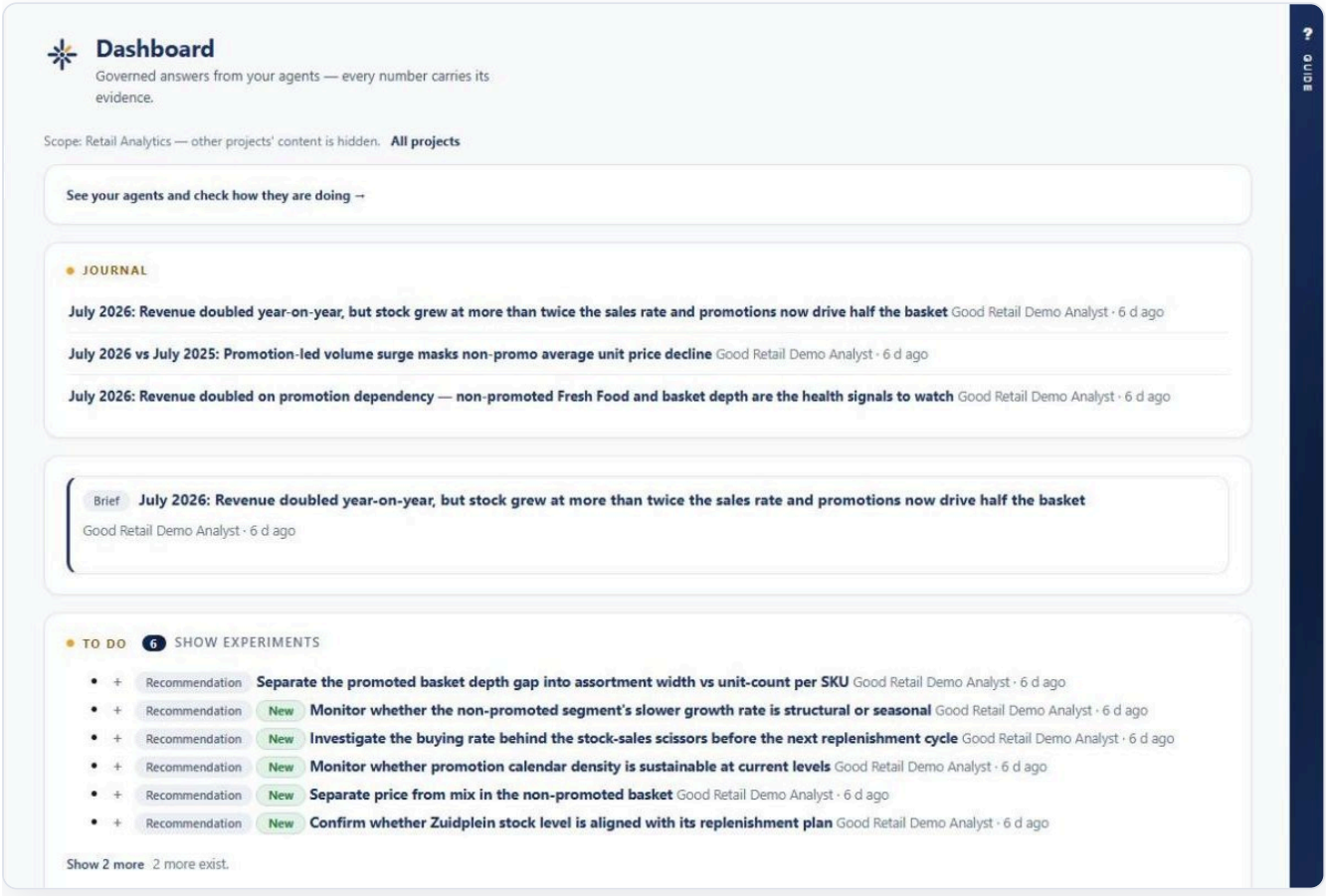


Figure 3 Published objects in the product's own feed: briefs, findings and recommendations, each a citable record rather than a message. Demo workspace; brand and store names are the demo set.

Identity and persistence

Durable identity: the run, the agent, the workspace, a content hash, a revision lineage, a lifecycle stage. Superseding creates a new revision rather than overwriting. There is no physical delete in the model and no `DELETE` endpoint in the API.

A typed claim

Not free text with a title. A type, with obligations the system enforces *before* publication rather than after.

An evidence chain

Every business number bound by address to a value inside a governed query result, checked by a deterministic engine.

Frozen authority

Sealed at creation with the producing identity and the row-level visibility scope in force at that moment, then re-checked against the reader's live access every time it is opened. Two users never see conflicting versions of the same finding; a user who should not see it does not see that it exists.

Figure 4 The four structural properties every agentic object carries.

Three types, and the rule each is born under

FINDING

"What happened?"

A measured business observation: how a metric behaved, in a breakdown, over a period. Carries a mandatory severity (low / medium / high / critical). The metric, period and filters are written by the engine, not the model. If the query ran and returned nothing, that is not a finding but a *data state*: its own class, no severity badge, and nothing can be built on it.

Born under: a query that returned rows.

RECOMMENDATION

"What should we do?"

A business action grounded in at least one finding. **A recommendation with no evidential parent cannot exist.** Not "is discouraged". Cannot exist. It inherits the most restrictive usage policy of the findings beneath it, so it can never be more widely shareable than its own evidence.

Born under: `based_on` ≥ 1. No parent, no object.

BRIEF

"What mattered this week?"

The readable summary that gathers a run's findings into one text. At least one `covers` link is mandatory. First opening is stamped once and never rewritten, so "unread" means never opened.

Born under: `covers` ≥ 1.

Figure 5 The three object types shipping today, and the rule each is born under.

Three link types connect them: `based_on`, `covers`, `related_to`. Knowledge becomes a small navigable graph rather than a flat list. Deliberately small: a different birth context is never a new type, and a type opens only when its lifecycle or its governance meaning genuinely differs.

Figure 6 A recommendation expanded to show the findings it rests on. *Based on these findings* is not a footnote; without at least one `based_on` link the recommendation could not have been published.

How a number earns its place

The language model never produces a business number.

The number comes from a query executed against a governed semantic model, through a read-only gateway. The model narrates and orchestrates. Every figure in its text must be bound by a structural address to a value in the returned result, and a deterministic engine, one that contains no language model, verifies that binding.

Ambiguous multi-matches are rejected rather than guessed. A figure that cannot be resolved is withheld or visibly marked unverified, and the reason is written into the record.

And because Bansal's finding is real, verification is layered rather than dumped:

1. **Layer 1:** value and claim badges on the card itself.
2. **Layer 2:** evidence detail: which query, which period, which scope.
3. **Layer 3:** the executed query, behind a specific named permission granted directly rather than inherited from a role.

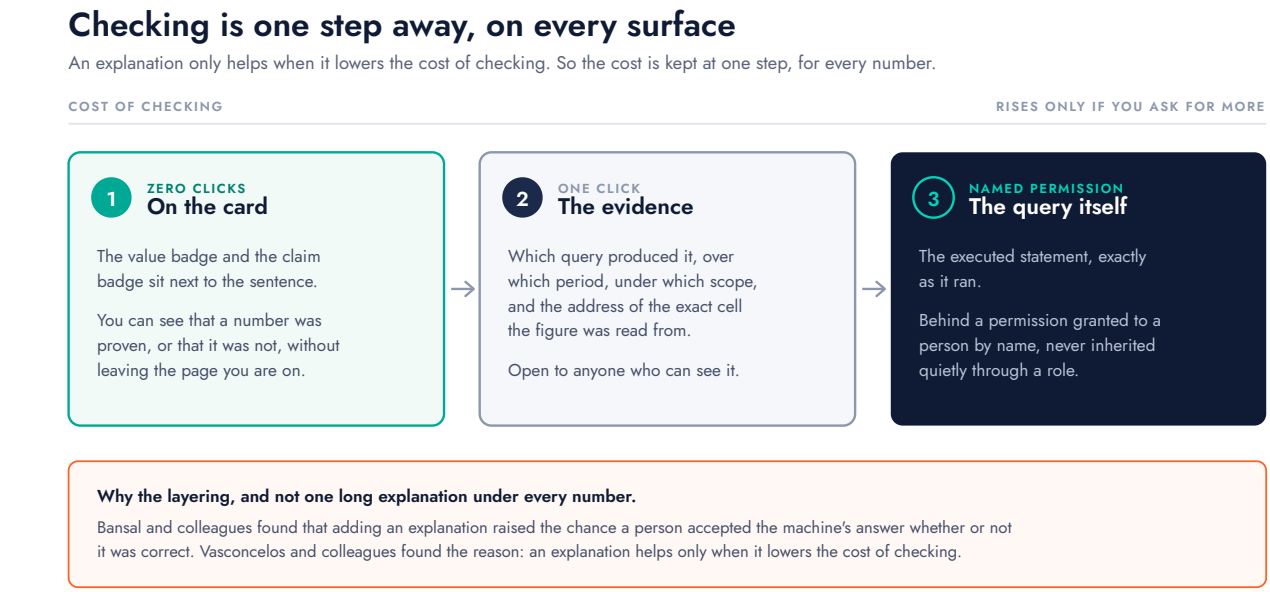


Figure 7 Verification is layered so that the first layer costs nothing. The badge is on the card, the evidence is one click away, and the executed query sits behind a permission granted by name.

Verification is one step from the number, for every number, on every surface. Transparency is not a screen you navigate to. It is a property of every screen.

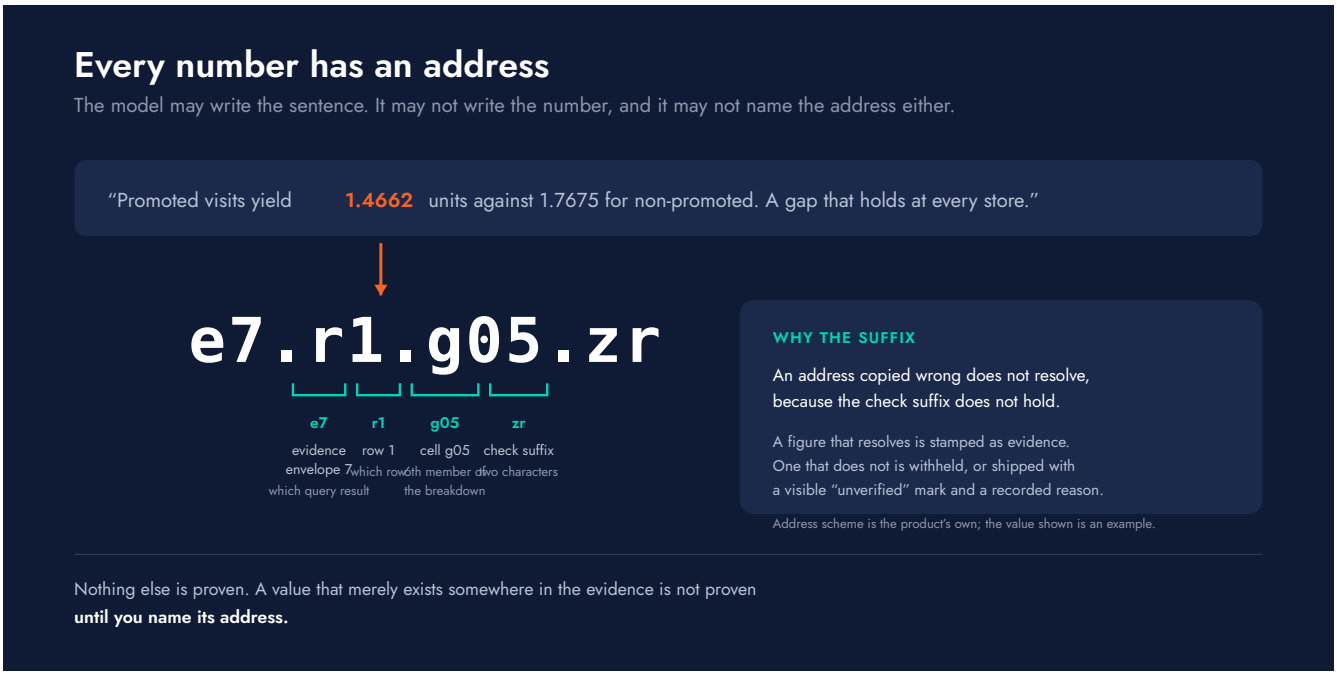


Figure 8 How a number is addressed. Every figure in an agent's text resolves to one cell of one row of one governed query result, or it does not ship.

The eight gates

A candidate object is drafted by the language model. Whether it becomes a record is not the model's decision.

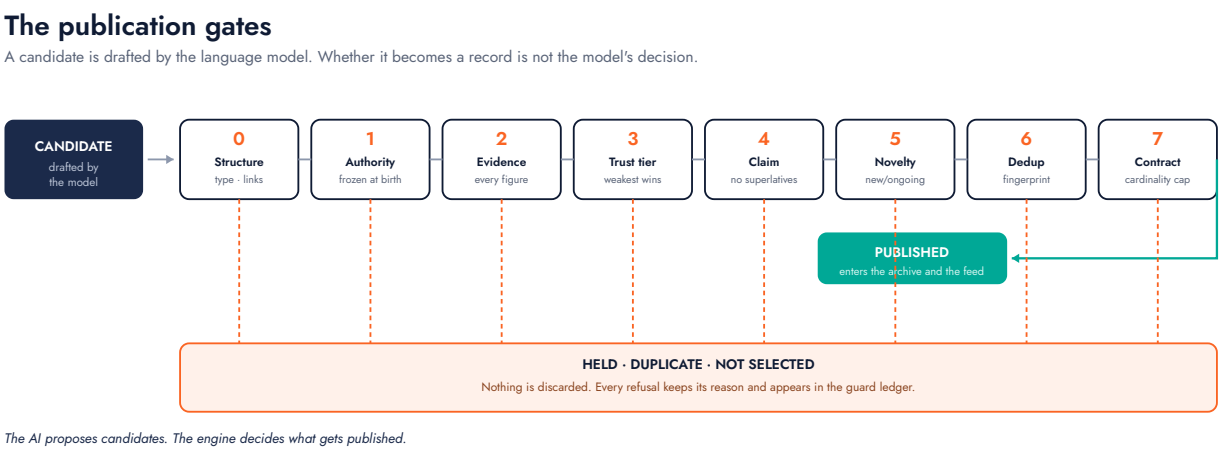


Figure 9 The eight publication gates. A candidate that fails a gate does not vanish. It lands in a recorded state, with its reason.

	GATE	WHAT IT DOES
0	Structure and links	Missing title, type or grounding → dropped
1	Authority freeze	Sealed with producing permission and access scope
2	Evidence	Every number tested against the cited query results
3	Trust stamp	The tier of the **weakest** evidence it rests on; the engine writes it
4	Claim check	Unevidenced causality and superlatives are structural defects
5	Novelty	New / ongoing / repeat, against this agent's own history
6	Deduplication	A second object with the same identity stays as a duplicate, counted
7	Cardinality	The run's frozen contract caps how many of each type may publish

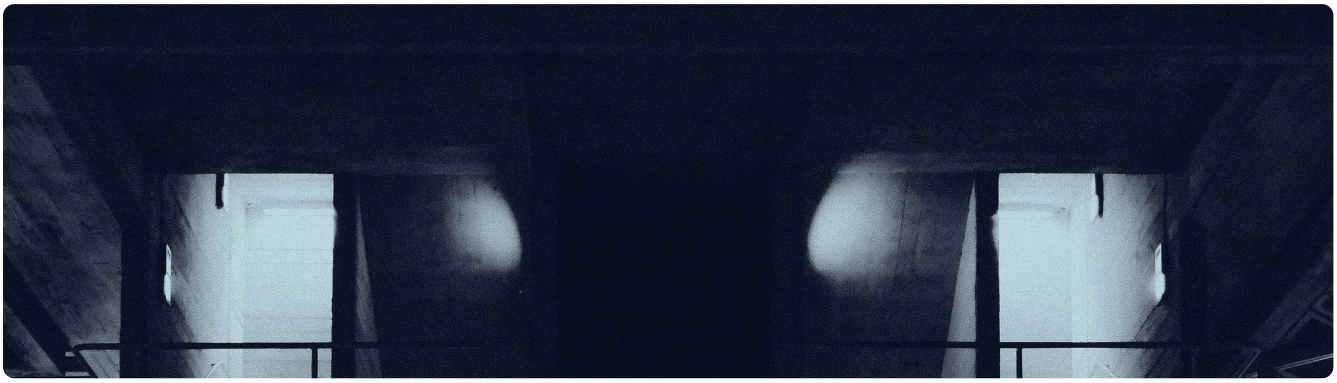
Outcome: `published`, `validated`, `held`, `duplicate` or `not_selected`. Archival is a separate timestamp.

Two details carry more weight than their size. **Held objects keep their reason**, visible in a guard ledger. A governance layer whose rejections are invisible cannot be audited, only trusted, and the point is to remove the need for trust. And **overflow candidates are not lost**: surplus is recorded as `not_selected` rather than discarded.

One more, because its absence is common: **the serving model does not silently change.** No automatic fallback, no quality-triggered escalation. You always know which model produced a given record.

> The AI proposes candidates. The engine decides what gets published.

4 What the gates actually caught



Everything so far is architecture. Every vendor in this category can describe one. This section is different: it is one real run, read directly from the product's own database.

The short version.

One run, scheduled at 05:00, no human question, 285 seconds, 78 cents of model cost. Thirteen candidate objects reached the database; eleven published and two were held, each with a recorded reason. The gate stopped a superlative it could not prove. Once, mid-run, it sent the agent's entire draft answer back because fourteen figures in it had no source. Every refusal is in the ledger with its cause, and that is the part no demo shows you.

The rest of this section is the evidence behind those four sentences. Your technical evaluator will want it. If you are content with the summary, section 5 is where the business effect is.

Company, brand and store names are masked. Object identifiers, badges, gate outcomes, refusal reasons, timings and costs are what the record holds.

> Provenance note. These are real outputs of the running system, read from its own database. The agents producing them run against analytics models in our own environment. This is system behaviour, not a customer case study, and we make no claim about business outcomes at a customer.

Run 961 was scheduled. Nobody asked a question. It began at 05:00 and finished 285 seconds later, at a model cost of **$0.78**, frozen from the price row in effect that day and never recalculated.

Thirteen candidate rows reached the database. **Eleven published**: six findings, four recommendations, one brief, every recommendation carrying

a `based_on` link to a specific finding. **Two were held.** And once, mid-run, the engine rejected the agent's entire draft answer and made it start again.

The refusals are the part worth reading.

Figure 10 Run 961, end to end. Thirteen candidates reached the database; eleven published, two were held with a recorded reason, and once, mid-run, the whole draft answer was sent back.

A superlative that could not be proven

Object 111 never published. Its title claimed a service sub-type was *the fastest-growing item* in its segment. Recorded reason: `claim_structural`.

The rule is deterministic. A superlative is provable only if the value sits at the end of an ordered and **un-truncated** competitor set. This set was cut at top ten. Note `revision: 2` on the record: the object was returned to the agent once with coaching, rewritten, and failed again. One coaching attempt, then held.

Nine of the eleven held objects in the whole record are the same family: "fastest-growing", "weakest", "highest."

The class of sentence the gate catches most often is exactly the class an executive is most likely to act on.

Superlatives are what turn a finding into a decision. They are also the hardest claims to prove.

Interpretive language, by contrast, is badged rather than blocked. A causal word in a published finding produced an `interpretive` flag from the same deterministic dictionary, and the object published anyway. A badge marks; a structural defect holds. The record shows both mechanisms, in the same run.

An answer that was not delivered

At step 5 the numeric-provenance check found fourteen figures in the agent's draft with no address. The answer was withheld and the agent was made to rewrite it. What the engine wrote back is in the record verbatim:

> `[engine] Structural rule (numeric provenance): every figure in a final answer is exactly one of three things. ADDRESSED (declared in the claims block with the ref you read it from), COMPUTED (declared with op and operands that are addresses), or STRUCTURE the engine or the user put there (…). Nothing else is proven. A value that merely EXISTS somewhere in the evidence is not proven until you name its address. These figures are none of those: 25, 0,01, 11,61, …`
>
> **Your answer was NOT delivered.**

The product keeps that count on a review page of its own, and the page refuses to draw a trend line through it:

That refusal is the same discipline as the gate itself, turned on our own numbers.

282 answers caught, across **275** runs.

87.2% of them the agent fixed on the next attempt.

This is not a safeguard we designed and hope never fires.

"There is deliberately NO month-by-month trend line here... A rising rejection rate does not mean 'the agents got worse'."

Source: the review screen, explaining its own metric

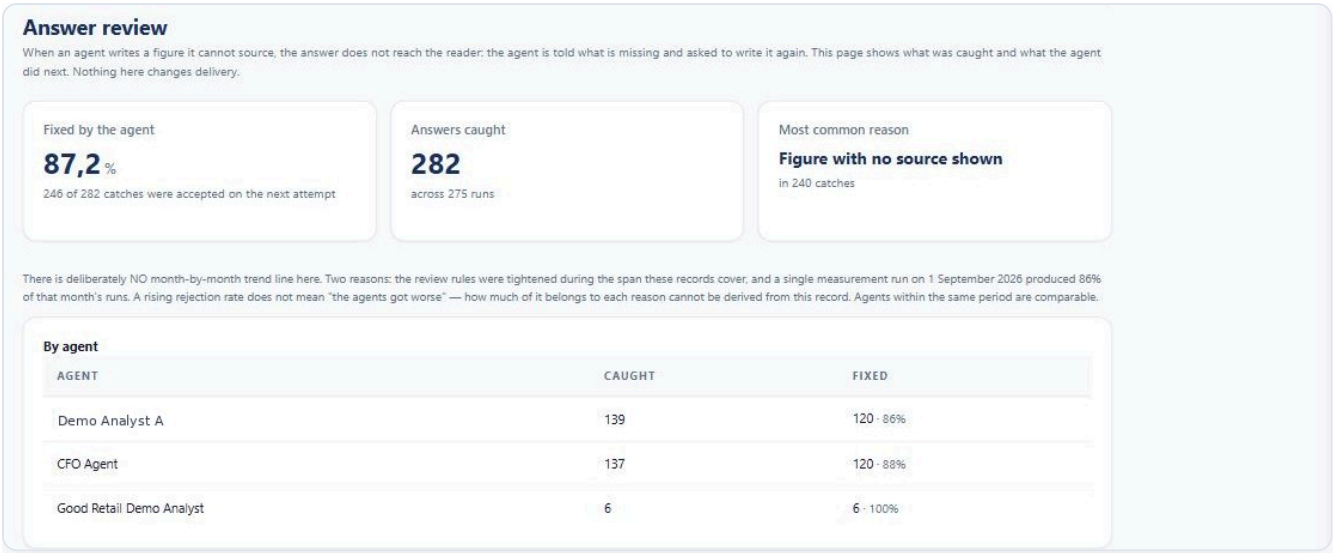


Figure 11 The review page in the product. The paragraph under the cards is the screen refusing to draw a trend line through its own metric. One agent name is masked.

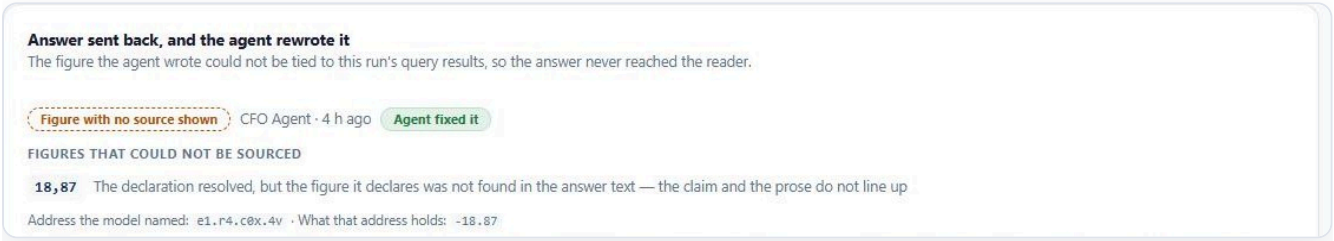


Figure 12 One caught answer, in the record. The engine names the address it expected (`e1.r4.c0x.4v`) and the value that address actually holds. The quoted answer text below it is omitted here.

Where the engine records that it does not know

One published finding carries `_identity_gaps: ["metric_ref:multi_metric_envelope", …]` . The evidence envelope held more than one metric, so rather than guess which metric the finding referred to, the engine left the column empty and wrote down why.

Absence, as a published object

A different run met a period with no loaded data. It neither stayed silent nor reported a zero. It published a finding whose `observation_basis` is `empty_result_set` : every revenue metric for the window returned empty. In the record: *"not a measured zero; there are no recorded transactions in this window."*

That object appears in a separate **Data status** region, under interface text that reads:

> *"These are not business findings: the agent looked and there were no records in the period. No severity badge is shown, because an empty*

A system that guesses is indistinguishable from a system that knows, right up until the moment it matters.

> *period has no business impact."*

Two decisions sit in that region and both are in the code. An absence row carries **no severity badge**, because a "Critical" stamp on an empty window would read as "the product is broken". And the region appears whenever the class exists, even if its rows fall below the fold, so that *"none at all"* never becomes indistinguishable from *"not visible in the list."*

There is a matching brake at the gate: a recommendation resting only on an empty-result observation is a structural defect. **Absence is information. It is not a foundation for advice.**

5 What this changes

The mechanisms are only worth building if a business feels them. Here is what the product's own operating record shows, and what it does not.

The short version.

Volume arrives already bounded: about four to five findings, three recommendations and one brief per run, against a human working-memory limit the literature puts at roughly four things at once. The provenance check caught 282 answers across 275 runs and the agent fixed 87.2% of them on the next attempt. A run costs about 63 cents of model spend and four and a half minutes, frozen at the price in effect rather than estimated afterwards. Six months on, “what did we base that on” has an answer, and so does the harder question: “what did the system decline to tell us”.

Four limits follow the table, because a CIO will ask.

> The record: sixteen completed discovery runs, across three agents, over three and a half weeks. 143 object rows produced, 125 published. These are the system's own operating numbers, not a customer benchmark.

MECHANISM	WHAT THE RECORD SHOWS	WHO FEELS IT
Output capped by contract, ranked by severity and novelty	**4.4 findings, 2.9 recommendations, 1.0 brief** per run	The reader: attention arrives already bounded
Every figure must carry an address	**282** answers caught across 275 runs; the agent fixed **87.2%** on the next attempt	The data team: fewer confident wrong numbers in circulation
Deterministic claim gate	11 held; **9 of them** unprovable superlatives	Anyone about to act on “the fastest-growing”
Duplicate detection on engine-derived identity	7 marked duplicate, kept, counted, not deleted	The reader: one topic, one record
Novelty stamping against the agent's own history	new / ongoing / repeat, correctly stamped across runs	Operations: a repeat is a pattern, not a fresh incident
Scheduled discovery, no human trigger	The example run began at 05:00 and nobody asked it anything	The executive: analysis arrives before the question
Frozen model cost per run	**$0.63** average, **4.5 minutes** average	Finance: an overnight analytical pass priced in cents, frozen not estimated
Persistence with no delete	All 13 published briefs opened at least once; opening stamped once, never rewritten	Audit: “was this read, and when” has an answer

Four limits, because a CIO will ask

1. **This is a small record:** three agents, three and a half weeks. It shows the mechanisms firing under real operating conditions. It is not a benchmark and not a customer result.
2. **The cost figure is model cost per run**, not total cost of ownership.
3. **Candidate counts are a lower bound.** Candidates rejected at the earliest two gates appear in the run report but do not yet leave a database row. The real suppression rate is higher than the table shows, not lower.
4. **We cannot give you a read *rate*.** Briefs are opened and first opening is stamped, but the recipient-list table is not yet populated. A numerator with no denominator. "Three of the six users with access opened it" is sayable. "50% open rate" is not.

Roughly four to five findings per run, against a working-memory limit the literature places at about four chunks.

That is not a coincidence. It is the output contract doing its job.

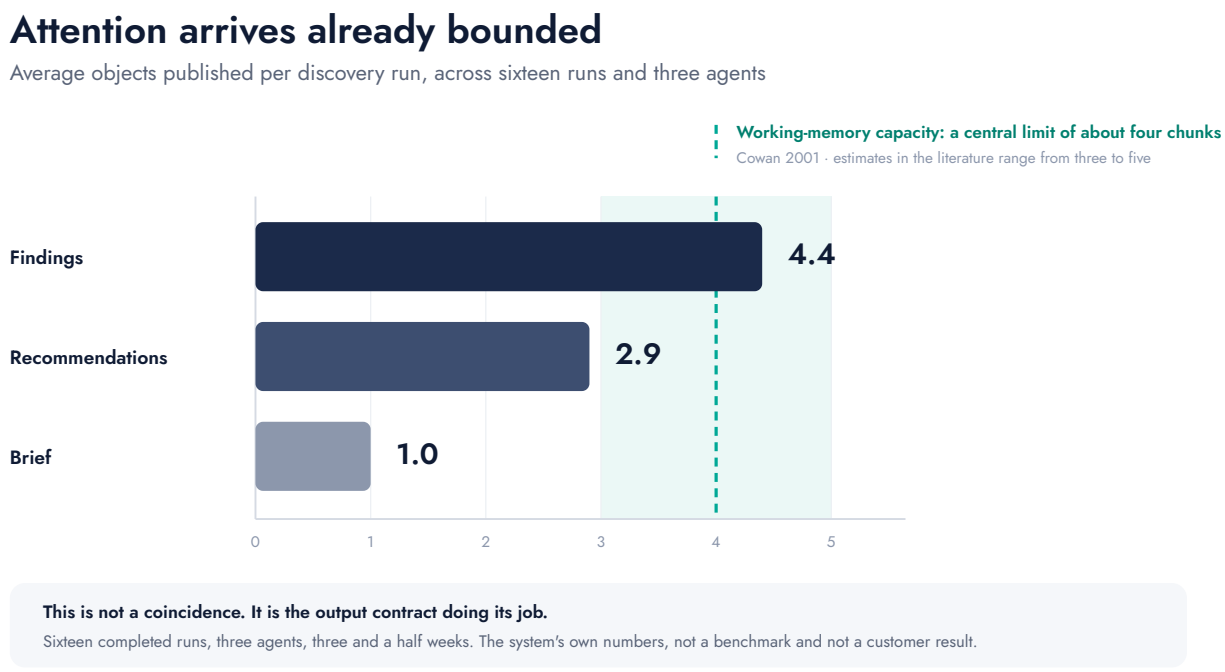


Figure 13 Objects per run, against the working-memory limit the literature places at about four chunks. The output contract, not restraint, is what keeps it there.

What it means for each seat

For the business leader. Volume arrives capped by contract, not by hope. The failure mode of agentic analytics is not that it produces too little.

For the CFO and for audit. Six months later, "what did we base that on" has an answer, because evidence is frozen onto the record and the record is never deleted. The refusals are in the ledger too, so "what did the system decline to tell us" is also answerable, and that is usually the harder question.

For the CIO. Authority is sealed at creation and re-checked at read, so one finding cannot circulate in two versions. The serving model does not change on its own, so last quarter's output is reproducible. Spend is capped before a run begins, and a run that hits its limit stops and says so.

For the data team. The gate absorbs the error that costs most to clean up later: a plausible, confident, decision-shaped sentence with a number in it that nobody can trace.

6 What it is not, and what is next

Positioning is clearer when the fences are explicit. Here are ours.

Not a second semantic or BI engine. The semantic model is authored outside this product, in Axoria Data Studio, and imported as a versioned file. This product imports, validates, versions and exports models. It does not author them.

Not a writer to your systems. The data gateway accepts read-only statement grammar; write and multi-statement forms are rejected at the gateway. Today an agentic object is something a person decides on, not something the platform executes. That is the correct order to build in: the evidence layer has to be trustworthy before anything should be permitted to act on it.

Not a general-purpose knowledge graph. The type set is small and braked, and every type must carry a governance meaning. We would rather ship three types always populated correctly than thirty populated occasionally. That a large type set *causes* degradation is our engineering position rather than a proven finding.

Not an LLM chat wrapper. The model narrates and orchestrates. It never produces the number and never persists knowledge without passing the gates.

Not a claim of perfection. We do not claim zero error, we do not claim every number is necessarily right, and we do not claim equal accuracy in every language. The honest claim describes the mechanism and its refusal behaviour.

Where this goes next

Today's three types are built around one class of claim: **measured**. A number that came out of a query and carries an address. It is the class a machine can check without help, so it is the class we shipped first.

Three more classes are designed, and one principle holds across all of them: **a claim never inherits a verification status it did not earn.** An interpretation has to look like an interpretation. A question has to look like a question. A person's answer has to carry that person's name.

The **Status** column below is the whole disclaimer. *Shipping* is in the product today. *Next* is designed and not yet built. *Later* waits until real usage has accumulated.

	WHAT IT IS	STATUS
Interpretation	Beside every numeric claim, a second list: the inferences the agent drew that no number can settle. Causal and superlative wording is already caught and badged. Next it becomes its own kind of object, so an interpretation is never mistaken for a measurement.	Badge shipping, list next
Question	When the data cannot answer something, the right output is neither silence nor a guess. It is a question, addressed to a named person, surfaced where they already work, with their answer bound back into the knowledge. In every prior tradition a person authors the question. Here the agent finds the edge of its own competence and asks.	Next
Human input	*"The main entrance has been under renovation for three weeks."* No warehouse contains that sentence and none ever will. When a person supplies it, it enters the record with their name on it. Never equated with data, and never thrown away, because it is frequently the part that explains the number.	Next
Absence with an owner	Section 4 shows an absence already published as an object rather than swallowed. Next it gets a life: owned, tracked, closed, instead of reappearing at every scan. What a system can honestly report is *that* something is absent and what it attempted; whether the absence is harmless stays a human call.	Object shipping, lifecycle next
Situation, Investigation, Decision	The macro layer. A living business situation. A persistent investigation file. A decision context that still holds its evidence years after the decision. These wait until the layer beneath them is populated correctly, because a macro layer over an empty foundation is theatre.	Later

Also ahead: the swipe-based consumption surface, endpoint and MCP invocation modes, the Phase 2 drift surface, and full background Watch delivery.

Defined in the schema, not yet exercised. Four states exist in the shipping schema but have not produced a live record: the **urgency** badge on recommendations, **trust tiers B and C** (every object so far has come out A), the **overflow** state (no run has yet exceeded its type cap), and the **lesson-capture** path. They are listed so you find them here rather than in a demo.

7 Your path forward

The decision to adopt a product in this category should not rest on whether the interface impressed anyone in the room. It should rest on evidence, auditability, cost and real usage signals, all of which this product measures about itself.

WEEK 1

Connectivity and the first end-to-end path

Warehouse connection, user roles, semantic-model baseline, and the first question-and-answer screen with its evidence panel behind it.

WEEK 2

First autonomous analysis

Scheduled scanning in one business domain you choose, the first automated finding, and a briefing cadence. Your first brief arrives this week.

WEEKS 3–4

Second domain and evaluation

A second business domain, executive users on the system, the feedback loop running, and the pilot assessed against the criteria below.

Acceptance criteria

Four, and each is testable rather than impressionistic.

1. **Every number shows its source:** traceable to the query and its referenced evidence, in one step, on every surface.
2. **Every assumption is explicit:** an interpretation is visibly an interpretation and never arrives carrying a measurement's stamp.
3. **The spend limit is known before the analysis starts:** step, time and cost limits locked before a run begins; a run that hits one stops and says so.
4. **Every action is in the audit log:** model, user, time, cost and outcome, append-only.

What gets measured, and where it comes from

The pilot is evaluated from the product's own records rather than from a satisfaction survey:

- **cost per analysis:** frozen per run against the price row in effect
- **the rate at which the provenance check catches figures:** how often a number was stopped or flagged before delivery
- **briefing opens and feedback:** first opening stamped once, never rewritten
- **the refusal rate and its reasons:** held, duplicate and overflowed candidates, each with a recorded cause

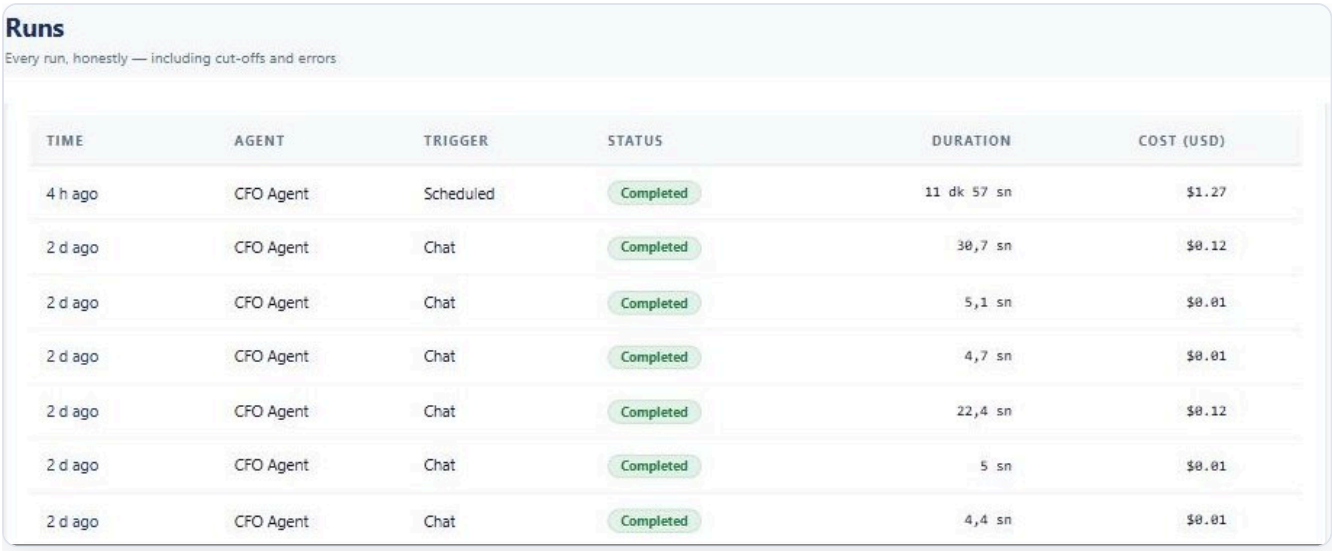


Figure 14 The run log. Every run, with its duration and its frozen model cost, the same rows the pilot is evaluated from.

For reference, our own operating record puts model cost at roughly **$0.63 per run** and shows the provenance check catching **282 answers** across 275 runs. Your numbers will be different. The point is that they will be numbers the system produced, not a matter of anyone's impression.

Before the pilot, though, come the questions.

Appendix A lists twelve worth putting to every vendor you evaluate in this category. We would be pleased if you asked them of us.

Every form your AI output takes today is the wrong container for a business decision, and each is wrong structurally rather than cosmetically. The chat answer cannot be found again. The generated document persists as a copy severed from its source. The dashboard is live and governed but carries no claim, so the work of turning data into a statement stays in a human head and goes unrecorded.

Any vendor can show you a good briefing. Ask to see what the system refused to say, and why.

Appendix A: Twelve questions for every vendor in this category

You will evaluate more than one product in this category over the next eighteen months, and the demos will look alike. These twelve separate architecture from presentation.

On evidence

1. Does the language model produce the business numbers, or does a query engine? Ask to see the boundary in the architecture, not in the pitch.
2. From any number on any screen, how many clicks to the query that produced it? Count them. If explanation is displayed but verification is expensive, the research says you will get acceptance, not accuracy.
3. What happens to a number the system cannot prove? Silently dropped, silently shipped, or held with a recorded reason you can inspect?

On persistence

4. Is the output of a run an object with an identity, or a message? Can you cite it in six months?
5. Can anything be physically deleted? If yes, your audit trail has a hole in it by design.
6. When an object is revised, is the previous version recoverable?

On governance

7. Is authorisation enforced on the server and inside the query, or is the interface hiding rows the backend returned?
8. Is the producing identity and row-level scope frozen onto the object at creation, and re-checked at read time?
9. Can the model serving a run change automatically, through a fallback or a quality escalation? If it can, you cannot reproduce last quarter's output.

On honesty

10. Ask which capabilities in the demo are shipping and which are roadmap. Ask for it in writing. The answer, and the willingness to give one, tells you more than the demo did.
11. Does the product represent absence (no data, stale data, a truncated result), or does absence look identical to a clean negative finding?
12. Which records and AI-governance standards do you position against, and what in the architecture maps to them?

A note on sources

Every external work cited was retrieved and verified from a primary or publisher record before use, and quoted verbatim rather than paraphrased.

- Bansal, G., Wu, T., Zhou, J., Fok, R., Nushi, B., Kamar, E., Ribeiro, M. T., & Weld, D. S. (2021). *Does the Whole Exceed its Parts? The Effect of AI Explanations on Complementary Team Performance.* CHI '21. DOI: 10.1145/3411764.3445717
- Cowan, N. (2001). *The magical number 4 in short-term memory: A reconsideration of mental storage capacity.* Behavioral and Brain Sciences 24(1): 87–114. DOI: 10.1017/S0140525X01003922
- ISO 15489-1:2016. *Information and documentation — Records management — Part 1: Concepts and principles.* 2nd edition, April 2016, ISO/TC 46/SC 11.
- ISO 30301:2019. *Information and documentation — Management systems for records — Requirements.*
- Vasconcelos, H., Jörke, M., Grunde-McLaughlin, M., Gerstenberg, T., Bernstein, M. S., & Krishna, R. (2023). *Explanations Can Reduce Overreliance on AI Systems During Decision-Making.* Proc. ACM Hum.-Comput. Interact. 7, CSCW1. DOI: 10.1145/3579605

> The design of this layer also draws on established work in data provenance, records management and evidence modelling, among others the W3C PROV data model, the nanopublication and micropublication models from scientific publishing, and the separation of evidence quality from recommendation strength in clinical guideline methodology. A fuller bibliography with retrieval records and verbatim source quotations is maintained alongside this paper and available on request.

AgenticObjects

Persistent, evidence-bound, auditable business knowledge.

Agents execute. Records remain.

A D-CAT Group product

FROM ANSWERS TO RECORDS · V1.0 · SEPTEMBER 2026