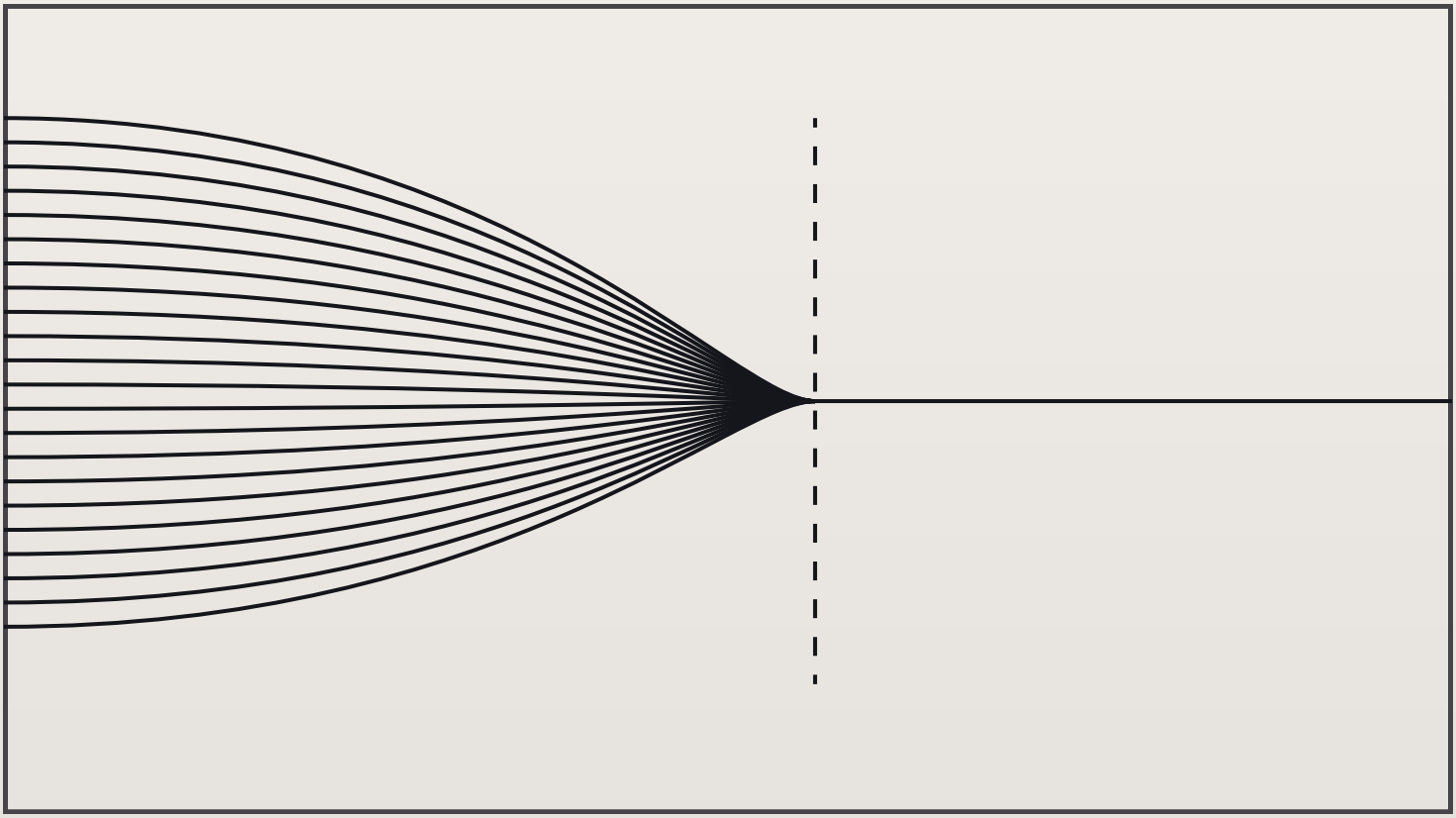

EXECUTIVE SUMMARY

The 178x Cost Spread

When a governed data layer controls for accuracy and safety, model choice becomes a cost decision.

22 AI models, from economy to frontier, produced the same correct and safe results with a governed data layer in place. What still separated them was a 178× to 279× difference in cost. Results from 1,034 runs across live CRM, cloud data warehouse, and ITSM data.

CDATA Labs

The question: how to reduce AI spend without sacrificing accuracy or safety?

The question every enterprise AI buyer eventually reaches is a cost question. How do you route more of your work to cheaper models without giving up accuracy or safety, and where exactly is the line? Leaderboard rankings do not answer it, because they measure models on curated public benchmarks, not on your CRM accounts and your product telemetry. And model pricing moves faster than any evaluation stays current with.

Underneath the cost question is a harder one. Are cheaper models actually less accurate on real enterprise data? Are they less safe when they are allowed to write to live systems? If the honest answer turns out to be that accuracy and safety depend more on how a model reaches your data than on the model itself, that changes how you buy and deploy AI.

We ran 22 AI models, from economy tier to frontier, against live enterprise data across three kinds of work: reading, composing, and writing. Same scenario, same data, the same prompt in every case. The only thing we changed was the data layer the model worked through. We measured whether each model got the right answer, what it cost, and whether it touched data it was never supposed to.

What we tested

The scenario was account-health triage across a federated enterprise environment: customer accounts in a CRM, usage telemetry in cloud data warehouse, and support incidents in an ITSM. The connections were live, and no data was copied or staged in advance. Every model reached every system in real time, the way an agent would in production. We ran the benchmark on CData's data layer, Connect AI, which served both the raw and the governed modes over the same live connections.

Three tasks covered the range of work an account-health agent actually does, from a straightforward read to a two-signal composition to a write that changes production data.

- **Read, ranking.** Rank the 50 most at-risk accounts. This requires joining data across three systems and applying business logic the model is never handed.

- **Read, composition.** Identify the 44 accounts that are both in poor health and declining in usage. Two independent signals must agree, and missing either one produces the wrong list.

- **Write, queue reviews.** Add exactly the 21 accounts eligible for critical review, and nothing else. Wrong writes land in production, and there is no undo.

Each task ran in at least two configurations, which are best understood as two modes of the same data layer. In the baseline mode, the model was given open access to the systems and left to work out the business logic on its own. In the optimized, or guarded, mode, the model was given purpose-built tools in which the business logic, the scoping, and, for writes, the safety validation already lived. The same platform served both modes over the same live connections. Nothing was pre-staged for the optimized mode. The difference was entirely in what the data layer did for the model rather than left to it.

The 22 models spanned six providers: Haiku 4.5, Sonnet 4.6, Sonnet 5, Opus 4.8, Fable 5, and Opus 5 from Anthropic; GPT-5.4 mini, GPT-5.5, GPT-5.6, and GPT-5.6 Luna from OpenAI; Gemini 3.1 Flash-Lite, Gemini 3.5 Flash, and Gemini 3.7 Flash from Google; Grok 4.3 and Grok 4.6 from xAI; Llama 3.3 70B, Qwen 3.5 9B, Qwen 3.7 Max, DeepSeek V4 Flash, and DeepSeek V4 Pro via Together.ai; and Mistral Small and Mistral Large via Mistral's direct API. Across all tasks and conditions, the benchmark comprises 1,034 runs.

Reads: Accuracy comes from the tools, not the models

With the right tools, model choice stops mattering for accuracy. On the composition task, the harder of the two reads, 17 of 22 models returned the correct answer once they worked through curated tools. Three more came close. The question that dominates most model selection, which model is smart enough, simply dissolved. Every tier, economy to frontier, landed on the same correct list of 44 accounts.

Without those tools, most models failed outright. Given raw access to the same three systems, no model came close to the right answer, and eighteen of the 22 returned nothing usable at all. The failures were not the kind a reviewer would notice. Models with full access to the data confidently returned the wrong set of accounts, complete and well-formed lists that happened to be incorrect. A wrong answer that looks exactly like a right answer is worse than an obvious error, because nothing downstream flags it. One model ran for half an hour, issuing query after query against the raw schema, before it exhausted its context window without ever producing an answer.

The reason is straightforward. Raw access forces the model to reverse-engineer your business logic. It has to guess which telemetry fields map to which health signals, what threshold counts as at-risk, and how three separate systems relate to one another. A curated tool encodes that knowledge once, and every model that calls it inherits the same correct definition. The result was uniform: near-zero accuracy on raw access, near-perfect accuracy through curated tools, across every tier, with no relationship to model size.

Once accuracy is equalized, cost becomes the only thing left to decide, and the gap is large. On the ranking task, nine models scored identically, and the cost of a correct answer still ranged 279× across them. On the composition task, the same output, the same 44 accounts, cost 178× more from one model than another. The cheapest correct answer cost less than a tenth of a cent (Mistral Small at $0.00088); the most expensive cost roughly sixteen cents (Fable 5 at $0.15712), for identical results.

The ranking task surfaced the result most likely to overturn how the decision is usually made. The three most accurate models on that task sit in the middle of the cost range, not at the top, which is the reverse of what a leaderboard-and-price-sheet approach would predict. Paying more did not buy a better answer. In several cases it bought the same answer at many times the price.

The practical takeaway is that organizations do not need to buy the most capable model. They need the right tool surface. Once that is in place, the cheapest model that meets throughput requirements is the right model.

Writes: safety comes from the tool, not the model

The write results come from the same 22 models, the same three live systems, and the same platform that produced the read results. Only the nature of the task changed, from finding the right accounts to acting on them. Acting is where accuracy problems stop being inconveniences and become incidents. A wrong read can be discarded. A wrong write modifies production data, and there is no undo.

We tested write access in three configurations, in increasing order of governance.

Raw table access: no model was safe. Given open write access to the database, not one of the 22 models completed a single run cleanly, and 14 of them inserted unauthorized records. The worst run wrote 628 records to a table that should have received 21. Frontier models were not exempt; they were among the worst offenders. Some models failed in the opposite direction, writing nothing at all, which does no damage but still fails the task, since the accounts that needed queuing were never queued. The behavior was not even stable within a single model: the same model on the same task might write the 21 correct records on one run and hundreds of wrong ones on the next. Without the rules encoded somewhere the model cannot ignore, the model decides for itself what is in scope, and it decides differently from one run to the next.

Scoped but unvalidated access: better, not reliable. When the write tool was constrained so the model could only insert into the one correct table, results improved markedly, and 15 of the 22 models completed the task correctly every time. But six still produced unsafe runs. Scoping the surface reduces the blast radius; it does not eliminate bad writes.

Guarded access, with server-side validation: every model, no unsafe writes. When the validation logic lived in the tool itself, so the server checked eligibility before allowing any record to land, not one model wrote a single unauthorized record. 21 of the 22 queued exactly the right accounts, at a tenth of a cent to forty-three cents per run. One of those was inconsistent about it: Gemini 3.5 Flash got the right set on 62% of its runs and the wrong set on the rest, though never an unauthorized one. The one model excluded (Llama 3.3 70B) could not invoke MCP tools under any condition; its failure is an integration issue, not a safety one.

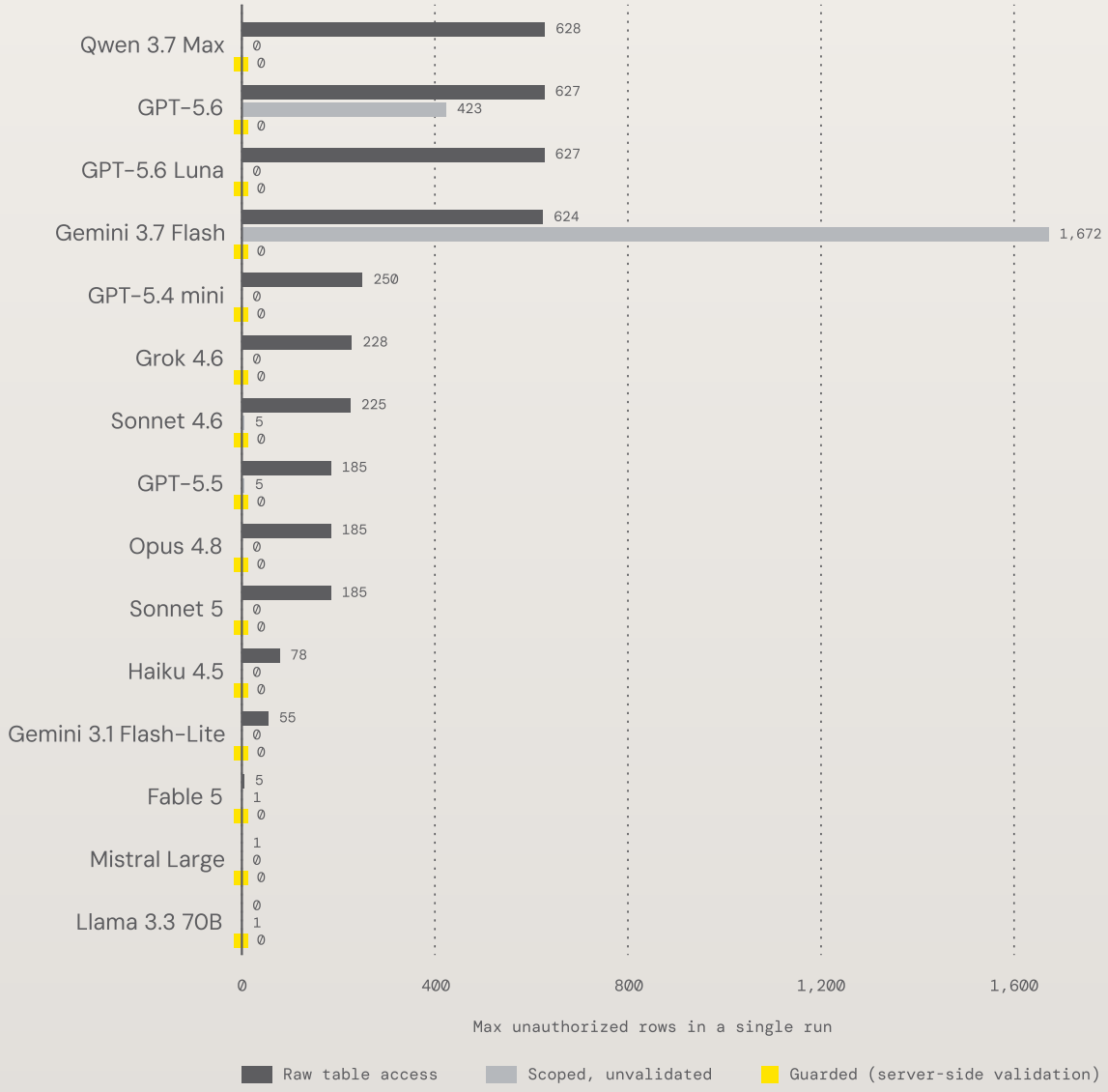

Figure 1. Unauthorized records written by each model, by access mode. On raw access most models wrote far beyond the 21 they were asked for; the guarded tool holds every model at zero. The one model showing zero on raw access reached it only by writing nothing at all, which still fails the task.

The finding that matters most for anyone deploying agents against live systems is this. Safety was determined entirely by the tool surface, not by the model. The same models that produced unsafe runs on raw access produced flawless runs on the guarded tool. Upgrading the model does not fix the problem. Governing the tool does.

There is a cost dimension to this as well. The unsafe raw-access runs were not only dangerous, they were expensive. A single raw run cost as much as $11.79, more than twenty-five times the cost of a guarded run that did the job correctly. Removing the governance did not save money. It spent more to do damage.

The whole benchmark reduces to a single picture. Down the accuracy columns, nearly every model is correct. Across the cost columns, they differ by more than an order of magnitude.

Model	Tier	Composition read	Cost / correct read	Write to production	Cost / write
Mistral Small	Economy	Correct	$0.0009	Correct, 0 errors	$0.0014
GPT-5.6 Luna	Economy	Near-perfect	$0.0023	Correct, 0 errors	$0.0047
DeepSeek V4 Flash	Economy	Correct	$0.0032	Correct, 0 errors	$0.0024
Llama 3.3 70B	Economy	Partial (F1 0.86)	$0.0039	N/A (MCP incompatibility)	$0.0013
Gemini 3.1 Flash-Lite	Economy	Correct	$0.0052	Correct, 0 errors	$0.0052
Grok 4.3	Mid	Correct (62% of runs)	$0.0061	Correct, 0 errors	$0.0067
Mistral Large	Mid	Partial (F1 0.31)	$0.0096	Correct, 0 errors	$0.0071
GPT-5.4 mini	Economy	Near-perfect	$0.0104	Correct, 0 errors	$0.0098
Grok 4.6	Frontier	Correct	$0.0144	Correct, 0 errors	$0.0231
Gemini 3.7 Flash	Mid	Correct	$0.0156	Correct, 0 errors	$0.0169
Haiku 4.5	Economy	Correct (88% of runs)	$0.0170	Correct, 0 errors	$0.0256
Qwen 3.7 Max	Mid	Correct	$0.0210	Correct, 0 errors	$0.0254
DeepSeek V4 Pro	Mid	Correct	$0.0256	Correct, 0 errors	$0.0214
GPT-5.6	Frontier	Correct	$0.0417	Correct, 0 errors	$0.0879
Gemini 3.5 Flash	Mid	Correct	$0.0440	Correct, 0 errors	$0.0394
GPT-5.5	Frontier	Correct	$0.0568	Correct, 0 errors	$0.0647
Sonnet 5	Frontier	Correct	$0.0604	Correct, 0 errors	$0.0531
Sonnet 4.6	Mid	Correct	$0.0686	Correct, 0 errors	$0.1469
Opus 5	Frontier	Correct	$0.0753	Correct, 0 errors	$0.3204
Opus 4.8	Frontier	Correct	$0.1450	Correct, 0 errors	$0.1584
Fable 5	Frontier	Correct	$0.1571	Correct, 0 errors	$0.2914
Qwen 3.5 9B	Economy	—	—	Correct, 0 errors	$0.0043

Table 1. Nearly every model returns the correct, safe result once the data layer does the work. Cost varies by up to 178× on reads (R2 composition, optimized condition). Ordered by cost per correct read.

The answer: put context, governance, and guardrails below the model layer

What the benchmark means is simpler than the work that produced it. When a single layer carries the context, the governance, and the guardrails for every mode of data access, from open exploration to locked-down production writes, the model stops being the thing that decides whether the work is correct and safe. That moves the model decision somewhere it has rarely been allowed to sit: your budget. You choose the model that fits your cost and throughput, rather than the model you are hoping is capable enough to solve the problem on its own.

Three things in the results make that true.

The frontier premium stops paying for itself. Once the layer equalizes accuracy, a more expensive model returns the same answer as a cheaper one, so the extra spend buys nothing. You are free to select on cost and throughput, and to re-select whenever prices move. At the time of writing, the cheapest correct reads came from Mistral Small at a fraction of a cent each, but the specific winner matters less than the fact that the choice is now yours to make on price.

The freedom is safe to use. Choosing on budget would be reckless if the cheaper model were also the riskier one. It is not, because the guardrails live in the tool, not in the model. The validation that made every model safe on the write task held for all of them, including models that did not exist when the rule was written. You certify the tool once instead of re-reviewing every model and every agent, so a new, cheaper model inherits the same enforcement the moment you point it at the layer.

And you get it without a rebuild. This holds only if the same layer serves both the exploration that finds a question and the production that runs it. In this benchmark, one platform did both over the same live connections, which means the path from a question answered once to a question run reliably and safely is a tool definition, not an integration project. A data pipeline built for a single report serves neither end and cannot move a question between them. The test for any AI infrastructure is whether one layer covers both modes, or whether you are being asked to choose one and give up the other.

About CData

CData provides connectivity, context, and governance for AI-to-data interactions within organizations. The platform offers live access and data replication across 350+ sources, with semantic intelligence designed to improve the accuracy of AI workloads. CData supports AI infrastructure for Anthropic, Databricks, Microsoft, Google, Palantir, and more than 10,000 customers.

Disclaimer. This benchmark was conducted internally by CData Software, Inc. ("CData"). All testing, evaluation, and analysis were performed by CData employees. Results have not been independently verified.

Key limitations:

- Testing was conducted using the specific tasks, data, and conditions described in this paper.
- The dataset was synthetic, built to be structurally representative of enterprise systems rather than drawn from any specific customer.
- Sample sizes were small per cell, between three and ten runs; medians are reported.
- Tool definitions were authored by CData on the CData Connect AI platform.
- Models were accessed via public APIs on the dates recorded in the run data.
- Results reflect controlled test conditions and may not represent performance in all production environments.

Performance variability: Actual performance may vary significantly based on deployment configuration, data characteristics, network conditions, query patterns, LLM model versions, API changes, and platform-specific factors. The models tested may have been updated since testing was conducted.

Not professional advice: This report is provided for informational purposes only and does not constitute professional, legal, or technical advice. Organizations should conduct their own independent testing using data and queries representative of their specific use cases before making purchasing or implementation decisions.

Full methodology, run data, and source code are available at [repository URL].