FOR THE PEOPLE BEING ASKED TO APPROVE IT

Why Direct LLM Access to Enterprise Data Is Risky

The case for a governed intelligence layer between language models and enterprise data

JASON PUGH, CHIEF EXECUTIVE OFFICER AND CO-FOUNDER

RAYSON TECHNOLOGIES, LLC | HUNTSVILLE, ALABAMA

VERSION 2.0 | JULY 2026

WHAT IS INSIDE

12 SECTIONS · 4 FIGURES · 25 REFERENCES

> redline · every accuracy figure below is point-in-time and sourced. leaderboards move; the references carry dates.

Executive Summary

Language models have made natural language a workable interface to enterprise data. The shortcut that follows is predictable: connect a frontier or open-weight model straight to the warehouse, either through a database driver or a Model Context Protocol (MCP) server, and let the business ask questions.

That shortcut fails for two reasons. One is measurable. One is structural.

The measurable reason. Text-to-SQL accuracy collapses under enterprise conditions. On clean academic schemas, systems clear 85 percent execution accuracy. On BIRD, which uses real databases with messy values, human experts reach 92.96 percent and the leading published systems sit near 80 percent. On Spider 2.0, which uses enterprise workflows with schemas above 1,000 columns and multiple SQL dialects, frontier models land in the 20 percent range. On BEAVER, built from query logs inside private data warehouses, state-of-the-art agentic frameworks running a current frontier model reach 10.8 percent. Enterprise data is the hard case, not the easy one.

The structural reason. The failure mode is a confident wrong number, not an error message. A model that picks the wrong join path, the wrong grain, or the wrong revenue column returns a plausible figure with no signal that anything went wrong. An engineer can read the generated SQL and catch it. A business user cannot, and should not be expected to.

MCP does not close either gap, and it was never designed to. MCP standardizes transport and tool description. It does not define identity, authorization, business semantics, query validation, lineage, or audit. In May 2026 the National Security Agency's Artificial Intelligence Security Center published guidance making the same point in plain terms: MCP does not define how a session maps to an identity, authentication is optional rather than required, and role-based access control is not part of the protocol. The specification itself states that MCP "cannot enforce these security principles at the protocol level."

The conclusion is architectural, not editorial. Place a governed intelligence layer between the model and the data. The model handles language. The layer owns semantics, authorization, validation, execution, lineage, and explanation. Accountability then lives in software the enterprise controls and can audit, rather than in a probability distribution it rents by the token.

Contents

§1 Scope and audience

This paper is written for CIOs, chief data officers, heads of data engineering, and security leaders who are being asked to approve conversational access to enterprise data. It covers structured data in warehouses, lakehouses, and operational databases.

It does not cover document retrieval, code generation, or customer-service chat. The governance argument transfers to those cases, but the accuracy evidence cited here is specific to querying structured data.

Two definitions are used throughout. **Direct access** means a language model that receives a schema and emits SQL executed against production data, whether through a driver, an API, or an MCP server. A **governed intelligence layer** means a separately deployed component that holds business semantics, compiles and validates queries, enforces authorization at execution time, and records lineage.

---

§2 The accuracy problem is measured, not theoretical

The most common objection to a governed layer is that model quality will make it unnecessary. The benchmark record argues otherwise. Accuracy does not degrade gradually as data gets more realistic. It falls off a cliff.

BENCHMARK	DATA CHARACTERISTICS	BEST REPORTED ACCURACY	WHAT IT TELLS YOU
Spider 1.0 (2018)	Small, clean, curated schemas	Roughly 86 to 91 percent	Syntax and schema structure are largely solved
BIRD (2023)	95 real databases, 12,751 pairs, dirty values, 37 domains	Roughly 80 percent, against 92.96 percent human expert	Dirty data and domain knowledge cost 10 to 20 points
Spider 2.0 (2025)	632 enterprise workflows, 1,000+ columns, BigQuery and Snowflake dialects, multi-step	Roughly 21 percent (6 percent at release)	Enterprise scale and workflow realism break single-shot generation
BEAVER (2024, extended 2026)	9,128 pairs from private warehouse query logs, 812 tables, 19 domains	10.8 percent using a frontier model in an agentic framework	Private enterprise data is the worst case and the actual case

Figure 1. Published text-to-SQL accuracy by benchmark realism. Figures are point-in-time; leaderboards move. See references [1] through [5].

Five conditions drive the collapse, and only one of them gets better with a larger model.

- **Schema scale.** Academic benchmark databases average tens of columns. Enterprise environments carry thousands of columns across hundreds of tables, plus deprecated copies, staging clones, and vendor-managed tables nobody owns. Selecting the right tables becomes a combinatorial retrieval problem before any SQL is written.
- **Encoded and dirty values.** Status codes, sentinel dates such as 9999-12-31, soft-delete flags, mixed currencies, and free-text fields that were never constrained. A query can be structurally correct and still return a wrong number because the model did not know that status 7 means voided.

- **Definitions that live outside the database.** Recognized revenue, active customer, billable hour, on-time delivery, and fully loaded labor cost are policy decisions, not columns. They live in accounting memos, spreadsheets, and the heads of three people. Nothing in the schema encodes them.
- **Dialect and platform variance.** The same logical question compiles differently against Snowflake, BigQuery, Athena, Redshift, and Postgres. Window function support, date handling, and type coercion all differ.
- **No training signal.** The BEAVER authors make this point directly: private data warehouses are not in any public training corpus, so a model cannot have learned the schema conventions or the query patterns of a specific enterprise.

The third condition is the important one. BIRD showed that supplying per-question domain hints moved GPT-4 from roughly 35 percent to roughly 55 percent execution accuracy. That gain is the measured value of institutional knowledge. It proves the bottleneck is knowledge that has not been written down, not raw reasoning capacity. If that knowledge is not captured somewhere durable, versioned, and governed, then every query re-derives it from scratch and every answer is a fresh roll of the dice.

---

§3 Six failure modes of direct access

3.1 Generation is not execution of business logic

A language model predicts tokens. It does not evaluate business rules. When a model produces SQL that implements a bonus calculation or a revenue recognition rule, it is producing the most statistically likely rendering of that rule given its training and the prompt. Sometimes that matches policy. There is no mechanism that guarantees it, and no mechanism that reports when it does not.

This distinction matters most where the rule is proprietary. A pay-for-performance bonus formula, a contract-specific margin calculation, or a regulator-specific allocation method has no correct answer in the training distribution. The model will still produce one.

3.2 Schema names do not carry business meaning

A column named `sales_amount` is a label, not a definition. In a real warehouse it might mean gross invoiced amount, net of discounts, net of returns, tax inclusive, tax exclusive, recognized revenue under the current policy, booked revenue at contract signature, or an intercompany figure that must be eliminated before reporting.

Each of those is a defensible reading. Only one is correct for the question being asked, and which one is correct depends on who is asking and why. A model given only the schema has to guess, and it guesses confidently.

3.3 The silent wrong answer

This is the failure mode that should decide the architecture. The following errors all produce a query that runs cleanly and returns a number that looks reasonable.

01 **Join fan-out.** Joining a fact table to a many-to-many bridge without deduplication multiplies rows and inflates every sum. Revenue doubles. Nothing errors.

02 **Grain mismatch.** Summing a monthly snapshot column across daily rows, or summing a pre-aggregated measure at the wrong level.

03 **Missing mandatory filter.** Omitting a soft-delete flag, a test-account exclusion, or a fiscal calendar restriction.

04 **Average of averages.** Taking the mean of per-region averages rather than a weighted mean, which quietly reweights the entire result.

05 **Slowly changing dimension errors.** Joining to a type-2 dimension without effective-date predicates, so historical figures shift every time a record changes.

06 **Time zone and fiscal boundary drift.** Bucketing by UTC when the business reports on local fiscal periods, which moves revenue across period boundaries.

None of these are exotic. They are the ordinary hazards of analytical SQL, and they are precisely the hazards a data team spends years encoding into curated models so that nobody has to rediscover them. Direct access discards that work on every query.

3.4 Authorization by prompt is not authorization

An instruction in a system prompt that says to restrict results to the requester's own region is a suggestion to a probabilistic system. It is not an access control. It can be overridden by later context, by an injected instruction, or simply by the model reasoning its way past it.

Worse, direct access patterns typically use one service account with broad read privileges, because narrow per-user credentials are inconvenient to plumb through. That single decision collapses the entire authorization model. Every user inherits the union of all permissions, and the access log shows one identity issuing every query. Under HIPAA's minimum necessary standard, that arrangement is difficult to defend at all.

Authorization has to be enforced where the query executes, through row-level and column-level policies bound to a verified end-user identity, with masking on sensitive attributes and a deny-by-default posture on anything not explicitly exposed.

3.5 Prompt injection and excessive agency

Once a model can call tools, untrusted text becomes executable influence. OWASP ranks prompt injection first in its 2025 Top 10 for LLM Applications and lists excessive agency, which is granting a model more capability or permission than the task

requires, at number six [9].

In a data context the injection surface is larger than it first appears. Instructions can arrive through a free-text field in a source system, a customer-supplied comment ingested into the warehouse, a file name, or the description of an MCP tool. Documented incidents include exfiltration of private GitHub repository contents through an over-scoped MCP integration, extraction of message history through a malicious MCP server operating alongside a trusted one, and remote code execution in the MCP Inspector development tool tracked as CVE-2025-49596 [8].

The NSA guidance also identifies a governance failure that is easy to miss: once an MCP server has been approved, its capabilities and data access can change without triggering a new approval. The user is not prompted, and the change may go unnoticed. An integration that was reviewed as benign can quietly acquire access to sensitive resources [8].

3.6 No lineage, no reproducibility, no audit trail

Ask a direct-access system the same question twice and you may get two different queries and two different numbers. Both may be plausible. Neither is reproducible, because the derivation was invented at request time and discarded.

That is a compliance problem before it is an analytics problem. A number that feeds financial reporting needs a documented derivation, an owner, a version, and evidence that the control operated. A privacy program needs to know which identity saw which rows and when. A direct-access architecture produces a chat transcript, which is not evidence.

§4 Why MCP is not governance

This section is not a criticism of MCP. Rayson builds on MCP, and it is the correct integration standard for this class of system. The point is narrower: MCP solves interoperability, and interoperability is not trust. Reading it as a governance layer is a category error.

CONCERN	WHAT MCP DEFINES	WHAT IS LEFT TO THE IMPLEMENTER
Transport and message format	Standard JSON-RPC messaging, capability negotiation, tool discovery and description	Nothing significant
Identity	Nothing binding a session to a verified identity	All of it. Authentication is optional in the protocol
Authorization	References OAuth 2.1 patterns for the connection	Role-based access control, per-tool scoping, row and column policy. RBAC is not exchanged at instantiation
Business semantics	Nothing	Metric definitions, join paths, grain, glossary, allowed aggregations
Query validation	Nothing	Static checks, fan-out detection, cost bounds, DML prohibition
Audit	Basic logging guidance only	Traceable action sequences, parameter capture, anomaly detection, retention
Approval durability	Consent at connection time in most clients	Re-review when a server's capabilities or data access change
Determinism	Nothing	Everything. Identical prompts can produce divergent behavior across implementations

Figure 2. MCP scope versus enterprise governance requirements. Protocol behavior summarized from the MCP specification and the NSA Cybersecurity Information Sheet on MCP [7][8].

The NSA assessment is worth reading in full, and its framing is useful for internal risk conversations. It notes that MCP inverts the familiar client-server pattern by having servers query and act on behalf of clients, which creates attack paths that existing defenses were not designed to trace. It notes that many implementations omit authentication entirely and that those including it often lack role-based enforcement,

producing what the document calls ambiguous and untraceable access to data. It notes that audit requirements are left to implementers, so many deployments log nothing or log only minimal metadata [8].

The practical reading for an enterprise architect is straightforward. MCP is plumbing, and good plumbing. Plumbing does not decide who is allowed in the building, what the numbers mean, or what happened last Tuesday.

§5 Regulatory and liability context

Governance obligations attach to the enterprise, not to the model provider. The following are the frameworks most likely to be cited during a review of conversational analytics. This is a summary for planning purposes and not legal advice; specific applicability should be confirmed with counsel.

SOX SECTION 404	Management asserts on the effectiveness of internal control over financial reporting. Any metric that feeds financial reporting needs a reproducible derivation, a defined owner, change control, and evidence that the control operated as described.
HIPAA	The minimum necessary standard limits use and disclosure of protected health information to what is required for the purpose. A shared service account with broad table access is hard to reconcile with that requirement.

GDPR	Article 5(2) places the accountability burden on the controller to demonstrate compliance. Article 15 grants access rights that require knowing what data was processed. Article 22 constrains solely automated decisions with significant effects.
EU AI ACT	In force since August 2024 with staggered application. The Digital Omnibus on AI, agreed in May 2026 and entering into force in July 2026, deferred stand-alone Annex III high-risk obligations to 2 December 2027 and Annex I embedded high-risk obligations to 2 August 2028. Most Article 50 transparency obligations remain scheduled for 2 August 2026. Treat the deferral as additional preparation time, not as a withdrawal of the requirements [10][11].
NIST AI RMF 1.0 AND AI 600-1	Voluntary but widely used as common vocabulary. The Generative AI Profile names twelve generative-AI risks, including confabulation, and organizes suggested actions across the Govern, Map, Measure, and Manage functions [12][13].
ISO/IEC 42001:2023	The first certifiable AI management system standard. Relevant when a customer or insurer wants third-party assurance rather than self-attestation [14].
OWASP TOP 10 FOR LLM APPLICATIONS 2025	The practical security checklist. Prompt injection, sensitive information disclosure, improper output handling, excessive agency, and misinformation all apply directly to data-querying agents [9].

5.1 Liability follows the operator

In Moffatt v. Air Canada, 2024 BCCRT 149, the British Columbia Civil Resolution Tribunal held the airline liable for negligent misrepresentation after its chatbot gave a customer incorrect information about bereavement fares. Air Canada argued the chatbot should be treated as a separate entity responsible for its own output. The tribunal rejected that, finding the airline responsible for information on its own site regardless of whether a static page or a bot produced it, and that the customer was entitled to rely on it [15][16].

The dollar amount was trivial and a Canadian small-claims tribunal sets no precedent in the United States. The principle is what matters, and it is the cheapest possible way to learn it: the operator owns the output. Applied to internal analytics, an incorrect figure delivered to a lender, a regulator, an auditor, or a board is the enterprise's statement. "The model generated it" is not a defense anyone should plan to rely on.

§6 Recommended architecture

The design principle is a single inversion. The model should select from governed definitions rather than improvise a query against a schema it half understands.

Generation of the executable artifact belongs to deterministic software, not to the model.

LAYER	RESPONSIBILITY
User	Asks a question in natural language. Receives an answer, the definition used, and the ability to see how it was derived
Language model	Intent recognition, entity and metric resolution, disambiguation, narration of results. Frontier or open-weight, interchangeable
Governed intelligence layer	Semantic contract, glossary and idioms, deterministic compilation, pre-execution validation, policy enforcement, lineage capture, confidence and refusal, versioned memory
Data platform	Storage and execution. Row-level and column-level security enforced natively. Query logs retained

Figure 3. Reference architecture. The model never holds credentials to the data platform and never emits the executed query directly.

Nine capabilities define the layer. Each is testable, which matters when evaluating vendors or building internally.

01 **Semantic contract.** A versioned, machine-readable definition of entities, metrics, dimensions, grain, legal join paths, and permitted aggregations. This is the artifact that turns institutional knowledge into something a system can enforce.

02 **Glossary and idiom capture.** The vocabulary the business actually uses. When someone asks for last quarter's numbers or utilization, the layer needs to know which fiscal calendar and which of four utilization definitions applies. Idioms should be mined from real usage and promoted deliberately, not guessed.

03 **Deterministic compilation.** The model resolves the question to governed definitions. The layer compiles those definitions to SQL. Identical resolved intent produces byte-identical SQL, every time, in the correct dialect for the target platform.

04 **Validation before execution.** Static checks on the compiled query: join path legality, grain consistency, fan-out detection, aggregation legality against the metric definition, dialect conformance, cost and row bounds, and a hard prohibition on DML and DDL. A query that fails validation is not executed and the failure is reported.

05 **Policy enforcement at execution.** The end-user identity is carried through, not collapsed into a service account. Row-level and column-level security are enforced by the platform. Sensitive attributes are masked. Anything not explicitly exposed is denied.

06 **Lineage and audit.** Every request records the question, the resolved semantics, the semantic contract version, the compiled SQL, the executing identity, the row count, and the timestamp. That record is the audit evidence, and it should be exportable to a SIEM and retained on the same schedule as other financial and access logs.

07 **Confidence and refusal.** The layer must be able to decline. If a question cannot be answered from governed definitions, the correct response is to say so and name what is missing. A system that always answers has no way to signal that it is guessing, which is the single most valuable signal it can send.

08 **Versioned memory with promotion states.** Validated reasoning and newly learned definitions should be reusable, but not automatically authoritative. A promotion pipeline that moves knowledge from candidate to provisional to canonical, with human approval at the canonical gate and an explicit deprecated state, keeps learning from becoming drift.

09 **Model agnosticism.** Semantics, policy, and audit belong to the enterprise. If replacing the underlying model requires rebuilding governance, governance was implemented in the wrong place.

6.1 Where senti fits

senti is Rayson's implementation of this pattern. It sits between the model and the customer's warehouse, holds the semantic contract, compiles and validates SQL before execution, enforces row-level and column-level policy against the caller's identity, records lineage per request, and exposes governed definitions over MCP so any compliant client can consume them. It runs against Athena, Snowflake, Redshift, Postgres, and DuckDB, and it treats the model as replaceable.

The pattern matters more than the product. An organization that builds the equivalent internally and can answer the evaluation questions in Section 9 is in a defensible position. An organization that wires a model to a warehouse and relies on user judgment is not, regardless of which model it chose.

§7 Mapping controls to frameworks

The table below is intended for use in a control narrative or a vendor assessment. It maps a governance obligation to what direct access supplies and what the layer has to supply instead.

OBLIGATION	DIRECT LLM ACCESS PROVIDES	GOVERNED LAYER MUST PROVIDE
Consistent metric definitions (SOX 404, ISO 42001)	Per-query interpretation of column names	Versioned semantic contract with change control and named owners
Least privilege (HIPAA minimum necessary, GDPR Art. 5)	Typically one broad service account	End-user identity passthrough with row and column policy enforced at execution
Auditability (SOX 404, GDPR Art. 5(2))	Chat transcript	Per-request lineage record exportable to a SIEM and retained on policy
Reproducibility	Non-deterministic generation	Deterministic compilation from governed definitions
Injection resistance (OWASP LLM01)	Model judgment	Untrusted content treated as data, validation gate before execution, no DML path
Bounded agency (OWASP LLM06)	Whatever the tools expose	Explicit allowlist of operations, deny by default, re-approval on capability change
Misinformation control (OWASP LLM09, NIST AI 600-1)	Confident prose	Confidence scoring, refusal path, definition shown alongside the answer
Transparency to the user (EU AI Act Art. 50)	A number	The number, the definition applied, and the derivation on request

Figure 4. Control mapping. Framework references are indicative and should be confirmed against the applicable version and jurisdiction.

§8 The verification asymmetry between

developers and business users

Engineers evaluate these systems and conclude they work. That conclusion is real, and it does not generalize.

An engineer reads the generated SQL, checks the join path, notices a missing filter, glances at the row count, and sanity-checks the result against a total they already know. Verification is nearly free because they already hold the schema in their head. When the model is wrong, they catch it, correct the prompt, and remember the system as useful.

A business user receives a number and a sentence of explanation. They have no way to inspect the derivation and no independent total to compare against. Fluent, well-structured output reads as competence, and that is a documented human tendency rather than a personal failing. There is also a timing problem: the wrong number is not caught at the point of use. It is caught in a board meeting, a lender review, or an audit, weeks later, if it is caught at all.

Governance that depends on the reader's expertise is not governance. It is a transfer of risk to the person least equipped to absorb it, and it fails exactly when the organization scales access beyond the pilot group. *Design for the user who cannot check the work,* because within a year that is most of the users.

§9 Evaluation criteria for a governed layer

Whether the layer is bought or built, these questions separate governance from a demo. Each one should have a demonstrable answer, not a roadmap answer.

01	Ask the same question ten times. Is the generated SQL byte-identical each time?
02	Where is the metric definition stored, who owns it, and how is a change reviewed and versioned?
03	Does the query execute as the end user or as a service account? Show the platform-side access log.
04	Ask a question the layer cannot answer from governed definitions. Does it refuse and explain, or does it guess?
05	Introduce a deliberate join fan-out. Does validation catch it before execution?
06	Attempt an update or delete through the natural language interface. Is the path structurally absent?
07	Plant an injected instruction in a free-text data value. Does it influence behavior?
08	Produce the lineage record for a single answer. Does it include identity, contract version, compiled SQL, and row count?
09	Change a row-level policy at the platform. Does the answer change immediately without a redeploy?
10	Swap the underlying model. What breaks, and how long does it take?

11	Run an adversarial evaluation set of 100 questions with known correct answers, written by the business, not by the vendor. Report accuracy and refusal rate separately.
12	Show a case where the layer was wrong. A vendor that cannot produce one has not looked.

> redline · twelve questions, twelve demonstrable answers.
> a roadmap answer is a no.

---

§10 Implementation sequence

The failure pattern in these projects is starting broad. A layer that covers one domain correctly is worth more than a layer that covers everything approximately, because the first one earns trust and the second one loses it permanently on the first wrong number in front of an executive.

PHASE 0

Inventory and scope.

Identify the questions that actually get asked, the metrics that feed external reporting, and the data domains carrying regulatory exposure. Pick one domain with high question volume and low regulatory blast radius.

> ✓ Gate · a written list of the 30 questions the pilot must answer correctly.

PHASE 1

Semantic contract for one domain.

Define entities, metrics, grain, legal join paths, and permitted aggregations. This is where the real work is, and it is mostly interviews rather than engineering.

> ✓ Gate · two people who disagreed about a definition now agree, in writing.

PHASE 2

Validation and policy enforcement.

Stand up deterministic compilation, the pre-execution validation gate, identity passthrough, and row and column policy. Nothing goes to a business user before this phase completes.

> ✓ Gate · the adversarial checks in Section 9 pass.

PHASE 3

Controlled pilot.

Ten to twenty business users, the domain from Phase 0, and an evaluation set with known answers. Track accuracy, refusal rate, and time to answer. Review every refusal, because refusals are the most informative output the system produces.

> ✓ Gate · accuracy on the evaluation set exceeds the analyst baseline and refusals are explainable.

PHASE 4

Audit integration and expansion.

Wire lineage into the SIEM, agree retention with audit and legal, document the control narrative, then add the next domain. Repeat Phase 1 per domain. Do not skip back to broad coverage because the pilot went well.

§11 Objections worth taking seriously

“Models keep improving. This is a temporary problem.”

Partly right and mostly beside the point. Model improvement will raise accuracy on questions where the answer is inferable from the schema. It will not encode an accounting policy that exists only in a memo, it will not enforce row-level security, and it will not generate an audit record. Those are architectural properties, not capability properties. A better model reduces one failure mode and leaves five.

“We will put the schema and a glossary in the prompt.”

This is a reasonable first experiment and it does help. It is not governance. Context is advisory rather than enforced, it does not survive a conflicting instruction, it has no version history, it degrades as the context grows, and it produces no audit trail. It also does not scale past a few hundred tables.

“We already have a semantic model in our BI tool.”

Then the hardest part is partly done and should be reused rather than duplicated. The gaps are usually pre-execution validation, agent-facing access to those definitions, per-request lineage, and a refusal path. Duplicating definitions in a second place is worse than either option, because the two copies drift and nobody knows which is authoritative.

“This adds latency and cost.”

It adds a compilation and validation step, and caching plus pre-aggregation typically offsets it. The comparison that matters is not milliseconds against milliseconds. It is the cost of the layer against the cost of one wrong number in a covenant calculation, a regulatory filing, or a board deck, plus the cost of the trust that does not come back afterward.

“Our users are technical enough to check the SQL.”

Today, with the pilot group. The entire value proposition of conversational analytics is that access widens to people who do not write SQL. Designing for the current audience guarantees a rebuild.

---

§12 Conclusion

Direct model access to enterprise data optimizes for the wrong thing. It optimizes for time to first answer and pays for it in accuracy, authorization, and auditability, which are the three properties that determine whether an answer can be used for anything that matters.

The benchmark record shows accuracy falling from the high eighties on clean academic schemas to roughly ten to twenty percent on real enterprise warehouses. The security guidance from the NSA shows that MCP deliberately leaves identity, authorization, and audit to the implementer. The liability record shows that the operator owns the output regardless of which system produced it.

A governed intelligence layer addresses all three. It encodes business meaning once and versions it. It compiles queries deterministically and validates them before execution. It enforces authorization against a real identity. It records lineage that an auditor can read. It can refuse. And because semantics, policy, and audit live outside the model, the enterprise can adopt whatever frontier or open-weight model comes next without rebuilding its governance.

The question to ask before approving conversational access is not whether the model is good. It is whether the organization can *reproduce, explain, and defend* any number the system produces. If the answer is no, the model quality is not the variable that needs attention.

NEXT STEP

Bring us the question you can’t currently defend.

Two ways to start, both of them short.

A SCOPED PROOF

One metric you cannot currently defend, run against your own warehouse. You keep the trace: the question, the SQL, the result, and the record.

A GOVERNANCE REVIEW

One hour. Your architecture walked against the twelve questions in §9, with a written note on where the evidence is thin.

References

ALL LINKS VERIFIED JULY 2026

[1] Yu, T. et al. Spider 1.0: Yale Semantic Parsing and Text-to-SQL Challenge. yale-lily.github.io/spider

[2] Li, J. et al. BIRD: Can LLM Already Serve as a Database Interface? NeurIPS 2023. Benchmark and leaderboard. bird-bench.github.io

[3] Lei, F. et al. Spider 2.0: Evaluating Language Models on Real-World Enterprise Text-to-SQL Workflows. ICLR 2025. arxiv.org/abs/2411.07763

[4] Spider 2.0 project site and leaderboard. spider2-sql.github.io

[5] Chen, P. B. et al. BEAVER: An Enterprise Benchmark for Text-to-SQL. arxiv.org/abs/2409.02038

[6] Model Context Protocol specification. modelcontextprotocol.io/specification

[7] Model Context Protocol, Security Best Practices. modelcontextprotocol.io/docs/tutorials/security

[8] National Security Agency, Artificial Intelligence Security Center. Model Context Protocol (MCP): Security Design Considerations for AI-Driven Automation. Cybersecurity Information Sheet, May 2026. media.defense.gov · CSI_MCP_SECURITY.PDF

[9] OWASP GenAI Security Project. OWASP Top 10 for Large Language Model Applications 2025. owasp.org · top-10-for-llm-applications

[10] Regulation (EU) 2024/1689 (Artificial Intelligence Act), consolidated text. eur-lex.europa.eu/eli/reg/2024/1689/oj

[11] Gibson Dunn. EU AI Act Omnibus Agreement: Postponed High-Risk Deadlines and Other Key Changes, May 2026. gibsondunn.com · eu-ai-act-omnibus-agreement

[12] NIST. Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1. nist.gov/itl/ai-risk-management-framework

[13] NIST. AI RMF Generative Artificial Intelligence Profile, NIST AI 600-1, July 2024. nvlpubs.nist.gov · NIST.AI.600-1.pdf

[14] ISO/IEC 42001:2023, Information technology, Artificial intelligence, Management system. iso.org/standard/42001

[15] Moffatt v. Air Canada, 2024 BCCRT 149. canlii.ca/t/k2spq

[16] American Bar Association. BC Tribunal Confirms Companies Remain Liable for Information Provided by AI Chatbot, February 2024. americanbar.org · business-law-today

[17] Invariant Labs. GitHub MCP Exploited: Accessing Private Repositories via MCP, 2025. invariantlabs.ai/blog/mcp-github-vulnerability

[18] Invariant Labs. WhatsApp MCP Exploited: Exfiltrating Message History via MCP, 2025. invariantlabs.ai/blog/whatsapp-mcp-exploited

[19] Oligo Security. Critical RCE Vulnerability in Anthropic MCP Inspector, CVE-2025-49596, 2025. oligo.security/blog · cve-2025-49596

[20] Microsoft. Protecting Against Indirect Prompt Injection Attacks in MCP, 2025. developer.microsoft.com/blog · indirect-injection-attacks-mcp

[21] Zhao, S. et al. Mind Your Server: A Systematic Study of Parasitic Toolchain Attacks on the MCP Ecosystem, 2025. arxiv.org/abs/2509.06572

[22] U.S. Department of Health and Human Services. HIPAA Minimum Necessary Requirement. hhs.gov/hipaa · minimum-necessary-requirement

[23] U.S. Securities and Exchange Commission. Management's Report on Internal Control Over Financial Reporting, Final Rule 33-8238. sec.gov/rules/final/33-8238.htm

[24] Anthropic. Model Context Protocol documentation and tool use guidance. docs.claude.com · agents-and-tools/mcp

[25] Australian Signals Directorate, ACSC. Careful Adoption of Agentic AI Services, 2026. cyber.gov.au · careful-adoption-of-agentic-ai-services

---