# Tribunal Session Summary

Session: conclave-20260604-072747-is-there-a-material-5b9c

Depth: T6

Advocates: 6 | Judges: 5 | Cost: $8.9198 | Time: 28m 23s

**THE PROMPT:** Is there a material difference in supported use cases between the following: using Claude code with an MCP to Snowflake vs just using Snowflake Cortex?

**THE RULING: Yes, there is a material difference, but it is strictly non-functional: the distinction lies in inference location (governance boundary) and default user persona, not in functional capability.**

# The Prompt

Is there a material difference in supported use cases between the following: using Claude code with an MCP to Snowflake vs just using Snowflake Cortex?

# The Ruling

**Yes, there is a material difference, but it is strictly non-functional: the distinction lies in inference location (governance boundary) and default user persona, not in functional capability.**

The functional gap between the two approaches has largely collapsed. Both systems now support cross-system orchestration (e.g., integrating with Jira, Slack, Salesforce) and advanced SQL generation. Snowflake Cortex Agents now natively support MCP connectors, allowing them to perform external actions previously attributed only to external Claude clients. The enduring material difference is **where the LLM reasoning occurs** and **who the primary user is**. Snowflake Cortex runs inference on Snowflake-hosted models inside the perimeter, satisfying strict "no external AI" compliance policies. Claude Code with MCP runs inference on Anthropic’s external infrastructure, making it unsuitable for those specific compliance tiers but preferable for developers needing deep IDE integration and local file manipulation.

# Core Analysis

### The Collapse of the Functional Capability Gap

The deliberation began with a consensus among advocates (DeepSeek V4 Pro, Grok 4, Gemini 3.1 Pro, GPT-5) that a material capability gap existed: Claude + MCP could orchestrate multi-system workflows (Snowflake + Jira + Slack), while Snowflake Cortex was viewed as a "walled garden" limited to internal SQL functions.

This binary was dismantled by primary evidence introduced in the later rounds (specifically by Claude Opus 4.8 and corroborated by others). Snowflake Cortex Agents (and the related Snowflake Intelligence/CoWork surface) now natively support **MCP connectors**. This allows Cortex to discover and invoke tools hosted by external providers like Atlassian (Jira), Salesforce, and GitHub directly from within the Snowflake perimeter.

- **The Mechanism:** Cortex Agents execute their own orchestration loop (`tools/list` and `tools/call`) inside Snowflake. When configured with an `mcp_servers` list (e.g., `jira_mcp_server`), the agent can autonomously create Jira tickets or update Salesforce records without relying on an external reasoning host.
- **The Result:** The "action surface" is no longer a differentiator. Both architectures can now execute end-to-end workflows that span Snowflake data and external SaaS tools.

## The Primary Distinction: Inference Location and Compliance

With functional capabilities converged, the material difference shifts to the **trust boundary**—specifically, where the LLM inference happens.

- **Snowflake Cortex (In-Perimeter):** Cortex can run the entire reasoning loop on Snowflake-hosted models (e.g., Claude Sonnet 4.6, Mistral, Llama). The model's "brain" and the tool execution happen inside the Snowflake account. This creates a **zero-egress inference** option.
- **Claude + MCP (External Inference):** Even with a Snowflake-managed MCP server that keeps SQL execution inside Snowflake, the *reasoning*—the LLM processing the prompt and tool results—occurs on Anthropic's external infrastructure.

**Why this matters:** This is a hard constraint for regulated industries. Organizations with policies prohibiting external LLM inference (even if only metadata or aggregates leave the perimeter) must use Cortex. For these entities, the difference is not about preference but about legal and contractual viability.

## The Secondary Distinction: Persona and Governance Surface

The second material difference is the default user experience and operational model.

- **Cortex (Governed Business User):** Cortex surfaces (Analyst, Agents, Snowsight) are designed for business users. They inherit Snowflake's native Role-Based Access Control (RBAC), masking policies, and audit trails. The "entry point" is a governed UI where data access is strictly scoped by roles.
- **Claude + MCP (Developer/Builder):** This approach is optimized for engineers. It operates as a CLI or desktop agent (e.g., VS Code extension) with access to the local file system, Git repositories, and arbitrary external APIs. While it can connect to Snowflake, the governance model relies on the developer's external credentials and the managed MCP server's configuration, rather than a native Snowflake RBAC inheritance.

**Why this matters:** A finance analyst asking "Why did revenue drop?" is best served by Cortex's governed Q&A interface. A data engineer building a pipeline that queries Snowflake, generates a local Python script, and commits it to GitHub requires the developer-centric surface of Claude Code.

## The "Scale" Myth: Context Window vs. Push-Down Compute

A significant point of contention was whether Claude + MCP could handle production-scale data. Gemini 3.1 Pro initially claimed MCP "fundamentally breaks" past 10,000–50,000 rows due to context window limits. This was refuted by the architecture of the Snowflake-managed MCP server.

- **The Reality:** MCP is an orchestration protocol, not a data pipe. The Snowflake MCP server enforces a **250 KB truncation limit** on SQL execution tool responses. This forces a "push-down" design pattern: the LLM generates aggregate SQL (e.g., `SELECT COUNT(*) ...` or `SELECT SNOWFLAKE.CORTEX.SENTIMENT(...) FROM massive_table`), Snowflake executes it at scale on distributed compute, and only the summary result returns to the LLM.
- **The Implication:** Claude + MCP can orchestrate batch jobs over billions of rows; it simply cannot *ingest* the raw rows. This makes it structurally capable of production-scale analytics, provided the workflow is designed to push computation to Snowflake rather than pull data to the LLM.

### The "Builder vs. Serve" Nuance

Claude Opus 4.8 initially framed the difference as "Claude builds, Cortex serves." While this evolved, a kernel of truth remains regarding local development.

- **Cortex Code:** Snowflake has released Cortex Code (CoCo), a Snowflake-native CLI/agent for building pipelines and apps. It supports MCP extensibility, narrowing the gap with external Claude Code.
- **Local State:** However, external Claude Code retains an advantage in manipulating the developer's local state—editing files on the local disk, managing uncommitted Git changes, and interacting with the local OS. Cortex, being cloud-native, cannot directly manipulate a developer's local file system in the same way.

# How We Got Here

## Scorecard

| Judge | Claude Opus 4.8 | Magistral Medium | DeepSeek V4 Pro | Grok 4 | Gemini 3.1 Pro | GPT-5 | Winner |
|---|---|---|---|---|---|---|---|
| **GPT-OSS 120B** | ACCEPT | REJECT | REJECT | REJECT | REJECT | REJECT | Claude Opus 4.8 |
| **Zhipu GLM-4.7** | ACCEPT | REJECT | ADOPT | ADOPT | REJECT | ADOPT | Claude Opus 4.8 |
| **Qwen 3.5 397B** | ACCEPT | REJECT | ADOPT | ADOPT | REJECT | ADOPT | Claude Opus 4.8 |
| **DeepCogito v2.1 671B** | ACCEPT | REJECT | ADOPT | ADOPT | REJECT | ACCEPT | Claude Opus 4.8 |
| **Mistral Large 3** | ADOPT | REJECT | ADOPT | ADOPT | ADOPT | ADOPT | Claude Opus 4.8 |

## Convergence Assessment

The convergence was **epistemic** (evidence-driven), not affective. The shift in position—from "Cortex cannot do external actions" to "Cortex can do external actions via MCP connectors"—was triggered by the introduction of specific primary documentation (Snowflake Cortex Agents & MCP Connectors docs) in the later rounds. Advocates did not change their minds to align with a majority; they changed their minds because the factual premise of their arguments (Cortex's limited capabilities) was proven wrong.

## Closing Statements

- **Claude Opus 4.8:** Conceded the "build vs. serve" framing and acknowledged that Cortex Agents now support native MCP connectors. Final position: The material difference is strictly **Inference Location** (Snowflake-hosted vs. external) and **Persona** (governed UI vs. developer CLI). Argued this is the only non-substitutable distinction for regulated workloads.
- **DeepSeek V4 Pro:** Abandoned the "fixed catalog" thesis entirely. Final position: The difference is the **Trust Boundary**. Cortex keeps LLM inference in-perimeter; Claude + MCP reasons externally. Acknowledged that cross-system orchestration is now supported natively by Cortex.
- **Grok 4:** Revised from "multi-system vs. native" to **Inference Residency**. Conceded that Cortex Agents can orchestrate externally without an external host. Final position: The choice reduces to whether the reasoning host must be inside Snowflake or if an external client is acceptable.
- **Gemini 3.1 Pro:** Issued a major revision, retracting claims of "legal prohibition" and "context window exhaustion." Final position: The functional capability gap has collapsed. The remaining differences are **Inference Boundary** (compliance) and **Client Environment** (local CLI vs. hosted UI).
- **GPT-5:** Refined the argument to focus on **Inference Location** and **Operational Ownership**. Conceded that Cortex Agents can natively execute SaaS actions. Final position: Cortex is preferred for in-perimeter reasoning; Claude + MCP is preferred for open-ended, cross-system development.
- **Magistral Medium:** Maintained a distinction based on "flexibility" and "orchestration scope" but failed to provide concrete evidence. Did not adequately incorporate the documentation regarding Cortex Agents' native MCP capabilities, resulting in a weak final position.

## Key Moments

- **The Capability Collapse (Rounds 6–7):** Claude Opus 4.8 surfaced primary documentation showing that Snowflake Cortex Agents natively support MCP connectors for Jira, Slack, and Salesforce. This directly contradicted the prevailing narrative (held by DeepSeek V4 Pro, Grok 4, and Gemini 3.1 Pro, and F) that Cortex was strictly an internal tool. This evidence forced a massive revision across the room, collapsing the "functional gap" argument.
- **The "Scale" Correction (Rounds 5–6):** Gemini 3.1 Pro asserted that MCP was "legally prohibited" for HIPAA use cases and would break at scale due to context windows. Multiple advocates (Claude Opus 4.8, GPT-5, DeepSeek V4 Pro) challenged this, pointing out that managed MCP keeps execution in-perimeter and uses push-down compute. Gemini 3.1 Pro conceded both points, retracting the "legal prohibition" language and the specific row limits.
- **The Governance Refinement (Rounds 2–5):** Several advocates (DeepSeek V4 Pro, Grok 4, GPT-5) initially claimed that data *always* left the Snowflake perimeter when using Claude + MCP. Challenges clarified that the Snowflake-managed MCP server keeps SQL execution inside the account; only the reasoning step and bounded results (≤250 KB) egress. This refined the governance argument from "data residency" to "inference location."

# Reality Checks

| Claim | Challenge | Result |
|---|---|---|
| **Gemini 3.1 Pro: "MCP is legally prohibited for HIPAA/SOC-2/PII use cases."** | **GPT-5/A:** Produce the regulatory text or BAA template proving this. Managed MCP + OAuth + enterprise BAAs allows governed usage. | **Full Concession.** Gemini 3.1 Pro retracted the word "prohibited," acknowledging that compliance is "context-dependent with contractual controls," not an architectural prohibition. |
| **Gemini 3.1 Pro: "MCP fundamentally breaks at scale... limited to 10,000–50,000 rows."** | **Claude Opus 4.8/F:** This is a straw man. MCP pushes computation to Snowflake; the LLM only receives aggregates. | **Full Concession.** Gemini 3.1 Pro admitted the "hard constraint" was a misrepresentation of the agentic design and revised the claim to a "design-pattern constraint." |
| **DeepSeek V4 Pro: "Cortex functions are a fixed, finite catalog... cannot be composed into novel, multi-step decision chains."** | **Claude Opus 4.8/D:** Cortex Agents provide multi-step planning and tool use. You are ignoring the agentic layer. | **Major Revision.** DeepSeek V4 Pro conceded that Cortex is not stateless and that Agents support multi-step orchestration, withdrawing the "fixed catalog" thesis. |
| **Grok 4: "Cortex executes using Claude Opus 4.7."** | **Claude Opus 4.8:** Verify the version number. Current docs list Opus 4.6/Sonnet 4.5. | **Concession.** Grok 4 acknowledged the version number was imprecise/incorrect. |

## Next Steps

| Next Step | Why It Matters | Confidence |
|---|---|---|
| **Verify Cortex Agents MCP Connector GA Status** | The "functional convergence" argument relies on Cortex Agents' native MCP support being generally available (GA) and stable. If this is still in preview/limited release, the capability gap may not be fully closed for all users. | High (Judges verified via primary docs, but Fresh Eyes flagged this as a key dependency) |
| **Benchmark Inference Latency (Snowflake-hosted vs. External)** | While inference location is a compliance differentiator, the performance impact (latency) of running models in-perimeter vs. external is unknown. This could be a deciding factor for non-regulated workloads. | Medium (No data provided in deliberation) |

| Next Step | Why It Matters | Confidence |
|---|---|---|
| **Quantify TCO of Managed MCP vs. Native Cortex** | The operational cost difference (Snowflake compute credits vs. Anthropic API tokens + MCP infra) was discussed but not quantified. A concrete cost model is needed for procurement decisions. | Medium (Advocates acknowledged the gap exists but provided no numbers) |

# Dissenting Opinions

No dissenting opinions were filed in this session.

# Glossary

| Term | Definition |
|---|---|
| **Cortex Agents** | A Snowflake feature that uses LLMs to plan and execute multi-step tasks across structured and unstructured data, capable of tool calling and reflection. |
| **Cortex Code (CoCo)** | A Snowflake-native CLI/agent for data engineering tasks, capable of building pipelines and apps, with support for MCP extensibility. |
| **Inference Location** | The physical location where the LLM processes the prompt and generates the response (e.g., inside Snowflake's perimeter vs. on Anthropic's external servers). |
| **MCP (Model Context Protocol)** | An open standard that connects AI assistants to local and remote tools (like databases, file systems, and APIs) via a standardized client-server architecture. |
| **Push-Down Compute** | An architectural pattern where computation (e.g., SQL aggregation) is executed on the data source (Snowflake) rather than moving raw data to the AI model for processing. |
| **RBAC (Role-Based Access Control)** | A method of regulating access to computer or network resources based on the roles of individual users within the enterprise. |
| **Snowflake-Hosted Models** | LLMs (e.g., Claude, Mistral, Llama) that run entirely within Snowflake's cloud infrastructure, ensuring data does not leave the perimeter for inference. |

# How The Conclave Works

The Conclave is a structured multi-model deliberation system. Instead of asking one AI model a question and trusting its answer, the Conclave convenes a panel of independent models, forces them to argue, and subjects their conclusions to judicial review. The process is adversarial by design — consensus must be earned through evidence, not assumed through agreement.

**Advocates** are independent AI models (e.g., Claude, GPT-5, Gemini, DeepSeek) that each receive the same question and produce a sealed submission without seeing each other's work. Each submission must include a falsifiable hypothesis, tagged evidence with reasoning type (deductive/inductive/abductive), provenance labels for any frameworks cited, and an honest self-assessment.

**Challenges** follow submissions. Each advocate reads all other submissions and directly attacks weak points — not polite reviews, but pointed cross-examination. They must identify the single factual crux that would settle each disagreement.

**Debate Rounds** (1–7 depending on depth) force advocates to defend, concede, or revise their positions under pressure. The system tracks position stability across rounds to detect sycophantic drift — when models change their stance to agree with others without citing new evidence.

**Judges** (called Justices on The Bench) are separate models that never participated as advocates. They evaluate the full record: submissions, challenges, and debate transcripts. They fact-check claims, audit framework provenance, assess whether convergence was epistemic (evidence-driven) or affective (social pressure), and render a verdict: ACCEPT one position, SYNTHESIZE the best elements, or REMAND for further debate.

**Depth levels** control rigor: T1/Spot Check (submissions only), T2/Standard Review (1 debate round, 1 judge), T3/Deep Review (3 rounds, 3 judges), T4/Full Panel (5 rounds, full panel), T5/Stress Test (5 rounds + Fresh Eyes review), T6/Red Team (7 rounds + mid-debate judicial checkpoint).

The system is deterministic code — it cannot be sycophantic. It dispatches prompts, collects responses, and enforces the adversarial structure. The models argue; the code referees.