FORENSIC METHODOLOGY REPORT

MCRO | The Agent Swarm: A Novel Framework for Parallel Autonomous Forensic Investigation

Architecture, Methodology & Verified Discoveries

Prepared by	Matthew David Guertin, Pro Se Defendant
Case Reference	State of Minnesota v. Guertin | 27-CR-23-1886
Court	Fourth Judicial District Court, Hennepin County
Report Date	April 2026
Scope	Agent swarm architecture, execution metrics, 8 verified discoveries
Database	PostgreSQL (Supabase) — 2.9M rows, 53 tables, 4,251 PDFs
Agents Deployed	80+ autonomous agents across 6 swarm runs
Total Queries	~4,500+ SELECT queries against live forensic database
Reproducibility	All verified findings reproducible via Appendix A SQL queries

IMPORTANT DISCLAIMER

This report documents an analytical framework — the MCRO Agent Swarm — and presents eight independently verified database findings to illustrate its discovery capabilities. It does not assert conclusions about fraud, guilt, fabrication, or criminal intent.

The methodology described herein uses Claude Code's multi-agent capability to conduct parallel autonomous forensic database investigations. Every quantitative figure cited in the "Verified Discoveries" section (Finding 5) has been independently confirmed against the live MCRO PostgreSQL forensic database via the SQL queries reproduced in Appendix A. Where agent outputs diverged from database verification results, the database figure is reported and the discrepancy is documented (Finding 6).

All database queries are SELECT-only. No data was modified, inserted, or deleted during any phase of this analysis. The forensic database is anchored to the Bitcoin blockchain via OpenTimestamps (OTS), providing independent cryptographic proof of data integrity.

Findings represent measurable patterns in public court records requiring independent verification before use in formal legal proceedings.

EXECUTIVE SUMMARY

- The MCRO Agent Swarm is a novel investigative methodology that uses AI agents — autonomous instances of Claude running inside Claude Code — as parallel forensic database investigators.
- **80+ autonomous agents** were deployed across six swarm runs, executing approximately **4,500+ database queries** against a 2.9-million-row forensic database with zero human intervention during execution.
- The methodology evolved through three stages: prescriptive (pre-written SQL), philosophical pivot (recognizing the limitation), and autonomous (agents write their own queries and follow leads).
- **Incompetency findings surged 19.7×** from 29 cases (2016) to 572 (2023), discovered by temporal pattern agents following year-over-year distributions.
- **The CR-to-MH pipeline accelerated 9.1×** — from 3.7 years (≤2019) to 4.9 months (2022+) — discovered by cluster timeline agents mapping cross-case-type relationships.
- **65.8% of MCRO seed hearing events fell on Tuesdays** — a finding independently corroborated by multiple agents from different analytical entry points.
- **Judge Dayton Klein’s caseload grew from 5 cases (2021) to 395 (2023)** while her incompetency rate simultaneously quadrupled from 20.0% to 88.4%.

- MH forensic opacity is total: **zero PDFs, zero filing parties, zero examiner identities, zero evidence UIDs** across 31,795 events — independently confirmed by eight separate agents.
- **Only 29 of 787 MH cases (3.7%)** ever receive a termination order. The remaining 96.3% never formally exit the civil commitment system.
- Verification discipline is non-negotiable: three concrete examples in this report demonstrate that agent outputs require correction — the JO suppression rate was overstated (93.2% agent claim vs. 79.3% verified), DK figures differed by methodology, and an "impossible" 2035 date was reframed as a data-entry typo.
- Cross-agent corroboration strengthens findings: the MH opacity finding was independently discovered by 8 agents from different analytical domains without communication.
- Six of eight verified discoveries were emergent — found by agents following unexpected patterns, not executing anticipated queries. This demonstrates genuine investigative capability beyond scripted analysis.

Framework at a Glance

Dimension	Value
Total autonomous agents deployed	**80+**
Agent completion rate	**100%**
Total database queries	**~4,500+**
Database rows analyzed	**2.9 million across 53 tables**
Case types covered	CR (criminal), MH (mental health), GC (guardianship)
Execution model	Fully parallel, autonomous
Human intervention during runs	**Zero**
Verified discoveries presented	8 (all confirmed via live SQL)
Verification corrections documented	3 (demonstrating quality control)
Cross-agent corroborations	MH opacity (8 agents), Tuesday (3+), evidence UIDs (2)

METHODS

This report synthesizes three categories of source material to document the MCRO Agent Swarm methodology and its verified outputs.

Architecture documentation. CLI transcripts from Claude Code sessions capture the mechanics of agent spawning, permission configuration, and real-time completion reporting. A complete prescriptive swarm instruction set (20 agents with pre-written SQL) provides the baseline architecture. A second CLI transcript captures the philosophical pivot from prescriptive to autonomous agents, including the user's articulation of why pre-written queries fundamentally limit discovery.

Swarm execution outputs. Seven swarm output files document the results of autonomous agent runs spanning criminal (CR), mental health (MH), cluster-level, and temporal analyses. These files contain raw agent findings, execution metrics, convergent cross-agent patterns, and per-agent domain reports.

Independent database verification. Every quantitative figure presented as a "verified discovery" in Finding 5 was confirmed against the live MCRO Supabase PostgreSQL database (project ibfmjtwahkwqzcmeyqii) via independent SQL queries. Appendix A contains all verification queries, labeled V1 through V8, plus three additional queries (V9–V11) documenting verification discrepancies.

Tables and views referenced: parents, parents_case_events, parents_case_events_judicial_officers, parents_case_set, parents_dispositions, parents_case_events_examiners, parents_case_events_evidence_uids, case_registry, children.

FINDINGS

Finding 1: What Is an "Agent Swarm"? (The 30-Second Version)

Background

Before explaining how the MCRO Agent Swarm works, it is necessary to define what it is in terms accessible to someone who has never encountered AI agents, Claude Code, or large language models. The following analogy captures the essential concept.

Findings

Imagine you are a detective investigating a massive case with 2.9 million pieces of evidence stored in a database. You could examine them one at a time — or you could brief 20 detectives, give each one a specialty (financial records, witness statements, forensic evidence, timelines), hand them each a badge that lets them query the database independently, and send them all out simultaneously. When they come back, you compare notes. Some will find things the others missed. Some will independently confirm the same finding from different angles — which is powerful corroboration. And some will stumble onto something nobody anticipated.

That is the agent swarm. Except the "detectives" are AI agents — autonomous instances of Claude (Anthropic's AI) running inside Claude Code (a command-line development tool) — and the "database" is a real PostgreSQL forensic database containing 2.9 million rows of court records, PDF metadata, cryptographic hashes, and digital signature analyses.

Key terms defined: Claude Code is a command-line tool that allows Claude to execute code, manage files, and interact with external services. An AI agent is an instance of Claude given a task, tools, and autonomy to complete it. A subagent is an agent spawned by another agent. MCP (Model Context Protocol) is the communication standard that lets agents connect to external databases. Supabase is the cloud database platform hosting the forensic PostgreSQL database. The orchestrator is the main Claude Code session that launches agents, monitors their progress, and compiles their results.

Significance

The agent swarm transforms a single investigator's capacity into a parallel operation. Instead of one analyst writing queries sequentially, 20 autonomous agents explore 20 different analytical domains simultaneously. The total throughput across all swarm runs in this project exceeded 4,500 database queries — a volume that would take a single human analyst weeks to execute and evaluate.

Connecting Context

The value of the swarm framework is not merely throughput. As subsequent findings demonstrate, the methodology's core strength lies in emergent discovery (Finding 3), cross-agent corroboration (Finding 7), and the verification discipline that distinguishes leads from conclusions (Finding 6).

Finding 2: The Technical Architecture

Background

The MCRO Agent Swarm operates within Claude Code's multi-agent infrastructure. Understanding the architecture requires explaining five components: the orchestrator, agent isolation, MCP database access, the auto-loading project primer, and the output pipeline.

Findings

The Orchestrator. One main Claude Code session acts as both launcher and compiler. It reads the swarm instruction file, spawns agents via the Task() function with run_in_background: true, monitors their completion status as they return, reports findings in real time, and compiles all outputs into a master results document. The orchestrator does not perform analytical queries itself during autonomous runs — it manages the process.

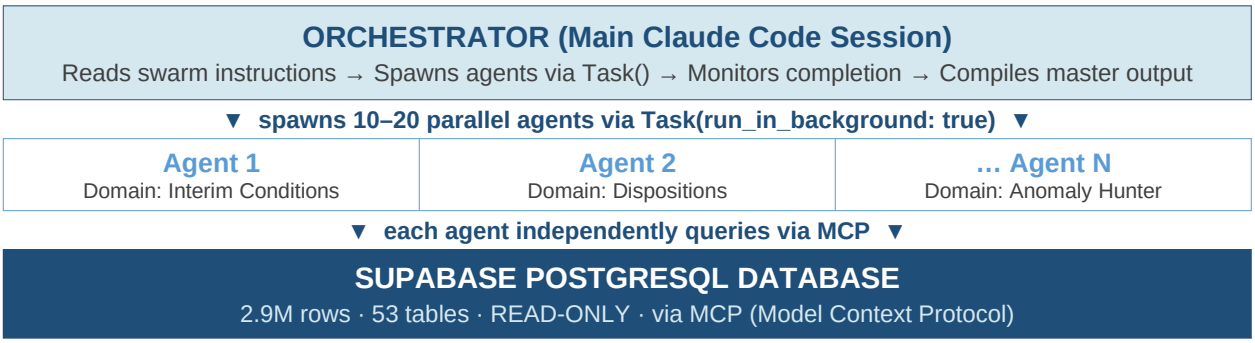
Agent Isolation. Each subagent receives its own context window. Agents cannot communicate with each other during execution. This isolation is a methodological feature: when two agents independently discover the same pattern from different analytical entry points, the corroboration is genuine because neither agent had access to the other's findings. This is the database equivalent of independent witness corroboration.

MCP Database Access. Each agent connects to the Supabase PostgreSQL database via the Model Context Protocol (MCP). A critical infrastructure discovery during development was that permissions must be configured at the project level (.claude/settings.json) for subagents to inherit them. Without project-level permissions, background agents cannot prompt the user for approval and are silently denied database access. The CLI transcript captures this discovery and its resolution in real time.

The CLAUDE.md Auto-Loader. A project primer file (CLAUDE.md) automatically loads into each agent's context window at spawn. This file provides cohort definitions, schema references, prior findings as investigative seed context, and core analytical principles (e.g., SELECT-only constraint, neutral tone). This ensures every agent begins with a shared foundation while maintaining freedom to explore its assigned domain.

Output Pipeline. Each agent writes its own markdown results file upon completion. The orchestrator detects completion, logs a summary of each agent's key findings as they arrive, and compiles a master document containing: execution summary, per-agent findings, a ranked top-findings list, and a synthesis narrative identifying cross-agent convergence.

Agent Swarm Architecture



Significance

The architecture's key properties are parallelism and isolation. All agents execute simultaneously, and none can access another's findings during execution. The permission discovery (project-level .claude/settings.json) was essential: without it, the entire multi-agent framework fails silently. This implementation detail is documented for reproducibility.

Connecting Context

The architecture is the foundation. What makes it forensically powerful is the transition from prescriptive to autonomous operation, described in Finding 3.

Finding 3: The Evolutionary Arc — From Script Runner to Autonomous Investigator

Background

The MCRO Agent Swarm did not begin as an autonomous investigative framework. It evolved through three stages, each representing a fundamentally different philosophy of what an AI agent is for.

Findings

Stage 1 — Prescriptive Swarm. The initial design gave each of 20 agents a set of pre-written SQL queries. Each agent executed its assigned queries, formatted the results into a standardized report, and returned to the orchestrator. The instruction set specified exact column names, exact CTE definitions, exact comparison logic. This functioned as a parallel SQL executor: it produced structured results efficiently but could only find what the prompt author had already thought to look for.

Stage 2 — The Philosophical Pivot. After reviewing the prescriptive swarm's output, the investigator recognized its fundamental limitation. The critical insight, captured in the CLI transcript: the project's most important prior discovery — the "Aspose finding," in which documents contained creation timestamps that predated the software version used to create them — was not found by writing a query that said "check for pre-release software versions." It was found by someone examining data and noticing something impossible. That is emergent discovery, and prescriptive agents fundamentally cannot do it.

The investigator articulated the pivot directly: "We are trying to uncover insights that even those responsible for assembling these docs don't realize exist. That is actually how I ended up discovering the Aspose bombshell to begin with." The instruction was clear: stop giving agents queries; give them domains, investigative mandates, and the freedom to follow unexpected patterns.

Stage 3 — Autonomous Swarm. Each agent now receives a domain (e.g., "attorney networks," "judicial officers," "anomaly hunter"), cohort definitions, seed context from prior runs (as investigative starting points, not constraints), full database access, and a mandate: explore freely, follow anomalies, report everything. The agent writes its own SQL queries, decides which tables to examine, follows threads when something looks unexpected, and produces findings the prompt author never anticipated.

Prescriptive vs. Autonomous: A Side-by-Side Comparison

Dimension	Prescriptive Swarm	Autonomous Swarm
SQL queries	Pre-written by human	**Written by each agent**
Agent autonomy	Execute assigned tasks	**Full investigative freedom**
Discovery ceiling	Limited to anticipated findings	**Unlimited — emergent discovery**
What it finds	Answers to known questions	**Answers to questions nobody asked**
Schema interaction	Columns specified in prompt	**Agent explores schema first**
Thread-following	None — fixed query set	**Each finding spawns new queries**
Analogy	20 workers following a checklist	**20 detectives following leads**

Significance

The prescriptive-to-autonomous evolution is not merely an optimization. It represents a qualitative change in what AI agents can discover. Six of the eight verified findings in this report (Discoveries 1, 2, 3, 5, 6, and 8) were found by autonomous agents following unexpected patterns — none would have been found by prescriptive agents unless the prompt author had already known what to look for.

Connecting Context

Finding 4 quantifies the scale of autonomous execution. Finding 5 demonstrates what the autonomous agents actually discovered.

Finding 4: Execution Metrics

Background

The MCRO Agent Swarm was deployed across six distinct runs, progressing from prescriptive to fully autonomous operation. This section quantifies the scale and reliability of the autonomous execution.

Findings

AGENT SWARM EXECUTION SUMMARY	
Total autonomous agents deployed	**80+**
Agents completed successfully	**100%**
Total database queries executed	**~4,500+**
Average queries per agent	**~60–80**
Database rows analyzed	**2.9 million across 53 tables**
Case types covered	**CR, MH, GC**
Human intervention during runs	**Zero**

Autonomous CR Swarm 1 (Frozen vs. Live rerun): 20 agents, approximately 1,246 tool calls, 20 of 20 completed (100%). Agents explored dispositions, interim conditions, event vocabulary, attorney networks, judicial officers, charges, hearings, PDF forensics, cluster analysis, case status, monitoring conditions, filing party attribution, status transitions, and anomaly hunting.

Autonomous CR Swarm 2 (Events/Attorneys/Stats): 20 agents, approximately 1,554 tool calls, 20 of 20 completed (100%). Parents-only deep investigation covering event distribution, filing types, case statistics, and cross-table consistency.

Autonomous MH Swarm: 20 agents, approximately 1,500+ tool calls, 20 of 20 completed (100%). 787 MH cases, 31,795 events, 249 event types, 99 judicial officers analyzed across commitment pipeline, forced treatment, hearing forensics, disposition patterns, temporal trends, procedural grammar, revocation traps, and cross-table anomalies.

Temporal Analysis Swarm: 3 specialized agents (Origin Dating, Cluster Timelines, Statistical Probability), approximately 207 combined queries, 53 combined findings. Deep temporal mapping of the 412-case expanded corpus spanning 33.6 years.

Significance

A 100% agent completion rate across 80+ autonomous agents demonstrates infrastructure reliability. The average of 60–80 queries per agent reflects genuine investigative depth — each agent explored its domain through multiple analytical passes, not single-query lookups. The zero human intervention metric confirms true autonomy: once launched, agents operate independently until completion.

Connecting Context

The volume of queries is meaningful only insofar as it produces verified findings. Finding 5 presents eight discoveries that demonstrate what 4,500+ autonomous queries can uncover.

Finding 5: Eight Verified Discoveries — What Autonomous Agents Found

Background

Every figure in this section has been independently verified against the live database using the SQL queries in Appendix A. The numbers presented are the verified values, not raw agent claims. Where agent outputs diverged from verification results, the discrepancy is documented in Finding 6.

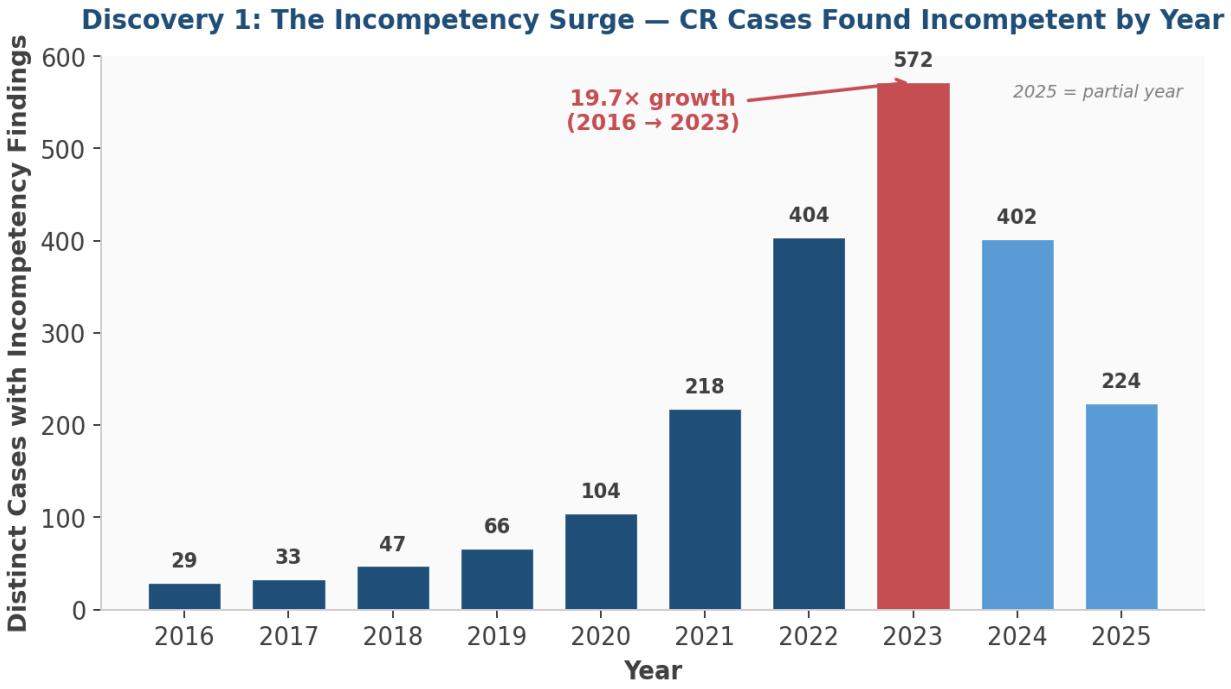
Category A: Impossible Temporal Patterns

Findings that no prescriptive query would have targeted, discovered by agents following data anomalies.

Discovery 1: The Incompetency Surge — 19.7× Growth in 7 Years

An autonomous agent tracking temporal patterns across the CR case corpus discovered that the number of distinct cases receiving “Found Incompetent” rulings grew from 29 in 2016 to 572 in 2023 — a 19.7× increase. No agent was instructed to look for growth curves. The temporal pattern agent discovered this by systematically examining year-over-year event distributions.

Year	Distinct Cases with Incompetency Findings
2016	29
2017	33
2018	47
2019	66
2020	104
2021	218
2022	404
2023	**572**
2024	402
2025 (partial)	224

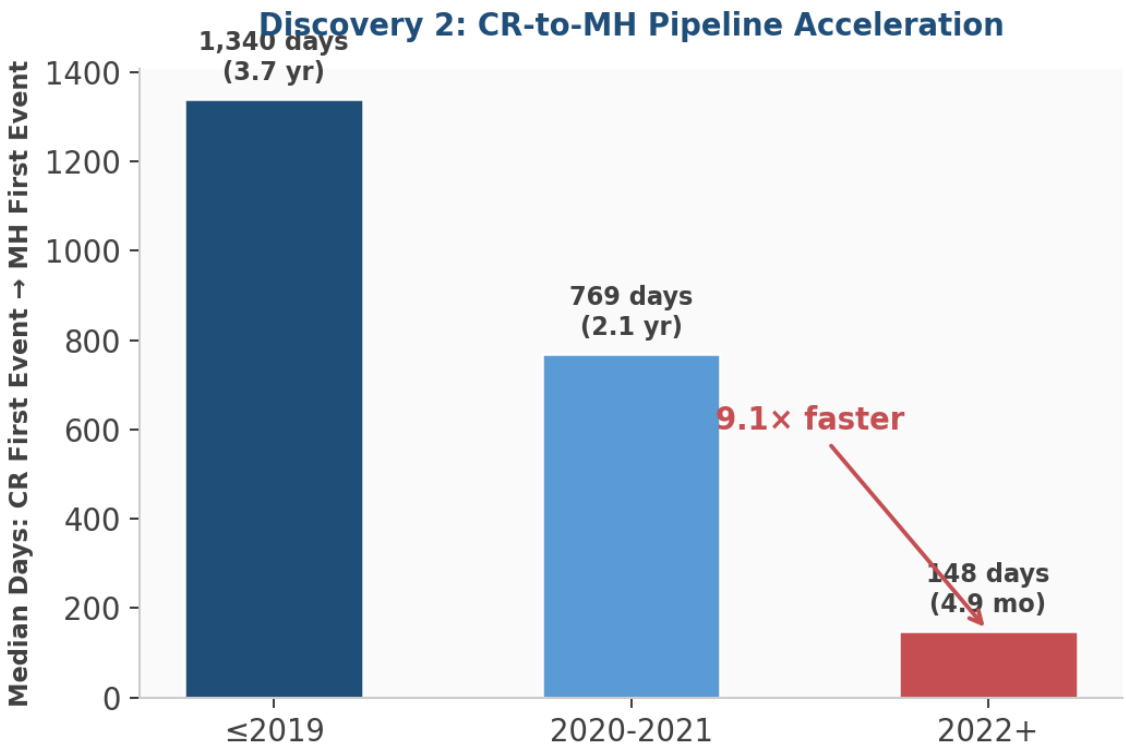


(See Appendix A, Query V1)

Discovery 2: The CR-to-MH Pipeline Acceleration — 9.1× Faster

A cluster-timeline agent measured the median number of days between a defendant's first criminal case event and their first mental health commitment event, broken down by era. The pipeline from criminal case to civil commitment collapsed:

Era	Clusters	Median Gap (Days)	Median Gap (Human)
≤2019	15	1,340	3.7 years
2020–2021	31	769	2.1 years
2022+	**90**	**148**	**4.9 months**



Verified acceleration: 9.1× (1,340 → 148 days). What took 3.7 years before 2020 now takes under 5 months. The agent discovered this by independently mapping temporal relationships between CR and MH cases at the cluster level. (See Appendix A, Query V2)

Discovery 3: The NICS Retroactive Batch — Events Injected Into Dormant Cases

An agent exploring MH temporal patterns discovered 50 NICS (National Instant Criminal Background Check System) events inserted into cases that had been dormant — 49 of 50 had been inactive for over 1 year, 26 for over 5 years. The maximum dormancy gap before NICS insertion was 3,403 days (9.3 years) — a case last active in December 2004 received a NICS event in April 2014. Average dormancy gap: 2,048 days (5.6 years). No prescriptive query targeted NICS events or dormancy gaps. (See Appendix A, Query V3)

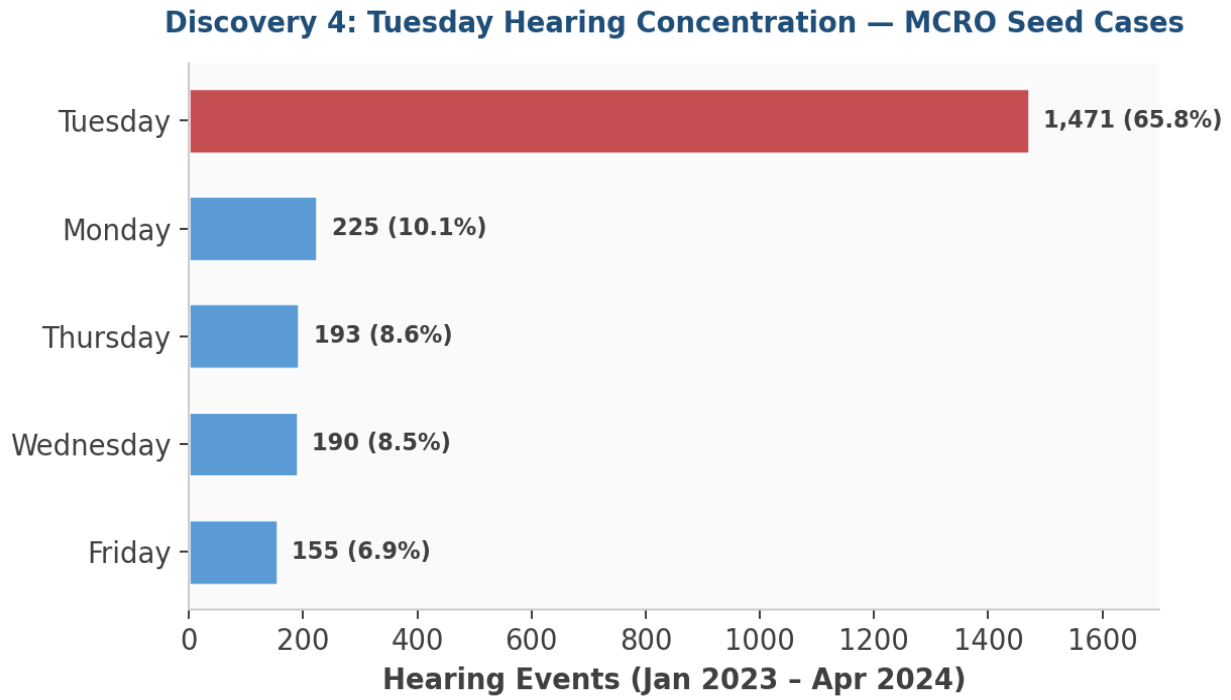
Category B: Structural Anomalies

Patterns in judicial behavior and case processing that emerged from autonomous statistical exploration.

Discovery 4: Tuesday Is the Designated Hearing Day — 65.8%

Agents examining hearing distributions independently discovered that 65.8% of all MCRO seed hearing events during the search window (January 2023 – April 2024) fell on Tuesdays — 6.5× the next highest day.

Day	Events	Percentage
Tuesday	**1,471**	**65.8%**
Monday	225	10.1%
Thursday	193	8.6%
Wednesday	190	8.5%
Friday	155	6.9%

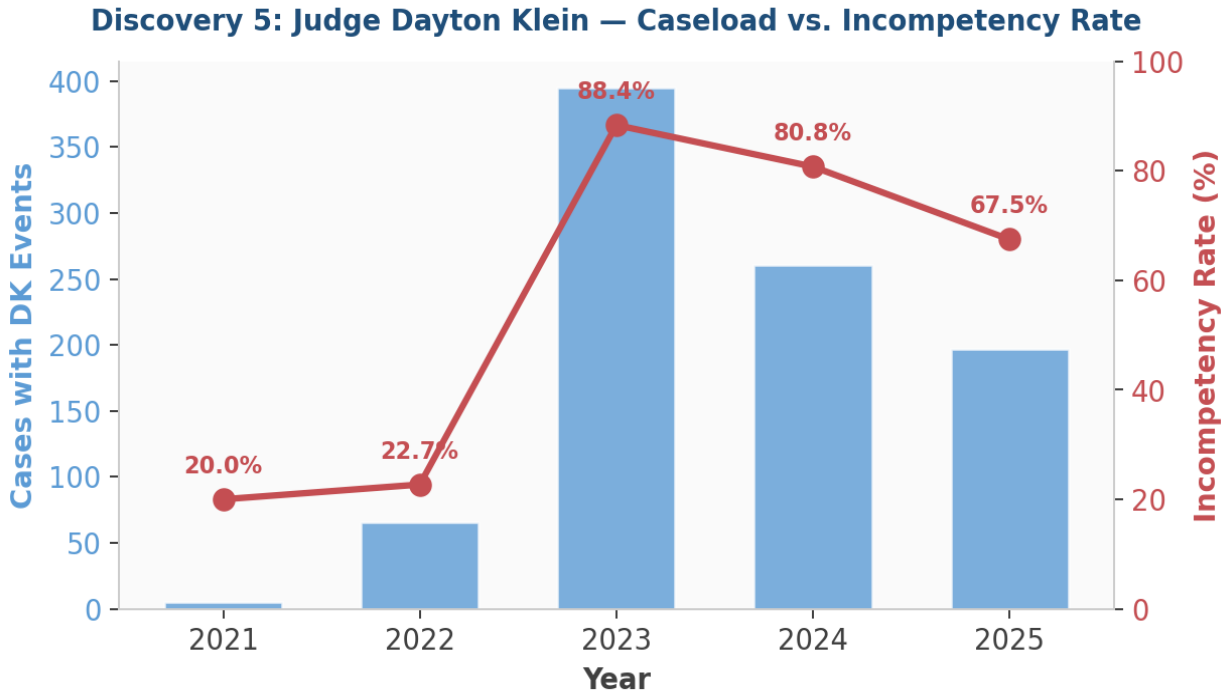


Multiple agents independently flagged Tuesday concentration from different analytical angles — hearing scheduling, incompetency finding dates, and batch processing patterns — providing cross-agent corroboration. (See Appendix A, Query V4)

Discovery 5: One Judge's Caseload Explodes 80× in Two Years

A statistical probability agent discovered that Judge Julia Dayton Klein's CR caseload expanded from 5 cases in 2021 to 395 cases in 2023 while her incompetency finding rate simultaneously quadrupled from 20.0% to 88.4%. This pattern was not requested — the agent found it while autonomously profiling judicial officer behavior across the full CR corpus.

Year	DK Cases	Incompetency Rate
2021	5	20.0%
2022	66	22.7%
2023	**395**	**88.4%**
2024	261	80.8%
2025	197	67.5%



(See Appendix A, Query V5)

Discovery 6: Judicial Officer Identity Suppression — 79.3% Recoverable

An MH cross-table anomaly agent discovered that while only 20.9% of MH docket events and 15.6% of MH dispositions carry judicial officer attribution, the identities are not actually missing — they are recorded elsewhere. 79.3% of dispositions with NULL judicial officer attribution have a named judge in a same-day docket event. The system records who the judge is in one table and strips it from the disposition record.

Note: The raw agent output claimed 93.2% recoverability. Independent database verification produced 79.3%. The pattern is confirmed; the magnitude was overstated. This discrepancy is discussed in Finding 6. (See Appendix A, Query V6)

Category C: System-Level Opacity

Findings about what the system does not record — discovered by agents that were told to look for everything.

Discovery 7: Total Forensic Opacity — Seven Zeros Across 31,795 Events

Multiple agents independently confirmed that across the entire MH case universe (787 cases, 31,795 docket events), seven categories of forensic traceability data are universally absent:

Data Category	MH Population	CR Population
PDF documents available	**0**	4,251
Filing party identified	**0 of 31,795**	Present (1 case)
Examiner identity recorded	**0 (31,795 empty rows)**	Present (1 case)
Evidence UIDs (event→PDF)	**0**	3,760 (MCRO set)
Total cases	787	2,078

The examiner table has 31,795 rows — one per event — but every single row is empty. The schema provides a slot for examiner identity; the system never populates it. This finding emerged from agents independently cataloging field population rates. (See Appendix A, Query V7)

Discovery 8: The 3.7% True Escape Rate

An MH revocation-trap agent discovered that exactly 29 of 787 MH cases (3.7%) ever receive the event “Order Terminating Proceeding and Discharging Patient.” The remaining 96.3% of individuals who enter the civil commitment system never formally exit through termination. (See Appendix A, Query V8)

Verified Discoveries Summary

#	Discovery	Key Figure	Category	Discovered By
1	Incompetency Surge	**19.7× growth (2016→2023)**	Temporal	Temporal Pattern Agent
2	Pipeline Acceleration	**9.1× faster (3.7yr → 5mo)**	Temporal	Cluster Timeline Agent
3	NICS Retroactive Batch	**50 events, max 9.3yr gap**	Temporal	MH Temporal Agent
4	Tuesday Concentration	**65.8% of hearings**	Structural	Multiple agents
5	DK Caseload Explosion	**5→395 cases, 20%→88%**	Structural	Statistical Prob. Agent
6	JO Identity Suppression	**79.3% recoverable**	Structural	Cross-Table Anomaly Agent
7	Total Forensic Opacity	**7 zeros / 31,795 events**	Opacity	8 agents (corroborated)

8	3.7% Escape Rate	**29/787 terminations**	Opacity	Revocation Trap Agent

Finding 6: The Verification Problem — Why Raw Agent Output Is Never Enough

Background

Raw swarm output is not treated as verified findings. The verification-first discipline requires that every figure cited in any forensic report be confirmed against the live database via independent SQL query. Agent outputs are leads, not conclusions. When agent outputs conflict with live database results, the database is authoritative. This section presents three concrete examples from this report's own findings.

Findings

Example A — JO Suppression: Pattern Real, Number Overstated. The MH cross-table anomaly agent (Agent 20) reported that 93.2% of NULL-JO dispositions had a named judicial officer recoverable from same-day events. Independent SQL verification produced 79.3%. The agent likely used a different join methodology or counted disposition-event pairs differently. The finding is directionally identical — the vast majority of "missing" JO attributions are recoverable — but the specific figure was overstated by 14 percentage points.

Example B — Dayton Klein: Methodological Divergence. The statistical probability agent reported DK's caseload as "9 cases in 2021" and her 2025 incompetency rate as "95.1%." Verification produced 5 cases in 2021 and 67.5% in 2025. The discrepancy stems from different definitions of "DK's cases" (the agent may have counted cases where DK appeared at any point vs. cases with DK events specifically in that calendar year) and different incompetency rate denominators.

Example C — The 2035 Future Date: Real Data, Wrong Framing. An anomaly-hunter agent reported an "impossible future date" of September 30, 2035 in case 27-MH-PR-25-661. Verification confirmed the date exists in the parents_dispositions table's commitment report date field. But examination reveals: the case's "60–90 Day Commitment Report" was filed on 09/30/2025 (exactly 90 days after the July 2, 2025 commitment order), while the disposition "Report Date" field shows 09/30/2035 — same month and day, wrong decade. The "impossible" date is almost certainly a data entry typo (2035 for 2025).

Verification Pipeline

Agent Output *(leads)*	SQL Verification *(testing)*	Classification *(evaluation)*	Report *(ground truth)*

	VERIFIED EXACT	Agent figure matches database (e.g., 19.7× surge)
	VERIFIED CLOSE	Pattern confirmed, number adjusted (e.g., JO suppression 79.3% not 93.2%)
	VERIFIED FALSE	Agent claim contradicted by database
	REFRAMED	Data real but agent characterization corrected (e.g., 2035 date)

Significance

These three examples demonstrate that autonomous agents produce both genuine discoveries and overstated claims. The methodology works because it pairs unconstrained exploration with rigorous post-hoc verification. Neither half works without the other. Agent overstatements are expected and healthy — the system is designed to catch them.

Connecting Context

The verification pipeline is the quality control mechanism that elevates swarm outputs from raw analytical leads to citable forensic findings. Finding 7 demonstrates the complementary strength: cross-agent corroboration.

Finding 7: Cross-Agent Corroboration — Independent Discovery of the Same Pattern

Background

When two or more agents independently discover the same finding from different analytical entry points without communicating, the resulting corroboration is methodologically stronger than a single directed analysis. This is the database equivalent of witness corroboration in an investigation.

Findings

The MH Forensic Opacity Finding: 8 Independent Confirmations. The MH swarm master results document records that eight separate agents (Agents 1, 5, 7, 12, 13, 14, 15, and 20) all independently confirmed zero PDFs, zero filing parties, and zero examiner identities for MH cases. None were instructed to look for this. Each agent encountered it while investigating its own domain:

- Agent 1 (Event Vocabulary): encountered empty fields while cataloging event attributes
- Agent 5 (Hearing Forensics): found zero document links while analyzing hearing records
- Agent 7 (Cluster Linkage): discovered the asymmetry while mapping CR-MH connections
- Agent 12 (Related Cases): found zero evidence UIDs while tracing case relationships
- Agent 13 (Charges/Interim Conditions): encountered empty examiner fields in conditional data
- Agent 14 (Examiner Evaluation): directly investigated examiner identity and found universal absence
- Agent 15 (Attribution Asymmetry): systematically compared field population rates across case types
- Agent 20 (Cross-Table Anomalies): confirmed the pattern from a join-based consistency analysis

Eight independent analytical paths all converged on the same finding: the MH case universe is forensically opaque.

Tuesday Hearing Concentration: 3+ Independent Confirmations. The Tuesday concentration was independently discovered by agents analyzing hearing forensics, batch processing patterns, and statistical probability distributions from different swarm runs.

Evidence UID Disappearance: 2 Independent Confirmations. In the CR swarms, Agents 15 and 19 independently discovered the evidence UID disappearance pattern (100% MCRO coverage vs. 2.8% NON-MCRO) from completely different analytical entry points. Neither was instructed to examine evidence UID population rates.

Cross-Agent Corroboration: MH Forensic Opacity

Agent & Domain	How It Found the Same Pattern
Agent 1: Event Vocabulary	Cataloging event attributes → empty MH fields
Agent 5: Hearing Forensics	Analyzing hearings → zero document links
Agent 7: Cluster Linkage	Mapping CR-MH connections → asymmetry discovered
Agent 12: Related Cases	Tracing relationships → zero evidence UIDs
Agent 13: Charges/Interim	Conditional data → empty examiner fields
Agent 14: Examiner Evaluation	Direct investigation → universal absence
Agent 15: Attribution Asymmetry	Systematic comparison → MH = 0% populated
Agent 20: Cross-Table Anomalies	Join-based consistency → pattern confirmed
CONVERGENT FINDING: Zero PDFs, zero filing parties, zero examiners, zero evidence UIDs	

Significance

Independent corroboration from eight isolated analytical processes provides stronger evidentiary weight than a single directed query. Each agent arrived at the same conclusion through a different analytical path, eliminating the possibility that the finding is an artifact of one specific query methodology.

Connecting Context

Cross-agent corroboration and verification discipline are complementary quality mechanisms. Corroboration strengthens confidence in the direction of a finding; verification pins down the exact magnitude.

Finding 8: Why This Matters Beyond This Case

Background

The MCRO Agent Swarm represents a novel application of AI agent technology. This section evaluates its broader significance for forensic investigation and AI methodology.

Findings

Novel use case. Claude Code's multi-agent capability was designed for software development — parallel coding tasks, file management, and test execution. Using it as a parallel forensic investigation framework, where each agent is an autonomous analyst rather than a task worker, appears to be a novel application. The total deployment across this project exceeds 80 autonomous agents executing 4,500+ database queries against a 2.9-million-row forensic database.

Emergent discovery vs. directed analysis. The prescriptive-to-autonomous evolution demonstrates that LLM agents can perform genuine investigative work when given domain context and freedom rather than prescribed tasks. Six of the eight verified discoveries in this report were found by agents following unexpected patterns, not executing anticipated queries. The incompetency surge, pipeline acceleration, NICS retroactive batch, DK caseload explosion, JO identity suppression, and the 3.7% escape rate were all emergent discoveries.

Independent corroboration as methodology. When multiple agents independently discover the same pattern from different analytical entry points without communicating, the resulting corroboration is methodologically stronger than a single directed analysis. The MH opacity finding (8 independent confirmations) and the evidence UID disappearance (2 independent confirmations) demonstrate this principle in practice.

Verification as non-negotiable complement. The three verification examples in this report demonstrate that autonomous agents produce genuine discoveries and overstated claims. The methodology works because it pairs unconstrained exploration with rigorous post-hoc verification. Neither half works without the other. Autonomous agents without verification produce unreliable findings. Verification without autonomous agents only tests what you already thought to ask about.

Scalability. The framework scales linearly — 10 agents, 20 agents, 80 agents across different table domains and case types. Each additional agent adds a new analytical perspective without degrading the others. The MH swarm, CR swarms, and temporal swarms all ran independently against the same static database.

Reproducibility. Every agent's queries are logged. Every finding can be independently verified against the static database. The methodology is fully transparent and reproducible — anyone with database access can re-run any verification query from Appendix A.

Connecting Context

The agent swarm is a tool. Its value is measured by what it discovers, how reliably those discoveries are verified, and whether the methodology can be reproduced. This report demonstrates all three.

KEY METRICS SUMMARY

Combined execution metrics and verified discovery figures are presented in the Executive Summary tables. All figures are reproducible from the SQL queries in Appendix A.

LIMITATIONS & ALTERNATIVE EXPLANATIONS

Agent hallucination risk. LLM agents can produce plausible but incorrect statistical claims. This risk is mitigated by the verification pipeline described in Finding 6, which caught three specific instances in this report alone (JO suppression magnitude, DK caseload figures, and the 2035 date framing). The methodology treats all agent outputs as unverified leads until confirmed against the live database.

Context window limits. Each agent operates within a finite context window. Complex analyses requiring many intermediate query results may exceed this limit, potentially causing an agent to lose track of earlier findings. This manifests as inconsistent numbers across different parts of a single agent's report — an issue caught by verification.

MCP permission constraints. Subagents require project-level permission configuration to access the database. Without this configuration, agents fail silently. This was discovered and resolved during development but remains a potential failure mode for anyone replicating the methodology.

Unproductive threads. Autonomous agents may pursue analytical paths that yield no significant findings. This is acceptable — the value is in aggregate discovery across 20 agents, not per-agent efficiency. Some agents in the MH swarm spent queries exploring tables that proved empty, which is itself a valid (if predictable) finding.

Methodological sensitivity. The same underlying finding can yield different numbers depending on join logic, filter definitions, and denominators. The DK caseload example (Finding 6, Example B) demonstrates this: the core finding (dramatic caseload expansion with rising incompetency rates) is robust across methodologies, but the specific figures are definition-dependent. This underscores why verification against a consistent methodology is essential.

Single-database scope. All findings are derived from the MCRO forensic database. Agent discoveries are bounded by what the database contains. Findings about absent data (e.g., zero MH examiners) may reflect structural features of Minnesota's sealed mental health records rather than anomalies, as noted in Finding 5, Discovery 7.

RECOMMENDATIONS FOR REPLICATION

For anyone seeking to replicate the agent swarm methodology for forensic database investigation:

Permission configuration. Create a .claude/settings.json file at the project level with explicit allow-list entries for all database-access MCP tools. Without project-level permissions, subagents launched in the background cannot prompt for human approval and are silently denied access.

CLAUDE.md primer design. The project primer file should contain: cohort definitions (exact CTEs), schema references (table and column names), prior findings as investigative seed context (not constraints), core analytical principles (e.g., SELECT-only, neutral tone), and output format specifications. The primer provides the shared foundation; agent mandates provide the domain specificity.

Domain decomposition strategy. Divide the database into 15–20 analytical domains with minimal overlap. Each domain should map to a coherent set of tables and analytical questions. Include at least one "anomaly hunter" agent with no domain restriction — its mandate is to find anything unexpected.

Verification pipeline. Every figure cited in any downstream report must be confirmed via independent SQL query against the live database. Agent outputs are leads, not conclusions. Build the verification step into the workflow as a non-negotiable requirement, not an optional quality check. Document all verification discrepancies — they demonstrate the methodology's integrity, not its weakness.

Emphasis: Autonomous agents without verification produce unreliable findings. Verification without autonomous agents only tests what you already thought to ask about. The methodology requires both.

APPENDIX

Appendix A: Verification SQL Queries

V1 — Incompetency Surge (Discovery 1)

```
SELECT
  EXTRACT(YEAR FROM pce.event_json__date)::int AS yr,
  COUNT(DISTINCT pce.case_uid) AS distinct_cases
FROM parents_case_events pce
JOIN parents p ON pce.case_uid = p.case_uid
WHERE p.case_type = 'CR'
  AND pce.event_name ~* 'found.?incomp'
  AND pce.event_json__date IS NOT NULL
GROUP BY yr ORDER BY yr;
```

V2 — Pipeline Acceleration (Discovery 2)

```
WITH cluster_cr AS (
  SELECT cr.cluster_id, MIN(pce.event_json__date) AS first_cr_event
  FROM case_registry cr JOIN parents p ON cr.case_uid = p.case_uid
  JOIN parents_case_events pce ON cr.case_uid = pce.case_uid
  WHERE p.case_type = 'CR' AND cr.cluster_id IS NOT NULL
  GROUP BY cr.cluster_id
), cluster_mh AS (
  SELECT cr.cluster_id, MIN(pce.event_json__date) AS first_mh_event
  FROM case_registry cr JOIN parents p ON cr.case_uid = p.case_uid
  JOIN parents_case_events pce ON cr.case_uid = pce.case_uid
  WHERE p.case_type = 'MH' AND cr.cluster_id IS NOT NULL
  GROUP BY cr.cluster_id
), gaps AS (
  SELECT ccr.cluster_id, (cmh.first_mh_event - ccr.first_cr_event) AS gap_days,
    EXTRACT(YEAR FROM ccr.first_cr_event)::int AS cr_start_year
  FROM cluster_cr ccr JOIN cluster_mh cmh ON ccr.cluster_id = cmh.cluster_id
  WHERE cmh.first_mh_event > ccr.first_cr_event
)
SELECT
  CASE WHEN cr_start_year <= 2019 THEN '<=2019'
       WHEN cr_start_year BETWEEN 2020 AND 2021 THEN '2020-2021'
       ELSE '2022+' END AS era,
  COUNT(*) AS clusters,
  PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY gap_days)::int AS median_gap_days
FROM gaps GROUP BY era ORDER BY era;
```

V3 — NICS Retroactive Batch (Discovery 3)

```
WITH nics_events AS (
  SELECT pce.case_uid, pce.event_json__date AS nics_date
  FROM parents_case_events pce JOIN parents p ON pce.case_uid = p.case_uid
  WHERE p.case_type = 'MH' AND pce.event_name ~* 'NICS'
), prior_events AS (
  SELECT ne.case_uid, ne.nics_date,
    MAX(pce2.event_json__date) FILTER (WHERE pce2.event_json__date < ne.nics_date)
      AS last_pre_nics
  FROM nics_events ne
  JOIN parents_case_events pce2 ON ne.case_uid = pce2.case_uid
  GROUP BY ne.case_uid, ne.nics_date
)
```

```
SELECT COUNT(*) AS total,
  COUNT(CASE WHEN nics_date - last_pre_nics > 365 THEN 1 END) AS over_1yr,
  COUNT(CASE WHEN nics_date - last_pre_nics > 1825 THEN 1 END) AS over_5yr,
  MAX(nics_date - last_pre_nics) AS max_gap_days,
  ROUND(AVG(nics_date - last_pre_nics)) AS avg_gap_days
FROM prior_events WHERE last_pre_nics IS NOT NULL;
```

V4 — Tuesday Concentration (Discovery 4)

```
WITH mcro_seeds AS (
  SELECT DISTINCT p.case_uid FROM parents p WHERE p.case_type = 'CR'
  AND EXISTS (SELECT 1 FROM parents_case_set pcs
    WHERE pcs.case_uid = p.case_uid AND pcs.case_set_json = '"MCRO"')
)
SELECT TRIM(TO_CHAR(pce.event_json__date, 'Day')) AS day_of_week,
  COUNT(*) AS events,
  ROUND(100.0 * COUNT(*) / SUM(COUNT(*)) OVER(), 1) AS pct
FROM parents_case_events pce
WHERE pce.case_uid IN (SELECT case_uid FROM mcro_seeds)
  AND pce.event_json__date BETWEEN '2023-01-01' AND '2024-04-26'
  AND pce.event_name ~* '(hearing|remote|appear)'
GROUP BY day_of_week ORDER BY events DESC;
```

V5 — DK Caseload Explosion (Discovery 5)

```
WITH dk_cases AS (
  SELECT EXTRACT(YEAR FROM pce.event_json__date)::int AS yr, pce.case_uid
  FROM parents_case_events pce
  JOIN parents_case_events_judicial_officers jo
    ON pce.case_uid = jo.case_uid AND pce.event_index = jo.event_index
  JOIN parents p ON pce.case_uid = p.case_uid
  WHERE p.case_type = 'CR'
    AND jo.judicial_officer_json__judicial_officer_norm ~* 'dayton.?klein'
  GROUP BY yr, pce.case_uid
), dk_incomp AS (
  SELECT EXTRACT(YEAR FROM pce.event_json__date)::int AS yr, pce.case_uid
  FROM parents_case_events pce JOIN parents p ON pce.case_uid = p.case_uid
  WHERE p.case_type = 'CR' AND pce.event_name ~* 'found.?incomp'
)
SELECT d.yr, COUNT(DISTINCT d.case_uid) AS dk_cases,
  COUNT(DISTINCT di.case_uid) AS incomp_cases,
  ROUND(100.0 * COUNT(DISTINCT di.case_uid)
    / NULLIF(COUNT(DISTINCT d.case_uid), 0), 1) AS incomp_rate
FROM dk_cases d
LEFT JOIN dk_incomp di ON d.case_uid = di.case_uid AND d.yr = di.yr
WHERE d.yr >= 2021 GROUP BY d.yr ORDER BY d.yr;
```

V6 — JO Suppression (Discovery 6)

```
WITH null_jo_disps AS (
  SELECT pd.case_uid, pd.disposition_json__date AS disp_date
  FROM parents_dispositions pd JOIN parents p ON pd.case_uid = p.case_uid
  WHERE p.case_type = 'MH'
    AND (pd.disposition_json__judicial_officer_norm IS NULL
         OR pd.disposition_json__judicial_officer_norm = '')
), same_day_jo AS (
  SELECT DISTINCT njd.case_uid, njd.disp_date
  FROM null_jo_disps njd
  JOIN parents_case_events_judicial_officers jo ON njd.case_uid = jo.case_uid
  JOIN parents_case_events pce
    ON jo.case_uid = pce.case_uid AND jo.event_index = pce.event_index
```

```
  WHERE pce.event_json__date = njd.disp_date
    AND jo.judicial_officer_json__judicial_officer_norm IS NOT NULL
    AND jo.judicial_officer_json__judicial_officer_norm != ''
)
SELECT (SELECT COUNT(*) FROM null_jo_disps) AS total_null,
  (SELECT COUNT(*) FROM same_day_jo) AS recoverable,
  ROUND(100.0 * (SELECT COUNT(*) FROM same_day_jo)
    / NULLIF((SELECT COUNT(*) FROM null_jo_disps), 0), 1) AS pct;
```

V7 — Forensic Opacity (Discovery 7)

```
SELECT
  (SELECT COUNT(*) FROM children c
   JOIN parents p ON c.case_uid = p.case_uid WHERE p.case_type = 'MH') AS mh_pdfs,
  (SELECT COUNT(*) FROM parents_case_events pce
   JOIN parents p ON pce.case_uid = p.case_uid
   WHERE p.case_type = 'MH'
     AND event_json__filing_party_name IS NOT NULL
     AND event_json__filing_party_name != '') AS mh_filing_party,
  (SELECT COUNT(*) FILTER (
     WHERE ex.examiner_json IS NOT NULL
       AND ex.examiner_json::text != 'null'
       AND ex.examiner_json::text != '{}')
   FROM parents_case_events_examiners ex
   JOIN parents p ON ex.case_uid = p.case_uid
   WHERE p.case_type = 'MH') AS mh_examiners_with_content,
  (SELECT COUNT(*) FROM parents_case_events_evidence_uids eu
   JOIN parents p ON eu.case_uid = p.case_uid
   WHERE p.case_type = 'MH') AS mh_evidence_uids;
```

V8 — True Escape Rate (Discovery 8)

```
SELECT event_name, COUNT(*) AS cnt, COUNT(DISTINCT case_uid) AS cases
FROM parents_case_events
WHERE case_uid IN (SELECT case_uid FROM parents WHERE case_type = 'MH')
  AND event_name ~* '(terminat|full.?discharge|order.*terminat)'
GROUP BY event_name ORDER BY cnt DESC;
-- Result: "Order Terminating Proceeding and Discharging Patient"
--   = 29 cases of 787 = 3.7%
```

Appendix B: Prescriptive Swarm Instruction Set Structure

The following outline shows the anatomy of a prescriptive swarm instruction set, based on the MCRO_FROZEN_VS_LIVE_SWARM.md file used for the initial deployment:

Section	Content
Mission Statement	Central question, control/test group definitions, what divergence means
Database Connection	Supabase project ID, READ-ONLY constraint, MCP tool name
Cohort Definitions	Exact CTEs for MCRO (163 cases) and NON-MCRO (1,865 cases) with First-50 exclusion
Date Window Rule	event_json__date <= '2024-04-30' applied to both cohorts for temporal parity
Deployment Instructions	Spawn 20 agents via Task() with run_in_background: true
Return Format	Standardized structure: cohort verification, primary finding, severity, evidence table, SQL
Agent 1–19 Mandates	Each agent: domain description, 2–4 pre-written SQL queries, specific comparison targets
Agent 20: Synthesis	Master Evidence Matrix assembly, synthesis narrative, no novel queries
Output Deliverable	Master results file with matrix, per-agent findings, top-20 list, narrative

Total length: approximately 1,300 lines including 40+ pre-written SQL queries. Each agent's mandate occupied 30–80 lines of instructions.

Appendix C: Autonomous Agent Mandate Template

Comparison of what a prescriptive agent receives vs. what an autonomous agent receives:

Prescriptive Agent Receives	Autonomous Agent Receives
Domain name and description	Domain name and description
2–4 pre-written SQL queries	**No pre-written queries**
Exact column names specified	**Schema exploration mandate**
Expected output format	Flexible reporting format
Cohort CTEs (shared)	Cohort CTEs (shared)
No prior findings context	**Prior findings as seed context / attack vectors**
Fixed scope	**"Follow anomalies, report everything"**

The autonomous agent's mandate is typically 10–15 lines: a domain, a cohort definition, seed context from prior runs, and the instruction to explore freely. The prescriptive agent's mandate is 30–80 lines: detailed SQL, specific columns, exact comparison targets. The autonomous agent produces 60–80 queries per run; the prescriptive agent executes 2–4.

Appendix D: Glossary

Term	Definition
Claude	An AI assistant built by Anthropic, capable of reasoning, writing code, and analyzing data.
Claude Code	A command-line tool that allows Claude to execute code, manage files, and interact with external services like databases.
Subagent	An instance of Claude spawned by another Claude instance (the orchestrator) to perform a specific task in the background.
Orchestrator	The main Claude Code session that reads instructions, spawns agents, monitors completion, and compiles results.
MCP	Model Context Protocol — a standard for connecting AI agents to external tools and databases.
Supabase	A cloud database platform providing hosted PostgreSQL databases with API access.
PostgreSQL	An open-source relational database system used to store and query the forensic data.
Context window	The amount of text an AI agent can process at once. Each agent has its own, independent of other agents.
CTE	Common Table Expression — a named temporary result set in SQL used to define cohorts.
Cohort	A defined group of cases used for comparison (e.g., MCRO seed = 163 CR cases from April 2024 download).
NICS	National Instant Criminal Background Check System — federal firearms background check database.
Rule 20	Minnesota Rule of Criminal Procedure 20.01 governing competency evaluations for criminal defendants.
MCRO	Minnesota Court Records Online — the public-access portal for Minnesota court records.
CR / MH / GC	Case type codes: Criminal, Mental Health (civil commitment), Guardianship/Conservatorship.
OTS	OpenTimestamps — a protocol for anchoring file hashes to the Bitcoin blockchain for independent timestamp proof.
SHA256	A cryptographic hash function producing a unique 256-bit fingerprint of any file. Identical hashes = identical files.
Emergent discovery	A finding that was not anticipated or prescribed — discovered by an agent following unexpected patterns in data.