MCP SERVER | NO CODE | CLOUD HOSTED

Metaplane MCP

Manage Data Quality and Incidents via Chat

Metaplane connects data observability directly into your AI client. Get full control over data quality by tracking monitors, incident history, and system health through natural conversation. You can check connection schemas, list all active alerts, or trigger a manual monitor run without leaving your chat interface.

A+ Quality Score 98.33/100

data-quality | data-observability | incident-management | data-pipelines | monitoring | alerts

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/metaplane

SCAN TO VIEW

The infrastructure that powers AI agents in the real world.

Vinkius connects AI to the world's software through secure, enterprise-grade infrastructure — enabling real-world execution at scale, built on the Model Context Protocol (MCP).

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the cloud infrastructure where AI agents connect to the software your business already runs. We handle the hosting, the security, the credentials, the uptime — you get agents that actually do things.

We operate the world's largest managed MCP catalog. Major SaaS platforms, CRMs, databases, and cloud providers — running, monitored, production-ready. This MCP server is hosted and maintained by the Vinkius Cloud for AI Agents.

> *The agent doesn't manage credentials, doesn't manage uptime, doesn't manage security. Vinkius does.*
>
> — Architecture principle

Four Pillars of the Vinkius Runtime

01 — Security by design

Credentials stay encrypted at rest via AES-256. The AI agent never touches raw keys — they're injected into a sandboxed V8 isolate at runtime. Actions are logged, and connections have an emergency kill switch.

02 — Built on MCP Fusion

This MCP server was built with **MCP Fusion**, the open-source framework (Apache 2.0) that powers the entire Vinkius catalog. Schema-as-firewall strips undeclared fields, compiled PII redaction runs at zero overhead, and cryptographic lockfiles produce git-diffable audit trails.

03 — Deterministic observability

Eight immutable metrics per endpoint: request volume, p95 latency, error rate, active connections, cost attribution. A live payload feed logs every tool call with mutation detection.

04 — Autonomous operations

Servers are deployed, monitored, and patched autonomously. New capabilities and security patches ship weekly. Zero-downtime deployments ensure continuous availability across all managed MCP servers.

AES-256	Ed25519	24h TTL	V8 Isolate
Encryption at rest	PKI vault signatures	Ephemeral session keys	Sandboxed execution

One Token. Instant Access.

Every MCP server on Vinkius is accessed through a **Connection Token**. Tokens are generated in the cloud dashboard and produce a unique MCP endpoint URL. Paste this URL into any MCP-compatible client — no SDK required.

A single token can serve **multiple AI clients simultaneously**, or you can issue separate tokens per client for granular access control. Each token tracks its own request count, last activity timestamp, and can be individually enabled or revoked.

MCP ENDPOINT

```
https://edge.vinkius.com/{token}/mcp
```

Claude · Cursor · VS Code · Windsurf · Grok · Gemini

Security Is the Architecture

Security in Vinkius is not a feature — it's the foundation of the runtime. The gateway enforces multiple independent protection layers between AI agents and third-party APIs.

01 — Ed25519 PKI Vault

Every workspace has an Ed25519 Master Key. Session keys are generated ephemerally (24h TTL) and signed by the Master Key. Credentials never leave the vault boundary.

02 — V8 Isolate Sandboxing

Tool code runs inside isolated-vm V8 isolates with 64 MB memory caps and per-request timeouts. No filesystem access, no network access except through the SSRF-guarded fetch bridge.

03 — SSRF Guard

All outbound HTTP requests are DNS-resolved and validated before execution. Private IP ranges (10.x, 172.16-31.x, 192.168.x, AWS metadata 169.254.x) are blocked at the network layer.

04 — DLP & PII Redaction

A ResponseGuard pipeline intercepts every tool response. Configurable redaction patterns strip sensitive fields (emails, SSNs, card numbers) before data reaches the AI agent.

05 — Cryptographic Audit Trail

Every request is signed into a SHA-256 hash chain with Ed25519 signatures. Events form a tamper-proof, SIEM-exportable forensic record.

06 — Honeytoken Trap System

Phantom credentials are injected into isolated environments. If a honeytoken is used outside Vinkius infrastructure, the server is quarantined instantly.

Emergency Kill Switch

EU AI Act Art. 14(1) Compliant

The kill switch is an **emergency halt** mechanism — not a simple toggle. When triggered, it executes three actions atomically:

01 — Server deactivated

The MCP server is immediately taken offline across the entire cluster.

02 — All tokens revoked

Every connection token is invalidated. Total lockout — reconnection blocked until new tokens are issued.

03 — WebSocket connections killed

Active connections terminated via Redis pubsub broadcast. Propagates to every runtime node in the cluster.

Full Visibility. Zero Guesswork.

The Vinkius cloud dashboard includes a full MCP Governance suite — real-time analytics and security controls for production AI operations.

Control Plane

KPI dashboard with request volume, latency, success rate, token consumption, and AI-generated operational briefings.

Agent Activity

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

FinOps

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

Tool Health

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

Firewall & DLP

PII redaction activity, sensitive data protection counters, and security event timeline.

Incident Log

Error trends, failure rates, status-code breakdowns, and forensic audit trail access.

> Get started at **cloud.vinkius.com** — connect your AI agent in under 60 seconds.

Metaplane MCP

10 tools available
Cloud-hosted on Vinkius

This MCP lets you manage data quality and understand the status of your pipelines using only conversation. Instead of logging into Metaplane's dashboard and clicking through multiple tabs to see if everything's running okay, you just ask your AI client. You can pull a list of all configured monitors or fetch details on a specific incident that popped up last night. Need to validate data quality right now? Your agent triggers a monitor run for you. Furthermore, you can view the connection schemas across different databases and even check active alert rules. When you connect this through Vinkius, your AI client becomes your single pane of glass for all things data observability.

Core Capabilities

01 — Check Data Monitor Status

List existing monitors or fetch detailed health metrics and metadata for specific monitoring points.

02 — Review Incident History

Pull real-time details on data quality incidents, including historical records and resolution status.

03 — Force Data Checks

Programmatically start a monitor run to validate data quality immediately, regardless of the schedule.

04 — Map Data Sources

View all connected databases, warehouses, and schemas to build a map of your company's data lineage.

05 — Manage Alerts

List and examine configured alert rules to ensure your team gets notified when things go wrong.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/metaplane** — connect your AI agent in three steps.

01 Subscribe to the Metaplane MCP using your API key.

02 Connect this MCP via Vinkius to any AI client (like Cursor or Claude).

03 Ask your agent natural language questions, like 'What were the top three incidents yesterday?' and get immediate data answers.

The bottom line is that you manage complex data health checks using simple chat prompts instead of navigating multiple web dashboards.

Built For

Data engineers, platform reliability teams, and BI analysts need this. It's for the ops engineer who's tired of clicking through dozens of dashboard tabs at 2 am just to find out which pipeline broke last night.

Data Engineer

Uses the MCP to list connected schemas and trigger monitor runs when they suspect a data pipeline failure, validating quality on demand.

Site Reliability Engineer (SRE)

Checks incident history and fetches details for specific incidents to quickly triage alerts without manual dashboard access.

Data Analyst

Lists all data monitors and reviews alert configurations to understand what metrics are being tracked across the company's datasets.

What Changes When You Connect

01 Stop jumping between dashboards. You can list all monitors or check specific incident details directly through your AI client, keeping context in one place.

02 Validate data quality on demand. Instead of waiting for a scheduled run, you use the `trigger_monitor_run` tool to manually test data integrity immediately after an ETL job.

03 Understand your entire stack. The MCP lets you list all data connections and schemas, giving you full visibility into where your data comes from.

04 Proactive monitoring is simple. You can list configured alerts or get details on a specific monitor to ensure the right rules are running.

05 Contextualized troubleshooting. When an incident happens, you can run `list_incidents` and then drill down with `get_incident` without leaving your chat window.

Real-World Applications

Investigating a sudden dashboard dip

A data analyst notices revenue numbers are off. Instead of filing a ticket, they ask their agent to list all monitored connections and schemas for the affected reports. This reveals that the primary Snowflake warehouse connection was listed as disconnected, pinpointing the failure point immediately.

Pre-release data validation

A data engineer finishes an ETL pipeline and needs to confirm quality before deployment. They use their agent to `trigger_monitor_run` on the key tables, validating that row counts and null checks pass instantly, saving a manual QA cycle.

Reviewing historical system failures

An SRE is prepping for an audit. They ask their agent to list all data incidents over the last quarter. This quickly surfaces recurring high-severity issues that might require architectural fixes, rather than just patching code.

Auditing pipeline dependencies

A BI architect needs to know which datasets feed into a critical dashboard. They use the MCP to list all data connections and connection schemas, mapping out every dependency in minutes using simple prompts.

Patterns to Avoid

Treating it like a general query tool

✗ AVOID

Asking 'What is my data observability?' The agent will give vague marketing answers because the prompt lacks specific actions.

✓ INSTEAD

Instead, ask for concrete data: 'List all configured alerts' or 'Show me the run history for the Postgres Row Count monitor'. Use tools like `list_configured_alerts` to get actionable status.

Ignoring connection scope

✗ AVOID

Trying to find a schema without knowing which database it belongs to, resulting in a vague list of all possible schemas.

✓ INSTEAD

First use `list_data_connections` to identify the correct source. Then ask for the specific data structures using `list_connection_schemas` for precision.

Manually checking every status

✗ AVOID

Logging into the web UI and clicking through monitors, incidents, runs, and connections one by one until all are verified.

✓ INSTEAD

Ask your agent to perform a bulk check: 'List all data monitors' followed by 'list_incidents'. This aggregates the necessary status updates instantly.

The Right Fit

Use this MCP if your primary pain point is siloed visibility. If you currently have multiple dashboards—one for incidents, one for monitor health, and another for connection schemas—you need this. It centralizes that data into conversation. However, don't use it if you simply need to write a SQL query or perform ad-hoc data transformations; this MCP is about *observability* and *reporting* on state, not execution of raw code. If your goal is pure ETL development, you might prefer a dedicated CI/CD tool instead. But if the goal is 'Did my pipeline break? And what's happening right now?' then Metaplane is exactly what you need.

Data health status is spread across five different web tabs.

Today, checking your data quality requires a multi-tab session. You jump to the Monitor tab to see if things are running; then you open the Incident dashboard to check for alerts; next, you click on connection details just to verify schemas; and finally, you scroll through run history logs to find the root cause of an anomaly.

With this MCP, your agent handles all that clicking. You ask it to list all data monitors, see if there are active incidents, and even check connected databases—all in one conversation. It collapses a 10-minute manual audit into seconds.

Get full visibility using Metaplane's tools.

You no longer need to remember which tool tracks what. You can use `list_monitors` to see every check configured, then ask your agent to list connection schemas across the affected data source, and finally run `get_monitor_runs` to prove when it last passed.

The process shifts from investigation (clicking tabs) to inquiry (asking questions). The state of your entire data platform is now available via natural language prompts.

Metaplane MCP with 10 Tools

These tools let you read account info, list incidents or monitors, view connections, and manually trigger checks for complete data visibility.

#	TOOL	DESCRIPTION
01	`get_account_info`	Retrieves general account information for your Metaplane setup.
02	`get_incident`	Fetches detailed records and status updates for one specific data quality incident.
03	`get_monitor`	Gets comprehensive details about a single configured data monitor.
04	`get_monitor_runs`	Retrieves the execution history and results for a specific data monitor.
05	`list_configured_alerts`	Lists all active notification settings and alert rules configured in Metaplane.
06	`list_data_connections`	Enumerates every connected data source, warehouse, and database within your account.
07	`list_incidents`	Generates a list of historical or active data incidents across the entire system.
08	`list_monitors`	Provides an overview and list of every configured data monitor currently set up.
09	`list_connection_schemas`	Lists all available schemas within a specified data connection or warehouse.
10	`trigger_monitor_run`	Manually initiates a monitor run to check the current data quality against defined rules.

See It in Action

Real prompts you can use once this MCP is connected to your AI agent through Vinkius Cloud.

U List all my data monitors in Metaplane.

Retrieving monitors... I found 5 active monitors including 'Postgres Row Count' and 'Snowflake Schema Change'.

U Show recent incidents for the last 24 hours.

Checking incidents... There is one active high-severity incident: 'Null values detected in production.users'.

U Trigger a run for monitor ID 'mon_12345'.

Triggering run... Monitor 'mon_12345' (Postgres Volume) has been started and is currently processing.

Frequently Asked Questions

01 **How do I use the Metaplane MCP to check if a specific monitor failed?**

You can first run `list_monitors` to find the monitor ID, and then use `get_monitor` for detailed status. If it failed recently, you should also check `list_incidents`.

02 **Can Metaplane MCP list all my databases?**

Yes. You can run the `list_data_connections` tool to enumerate every data source connected to your account.

03 **What is the difference between listing monitors and triggering a monitor run with Metaplane MCP?**

Listing monitors (`list_monitors`) shows you what checks are configured. Triggering a run (`trigger_monitor_run`) actually executes those checks right now to validate current data quality.

04 **Does the Metaplane MCP help me find which schemas exist?**

Yes, it does. You can use `list_connection_schemas` after identifying a specific connection to see all available structures within that warehouse.

Go Live in 60 Seconds

Get your connection token from **cloud.vinkius.com**, then paste the endpoint URL into any MCP-compatible client.

YOUR MCP ENDPOINT

```
https://edge.vinkius.com/[TOKEN]/mcp
```

CLIENT	WHERE TO CONFIGURE
Claude AI	Profile → Customize → Connectors → "+" → Add custom connector → Paste endpoint
Cursor	Settings → Features → MCP Servers → "+ Add New MCP Server" → Type: SSE → Paste endpoint
VS Code	Ctrl/Cmd+Shift+P → "MCP: Add Server" → add `"metaplane": { "url": "…" }`
Windsurf	MCP Settings → `mcp_settings.json` → Add endpoint URL
ChatGPT	Settings → Tools & plugins → Add MCP server → Paste endpoint
Gemini	Extensions → Add MCP Server → Paste endpoint URL

ASK AN AI ABOUT THIS

Let your preferred AI explain this MCP server

- Ask ChatGPT
- Ask Claude
- Ask Perplexity
- Ask Gemini
- Ask Grok

READY TO CONNECT

Metaplane is live on Vinkius Cloud.

Get your connection token, paste it into your AI agent, and start building. No SDK. No deployment. Just results.

Start at cloud.vinkius.com →

vinkius.com · support@vinkius.com

INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by Metaplane. All third-party trademarks, logos, and brand names are the property of their respective owners. Their use in this document is strictly for informational purposes to identify service compatibility and interoperability.

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	Metaplane MCP
Server ID	`019d75d3-869f-7334-9a90-0f6125786560`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE

This document is generated automatically by the Vinkius PDF Engine. Content reflects the MCP server configuration at the time of generation and may change as updates are deployed. For the most current information, visit vinkius.com/mcp/metaplane.