MCP SERVER | NO CODE | CLOUD HOSTED

Cube.dev MCP for AI Agents

Query Semantic Data Warehouses and Manage Aggregations

Cube.dev MCP connects your AI client directly to a semantic data layer, letting you query complex data warehouses using natural language. Instead of writing boilerplate SQL or navigating multiple dashboards, your agent executes queries, inspects generated SQL code, and manages data model metadata instantly. You get consistent metrics and high-performance insights without knowing the underlying database structure.

F Quality Score 3.6/100

semantic-layer | data-modeling | sql-api | rest-api | pre-aggregations

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

SCAN TO VIEW

The connectivity layer between AI and the world's software.

Vinkius sits between AI and every application. All communication passes through Vinkius Cloud via the Model Context Protocol (MCP) — with governance, observability, and security at every layer.

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the connectivity layer where AI connects 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.

Cube.dev MCP

15 tools available
Cloud-hosted on Vinkius

This MCP gives your AI client a direct line into your data warehouse's semantic layer. Think of it as bypassing all the manual setup—you don't need to know if your data lives in Snowflake or BigQuery, just what you want to know about it.

It lets your agent run complex queries by translating natural language directly into reliable metrics and dimensions. You can debug models instantly by asking the MCP to show the raw SQL that was generated for a query. Need to check performance? You can trigger background jobs to pre-aggregate data, ensuring your dashboards stay fast even when querying huge datasets.

It's all managed through Vinkius, which makes connecting this power source simple. Your agent doesn't just retrieve numbers; it understands the structure of your entire data model—the cubes and views—letting you explore metadata right from the chat window. This is how you get reliable answers to tricky business questions without writing a single line of SQL.

Core Capabilities

01 — Execute Aggregated Data Queries

Run complex reports using measures, dimensions, and filters by invoking the load_query tool.

02 — Debug Raw SQL Code

Use tools like get_sql or execute_cube_sql to see or run raw SQL queries against your database for deep investigation.

03 — Inspect Data Model Metadata

Retrieve details about cubes, views, and segments using get_meta to understand the data structure without leaving your chat interface.

04 — Manage Performance Jobs

Trigger and check the status of background pre-aggregation builds using trigger_pre_aggregation_job.

05 — Examine Data Sources and Deployments

List configured data sources (list_data_sources) or manage cloud infrastructure details like deployments (list_deployments).

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to the Cube.dev MCP and provide your unique API URL and Secret Token credentials.

02 Your AI client connects these tokens, allowing it to interact with Cube's semantic layer via the Vinkius framework.

03 You ask a natural language question (e.g., 'What were total sales last month?'), and the agent executes the necessary data functions and returns the final, structured answer.

The bottom line is: your AI client uses this MCP to handle all the complex connection logic between plain English questions and your underlying database structure.

Built For

This connector solves a huge headache for any professional who spends time translating business needs into data queries. If you're tired of manually debugging generated SQL or waiting for a BI analyst to run a simple report, this is for you.

Analytics Engineer

Debugging complex metric calculations by inspecting the raw SQL that gets generated from natural language prompts.

Data Analyst / Product Manager

Getting instant, reliable answers to ad-hoc business questions by letting the AI query the semantic layer directly, without submitting a ticket.

Data Engineer

Quickly verifying data model definitions and triggering cache refreshes or pre-aggregation jobs via an intuitive conversational interface.

What Changes When You Connect

01 Stop fighting boilerplate SQL. Instead of crafting complex `SELECT` statements, the agent handles the syntax, allowing you to focus purely on business logic.

02 Maintain metric consistency across your organization. By querying through a semantic layer, every user gets the same definition for 'Total Revenue,' regardless of who writes the query.

03 Save hours debugging data models. If a number looks wrong, use `get_sql` or `get_meta` to instantly see *why* that number was calculated, right in your chat.

04 Keep dashboards lightning fast. Instead of letting slow queries bog down reporting, you can proactively trigger jobs using `trigger_pre_aggregation_job` to optimize performance.

05 Manage infrastructure context easily. You can list deployments and environments (like staging or production) directly through the MCP, ensuring your agent is talking to the right data source.

Real-World Applications

Calculating year-over-year growth for a Product Line

A Product Manager needs to compare Q3 2024 sales against Q3 2023. Instead of writing complex date logic, they ask the agent directly. The MCP uses `load_query` and handles all the necessary filtering and aggregation to deliver the comparative report.

Finding out why a key metric dropped last week

An Analytics Engineer notices 'Active Users' dipped suddenly. They ask the agent, prompting it to use `get_sql` to show the underlying query logic. By inspecting the generated SQL, they pinpoint that a specific dimension filter was incorrectly applied.

Setting up performance for a new executive dashboard

The team is building a board with dozens of widgets. Before launching, an engineer triggers `trigger_pre_aggregation_job` on the core 'Sales' cube. This proactively builds necessary indexes and aggregates, ensuring the dashboards run smoothly when launched.

Understanding how different data sources connect

A new team member joins and needs to know what data is available. They ask the agent for metadata, prompting it to use `get_meta` or `list_entities`. The MCP returns a structured list of all cubes (like 'Orders' or 'Users') and their definitions.

Patterns to Avoid

Writing manual SQL for every ad-hoc question

✗ AVOID

The analyst spends 45 minutes writing a complex JOIN query, debugging syntax errors, and dealing with inconsistent column names because they don't know the underlying schema.

✓ INSTEAD

Just ask your agent. Let it use `load_query` to interpret 'Show me all orders from California last month.' The MCP handles the joins and filters automatically.

Assuming metrics are always calculated correctly

✗ AVOID

A PM sees a total revenue number that looks wrong. They waste time asking senior devs, only to find out the metric definition is flawed.

✓ INSTEAD

Use `get_sql` immediately. The agent shows you exactly what SQL was run for the number, letting you verify data logic without bothering anyone else.

Ignoring data freshness and performance bottlenecks

✗ AVOID

The dashboard runs slowly every morning because the underlying data volume has grown past the current cache limits. The user just assumes the slow speed is normal.

✓ INSTEAD

Use `trigger_pre_aggregation_job` to schedule a build, optimizing the model so that future queries run at peak performance.

The Right Fit

Use this MCP if your team relies on complex data models defined in a semantic layer and needs natural language access to those metrics. You're here if you need to query aggregated results or inspect underlying SQL logic—tools like `load_query`, `get_sql`, and `get_meta` are your best friends.

Don't use this MCP if you just want to run a simple, isolated report that doesn't touch the core data model. If all you need is basic database connectivity without semantic rules, a direct JDBC/ODBC connector might be better. Also, don't rely on it for general text generation; its sole focus is structured, reliable data retrieval.

Cube.dev MCP: Simplifying Data Warehouse Queries

Right now, getting a single answer from your massive data warehouse means jumping between tools: running reports in BI software, writing SQL in an IDE, and then copying the resulting numbers into a spreadsheet for analysis. It's slow, prone to syntax errors, and every time you hit 'run,' you risk pulling conflicting metrics because different people write queries differently.

With this MCP, your agent connects directly to Cube.dev's semantic layer. You ask a question in plain language—like, 'What was the average order value for Gold Tier customers?' The system translates that into the right query and returns one definitive answer. It’s instant data confidence.

Cube.dev MCP: Managing Data Model Metadata

Previously, if you weren't a senior engineer, figuring out which table held the 'user status' or what exactly 'total amount' meant was a guessing game involving reading documentation that nobody actually reads.

Now, you can ask the agent for metadata. The MCP uses tools like `get_meta` to explain your entire data model in plain English, showing you every cube and view available. It means zero guesswork and faster onboarding.

Cube.dev: 15 Tools for Data Model Querying

These tools allow your AI agent to perform deep technical actions like running raw queries, checking metadata, or optimizing data aggregation jobs.

#	TOOL	DESCRIPTION
01	`check_live`	Confirms whether the Cube deployment is currently active and accessible.
02	`check_ready`	Checks if a specific Cube deployment has completed its initial setup process.
03	`convert_query`	Translates standard SQL queries into the specialized REST API format required by Cube.dev.
04	`execute_cube_sql`	Runs a raw, custom SQL query against the database for deep data investigation or testing purposes.
05	`generate_meta_token`	Creates a temporary JSON Web Token (JWT) needed to access the Metadata API when using Cube Cloud.
06	`get_entity`	Retrieves detailed metadata for one specific cube or view definition.
07	`get_meta`	Provides a list of general metadata covering all available cubes and views in the data model.
08	`get_pre_aggregation_job_status`	Retrieves the current status (running, failed, completed) of background pre-aggregation jobs.
09	`get_sql`	Displays the actual SQL code that Cube.dev generates when running a query, useful for debugging data discrepancies.
10	`list_data_sources`	Lists all external databases or services that are currently configured and connected to the Cube instance.
11	`list_deployments`	Retrieves a list of all available deployments if you are using Cube Cloud infrastructure.
12	`list_entities`	Shows an overview listing of every cube and view defined in the semantic layer.
13	`list_environments`	Lists all supported environments (e.g., staging, production) for a given deployment using Cube Cloud.

#	TOOL	DESCRIPTION
14	`load_query`	Executes the primary query function to return aggregated data results based on natural language parameters.
15	`trigger_pre_aggregation_job`	Initiates a background job build to pre-calculate and optimize metrics for faster future querying.

See It in Action

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

U What were our total sales last quarter, grouped by product status?

Sales Report: Q3 2024

PRODUCT STATUS	TOTAL REVENUE
Completed	$1.2M
Pending	$85k
Cancelled	$12k

The highest volume came from 'Completed' orders, particularly those related to the Premium line.

U Show me how the system calculates the active user count.

Underlying Query Logic

```sql
SELECT
count(DISTINCT users.user_id) AS active_users
FROM orders o
JOIN users u ON o.user_id = u.user_id
WHERE o.created_at ⩾ '2024-12-01' AND o.created_at < '2025-01-01'
```

This confirms that the calculation correctly counts unique users who placed an order in December.

U I need to check if pre-aggregations are running for the marketing cube.

Pre-Aggregation Job Status

Job ID: 8b3c4d2e
Cube: Marketing Metrics
Status: Running
Estimated Completion Time: ~15 minutes

Please wait a few moments, or check the status again using your agent.

Frequently Asked Questions

01 **How does Cube.dev MCP help me get data insights without writing SQL?**

It translates your natural language questions into reliable database queries automatically. You just ask the question, and the agent handles all the complex code generation, giving you accurate answers directly.

02 **Can I use Cube.dev MCP to check if my data model is consistent?**

Yes. By using metadata tools like get_meta, the system shows you every cube and view available. This lets you verify your data model structure and understand how different pieces of data relate.

03 **What if my dashboard runs slowly? Can Cube.dev MCP fix that?**

It can help you optimize performance. You can trigger background pre-aggregation jobs, which calculate complex metrics ahead of time so your dashboards load instantly when needed.

04 **Does Cube.dev MCP only work for one type of database?**

No. Because it uses a semantic layer, it abstracts the underlying database complexity away from you. You focus on the data metrics, not the specific SQL dialect or connection details.

05 **Can I test my custom queries using Cube.dev MCP?**

Absolutely. It provides tools like execute_cube_sql that let your agent run raw, customized SQL against the database for deep testing and investigation purposes.

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 `"cubedev": { "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

Cube.dev 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 Cube.dev. 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	Cube.dev MCP
Server ID	`019e3882-3c56-7025-8136-0b8f9938702a`
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/cubedev.