MCP SERVER | NO CODE | CLOUD HOSTED

# Amazon Redshift MCP for AI Agents

## Querying and analyzing petabyte-scale data warehousing datasets

Amazon Redshift MCP connects your AI agent directly to a petabyte-scale data warehouse. It lets you run complex SQL queries, check schema structure, and analyze massive datasets right through conversation, eliminating the need for external database connections or complicated drivers.

A+ Quality Score 100/100

data-warehousing | big-data | sql-queries | cloud-analytics | data-processing | business-intelligence

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/amazon-redshift

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.

# Amazon Redshift MCP

7 tools available
Cloud-hosted on Vinkius

Running analytics on enormous data warehouses usually means switching tools: jumping from your chat window to an IDE like DBeaver, managing credentials, and dealing with slow network setups. This MCP changes that.

It gives your AI agent a direct, secure line into Amazon Redshift. You can prompt it to run complex SQL commands—anything from counting sales across regions to creating new tables or just looking up column definitions. Because the connection uses AWS's Data API, the process is built for scale and speed; long-running reports happen in the background without bogging down your chat session.

Whether you're a data scientist needing ad-hoc metrics or a developer testing schema changes, you simply ask. The agent handles submitting the query, monitors its status, and pulls the final result set right into your conversation feed. This capability makes large-scale data exploration feel as natural as texting a coworker.

## Core Capabilities

**01 — Discover schemas and tables**

The agent lists all available database structures, allowing you to pinpoint the exact data source needed for your query.

**02 — View column metadata**

It describes any table's columns, showing their names, types, and whether they can accept null values.

**03 — Execute complex SQL queries**

You run full SQL statements for aggregation or modification, which the system manages as a background job.

**04 — Track query status and results**

The agent monitors your running job ID, alerting you when it's done, and then securely retrieving the final data rows into your chat conversation.

## One Click on Vinkius — From Prompt to Execution

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

01 Authorize the Amazon Redshift MCP plugin from your connected extension hub.

02 Configure the serverless integration using standard AWS IAM principles, providing access keys and defining the target database endpoint.

03 Prompt your AI client with a request like "Show me all tables in the marketing schema" or "Calculate total Q4 sales."

The bottom line is: you talk to your agent, it executes complex data logic against Redshift, and you get the clean answers back inside your chat interface.

## Built For

This MCP is for anyone drowning in massive datasets who hates switching between tools. If your job involves looking at schema definitions or running ad-hoc reports on petabytes of data, this saves you hours of context switching and manual API calls.

**Data Analyst**

Runs exploratory aggregations by asking natural language questions about datasets. They use the MCP to get instant metrics and validate schema structure without leaving their chat.

**Backend Developer**

Tests database migrations or data integrity checks interactively. They use it to check table state and troubleshoot connections directly from their development environment.

**Data Engineer**

Audits cluster loads and verifies the execution lifecycles for large reporting jobs. They monitor status and track statements asynchronously through the MCP.

## What Changes When You Connect

01 Skip the ODBC drivers and connection pools. Your agent handles the complex, secure communication layer to massive data sources.

02 Instantly get metadata. Instead of navigating multiple console views, use the MCP's ability to describe tables and schemas in plain text chat.

03 Handle long-running reports without timing out. Use `execute_sql` to run large aggregations in the background and check status later with `statement_status`.

04 Streamline auditing. Need to know what queries ran last week? The MCP lets you list historical statements, making compliance checks quick.

05 Centralized data access means less switching between tools. You manage schema discovery, query execution, and result retrieval all in one conversational flow.

# Real-World Applications

**Auditing quarterly revenue totals**

A finance analyst needs to verify the total sales amount for a specific region from last quarter. They ask the agent to run an aggregation query, and it uses `execute_sql` to generate a job ID. The analyst then monitors progress using the status tool until they pull the final sum.

**Testing new data models**

A backend developer needs to see if their proposed schema changes will break existing reporting tables. They use `list_schemas` and `describe_table` interactively, checking column definitions before committing code to the database.

**Discovering available data sources**

A new data scientist joins a project and needs to know what datasets are available. They ask the agent to list all schemas using `list_schemas` and then drill down into specific tables, making their initial investigation fast.

**Troubleshooting failed reports**

A data engineer finds a report failed overnight. Instead of wading through system logs, they use the MCP to list recent statements (`list_statements`) and check the status of the failure point immediately.

# Patterns to Avoid

### Trying to query everything at once

✗ AVOID

A user tries to run a complex SELECT statement that requires checking 15 different schemas, often leading to connection failures or confusing error messages.

✓ INSTEAD

First, use `list_schemas` and then target the specific data set. If you need column definitions for a known table, always start by using `describe_table` before running any aggregation.

### Assuming immediate results

✗ AVOID

The agent runs an aggregate query but the user expects the answer instantly and gets frustrated when the chat is empty.

✓ INSTEAD

Remember that large queries run asynchronously. After using `execute_sql`, immediately follow up by asking to check the status using `statement_status` until it confirms completion.

### Manual schema exploration

✗ AVOID

The user has to manually navigate multiple console tabs to find out if a required column (like 'user_id') even exists in the target table.

✓ INSTEAD

Ask the agent to run `describe_table` on the suspected source table. This shows you the metadata and confirms the exact column name and data type instantly.

## The Right Fit

Use this MCP if your primary pain point is interacting with massive, secure data warehouses without switching tools. It excels when you need to perform ad-hoc aggregations or audit schema definitions via natural language conversation. Don't use it if the data source itself is lightweight, simple, or requires proprietary, non-SQL interaction (like calling a specific internal microservice). If your goal is merely viewing dashboards that already exist in Tableau or PowerBI, you don’t need this MCP; those tools are for visualization. However, if the goal is to *run* the logic and extract raw metrics from the underlying data warehouse, this is exactly what you need.

---

# Amazon Redshift MCP: Solving Data Aggregation Pain with SQL

Right now, calculating a simple quarterly metric involves logging into your cloud provider console. You find the data warehouse, select the database, navigate to the correct schema, and then write or paste your SELECT statement. If the query is big, you wait for it to process in a separate window, often needing to manually track its status before finally pulling the result set.

With this MCP, you simply prompt your agent: "Give me the total revenue grouped by state last quarter." The system handles the entire complex workflow—from initiating the massive job run to monitoring its completion and delivering the clean, final table right into our chat. You just get the answer.

---

# Amazon Redshift MCP: Schema Discovery for Data Warehousing

Before you can write a single query, you have to figure out what data exists and how it's structured. This means running through lists of schemas, then listing every table inside them, and finally digging into `describe_table` just to confirm if

This MCP automates that entire discovery cycle. You tell your agent what you need, and it methodically checks the environment for schemas, tables, and column definitions, giving you confidence in the data before you even write an aggregation query.

'purchase_date' is stored as a timestamp or a string.

# 7 Tools for Amazon Redshift Data Warehousing Analysis

Use these tools to execute SQL commands, discover schemas, check column details, and track every query run on your data warehouse.

| # | TOOL | DESCRIPTION |
|---|---|---|
| 01 | `describe_table` | Shows the data types and metadata for any selected table's columns. |
| 02 | `execute_sql` | Runs a full SQL statement asynchronously, giving you a unique job ID to track its progress. |
| 03 | `get_results` | Pulls the final rows of data for an SQL query after it has completed successfully. |
| 04 | `statement_status` | Checks if a previously executed SQL job is still running or if it finished with success or failure. |
| 05 | `list_schemas` | Retrieves a list of all database schemas available within the Redshift environment. |
| 06 | `list_statements` | Lists recent SQL query attempts to help audit past analytical workloads on the cluster. |
| 07 | `list_tables` | Retrieves a list of all tables residing within a specific, defined schema. |

# See It in Action

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

**U** What was the total number of active users last month?

**Total Active Users Report**

The system successfully queried the `user_metrics` table. The calculation shows:

| METRIC | VALUE |
|---|---|
| Total Unique Users | 1,452,901 |
| Monthly Growth Rate | +3.2% |

*Note: This figure reflects active users who logged in between the 1st and the last day of the previous month.*

**U** Can you list all tables available under the finance schema?

**Schema Contents: 'finance'**

The following tables are available for querying:

- `quarterly_revenue`: Contains aggregated sales figures.
- `user_cohorts`: Tracks user enrollment and retention scores.
- `payment_transactions`: Detailed record of individual payments processed.

U Show me the column details for the 'payment_transactions' table.

**Table: payment_transactions**

This table tracks every recorded transaction. Key metadata:

- `transaction_id` : UUID (Primary Key)
- `user_id` : Integer (Required)
- `amount` : Decimal (Money value, up to 2 places)
- `processed_at` : Timestamp (When the payment finished processing)

# Frequently Asked Questions

01 **How does Amazon Redshift MCP help me run big queries without losing my connection?**

It manages large jobs asynchronously. You ask for a complex calculation, and instead of waiting in the chat window, the job runs in the background using AWS's Data API. The agent keeps track of it so you can retrieve results when they are ready.

02 **Can I use Amazon Redshift MCP to find out what columns a table has?**

Yes, absolutely. You simply ask the agent to describe any table—like 'user_cohorts'. It will instantly pull up all the column names and tell you their data types (integer, timestamp, etc.) so you know exactly how to query them.

03 **What if I need to check historical data or past reports?**

The MCP keeps track of recent activity. You can ask it to list all executed statements, letting you audit who ran what and when. This is critical for compliance and troubleshooting old reports.

04 **Is Amazon Redshift MCP useful if I only need simple data lookups?**

While it handles simple lookups fine, its real value comes from complexity. If you're doing anything involving aggregation (sums, counts, averages) or joining multiple tables, this tool is built for that scale.

05 **Does Amazon Redshift MCP work with my current development environment?**

It connects via a secure, serverless API layer. This means you don't need to worry about local JDBC drivers or maintaining complex network pools in your code editor or chat interface.

# 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 `"amazon-redshift": { "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

# Amazon Redshift 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 Amazon Redshift. 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 | Amazon Redshift MCP |
| Server ID | `019d75fd-e620-705a-90ad-a1cfc63ce709` |
| 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/amazon-redshift.