MCP SERVER | NO CODE | CLOUD HOSTED

Neon MCP

Manage Postgres branching via conversation.

Neon MCP gives your agent full control over serverless PostgreSQL infrastructure. You can manage complex database projects, automatically clone entire datasets for testing, and audit resource usage without writing a single command line script. It helps you spin up isolated feature branches, provision new workspaces, or wipe out old environments just by talking to your AI client.

A+ Quality Score 100/100

postgresql | serverless | database-branching | cloud-native | autoscaling | infrastructure-as-code

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/neon-serverless-postgresql

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.

Neon (Serverless PostgreSQL) MCP

10 tools available
Cloud-hosted on Vinkius

Managing modern cloud databases is usually a nightmare of CLI commands and complex networking diagrams. This MCP lets you treat your entire serverless PostgreSQL environment like a conversation. You connect your Neon account through Vinkius, giving your agent the keys to everything: project structure, data branches, and user permissions.

Forget logging into ten different dashboards just to check storage caps or figure out where your application needs to point. Your AI client handles that complexity for you. Need to test a massive feature change? You can ask it to instantly create a zero-copy clone of the entire production dataset in an isolated branch. Want to verify if a user has permission to run certain queries? The agent lists and audits all roles, keeping your system secure. It's about getting immediate, conversational control over every part of your data stack—from provisioning whole new workspaces to inspecting deep execution footprints on specific branches.

Core Capabilities

01 — Analyze entire database environments

List all managed projects and retrieve metrics like regional caps and deployment stats for an overview of your entire infrastructure.

02 — Isolate data for testing

Instantly clone the entire production schema into a new, isolated branch. This lets developers test features against identical live data without touching the main database.

03 — Trace resource usage and schemas

Audit how much storage specific branches are using or map out all internal SQL schemas tied to a given project.

04 — Manage user access rights

List, audit, and manage the various PostgreSQL roles to ensure only authorized users can query specific data nodes.

05 — Provision or wipe out workspaces

Initialize brand new serverless projects or permanently delete entire database ecosystems with one command.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/neon-serverless-postgresql** — connect your AI agent in three steps.

01 Subscribe to this MCP on Vinkius and provide your Neon API Key.

02 Connect your preferred AI client (Claude, Cursor, etc.) to the service.

03 Ask your agent a question like, 'Can you create a testing branch from main?' and let it handle the rest.

The bottom line is that this MCP turns complex database administration into simple natural language chat commands.

Built For

This is for Platform Engineers, Backend Developers, and Database Administrators who get frustrated with manual CLI scripting or having to jump between multiple cloud dashboards just to run a basic test. You need full visibility into your database lifecycle.

Backend Developer

Uses the MCP to create temporary, isolated branches for feature testing and verifying schema changes without risking production data.

Database Administrator (DBA)

Monitors storage caps across multiple Neon projects, audits compute endpoint usage, and manages user roles securely from a conversational interface.

Platform Engineer

Automates the provisioning of new serverless workspaces or cleans up deprecated environments using single chat commands to maintain high reliability standards.

What Changes When You Connect

01 Spin up isolated testing environments instantly. Use `create_branch` to clone production data, allowing developers to test major features without ever impacting the live system.

02 Get full infrastructure visibility with `list_projects`. You can see all your workspaces and understand their regional deployment metrics in one chat session.

03 Debug connection issues quickly. Run `list_endpoints` to discover the exact routing addresses needed for your application drivers, eliminating guesswork.

04 Maintain security through conversation. Audit user access rights by calling `list_roles`, ensuring only authorized personnel can query sensitive data nodes.

05 Control your lifecycle efficiently. Use `delete_project` when a workspace is deprecated, or use `create_project` to spin up fresh environments on demand.

Real-World Applications

Testing a major API rewrite

A backend developer needs to test how a new payment module interacts with live data. Instead of manually setting up a staging database, they ask the agent to `create_branch` from main. The branch is ready in minutes, isolated and perfectly mirroring production.

Preparing for quarterly cleanup

A platform engineer knows several old development workspaces are unused. They use the MCP to run `list_projects`, identify the targets, and then issue a single command using `delete_project` to wipe out the entire ecosystem safely.

Debugging connection strings

The application team is having connectivity issues. The DBA asks the agent to run `list_endpoints` and `get_project`. The agent provides a list of active routing endpoints, solving the connection string problem instantly.

Compliance audit prep

A security officer needs to know exactly who can access the production data. They run `list_roles` on the specific branch and get an immediate audit report of all defined user identities and their permissions.

Patterns to Avoid

Manual schema mapping

✗ AVOID

Trying to manually map out every internal SQL database schema by logging into multiple dashboards or referencing old documentation.

✓ INSTEAD

Instead, ask the agent to `list_databases` on your target branch. It pulls all inherent schemas automatically and provides them in plain text.

Guessing connection points

✗ AVOID

Assuming that one database endpoint works for every part of your application stack when it might only cover a specific region or project.

✓ INSTEAD

Always run `list_endpoints` and verify the output with `get_project`. This confirms all active, usable routing connections.

Over-provisioning resources

✗ AVOID

Creating dozens of development workspaces just because 'it's easy to spin up a new one.' This leads to massive storage costs and management overhead.

✓ INSTEAD

Use `list_projects` first. If you need a temporary test, use `create_branch`. Only `create_project` when you genuinely need an entirely separate workspace.

The Right Fit

Use this MCP if your workflow involves managing the lifecycle of multiple serverless database environments—specifically if you deal with branching, provisioning, or auditing complex Postgres setups. If you constantly ask: 'What can I test without breaking production?' or 'Where is my connection string for this specific feature branch?', then this tool handles that complexity via conversation. Don't use it if all you need is to run a simple SELECT query on an already configured and stable database; in those cases, a direct connection using standard credentials is faster.

Debugging Postgres infrastructure used to mean clicking through five different tabs.

Today, debugging complex serverless databases feels like detective work. You have to jump from the project overview tab, over to the resource usage dashboard, then check which branch is active, and finally look at a separate schema inventory list—all while trying to figure out if the connection endpoint you found yesterday is still valid today.

With this MCP, you talk to your agent. You describe what's broken or what needs testing. The system handles the multi-step process: it identifies the correct branch, checks its resource footprint using `get_branch`, and tells you exactly which connection point to use. It just works.

Using Neon MCP gives you full control over your data branches.

Before this, setting up a test environment meant manual scripting: cloning the schema definition, configuring network access, and hoping the connection strings matched. It was slow, brittle, and prone to human error every single time.

Now you simply ask your agent to `create_branch`. The system handles the zero-copy duplication of petabytes of data instantly. You get an isolated testing ground ready for queries in minutes.

Neon (Serverless PostgreSQL) MCP: 10 Tools

These tools let you manage every aspect of your serverless database infrastructure—from listing workspaces to cloning production data.

#	TOOL	DESCRIPTION
01	`create_branch`	Clones an entire production dataset into a brand new, isolated testing branch for safe query execution.
02	`create_project`	Provisions and initializes a completely empty Neon Project serverless workspace.
03	`delete_project`	Permanently wipes out an entire database ecosystem, destroying all connected resources and data blocks.
04	`get_branch`	Provides a detailed report on the execution activity and resource footprint of one specific data branch.
05	`get_project`	Analyzes core routing logic for a project, providing details like regional metrics and storage size limits.
06	`list_branches`	Maps all available Copy-on-Write branches associated with a main project source.
07	`list_databases`	Identifies all internal SQL schemas and catalog namespaces mapped to a specific data branch, helping form correct connection strings.
08	`list_endpoints`	Discovers the active connection routing endpoints used by your application drivers across the project.
09	`list_projects`	Lists all managed serverless PostgreSQL projects and workspaces within the Neon account.
10	`list_roles`	Extracts a list of all active PostgreSQL user roles operating on a branch, showing who can query the data.

See It in Action

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

U List all serverless projects in my Neon account

I've retrieved your Neon projects. Currently, you have 3 workspaces: 'prod-cluster' (Region: AWS us-east-1), 'staging-env', and 'dev-sandbox'. Would you like to see the active branches for the production cluster?

U Create a new branch called 'feat-user-auth' from the 'main' branch

Spawning zero-copy clone... Success. I've created the 'feat-user-auth' branch (ID: br-12345) from your main source. This branch is now isolated and ready for queries. I'll provide the new compute endpoint URL for this branch immediately.

U What databases and roles are configured on branch 'br-12345'?

Retrieving branch inventory... On 'br-12345', I've identified 2 databases: 'main_app' and 'analytics'. There are 3 active roles: 'neondb_owner', 'app_writer', and 'readonly_user'. I can help you verify the specific permissions for any of these roles if you'd like.

Frequently Asked Questions

01 **How do I list all my Neon projects using the Neon MCP?**

You run `list_projects`. This tool gives you a complete map of every serverless workspace you own, letting you see which environments are active and where they are located.

02 **What is the difference between creating a project and creating a branch with Neon MCP?**

Creating a project (`create_project`) builds an entirely new, empty serverless workspace. Creating a branch (`create_branch`) clones all of the data from an existing source into a new testing area.

03 **Can I wipe out an entire database using Neon MCP?**

Yes, you use `delete_project`. This command permanently destroys every resource and connection associated with that project. Use it carefully; there's no undo.

04 **How can I find the correct connection string for my application?**

The agent uses `list_endpoints` to discover all active connection routing points across your Neon project, giving you the precise URLs needed for your drivers.

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 `"neon-serverless-postgresql": { "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

Neon (Serverless PostgreSQL) 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 Neon (Serverless PostgreSQL). 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	Neon (Serverless PostgreSQL) MCP
Server ID	`019d75dc-3168-7318-b86d-c2e7878c57ae`
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/neon-serverless-postgresql.