MCP SERVER | NO CODE | CLOUD HOSTED

Neon MCP

Manage serverless databases through chat.

Neon MCP lets your AI agent manage serverless PostgreSQL databases entirely through conversation. You provision projects, clone development branches instantly, create read-only endpoints, and set up database roles—all without touching a graphical console.

A+ Quality Score 100/100

postgresql | serverless | database-branching | auto-scaling | infrastructure-as-code | cloud-native

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

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 MCP

17 tools available
Cloud-hosted on Vinkius

You shouldn't have to click through five different tabs in the Neon dashboard just to set up a new feature environment. This MCP lets your AI agent handle all of that complex infrastructure work via natural conversation.

Need to spin up a test database for a pull request? You tell your agent, and it handles the cloning process using copy-on-write technology. It can create brand new projects in specific AWS regions or provision read-only compute endpoints for testing. Forget remembering connection URIs; just ask your AI client, and it delivers the ready-to-use psql string.

Managing a growing stack of services—from creating dedicated database roles to listing every active project—used to be a manual chore. Now, you simply talk to your agent. Because this MCP is hosted on Vinkius, connecting it to your existing workflow means all the power of Neon's serverless PostgreSQL infrastructure is available directly within your preferred AI client.

Core Capabilities

01 — Provision and Modify Projects

You can create entirely new Neon projects with specific AWS regions or PostgreSQL versions, or update existing project names.

02 — Manage Database Branches

Instantly clone a branch from any point-in-time using copy-on-write, set the primary development source, and delete old branches when they're done.

03 — Configure Endpoints and Connections

Provision both read/write compute hosts or dedicated read-only endpoints for specific branches, and fetch complete connection URIs for psql tools.

04 — Create and Control Resources

Build new databases within a branch, set up secure access by creating database roles with unique passwords, and list all existing resources across the project.

05 — Audit and Govern Access

Retrieve detailed lists of every database, role, and endpoint associated with a specific Neon branch or project for compliance checks.

One Click on Vinkius — From Prompt to Execution

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

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

02 Connect your AI client (Claude, Cursor, etc.) to the catalog. Your agent now has full control over your PostgreSQL infrastructure.

03 Simply ask your agent for a specific action, like 'Create a read-only endpoint for my staging branch' or 'List all database roles in Project X'.

The bottom line is you manage complex, multi-step infrastructure changes using simple natural language prompts.

Built For

This MCP is essential for the developer who gets frustrated clicking through 10 tabs in a dashboard just to set up an environment. It's built for engineers who need reliable, repeatable infrastructure provisioning without leaving their IDE.

DevOps Engineer

Managing CI/CD pipelines requires auditing project configurations and ensuring staging environments are properly branched and endpoint-configured.

Backend Developer

Needing to quickly provision isolated development branches, grab connection URIs for local testing, and create temporary test databases without manual setup.

Database Administrator (DBA)

Creating new secure database roles, managing resource lifecycle across multiple branches, and tracking PostgreSQL version upgrades via conversation.

What Changes When You Connect

01 Instant Environment Setup: Instead of manually clicking to clone a branch, you ask your agent to create an instant development environment from the primary source. This saves minutes on every feature build.

02 Zero-Touch Connectivity: Getting connection strings is simple. Your agent handles the `get_connection_uri` tool and provides psql-ready credentials immediately, saving copy/pasting time.

03 Granular Resource Control: You don't need to mess with global settings. Use tools like `create_database` or `create_role` to manage resources only within a specific branch context.

04 Simplified Scaling: Provisioning compute hosts is effortless. Your agent provisions either read-write endpoints (`create_endpoint`) or dedicated, low-cost read-only replicas on demand.

05 Full Lifecycle Management: Need to clean up? You can run `delete_project` or `delete_branch` commands via conversation, ensuring your infrastructure remains tidy and costs are controlled.

Real-World Applications

A developer needs a feature branch for testing.

The dev asks their agent to clone the main branch into 'feature-auth'. The MCP executes `create_branch`, instantly giving them an isolated, working environment without needing to manually copy settings or worry about data overlap.

A DBA needs to audit all credentials.

The DBA asks their agent to run `list_roles` and then `list_databases`. The agent gathers all the necessary role names, creation dates, and database containers in a single response for review.

An ops engineer needs to provision a staging read-only copy.

The engineer instructs their agent to create a `read_only` compute endpoint using `create_endpoint`. This ensures the testing team can hit production data without risking any writes.

A developer needs credentials for a microservice test.

Instead of visiting the console, the dev asks their agent to generate the connection string. The MCP uses `get_connection_uri` and returns the full, working URI immediately.

Patterns to Avoid

Trying to manage everything manually.

✗ AVOID

Copying project IDs from one screen into a separate terminal or documentation file just to create a new database or endpoint. This is slow and prone to copy/paste errors.

✓ INSTEAD

Use your AI agent to orchestrate the entire process. Start by listing all projects with `list_projects`, then ask the agent to handle subsequent actions like creating a branch using `create_branch` and provisioning an endpoint using `create_endpoint`.

Confusing resource deletion scope.

✗ AVOID

Thinking that deleting a database also cleans up its associated roles or endpoints, leading to orphaned, unusable resources.

✓ INSTEAD

Be specific with the agent. First, use `list_roles` and `list_databases` to verify all dependencies are accounted for before running `delete_database`. This prevents critical data loss.

Over-relying on default settings.

✗ AVOID

Creating a new project without specifying the AWS region or PostgreSQL version, leading to unexpected deployment environments.

✓ INSTEAD

Always specify your desired parameters when calling `create_project`. Providing the region and PG version upfront ensures environment parity.

The Right Fit

Use this MCP if you manage infrastructure—specifically serverless databases—and want that management done via natural conversation. If your workflow requires provisioning resources, cloning branches, or generating connection strings across multiple steps, this is ideal. Don't use it if your task is simply reviewing raw SQL code syntax; for that, a dedicated coding tool works better. Also, don't use it if you need to implement complex network routing rules outside of the standard Neon endpoint configuration, as the MCP focuses purely on resource provisioning and administration via conversational commands like `create_endpoint` or `set_primary_branch`. This is your go-to for infrastructure *setup* and *auditing*, not deep runtime debugging.

Database setup used to feel like a maze of clicking through consoles.

Setting up a new feature environment means jumping between the project overview, then creating a branch in the settings panel, finding the right endpoint configuration, and finally grabbing the connection URI. It's five clicks deep just to get started on a single pull request.

With this MCP, you simply tell your agent what you need—for example, 'I need an isolated testing environment.' The agent handles the entire sequence of operations: cloning the branch, setting up the compute endpoint, and getting the connection string. You get back instant operational capability.

Neon MCP delivers database control through natural language.

You eliminate the need to manually audit resource states across multiple tabs. Instead of running three different queries in a dashboard, you prompt your agent to `list_endpoints` and `list_roles`, getting all necessary information compiled instantly.

This changes everything. Infrastructure provisioning is no longer a checklist of clicks; it's a single conversation that delivers fully configured, ready-to-use resources.

Neon – 17 Tools for Database Management

Use these tools to orchestrate every aspect of your database lifecycle, from creating new workspace projects to generating specific connection URIs.

#	TOOL	DESCRIPTION
01	`get_connection_uri`	Retrieves the full connection string for a specified or default Neon branch, ready for use in psql tools or applications.
02	`create_branch`	Clones an instant new development branch from another point-in-time using copy-on-write technology.
03	`create_database`	Establishes a brand new, isolated database container within an existing Neon branch.
04	`create_endpoint`	Provisions the necessary compute host for a specific branch, choosing either full read-write or limited read-only access.
05	`create_project`	Initializes an entirely new Neon workspace, setting the region and PostgreSQL version from scratch.
06	`create_role`	Creates a dedicated, authenticated user role within a branch and generates its initial secure password.
07	`delete_branch`	Permanently deletes an entire isolated branch, destroying all associated data. (Cannot delete the primary branch).
08	`delete_project`	Deletes all resources within a project workspace; note that this action is irreversible.
09	`get_branch`	Retrieves detailed metadata for a specific Neon branch using its ID and parent project's ID.
10	`get_project`	Fetches all configuration details for an entire Neon project workspace.
11	`list_branches`	Lists every existing branch within a project, detailing its status and parent ID.
12	`list_databases`	Provides a list of all databases contained within a specific Neon branch.
13	`list_endpoints`	Lists the compute endpoints for a given branch, detailing their type and current connection state.
14	`list_projects`	Retrieves an overview of every Neon project account you manage, including region and version details.

#	TOOL	DESCRIPTION
15	list_roles	Lists all users (roles) that have been created within a specific branch for access control.
16	set_primary_branch	Promotes one existing branch to be the default primary source for new development work in the project.
17	update_project	Allows you to change the human-readable name of an entire Neon project workspace.

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 Neon projects and show me which regions they're in.

I found 3 projects in your Neon account: 'prod-analytics' (aws-us-east-2, PG 16), 'staging-api' (aws-eu-central-1, PG 15) and 'dev-sandbox' (aws-us-east-2, PG 17). Would you like more details on any of them?

U Create a new branch called 'feature-auth' from the primary branch of my project.

I've created branch 'feature-auth' (br-wispy-leaf-789012) in your project. It was cloned instantly from the primary branch using copy-on-write. The branch is ready with its own compute endpoint and databases. Want me to get the connection URI?

U Get the connection URI for the main branch of my project.

Here's your connection URI: `postgresql://neondb_owner:xxxxxxxx@ep-cool-darkness-123456.us-east-2.aws.neon.tech/neondb?sslmode=require`. You can use this with psql, your ORM or any PostgreSQL client. The password was auto-generated by Neon.

Frequently Asked Questions

01 **How do I start using the Neon MCP to manage my databases?**

You first subscribe to this MCP on Vinkius and enter your unique API Key. Then, you can instruct your agent to list all projects with `list_projects` to begin.

02 **Can I use Neon MCP to create a staging copy of my data?**

Yes, you use the `create_branch` tool. This instantly clones an isolated development branch from your primary source using copy-on-write technology.

03 **What if I need credentials for a new service user? Do I use Neon MCP?**

Yes, you create the user by calling `create_role`. You specify the project and branch IDs needed to set up the dedicated role and its password.

04 **Is there an easy way to get a connection URI for my main development branch?**

You use the `get_connection_uri` tool. You just need to ask your agent, and it returns the full psql string ready for immediate use.

05 **Does Neon MCP handle resource cleanup when I delete a project?**

The `delete_project` tool deletes all associated resources—branches, databases, and endpoints. Remember that this action destroys all data in the project.

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": { "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 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. 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 MCP
Server ID	`019d845e-704f-71d8-8f65-b644ba0f2118`
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.