MCP SERVER | NO CODE | CLOUD HOSTED

Supabase MCP for AI Agents

Manage your PostgreSQL database and Supabase backend records with natural language.

Supabase MCP lets your AI client interact directly with your database. You can run queries, manage user accounts, and trigger PostgreSQL functions using plain English. It gives your agent full-privileged access to your backend infrastructure for faster debugging and data management.

A+ Quality Score 100/100

postgresql | backend-as-a-service | database-management | api-generation | row-level-security

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JULY 2026
vinkius.com/mcp/supabase

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.

Supabase MCP

13 tools available
Cloud-hosted on Vinkius

This Supabase MCP lets you handle database operations without constantly switching tabs. Instead of manually writing SQL queries or hunting through the Supabase dashboard to find a specific user, you can just tell your agent what to do. It uses your service role key to act as a privileged admin, meaning it can run complex PL/pgSQL functions, audit your user roster, or check your storage buckets on the fly. This makes it way faster to debug your environment and iterate on your data because you stay in your terminal or IDE. You can find this connection along with thousands of others in the Vinkius catalog to keep your tools in sync. It handles the heavy lifting of the backend infrastructure so you can focus on building features instead of managing rows.

Core Capabilities

01 — Query database records

Pull specific data sets into your chat using natural language.

02 — Insert new data rows

Add new records to your tables to populate your database.

03 — Update existing records

Modify specific rows in your database without writing manual SQL.

04 — Delete specific rows

Remove old or unnecessary data from your tables instantly.

05 — Execute custom PostgreSQL functions

Trigger complex backend logic using your pre-compiled RPCs.

06 — Manage storage buckets

Audit and organize your file architecture in your storage containers.

07 — Audit authenticated users

Fetch and manage your user roster from Supabase Auth.

One Click on Vinkius — From Prompt to Execution

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

01 Add the Supabase MCP to your AI client configuration.

02 Enter your SUPABASE_URL and SUPABASE_SERVICE_ROLE_KEY.

03 Ask your agent to perform database actions or check storage.

The bottom line is you get a conversational interface for your entire Supabase backend.

Built For

The backend engineer who is tired of manually updating rows in the dashboard at 2am or the dev who needs to iterate on data quickly without context switching.

Database Administrator

Running complex validation queries and invoking custom RPCs during analytical reviews.

Backend Developer

Iterating on database structures and mocking rapid data insertions without opening a browser.

Full-Stack Engineer

Resolving user authorization bugs by tracking dependencies directly in the IDE.

What Changes When You Connect

01 Pull data instantly using db_select to see records without writing manual SQL.

02 Populate your tables with mock data using db_insert to speed up development.

03 Trigger complex logic with db_rpc to run existing PostgreSQL functions.

04 Audit your user base with list_auth_users to find and manage accounts.

05 Organize your files using list_storage_buckets to see your entire storage map.

06 Clear out old assets with delete_storage_file to keep your buckets clean.

Real-World Applications

Finding stuck accounts

A user can't log in, so you ask the agent to list all users. It uses list_auth_users to find the account and check its status.

Restocking inventory

You need to update inventory. You ask the agent to run the restock function, and it uses db_rpc to update the values.

Verifying storage

You want to make sure your images are uploaded. The agent uses list_storage_buckets to check the status of your files.

Data cleanup

You need to remove old test data. The agent uses db_delete to wipe out the specific rows you identify.

Patterns to Avoid

Using for large migrations

✗ AVOID

Trying to move millions of rows using db_insert.

✓ INSTEAD

Use a dedicated migration script for large data moves. This MCP is best for iterative development and targeted updates.

Exposing public keys

✗ AVOID

Using a public key instead of the service_role key.

✓ INSTEAD

Always use the service_role key in your config to give the agent the correct permissions to bypass row-level security.

Deleting without filters

✗ AVOID

Running db_delete without a match query.

✓ INSTEAD

Always include a specific match_query to target the exact rows you want to remove to prevent accidental data loss.

The Right Fit

Use this if you need to perform administrative tasks on your Supabase backend like auditing users, running RPCs, or managing storage. It is perfect for rapid prototyping and debugging where you need to see data or change states quickly. Don't use it for high-frequency production UI actions where you need row-level security. If you just need to read data with limited permissions, a standard frontend client is a better fit. This MCP is for the person who wants to treat their database like a conversation.

Supabase MCP for PostgreSQL Database Management

Developers often waste time switching between their code editor and the Supabase dashboard. You have to copy IDs, paste them into the SQL editor, hit run, and then copy the results back. It's a constant cycle of context switching that breaks your flow.

With this MCP, you just tell your agent what you need. It handles the SQL, pulls the data, and shows it to you in the chat. You get your answers instantly without ever leaving your workspace.

Supabase MCP for Backend Infrastructure Operations

This means no more manual row updates or hunting through the storage console to find a specific bucket. You can check auth statuses or trigger complex backend functions with a single sentence.

You get a direct line to your backend infrastructure that actually understands your intent.

Supabase MCP with 13 tools for PostgreSQL Management

Query records, manage users, run RPC functions, and audit storage buckets using natural language.

#	TOOL	DESCRIPTION
01	delete_auth_user	Permanently remove an authenticated user from the system. Use this to clean up your user roster.
02	create_storage_bucket	Create a new storage bucket with optional public access. This is great for setting up new asset folders.
03	delete_storage_file	Delete specific files from a storage bucket using a path list. Use this to manage your storage space.
04	get_auth_user	Retrieve detailed information for a specific authenticated user. This helps you debug account-specific issues.
05	list_auth_users	List all authenticated users from Supabase Auth. Use this to see who has an account in your system.
06	db_count	Count rows in a database table for pagination or analytics. This helps you get quick totals on your data.
07	db_delete	Remove rows from a database table based on a mandatory match query. This helps you clear out old data quickly.
08	db_insert	Add a new row to a database table by providing a JSON payload. This is perfect for populating your tables with mock data.
09	db_rpc	Execute a Supabase Postgres Function with dynamic JSON arguments. This lets you run your own custom backend logic using natural language.
10	db_select	Query records from any PostgreSQL database table using PostgREST syntax. This helps you pull specific data sets into your chat instantly.
11	db_update	Modify existing rows in a database table using a match query. Use this to update specific records without writing manual SQL.
12	list_storage_buckets	See all available storage containers in your project. This helps you audit your file architecture.
13	list_storage_files	View the files currently stored inside a specific bucket. Use this to check your storage contents.

See It in Action

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

U Can you show me the last 5 users who signed up?

Checking your Supabase Auth roster now...

USER ID	EMAIL	CREATED AT
user_882	dev_test@example.com	2026-07-14 10:00:00
user_881	admin_test@example.com	2026-07-14 09:45:00
user_880	guest_user@example.com	2026-07-14 09:30:00
user_879	beta_tester@example.com	2026-07-14 09:15:00
user_878	early_adopter@example.com	2026-07-14 09:00:00

Found 5 recent registrations.

U Run the restock function for product IDs 50 and 60.

Executing the `restock_items` RPC with the requested IDs...

Request Payload:
- Target IDs: `[50, 60]`
- Function: `restock_items`

Response:
✅ Success. The inventory for products 50 and 60 has been updated in the database.

U Are there any buckets in my storage that are set to public?

Scanning your storage buckets...

Bucket Name	Public Access
profile_images	✅ Public
internal_assets	❌ Private
user_uploads	❌ Private

Only the `profile_images` bucket is currently set for unauthenticated public reads.

Frequently Asked Questions

01 **How does the Supabase MCP help with database tasks?**

It lets you perform database actions like querying, inserting, and updating records using natural language. Instead of writing SQL, you just ask your agent to do it for you.

02 **Can I use the Supabase MCP to manage my users?**

Yes, you can audit your user roster, get details on specific accounts, or remove users from your system directly through your AI client.

03 **Is it safe to use the Supabase MCP for backend edits?**

It uses your service role key to perform actions, which gives it administrative access. It is designed for backend developers and DBAs who need to manage the infrastructure securely.

04 **Can the Supabase MCP help with my storage buckets?**

Yes, it can list all your buckets, show you the files inside them, and even create new buckets or delete specific files.

05 **How do I connect my Supabase backend to my AI client?**

You just need to add the Supabase MCP to your configuration and provide your Supabase URL and service role key. From there, your agent can start interacting with your data.

06 **What kind of queries can the Supabase MCP run?**

It can run any query that your Supabase backend supports, including standard row lookups, complex filters, and custom PostgreSQL functions via RPC.

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 `"supabase": { "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

Supabase 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 Supabase. 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	July 2026
MCP Server	Supabase MCP
Server ID	`019d760e-ec68-71c0-8d3d-bf79f9b615cb`
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/supabase.