MCP SERVER | NO CODE | CLOUD HOSTED

Amazon DynamoDB Table MCP for AI Agents

Manage structured NoSQL records and complex database queries

The Amazon DynamoDB Table MCP gives your AI agent one focused superpower: secure, controlled access to a single NoSQL database table. It lets your agent read records using `get_item` or complex searches with `query_table`, and it handles writing data by inserting new records via `put_item` or removing them entirely with `delete_item`. This is built for giving AI applications reliable, contained data persistence without exposing your entire AWS infrastructure.

F Quality Score 3.6/100

nosql | aws | database-management | data-storage | security-scoping | serverless

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

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 DynamoDB Table MCP

5 tools available
Cloud-hosted on Vinkius

Need to give your AI agent access to structured data but can't risk handing over global cloud permissions? This MCP solves that. It wraps up all the necessary DynamoDB interactions into one secure connection, strictly limiting the agent to a single table. You can now let your AI client perform complex database tasks—like fetching user profiles or tracking chat histories—without ever touching your critical production databases. The agent uses dedicated tools for everything: it pulls specific records using `get_item`, runs targeted data searches with `query_table`, and adds new information whenever you use the `put_item` function. If you need to clean up old entries, it handles that too, letting the agent run `delete_item`. Because Vinkius hosts this MCP, you connect once from your preferred AI client (like Cursor or Claude) and get immediate, safe database access for any application.

Core Capabilities

01 — Retrieve specific records

The agent retrieves a single item's data using the `get_item` tool.

02 — Insert or update records

You can add new entries into the table using `put_item`, or modify existing ones.

03 — Perform targeted data queries

The agent runs complex, filtered searches across related items with `query_table`.

04 — Scan the entire table content

You can execute a full scan of all data within the DynamoDB table using `scan_table`.

05 — Remove stored items

The agent deletes specific entries from the table using the `delete_item` tool.

One Click on Vinkius — From Prompt to Execution

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

01 First, you connect your AI client to this MCP via Vinkius and tell it which DynamoDB table needs access.

02 Next, you ask your agent a question like 'Find all inactive user accounts.' The agent determines that `query_table` is the right tool and calls it with necessary filters.

03 The agent runs the query against the database and sends the structured results back to your AI client for review.

The bottom line is, you're giving your AI a safe API endpoint that speaks DynamoDB, allowing natural language commands to become reliable data actions.

Built For

This MCP is essential for any developer or data architect building applications powered by large language models. If you're building an internal tool or a customer-facing app that needs persistent memory, this gives your agent the database backbone it requires without compromising security.

Full Stack Developer

You build and test the connection logic. You need to make sure your AI client can reliably use `put_item` to save state, or call `get_item` when fetching user data.

Data Engineer

You manage how the LLM interacts with the database structure. You'll use `query_table` and `scan_table` to validate that the agent is extracting exactly the right subset of information.

AI Product Manager

You define the data requirements for your AI application. You rely on this MCP's contained power to prove that the agent can manage structured memory, like processing an entire chat history or order queue.

What Changes When You Connect

01 Security: By using this MCP, your agent is locked down to one table. It can't list other tables or drop production data.

02 Precision: Instead of vague instructions, you use `get_item` for direct lookups or `query_table` for highly filtered searches.

03 Reliability: You get native DynamoDB integration that supports complex indexes and structured memory storage for your AI application state.

04 Efficiency: When bulk data cleanup is needed, the agent can run a full audit with `scan_table` before executing targeted deletions using `delete_item`.

05 Simplicity: It gives you instant, scalable NoSQL database access without needing to manage complex AWS credentials or networking setup.

Real-World Applications

Tracking chat history for a support bot

A user asks their agent to recall the context of last week's conversation. The agent uses `get_item` based on the user ID and date range, pulling up the complete thread structure so it can respond accurately without manual intervention.

Building a product catalog database

The team needs to populate 500 new items. They instruct the agent to run `put_item` repeatedly with structured JSON data, allowing them to build out their entire NoSQL inventory in bulk and maintain data integrity.

Identifying old or stale user accounts

The ops engineer needs a list of all users who haven't logged in for six months. They ask the agent to `query_table` using time filters, generating a report that helps them plan targeted re-engagement campaigns.

Auditing data before migration

Before moving to a new database, a developer asks the agent to `scan_table`. This provides a complete snapshot of all current records, allowing them to validate schema compliance and check for missing fields across the entire dataset.

Patterns to Avoid

Giving full AWS database access

✗ AVOID

The developer connects the AI agent with generic IAM credentials that allow it to list, modify, or delete data across *all* DynamoDB tables in the account.

✓ INSTEAD

Instead, connect the MCP. This limits the agent's power only to the single table you specify. Use tools like `get_item` and `query_table` for safe operations.

Writing rigid SQL queries

✗ AVOID

Trying to force the AI agent to use standard relational database query language, which doesn't map cleanly to NoSQL structures.

✓ INSTEAD

Let the specialized tools handle it. Use `query_table` for filtered lookups or `scan_table` when you genuinely need all data. Don't try to make DynamoDB act like a SQL engine.

Assuming writes are safe

✗ AVOID

The agent runs an unverified batch operation that calls `put_item` with partial or incorrect schema, corrupting the table's data structure.

✓ INSTEAD

Always review the payload before execution. Use a combination of `get_item` to verify existing fields and then use `put_item` only when the input data is fully validated by your client application.

The Right Fit

Use this MCP if you are building an AI-driven application that requires persistent, structured storage in a NoSQL format. You need to let your agent perform CRUD operations (Create, Read, Update, Delete) on a single, defined dataset. The scoped nature is key here; it's designed for isolation. Don't use this if your primary goal is complex relational joins across multiple different data sources—you might need an integration that connects several services. Also, don't use this if you need the AI to manage user permissions or access system metadata, as its power is strictly limited to item manipulation within one table. When in doubt about security, scope it down; this MCP provides maximum safety with minimum required access.

Amazon DynamoDB Table MCP for AI Agents: Managing NoSQL Data Persistence

Before this tool, connecting an LLM to a database meant granting massive permissions. Developers often had to write complex middleware that handled connection pooling, schema validation, and security scoping—all before the prompt even got sent. It was tedious copy-pasting of boilerplate code just to make sure the AI agent couldn't accidentally wipe out the whole production environment.

Now, this MCP does the heavy lifting. You simply connect your client, and it provides surgical access. The agent can execute `get_item` or `put_item`, knowing that its actions are confined to one table. This makes building robust, secure memory for your AI application straightforward.

Amazon DynamoDB Table MCP for AI Agents: Executing Complex Data Queries

Manually writing the necessary query logic is a pain. You have to write code that checks if the user asked for an ID lookup (`get_item`) or if they needed a list of related items (`query_table`). This requires complex conditional branching in your backend services.

With this MCP, you just tell the agent what you want. It handles the logic internally, allowing it to run `query_table` with specific filters based on natural language input. Your code stays clean; the intelligence is handled by the tool.

Amazon DynamoDB Table: 5 Tools for NoSQL Data Management

Your agent can read specific data (`get_item`), run targeted searches (`query_table`), insert new records (`put_item`), delete old entries (`delete_item`), or scan the whole table (`scan_table`).

#	TOOL	DESCRIPTION
01	`delete_item`	Removes a specific item record from the DynamoDB table.
02	`get_item`	Retrieves a single, specific item by its unique identifier in the table.
03	`put_item`	Adds a brand new record to the table or overwrites an existing one with new data.
04	`query_table`	Executes a targeted search query across multiple related items in the table.
05	`scan_table`	Reads every single item in the entire DynamoDB table, useful for full audits or bulk data checks.

See It in Action

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

U Find all user records associated with account ID 'CUST-456' that have status 'Suspended'.

User Search Results for CUST-456

- **Record 1:** Name: Sarah Connor, Email: s.connor@example.com, Status: Suspended, Last Access: 2023-01-15
- **Record 2:** Name: John Doe, Email: jdoe@corp.net, Status: Suspended, Last Access: 2024-02-01

(Found 2 records matching the criteria.)

U I need to add a new order for product 'SKU-987'. The customer is Jane Smith.

Order Record Added Successfully

We've inserted the following details into the database:

- **OrderID:** ORD-2024-112
- **Product:** SKU-987 (Widget Pro)
- **Customer:** Jane Smith
- **Status:** Pending Payment
- **Timestamp:** 2024-06-13

U Scan the table and list all records that are marked as 'Draft' or 'Archived'.

Table Scan Audit Report

I ran a full scan and found 12 items matching your request. Here is a summary:

ID	TYPE	STATUS
R-101	Project	Draft
R-450	User	Archived
R-992	Content	Draft

Frequently Asked Questions

01 **How do I safely let my AI agent access my database without giving away all AWS permissions?**

You use the Amazon DynamoDB Table MCP. It gives your AI client limited, surgical access to only one table. You can perform necessary read/write operations using tools like `get_item` and `put_item`, but nothing more.

02 **Can this MCP handle complex data searches or just simple lookups?**

It handles both. For single records, you use `get_item`. If you need to filter by multiple criteria (e.g., status AND date range), the agent runs a targeted search using `query_table`.

03 **Is this suitable for storing application state or chat history?**

Yes, absolutely. This is ideal for giving your AI client persistent memory. You can save conversation threads and application settings by writing new records with `put_item`.

04 **If I want to delete old data, how do I do it with Amazon DynamoDB Table MCP?**

You first use `query_table` or `scan_table` to find the IDs of records you want gone. Then, you tell the agent to run `delete_item` on those specific IDs to clean up the data safely.

05 **Does this MCP work with all my AI clients like Cursor and Claude?**

Yes. As long as your client is MCP-compatible, you can connect it here. It lets any connected agent route data operations through natural language commands.

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-dynamodb-table": { "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 DynamoDB Table 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 →

INDEPENDENT PLATFORM DISCLAIMER

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	Amazon DynamoDB Table MCP
Server ID	`019e3862-5136-7385-8c22-b74ce49d3dd9`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE