MCP SERVER | NO CODE | CLOUD HOSTED

OceanBase MCP

Audit, manage, and map your entire database infrastructure.

OceanBase MCP connects your AI agent directly to a distributed relational database infrastructure. It lets you manage complex data systems—including clusters, tenants, and resource usage—using natural conversation instead of clicking through console dashboards. You can audit performance, check configurations, and map out entire data architectures instantly.

A+ Quality Score 100/100

distributed-database | relational-database | cluster-management | database-auditing | sql | infrastructure-as-code

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

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.

OceanBase MCP

10 tools available
Cloud-hosted on Vinkius

Stop navigating endless cloud consoles just to find basic cluster status or resource limits. This MCP gives your AI client direct access to OceanBase's core database functions. Instead of manually checking separate tabs for tenant details or running multiple reports on resource consumption, you talk to your agent about what you need. It acts as a real-time database reliability assistant.

Need to know if the production environment is stable? Ask it. Want to audit data structures across different teams? You can ask that too. Your agent handles everything from listing all available clusters to retrieving detailed resource statistics, keeping your entire data infrastructure accurate and performant. Since you're connecting this via Vinkius, you get a single entry point for hundreds of services, meaning one connection gives your AI client access to the full scope of database operations.

Core Capabilities

01 — Inventorying Database Infrastructure

List all active clusters and retrieve their specific configurations and current status.

02 — Auditing Tenants and Databases

Browse logical tenants within a cluster or list all databases belonging to a specified tenant for asset tracking.

03 — Tracking System Performance

Get aggregated resource usage statistics across the entire organization, checking CPU, memory, and storage limits.

04 — Mapping Ecosystem Components

Identify core organizational elements like projects, instances, and workspaces to maintain a unified view of your setup.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP in Vinkius and enter your OceanBase Access Key ID and Secret.

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

03 Use natural language to prompt your agent with a query about your database infrastructure.

The bottom line is that you talk naturally to your AI client, and it translates that request into complex database commands using this MCP.

Built For

This connector is for the DBAs and Infrastructure Engineers who are sick of spending their mornings clicking through five different cloud provider dashboards just to audit basic performance metrics. It's also perfect for Data Architects who need to coordinate tenant strategies across multiple teams without ever leaving their chat window.

Database Administrator (DBA)

Running capacity planning reviews and auditing database health across a global deployment by asking the agent to list all clusters or check resource usage.

Infrastructure Engineer

Automating cluster audits and checking system performance by querying detailed configuration status for specific tenants.

Data Architect

Coordinating tenant strategies and mapping out database structures directly from their AI-powered workspace to plan new data services.

What Changes When You Connect

01 Instead of running separate commands to list clusters and then checking their status manually, you simply ask the agent, which uses `list_clusters` to give you a consolidated view immediately.

02 When capacity planning is necessary, the ability to use `get_resource_stats` means you get immediate, aggregate performance numbers without logging into multiple monitoring dashboards.

03 You can quickly understand data boundaries by using `list_tenants` and then following up with `list_databases`, instantly identifying every data asset in a specific area.

04 The agent keeps track of your entire ecosystem. Beyond databases, you get to view projects and workspaces using `list_projects` and `get_workspaces`, giving full context to any audit.

05 You don't have to guess which resources are running low. By combining `get_tenant_details` with performance checks, the agent points directly to where limits are being hit.

Real-World Applications

Auditing a New Department's Data Scope

A data architect needs to scope out what data exists in a new client department. They ask the agent to first run `list_tenants` to find the departmental container, then use `list_databases` on that tenant. The agent responds by listing all relevant databases and asking if they need detailed configuration for any.

Checking Global Performance During Peak Load

An infra engineer notices strange latency spikes during peak hours. They prompt the agent to run `get_resource_stats`. The agent immediately responds with CPU and memory usage percentages, allowing them to pinpoint if a specific area needs immediate scaling.

Troubleshooting Instance Connectivity

A backend developer reports that an application is slow. Instead of manually checking the console, they ask the agent about the instance details. The agent uses `get_instance_details` and provides all necessary connectivity metrics right in the chat.

Mapping Out a Multi-Tiered Application

A data architect needs to document every piece of software connected to OceanBase. They ask the agent to use `list_projects` first, then check all available workspaces with `get_workspaces`, building a complete map of dependencies.

Patterns to Avoid

Assuming one command does it all

✗ AVOID

Trying to get both the list of tenants and their resource usage by running only 'Show me my data.' This vague prompt leaves you with no actionable information.

✓ INSTEAD

Be specific. Start by asking the agent to run `list_tenants` to see who's in the cluster, then follow up by requesting detailed resource stats using `get_tenant_details` for a specific tenant.

Confusing project scope with data scope

✗ AVOID

Mistaking a listed project name for an actual database name. You might get a list of projects but then assume the next thing you see is a database, which isn't true.

✓ INSTEAD

When looking at structure, first use `list_projects` to map out the container level, and only then follow up with `list_databases` once you know the specific tenant name.

Ignoring historical data context

✗ AVOID

Only checking current resource usage (`get_resource_stats`) without knowing if that is normal for this time of day. The number looks fine, but it might be wrong.

✓ INSTEAD

Use your agent to list the clusters first with `list_clusters`, and then ask about historical data or expected performance ranges for those specific clusters.

The Right Fit

You should use this MCP if you spend time manually navigating cloud provider consoles, clicking through dashboards just to gather audit metrics, or having to copy-paste IDs from one screen to another. It's ideal when your primary need is querying the state of a large, distributed database infrastructure—checking resource limits (`get_resource_stats`), mapping out tenants (`list_tenants`), or verifying cluster configurations (`get_cluster_details`).

Don't use this if you are just trying to write a simple SQL query against a single table. For that, your agent can connect directly via standard SQL tools. You need the MCP when the problem isn't the data itself, but figuring out *where* the data is, *who* owns it, and *if* the infrastructure around it can handle the load.

Database auditing used to be a nightmare of tabs and dropdown menus.

Today, running a full audit means logging into the console. You click 'Clusters,' then you select 'Tenants.' Then you open another tab for resource monitoring. If you want to check every database asset, you have to manually drill down through multiple layers of menus, copying names and IDs along the way. It's slow, prone to human error, and takes at least an hour.

With this MCP, that manual process disappears. You simply tell your agent: 'Show me all databases for Tenant X and check their resource usage.' The agent handles the entire multi-step audit trail in a single conversation, giving you immediate, comprehensive answers.

OceanBase MCP gives you clear visibility into every component.

Manual tasks that vanish include listing all clusters using `list_clusters` and then having to individually verify the status of each one. You no

Now you get a unified view. Your agent doesn't just report data; it organizes your entire database

longer have to treat each cluster like a separate, forgotten silo.

ecosystem—from projects down through individual databases—into actionable intelligence.

OceanBase With 10 Tools

Use these tools to manage everything from listing core clusters to checking granular resource statistics across your distributed OceanBase environment.

#	TOOL	DESCRIPTION
01	`get_cluster_details`	Retrieves the specific details and current status of a single designated cluster.
02	`get_instance_details`	Gets detailed information about an individual database instance within the system.
03	`get_resource_stats`	Pulls combined resource usage numbers, including CPU and memory, for system performance checks.
04	`get_tenant_details`	Retrieves the specific metadata about a logical tenant's allocated resources.
05	`get_workspaces`	Lists and retrieves information for all available account workspaces.
06	`list_clusters`	Generates a list of every active database cluster in your OceanBase account.
07	`list_databases`	Lists all databases that exist within a specific tenant's scope.
08	`list_instances`	Provides a list of all individual OceanBase database instances.
09	`list_projects`	Lists all active projects defined within the OceanBase environment.
10	`list_tenants`	Generates a comprehensive list of all tenants managed by the cluster.

See It in Action

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

U List all database clusters in my OceanBase account.

I've retrieved your OceanBase clusters. You have 3 active clusters, including 'Prod-Cluster-01' and 'Dev-Sandbox'. Would you like to view the tenants for the production cluster?

U Show me the resource usage statistics for the organization.

I've retrieved the resource summary. Your organization is currently using 45% of allocated CPU capacity and 62% of memory across all instances. Storage usage is stable at 1.2TB. Should I check for any tenants hitting their limits?

U List all databases in tenant 'tenant-8821' inside cluster 'cluster-9920'.

I've retrieved the databases for the specified tenant. There are 5 logical databases, including 'InventoryDB', 'CustomerPortal', and 'AnalyticsRaw'. Would you like detailed configuration for any of them?

Frequently Asked Questions

01 **How do I check my resource usage with OceanBase MCP?**

Use the `get_resource_stats` tool. You simply ask your agent to retrieve aggregate resource statistics, and it will report CPU, memory, and storage usage across the entire organization.

02 **What if I need to see all my clusters? Do I use list_clusters?**

Yes, that's right. Running `list_clusters` is the command you use when you need a complete inventory of every active database cluster in your account.

03 **Does OceanBase MCP help me find specific data structures?**

Absolutely. You can start by listing tenants with `list_tenants`, then pinpoint the exact databases using `list_databases` within that tenant's scope.

04 **Can I see details about a single database instance?**

Use the `get_instance_details` tool. This function pulls all necessary technical metrics and configurations for one specific OB instance so you can troubleshoot it.

05 **What is the difference between list_projects and get_workspaces?**

Think of projects as a grouping mechanism, which `list_projects` shows you. Workspaces are related development containers that `get_workspaces` helps you identify.

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 `"oceanbase": { "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

OceanBase 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 OceanBase. 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	OceanBase MCP
Server ID	`019d8462-437a-725a-b988-4673481ed9f9`
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/oceanbase.