

Frequently asked questions

Precise technical answers. No marketing-speak.

Product

What exactly is AST parsing and why is it better than an allowlist?

An operation allowlist blocks based on the first token of the query: if it starts with DELETE, it blocks; if it starts with SELECT, it allows. It's an instrument with no granularity.

AST (Abstract Syntax Tree) parsing builds the complete syntax tree of the query. Vericto traverses that tree looking for nodes that violate the rules. This allows detecting:

- `UPDATE products SET price = 0` without a WHERE clause (UpdateStmt > WhereClause = NULL)
- `DELETE FROM users` without WHERE at any nesting level
- `TRUNCATE` disguised inside a CTE
- `DELETE FROM orders LIMIT 0`, a DELETE that deletes nothing but is syntactically dangerous

An allowlist cannot detect any of these cases because they all start with tokens that appear safe on the surface.

How can I connect Vericto?

Five ways, at two moments:

- **TCP proxy** (runtime): point your connection string at the proxy instead of your database. It intercepts each query on the wire protocol (Postgres/MySQL) and blocks inline. Zero code changes.
- **Runtime API** (runtime): your application sends each query to Vericto before running it, one per call, and runs it only if the answer is ALLOWED. Useful for Oracle and SQL Server, or when you cannot change the connection string.
- **CLI** (before execution): the `vericto` binary validates SQL in pre-commit hooks and CI/CD, with an exit code based on the verdict.
- **Validation API** (before execution): send files or batches of SQL over HTTP and get a verdict for each statement, with nothing to install.
- **MCP** (before execution): AI agents such as Claude Code or Kiro validate SQL while they write it.

All of them are available on every plan, including the free one. See [Integrations](#) to choose.

Do I have to modify my application code to use Vericto?

No. Vericto acts as a transparent TCP proxy. You only change the host in your connection string:

```
- DATABASE_URL=postgres://user:pass@prod-db.host:5432/db
+ DATABASE_URL=postgres://user:pass@localhost:5433/db
```

Your ORM (SQLAlchemy, Prisma, ActiveRecord, Drizzle) doesn't know there's a proxy in the middle. Queries arrive exactly the same way. Vericto intercepts them, analyzes them, and forwards them if they are safe.

Does Vericto support NoSQL databases like MongoDB or Redis?

No. Vericto is specific to SQL databases. AST parsing is a technique that applies to the SQL language: extending it to NoSQL would require entirely different parsers and the concept of a "destructive query" is less well-defined in those contexts.

The current focus is SQL with the most widely used dialects in production with AI agents: Postgres, MySQL, Oracle, and SQL Server, with SQLite and Snowflake on the roadmap. This covers 95%+ of LLM-to-SQL use cases in production.

Performance and availability

How much overhead does Vericto add to my queries?

It depends on the integration mode:

- **TCP proxy (recommended)**: evaluation happens inside the proxy, in your network, with no external calls. In our tests it adds under 1 ms per query, including medium and complex queries.
- **Runtime API (HTTP)**: each query travels to the Vericto API to be evaluated before you run it, so the time includes the network. In our tests the average is about 16 ms per evaluation.

The proxy is written in Rust with tokio for asynchronous TCP connection handling, with no garbage-collection pauses. Audit-trail logging is asynchronous and stays off the query's critical path.

What happens if Vericto has downtime? Will my application stop working?

The TCP proxy runs in your infrastructure and you operate it, with your own security controls and secrets management. Vericto hosts the control plane: dashboard, rules and audit trail.

If the control plane is unavailable, your application keeps working: the proxy keeps evaluating every query with the last synced ruleset and policy, and picks up changes once the service is back.

We recommend running the proxy with `VERICTO_TELEMETRY_BUFFER=disk`, so audit-trail events are stored on disk during the outage, survive restarts and are delivered once it recovers.

Check service status at vericto.com/status.

Security and data

Does Vericto store the content of my SQL queries?

It depends on the telemetry mode you choose in the dashboard:

- **Raw** (default): the text of each query is stored in the audit trail, encrypted at rest with AES-256-GCM. It gives you full forensic detail and the explanation of every block.
- **Sanitized**: Vericto never stores real values. Literals are replaced with placeholders before the query is stored, and the dashboard shows the obfuscated statement, for example `DELETE FROM users WHERE id = $1`, not the actual query that ran. With the TCP proxy and the CLI the replacement happens in your network, so the values never leave it. With the Runtime API, the Validation API and MCP, the query travels to Vericto to be evaluated and is sanitized before it is stored.

Access to the audit trail requires authentication with workspace permissions.

Retention depends on the plan: 7 days (Free), 30 days (Builder), 90 days (Team) and configurable on Enterprise. You can export and purge the audit trail at any time.

Query data is never used to train models or shared with third parties.

Is Vericto suitable for enterprise and compliance requirements?

Yes. Vericto deploys in your own infrastructure (your credentials never leave your network) and includes the pieces an enterprise environment requires:

- **Ed25519-signed audit trail**, offline-verifiable: direct evidence for SOC2 (CC6.1, CC6.3, CC7.2) and ISO 27001.
- **SSO / OIDC** and role-based access control for your team.
- **Custom YAML rules** for your organization's specific policies.

The Enterprise plan adds assisted deployment, an SLA, and dedicated support.

Is Vericto SOC 2 or ISO 27001 certified?

Vericto is in the process of SOC 2 certification. We do not have a SOC 2 report or ISO 27001 certification yet, and we will publish it here when it is available.

What Vericto gives you today is **evidence for your own audits**: an audit trail of every decision, exportable as CSV/JSON with an Ed25519 signature you can verify offline, and signed validation reports. For security reviews, write to security@vericto.com.

Does Vericto sign a data processing agreement (DPA)?

Yes. The [Data Processing Agreement](#) is part of the Terms of Service and applies to every customer from the moment they accept them: Vericto acts as processor and your organisation as controller. It covers GDPR, Law 1581, LGPD and the LFPDPPP, and includes sub-processors, security measures and international transfers.

If your procurement team needs a signed copy, download the PDF from vericto.com/dpa, complete it and send it to legal@vericto.com.

CI/CD

How does the dry-run mode for CI/CD work?

The Vericto CLI allows you to validate a SQL query file against your workspace's ruleset without needing a database connection:

```
vericto check migrations.sql --dialect postgres
```

```
✓ 12 queries analyzed.
```

```
✗ 1 destructive query detected:
```

```
Line 47: TRUNCATE TABLE orders
```

```
Rule: VERICTO-011 (severity: CRITICAL)
```

```
AST node: TruncateStmt > relation: orders
```

```
Suggested fix: DELETE FROM orders WHERE created_at < NOW() - INTERVAL  
'90 days'
```

The process exits with code 1 if there are destructive queries (the CI pipeline fails) or exit code 0 if everything is clean. Compatible with GitHub Actions, GitLab CI, and Jenkins.

Can't find your answer? Write to hello@vericto.com and we'll reply within 1 business day.