OSA CON 25

AI and Analytical Databases

Peter Zaitsev
Founder, Percona

November 4-5, 2025

AI and Databases is a Broad Question

This is a pretty broad topic!

AI is Moving so fast

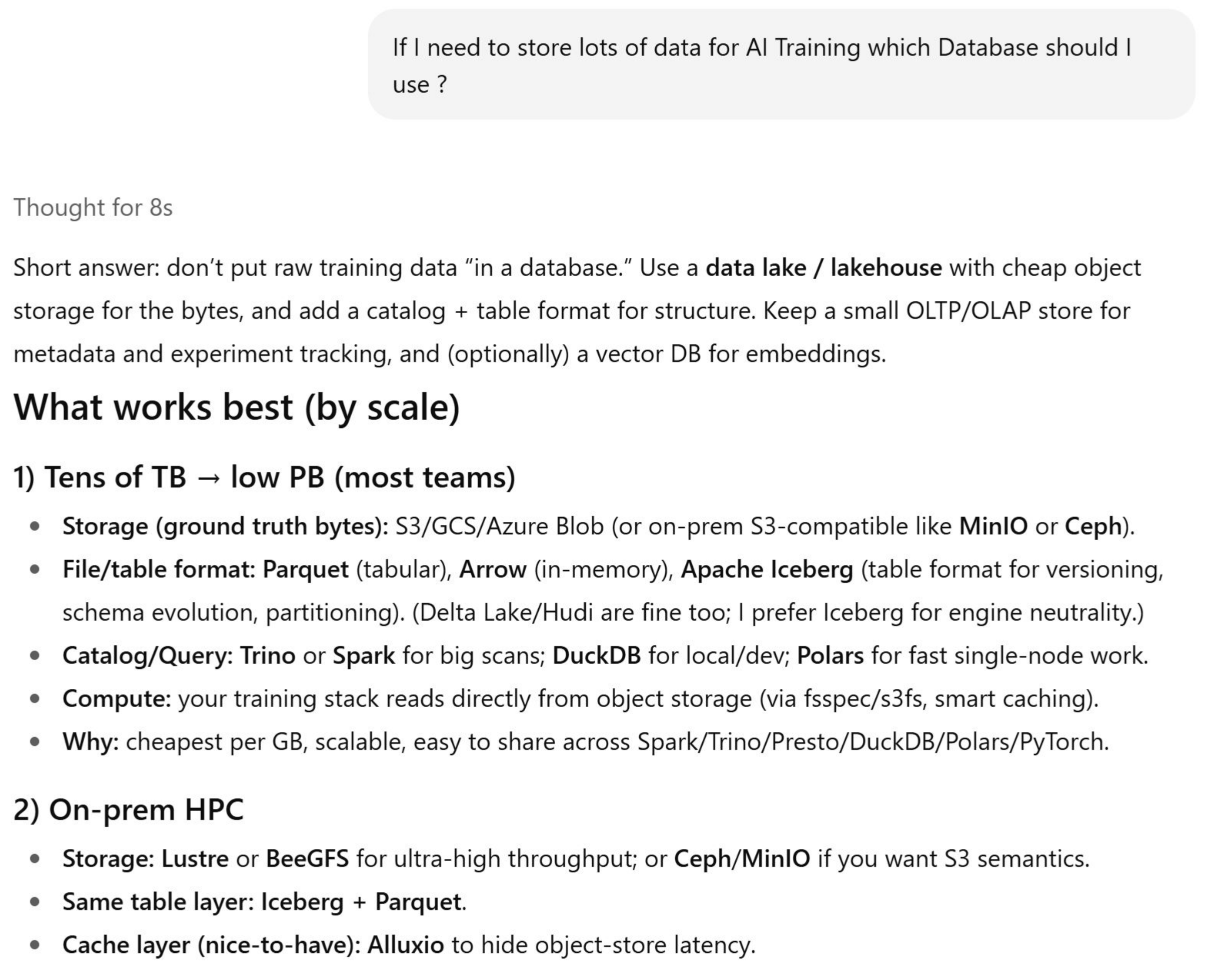
There are so many perspectives based on what your job is

AI and Data and Databases

- Storage of the data for AI Training
- In Database Processing, Embedding, Vector Search, ML
- Natural Language Interfacing (MCP)
- AI for Operations, "AI DBA"
- AI for Development
- Analyses and Visualizations

PERCONA

Storage of Data for AI Training



If I need to store lots of data for AI Training which Database should I use ?

Thought for 8s

Short answer: don't put raw training data "in a database." Use a **data lake / lakehouse** with cheap object storage for the bytes, and add a catalog + table format for structure. Keep a small OLTP/OLAP store for metadata and experiment tracking, and (optionally) a vector DB for embeddings.

What works best (by scale)

1) Tens of TB → low PB (most teams)

- **Storage (ground truth bytes):** S3/GCS/Azure Blob (or on-prem S3-compatible like **MinIO** or **Ceph**).
- **File/table format: Parquet** (tabular), **Arrow** (in-memory), **Apache Iceberg** (table format for versioning, schema evolution, partitioning). (Delta Lake/Hudi are fine too; I prefer Iceberg for engine neutrality.)
- **Catalog/Query: Trino** or **Spark** for big scans; **DuckDB** for local/dev; **Polars** for fast single-node work.
- **Compute:** your training stack reads directly from object storage (via fsspec/s3fs, smart caching).
- **Why:** cheapest per GB, scalable, easy to share across Spark/Trino/Presto/DuckDB/Polars/PyTorch.

2) On-prem HPC

- **Storage: Lustre** or **BeeGFS** for ultra-high throughput; or **Ceph/MinIO** if you want S3 semantics.
- **Same table layer: Iceberg** + **Parquet**.
- **Cache layer (nice-to-have): Alluxio** to hide object-store latency.

Type of Data Matters

- Unstructured Data
- Metadata
- Structured Data
- Specific Formats (i.e., Time Series)

PERCONA

Scale and Type of Training

We are excited to hear about extreme scale but often work with “small” scale

“Small” scale – what you have often works best

Few do full blown hyperscale model training, more do fine-tuning/ domain specific tuning with relatively little data

PERCONA

Training Model From Scratch

From Scratch - Massive, general corpus – GPT-5 Base Model

Fine Tuning for behavior

Pretrained - Task of Instruction Dataset - ChatGPT

Domain specific Training

Pretrained – Domain Specific corpus to specialize in a field – LegalGPT, CodeLLAMA

Difference

In Database Processing

- Vector Datatype and Indexing
- In Database Embedding Generation
- Semantic Search and Hybrid Search
- RAG “Inside Database”
- External Model Interfacing
- AI Supported Functions (Translate, Summarize etc)
- Machine Learning, Anomaly Detection etc

PERCONA

Natural Language Interfacing

- Supporting AI Agents (MCP Servers)
- Database Interface can be broad part of your interface to your infrastructure
- Management and Maintenance
- Natural Language Querying

Inside the Database Kernel

External "AI DBA"

AI for Operations

Inside Database Kernel

- Learned Indexes and Cardinality Estimation
- AI Based Optimizer
- Auto Indexing
- Adaptive Algorithms
- Autotuning
- Contention Prediction
- Security

External Solutions

- Configuration Tuning
- Query Optimization
- Autonomous Database (kind of)

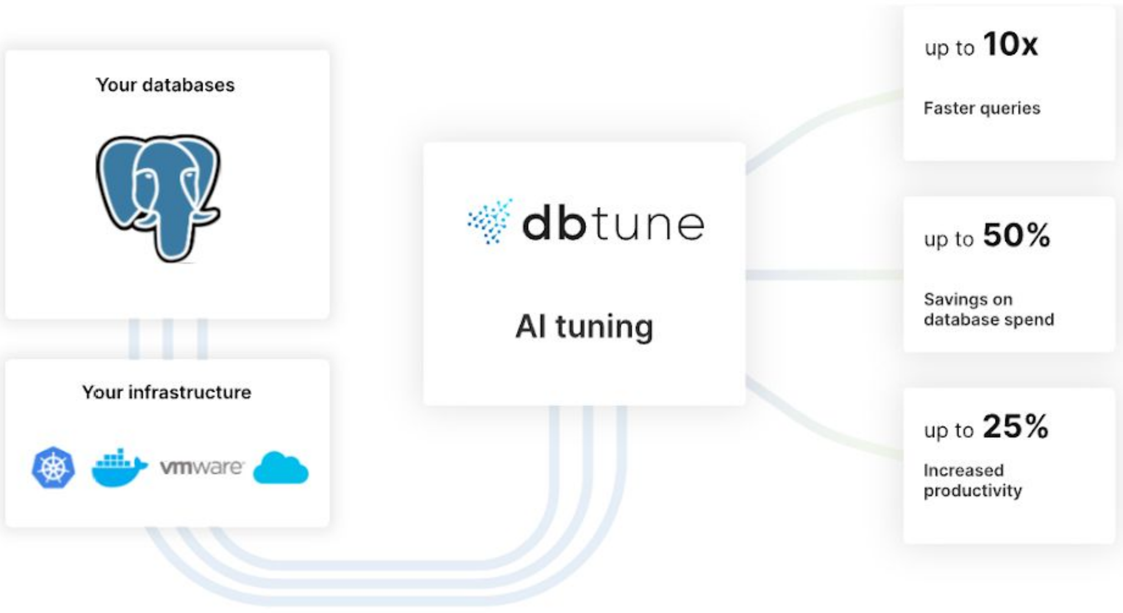
dbtune Products Resources Media Company

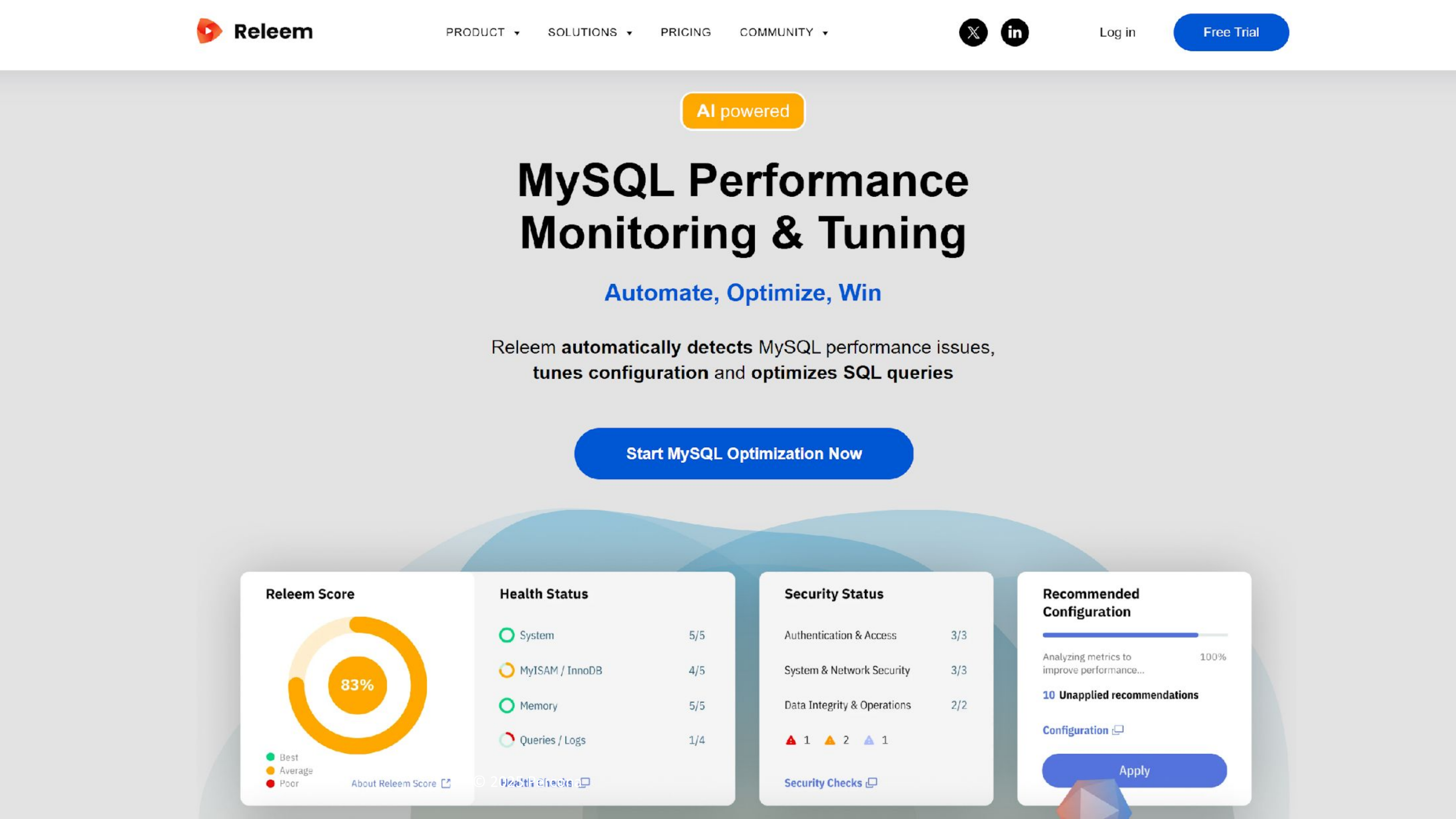
Perform better
Spend less

DBtune's AI-powered optimizer tunes your PostgreSQL server parameters for top performance, regardless of workload, use case, or machine size.

Focus on strategic tasks, while DBtune saves you time and money.

Get started Book a demo




Releem

PRODUCT ▾ SOLUTIONS ▾ PRICING COMMUNITY ▾

Log in

Free Trial

AI powered

MySQL Performance Monitoring & Tuning

Automate, Optimize, Win

Releem **automatically detects** MySQL performance issues, **tunes configuration** and **optimizes SQL queries**

Start MySQL Optimization Now

Releem Score

83%

● Best
● Average
● Poor

About Releem Score

Health Status

System	5/5
MyISAM / InnoDB	4/5
Memory	5/5
Queries / Logs	1/4

Health Checks

Security Status

Authentication & Access	3/3
System & Network Security	3/3
Data Integrity & Operations	2/2

▲ 1 ▲ 2 ▲ 1

Security Checks

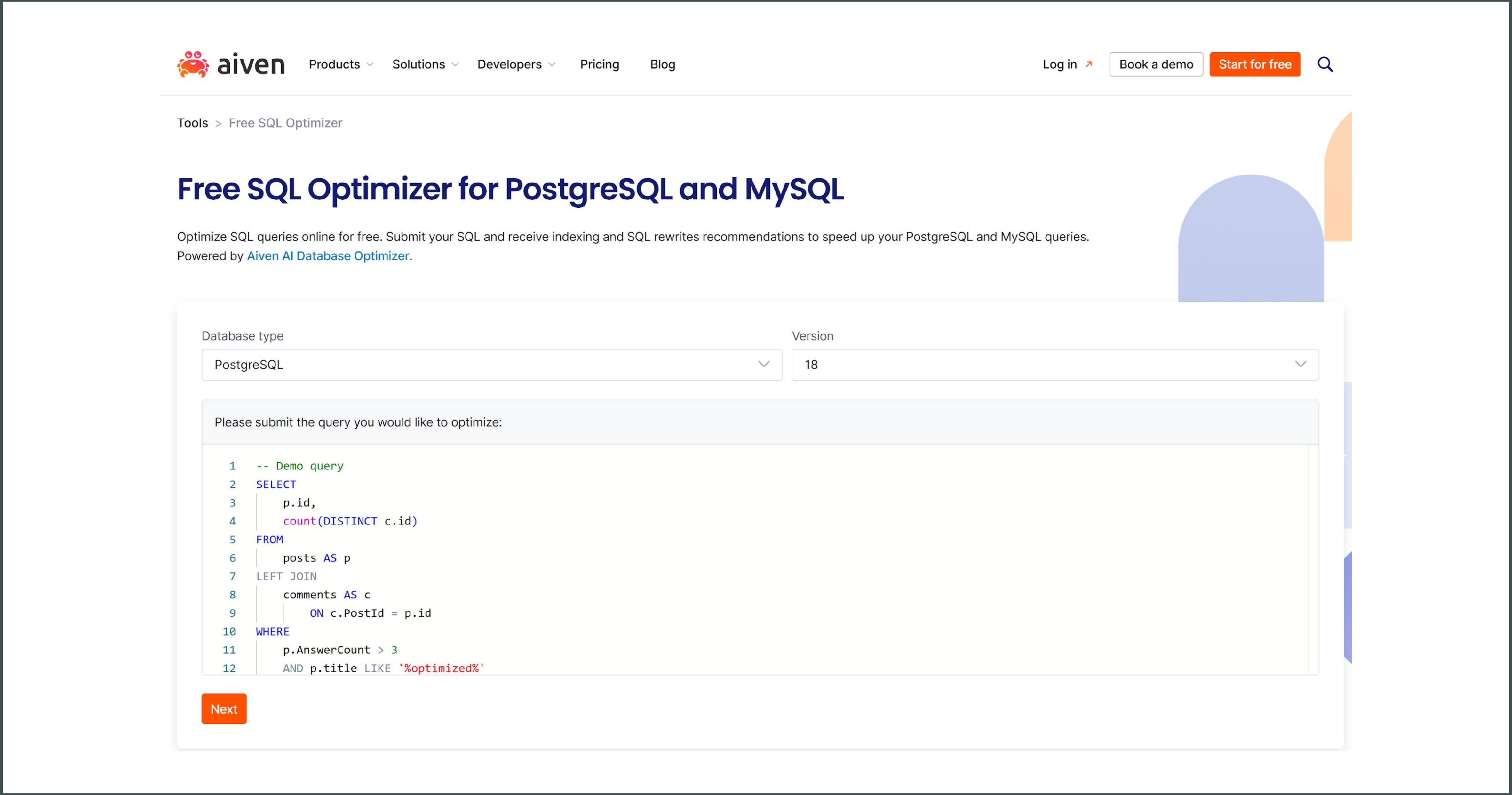
Recommended Configuration

Analyzing metrics to improve performance... 100%

10 Unapplied recommendations

Configuration

Apply



Tools > Free SQL Optimizer

Free SQL Optimizer for PostgreSQL and MySQL

Optimize SQL queries online for free. Submit your SQL and receive indexing and SQL rewrites recommendations to speed up your PostgreSQL and MySQL queries. Powered by Aiven AI Database Optimizer.

Database type: PostgreSQL

Version: 18

Please submit the query you would like to optimize:

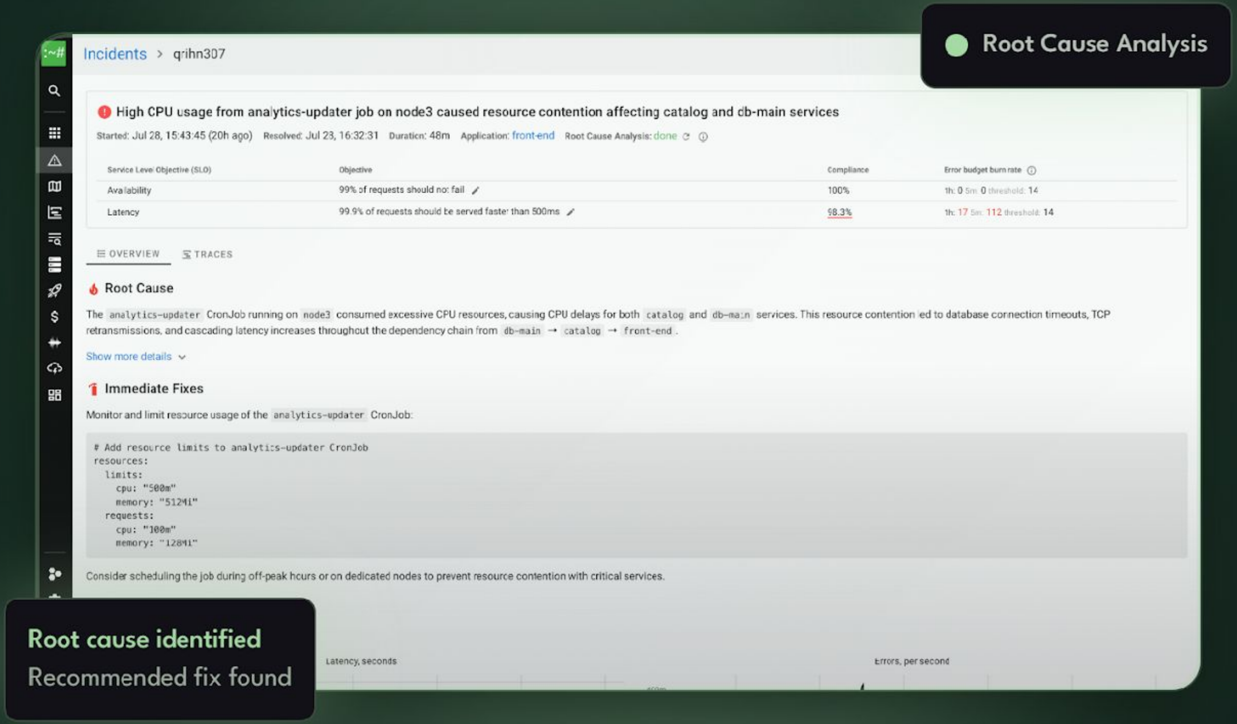
```
-- Demo query
SELECT
    p.id,
    count(DISTINCT c.id)
FROM
    posts AS p
LEFT JOIN
    comments AS c
        ON c.PostId = p.id
WHERE
    p.AnswerCount > 3
    AND p.title LIKE '%optimized%'
```

AI that finds the **root cause** of your incidents — instantly

Coroot uses eBPF to connect in minutes. No code changes needed. It shows you exactly what broke, why it happened, and how to fix it.

Start free trial

Book a demo



High CPU usage from analytics-updater job on node3 caused resource contention affecting catalog and db-main services

Started: Jul 28, 15:43:45 (20h ago) Resolved: Jul 23, 16:32:31 Duration: 48m Application: front-end Root Cause Analysis: done

Service Level Objective (SLO)	Objective	Compliance	Error budget burn rate
Availability	99% of requests should not fail	100%	1h: 0 5m: 0 threshold: 14
Latency	99.9% of requests should be served faster than 500ms	98.3%	1h: 17 5m: 112 threshold: 14

Root Cause

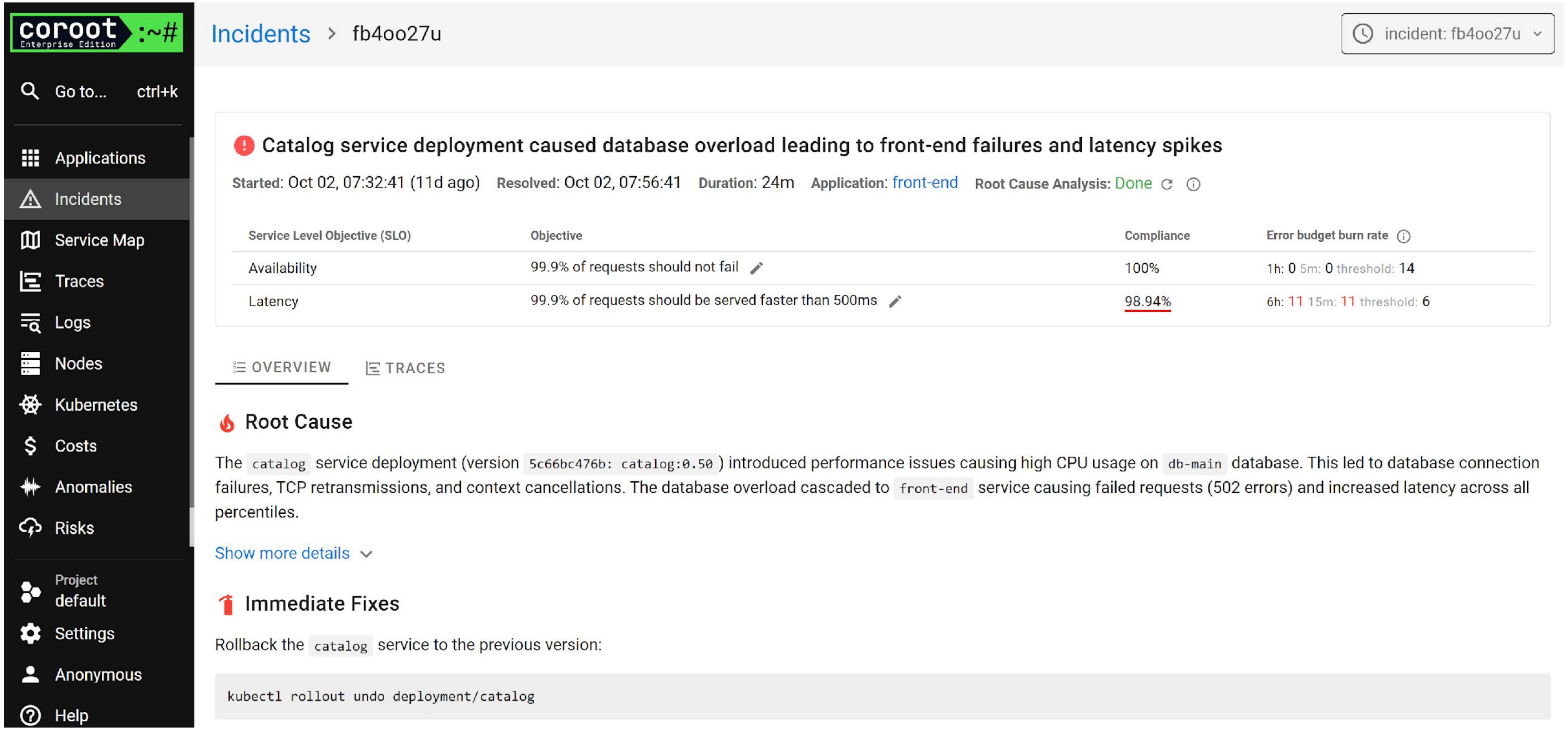
The `analytics-updater` CronJob running on `node3` consumed excessive CPU resources, causing CPU delays for both `catalog` and `db-main` services. This resource contention led to database connection timeouts, TCP retransmissions, and cascading latency increases throughout the dependency chain from `db-main` → `catalog` → `front-end`.

Immediate Fixes

Monitor and limit resource usage of the `analytics-updater` CronJob:

```
# Add resource limits to analytics-updater CronJob
resources:
  limits:
    cpu: "500m"
    memory: "512Mi"
  requests:
    cpu: "100m"
    memory: "128Mi"
```

Consider scheduling the job during off-peak hours or on dedicated nodes to prevent resource contention with critical services.



Catalog service deployment caused database overload leading to front-end failures and latency spikes

Started: Oct 02, 07:32:41 (11d ago) Resolved: Oct 02, 07:56:41 Duration: 24m Application: front-end Root Cause Analysis: Done

Service Level Objective (SLO)	Objective	Compliance	Error budget burn rate
Availability	99.9% of requests should not fail	100%	1h: 0 5m: 0 threshold: 14
Latency	99.9% of requests should be served faster than 500ms	98.94%	6h: 11 15m: 11 threshold: 6

Root Cause

The `catalog` service deployment (version `5c66bc476b: catalog:0.50`) introduced performance issues causing high CPU usage on `db-main` database. This led to database connection failures, TCP retransmissions, and context cancellations. The database overload cascaded to `front-end` service causing failed requests (502 errors) and increased latency across all percentiles.

Immediate Fixes

Rollback the `catalog` service to the previous version:

```
kubectl rollout undo deployment/catalog
```

SCHEMA DESIGN

TEXT TO SQL

QUERY IMPROVEMENT AND INDEX RECOMMENDATION

EXPLAIN EXPLAIN

SYNTHETIC DATA

AI for Database Development

AI 2sql*

Features Solutions Use Cases Pricing

Start a free trial →

Text to SQL
Convert your natural language queries into SQL commands e...

Explain SQL
Understand your SQL queries better for clear insights.

Optimize SQL
Enhance your SQL query performance.

Format SQL
Clean and organize your SQL code effortlessly.

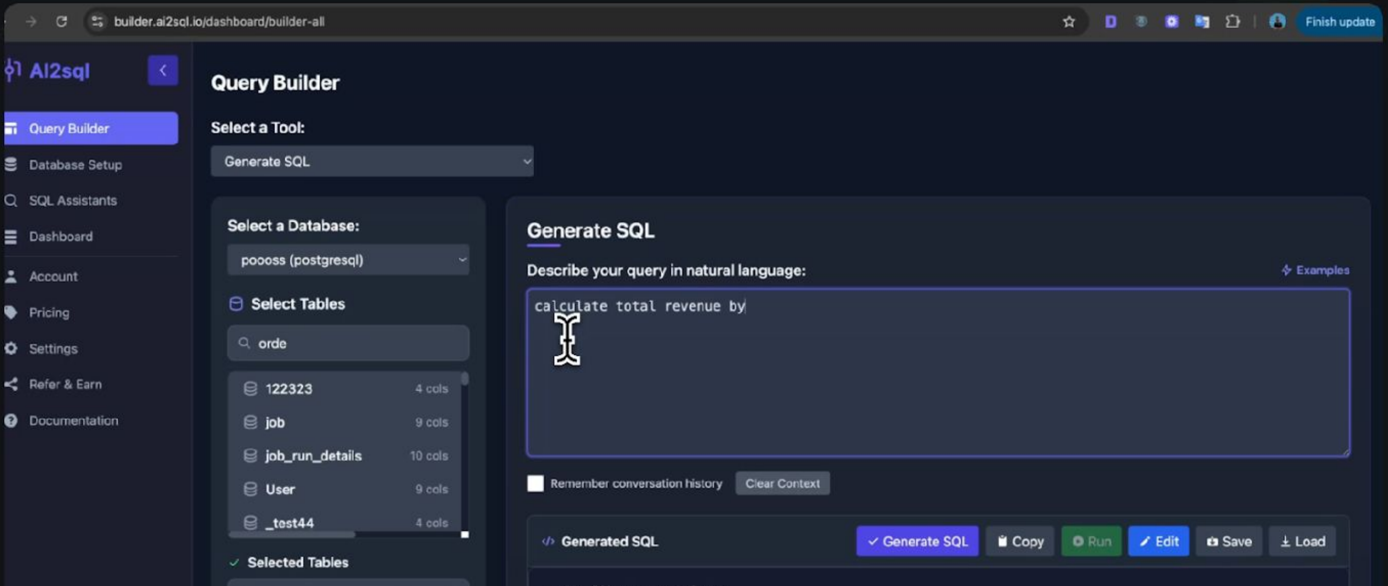
Data Insight Gener...
Exploring potential angles of analysis for your datasets.

Generate Complex SQL in 10 Seconds — No Coding Required

Instant SQL generation from natural language descriptions

★★★★★ Trusted by 50,000+ users across 80+ countries

Start a free trial →


builder.ai2sql.io/dashboard/builder-all

AI2sql

Query Builder
Database Setup
SQL Assistants
Dashboard
Account
Pricing
Settings
Refer & Earn
Documentation

Query Builder

Select a Tool:
Generate SQL

Select a Database:
pooss (postgresql)

Select Tables
orde

122323	4 cols
job	9 cols
job_run_details	10 cols
User	9 cols
_test44	4 cols

Selected Tables

Generate SQL

Describe your query in natural language:

```
calculate total revenue by
```

Examples

Remember conversation history Clear Context

Generated SQL

Data Preparation and Cleanup

Ask questions in human language

Visualization, Creating Dashboards

Explaining Meaning of Data and Potential Causes

Analyses and Visualization

DIKW

Data - 100,102,98,105

Information – “Temperatures recorded Hourly in Raleigh, NC Starting at Noon 12th of July 2025”

Knowledge – Temperatures Rose due to Passing Front

Insights – Warm fronts like this often precede storms. Schedule maintenance early

Wisdom – We will adjust our seasonal maintenance to anticipate similar patterns

PERCONA

Final Thoughts

The high pace AI world is “clashing” with slow moving world of databases

A lot of uncertainty how industry will evolve and how quickly

No matter what your job is, track how AI capabilities are evolving to do it faster and better

Thank you, Let's Connect!

https://www.linkedin.com/in/peterzaitsev/

https://twitter.com/PeterZaitsev

http://www.peterzaitsev.com

OSA CON 25

PERCONA