OURSQL FOUNDATION · NOVEMBER 2026

Twenty Years of Bolting Analytics onto MySQL: Will DuckDB Finally Make It HTAP?

An introductory talk; detailed DuckDB sessions follow later today

2008
Plug-in columnar engines

2013
MySQL-compatible clusters

2020
Accelerators and replicas

SAME SERVER · SAME 500 GB · SAME 22 TPC-H QUERIES

28+ hours

MySQL with InnoDB

3 minutes

Same MySQL with a DuckDB engine

Why did this take 18 years?

If you've ever copied MySQL data elsewhere just to run a report, this talk is for you.

Percona blog (Aug. 2026): TPC-H scale factor 500 on one 80-core server; 18 queries took 28+ hours on InnoDB, 4 never finished.

ABOUT THE SPEAKER

Vadim Tkachenko

President, OurSQL Foundation

Founder, Percona

20+

years working with MySQL

Why the OurSQL Foundation cares about MySQL analytics

OUR MISSION

Grow the MySQL ecosystem

01 MySQL by itself has always lacked analytical capabilities; in 2009, users already hit limits near 50 GB.

02 Analytical workloads are an area where we can improve MySQL's position.

03 So the foundation is interested in tools and projects that elevate MySQL analytics.

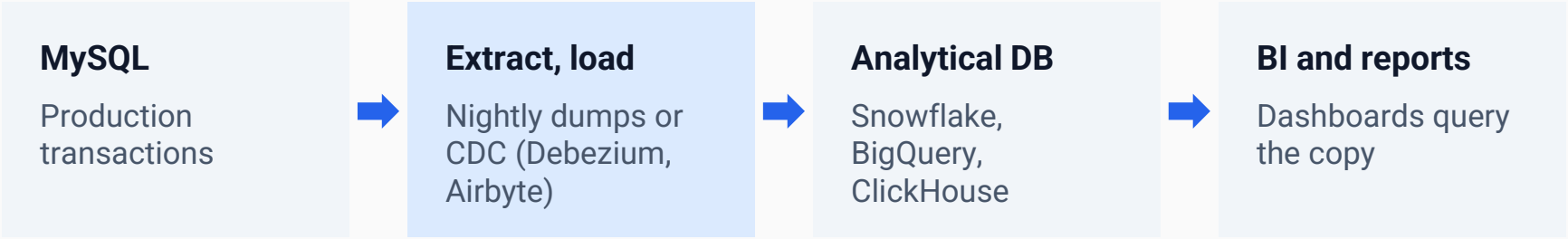
Sources: OurSQL Foundation; eWeek (April 2009).

MySQL users have tried five ways to get analytics; DuckDB adds two more

Approach	Since	Examples	Where analytics runs
ETL to a warehouse	Always	Nightly jobs or CDC pipelines	A separate warehouse
Plug-in columnar engine	2008	Infobright, Kickfire, InfiniDB	Inside MySQL
MySQL-compatible HTAP	2013	SingleStore, TiDB with TiFlash	A different database
Managed accelerator	2020	MySQL HeatWave	A vendor's cloud
Continuous replica	2020	Aurora zero-ETL, MaterializedMySQL	A synced copy
DuckDB beside MySQL	2024	DBTrail, MyDuck Server	A DuckDB copy
DuckDB inside MySQL	2025	AliSQL, Percona, MariaDB, VillageSQL	Inside MySQL, again

Each of the first five works for some workloads. The next slides cover where each fell short; the DuckDB rows follow.

The default answer is ETL: copy MySQL data into a warehouse



Why teams use it

Mature tools, warehouses built for scans, and reporting load kept off production.

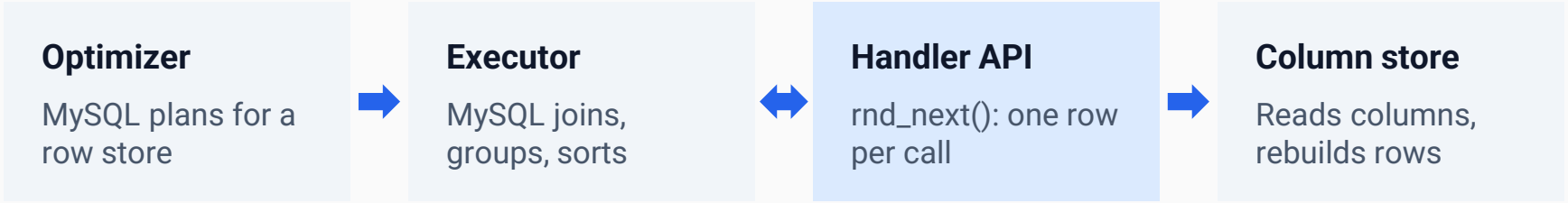
Where it falls short

Hours of lag with batch jobs, pipelines that break on schema changes, and a second system to run and pay for.

Simplified. Tools named are examples, not recommendations.

Plug-in engines had to bypass MySQL's row-at-a-time engine API

DEFAULT PATH IN MYSQL 5.1



Bypass: vendor executors took the query above the handler

Why the default path was slow

Columns were rebuilt into rows, then aggregated one row at a time by MySQL's executor.

What the bypass cost

Each engine needed its own, often patched, MySQL build and had to chase every release.

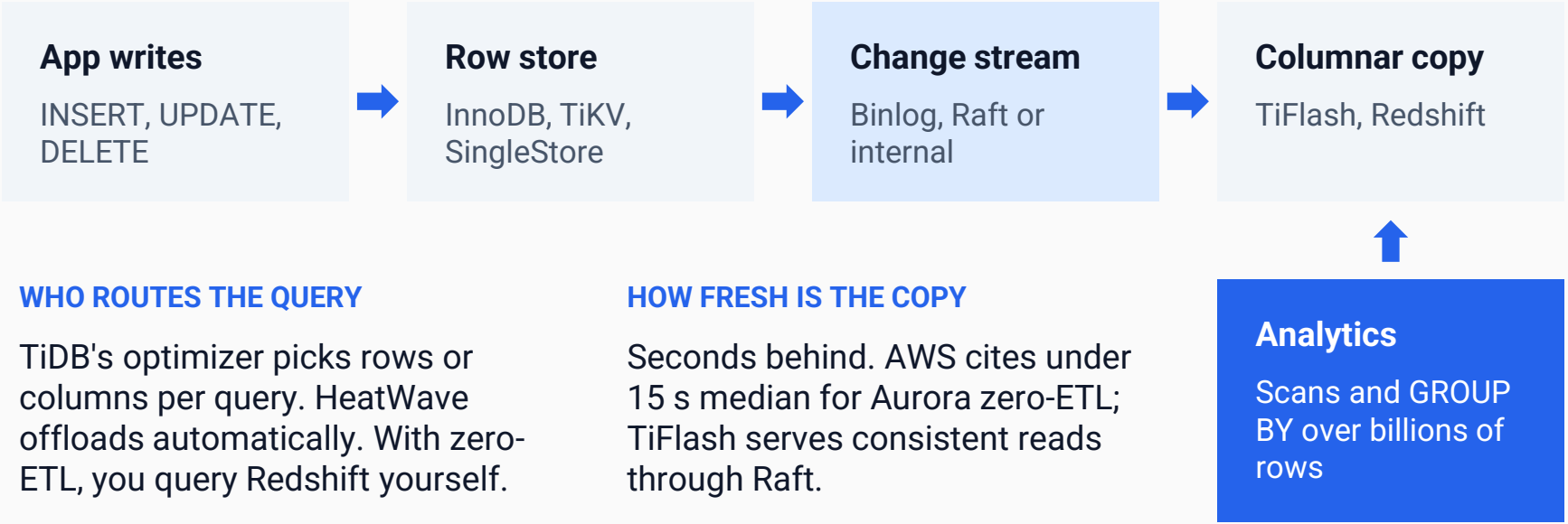
Simplified. Source: MySQL Reference Manual, storage engine API (handler class). Vendor bypass details varied by engine.

Plug-in engines worked, but the businesses behind them didn't

Engine	Main limitation	Why it ended
Infobright	Free edition was load-only: no INSERT, UPDATE or DELETE	Few free users paid; open edition dropped in 2016
Kickfire	Proprietary appliance with a custom SQL chip; one box	Commodity CPUs caught up; Teradata bought it in 2010
Calpont InfiniDB	Patched MySQL build; slow updates and InnoDB joins	No investor found; closed Sept. 2014
MariaDB ColumnStore	Works only in MariaDB, not Oracle's MySQL	Still ships; depends on MariaDB plc

Sources: SiliconANGLE and PCWorld (Sept. 2014); DBDB.io, Infobright; MariaDB documentation.

HTAP databases and replicas split work between rows and columns



WHO ROUTES THE QUERY

TiDB's optimizer picks rows or columns per query. HeatWave offloads automatically. With zero-ETL, you query Redshift yourself.

HOW FRESH IS THE COPY

Seconds behind. AWS cites under 15 s median for Aurora zero-ETL; TiFlash serves consistent reads through Raft.

Simplified. Sources: PingCAP TiFlash docs; Oracle HeatWave docs; AWS (Nov. 2023).

Later approaches worked by leaving MySQL or staying closed

Approach	Main limitation	Outcome
SingleStore	A separate database you migrate to	Thriving, but proprietary
TiDB with TiFlash	Complex to run; no stored procedures	Open source, but a migration
MySQL HeatWave	Closed, Oracle-managed service only	Not for community MySQL
ClickHouse MaterializedMySQL	Binlog and DDL were hard to replicate	Experimental; removed Dec. 2024
AWS zero-ETL	AWS only; a second system to pay for	Analytics leave MySQL

Sources: SingleStore, PingCAP and Oracle docs; ClickHouse changelog (Dec. 2024); AWS (Nov. 2023).

Four lessons from the approaches that fell short

01

Hand over the whole query

A row-at-a-time engine API forced every columnar engine to patch MySQL.

02

Build upstream, in the open

Closed products died with their vendors; InfiniDB's GPL code lived on.

03

Columns must handle updates

Bulk-load-only column stores don't fit how MySQL is used.

04

Every copy has a cost

ETL, replicas and zero-ETL add lag and a second system to run.

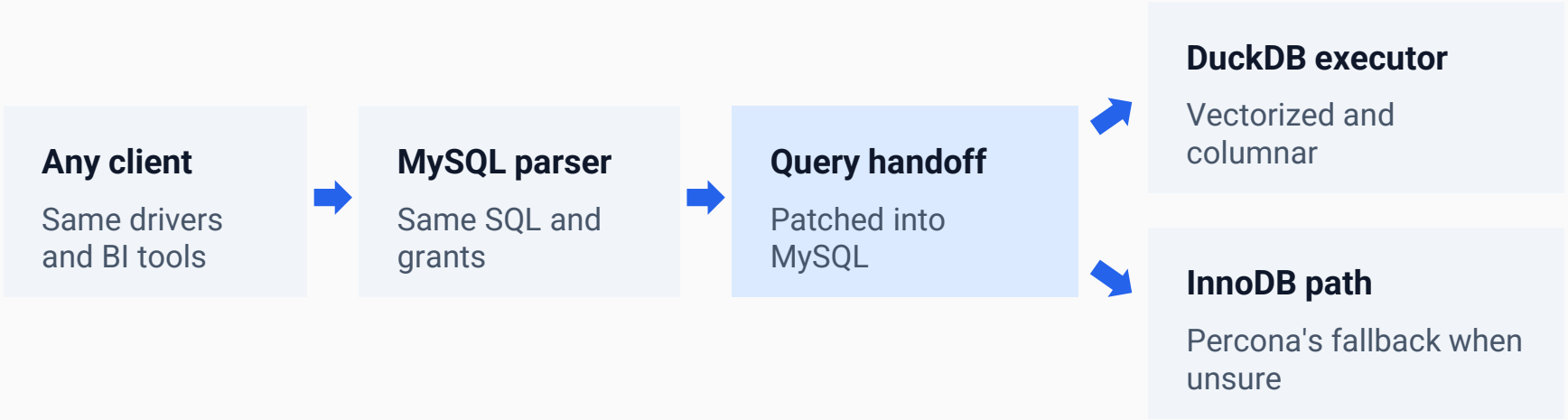
Lessons 01 and 02 are what today's DuckDB engines still need; see slide 16.

Four teams built DuckDB into MySQL-family servers in a year

Engine	Built on	Released	Status	Stands out for
AliSQL DuckDB engine	Alibaba's MySQL 8.0 branch	Dec. 2025	Production, 1,000+ nodes	Proposed upstream
Percona DuckDB engine	MySQL 9.7 + 3 patches	July 2026	Experimental	Falls back to InnoDB
MariaDB DuckDB engine	MariaDB 11.4	June 2026	Alpha	Joins DuckDB and InnoDB tables
VillageSQL vsql-duckdb	VillageSQL (MySQL fork)	Sept. 2026	Preview	Queries Parquet files, not InnoDB

Sources: AliSQL on GitHub; MySQL discussion #79; Percona blog (July 2026); lefred.be (June 2026); VillageSQL blog (Sept. 2026).

With a DuckDB engine, MySQL hands the whole query to DuckDB



Same idea as the plug-in bypass

But the executor is open source and shared.

The box everyone patches

The handoff hook is the gap on slide 16.

Simplified. Sources: Alibaba AliSQL on GitHub; Percona blog (July 2026); lefred.be (June 2026).

At 500 GB, DuckDB in MySQL ran TPC-H in 3 minutes, not 28+ hours

186 s

All 22 queries on the DuckDB engine

28+ h

18 of 22 queries on InnoDB; 4 never finished

- **Load:** 36 min vs. 15 h 21 min
- **Disk:** 132 GB vs. 673 GB
- **Correctness:** 21 of 22 results matched native DuckDB; none mismatched
- **Caveat:** Percona's experimental engine, one 80-core server

Source: Percona blog, "The DuckDB MySQL engine at 500 GB" (Aug. 2026); TPC-H scale factor 500, 3 billion lineitem rows.

Without changing MySQL, DuckDB connects in four other ways

DuckDB mysql extension

ATTACH a live MySQL database; read, write and export to Parquet

Stable · DuckDB core, MIT

MyDuck Server

DuckDB server that speaks MySQL and Postgres; runs as a MySQL replica

Active · ApeCloud, Apache 2.0

DBTrail

Binlog-fed Parquet copy; DuckDB runs reports on another machine

Active, v0.99 · Apache 2.0

Alibaba Cloud RDS for MySQL

Managed DuckDB analytical instances fed by MySQL replication

Since 2025 · managed service

Sources: DuckDB docs; ApeCloud MyDuck Server on GitHub; Percona blog (Oct. 2026); Alibaba Cloud (Aug. and Sept. 2025).

The missing piece is a shared, upstream engine API in MySQL

01 Stock MySQL can't hand a whole query to an engine; Percona needed three server patches.

02 AliSQL's upstream proposal is a draft for MySQL 8.0; Oracle has asked questions but not decided.

03 Four teams built DuckDB integrations independently, so the effort is split four ways.

OPEN ISSUES IN THE DESIGN

- Physically unique primary keys
- Joins between DuckDB and InnoDB
- Backup and Clone support
- Dependence on a forked DuckDB

Sources: MySQL community discussion #79 and AliSQL design draft (July–Aug. 2026); Percona blog (July 2026).

Takeaways

Every copy has a cost

ETL and replicas work, but add lag and a second system to run.

Engines need a real API

Most in-server engines had to patch or fork MySQL to run whole queries.

Open source outlasts vendors

InfiniDB's GPL code outlived Calpont; open engines like ClickHouse, Doris and DuckDB run anywhere.

HTAP is within reach

DuckDB brings both workloads into one server; an upstream API would finish the job.