ProxySQL: Supercharge Your MySQL Performance

René Cannaò
CEO , ProxySQL LLC

preFOSDEM MySQL Belgian Days 2025
30 January 2025

Agenda

Who am I?

Introduction to ProxySQL

Key Features in Version 2.6 , 2.7 and a short previous of 3.0

- Performance enhancements
- Security improvements
- Management simplifications

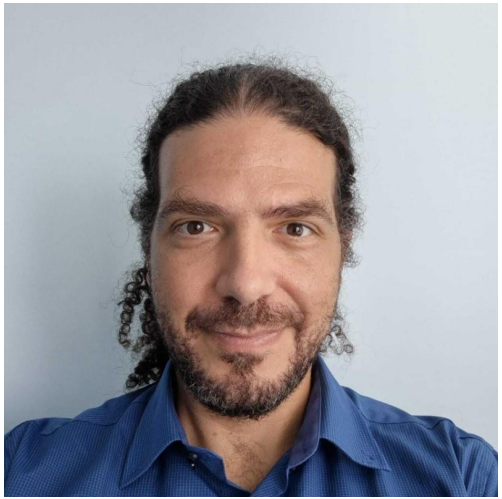
User-driven development: PROXY protocol support
User-driven development: Advanced in-memory query logging
Sustainable open source

Q&A

Meet the Creator – René Cannaò

- Founder & CEO of ProxySQL
- Over 25 years of experience in database architecture & performance optimization
- Former MySQL consultant and performance expert
- Creator of ProxySQL to solve real-world MySQL scalability challenges
- Passionate about high-performance databases, open-source development, and community collaboration

#OptimizingMySQLAtScale



ProxySQL – The Intelligent MySQL Proxy

- High-performance SQL proxy designed for MySQL, MariaDB, and Percona Server
- Engineered to handle millions of queries per second
- Optimizes database traffic with advanced query routing and caching
- Connection Management
- Used by high-profile companies, including Nasdaq & S&P 500 businesses
- Trusted by enterprises to improve scalability, security, and manageability

Why Choose ProxySQL ?

✅ Scalability – Handle hundreds of thousands of connections effortlessly

✅ Performance – Reduce query latency and backend load

✅ Reliability – Built-in failover and high availability

✅ Security – Advanced authentication , traffic filtering and data masking

✅ Observability – Real-time query analytics and monitoring

The Ultimate MySQL Performance Booster

- Advanced Connection Pooling → Efficiently manage millions of connections
- Query Routing & Rewriting → Optimize query execution dynamically
- Caching Engine → Reduce database load & improve response times
- High Availability & Failover → Seamless switchover without downtime
- Security & Authentication → Proxy user authentication, whitelisting, auditing and traffic filtering
- Real-time Query Monitoring → Deep insights into query patterns and database performance

readyset

Modern **database scaling platform** and **SQL query caching** system.

Boost performance, optimize infrastructure costs, and improve engineering velocity with proactive enhancements for PostgreSQL & MySQL

Reduce hosting costs **by up to 70%**

In memory query **read cache - sub millisecond**

Auto synced to source db changes **in milliseconds**

No **TTLs**, no **cache invalidations**

Unchanged cache latency even when **data changes**

Available through aws marketplace

Performance Enhancements in Version 2.6.x and 2.7.x

- Several dependencies upgrades
- Overall improved performance
  - Various micro optimizations
- Configurable protocol compression level
  - mysql-protocol_compression_level
  - default to 3 (as per MySQL) instead of 6 (zlib)

Improved Usability in Version 2.6.x

- Added Native Support for Group Replication autodiscovery for MySQL 8
- Added support for Bootstrap Mode for MySQL 8 Group Replication
- Added hostgroup_attributes for AWS Aurora autodiscovery
- Added support for AWS RDS MySQL Multi-AZ Cluster autodiscovery
- hostgroup_settings to mysql_hostgroup_attributes
- max_num_online_servers in mysql_hostgroup_attributes

Improved Usability in Version 2.6.x

- Added support for SHOW WARNINGS command
- Prepared statement memory usage tracking
  - prepared_statement_metadata_memory
  - prepared_statement_backend_memory
- Status variable for PROXYSQL PAUSE
  - mysql_listener_paused
- TCP keepalive enabled by default
  - mysql-use_tcp_keepalive
- Daemon restart with exponential backoff
- Built-in DNS cache automatically disabled when monitoring is disabled

Security Improvements in Version 2.6.x

- Added support for caching_sha2_password also for frontend connections
  - SSL required
  - Allows mismatch between frontend and backend authentication plugin
  - Deprecate variable admin-hash_passwords
  - Added new variable mysql-default_authentication_plugin
- New ProxySQL Admin functions:
  - MYSQL_NATIVE_PASSWORD()
  - CACHING_SHA2_PASSWORD()

Security Improvements in Version 2.6.x

- Added new table mysql_servers_ssl_params
  - It is now possible to configure SSL parameters on a per-host basis
  - ssl_ca, ssl_cert, ssl_key, ssl_capath, ssl_cipher , etc

Improved Usability in Version 2.7.x

Two major user driven developments

- Support for PROXY V1 Protocol
  - Sponsored by DigitalOcean
- Real-time in-memory and on-disk database query logging
  - Sponsored by Booking.com

PROXY V1 Protocol Support

- Support for network load balancers between clients and ProxySQL
- IP-based query rules
  - Caching, rewrite, routing, filtering, etc
- Improved processing and auditing
  - Client's IP is preserved
- Enhanced security
  - Restrict access based on IP ranges
- More accurate performance monitoring
  - Possible to track the origin of database connections
  - Possible to identify performance issues related to specific clients

More details: https://www.digitalocean.com/blog/digitalocean-proxysql-collaborate

PROXY V1 Protocol Support

New variable:

- mysql-proxy_protocol_networks
  - Comma separated list of IPs and/or subnet masks
  - Or empty string

Enhanced real-time in-memory query logging

- Events/queries logging in memory
  - High performance configurable fixed-size circular buffer
- In-memory SQLite table
  - On-demand dump from circular buffer
  - Configurable size limit to constrain memory usage
- On-disk SQLite table
  - On-demand dump from circular buffer
  - Configurable automatic dump from circular buffer
- Detailed metrics on usage

More details: https://proxysql.com/documentation/advanced-event-and-query-logging/

Enhanced real-time in-memory query logging : variables

- mysql-eventslog_buffer_history_size
- mysql-eventslog_table_memory_size
- mysql-eventslog_sync_buffer_to_disk
- mysql-eventslog_buffer_max_query_length

Example:
SET mysql-eventslog_buffer_history_size = 1000000;
SET mysql-eventslog_default_log = 1;
LOAD MYSQL VARIABLES TO RUNTIME;

Enhanced real-time in-memory query logging : commands

New commands:

- DUMP EVENTSLOG FROM BUFFER TO MEMORY;
  - From circular buffer to in-memory SQLite
- DUMP EVENTSLOG FROM BUFFER TO DISK;
  - From circular buffer to on-disk SQLite
- DUMP EVENTSLOG FROM BUFFER TO BOTH;
  - From circular buffer to both in-memory and on-disk SQLite

Tables:

- In-memory: stats.stats_mysql_query_events
  - The first writable and not dynamic table in the stats schema
- On-disk: history.history_mysql_query_events

Enhanced real-time in-memory query logging : metrics

New metrics (Prometheus , but also available in stats_mysql_global)

- proxysql_mysql_logger_copy_total target={memory/disk}
- proxysql_mysql_logger_get_all_events_calls_total
- proxysql_mysql_logger_get_all_events_events_total
- proxysql_mysql_logger_get_all_events_seconds_total target={memory/disk}
- proxysql_mysql_logger_copy_seconds_total target={memory/disk}
- proxysql_mysql_logger_events_copied_total
- proxysql_mysql_logger_circular_buffer_events_total type={added/dropped}
- proxysql_mysql_logger_circular_buffer_events

Sustainable Open Source

Commitment to sustainability in open source

- ProxySQL started as an open-source project and remains community-driven.
- We ensure long-term development by balancing enterprise support and open-source contributions

Growing community and adoption

- Used by top companies worldwide
- An active community of DBAs, DevOps engineers, and developers
- Regular updates driven by real-world database challenges and user feedback

How You Can Contribute

📢 **Provide Feedback** – Share your experience and performance insights.

🐞 **Report Bugs** – Help improve stability by identifying issues.

✨ **Suggest Features** – Influence ProxySQL's roadmap by proposing enhancements.

🤝 **Join the Discussion** – Engage in forums, GitHub, and open-source events.

What's new in ProxySQL 3.0

Disclaimer: not intended for production developments yet!

- Support for PostgreSQL Protocol
  - Bring to PostgreSQL ecosystem all the features already available in MySQL ecosystem
- Extensive code-refactoring
  - C++ Polymorphism
  - Better abstractions
  - More clean functionality boundaries

Call to Action

Try ProxySQL 2.7 and 3.0 today!

Join our community:

- https://github.com/sysown/proxysql
- https://proxysql.com/
- https://proxysql.com/documentation/

Q&A - Questions?

Contact: René Cannaò

Email: rene@proxysql.com