Introducción a ProxySQL

Rápido la infra!

- Clonamos el repositorio:

  https://gitlab.com/JavierJF/eslibre_workshop.git

- Hacemos ‘docker-compose build’ en el directorio ‘infra’.

We are hiring!

We are hiring!

- Programador senior con experiencia sólida en C++. (min 3 años)
- Experiencia en programación asíncrona y multihilo.
- Énfasis en rendimiento y optimización.
- Buenas habilidades de depuración con GDB y de interpretar GDB dumps.
- Entendimiento de la codebase de MySQL y el protocolo MySQL.
- Familiarizado con el opensource, con contribuciones demostrables.
- Requisitos de lenguajes: Inglés.

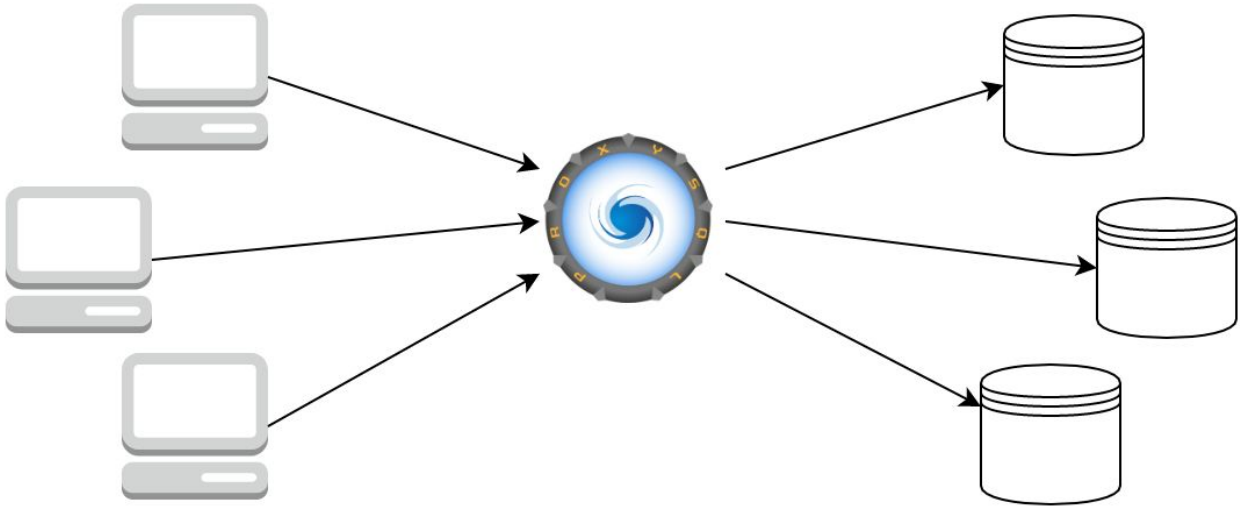
ProxySQL: El proyecto

- Creado por René Cannao, autor original y CEO de ProxySQL
  - Comienza su desarrollo en el verano de 2013.
  - Cómo DBA sentía frustración por no poder prevenir que malas consultas alcanzar la base de datos, para prevenirlo necesitaba:
    - Query rewrite
    - Query cache

ProxySQL: ¿Qué es?

- Proxy inverso de alto rendimiento consciente de protocolo para MySQL.
- Enfasis en:
  - Escalabilidad
  - Flexibilidad
  - Zero-downtime

Topología básica



ProxySQL: ¿Qué proporciona?

- Balanceo de carga
- Cortafuegos SQL
- Enrutamiento de consultas
- Monitorización de Clusters
- Estádisticas
- Mirroring / Etc!

Galera Cluster: ¿Qué es?

Galera Cluster for MySQL is a true Multi-Master Cluster based on synchronous replication. It's an easy-to-use, high-availability solution, which provides high system up-time, no data loss and scalability for future growth.

Galera Cluster: ¿Qué es?

- Multi-Master: Todos los nodos son en principio ‘Primario’ y ‘Replica’.
- Con replicación *virtualmente* síncrona.

Galera Cluster: ¿Qué es?

- Multi-Master: Todos los nodos son en principio ‘Primario’ y ‘Replica’.
- Con replicación *virtualmente* síncrona.

> *This approach is also called virtually synchronous replication, given that while it is logically synchronous, the actual writing and committing to the tablespace happens independently, and thus asynchronously on each node.*

Galera Cluster: Niveles de Aislamiento

- READ-UNCOMMITTED
- READ-COMMITTED
- READ-COMMITTED

Galera Cluster: Niveles de Aislamiento

- READ-UNCOMMITTED
- READ-COMMITTED
- READ-COMMITTED

→ A NIVEL DE CLUSTER

- SERIALIZABLE

→ A NIVEL DE NODO

Galera Cluster: Niveles de Aislamiento

- Veamos la documentación oficial brevemente:

https://galeracluster.com/library/documentation/isolation-levels.html

Revisamos la configuración de Galera

Explicación en vivo de ficheros de infra

Hands on: Lanzamos ProxySQL

- Vamos a la carpeta de contenido del taller
- Lanzamos ProxySQL

Configuración

- Métodos de configuration:
  - Configuración de inicio:
    - Fichero de configuración. E.g: proxysql.cnf
  - Admin interface:
    - Configuración en caliente.

Configuración

- Métodos de configuration:
  - Configuración de inicio:
    - Fichero de configuración. E.g: proxysql.cnf
  - Admin interface:
    - Configuración en caliente.

¡SOLO SE EJECUTA UNA VEZ!

Configuración

FICHERO DE CONFIGURACIÓN

Configuración y ejemplo con MySQL cluster

- Admin interface:
  - Interfaz compatible con cliente MySQL.
  - Sirve para:
    - Realizar configuraciones en caliente
    - Guardar la nueva configuración en disco
    - Inspeccionar el estado actual del proxy

Configuración

- Admin interface:

```
→  ESLIBRE mysql -h127.0.0.1 -P6032 -uradmin -pradmin
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 43
Server version: 8.0 (ProxySQL Admin Module)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Configuración

Configuración esencial:

- mysql_servers
- mysql_users
- monitoring user

Configuración

Configuración esencial:

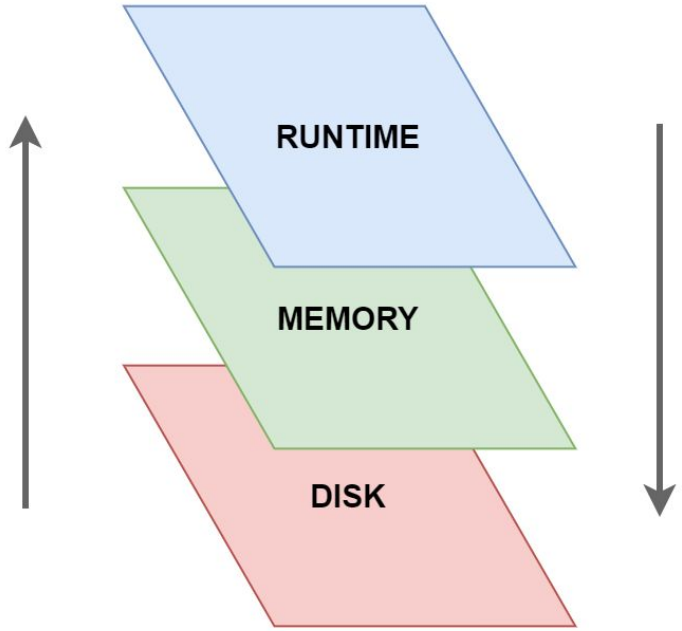
- mysql_servers
- mysql_users
- monitoring user

Tres capas de configuración

Configuración

LOAD {*} TO RUNTIME:

- MYSQL QUERY RULES
- MYSQL SERVERS
- MYSQL VARIABLES
- MYSQL USERS
- ADMIN VARIABLES
- ETC...



SAVE {*} TO DISK:

- MYSQL QUERY RULES
- MYSQL SERVERS
- MYSQL VARIABLES
- MYSQL USERS
- ADMIN VARIABLES
- ETC...

Configuración y ejemplo con MySQL cluster

- mysql_servers:

```
mysql> select * from runtime_mysql_servers;
+--------------+-----------+-------+-----------+--------+--------+-------------+-----------------+---------------------+---------+----------------+---------+
| hostgroup_id | hostname  | port  | gtid_port | status | weight | compression | max_connections | max_replication_lag | use_ssl | max_latency_ms | comment |
+--------------+-----------+-------+-----------+--------+--------+-------------+-----------------+---------------------+---------+----------------+---------+
| 0            | 127.0.0.1 | 13306 | 0         | ONLINE | 1      | 0           | 1000            | 180                 | 0       | 0              | mysql1  |
| 1            | 127.0.0.1 | 13308 | 0         | ONLINE | 1      | 0           | 1000            | 180                 | 0       | 0              | mysql3  |
| 1            | 127.0.0.1 | 13307 | 0         | ONLINE | 1      | 0           | 1000            | 180                 | 0       | 0              | mysql2  |
| 1            | 127.0.0.1 | 13306 | 0         | ONLINE | 1      | 0           | 1000            | 180                 | 0       | 0              | mysql1  |
+--------------+-----------+-------+-----------+--------+--------+-------------+-----------------+---------------------+---------+----------------+---------+
4 rows in set (0.00 sec)
```

Configuración y ejemplo con MySQL cluster

- mysql_servers:

```
mysql> select * from runtime_mysql_servers;
+--------------+-----------+-------+-----------+--------+--------+-------------+-----------------+---------------------+---------+----------------+---------+
| hostgroup_id | hostname  | port  | gtid_port | status | weight | compression | max_connections | max_replication_lag | use_ssl | max_latency_ms | comment |
+--------------+-----------+-------+-----------+--------+--------+-------------+-----------------+---------------------+---------+----------------+---------+
| 0            | 127.0.0.1 | 13306 | 0         | ONLINE | 1      | 0           | 1000            | 180                 | 0       | 0              | mysql1  |
| 1            | 127.0.0.1 | 13308 | 0         | ONLINE | 1      | 0           | 1000            | 180                 | 0       | 0              | mysql3  |
| 1            | 127.0.0.1 | 13307 | 0         | ONLINE | 1      | 0           | 1000            | 180                 | 0       | 0              | mysql2  |
| 1            | 127.0.0.1 | 13306 | 0         | ONLINE | 1      | 0           | 1000            | 180                 | 0       | 0              | mysql1  |
+--------------+-----------+-------+-----------+--------+--------+-------------+-----------------+---------------------+---------+----------------+---------+
4 rows in set (0.00 sec)
```

Configurados en dos hotgroups en ‘read/write’ split.

Configuración y ejemplo con MySQL cluster

- mysql_users:

```
mysql> select username, password, active, use_ssl, default_hostgroup, default_schema, transaction_persistent, fast_forward, backend, frontend from mysql_users;
+------------------+----------+--------+---------+-------------------+----------------+------------------------+--------------+---------+----------+
| username         | password | active | use_ssl | default_hostgroup | default_schema | transaction_persistent | fast_forward | backend | frontend |
+------------------+----------+--------+---------+-------------------+----------------+------------------------+--------------+---------+----------+
| root             | root     | 1      | 0       | 0                 | NULL           | 1                      | 0            | 1       | 1        |
| user             | user     | 1      | 0       | 0                 | NULL           | 1                      | 0            | 1       | 1        |
| proxysql         | eslibre  | 1      | 0       | 0                 | NULL           | 1                      | 0            | 1       | 1        |
| congress_user    | eslibre  | 1      | 0       | 0                 | NULL           | 1                      | 0            | 1       | 1        |
| proxysql_default | eslibre  | 1      | 0       | 0                 | eslibre        | 1                      | 0            | 1       | 1        |
+------------------+----------+--------+---------+-------------------+----------------+------------------------+--------------+---------+----------+
5 rows in set (0.00 sec)
```

Configuración y ejemplo con MySQL cluster

- Monitoring user:

```
mysql> select * from global_variables where variable_name like "%monitor%" limit 8 offset 22;
+---------------------------------+----------------+
| variable_name                   | variable_value |
+---------------------------------+----------------+
| mysql-monitor_username          | monitor        |
| mysql-monitor_password          | monitor        |
| mysql-monitor_history           | 600000         |
| mysql-monitor_connect_interval  | 10000          |
| mysql-monitor_connect_timeout   | 1000           |
| mysql-monitor_ping_interval     | 10000          |
| mysql-monitor_read_only_interval | 1500           |
| mysql-monitor_read_only_timeout | 500            |
+---------------------------------+----------------+
8 rows in set (0.00 sec)
```

Configuración

CONFIGURAMOS PROXYSQL

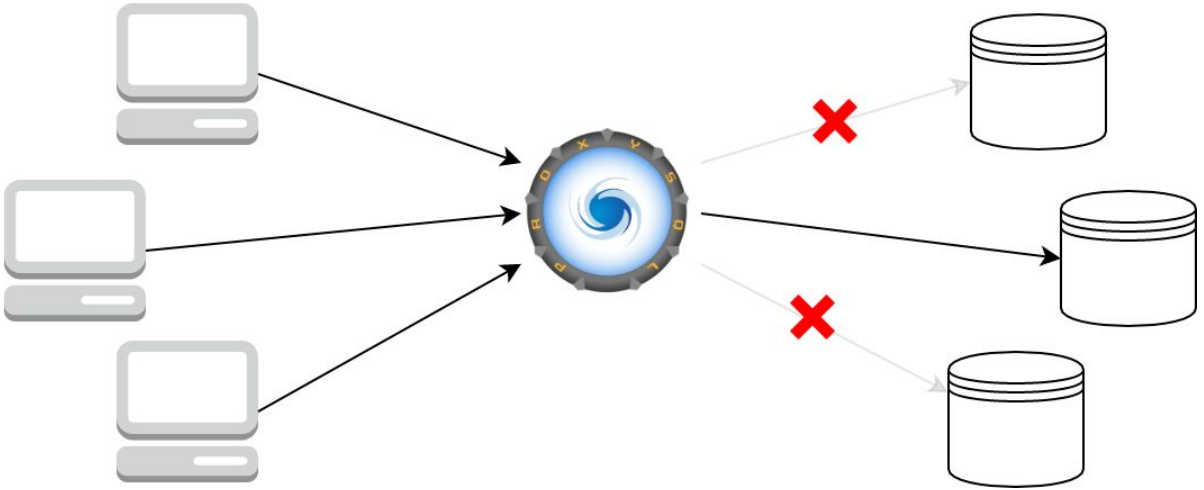
Balanceo de carga, demultiplexión y thread pool

- Demultiplexing

- Thread pool

Balanceo de carga, demultiplexión y thread pool

- Demultiplexing
- Thread pool



Reglas para consultas

- Enrutamiento
- Firewall

Reglas para consultas: mysql_query_rules

```
mysql> SHOW CREATE TABLE mysql_query_rules\G
*************************** 1. row ***************************
       table: mysql_query_rules
Create Table: CREATE TABLE mysql_query_rules (
    rule_id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    active INT CHECK (active IN (0,1)) NOT NULL DEFAULT 0,
    username VARCHAR,
    schemaname VARCHAR,
    flagIN INT CHECK (flagIN >= 0) NOT NULL DEFAULT 0,
    client_addr VARCHAR,
    proxy_addr VARCHAR,
    proxy_port INT CHECK (proxy_port >= 0 AND proxy_port <= 65535), digest VARCHAR,
    match_digest VARCHAR,
    match_pattern VARCHAR,
    negate_match_pattern INT CHECK (negate_match_pattern IN (0,1)) NOT NULL DEFAULT 0,
    re_modifiers VARCHAR DEFAULT 'CASELESS',
    flagOUT INT CHECK (flagOUT >= 0), replace_pattern VARCHAR CHECK(CASE WHEN replace_pattern IS NULL THEN 1 WHEN replace_pattern IS NOT NULL AND match_pattern IS NOT
    destination_hostgroup INT DEFAULT NULL,
    cache_ttl INT CHECK(cache_ttl > 0),
    cache_empty_result INT CHECK (cache_empty_result IN (0,1)) DEFAULT NULL,
    cache_timeout INT CHECK(cache_timeout >= 0),
    reconnect INT CHECK (reconnect IN (0,1)) DEFAULT NULL,
    timeout INT UNSIGNED CHECK (timeout >= 0),
    retries INT CHECK (retries>=0 AND retries <=1000),
    delay INT UNSIGNED CHECK (delay >=0),
    next_query_flagIN INT UNSIGNED,
    mirror_flagOUT INT UNSIGNED,
    mirror_hostgroup INT UNSIGNED,
    error_msg VARCHAR,
    OK_msg VARCHAR,
    sticky_conn INT CHECK (sticky_conn IN (0,1)),
    multiplex INT CHECK (multiplex IN (0,1,2)),
    gtid_from_hostgroup INT UNSIGNED,
    log INT CHECK (log IN (0,1)),
    apply INT CHECK(apply IN (0,1)) NOT NULL DEFAULT 0,
    comment VARCHAR)
1 row in set (0.00 sec)
```

Reglas para consultas: mysql_query_rules

```
mysql> SHOW CREATE TABLE mysql_query_rules\G
*************************** 1. row ***************************
       table: mysql_query_rules
Create Table: CREATE TABLE mysql_query_rules (
    rule_id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    active INT CHECK (active IN (0,1)) NOT NULL DEFAULT 0,
    username VARCHAR,
    schemaname VARCHAR,
    flagIN INT CHECK (flagIN >= 0) NOT NULL DEFAULT 0,
    client_addr VARCHAR,
    proxy_addr VARCHAR,
    proxy_port INT CHECK (proxy_port >= 0 AND proxy_port <= 65535), digest VARCHAR,
    match_digest VARCHAR,
    match_pattern VARCHAR,
    negate_match_pattern INT CHECK (negate_match_pattern IN (0,1)) NOT NULL DEFAULT 0,
    re_modifiers VARCHAR DEFAULT 'CASELESS',
    flagOUT INT CHECK (flagOUT >= 0), replace_pattern VARCHAR CHECK(CASE WHEN replace_pattern IS NULL THEN 1 WHEN replace_pattern IS NOT NULL AND match_pattern IS NOT
    destination_hostgroup INT DEFAULT NULL,
    cache_ttl INT CHECK(cache_ttl > 0),
    cache_empty_result INT CHECK (cache_empty_result IN (0,1)) DEFAULT NULL,
    cache_timeout INT CHECK(cache_timeout >= 0),
    reconnect INT CHECK (reconnect IN (0,1)) DEFAULT NULL,
    timeout INT UNSIGNED CHECK (timeout >= 0),
    retries INT CHECK (retries>=0 AND retries <=1000),
    delay INT UNSIGNED CHECK (delay >=0),
    next_query_flagIN INT UNSIGNED,
    mirror_flagOUT INT UNSIGNED,
    mirror_hostgroup INT UNSIGNED,
    error_msg VARCHAR,
    OK_msg VARCHAR,
    sticky_conn INT CHECK (sticky_conn IN (0,1)),
    multiplex INT CHECK (multiplex IN (0,1,2)),
    gtid_from_hostgroup INT UNSIGNED,
    log INT CHECK (log IN (0,1)),
    apply INT CHECK(apply IN (0,1)) NOT NULL DEFAULT 0,
    comment VARCHAR)
1 row in set (0.00 sec)
```

QUIZÁS DEMASIADA INFO…

Reglas para consultas: mysql_query_rules

- Centrémonos en lo básico:
  - rule_id
  - active
  - username
  - schema
  - match_digest
  - match_pattern
  - retries
  - apply
  - flagin
  - flagout

Reglas para consultas: mysql_query_rules

- Reglas simples de Read/Write split:

```
mysql> SELECT rule_id, username, destination_hostgroup, schemaname, active, retries, apply, match_digest FROM mysql_query_rule
s;
+---------+----------+-----------------------+------------+--------+---------+-------+-----------------------------------+
| rule_id | username | destination_hostgroup | schemaname | active | retries | apply | match_digest                      |
+---------+----------+-----------------------+------------+--------+---------+-------+-----------------------------------+
| 1       | NULL     | 0                     | NULL       | 0      | NULL    | 1     | ^SELECT.*FOR UPDATE               |
| 2       | NULL     | 1                     | NULL       | 0      | NULL    | 1     | ^SELECT                           |
```

Reglas para consultas: mysql_query_rules

- Reglas simples de Read/Write split:

```
mysql> SELECT rule_id, username, destination_hostgroup, schemaname, active, retries, apply, match_digest FROM mysql_query_rule
s;
+---------+----------+-----------------------+------------+--------+---------+-------+-----------------------------------+
| rule_id | username | destination_hostgroup | schemaname | active | retries | apply | match_digest                      |
+---------+----------+-----------------------+------------+--------+---------+-------+-----------------------------------+
| 1       | NULL     | 0                     | NULL       | 0      | NULL    | 1     | ^SELECT.*FOR UPDATE               |
| 2       | NULL     | 1                     | NULL       | 0      | NULL    | 1     | ^SELECT                           |
```

ATENCIÓN: NO USAR NUNCA EN PRODUCCIÓN

Reglas para consultas: mysql_query_rules

CREAMOS NUESTRAS QUERY RULES

¿Preguntas?

Feliz ProxySQLing!