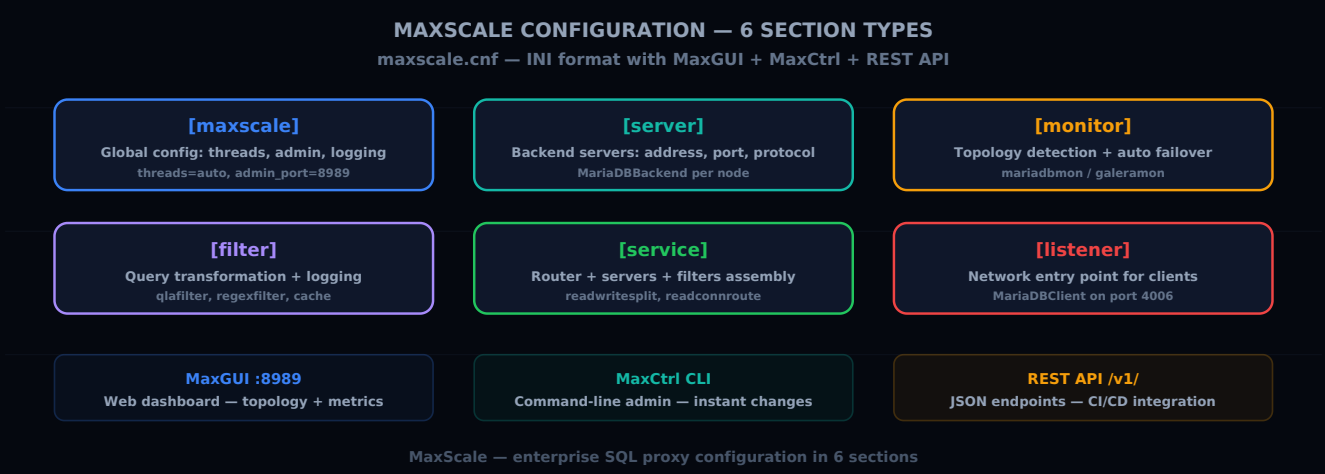
MaxScale : bien plus qu'un reverse proxy SQL (partie 2)

Sylvain ARBAUDIE · 1 septembre 2025

MAXSCALE MARIADB PROXY CONFIGURATION



De la théorie à la pratique

Dans la première partie, nous avons exploré les capacités architecturales de MaxScale : ses protocoles, ses routeurs, ses moniteurs et ses filtres. Cette seconde partie est un guide pratique : comment installer, configurer et administrer MaxScale au quotidien.

Installation

MaxScale est disponible via le dépôt officiel MariaDB. Sur Debian/Ubuntu :

```
curl -LsS https://r.mariadb.com/downloads/mariadb_repo_setup \
  | sudo bash -s -- --mariadb-maxscale-version=24.02

sudo apt-get install maxscale
```

Le fichier de configuration principal se trouve dans `/etc/maxscale.cnf`. C'est un fichier au format INI, organisé en sections clairement définies.

Anatomie du fichier de configuration

Le fichier `maxscale.cnf` est organisé en six types de sections, chacune identifiée par un en-tête entre crochets.

Section [maxscale] — Configuration globale

```
[maxscale]
threads        = auto
log_info       = false
admin_host     = 0.0.0.0
admin_port     = 8989
admin_secure_gui = true
```

- **threads=auto** : MaxScale détecte automatiquement le nombre de cœurs CPU disponibles.
- **admin_host/admin_port** : Active l'interface web MaxGUI et l'API REST.
- **admin_secure_gui** : Active HTTPS pour l'interface d'administration.

Section [server] — Définition des serveurs

Chaque serveur MariaDB / MySQL backend est déclaré individuellement :

```
[db-master]
type     = server
address  = 10.0.1.10
port     = 3306
protocol = MariaDBBackend

[db-slave1]
type     = server
address  = 10.0.1.11
port     = 3306
protocol = MariaDBBackend

[db-slave2]
type     = server
address  = 10.0.1.12
port     = 3306
protocol = MariaDBBackend
```

Chaque section `[server]` porte un nom logique (ici `db-master`, `db-slave1`, `db-slave2`) qui sera référencé par les moniteurs et les services.

Section [monitor] — Surveillance de la topologie

Le moniteur est le composant qui détecte automatiquement la topologie de réplication et l'état de chaque nœud :

```
[replication-monitor]
type            = monitor
module          = mariadbmon
servers         = db-master, db-slave1, db-slave2
user            = maxscale_monitor
password        = encrypted_password_here
monitor_interval = 2000
auto_failover    = true
auto_rejoin      = true
```

- **mariadbmon** : Le moniteur standard pour les topologies master-slave MariaDB / MySQL.
- **auto_failover** : En cas de perte du master, MaxScale promeut automatiquement un slave.
- **auto_rejoin** : Un ancien master qui revient en ligne est automatiquement reconfiguré comme slave.

Pour Galera, on utiliserait `galeramon` à la place de `mariadbmon`.

Section [filter] — Transformation des requêtes

Les filtres interceptent et modifient le flux SQL :

```
[query-log]
type   = filter
module = qlafilter
filebase = /var/log/maxscale/queries
flush   = true

[regex-rewrite]
type    = filter
module  = regexfilter
match   = SELECT.*FROM\s+legacy_table
replace = SELECT * FROM new_table
```

Le `qlafilter` enregistre toutes les requêtes (utile pour l'audit). Le `regexfilter` permet de réécrire des requêtes à la volée, sans modifier l'application.

Section [service] — Assemblage des composants

Le service lie les serveurs, les moniteurs, les filtres et le routeur :

```
[rw-service]
type     = service
router   = readwritesplit
servers  = db-master, db-slave1, db-slave2
user     = maxscale_service
password = encrypted_password_here
filters  = query-log | regex-rewrite

max_slave_connections    = 100%
max_slave_replication_lag = 5s
```

- **readwritesplit** : Les écritures vont au master, les lectures sont distribuées sur les slaves.
- **max_slave_replication_lag** : Un slave en retard de plus de 5 secondes est temporairement exclu du routage.
- **filters** : Les filtres sont chaînés avec le pipe `|`.

Section [listener] — Point d'entrée réseau

Le listener expose le service sur un port réseau :

```
[rw-listener]
type     = listener
service  = rw-service
protocol = MariaDBClient
port     = 4006
address  = 0.0.0.0
```

L'application se connecte à MaxScale sur le port 4006 comme s'il s'agissait d'un serveur MariaDB / MySQL standard. MaxScale est transparent du point de vue du protocole.

MaxGUI : l'interface web

Depuis MaxScale 2.5, l'interface web MaxGUI offre une vue graphique complète de l'infrastructure. Accessible via `https://maxscale-host:8989`, elle permet de :

- Visualiser la topologie en temps réel (master, slaves, état des connexions)
- Voir les métriques de performance (requêtes/seconde, latence, connexions actives)
- Gérer les serveurs (drain, maintenance, ajout)
- Consulter les logs et les alertes

MaxGUI est un outil de supervision, pas de configuration. Les modifications de configuration se font via le fichier `maxscale.cnf` ou via l'API REST.

MaxCtrl : l'outil en ligne de commande

MaxCtrl est le CLI officiel de MaxScale. Il communique avec l'API REST et offre une syntaxe intuitive :

```
# Lister les serveurs et leur état
maxctrl list servers

# Mettre un serveur en maintenance
maxctrl set server db-slave1 maintenance

# Retirer le mode maintenance
maxctrl clear server db-slave1 maintenance

# Lister les services et leurs statistiques
maxctrl list services

# Voir la topologie de réplication
maxctrl show monitor replication-monitor

# Créer un filtre dynamiquement
maxctrl create filter my-filter regexfilter \
  match="SELECT 1" replace="SELECT 2"
```

L'avantage de MaxCtrl sur l'édition directe du fichier est la prise en compte immédiate des modifications, sans redémarrage.

API REST : l'automatisation

L'API REST de MaxScale est le fondement de l'automatisation. Elle expose toutes les fonctionnalités via des endpoints HTTP/JSON :

```
# Obtenir la liste des serveurs
curl -s -u admin:password https://maxscale:8989/v1/servers | jq

# Failover manuel
curl -X POST -u admin:password \
  https://maxscale:8989/v1/monitors/replication-monitor/failover

# Obtenir les métriques du service
curl -s -u admin:password \
  https://maxscale:8989/v1/services/rw-service | jq '.data.attributes.statistics'
```

L'API REST permet d'intégrer MaxScale dans des pipelines CI/CD, des systèmes de monitoring (Prometheus, Grafana), et des outils d'orchestration (Ansible, Terraform).

Bonnes pratiques de configuration

Voici quelques recommandations issues de l'expérience terrain :

1. **Utilisez toujours des mots de passe chiffrés** dans le fichier de configuration. MaxScale fournit l'outil `maxkeys` pour générer les clés de chiffrement.
2. **Activez le failover automatique** avec `auto_failover=true`, mais testez-le régulièrement en environnement de pré-production.
3. **Dimensionnez les threads** selon la charge. `threads=auto` est un bon défaut, mais pour les très gros volumes (>50 000 requêtes/seconde), un tuning manuel peut être nécessaire.
4. **Surveillez le replication lag** via `max_slave_replication_lag`. Une valeur trop permissive envoie des lectures sur des slaves en retard. Une valeur trop restrictive concentre toutes les lectures sur le master.
5. **Utilisez les filtres avec parcimonie**. Chaque filtre ajoute une couche de traitement. Le `regexfilter` en particulier peut avoir un impact significatif sur les performances si les expressions régulières sont complexes.

Configuration type complète

Pour récapituler, voici un fichier de configuration type pour une topologie master + 2 slaves :

```
[maxscale]
threads = auto
admin_host = 0.0.0.0
admin_port = 8989

[db-master]
type = server
address = 10.0.1.10
port = 3306
protocol = MariaDBBackend

[db-slave1]
type = server
address = 10.0.1.11
port = 3306
protocol = MariaDBBackend

[db-slave2]
type = server
address = 10.0.1.12
port = 3306
protocol = MariaDBBackend

[replication-monitor]
type = monitor
module = mariadbmon
servers = db-master, db-slave1, db-slave2
user = maxscale_monitor
password = <encrypted>
monitor_interval = 2000
auto_failover = true
auto_rejoin = true

[rw-service]
type = service
router = readwritesplit
servers = db-master, db-slave1, db-slave2
user = maxscale_service
password = <encrypted>
```

```
max_slave_replication_lag = 5s

[rw-listener]
type = listener
service = rw-service
protocol = MariaDBClient
port = 4006
```

Conclusion

MaxScale se configure avec un fichier INI clair et structuré. Six types de sections suffisent pour décrire une infrastructure complète : la configuration globale, les serveurs, le moniteur, les filtres, le service et le listener.

L'administration au quotidien passe par trois outils complémentaires : MaxGUI pour la supervision visuelle, MaxCtrl pour les opérations en ligne de commande, et l'API REST pour l'automatisation. Ensemble, ils forment un écosystème complet pour gérer un proxy SQL de niveau entreprise.

Cet article a été initialement publié sur Medium.