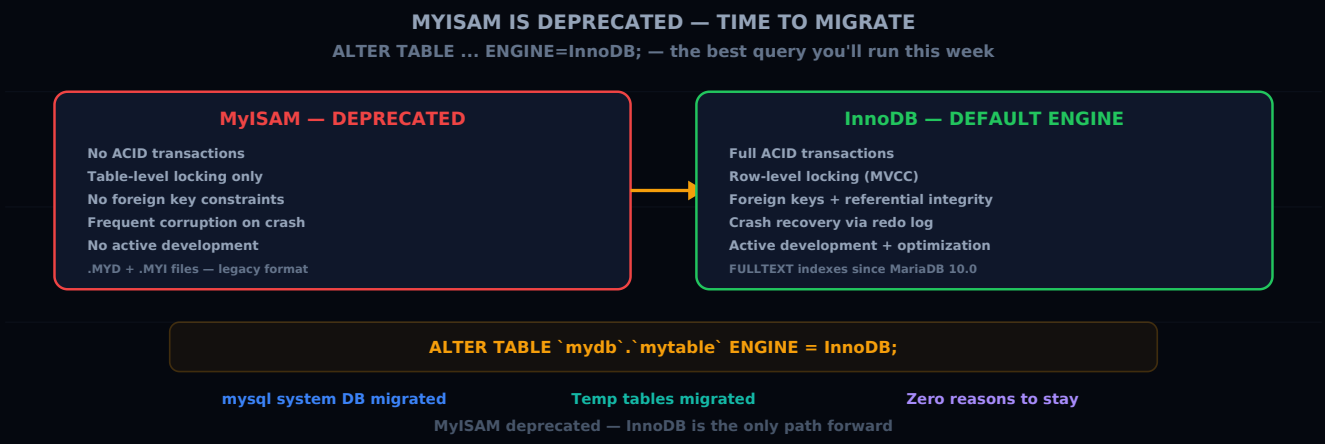
MyISAM est obsolète : le moment de migrer

Sylvain ARBAUDIE · 4 août 2025

MYSQL MYISAM INNODB MIGRATION



La fin d'une époque

MyISAM est officiellement obsolète. Ce n'est pas une surprise — c'était une évidence depuis des années. Mais cette fois, c'est acté dans le code source de MariaDB / MySQL : le moteur MyISAM est marqué comme deprecated, et les derniers bastions qui l'utilisaient en interne ont été migrés.

La base de données système `mysql` (celle qui stocke les utilisateurs, les privilèges, les tables de grant) n'utilise plus MyISAM. Les tables temporaires internes non plus. Les deux dernières excuses pour tolérer MyISAM en production viennent de disparaître.

Pourquoi MyISAM a survécu si longtemps

Pour comprendre la situation actuelle, il faut remonter à l'histoire de MyISAM et de son rôle dans l'écosystème.

MyISAM était le moteur de stockage par défaut de MySQL jusqu'à la version 5.5 (2010). Pendant plus d'une décennie, il a été le choix par défaut pour des millions d'applications web. WordPress, Joomla, Drupal, phpBB — toutes ces applications ont été développées et testées principalement avec MyISAM.

Les avantages de MyISAM étaient réels à l'époque :

- **Simplicité** : un fichier `.MYD` pour les données, un fichier `.MYI` pour les index. Facile à sauvegarder, facile à déplacer.
- **Performances en lecture** : pour les charges de travail en lecture pure (blogs, sites vitrines), MyISAM était rapide.
- **Full-text search** : MyISAM supportait la recherche plein texte bien avant InnoDB.
- **Empreinte mémoire faible** : MyISAM utilisait peu de RAM, ce qui était crucial à l'époque des serveurs à 512 Mo.

Mais ces avantages sont devenus des vestiges. InnoDB offre aujourd'hui tout cela et bien plus.

Pourquoi vous devez migrer maintenant

Pas de transactions

MyISAM ne supporte pas les transactions ACID. Pas de `BEGIN`, pas de `COMMIT`, pas de `ROLLBACK`. Chaque instruction est auto-commitée. En cas de crash pendant une écriture, vos données sont dans un état indéterminé.

Pas de verrouillage au niveau de la ligne

MyISAM utilise un verrouillage au niveau de la table. Une seule écriture verrouille la table entière, bloquant toutes les autres lectures et écritures. Avec InnoDB, le verrouillage se fait au niveau de la ligne, permettant la concurrence.

Pas de clés étrangères

MyISAM ne supporte pas les contraintes de clés étrangères. Pas d'intégrité référentielle au niveau de la base. Vous dépendez entièrement de l'application pour maintenir la cohérence des données.

Corruption fréquente

Les tables MyISAM sont notoirement fragiles. Un arrêt brutal du serveur, un disque plein, un `kill -9` du processus mysqld — et vos tables sont corrompues. `myisamchk` et `REPAIR TABLE` deviennent vos meilleurs amis, mais ils ne sont pas infaillibles.

Plus de développement actif

C'est peut-être l'argument le plus important. Personne ne travaille plus sur MyISAM. Pas de correctifs de bugs, pas d'optimisations de performances, pas de nouvelles fonctionnalités. C'est du code gelé qui accumule la dette technique.

La migration : plus simple qu'on ne le pense

La bonne nouvelle, c'est que migrer de MyISAM vers InnoDB est généralement straightforward.

Identifier les tables MyISAM

```
SELECT table_schema, table_name, engine, table_rows,
       ROUND(data_length / 1024 / 1024, 2) AS data_mb
FROM information_schema.tables
WHERE engine = 'MyISAM'
  AND table_schema NOT IN ('mysql', 'information_schema',
                           'performance_schema', 'sys')
ORDER BY data_length DESC;
```

Convertir une table

```
ALTER TABLE mydb.mytable ENGINE = InnoDB;
```

C'est tout. MariaDB / MySQL reconstruit la table avec le moteur InnoDB. Les index sont recréés, les données sont copiées. Pour les petites tables, c'est instantané. Pour les grosses tables, cela peut prendre quelques minutes.

Points d'attention

Quelques cas particuliers à surveiller lors de la migration :

Tables full-text : Si vous utilisez des index FULLTEXT sur MyISAM, bonne nouvelle — InnoDB supporte les index FULLTEXT depuis MySQL 5.6 / MariaDB 10.0. La syntaxe est identique.

Tables MERGE : Si vous utilisez le moteur MERGE (une union de tables MyISAM), vous devrez repenser votre architecture. Le partitionnement InnoDB ou les vues sont des alternatives.

**COUNT()* sans WHERE* *: MyISAM stocke le nombre exact de lignes, ce qui rend `SELECT COUNT(* ) FROM table `instantané. InnoDB doit scanner un index. Si votre application fait souvent des* COUNT(*)` sans condition, vous remarquerez une différence (mineure pour les tables de moins

d'un million de lignes).

Espace disque : InnoDB utilise plus d'espace disque que MyISAM pour les mêmes données (en moyenne 1,5 à 2x plus), principalement à cause du MVCC et de la gestion des transactions. Vérifiez votre espace disponible avant la migration.

Script de migration en masse

Pour les bases avec de nombreuses tables MyISAM, voici une approche systématique :

```
-- Générer les commandes ALTER TABLE
SELECT CONCAT('ALTER TABLE `', table_schema, '`.`', table_name,
              '` ENGINE=InnoDB;') AS migration_sql
FROM information_schema.tables
WHERE engine = 'MyISAM'
  AND table_schema NOT IN ('mysql', 'information_schema',
                           'performance_schema', 'sys')
ORDER BY data_length ASC;
```

Commencez par les plus petites tables pour valider le processus, puis passez aux plus grosses.

Après la migration

Une fois toutes vos tables migrées en InnoDB, quelques ajustements de configuration sont recommandés :

```
# my.cnf
[mysqld]
default_storage_engine = InnoDB
innodb_buffer_pool_size = 70%  # de la RAM disponible
innodb_log_file_size = 256M    # ou plus selon la charge
innodb_flush_log_at_trx_commit = 1  # durabilité complète
```

Et désactivez les fonctionnalités MyISAM dont vous n'avez plus besoin :

```
skip-external-locking
key_buffer_size = 8M  # minimum, pour les tables système restantes
```

Conclusion

MyISAM est obsolète. Ce n'est plus une opinion, c'est un fait technique. La base système a migré, les tables temporaires ont migré, le code est en mode maintenance sans futur.

Si vous avez encore des tables MyISAM en production, le moment de migrer est maintenant. La conversion est simple, les bénéfices sont immédiats (transactions, verrouillage par ligne, crash recovery), et les risques de rester sur MyISAM ne font qu'augmenter.

`ALTER TABLE ... ENGINE=InnoDB;` — c'est la meilleure requête que vous exécuterez cette semaine.

Cet article a été initialement publié sur Medium.