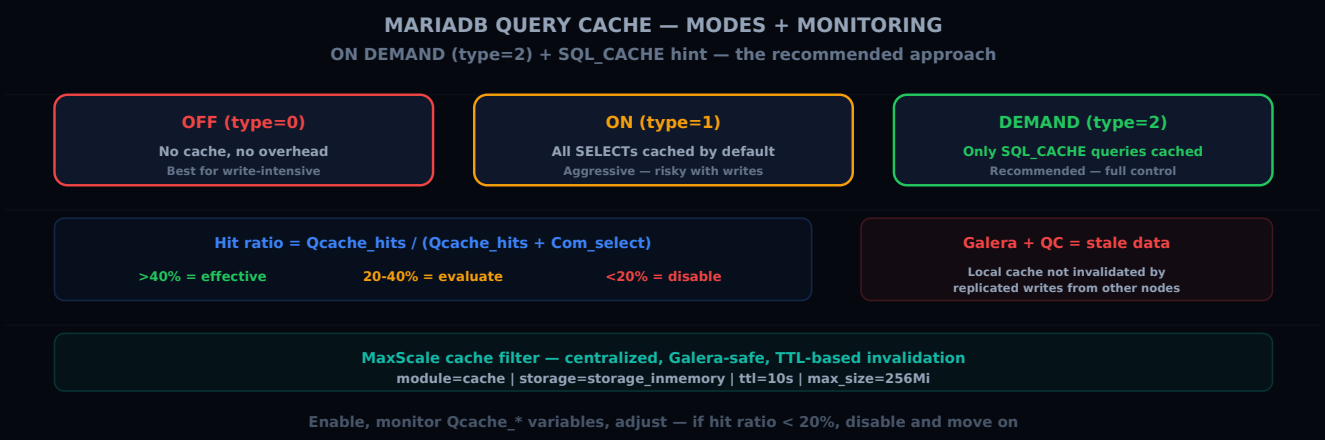
Эксперименты с MariaDB: query cache

Sylvain ARBAUDIE · June 30, 2025

MARIADB QUERY-CACHE PERFORMANCE TUNING



Концепция Query Cache

Query cache в MariaDB / MySQL — это встроенный в сервер механизм, который хранит результаты запросов SELECT в памяти. Когда идентичный запрос отправляется повторно, сервер возвращает результат прямо из кэша, не выполняя запрос. Это просто, элегантно и потенциально очень эффективно.

Ключевое слово здесь — «потенциально». Query cache — одна из самых плохо понимаемых и самых неправильно настроенных функций MariaDB / MySQL. При правильном использовании он может сократить время ответа в 10 раз. При неправильной настройке — убить производительность.

Три режима

Query cache может работать в трёх режимах, управляемых переменной `query_cache_type`:

OFF (0)

Query cache полностью отключён. Никакие запросы не кэшируются, никаких проверок кэша не выполняется. Это самый безопасный вариант для нагрузок с интенсивной записью.

ON (1)

Все запросы SELECT кэшируются по умолчанию, кроме помеченных хинтом `SQL_NO_CACHE`. Это самый агрессивный режим.

```
-- Этот запрос будет кэширован
SELECT * FROM products WHERE category_id = 5;

-- Этот запрос НЕ будет кэширован
SELECT SQL_NO_CACHE * FROM products WHERE category_id = 5;
```

ON DEMAND (2)

Никакие запросы не кэшируются по умолчанию. Только запросы, явно помеченные `SQL_CACHE`, будут кэшированы. Это самый контролируемый и часто самый эффективный режим.

```
-- Этот запрос НЕ будет кэширован (поведение по умолчанию)
SELECT * FROM products WHERE category_id = 5;

-- Этот запрос БУДЕТ кэширован
SELECT SQL_CACHE * FROM products WHERE category_id = 5;
```

Режим `ON DEMAND` — тот, который я рекомендую в большинстве случаев. Он заставляет задуматься, какие запросы заслуживают кэширования, вместо слепого кэширования всего.

Инвалидация: ахиллесова пята

Query cache инвалидирует ВЕСЬ кэш таблицы при любой записи в эту таблицу. Не только затронутые строки — всю таблицу.

```
-- Предположим, что 1000 SELECT по таблице 'products' в кэше
UPDATE products SET price = 19.99 WHERE product_id = 42;
-- → Все 1000 записей кэша для 'products' инвалидируются
```

Это грубый, но необходимый механизм для гарантии согласованности. Проблема в том, что для часто изменяемых таблиц кэш постоянно инвалидируется и перестраивается, что потребляет больше ресурсов, чем отсутствие кэша вообще.

Размер кэша

Размер query cache управляется параметром `query_cache_size`:

```
[mysqld]
query_cache_type = 2
query_cache_size = 64M
query_cache_limit = 2M
query_cache_min_res_unit = 2048
```

- **query_cache_size**: общий размер кэша. Не превышайте 256 МБ — выше глобальный мьютекс кэша становится узким местом.
- **query_cache_limit**: максимальный размер отдельного результата. Результаты крупнее не кэшируются.
- **query_cache_min_res_unit**: размер блока аллокации. Уменьшение этого значения для маленьких результатов снижает фрагментацию.

Мониторинг кэша

Переменные статуса `Qcache_*` необходимы для оценки эффективности:

```
SHOW GLOBAL STATUS LIKE 'Qcache%';
```

Ключевые метрики:

Переменная	Описание
`Qcache_hits`	Количество запросов, обслуженных из кэша
`Qcache_inserts`	Количество запросов, добавленных в кэш
`Qcache_not_cached`	Запросы, не кэшированные (слишком большие, хинты и т.д.)
`Qcache_lowmem_prunes`	Вытеснения из-за нехватки памяти
`Qcache_free_memory`	Свободная память в кэше
`Qcache_total_blocks`	Общее количество выделенных блоков
`Qcache_free_blocks`	Свободные блоки (высокое значение = фрагментация)

Коэффициент эффективности

Самый важный коэффициент — **hit ratio**:

```
Hit ratio = Qcache_hits / (Qcache_hits + Com_select) × 100
```

Интерпретация:

- **> 40%**: кэш эффективен, стоит держать включённым
- **20-40%**: серая зона, оценивайте индивидуально
- **< 20%**: кэш неэффективен, рассмотрите отключение

Второй важный коэффициент — **коэффициент вытеснения**:

```
Eviction ratio = Qcache_lowmem_prunes / Qcache_inserts × 100
```

Если этот коэффициент превышает 10%, кэш слишком мал — увеличьте `query_cache_size` или сократите объём кэшируемых данных.

Ловушка конкурентности

Query cache использует **глобальный мьютекс**. Это означает, что только один поток может читать из кэша или записывать в него одновременно. На сервере со 100 одновременными соединениями этот мьютекс становится серьёзным узким местом.

Именно поэтому MySQL 8.0 удалил query cache. Команда Oracle решила, что стоимость мьютекса превышает преимущества кэша при современных нагрузках (высокая конкурентность, частая запись).

MariaDB решила сохранить его, считая его полезным для некоторых специфических сценариев (нагрузка на чтение, низкая конкурентность, редко изменяемые таблицы).

Query cache и Galera: сложная комбинация

Использование query cache в кластере Galera проблематично. Galera реплицирует записи на все узлы, но query cache локален для каждого узла. Результат:

1. Узел A получает SELECT и кэширует результат

2. Узел B получает UPDATE через репликацию Galera
3. Кэш узла A НЕ инвалидируется — он не знает, что данные изменились
4. Следующий SELECT на узле A возвращает устаревшие данные

Единственный безопасный способ использовать query cache с Galera — с `wsrep_causal_reads = ON`, который принуждает к проверке согласованности перед каждым чтением. Но это во многом нивелирует преимущество кэша.

Альтернатива: cache filter MaxScale

Для архитектур, которым нужен распределённый кэш запросов, cache filter MaxScale — лучший подход:

```
[query-cache]
type = filter
module = cache
storage = storage_inmemory
ttl = 10s
max_size = 256Mi
```

Кэш MaxScale централизован (на уровне прокси), что устраняет проблему несогласованности между узлами Galera. Кроме того, MaxScale может инвалидировать кэш интеллектуально, на основе типа запроса, а не только по изменённой таблице.

Когда активировать query cache

Query cache полезен, когда:

- Нагрузка **преимущественно на чтение** (> 80% SELECT)
- Таблицы **редко изменяются** (таблицы конфигурации, справочники)
- **Конкурентность умеренная** (< 50 одновременных соединений)
- **Одни и те же запросы повторяются** часто (веб-приложения с cache miss)
- Вы на **standalone-сервере**, а не в кластере Galera

Query cache НЕ подходит, когда:

- Нагрузка **смешанная чтение/запись**

- Таблицы **часто изменяются** (транзакционные таблицы)
- **Конкурентность высокая** (глобальный мьютекс)
- Вы в **кластере Galera** (несогласованность кэша)

Заключение

Query cache MariaDB — мощный, но деликатный инструмент. Режим `ON DEMAND` с хинтом `SQL_CACHE` обеспечивает лучший контроль. Мониторинг через переменные `Qcache_*` необходим для оценки эффективности. А для распределённых архитектур cache filter MaxScale часто является лучшим выбором.

Как и любой инструмент производительности, ключ — в измерениях. Активируйте, мониторьте, настраивайте. Если hit ratio остаётся ниже 20%, деактивируйте и переходите к другим вещам.

Эта статья была первоначально опубликована на Medium.