Оптимизация работы MySQL: кеширование запросов

Опубликовано muff в Втр, 2010-05-18 02:57



Занимаясь переносом хостинга, немало налопатил информации о СУБД MySQL. Заодно решил немного оптимизировать ее работу, включив поддержку кеширования.

После установки MySQL уже поддерживает механизм кеширования запросов, однако по умолчанию он выключен. Параметры по умолчанию следующие:

mysql> show variables like 'query%';		
+	+ Variable_name	Value
+	+ query_alloc_block_size	8192
query_cache_limit 10485	76 query_cache_min_res_unit	4096
query_cache_size	query_cache_type	ON
query_cache_wlock_invalidate OFF	query_prealloc_size	8192
++7 rows in set (0.00 sec)		

Размер кеша равен нулю. Для того, чтобы задать кеш размером 32 МБ, необходимо добавить следующую строку в **my.conf** (в секцию [mysqld]):

```
query cache size = 32M
```

Чтобы не перезапускать MySQL-сервер, изменим также и текущую конфигурацию, выполнив следующий запрос с правами суперпользователя:

```
mysql> set @ [at] global [dot] query_cache_size=32*1024*1024;
Query OK, 0 rows affected (0.00 sec)
```

Еще один параметр, на который следует обратить внимание - это **query_cache_limit** - он задает максимальный объем результата выполнения запроса, который может быть помещен в кэш.

Проверить состояние кеша можно следующим запросом:

```
      mysql> show global status like 'Qcache%';

      +-----+ Variable_name
      | Value | +-----

      -----+ Qcache_free_blocks
      | 21 | | Qcache_free_me

      mory | 30943456 || Qcache_hits | 3659 || Qcache_inserts

      | 2409 || Qcache_lowmem_prunes | 0 || Qcache_not_cached | 75

      | Qcache_queries_in_cache | 841 || Qcache_total_blocks | 1863 |+-

      ------+8 rows in set (0.00 sec)
```

Значение параметров:

- Qcache_free_memory объем свободной памяти, отведенной под кэш.
- Qcache hits количество запросов, отработанных из кэша.
- **Qcache inserts** количество вставок запросов в кэш.
- Qcache_lowmem_prunes количество высвобождений памяти из-за наполненности

кэша.

- **Qcache not cached** количество запросов, не подлежащих кэшированию.
- **Qcache_queries_in_cache** количество запросов, находящихся в кэше в настоящее время.

Рассчитать эфективность кеширования можно по следующей формуле: **Qcache_hits** / **(Qcache inserts + Qcache not cached)**.

Собственно пора задаться вопросом, как именно работает кеш. Все очень просто. При каждом запросе типа SELECT вычисляется хэш-сумма строки запроса и ищется в кэше. Если находится - возвращается рузельтат из кэша, если нет - выполняется запрос, а результат заносится в кэш (при условии, что результат не больше значения *query_cache_limit*). При каждом запросе типа UPDATE, REPLACE, INSERT, DELETE, TRUNCATE или ALTER, удаляются из кэша все запросы, использующие таблицу, подвергшуюся обновлению.

Стоит также отметить некоторые особенности кеширования, а именно:

- Различие запросов определяется буквально, сравнение чувствительно к реестру. Поэтому **SELECT** * **FROM news** и **select** * **FROM news** будут для кэша двумя разными запросами.
- В кэш всегда попадает результат выполнения запроса целиком, результаты выполнения подзапросов не кэшируются.
- Кэш работает одинаково для запросов к таблицам с различными механизмами хранения.
- Ряд запросов не подлежит кэшированию:
 - Запросы, содержащие одну из недетерминированных функций: NOW(), SLEEP(), RAND(), CURTIME(), LAST_INSERT_ID() и.др.
 - Запросы, использующие функции или хранимые процедуры, определенные пользователем.
 - Запросы, использующие значения локальных переменных.
 - Запросы, обращающиеся к базам данных mysql или INFORMATION SCHEMA.
 - Запросы типа SELECT ... FOR UPDATE, SELECT ... IN SHARE MODE, SELECT ... INTO OUTFILE, SELECT ... INTO DUMPFILE, SELECT * FROM ... WHERE autoincrement col IS NULL.
 - Запросы, использующие временные таблицы.
 - Запросы, не обращающиеся к таблицам.
 - Запросы, которые генерируют предупреждения (warnings).
 - В случае, если пользователь имеет права не на всю таблицу, а только на определенные колонки таблицы. Это исключение следствие того, что кэш запросов один для всех пользователей, а права доступа средствами кэша проверяются лишь на уровне таблиц.

Если необходимо, чтобы запрос не попадал в кеш, используется директива **SQL_NO_CACHE**, которая размещается сразу после оператора **SELECT**. Пример выполнения такого запроса:

mysql> SELECT SQL NO CACHE username FROM mail users;

Источник (получено 2025-11-23 14:09):

http://muff.kiev.ua/content/optimizatsiya-raboty-mysql-keshirovanie-zaprosov