Российские
интернет-технологии
2011

Поиск узких мест в производительности MySQL: ботанический определитель

Григорий Рубцов, SQLinfo.ru
rgbeast@sqlinfo.ru

SQL info.ru
Форум о MySQL

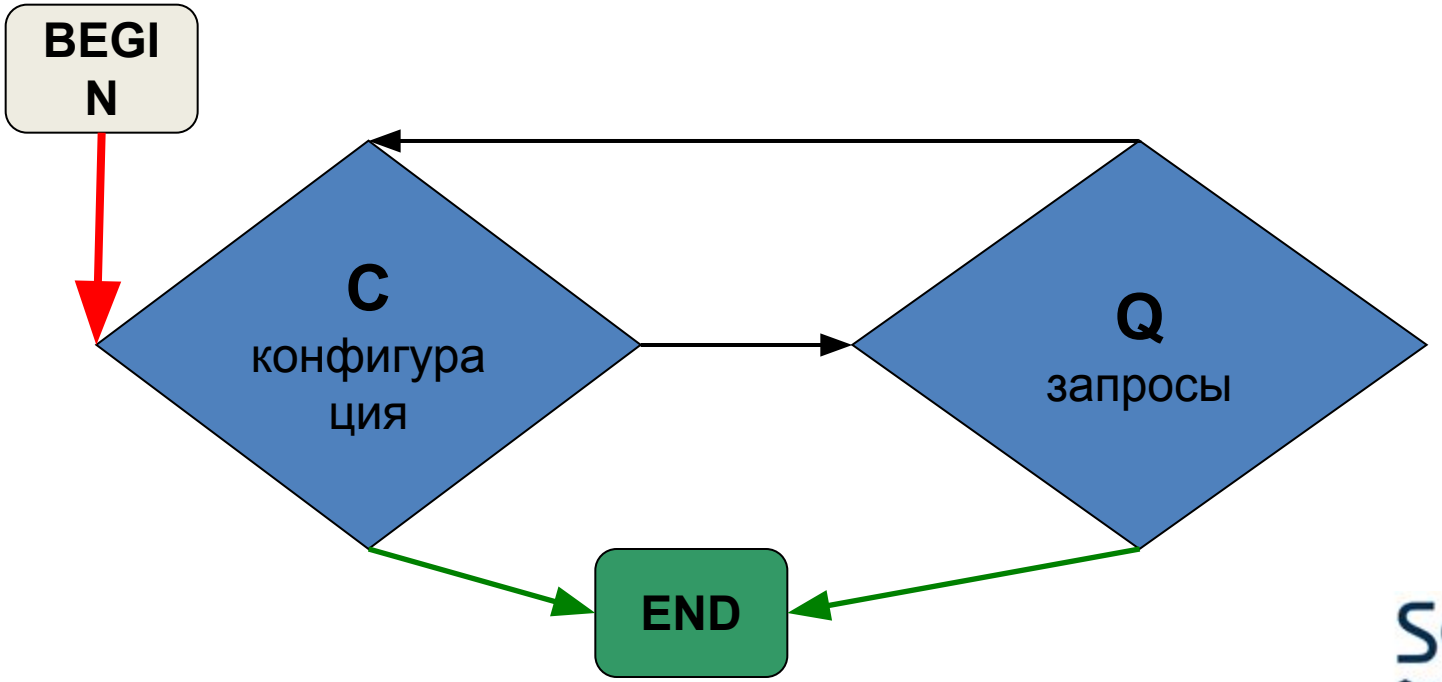
Поиск узких мест

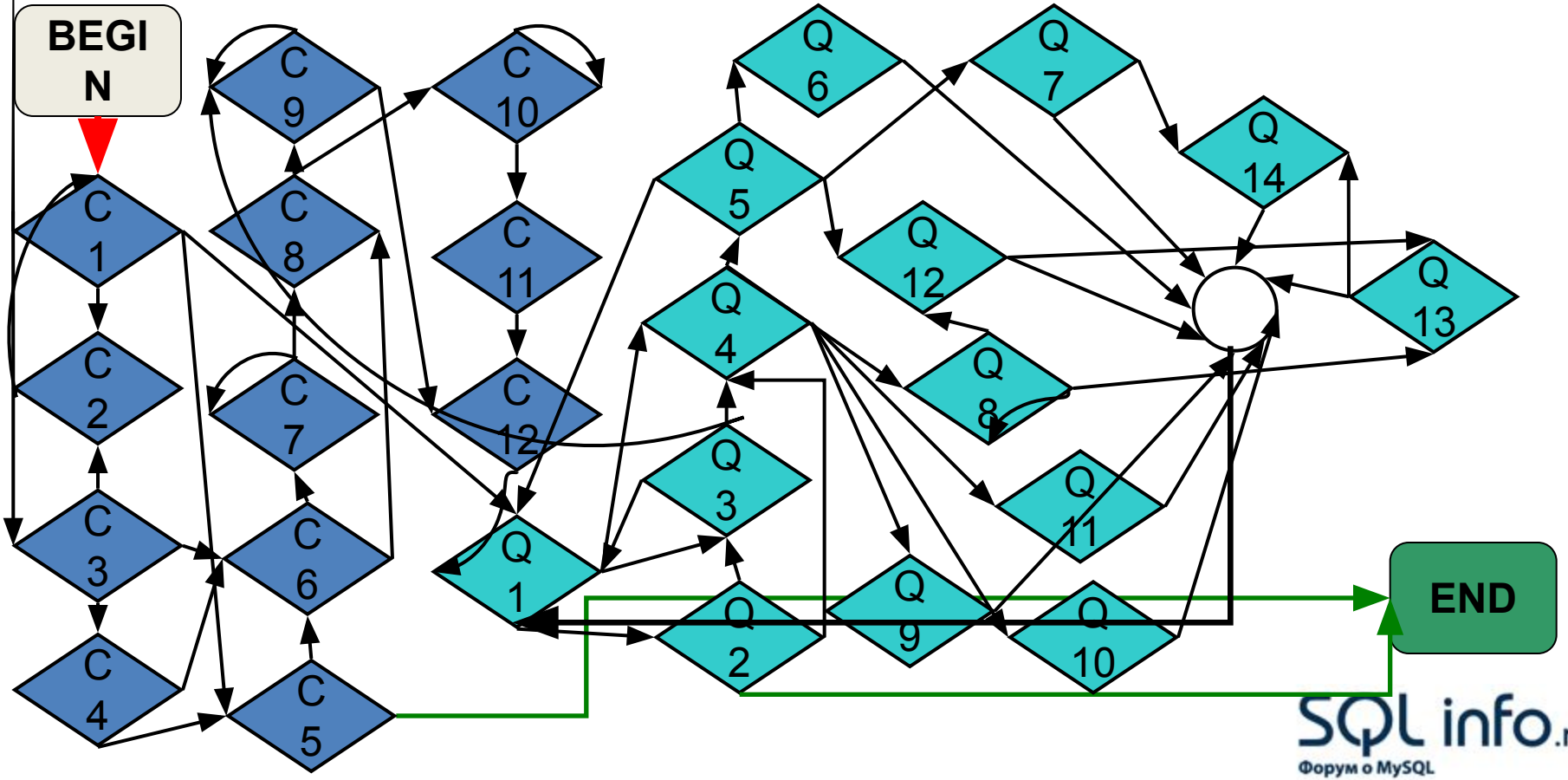
- Есть проблемы с производительностью?
- Если нет, создайте!
  - например, ускоренный повтор лога апача

```
$ cat access.log | awk '{print "GET
 http://localhost" $7 ";sleep 0.1"}' | bash
```

- Теперь можно классифицировать

Структура определителя





SQL info.ru
Форум о MySQL

C1 - начало

```
$ top
```

Кто съедает процессор?

- httpd/php/… => C2
- mysqld => Q1
- треды ядра => C5
- загрузка менее 20% => C3

C2 - апач ест ресурсы

- Заменить апач на lighttpd/nginx + fastcgi

  или

- Поменять настройки тредов апача

  или

- Избавиться от лишних циклов в скриптах

  или

- Настроить persistent соединения

  => C1

С3 – используется ли своп?

Используется ли своп?

- да, память занята httpd/php/… => С2
- да, память занята mysqld => С6
- нет, часть памяти свободна (free, buffers, cached) => С4

C4 – нагрузка диска

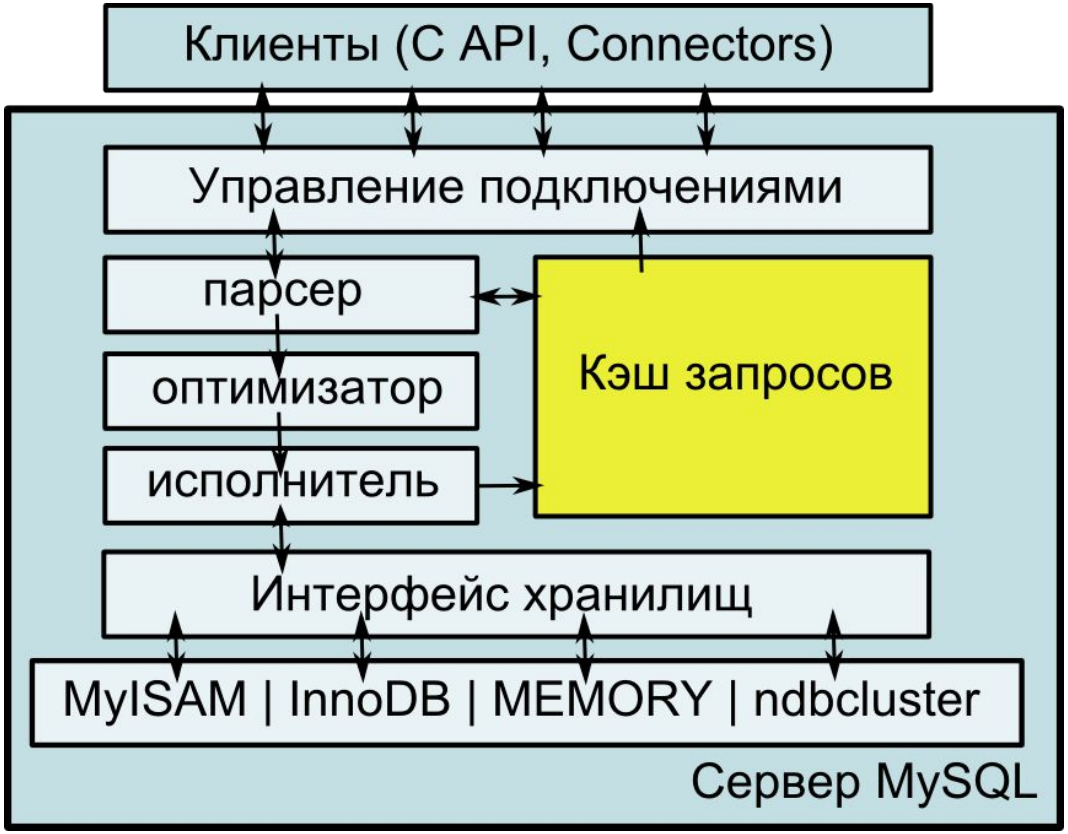
```
$ iostat -x 1
```

- процент использования большой по некоторым дисками или неравномерен для дисков одного soft RAID => C5
- процент использования небольшой по всем дискам => C6

C5 – проверка железа

- проверьте диски (`smartctl`; скорость прямой записи, чтения, свободное место на дисках; статус RAID)
- проверить /var/log/messages на предмет ошибок диска, памяти (см. также `mcelog`) и другого железа
- Неисправное железо найдено и заменено => END;
- Неисправность не обнаружена => C6

C6


Клиенты (C API, Connectors)
Управление подключениями
парсер
оптимизатор
исполнитель
Кэш запросов
Интерфейс хранилищ
MyISAM | InnoDB | MEMORY | ndbcluster
Сервер MySQL

C6 – настройка query cache

```
mysql> SHOW VARIABLES LIKE 'query_cache%';
```

- Если есть сложные запросы (например, JOIN или WHERE не по индексу) или не более двух процессорных ядер:
  - включить query_cache => C7
- Если запросы неповторяющиеся или все быстрые при многоядерном процессоре:
  - подумать про HandlerSocket; выключить query_cache => C8

см. доклад Константина Осипова на Highload++ 2008
http://www.highload.ru/papers2008/7650.html

C7 – настройка размера query cache

- Дождаться разогрева кэша

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

- Если Qcache_free_memory << query_cache_size => увеличить размер кэша, повторить C7
- Если Qcache_free_memory > 0.5 query_cache_size => уменьшить размер кэша

=>C8

http://webew.ru/articles/1041.webew

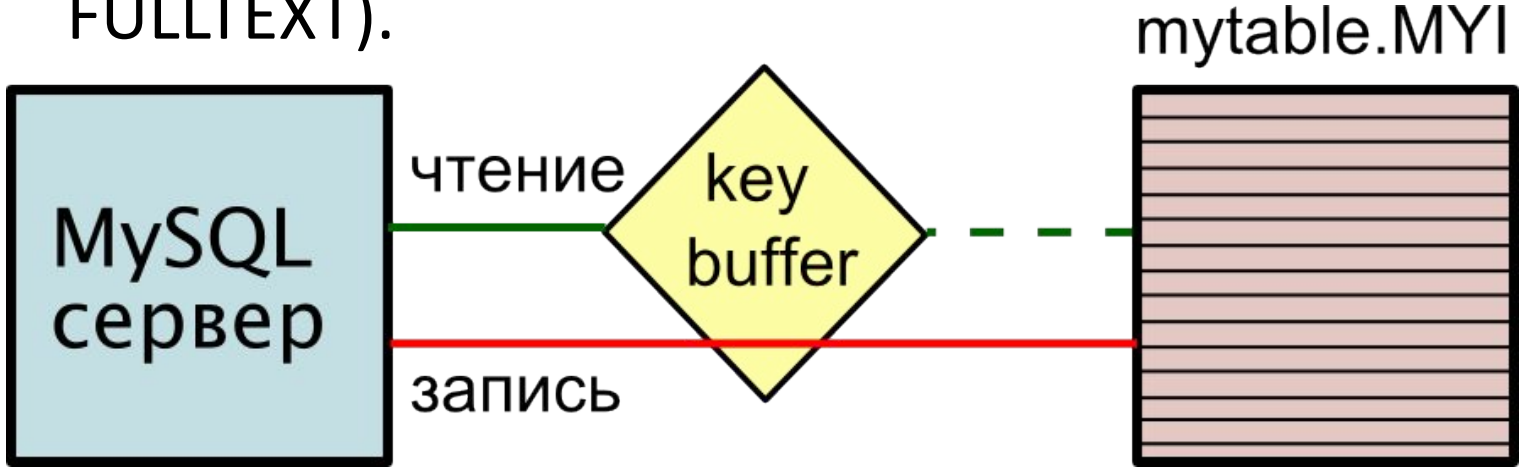
C8 – тип хранилища

```
mysql> SELECT ENGINE,TABLE_SCHEMA FROM
   INFORMATION_SCHEMA.TABLES GROUP BY
   ENGINE,TABLE_SCHEMA;
```

- Какой тип основных таблиц?
  - MyISAM => C9
  - InnoDB => C10
  - Часть MyISAM, часть Innodb => C9,C10

MyISAM key buffer

- MyISAM кэширует только индексы (вкл. FULLTEXT).



C9 – MyISAM key buffer

- MyISAM кэширует только индексы (вкл. FULLTEXT).

```
mysql> SHOW GLOBAL STATUS LIKE 'key_%';
```

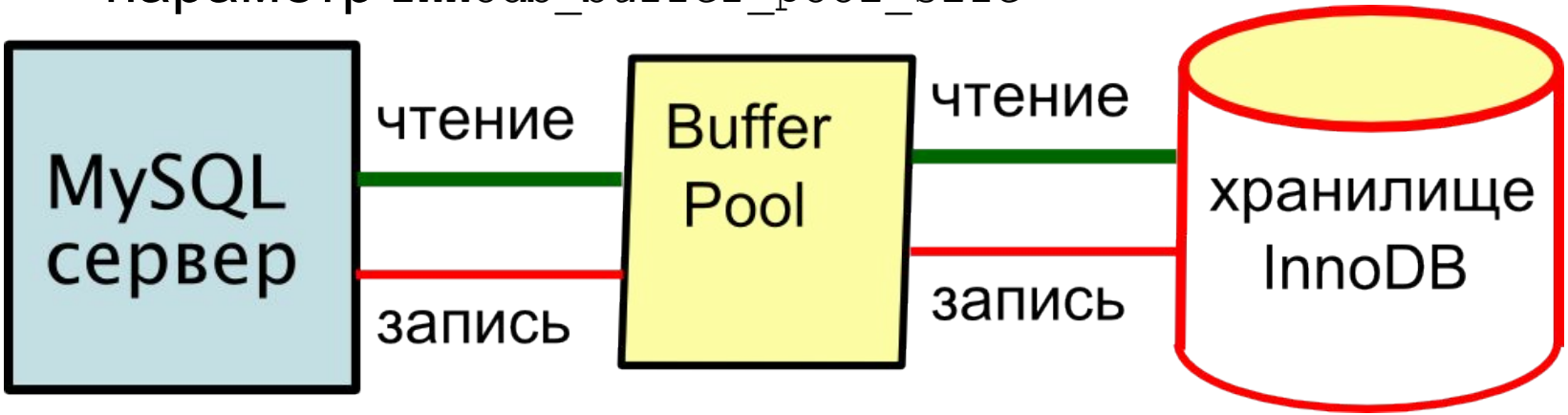
- Key_blocks_unused << key_blocks_used => увеличить key_buffer_size, повторить C9

- Key_blocks_unused > key_blocks_used => уменьшить key_buffer_size

  => C12

Буфер InnoDB

- InnoDB кэширует индексы и данные, размер буфера - параметр `innodb_buffer_pool_size`



C10 – InnoDB buffer pool

- Дождаться разогрева кэша

```
mysql> SHOW GLOBAL STATUS LIKE 'innodb_buf%';
```

- `Innodb_buffer_pool_pages_free` << `Innodb_buffer_pool_pages_total`

увеличить `innodb_buffer_pool_size`; повторить C10 (вплоть до трети от общей памяти)

=> C11

C11 – InnoDB options

- Если `innodb_file_per_table=OFF`
  - включить `innodb_file_per_table`
  - **ALTER TABLE `tbl` ENGINE=InnoDB**; для таблиц InnoDB
- `innodb_flush_log_at_trx_commit`:
  - 0 запись и flush лога раз в секунду
  - 1 (default) – flush лога после каждой транзакции
  - 2 запись после каждой транзакции, flush - раз в секунду
- установите значение 0 или 2, если не страшно потерять секунду данных

=> C12

C12 – файлы и треды

```
mysql> SHOW GLOBAL STATUS LIKE 'open_%';
```

- Если `opened_tables` >> `open_tables`, увеличить `table_cache`
- Если таблиц очень много, проверить ограничение ОС (в Linux: fs.file-max через sysctl)

```
mysql> SHOW GLOBAL STATUS LIKE 'threads_%';
```

- Если `threads_created` >> `threads_connected`, увеличить `thread_cache_size`

=> Q1

Q1 – поиск худшего запроса

- При пиковой нагрузке выполнить несколько раз подряд:

```
mysql> SHOW FULL PROCESSLIST;
```

- Найден запрос I рода: часто появляется в процессах, остальные запросы при этом выполняются быстро => Q4
- Найден запрос II рода: во время его выполнения другие запросы ожидают в состоянии Locked => Q3
- Не найдено медленных запросов => Q2

SQL info.ru
Форум о MySQL

Q2 – поиск худшего запроса

- Включить журнал медленных запросов (`log_slow_queries; long_query_time=1;`)
- Накопить статистику, включающую пиковое время.
- Сгенерировать дайджест: `mk-query-digest` из Percona Maatkit
- Проверить скорость исполнения 10 самых медленных SELECT-запросов (с опцией SQL_NO_CACHE).
  - Все они выполняются действительно медленно => это запросы первого рода => Q4
  - Некоторые из них на самом деле быстрые и выполнялись медленно из-за блокировки => среди тех, которые выполняются медленно есть запросы второго рода => Q3
- Не найдено медленных запросов => END;

Q3 – запрос II рода

- SELECT блокирует другие SELECT только если между ними в очереди UPDATE
- Если есть запросы типа апдейта статистики
  - UPDATE `articles` SET viewcount=viewcount++;

  перенести их в отдельную таблицу. Они не медленные, но допускают блокировку => Q1
- Если таблицы типа MyISAM и не используется полнотекстовый индекс, рассмотрите возможность перехода на InnoDB => C9
- default: оптимизировать так же, как запросы I рода => Q4

Q4 – оптимизация запроса

- Запрос содержит ORDER BY + LIMIT => Q9
- Запрос содержит подзапросы:
  - независимые => Q10
  - зависимые => Q11
- default:
  - содержит JOIN => Q8
  - не содержит JOIN => Q5

Q5 – простой запрос

- В запросе нет WHERE
  - это ошибка => исправьте запрос => Q1
  - это не ошибка => Q12
- Выполните `EXPLAIN` *ЗАПРОС*
- Нет подходящего индекса => Q6
- Индекс есть, но не используется => Q7
- Индекс есть и используется => Q13

Q6 – создание индекса

- Составной индекс используется только последовательно без пропусков
- ALTER TABLE `tbl` ADD KEY(A,B,C)
  - позволяет:
    - WHERE A=10 AND B>10;
    - WHERE A=10 AND B=7 AND C=12;
    - WHERE A=10 AND B=7 ORDER BY C;
  - не позволяет
    - WHERE B=10;
    - WHERE A=10 ORDER BY C;
    - WHERE A=10 AND B>10 AND C>10;
      - Операция сравнения последняя при использовании индекса

=> Q1

Q7 – индекс не используется

- Удалите избыточные индексы. Например из KEY(A,B), KEY(A), KEY(B) можно оставить KEY(A,B) и KEY(B). Лучше не иметь избыточности, чем использовать USE INDEX.
- Таблица слишком маленькая – оптимизатор предпочитает полный скан таблицы (наполнить данными или использовать FORCE INDEX)
- Бывает, что индексы отключили и забыли включить. Тест:

```
SELECT TABLE_SCHEMA,TABLE_NAME, GROUP_CONCAT(comment) FROM
    INFORMATION_SCHEMA.STATISTICS WHERE comment LIKE '%disabled%' GROUP BY 1,2;
```

Включить: ALTER TABLE `mytable` ENABLE KEYS;

Проблема решилась => Q1, не решилась => Q14

Q8 – составной запрос

- Выполните `EXPLAIN` *ЗАПРОС*
- Не все JOIN выполняются по индексу => создайте необходимые индексы (подключение каждой таблицы рассматривать как простой запрос), Q6
- Много записей на выходе и так и нужно => Q12
- Порядок JOIN неоптимальный => используйте SELECT STRAIGHT_JOIN => Q8
- Все объединения используют индексы => Q13

Q9 – ORDER BY+LIMIT (1/4)

- Самый медленный класс запросов в рунете. Пример из практики:

```
SELECT * FROM wp_posts
LEFT JOIN wp_post2cat
   ON (wp_posts.ID = wp_post2cat.post_id)
LEFT JOIN wp_categories
   ON (wp_post2cat.category_id = wp_categories.cat_ID)
WHERE (category_id = '1')
AND post_date_gmt <= '2008-09-12 00:15:59'
AND (post_status = 'publish')
AND post_status != 'attachment' GROUP BY wp_posts.ID
ORDER BY post_date DESC LIMIT 3, 3;
```

Q9 – ORDER BY+LIMIT (2/4)

- EXPLAIN SELECT …

```
           id: 1
  select_type: SIMPLE
        table: wp_posts
         type: ref
possible_keys: PRIMARY,post_status,post_date_gmt
          key: post_status
      key_len: 1
          ref: const
         rows: 3450
        Extra: Using where; Using temporary; Using filesort
.....................
```

Q9 – ORDER BY+LIMIT (3/4)

- Как выполняется запрос?
  - Используется индекс **post_status='publish'**
  - Перебор почти всей таблицы для проверки остальных условий
  - Временная таблица (в памяти, если меньше **tmp_table_size**), содержащая все поля таблиц, участвующих в JOIN
  - Сортировка всей временной таблицы (filesort, без использования индексов)
  - LIMIT 3,3 – оставляем записи с 4 по 6

Q9 – ORDER BY+LIMIT (4/4)

- Решение: разбить запрос
  - Сортировать только значения id
  - Наложить ограничение LIMIT
  - Получить значения остальных полей вторым запросом
- Схематическое решение:

```
SELECT * FROM large_table WHERE id IN
(SELECT id FROM large_table WHERE условие
AND условие AND условие ORDER BY порядок LIMIT M,N) ORDER BY порядок;
```

- ☹ IN + LIMIT будет только в MySQL 6.0
- На практике два последовательных запроса, => Q1

Q10 – независимые подзапросы

- Обычно в контексте WHERE или FROM. Пример:
  - `SELECT name FROM clients WHERE age < (SELECT c1.age FROM clients c1 WHERE c1.name='Paul');`
- Оптимизатор MySQL может ошибочно принять независимый подзапрос за зависимый и выполнять для каждой строки. В этом случае нужно разбить запрос на два. => Q1

Q11 – зависимые подзапросы

- Обычно в контексте SELECT или WHERE. Пример:
  - `SELECT clients.id, (SELECT count(*) FROM contracts WHERE contracts.client=clients.id) FROM clients;`
- Оптимизатор MySQL имеет больше возможностей оптимизации JOIN, чем подзапросов. Если возможно, преобразуйте в JOIN. Получилось ускорить => Q1, не получилось => Q13.

Q12 - выборка большого объема

- Используйте NOT NULL – сокращает объем данных и размер индексов.
- Используйте оптимальные типы: TINYINT, SMALLINT, FLOAT
- Используйте однобайтовую кодировку для хранения и при подключении.
- Вынесите в другую таблицу данные, которые не участвуют в выборке.
- Избавьтесь от лишних индексов.
- Создайте MEMORY-таблицу с данными для выборки.
- Получилось ускорить => Q1, не получилось => Q13.

Q13 – изменение логики

- Обдумайте запрос и перепишите его полностью. Может потребоваться:
  - изменение логики приложения
  - изменение структуры таблиц (денормализация)
  - создание временных таблиц, MEMORY-таблиц, и др.
  - явная сортировка таблицы MyISAM: ALTER TABLE `tbl` ORDER BY id;
  - использование пользовательских переменных. Простейший пример:
    - SET @i=NULL; SELECT id-@i AS gap, @i:=id FROM `table` ORDER BY id;

- Получилось ускорить => Q1, не получилось => Q14.

Q14 – задайте вопрос на SQLinfo.ru

- Задайте вопрос на форуме http://sqlinfo.ru/forum/
- дождитесь ответа
- => Q1

END;

Число экземпляров в наших музеях абсолютно ничтожно по сравнению с несметными поколениями видов, несомненно существовавших.

Чарльз Дарвин, «Происхождение видов»