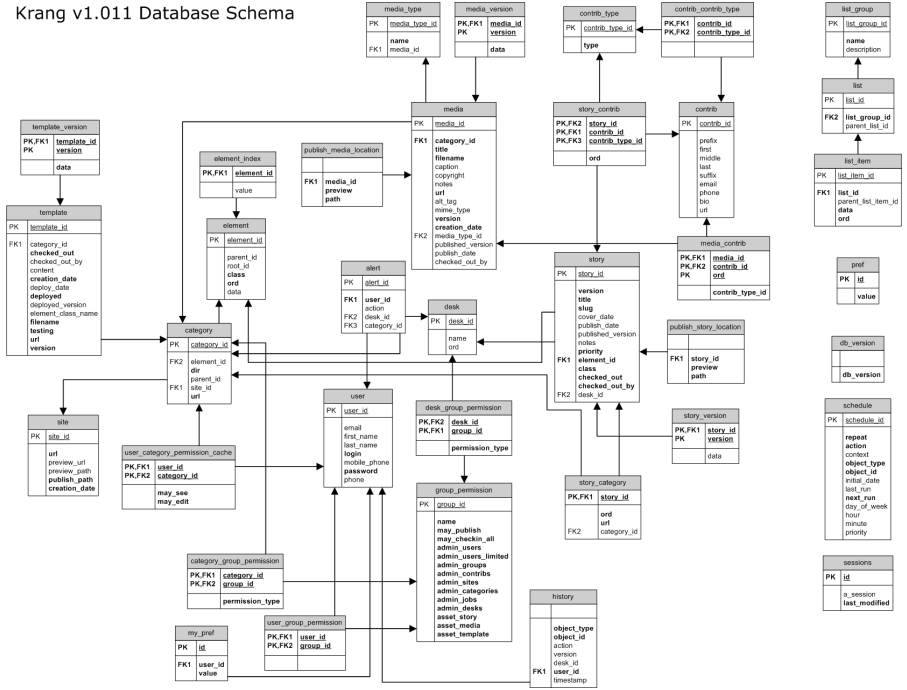
Сергей Аверин

То, что вы хотели знать о HandlerSocket, но не смогли нагуглить

Российские интернет-технологии



SQL — это, конечно, круто

Российские интернет-технологии

```
SELECT associations2.object_id, associations2.term_id, associations2.cat_ID, associations2.term_taxonomy_id
FROM (SELECT objects_tags.object_id, objects_tags.term_id, wp_cb_tags2cats.cat_ID, categories.term_taxonomy_id
    FROM (SELECT wp_term_relationships.object_id, wp_term_taxonomy.term_id, wp_term_taxonomy.term_taxonomy_id
        FROM wp_term_relationships
        LEFT JOIN wp_term_taxonomy ON wp_term_relationships.term_taxonomy_id = wp_term_taxonomy.term_taxonomy_id
        ORDER BY object_id ASC, term_id ASC)
        AS objects_tags
    LEFT JOIN wp_cb_tags2cats ON objects_tags.term_id = wp_cb_tags2cats.tag_ID
    LEFT JOIN (SELECT wp_term_relationships.object_id, wp_term_taxonomy.term_id as cat_ID, wp_term_taxonomy.term_taxonomy_id
        FROM wp_term_relationships
        LEFT JOIN wp_term_taxonomy ON wp_term_relationships.term_taxonomy_id = wp_term_taxonomy.term_taxonomy_id
        WHERE wp_term_taxonomy.taxonomy = 'category'
        GROUP BY object_id, cat_ID, term_taxonomy_id
        ORDER BY object_id, cat_ID, term_taxonomy_id)
        AS categories on wp_cb_tags2cats.cat_ID = categories.term_id
    WHERE objects_tags.term_id = wp_cb_tags2cats.tag_ID
    GROUP BY object_id, term_id, cat_ID, term_taxonomy_id
    ORDER BY object_id ASC, term_id ASC, cat_ID ASC)
    AS associations2
LEFT JOIN categories ON associations2.object_id = categories.object_id
WHERE associations2.cat_ID <> categories.cat_ID
GROUP BY object_id, term_id, cat_ID, term_taxonomy_id
ORDER BY object_id, term_id, cat_ID, term_taxonomy_id
```

И очень сложно

- Групповые функции
- Подзапросы
- JOIN’ы
- Навороченные WHERE-условия
- Транзакции и блокировки
- и т. д.

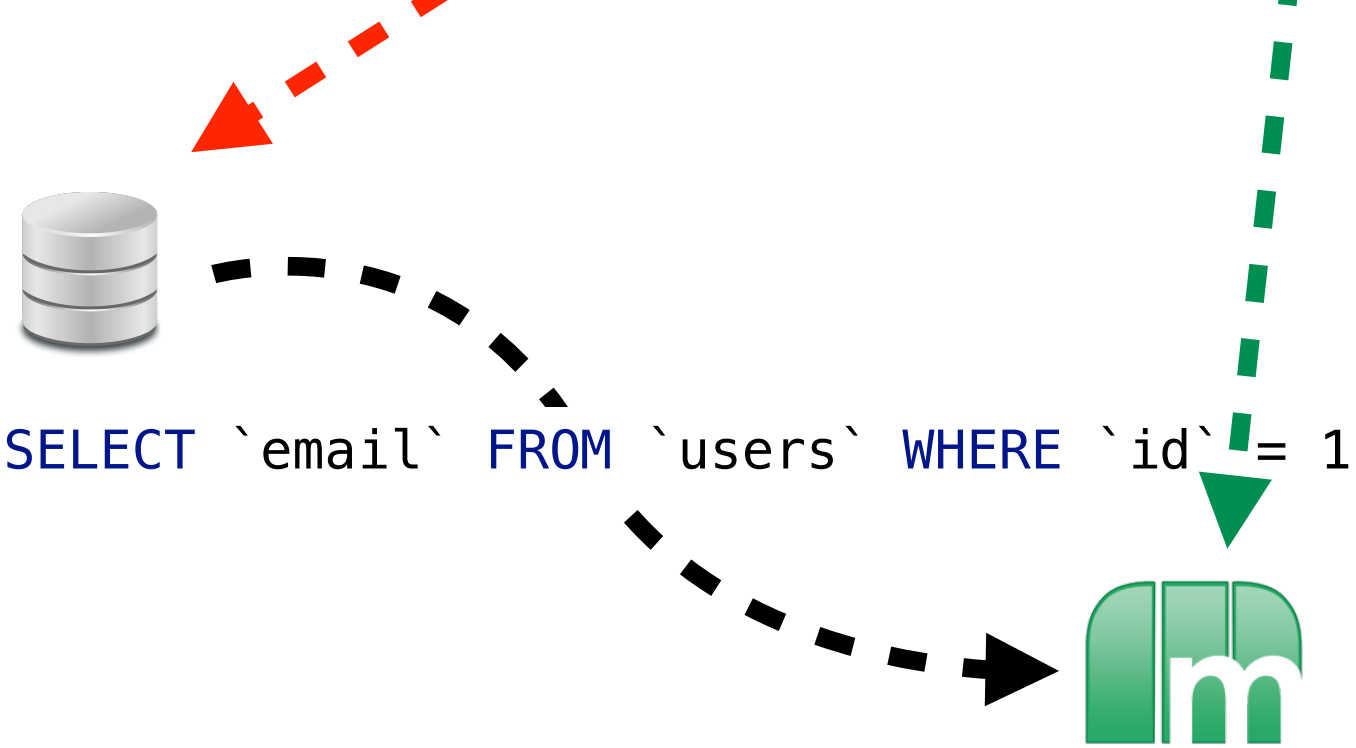
С кучей прибамбасов

Но большинство запросов простые

Российские интернет-технологии

```
SELECT `email` FROM `users` WHERE `id` = 1
```

Примерно такие



Примерно так

Российские интернет-технологии

```
SELECT `email` FROM `users` WHERE `id` = 1
```

Но можно проще

Российские интернет-технологии

Что такое Handlersocket

- MySQL plugin низкоуровневого доступа к InnoDB/XtraDB
- Открывает отдельные порты
- Имеет свой протокол и набор команд

SQL не поддерживает

Совсем

Плюсы

- Скорость
- Пакетная обработка операций
- Компактный протокол
- Выдерживает 10000+ соединений
- Не отменяет обычный SQL
- Совместим с репликацией
- Из коробки идет с Percona Server

Поэтому

не нужен memcache → больше свободной памяти

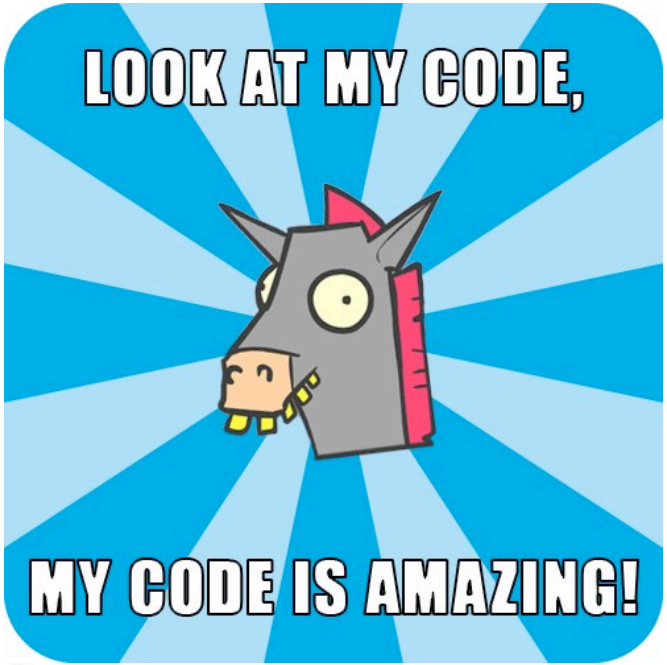
нет дублирования данных → теперь данные консистентны

Плюсы

- Скорость
- Пакетная обработка операций
- Компактный протокол
- Выдерживает 10000+ соединений
- Не отменяет обычный SQL
- Совместим с репликацией
- Из коробки идет с Percona Server

Минусы

- Глючит с не-InnoDB/XtraDB хранилищами
- Нет транзакций, хранимых процедур
- Некоторый базовый функционал MySQL не поддерживается
- Нет коммерческой поддержки
- Немного незрелый продукт
- Конфликтует с DDL-командами и LOCK TABLES



Иногда глючит

и еще...

- не очень внятная документация
- логика работы и протокол иногда меняются без всякого уведомления

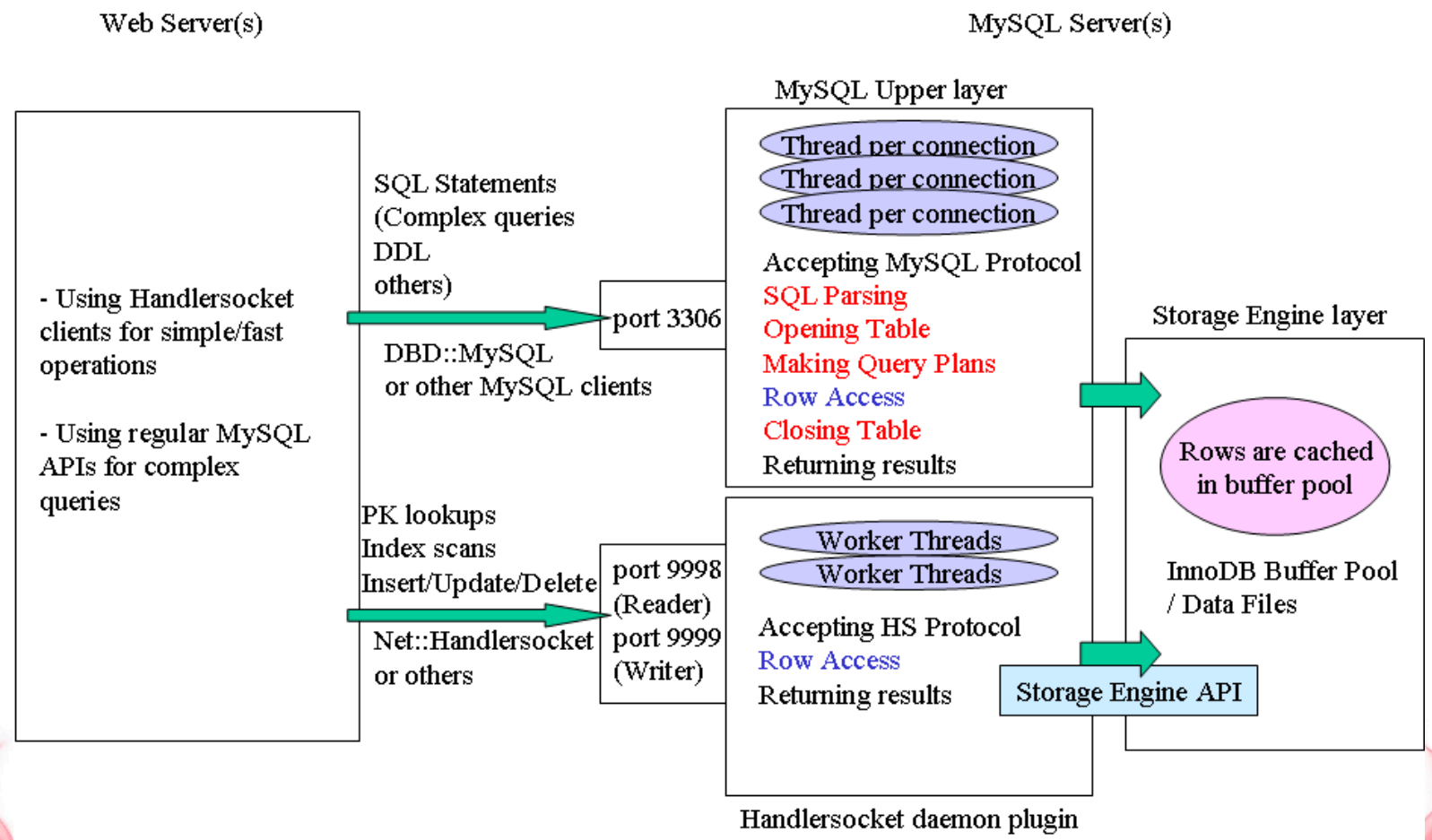
Российские интернет-технологии

Чем Handlersocket не является

- Не хранилище «ключ–значение»
- Не интерфейс бинарного SQL-протокола
- Не для сложных запросов
- Не для создания/изменения таблиц
- Не для хостинга (нет доступа с разграничением прав)

Принцип работы

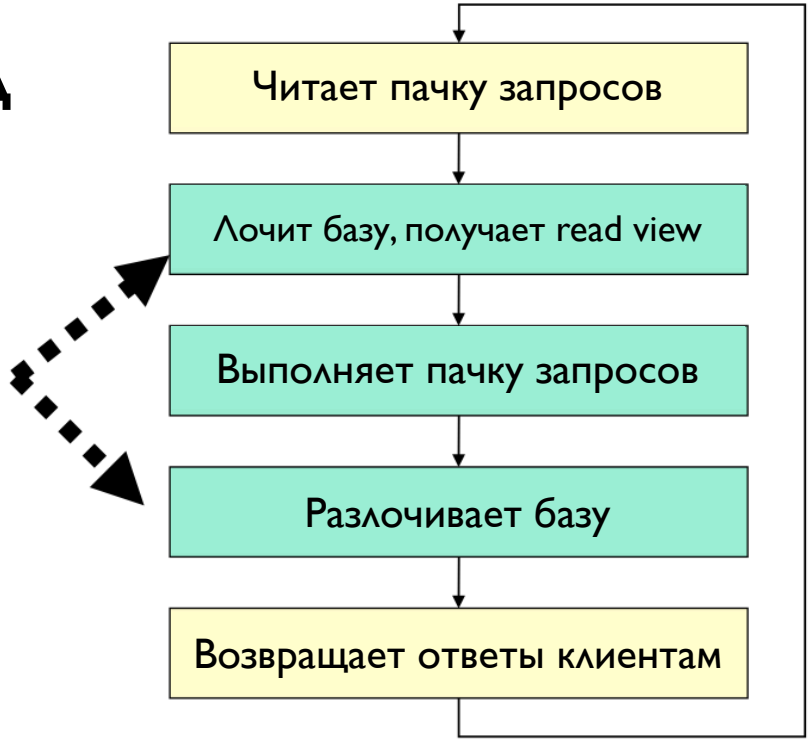
Российские интернет-технологии



Читающий тред

locks/unlocks

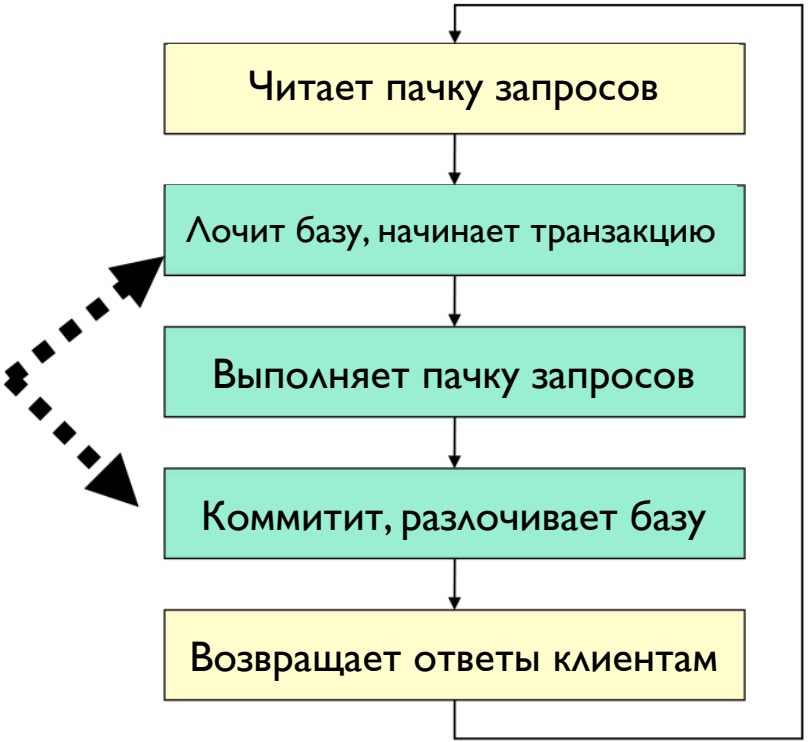
1 раз на несколько запросов



Пишущий тред

locks/unlocks

1 раз на несколько запросов



Взаимодействие HS и MySQL

- Консистентность соблюдается при доступе через и SQL и HS
- HS прекрасно работает с репликацией MySQL
- auto_increment поддерживается
- Современные версии HS инвалидируют query cache
- Система прав и пользователей MySQL в HS не поддерживается
- Блокировка таблиц через HS- и SQL-доступ конфликтует

Специально для рядового Кучи

Будьте осторожны с:

- ‘LOCK TABLES ... WRITE’
- ‘ALTER TABLE ...’

XtraBackup тоже не будет работать.

Российские интернет-технологии

Подсмотрено в интернетах

Кстати, это plugin, поэтому должно работать:

```
install plugin handlersocket soname
'handlersocket.so';

uninstall plugin handlersocket;
```

На практике вторая команда обычно вешает базу.

Как его «готовить»

Очень длинная часть...

Установили, сконфигурировали, подключили...

Теперь у вас есть  новых открытых порта: 9998, 9999

только для чтения

Российские интернет-технологии

Протокол

- Клиент открывает соединение
- Клиент посылает запрос
- Сервер посылает ответ
- Клиент посылает следующий запрос

...

(1 запрос — 1 ответ)

Можно послать N запросов подряд, придет N ответов в том же порядке.

Российские интернет-технологии

Протокол

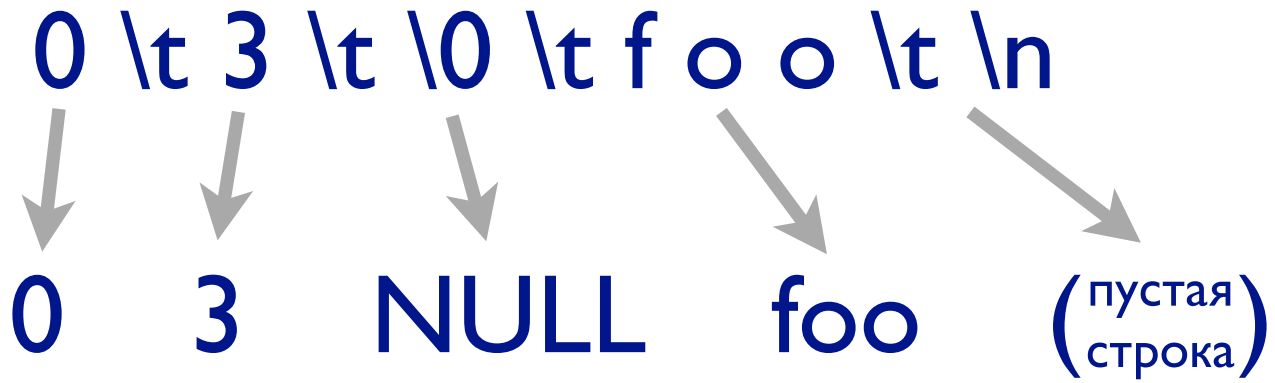
- Бинарный, но похож на текстовый. Telnet — наше все.
- Один запрос или ответ — одна строка.
- Каждая строка оканчивается \n (0x0A).
- Каждая строка состоит из набора токенов разделенных \t (0x09).
- Токен — это или NULL или кодированная строка.
- NULL кодируется как \0 (0x00).

Строки

- Пустая строка — это токен нулевой длины
- Каждый байт в диапазоне 0x00–0x0F предваряется 0x01 и сдвигается на 0x40. (Пример: 0x03 ➔ 0x01 0x43)
- Остальные байты не меняются

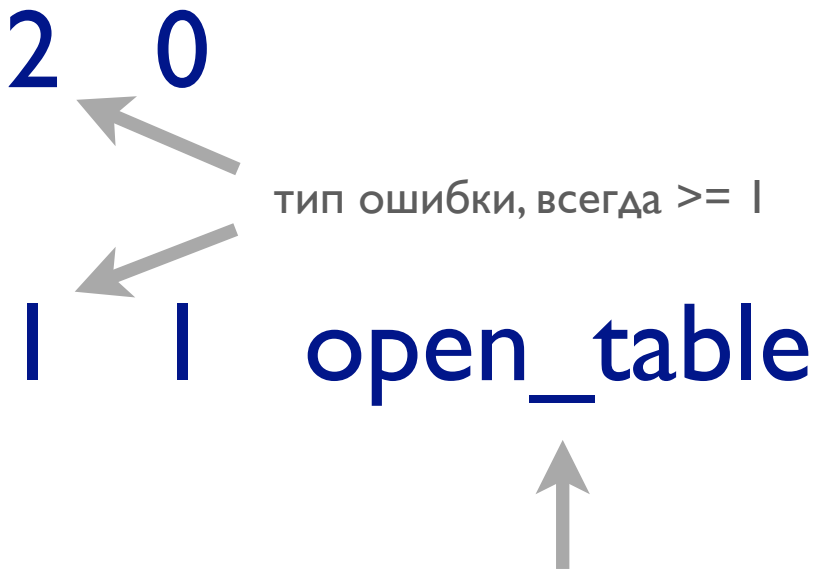
\t\t или \t\n означает, что между ними есть пустая строка

Пример команды:



Российские интернет-технологии

Ошибки:



Российские интернет-технологии

Команды

Российские интернет-технологии

Открытие индекса

P <index_id> <db> <table> <index> <columns> [<fcolumns>]

- <index_id>: любое целое число
- <db>,<table>,<index>: Имена базы, таблицы и индекса. Чтобы открыть первичный ключ используйте имя ключа PRIMARY.
- <columns>: разделенный запятыми список столбцов, с которыми вы будете работать
- <fcolumns>*: разделенный запятыми список столбцов, которые вы будете использовать для фильтрации

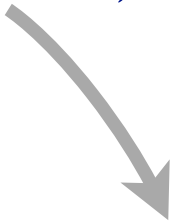
* — опционально

Открытие индекса

P <index_id> <db> <table> <index> <columns> [<fcolumns>]

P 1 test store PRIMARY id,box fruit

что-то типа prepared statement



SELECT id,box FROM test.store WHERE id=? AND fruit=?

Открытие индекса

P <index_id> <db> <table> <index> <columns> [<fcolumns>]

- Можно переоткрыть индекс под тем же <index_id> и возможно другими <db>/<table>/<index>.
- Можно открывать ту же самую комбинацию <db>, <table>, <index> несколько раз, и даже с разными <columns>.
- Команды «закрыть индекс» нет. Индексы закрываются с прекращением соединения.
- Много индексов жрет память и тормозит работу. Старайтесь обойтись < 1000 открытых индексов.
- Для скорости <index_id> должны быть как можно меньше.

Вставка

<index_id> + <vlen> <v1> ... <vn>

- <index_id>: номер открытого индекса
- <vlen>: количество <v1> ... <vn>. Должно быть <= кол-ва <columns> в открытом индексе.
- <v1> ... <vn>: данные для вставки в порядке <columns>. Остальные поля получают значения по умолчанию.

Крайне рекомендуется давать данные для всех полей из <columns>. (подробнее позже)

Вставка

Пример:

```
P   89   test   hs4   PRIMARY   warehouse,box,fruit,count
0   1

89   +   4   NewYork   A1   melon   4
0   1   1   ← last_insert_id

89   +   4   NewYork   A2   melon   4
0   1   2   ← last_insert_id
```

Российские интернет-технологии

Выборка

сломано ;-(

<index_id> <op> <vlen> <v1> ... <vn> [LIM] [IN] [FILTER]

- <index_id>: номер открытого индекса
- <op>: оператор — один из =, <, <=, >, >=
- <vlen>: количество <v1> ... <vn>. Должно быть <= кол-ва <columns> в открытом индексе.
- <v1> ... <vn>: значения, которые нужно искать в <index>.
- LIM*: выражение OFFSET-LIMIT
- IN*: выражение IN
- FILTER*,**: выражение FILTER

* — опционально

** — может повторяться

Выборка

Пример:

Выборка одного ряда по id = 3 (одноколоночный индекс)

```
P   89   test   hs2   PRIMARY   warehouse,box,fruit,count
0   1

89     =     1     3
0     4   Virginia     A1     grapes     5
```

кол-во <columns>

Российские интернет-технологии

Выборка

сломано ;-(

<index_id> <op> <vlen> <v1> ... <vn> [LIM] [IN] [FILTER]

Выражение LIM:

<limit> <offset>

- Та же логика, что и в SQL(только здесь <limit> должен включать кол-во пропускаемых рядов)
- Если в запросе нет этого выражения, подразумевается <limit> = 1 и <offset> = 0.
- Накладывается после применения FILTER.

Выборка

Пример:

Выборка 3 рядов начиная с id 2 (одноколоночный индекс)

```
P   89   test   hs2   PRIMARY   warehouse,box,fruit,count
0   1
                                  LIM
89     >=     1     2     3     0
0     4   Seattle    B1    banana    4   Virginia    A1    grapes    5
Virginia    B2    watermelon    1
```

! обратите внимание: кол-во колонок — 4, а не 12

Выборка

сломано ;-(

<index_id> <op> <vlen> <v1> ... <vn> [LIM] [IN] [FILTER]

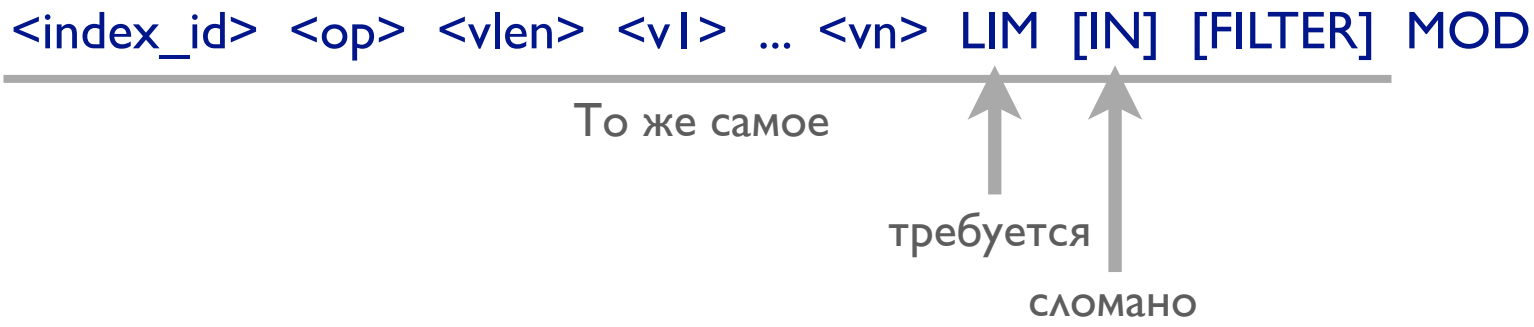
Выражение FILTER:

<ftyp> <fop> <fcol> <fval>

- <ftyp>: F (пропустить неподходящие ряды) или W (завершить на первом неподходящем ряду)
- <op>: операция, одна из =, !=, <, <=, >, >=
- <fcol>: номер колонки из <fcolumns> (начиная с нуля) в открытом индексе
- <fval>: значение

Если указано несколько фильтров, они работают через логическое «И».

Изменение/удаление



Изменение/удаление

<index_id> <op> <vlen> <v1> ... <vn> LIM [IN] [FILTER] MOD

Выражение MOD:

<mop> <m1> ... <mn>

- <mop>: Операция: U, U? (изменение), D, D? (удаление), +, +? (инкремент), −, −? (декремент). Операции с '?' возвращают значения до изменения.
- <m1> ... <mn>: значения полей в порядке <columns>. Должны быть <= кол-ва <columns> в открытом индексе. Остальные колонки не изменяются. Должны быть числами для '+', '−'. Не используются для 'D', 'D?'.

Изменение/удаление

Пример:

Выборка count по id = 8 с увеличением count на 10

```
P    90    test    hsmdemo3    PRIMARY    count
0    1
```

```
                                   LIM        MOD
90   =    1    8    1    0    +?    10
0    1    6
```

count до изменения

Российские интернет-технологии

Изменение/удаление

Пример:

Удаление рядов с id > 0 и count > 3

P 89 test hsmdemo3 PRIMARY count count

0 1

LIM FILTER MOD

89 > 1 0 1000 0 F > 0 3 D

0 1 5

кол-во удаленных рядов

Российские интернет-технологии

SQL – HS аналогии

- SELECT a,b,c FROM ...
- ... LIMIT 1 OFFSET 0
- id BETWEEN 1 AND 2
- WHERE a < 1 AND b > 2
- a IN (...)
- SELECT ... FOR UPDATE, UPDATE

- Открытие индекса с <columns>=a,b,c
- Выражение LIM
- Выборка по индексу >= 1 + FILTER W-типа id <= 2
- FILTER типа F a<1 + FILTER типа F b>2
- IN выражение (глючит)
- Изменение с операторами с '?'

Российские интернет-технологии

Было сложно, да?

Российские интернет-технологии

Особенности

Российские интернет-технологии

Поддерживаемые типы данных

- Любые типы данных MySQL нормально читаемы через HS
- Писать можно все, кроме типа TIMESTAMP
- Не умещающиеся по длине данные обрезаются так же, как через SQL
- ON UPDATE CURRENT_TIMESTAMP не поддерживается

Кодировки

- Если вы работаете только с UTF8 — все просто. Просто соблюдайте стандарт кодирования протокола HS.
- BLOB-поля — бинарные, читаем то, что писали, никаких кодировок.
- Поля с кодировками HS пишет и читает в кодировке столбца. То есть о кодировках он ничего не знает, строка — это набор байт.
- Тем не менее, байты не соответсвующие кодировке, меняются на символ ‘?’ при вставке.

Сортировка

- Collation'ы столбца влияют на операции >, >=, <, <= (но не на фильтры) и порядок, в котором вы получаете ряды ответов.
- HS при выборке читает ряды в порядке их хранения в индексе
- А в MySQL индексы хранятся сортированными согласно collation'ам столбцов из которых они состоят
- См. http://www.collation-charts.org/mysql60/

Значения по умолчанию

При вставке пропущенные поля получают значения по умолчанию.

P <index_id> <db> <table> <index> <columns> [<fcolumns>]

Относится только к столбцам, не указанным в <columns> при открытии индекса!

Всегда передавайте значения для всех полей из <columns>

NULL как значение по умолчанию не работает.
Вместо NULL вставится пустая строка.

Глючит с типами данных BINARY, ENUM, TIMESTAMP

Российские интернет-технологии

Примеры использования в

Российские интернет-технологии

Use case 1

Справочник забаненных email'ов

id	name	domain	mail_code	created	partner_id	ip
2	avoth	plol.com	8	2006-08-16 15:08:07	1	3232236478
3	anna	store.org	5	2006-08-16 16:30:46	19	3387996560
4	bubble	wiki.org	8	2006-08-16 16:39:30	1	3357320367
5	zlol	c12.net	8	2006-08-16 20:52:09	3	3386664249

Заменили SELECT * FROM ...WHERE name='...' AND domain='...' на выборку через HS

Одна таблица, один сервер на ДЦ. Master-master репликация между ДЦ.

~52 миллиона строк, ~5 Гб

Все данные в памяти. Используем постоянные соединения.

Use case 1

Справочник забаненных email'ов

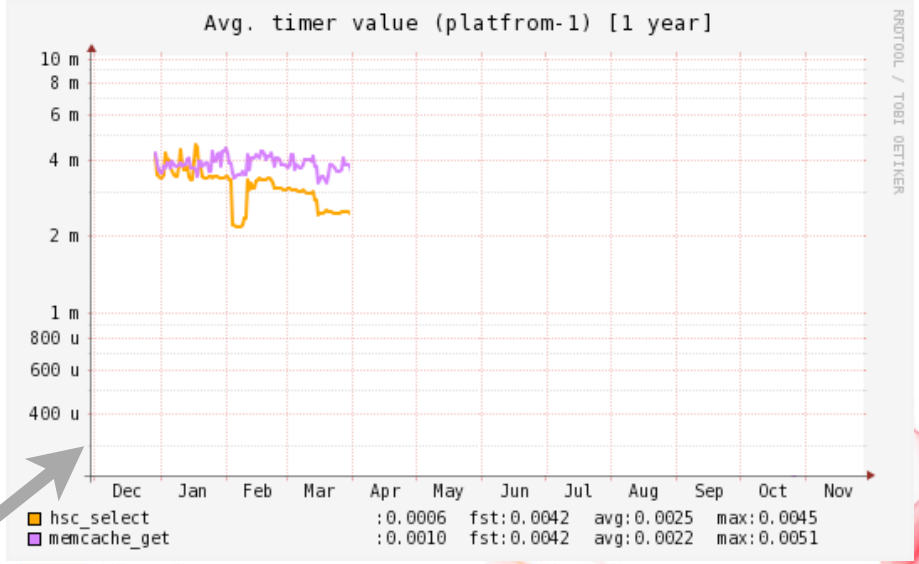
Dual-core Intel(R) Xeon(R) CPU E5503 @ 2.00 Ггц

60% CPU, LA ~ 0.5

Вставка/обновление идет через SQL, <10 RPS

Выборка через HS

~1000 RPS, 3 мс на чтение



время в миллисекундах

Российские интернет-технологии

Use case 2

Persistent хранилище сессий

id	ts	data
1:200:012b031ce0aa9f2357860bc884afabe1	1309728779	a:18:{s:7:"user_id";i:162130103;s:13:"located_state";i:0;s:19:"notify_when_located
1:200:01c1cc3d66bebca1ada7cbb1bb88c2e2	1313544532	a:18:{s:7:"user_id";i:0;s:13:"located_state";i:0;s:19:"notify_when_located";i:1;s:12:"
1:200:0243ba9fd935643022ad555ccb2e1fcf	1308866762	a:23:{s:7:"user_id";i:144063306;s:13:"located_state";i:0;s:19:"notify_when_located
1:200:02b1163810a4b92aec0be5ab2714c41b	1310056296	a:23:{s:7:"user_id";i:245870921;s:13:"located_state";i:0;s:19:"notify_when_located
1:200:0827e390a81452270fc1e63624d3fbe6	1315828681	a:18:{s:7:"user_id";i:0;s:13:"located_state";i:0;s:19:"notify_when_located";i:1;s:12:"
1:200:08374703b88a61a14c5ef246f58ea8dc	1310239929	a:11:{s:7:"user_id";i:0;s:13:"located_state";i:0;s:19:"notify_when_located";i:1;s:12:"
1:200:099c8449c6b3825b339260ffe57cca1e	1307918778	a:27:{s:7:"user_id";i:174196751;s:13:"located_state";i:1;s:19:"notify_when_located
1:200:0d168e6aa60e4dbf04e4338c12bc8d81	1326767013	a:30:{s:7:"user_id";i:226661853;s:13:"located_state";i:1;s:19:"notify_when_located
1:200:0d41dbc234b9d84210d6e3278877d285	1309462393	a:22:{s:7:"user_id";i:211055066;s:13:"located_state";i:0;s:19:"notify_when_located

Хранилище ключ-значение: выбор/изменение/удаление ряда через HS

Периодически удаляем устаревшие данные через SQL

1 таблица, 1 сервер/ДЦ, ~16 млн рядов, ~23 Гб

Все данные в памяти. Используем постоянные соединения.

Use case 2

Persistent хранилище сессий

12-core Intel(R) Xeon(R) CPU X5650 @ 2.67 Ггц8% CPU, LA ~ 5

Вставка: <10 RPS, ~1,2 мс/запрос

Изменение: ~180 RPS, ~1,3 мс/запрос

Выборка: ~3500 RPS, ~0,5 мс/запрос

Изначально было медленнее. После переезда с MySQL/InnoDB на Percona Server/XtraDB получили ~ 4x прирост производительности.

Use case 3

Шардированное persistent хранилище сессий

bucket	hash	ts	data
301	2b2afd841358c37d24f032675ee8e4be	1325717643	a:32:{s:7:"user_id";i:186309856;s:13:"located_state";i:1;s:19:"notify_when_l...
301	2ee318656f357b8a3aade908da0be37b	1329224832	a:33:{s:7:"user_id";i:240403199;s:13:"located_state";i:1;s:19:"notify_when_l...
301	4409e1663582b2e5563f10b394552361	1326269422	a:33:{s:7:"user_id";i:197481393;s:13:"located_state";i:1;s:19:"notify_when_l...
301	84e765c38a46093e2a06308c0234002e	1328089066	a:22:{s:7:"user_id";i:0;s:13:"located_state";i:0;s:19:"notify_when_located";i:1;...
301	8b8e982c4cf222f78cb9434f82c81b24	1347955687	a:32:{s:7:"user_id";i:211035528;s:13:"located_state";i:1;s:19:"notify_when_l...
301	9bf4917d960b4acbd1f30970ff6e2588	1354080964	a:28:{s:7:"user_id";i:164580752;s:13:"located_state";i:1;s:19:"notify_when_l...
302	948193e78f2cf2d10ce1c4c69c1dc4e7	1329422747	a:28:{s:7:"user_id";i:193050043;s:13:"located_state";i:0;s:19:"notify_when_l...
302	b229d7e37112563fbc3e277d9ddb42ac	1354108825	a:24:{s:7:"user_id";i:169647761;s:13:"located_state";i:0;s:19:"notify_when_l...
302	c0128b94c0e0fbaec0f4f223d134631f	1353396690	a:32:{s:7:"user_id";i:188038190;s:13:"located_state";i:1;s:19:"notify_when_l...

Теперь 10 000 таблиц/100 баз, 1 MySQL, 1 сервер

Распределены по случайно сгенерированному хешу

~10 млн рядов, ~20 Гб

Все данные в памяти. Используем постоянные соединения.

Use case 3

Шардированное persistent хранилище сессий

12-core Intel(R) Xeon(R) CPU X5650 @ 2.67 Ггц

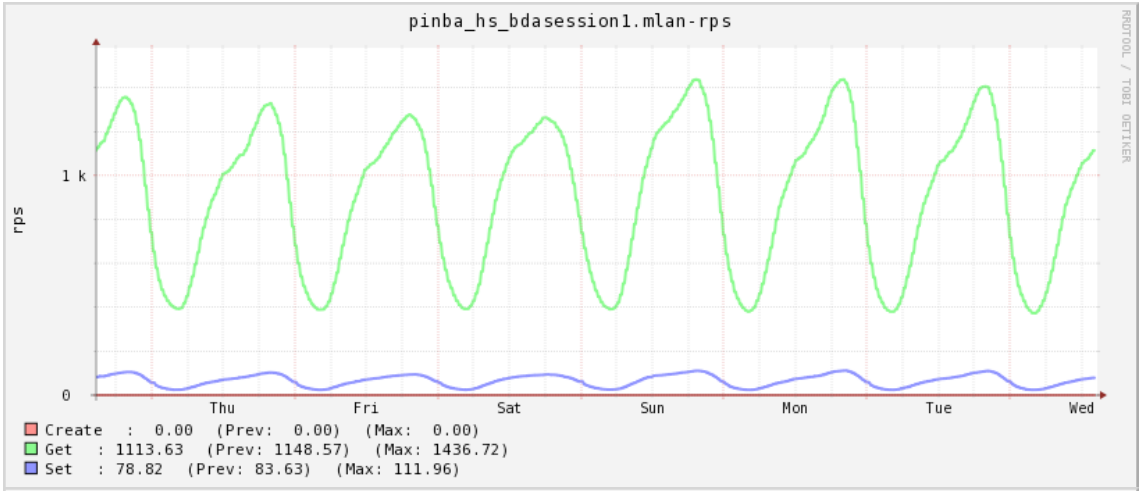
8% CPU, LA ~ 5

Вставка: <10 RPS, ~1,3 мс/запрос

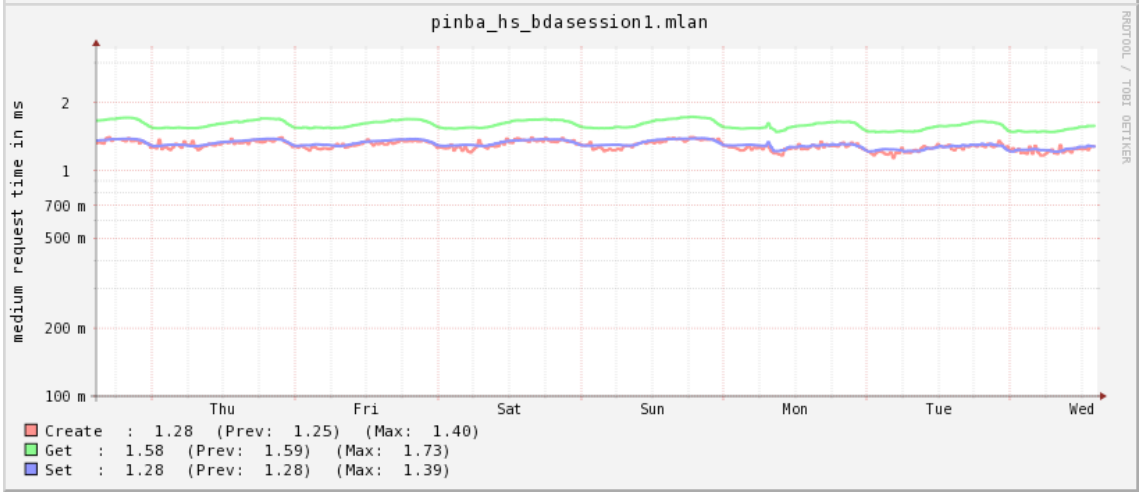
Изменение: ~180 RPS, ~1,3 мс/запрос

Выборка: ~3500 RPS, ~1,6 мс/запрос

RPS



Среднее время в мс



И в чем выгода шардинга?

Однотабличное решение работало хорошо, но плохо справлялось с большой нагрузкой на запись. Одна таблица была «горячим местом».

Вторая проблема: при росте таблицы скорость работы падает.

Удаляйте ненужные ряды ежедневно.

Попробуйте решение с шардингом и сравните с однотабличным вариантом в условиях вашего приложения.

Use case 4

Persistent кеш

user_id	ts	data
212920	1329403894	a:2:{s:9:"interests";a:9:{i:93;i:1;i:1553;i:1;i:3575;i:1;i:126478;i:1;i:287957;i:1;i:314134;i:1;i:...
272920	1350963277	a:2:{s:9:"interests";a:11:{i:357;i:1;i:1454;i:1;i:1532;i:1;i:1612;i:1;i:8469;i:1;i:65846;i:1;i:100...
452920	1324394686	a:2:{s:9:"interests";a:3:{i:1715;i:1;i:2663;i:1;i:15002;i:1;}s:12:"count_active";i:3;}
472920	1341772506	a:2:{s:9:"interests";a:17:{i:122;i:1;i:328;i:1;i:331;i:1;i:357;i:1;i:420;i:1;i:495;i:1;i:3086;i:1;i:4...
2422920	1324992145	a:2:{s:9:"interests";a:5:{i:86;i:1;i:99;i:1;i:1000000499;i:1;i:1000003220;i:1;i:1000009839;i:1...
3572920	1342571425	a:2:{s:9:"interests";a:17:{i:154;i:1;i:436;i:1;i:506;i:1;i:1243;i:1;i:1530;i:1;i:1556;i:1;i:2824;i:1...
6372920	1327340629	a:2:{s:9:"interests";a:7:{i:404;i:1;i:1663;i:1;i:2831;i:1;i:8285;i:1;i:14532;i:1;i:16698;i:1;i:100...
7952920	1326721814	a:2:{s:9:"interests";a:1:{i:420;i:1;}s:12:"count_active";i:1;}
8182920	1326721934	a:2:{s:9:"interests";a:3:{i:14038;i:1;i:15077;i:1;i:16316;i:1;}s:12:"count_active";i:3;}

Заменили memcached на HS из-за того, что реинициализация кеша шла долго.

32 млн рядов, 14 Гб, распределено по 10 000 таблицам, 1 сервер/ДЦ

Только операции ключ-значение: get и set.

Все данные в памяти. Используем постоянные соединения.

Use case 4

Persistent кеш

12-core Intel(R) Xeon(R) CPU X5650 @ 2.67 Ггц

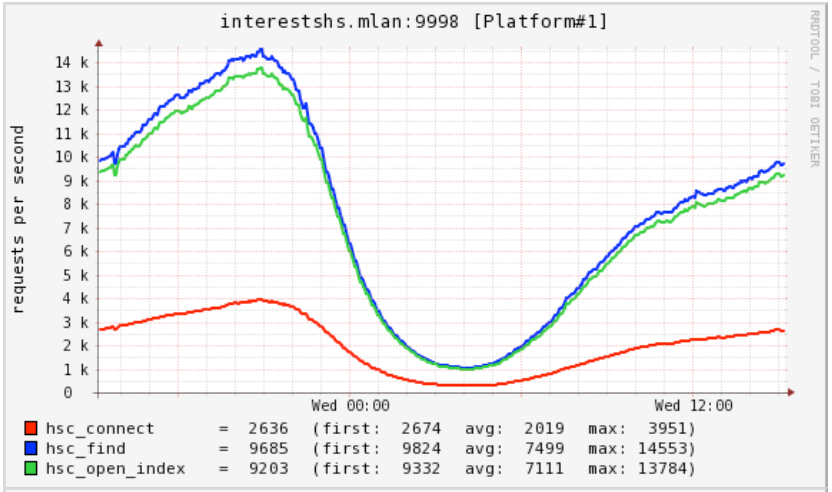
11% CPU, LA ~ 5

Вставка: <10 RPS, ~0,4 мс/запрос

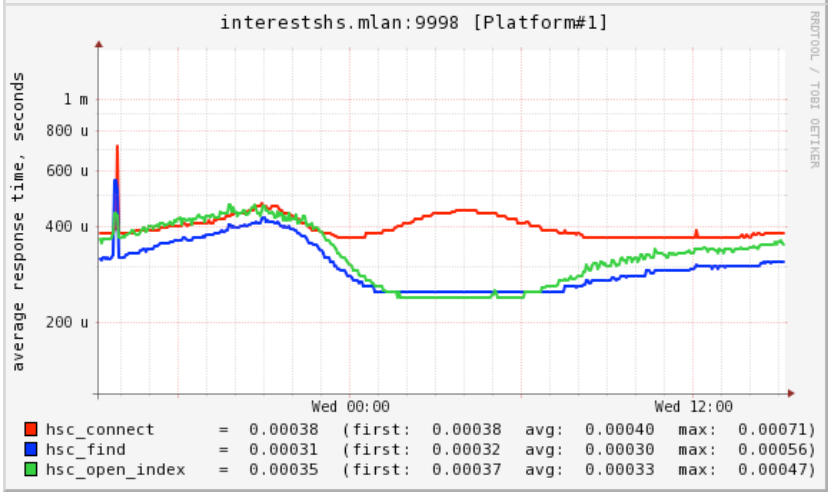
Изменение: <10 RPS, ~0,4 мс/запрос

Выборка: 14500 RPS в пике, ~0,5 мс/запрос

RPS



Среднее время в мс



«Тюнинг»

Российские интернет-технологии

- Попробуйте делать шардинг по ключу выборки при датасетах > 10 млн рядов
- Перейдите на Percona Server/XtraDB
- Используйте постоянные соединения при доступе к HS

Про pconnect'ы

Есть одна проблема с постоянными соединениями.

Следующая итерация/реквест наследует ваш открытый сокет.

- В протоколе HS запросы и ответы не имеют уникальных id, поэтому их нельзя надежно сопоставить
- Выбираем key, value где key = '...', проверяем что ключ в ответе совпадает с ключом в запросе
- Переоткрываем соединения при синтактических и I/O ошибках
- Не допускайте передачи сокета с недочитанными данными к следующей итерации

С чем еще можно поиграть

- InnoDB ROW_FORMAT
- InnoDB KEY_BLOCK_SIZE
- HASH-индексы
- Объединение нескольких индексов в один многоколоночный

Финальный аккорд

Российские интернет-технологии

FAQ

1) Могу ли я использовать одну из библиотек-клиентов для HS из интернетов или мне обязательно писать свою?

Если вы хотите использовать pconnect'ы то «допилите» существующую библиотеку или напишите свою так, чтобы решить проблемы, упомянутые тремя слайдами выше. В остальных случаях можно брать готовые решения.

2) Зачем мне использовать ~~непонятную фигню~~ Handlersocket вместо нормальной NoSQL БД типа MongoDB или Redis?

1) Для тех, кто использует MySQL и не может отказаться от нее — вы можете часть функционала перевести на более быстрый доступ через HS, работая с теми же данными консистентно. SQL при этом никто не отменяет и весь его функционал будет доступен.

2) Если вы можете «вместить» ваше приложение в простой набор команд HS — у вас будет отличный NoSQL с некоторыми старыми добрыми фичами SQL-мира: сохранность данных, хорошее масштабирование по ядрам, эффективное хранение данных и т. д.

Полезные ссылки

Клиентские библиотеки
https://github.com/DeNADev/HandlerSocket-Plugin-for-MySQL/blob/master/README

Исходники HS
https://github.com/DeNADev/HandlerSocket-Plugin-for-MySQL/

Документация
https://github.com/DeNADev/HandlerSocket-Plugin-for-MySQL/tree/master/docs-en

Статья от разработчиков HS
http://yoshinorimatsunobu.blogspot.ru/2010/10/using-mysql-as-nosql-story-for.html

Страница о HS у Percona
http://www.percona.com/doc/percona-server/5.5/performance/handlersocket.html

Must-see презентация от автора HS
http://www.slideshare.net/akirahiguchi/handlersocket-20100629en-5698215

Спасибо!

Вопросы?

 @ryba_xek

 s@averin.ru

Слайды, код, mind map:
http://averin.ru/slides/

Российские интернет-технологии