



MySQL Partitioning: The Spider Solution

Frédéric Descamps

Percona Live London 2011

Who am I ?

Frédéric Descamps

@lefred

<http://about.be/lefred>

Managing mysql since 3.23 (as far as I remember)

devops believer

Thank You to Our Sponsors

Platinum Sponsor



Gold Sponsor



Silver Sponsors



Percona Live London Sponsors

Exhibitor Sponsors



Friends of Percona Sponsors

Couchbase



Monty Program



Tokutek



Media Sponsors



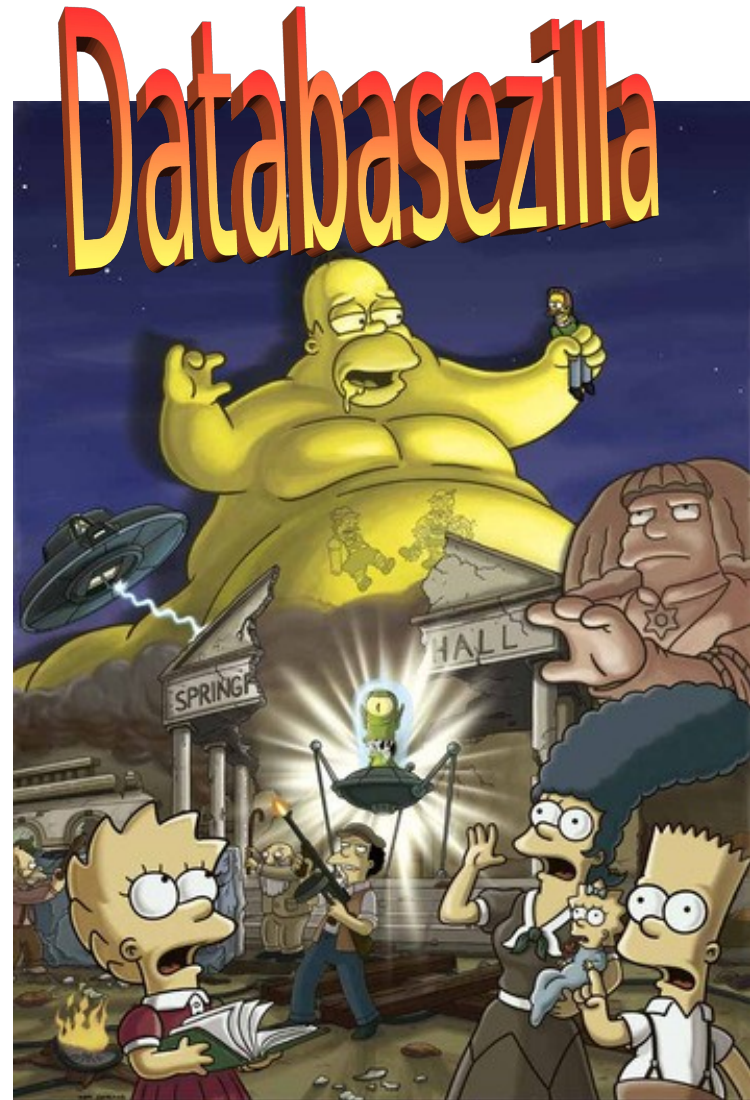
O'REILLY

Description of the problem

Currently the application is using one huge database with big tables

Slow to retrieve data

Impossible to maintain it
(create indexes, optimize or repair tables, archive, ...)



2.3

Big Tables

```
# du -sh
2.3T .
# ls -lh *.ibd -S | head -n 5
-rw-rw---- 1 mysql mysql 505G Aug 17 15:32 traceproperty_ejb3.ibd
-rw-rw---- 1 mysql mysql 322G Dec  4 2009 traceproperty.ibd
-rw-rw---- 1 mysql mysql 291G Dec  4 2009 trace.ibd
-rw-rw---- 1 mysql mysql 253G Aug 17 15:33 trace_ejb3.ibd
-rw-rw---- 1 mysql mysql 229G Aug 17 15:33 trace_poll_ejb3.ibd
```

500

Big Tables (2)

TABLES	rows	DATA	idx	total_size	idxfrac
3	4154.87M	322.52G	177.17G	499.69G	0.55
1	1409.36M	155.84G	161.50G	317.34G	1.04
1	1492.43M	115.09G	133.89G	248.98G	1.16

Is it bad ?

- Too much data (very large working set)
- Too many writes (the IO system, can't keep up with the amount of writes being sent)



We need to scale !

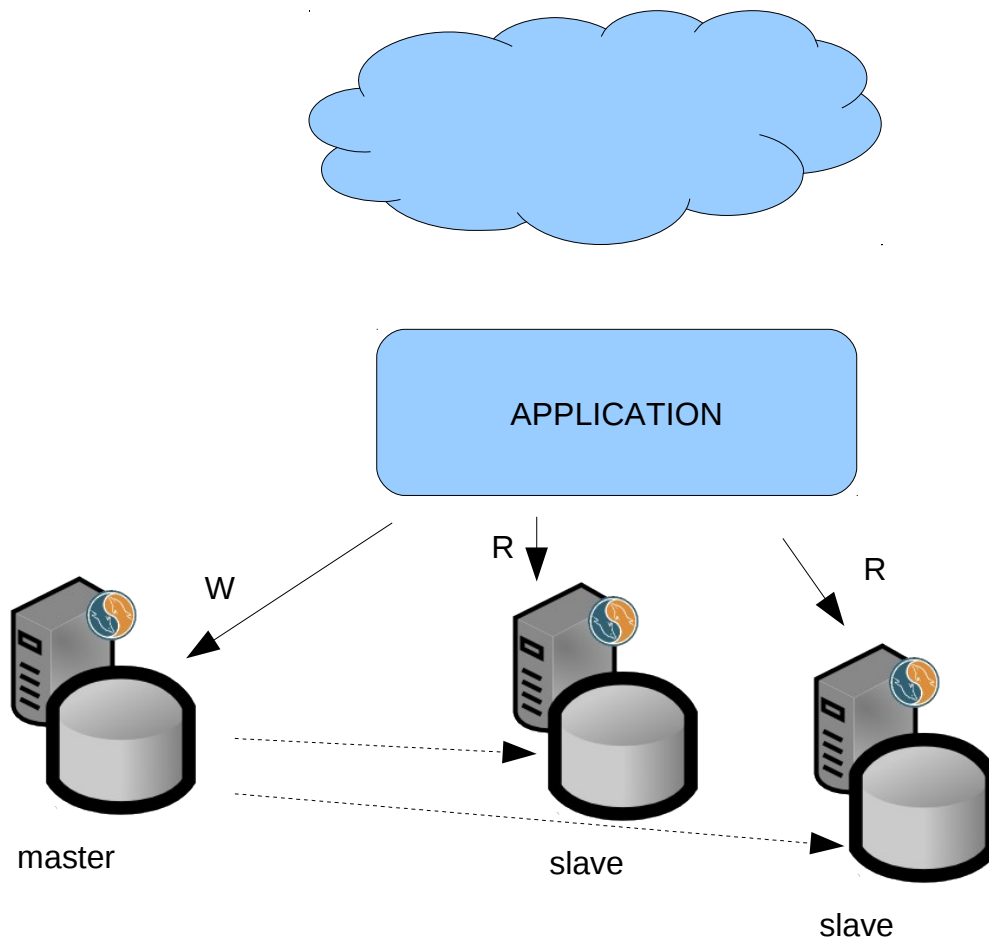
I have a dream !



This is my dream:

- Speed up the long requests (used for reporting)
- Be able to maintain the tables
- Scale easily reads.... and writes !

Is replication an option ?



Replication helps to split the read load, but doesn't improve for writes and certainly not to save disk space.

What could help then ?

- Certainly Partitioning
- Eventually Sub-Partitioning
- Try to spread the load on multiple servers



We need shards !

Discovering

Let's discover a promising engine.

What is a shard ?

A database shard is a horizontal partition in a database or search engine. Each individual partition is referred to as a shard or database shard.

Splitting shards across multiple isolated instances requires more than simple horizontal partitioning.

Sharding - splitting up large tables by locating the data across many servers

2.26

Spider

Spider storage engine enables tables of different MySQL instances to be treated like a table of a same instance. Because xa transaction and partitioning is supported, it can do decentralized arrangement to two or more servers of data of same table.



Spider: key features

- MySQL storage engine
- Built on top of the partition engine
- Decentralize a partition to a remote server
- Independent from application
- Transparent for users
- Developed by Kentoku Shiba on launchpad
<https://launchpad.net/spiderformysql>
- Binary download at http://spiderformysql.com/download_spider.html
- Easy to expand
- SSL support

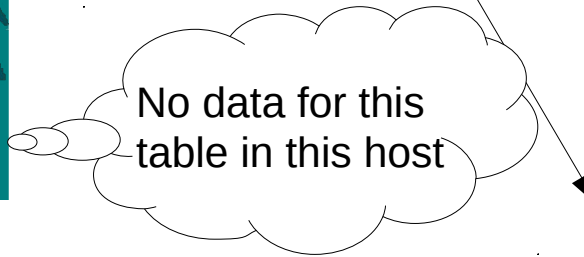
Spider is not GA

Spider is not production ready

What's the concept ?

Table traces		
Partition 1	Year <2008	host2
Partition 2	Year <2009	host3
Partition 3	Year <2010	host4
Partition 4	Year <2011	host5
Partition 5	Year < MAXVALUE	host6

host1
MySQL with spider



host2
MySQL without spider



host3
MySQL without spider



host4
MySQL without spider



host5
MySQL without spider



host6
MySQL without spider



What's the concept ? (2)

```
select *  
from traces where  
date = '2009-03-14';
```

Table traces		
Partition 1	Year <2008	host2
Partition 2	Year <2009	host3
Partition 3	Year <2010	host4
Partition 4	Year <2011	host5
Partition 5	Year < MAXVALUE	host6

host1
MySQL with spider



No data for this
table in this host

host2
MySQL without spider



host3
MySQL without spider



host4
MySQL without spider



host5
MySQL without spider



host6
MySQL without spider



What's the concept ? (3)

```
select *  
from traces where  
date = '2009-03-14';
```

Table traces		
Partition 1	Year <2008	host2
Partition 2	Year <2009	host3
Partition 3	Year <2010	host4
Partition 4	Year <2011	host5
Partition 5	Year < MAXVALUE	host6

host1
MySQL with spider



No data for this
table in this host

host2
MySQL without spider



host3
MySQL without spider



host4
MySQL without spider



host5
MySQL without spider



host6
MySQL without spider



Spider a MySQL engine

Engine	Support	Comment	Transactions	XA	Savepoints
SPIDER	YES	Spider storage engine	YES	YES	NO
InnoDB	DEFAULT	Supports transactions, row-level locking, and foreign keys	YES	YES	YES
MRG_MYISAM	YES	Collection of identical MyISAM tables	NO	NO	NO
PERFORMANCE_SCHEMA	YES	Performance Schema	NO	NO	NO
CSV	YES	CSV storage engine	NO	NO	NO
MEMORY	YES	Hash based, stored in memory, useful for temporary tables	NO	NO	NO
MyISAM	YES	MyISAM storage engine	NO	NO	NO

Engine: SPIDER
Support: YES
Comment: Spider storage engine
Transactions: YES
XA: YES
Savepoints: NO

Internals

Spider tables do not use query cache

Spider does not support full-text indexes

Spider does not support R-tree indexes

Spider creates internal tables in mysql schema.

```
+-----+
| Tables_in_mysql (spider%) |
+-----+
| spider_link_failed_log    |
| spider_link_mon_servers   |
| spider_tables             |
| spider_xa                 |
| spider_xa_member          |
+-----+
```

Internals

A spider engine table is created, a link to a table of a remote server is generated like the symbolic link to the file in MySQL.

The linked table can have any type of engine.

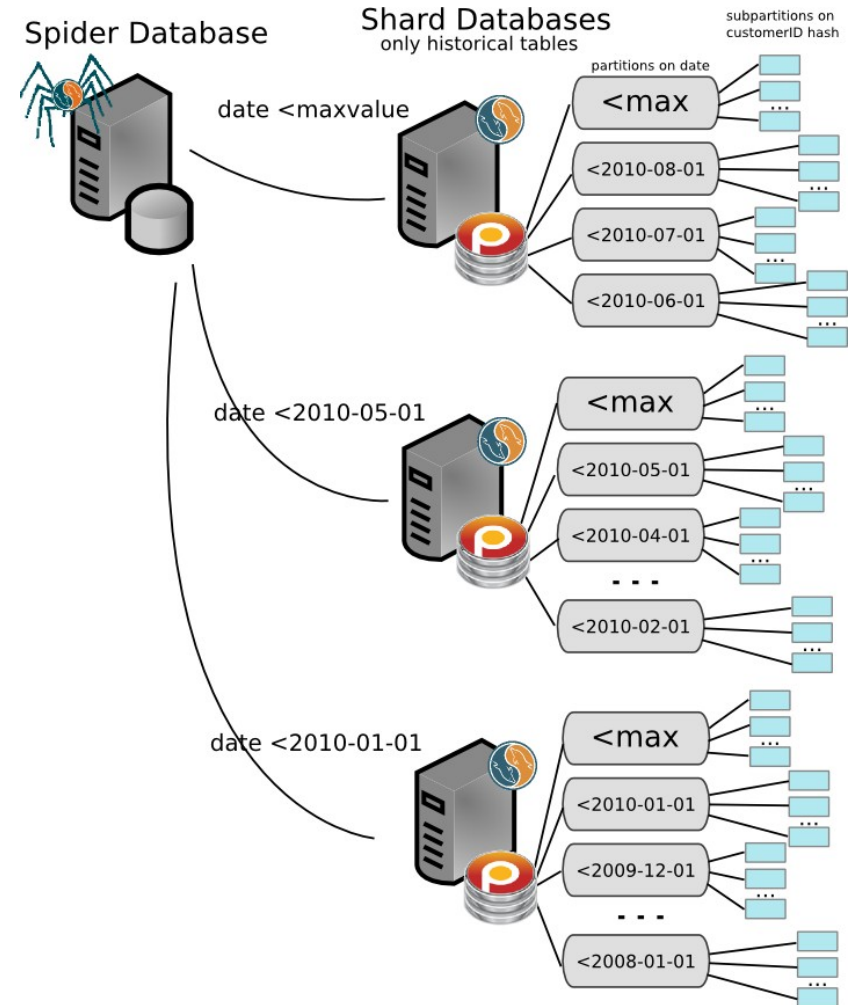
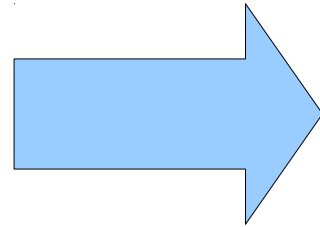
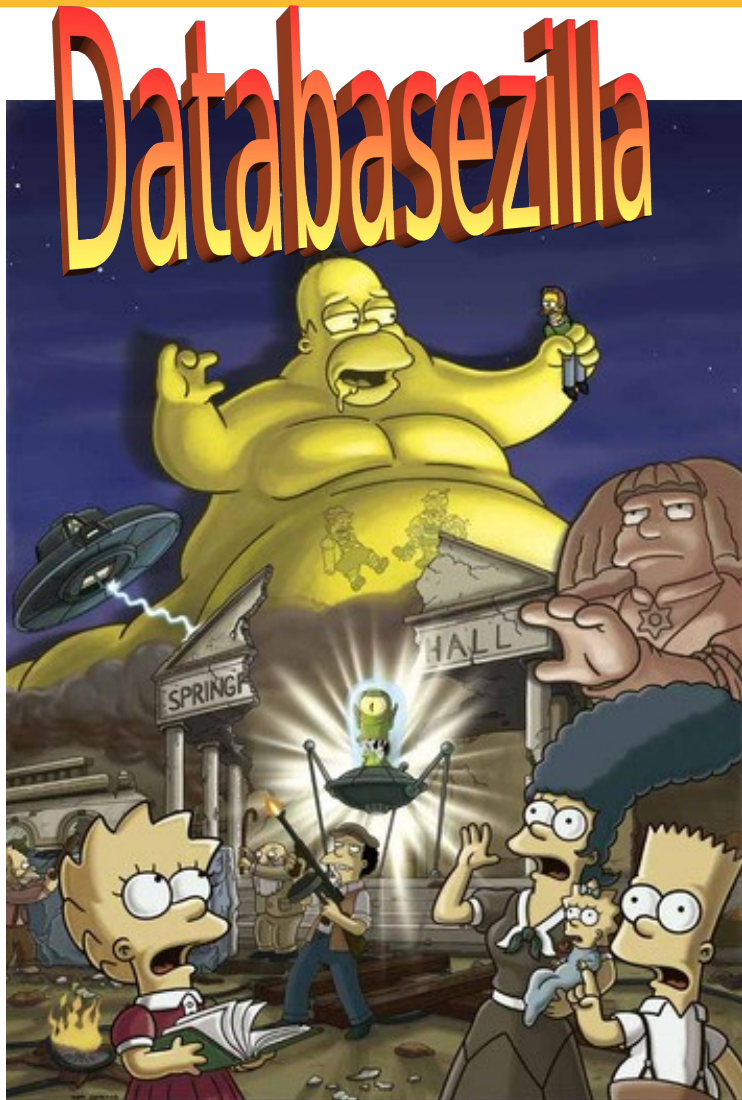
Internals (2)

This link is a connection from a local MySQL server to a remote one. If at least two tables of the spider engine point to one remote server, the connection is shared.

But if two or more connections point to the same table on the remote server, then they use their own connection to use a different respectively transaction.

4

Back to our problem



On the shards

- Create all the needed historical tables on each nodes.
- We use partitioning and sub-partitioning also on each nodes.
- Every partition is split in 10 sub-partitions

```
CREATE TABLE `trace_ejb3` (  
  `id` bigint(20) NOT NULL AUTO_INCREMENT,  
  `timestamp` bigint(20) NOT NULL,  
  `latitude` int(11) DEFAULT NULL,  
  `longitude` int(11) DEFAULT NULL,  
  `speed` smallint(6) DEFAULT NULL,  
  `heading` smallint(6) DEFAULT NULL,  
  `terminalID` int(11) NOT NULL,  
  `mileage` int(11) DEFAULT NULL,  
  `creationtime` bigint(20) NOT NULL,  
  `tracetype` int(11) DEFAULT NULL,  
  `customerID` int(11) NOT NULL DEFAULT '0',  
  KEY (`id`,`customerID`),  
  KEY `idx_trace_tracetype` (`tracetype`),  
  KEY `idx_terminalID_timestamp`  
  (`terminalID`,`timestamp`),  
  KEY `idx_terminalID_timestamp2`  
  (`terminalID`,`timestamp2`),  
  KEY `idx_creationtime` (`creationtime`)  
) ENGINE=InnoDB  
partition by range (timestamp)  
subpartition by linear hash (customerID) subpartitions 10  
(  
  partition p_2010_04 values less than (1272664800000),  
  partition p_2010_05 values less than (1275343200000),  
  partition p_2010_06 values less than (1277935200000),  
  partition p_max values less than (maxvalue)  
);
```

On the spider server

```
CREATE TABLE `trace_ejb3` (  
  `id` bigint(20) NOT NULL AUTO_INCREMENT,  
  `timestamp` bigint(20) NOT NULL,  
  `latitude` int(11) DEFAULT NULL,  
  `longitude` int(11) DEFAULT NULL,  
  `speed` smallint(6) DEFAULT NULL,  
  `heading` smallint(6) DEFAULT NULL,  
  `terminalID` int(11) NOT NULL,  
  `mileage` int(11) DEFAULT NULL,  
  `creationtime` bigint(20) NOT NULL,  
  `tracetype` int(11) DEFAULT NULL,  
  `customerID` int(11) NOT NULL DEFAULT '0',  
  KEY (`id`,`customerID`),  
  KEY `idx_trace_tracetype` (`tracetype`),  
  KEY `idx_terminalID_timestamp`  
  (`terminalID`,`timestamp`),  
  KEY `idx_terminalID_timestamp2`  
  (`terminalID`,`timestamp2`),  
  KEY `idx_creationtime` (`creationtime`)  
) ENGINE=Spider  
Connection ' table "trace_ejb3", user "msandbox",  
password "msandbox" '  
partition by range (timestamp) (  
  partition pt1 values less than (1199142000000)  
  comment 'host "127.0.0.1", port "16850"',  
  partition pt2 values less than (1270072800000)  
  comment 'host "127.0.0.1", port "16849"',  
  partition pt3 values less than (MAXVALUE)  
  comment 'host "127.0.0.1", port "16848"'  
);
```

- We create all the non historical tables as usual.
- We create the historical tables using the spider engine.



Zoom !

ENGINE=**Spider**

```
Connection ' table "trace_ejb3", user "msandbox",  
password "msandbox" '  
partition by range (timestamp) (  
    partition pt1 values less than (1199142000000)  
    comment 'host "127.0.0.1", port "16850"',  
    partition pt2 values less than (1270072800000)  
    comment 'host "127.0.0.1", port "16849"',  
    partition pt3 values less than (MAXVALUE)  
    comment 'host "127.0.0.1", port "16848" '  
);
```

A note on backups

- You can backup each shards individually
- You can create a slave of the spider server that will replicate all transaction to a single slave and take a full backup from it... (you need to enable XA transaction).
- Using XtraBackup on the spider machine won't backup the data.

XA Transactions

XA transactions give the ability to permit multiple separate transactional resources to participate in a global transaction.

10

Does it improve ?

```
select * from xxx where customerID=57 and  
timestamp between unix_timestamp('2010-03-  
01')*1000 and unix_timestamp('2010-03-15')*1000;
```

Without spider :

14589 rows in set
(9.04 sec)

With spider : *Query not spread on different servers*

14589 rows in set
(0.93 sec)



10 times faster !



```
select * from xxx where customerID=57 and
timestamp between unix_timestamp('2010-03-
01')*1000 and unix_timestamp('2010-04-15')*1000;
```

Without spider :

65342 rows in set
(2.40 sec)

With spider :

Query spread on two servers

65342 rows in set
(0.32 sec)



```
select * from xxx where customerID=59 and
timestamp between unix_timestamp('2010-03-
01')*1000 and unix_timestamp('2010-04-15')*1000;
```

Without spider :

2199 rows in set
(2.28 sec)

With spider :

Query spread on two servers

2199 rows in set
(0.05 sec)



```
select * from xxx where customerID=59 and
timestamp between unix_timestamp('2007-03-
01')*1000 and unix_timestamp('2010-04-15')*1000;
```

Without spider :

167701 rows in set
(2.56 sec)

With spider :

Query spread on all the servers

167701 rows in set
(2.74 sec)



```
select * from xxx where customerID between 50
and 62 and timestamp between
unix_timestamp('2007-03-01')*1000 and
unix_timestamp('2010-04-15')*1000;
```

Without spider :

1202412 rows in set
(4.13 sec)

With spider :

Query spread on all servers

1202412 rows in set
(7.79 sec)



```
select * from xxx where customerID between 70
and 80 and timestamp between
unix_timestamp('2010-05-01')*1000 and
unix_timestamp('2010-06-15')*1000;
```

Without spider :

92799 rows in set
(2.44 sec)

With spider :

Query spread on two servers

92799 rows in set
(0.73 sec)



2.27

Spider 2.27 (beta) is out !

Since Oct 19h, Spider 2.27 is out, Kentoku announced it on his blog.

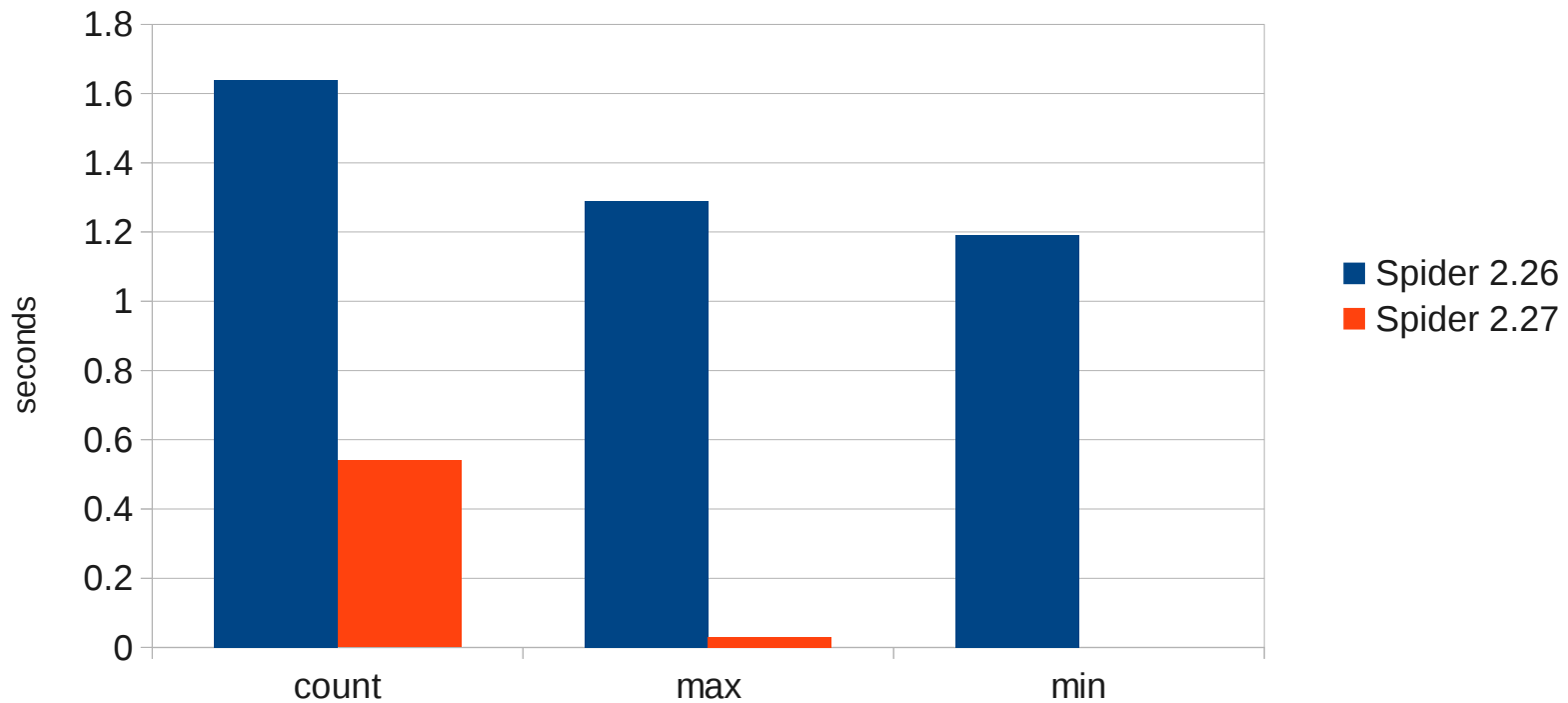
<http://wild-growth.blogspot.com/2011/10/mysqlspidervpspider-227-vp-016-released.html>

- Support R-Tree index.
- Support direct updating for SQL access.
- Support handlersocket increment and decrement.
- Change table parameter from "net_timeout" to "connect_timeout", "net_read_timeout" and "net_write_timeout".
- Change server parameter from "spider_net_timeout" to "spider_connect_timeout", "spider_net_read_timeout" and "spider_net_write_timeout".
- Add UDF paramter "access_mode".
- "spider_direct_sql" support handlersocket access.
- Add server parameter "spider_hs_ping_interval", "spider_error_read_mode" and "spider_error_write_mode".
- Add table parameter "hs_write_to_read", "error_read_mode" and "error_write_mode".
- Performance improvement for "COUNT(*)", "MAX" and "MIN" without clause.

Spider 2.27 (beta) is out !

Testing improvement for count, max and min functions

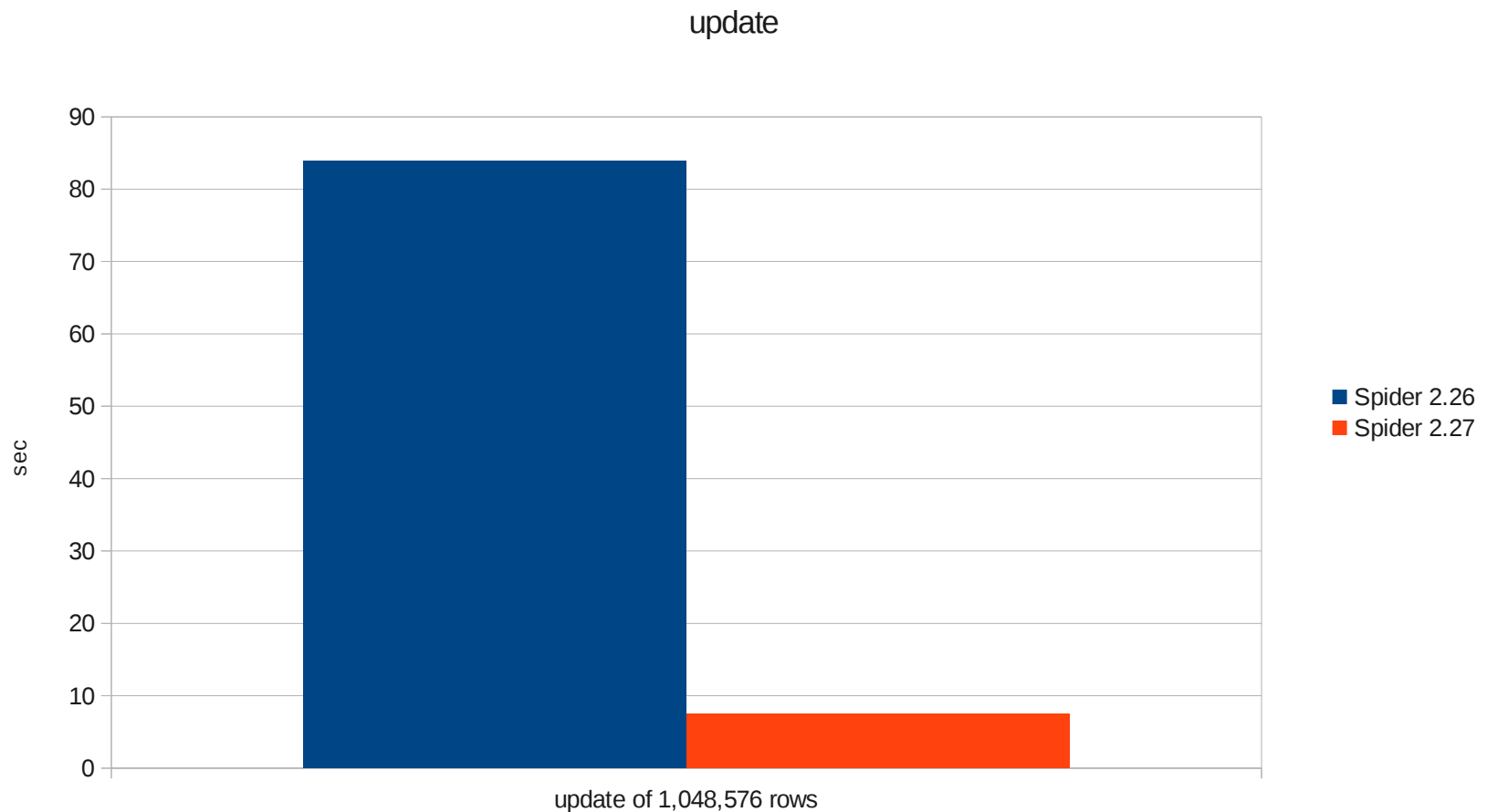
New 2.27 features



1,048,576 rows

Spider 2.27 (beta) is out !

Update improvement



Some examples (on EC2)

```
SINGLE mysql > select count(*) from traces ;  
+-----+  
| count(*) |  
+-----+  
| 20502013 |  
+-----+  
1 row in set (49.86 sec)
```

```
SPIDER mysql > select count(*) from traces;  
+-----+  
| count(*) |  
+-----+  
| 20502013 |  
+-----+  
1 row in set (11.42 sec)
```



Some examples (on EC2)

```
SPIDER mysql > explain partitions select * from traces where
customer_id=678 and tr_timestamp > '2010-03-01' and '2011-03-01'\G
***** 1. row *****
      id: 1
select_type: SIMPLE
      table: traces
  partitions: traces_2010
         type: ALL
possible_keys: NULL
          key: NULL
       key_len: NULL
          ref: NULL
         rows: 3046361
      Extra: Using where with pushed condition
1 row in set, 1 warning (0.00 sec)
```



2704 rows in set (14.37sec)



2704 rows in set (0.41 sec)

Some examples (on EC2)

```
SPIDER mysql > explain partitions select * from traces where
customer_id=62 and tr_timestamp > '2009-02-01' and '2011-09-01'\G
***** 1. row *****
      id: 1
  select_type: SIMPLE
        table: traces
  partitions: traces_2009, traces_2010
        type: ALL
possible_keys: NULL
          key: NULL
    key_len: NULL
         ref: NULL
        rows: 4789662
  Extra: Using where with pushed condition
```



4737 rows in set (22.32 sec)



4737 rows in set (2.07 sec)

Some examples (on EC2)

```
SPIDER mysql > explain partitions select * from traces where
customer_id=345 and tr_timestamp > '2008-03-01' and '2011-08-12'\G
***** 1. row *****
      id: 1
select_type: SIMPLE
      table: traces
  partitions: traces_2008, traces_2009, traces_2010
      type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
          rows: 8279923
      Extra: Using where with pushed condition
1 row in set, 1 warning (0.00 sec)
```



6221 rows in set (22.65 sec)



6221 rows in set (4.78 sec)

Some examples (on EC2)

```
SPIDER mysql > explain partitions select * from traces where
customer_id=490 and tr_timestamp > '2005-03-01' and '2011-02-12'\G
***** 1. row *****
      id: 1
  select_type: SIMPLE
        table: traces
  partitions: traces_2006, traces_2008, traces_2009, traces_2010
         type: ALL
possible_keys: NULL
          key: NULL
     key_len: NULL
         ref: NULL
        rows: 20503316
  Extra: Using where with pushed condition
1 row in set, 1 warning (0.00 sec)
```



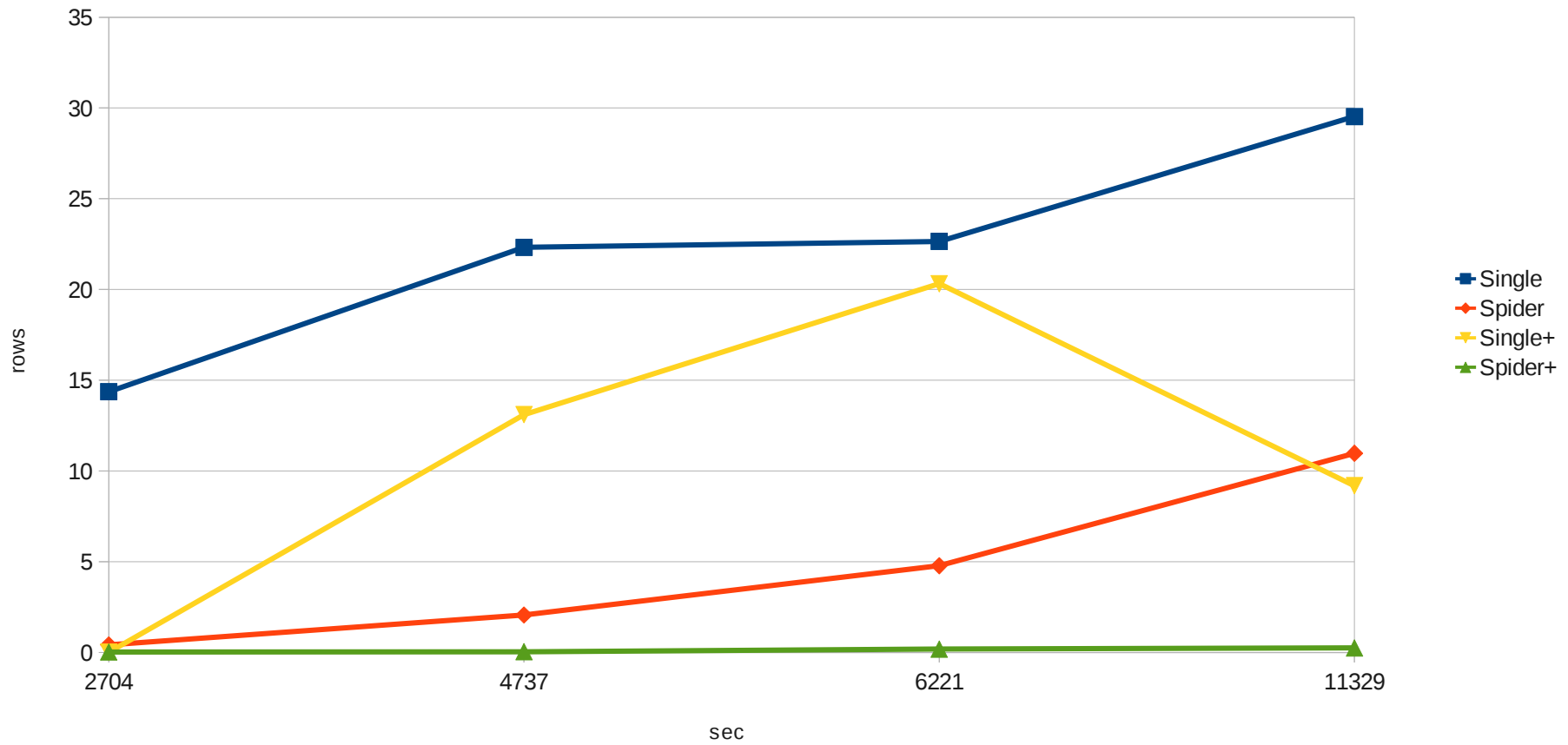
11329 rows in set (29.53 sec)



11329 rows in set (10.98 sec)

Some examples (on EC2)

Select example



sysbench prepare (inserts)

How does it scale for writings ?

The load and disk IO are also spread over the shards and the loss related to network latency can be quickly recovered as soon as we have more nodes.

```
sysbench --test=oltp --db-driver=mysql --mysql-table-engine=innodb  
--mysql-db=pluk --oltp-table-size=4000000 --mysql-user=root --oltp-  
test-mode=complex prepare
```



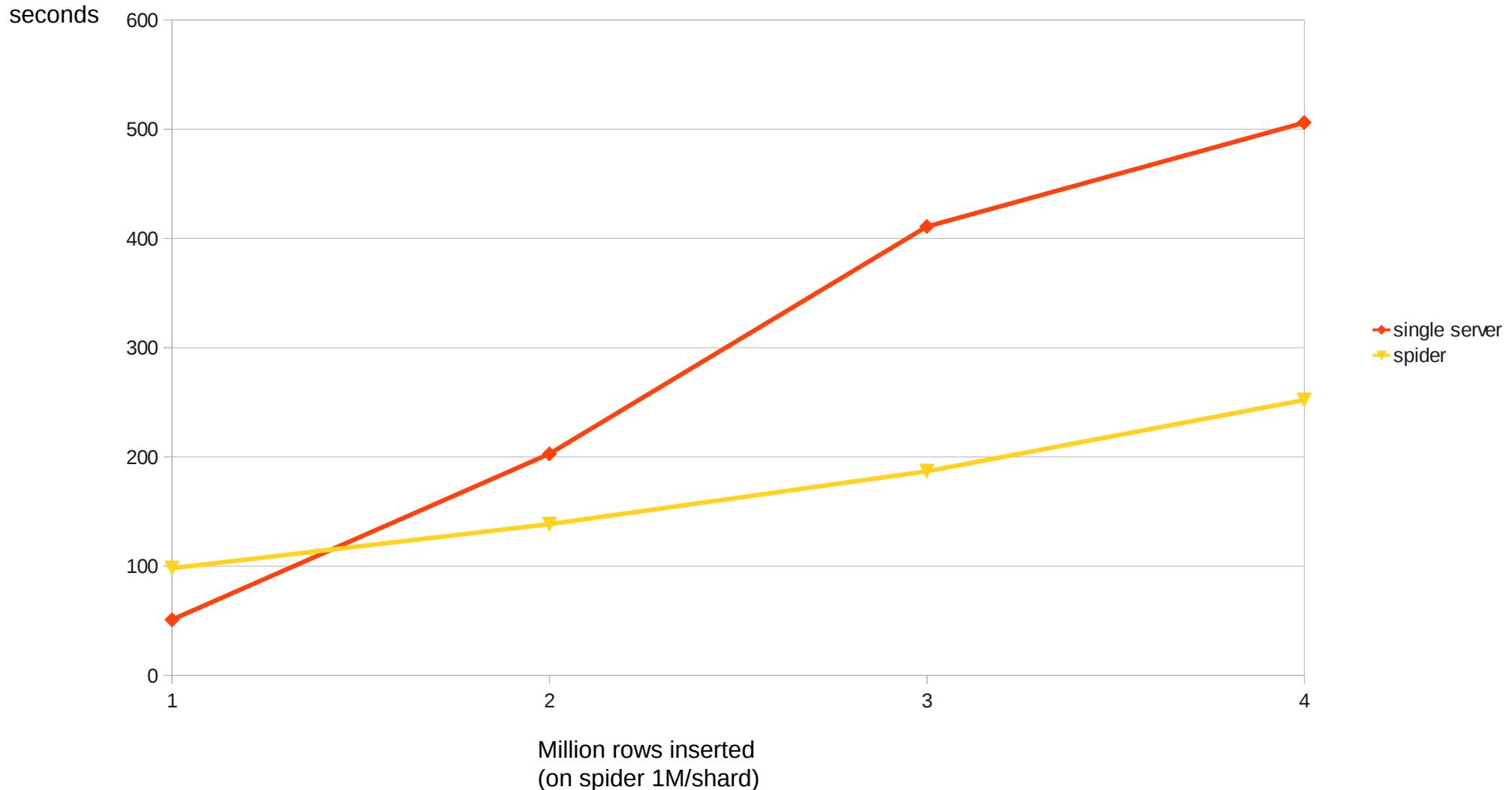
sysbench prepare (inserts)

```
sysbench --test=oltp --db-driver=mysql --mysql-table-engine=spider
--mysql-db=pluk --oltp-table-size=4000000 --mysql-user=root --mysql-
socket=/tmp/mysql.sock --mysql-create-options="ENGINE=spider
connection ' table \"sbtest\", user \"root\" ' partition by range
( id ) (      partition sbtest1 values less than (1000001)
comment 'host \"fred2\", port \"3306\"',      partition sbtest2
values less than (2000001)      comment 'host \"fred3\",
port \"3306\"',      partition sbtest3 values less than (3000001)
comment 'host \"fred4\", port \"3306\"',partition sbtest4 values
less than (MAXVALUE)      comment 'host \"fred5\", port \"3306\"' )"
--oltp-test-mode=complex prepare
```

(*) modified sysbench to support spider



sysbench prepare (inserts)



Updates

Selecting rows from all the shards is slower

And updating them ?

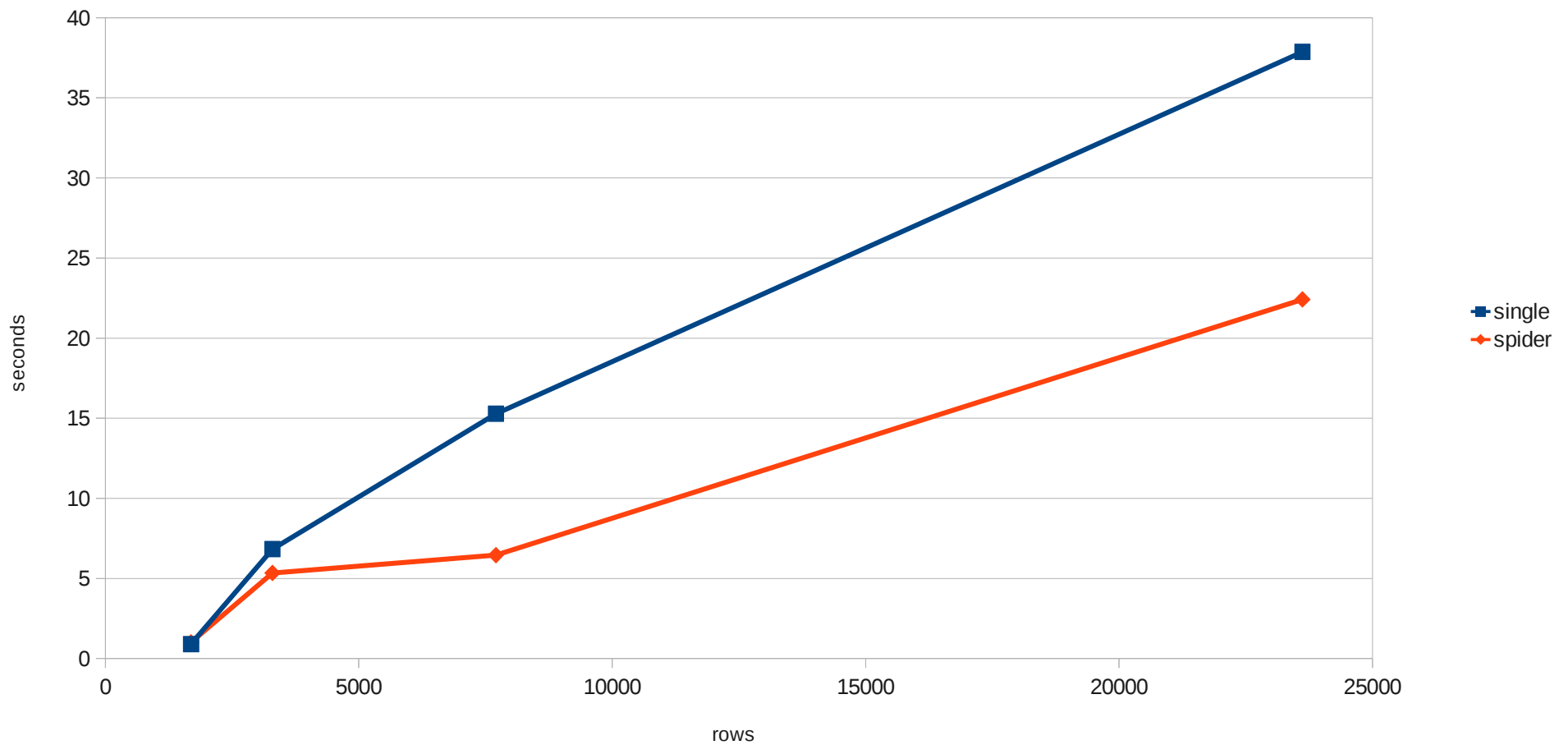
Test with using an update statement with a range covering 1 to 4 shards

```
update traces set speed=speed - 1,type=3 where  
customer_id=123 and tr_timestamp between '2010-01-  
01' and '2010-12-31';
```



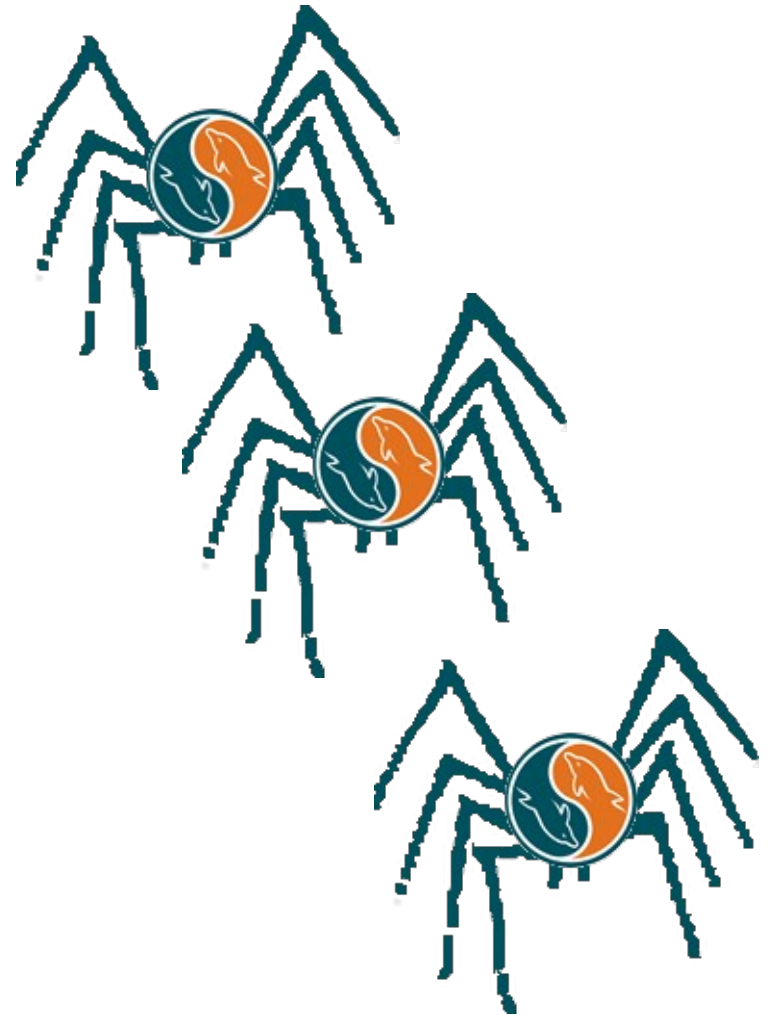
Updates : results

updates

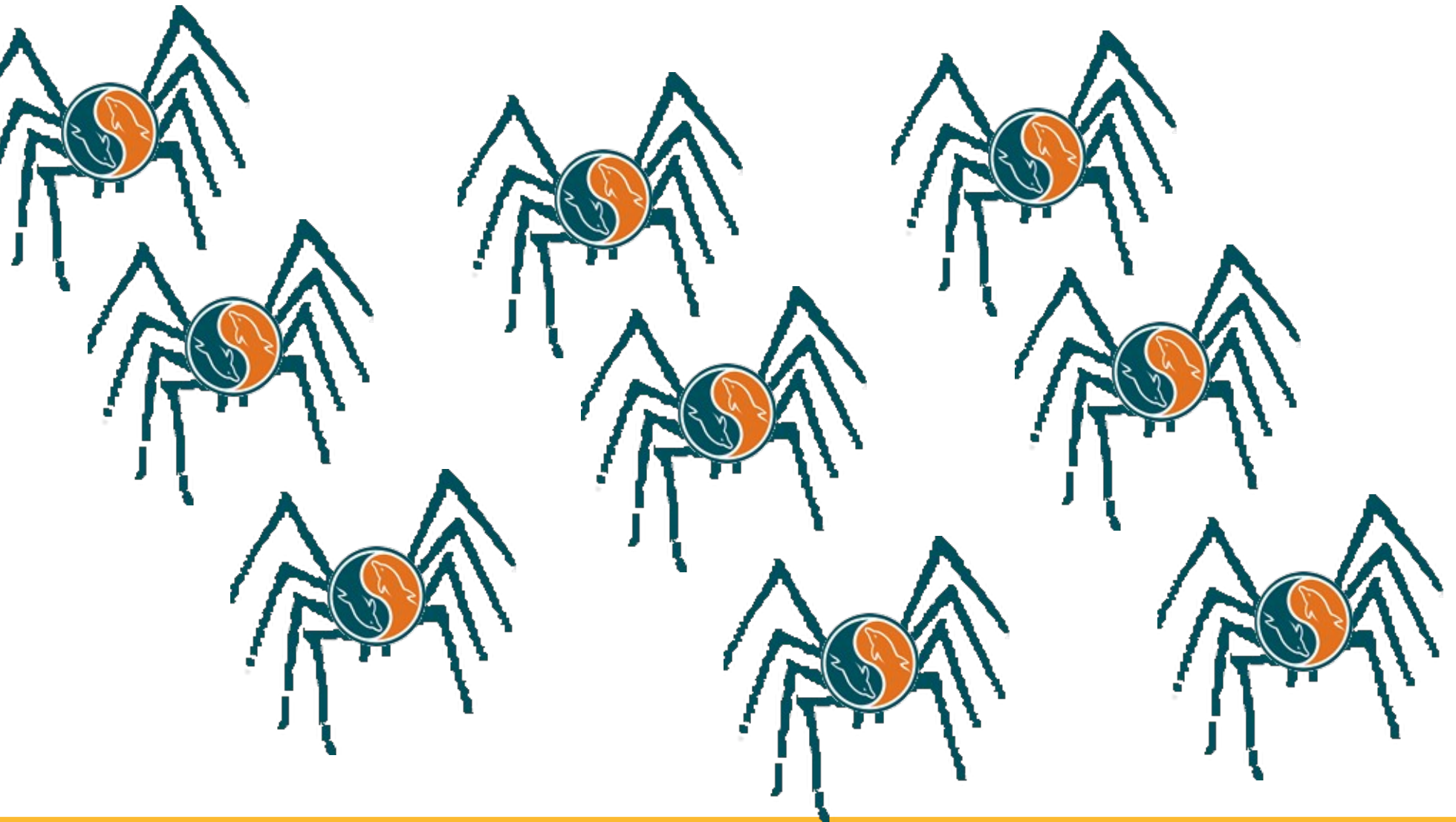


Conclusion

- When the data is spread all over the servers it's slower to retrieve it but **faster** to update it.
- You need then to select very carefully how you partition your tables
- Check with developers and try to profile your queries



Other numbers ?



54

54 files

The source code of spider mysql engine is composed of 54 files

5 1 6 5 2

51652

51,652 lines of code :-)

8

8 BUGS

There are 8 bugs in launchpad:

- 2 critical (fix released)
- 2 high (fix released)
- 2 medium (1 fix released, 1 fix committed)
- 2 low (1 fix releases, 1 fix committed)

92

Spider engine server variables

Spider engine add a large amount (currently 92) of **global variables** that change the behavior and help tuning the system.

Example:

- spider_auto_increment_mode

-1 : The table parameter is adopted.

0 : Normal mode.

(Use a count that Spider get from the remote server with exclusive lock for auto increment value.)

Slow. Switch to quick mode if you use Spider table with table partitioning feature and auto increment column is first column of index.

1 : Quick mode.

(Use a count that Spider has internal for auto increment value.)

Fast but you get duplicate entry if you update same table from multiple Spider and self tables.

2 : Set 0 value.

(Auto increment value is given by the remote server.)

Set 0 value even if you set a value to the auto increment column.

Set 0 value after choosing a inserted partition if you use a table with table partitioning feature.

3 : Auto increment value is given by the remote server, if you set null value to auto increment column. Auto increment value is given by the local server, if you set 0 value to auto increment column.

You should better to use "spider_reset_auto_increment=2or3" if you want to use an auto increment column at remote server.

86

Spider engine table variables

Spider engine provides a large amount of table parameters (86)

Most of them are the same as the global server parameters.

- `connect_timeout(cto)`

Wait timeout of connecting to remote server. This timeout applies only to TCP/IP connections.

0 or more : Seconds of timeout.

The default value is 6

- `error_read_mode(erm)`

Return 0 record at net read error.

That is given to priority when server parameter `spider_error_read_mode` is set.

0 : Return error at net read error.

1 : Return 0 record at net read error.

The default value is 0

5

Spider User Defined Functions

Spider provides also 5 udf's:

- spider_direct_sql : execute the SQL string at remote server.
- spider_bg_direct_sql : background execution of spider_direct_sql
- spider_ping_table
- spider_flush_table_mon_cache : this function is used for refreshing monitoring server information
- spider_copy_tables

29

Spider UDF parameters

Some udf have a large number of extra optional parameters, from 6 to 23 !

Spider

This is a very promising engine still in beta but ready for some tests and benchmarks

*Please enjoy using Spider and growing your business!**

(*) Kentoku Shiba

Annual MySQL Users Conference

Presented by Percona Live

The Hyatt Regency Hotel, Santa Clara, CA

April 10th-12th, 2012

Featured Speakers

Mark Callaghan, Facebook

Jeremy Zawodny, Craigslist

Marten Mickos, Eucalyptus Systems

Sarah Novotny, Blue Gecko

Peter Zaitsev, Percona

Baron Schwartz, Percona

The Call for Papers is Now Open!

Visit www.percona.com/live/mysql-conference-2012/



lefred@percona.com
@lefred

We're Hiring! www.percona.com/about-us/careers/

Questions ?



Thank you !





lefred@percona.com
@lefred

We're Hiring! www.percona.com/about-us/careers/

