

MySQL TokuDB Engine performance benchmark on FusionIO

Environment:

Machine	DELL PowerEdge R720 * 2
CPU	Intel(R) Xeon(R) CPU E5-2630 v2 @ 2.60GHz * 24
memory	128GB
OS	CentOS release 6.5 (Final)
Kernel	2.6.32-431.el6.x86_64 #1 SMP
DB version	10.0.21-MariaDB-log MariaDB Server
raid	Internal Raid
File system	Ext4 with options
disk	FusionIO SX300 1.6TB

Database configuration:

InnoDB:

[client]

port = 3306

socket = /storage/fioa/mysql3306/data/mysql.sock

[mysqld]

port = 3306

socket = /storage/fioa/mysql3306/data/mysql.sock

server-id = 100011

replicate_ignore_db = test

skip-name-resolve

lower_case_table_names = 1

character-set-server = utf8

replicate_wild_ignore_table = dbmon.%,sys.%,zabbix.%,mysql.%,test.%,bbs.%,slow_query_log.%,

tmpdir = /storage/fioa/mysql3306/mysql-tmpdir

key_buffer_size = 64M

sort_buffer_size = 8M

read_buffer_size = 8M

read_rnd_buffer_size = 8M

query_cache_type = 0

max_allowed_packet = 628M

myisam_sort_buffer_size = 128M

tmp_table_size = 1024M

```
join_buffer_size          = 16M
back_log                  = 500
open_files_limit          = 6000
innodb_open_files        = 4096
table_open_cache          = 768
thread_cache_size         = 12
max_connections           = 6000
thread_concurrency        = 48
datadir                   = /storage/fioa/mysql3306/data
tmpdir                    = /storage/fioa/mysql3306/mysql-tmpdir
slow_query_log            = 1
long_query_time           = 1
slow_query_log_file       = /storage/fioa/mysql3306/mysql-slow.log
log-error                  = /storage/fioa/mysql3306/mysql-error.log
log-warnings
sysdate-is-now
###replication  setting
log-slave-updates
sync_binlog = 0
binlog_format              = row
log_bin_trust_function_creators = 1
log-bin                    = /storage/fioa/mysql3306/binlog/mysql-bin
log-slave-updates
relay_log_purge            = 0
relay-log                  = /storage/fioa/mysql3306/binlog/mysql-relay-bin
relay-log-index            = /storage/fioa/mysql3306/binlog/mysql-relay-bin.index
```

```
relay_log_info_file          = /storage/fioa/mysql3306/binlog/relay.info
master-info-file             = /storage/fioa/mysql3306/binlog/master.info
#innodb setting
innodb_data_home_dir         = /storage/fioa/mysql3306/data
innodb_data_file_path         = ibdata0:1024M;ibdata1:1024M;ibdata2:1024M;ibdata3:1024M:autoextend
innodb_log_group_home_dir    = /storage/fioa/mysql3306/data
innodb_log_files_in_group    = 2
innodb_log_file_size         = 512M
innodb_buffer_pool_size      = 65536M
innodb_log_buffer_size       = 128M
innodb_flush_log_at_trx_commit = 2
innodb_flush_method = O_DIRECT
innodb_thread_concurrency    = 6
innodb_file_per_table        = 1
transaction-isolation        = READ-COMMITTED
[mysqldump]
quick
max_allowed_packet = 128M
[myisamchk]
key_buffer_size = 256M
sort_buffer_size = 256M
read_buffer = 4M
write_buffer = 4M
[mysql]
max_allowed_packet      = 128M
default-character-set = utf8
```

```
no_auto_rehash
prompt          = "\\R:\\m:\\s \\d> "
pager           = "more"
```

TokuDB :

```
[client]
port          = 3306
socket        = /storage/fioa/mysql3306/data/mysql.sock
[mysqld]
port          = 3306
socket        = /storage/fioa/mysql3306/data/mysql.sock
server-id     = 100011
replicate_ignore_db = test
skip-name-resolve
lower_case_table_names = 1
character-set-server = utf8
plugin-load=ha_tokudb
tokudb_cache_size=64G
tokudb_directio=1
tokudb_commit_sync = 0
replicate_wild_ignore_table = dbmon.%,sys.%,zabbix.%,mysql.%,test.%,bbs.%,slow_query_log.%
plugin-load=ha_tokudb
tmpdir        = /storage/fioa/mysql3306/mysql-tmpdir
```

key_buffer_size	= 64M
sort_buffer_size	= 8M
read_buffer_size	= 8M
read_rnd_buffer_size	= 8M
query_cache_type	= 0
max_allowed_packet	= 628M
myisam_sort_buffer_size	= 128M
tmp_table_size	= 1024M
join_buffer_size	= 16M
back_log	= 500
open_files_limit	= 6000
innodb_open_files	= 4096
table_open_cache	= 768
thread_cache_size	= 12
max_connections	= 6000
thread_concurrency	= 48
datadir	= /storage/fioa/mysql3306/data
tmpdir	= /storage/fioa/mysql3306/mysql-tmpdir
slow_query_log	= 1
long_query_time	= 1
slow_query_log_file	= /storage/fioa/mysql3306/mysql-slow.log
log-error	= /storage/fioa/mysql3306/mysql-error.log
log-warnings	
sysdate-is-now	
###replication	setting
log-slave-updates	

```
sync_binlog = 0
binlog_format = row
log_bin_trust_function_creators = 1
log-bin = /storage/fioa/mysql3306/binlog/mysql-bin
log-slave-updates
relay_log_purge = 0
relay-log = /storage/fioa/mysql3306/binlog/mysql-relay-bin
relay-log-index = /storage/fioa/mysql3306/binlog/mysql-relay-bin.index
relay_log_info_file = /storage/fioa/mysql3306/binlog/relay.info
master-info-file = /storage/fioa/mysql3306/binlog/master.info
#innodb setting
innodb_data_home_dir = /storage/fioa/mysql3306/data
innodb_data_file_path = ibdata0:1024M;ibdata1:1024M;ibdata2:1024M;ibdata3:1024M:autoextend
innodb_log_group_home_dir = /storage/fioa/mysql3306/data
innodb_log_files_in_group = 2
innodb_log_file_size = 512M
innodb_flush_method = O_DIRECT
innodb_buffer_pool_size = 65536M
innodb_log_buffer_size = 128M
innodb_flush_log_at_trx_commit = 2
innodb_thread_concurrency = 6
innodb_file_per_table = 1
transaction-isolation = READ-COMMITTED
[mysqldump]
quick
max_allowed_packet = 128M
```

```
[myisamchk]
key_buffer_size = 256M
sort_buffer_size = 256M
read_buffer = 4M
write_buffer = 4M
[mysql]
max_allowed_packet = 128M
default-character-set = utf8
no_auto_rehash
prompt = "\\R:\\m:\\s \\d> "
pager = "more"
```

We create 1000 DWs first to test on both InnoDB and TokuDB engine –separate test from 32 threads to 512 threads

TpmC result on different threads

InnoDB Threads	32	64	128	256	512	1024
TpmC	46376.699	48921.6	47090.801	41429.398	39957.102	33683.102
TokuDB Threads	32	64	128	256	512	1024
TpmC	41694.898	30663.7	22092	13243.9	N/A	N/A

TokuDB does not support foreign keys, so we add indexes for queries, even so in a heavy-load environment TokuDB still has too many system locks that result in a low TpmC.

In this test, TokuDB engine got a poor performance (TpmC decreases rapidly, many connections timeout appear due to system locks in MySQL when threads up to 512).

Look at total size of these two engines:

InnoDB tpcc1000 DB:

```
MariaDB [(none)]> SELECT count(*) tables,
-> concat(round(sum(table_rows)/1000000,2),'M') rows,
-> concat(round(sum(data_length)/(1024*1024*1024),2),'G') data,
-> concat(round(sum(index_length)/(1024*1024*1024),2),'G') idx,
l_size, concat(round(sum(data_length+index_length)/(1024*1024*1024),2),'G') tota
-> round(sum(index_length)/sum(data_length),2) idxfrac
-> FROM information_schema.TABLES WHERE TABLE_SCHEMA='tpcc1000';
```

tables	rows	data	idx	total_size	idxfrac
13	503.60M	80.95G	16.34G	97.29G	0.20

TokuDB tpcc1000 DB:

```

MariaDB [(none)]> SELECT
->   table_schema,
->   table_name,
->   concat(round(sum(bt_size_allocated)/(1024*1024*1024),2),'G') total_size
-> FROM
->   information_schema.`TokuDB_fractal_tree_info`
-> WHERE table_schema = 'tpcc1000' ;
+-----+-----+-----+
| table_schema | table_name | total_size |
+-----+-----+-----+
| tpcc1000    | customer  | 58.88G    |
+-----+-----+-----+
1 row in set (0.01 sec)

```

Simple test table mytest10 size:

```

MariaDB [tpcc1000]> SELECT count(*) tables,
-> concat(round(sum(table_rows)/1000000,2),'M') rows,
-> concat(round(sum(data_length)/(1024*1024*1024),2),'G') data,
-> concat(round(sum(index_length)/(1024*1024*1024),2),'G') idx,
-> concat(round(sum(data_length+index_length)/(1024*1024*1024),2),'G') total_size,
-> round(sum(index_length)/sum(data_length),2) idxfrac
-> FROM information_schema.TABLES WHERE TABLE_SCHEMA='tpcc1000' and table_name='mytest10';
+-----+-----+-----+-----+-----+-----+
| tables | rows  | data  | idx   | total_size | idxfrac |
+-----+-----+-----+-----+-----+-----+
| 1      | 26.23M | 2.82G | 0.58G | 3.40G      | 0.20    |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

MariaDB [tpcc1000]> alter table mytest10 engine=tokudb;
Query OK, 24546094 rows affected (3 min 8.79 sec)
Records: 24546094 Duplicates: 0 Warnings: 0

MariaDB [tpcc1000]> SELECT
-> table_schema,
-> table_name,
-> SUM(
-> ROUND(bt_size_allocated / 1024 / 1024 / 1024, 2)
-> ) AS table_size_gb
-> FROM
-> information_schema.`TokuDB_fractal_tree_info`
-> WHERE table_schema = 'tpcc1000'
-> AND table_name = 'mytest10' ;
+-----+-----+-----+
| table_schema | table_name | table_size_gb |
+-----+-----+-----+
| tpcc1000     | mytest10   | 1.45          |
+-----+-----+-----+

```

Compression ratio of these two engines is nearly 2:1

Mydbtest performance in different threads

We create four tables for this test on four modes (100% insert, 50%insert+50%select, 50%insert+50%update, mix)

mode \ threads	256	512
insert (innodb)	Summary: exec=23768/s, qtps=47669/s	Summary: exec=13319/s, qtps=54116/s
insert (tokudb)	Summary: exec=23640/s, qtps=47415/s	Summary: exec=12818/s, qtps=51969/s
50insert+50sel (innodb)	Summary: exec=16606/s, qtps=66803/s	Summary: exec=16418/s, qtps=66461/s
50insert+50sel (tokudb)	Summary: exec=15324/s, qtps=61667/s	Summary: exec=15317/s, qtps=61979/s
50insert+50upd (innodb)	Summary: exec=10078/s, qtps=40657/s	Summary: exec=10016/s, qtps=40842/s
50insert+50upd (tokudb)	Summary: exec=9890/s, qtps=39890/s	Summary: exec=9843/s, qtps=40155/s
mix mode (innodb)	Summary: exec=7573/s, qtps=61504/s	Summary: exec=7728/s, qtps=63939/s
mix mode (tokudb)	Summary: exec=6788/s, qtps=55262/s	Summary: exec=6833/s, qtps=56753/s

InnoDB still has a better performance than TokuDB when use simple queries to avoid MySQL locks.

option

user xxxxx/xxxx@10.128.6.35:3306:tpcc1000

name tpcc1000

time 10m

log toku_insert100%_512_parallel__%p.log

declare

a int 1 200000000

b int 40000 6000000000

c CHAR 1 20

d CHAR 1 50

e char 1 4

f CHAR 1 5

g char 1 6

h char 3 15

j int 35 555555555

h char 3 15

k date -90 -60

l int 1 49999999

begin

insert into mytest10 values(:a,:b,:c,:d,:e,:f,:g,:h,:j);

--- /*insert into mytest values (:a,:c,:d);*/

--- /*select * from mytest where a=:a;*/

--- /*insert into mytest2 values (:a,:b,:k,:h);*/

--- /*update mytest set a=a-1 where a=:a;*/

--- /*select * from mytest2 where k=:k;*/

end

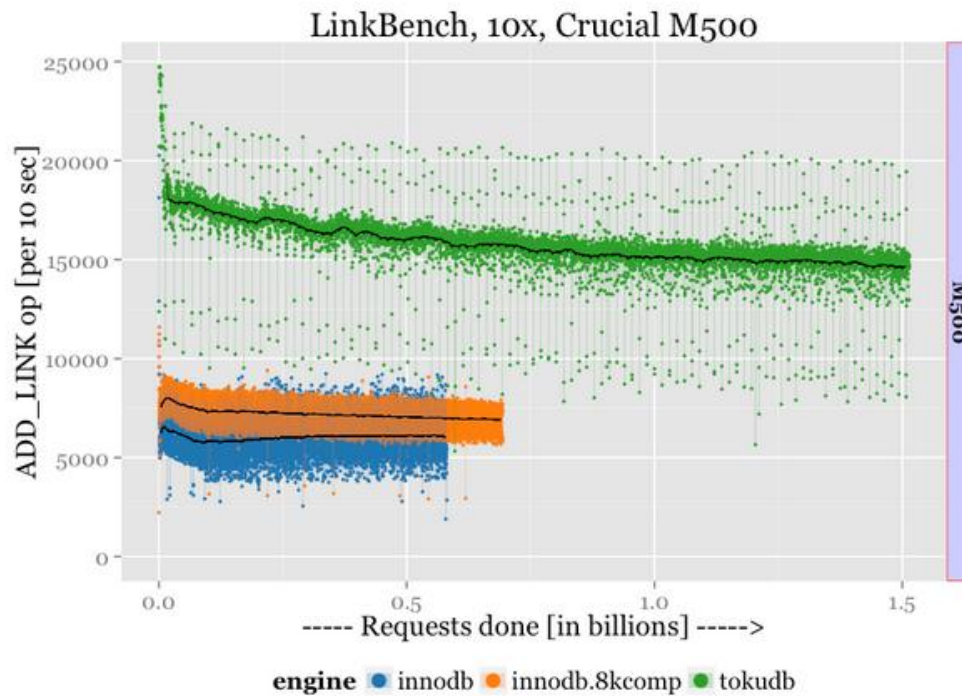
Conclusion:

Reference: <https://www.percona.com/blog/2015/07/24/innodb-vs-tokudb-in-linkbench-benchmark/>

Though, it worth remembering, that:

- *On a fast expensive storage, TokuDB provides a better compression, which allows to store more data in limited capacity*
- *TokuDB still writes two time less than InnoDB, that mean twice longer lifetime for SSD (still expensive).*

Also looking at the results, I can make the conclusion that InnoDB compression is inefficient in its implementation, as it is not able to get benefits: first, from doing less reads (well, it helps to get better than uncompressed InnoDB, but not much); and, second, from a fast storage.



In this picture, TokuDB has a better performance on IO slow storage like SATA SAS disk (Author uses M500 SSD disk for this test)

For us data quality is more important than storage cost. We have already used SSD&PCIe cards for all database system. TokuDB has big advantage for its high compression ratio. I think it's more suitable for log system rather than core system.