MySQL大数据处理方案

吴炳锡 - 知数堂培训联合创始人

关于我

- 知数堂培训联合创始人， MySQL中国用户组主席
- QQ/微信：82565387
- 10年+MySQL从业经验，专注MySQL架构，多IDC设计，高可用，高性能相关技术分析及实践，目前从事MySQL培训及企业MySQL相关服务。

agenda

- 现在行业数据特点
- MySQL在大数据面临的挑战
- 高速写入环境MySQL优化
- MySQL对于大表拆分设计
- TokuDB无痛上线架构
- 总结

现在行业数据的特点

- 数据量比较大
  - 大库比较多，上T的MySQL库比较长常见
  - 写入速度比较高
    - 交易型日志相关
    - 用户操作日志相关
  - 数据非常重要
    - 很多数据是用于审计

现在行业数据的特点

分库分表
高端硬件

Hbase
MongoDB
大数据方案

现在行业数据的特点

- 对面对大数据在MySQL中使用
  - 分库分表
  - 基于队例入库
  - 基于Redis缓存
  - 数据中间层
  - 大数据平台：Hbase &MongoDB…

agenda

- 现在行业数据特点
- MySQL在大数据面临的挑战
- 高速写入环境MySQL优化
- MySQL对于大表拆分设计
- TokuDB无痛上线架构
- 总结

高速写入环境MySQL优化

- 引擎选择
  - Innodb & TokuDB & 其它
  - 禁止使用MyISAM
- 配置优化
- 硬件优化
- SQL优化

高速写入环境MySQL优化

- 引擎选择测试
  - 环境： 某云高性能主机 16 core , 16G RAM , 1T空间 Ubuntu系统
  - 对比版本： Percona-Server-5.7.17 （Innodb, Tokudb), Xenon-Tokudb
  - 对比一，分别在bufffer最大的情况下 8G进行一个测试
  - 对比二，在使用200M的buffer的情况写入情况

高速写入环境MySQL优化

- 测试版本介绍
  - Innodb 使用 Percona 5.7.17
  - TokuDB使用Percona 5.7.17
  - Xenon-TokuDB 基于Percona 5.7下的TokuDB优化后的TokuDB分支 https://github.com/XeLabs/tokudb

高速写入环境MySQL优化

- 关于TokuDB
  - https://www.percona.com/software/mysql-database/percona-tokudb
- 关于Xenon TokuDB
  - https://github.com/XeLabs/tokudb
    - 支持xtrabackup直接备份，定制改良过
    - 加入ZSTD压缩算法（From MyRocks）
    - 支持binlog group commit
    - 引入性能计数器 show engine tokudb status ，更多选项。

主要参数

- innodb_buffer_pool_size = 8G
- innodb_buffer_pool_instances = 1
- innodb_data_file_path = ibdata1:100M:autoextend
- innodb_flush_log_at_trx_commit = 2
- innodb_log_buffer_size = 8M
- innodb_log_file_size = 200M
- innodb_log_files_in_group = 3
- innodb_max_dirty_pages_pct = 50
- innodb_file_per_table = 1
- innodb_rollback_on_timeout
- innodb_io_capacity = 100

- loose_tokudb_cache_size=200M
- loose_tokudb_directio=ON
- loose_tokudb_fsync_log_period=1000
- tokudb_commit_sync=0

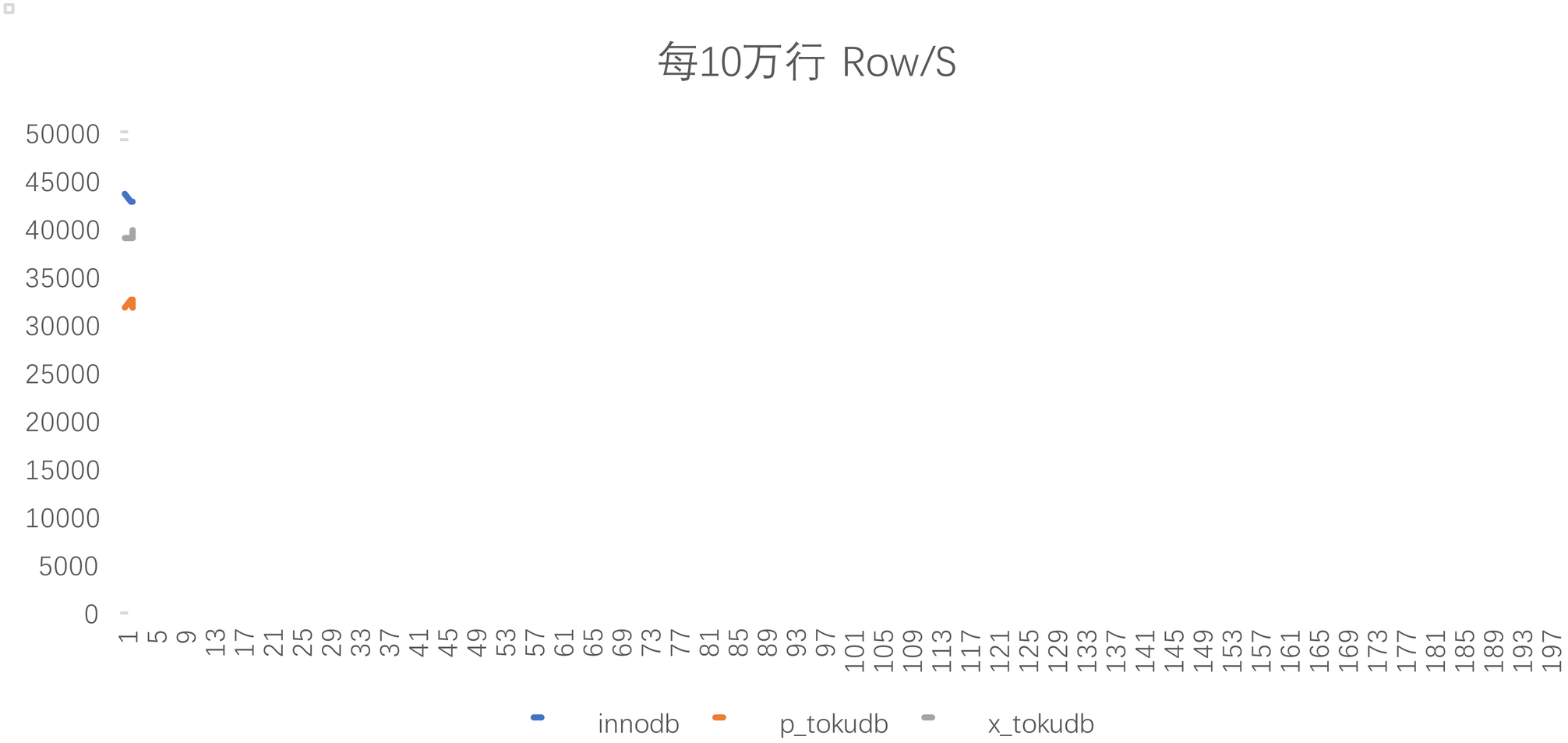
表结构

```
CREATE TABLE `wubx` (
  `transactionid` int(11) NOT NULL AUTO_INCREMENT,
  `dateandtime` datetime DEFAULT NULL,
  `cashregisterid` int(11) NOT NULL,
  `customerid` int(11) NOT NULL,
  `productid` int(11) NOT NULL,
  `price` float NOT NULL,
  `data` varchar(4000) DEFAULT NULL,
  PRIMARY KEY (`transactionid`),
  KEY `marketsegment` (`price`,`customerid`),
  KEY `registersegment` (`cashregisterid`,`price`,`customerid`),
  KEY `pdc` (`price`,`dateandtime`,`customerid`)
) ENGINE=TokuDB AUTO_INCREMENT=20000001 DEFAULT CHARSET=utf8
```

结果情况

版本	运行时间（S）	平均写入速度（Row/s）	文件大小
Percona Tokudb	934	21410.4	1232M
Percona Innodb	709	28210.5	3.2G
Xenon Tokudb	707	28289.8	1356M

结果情况

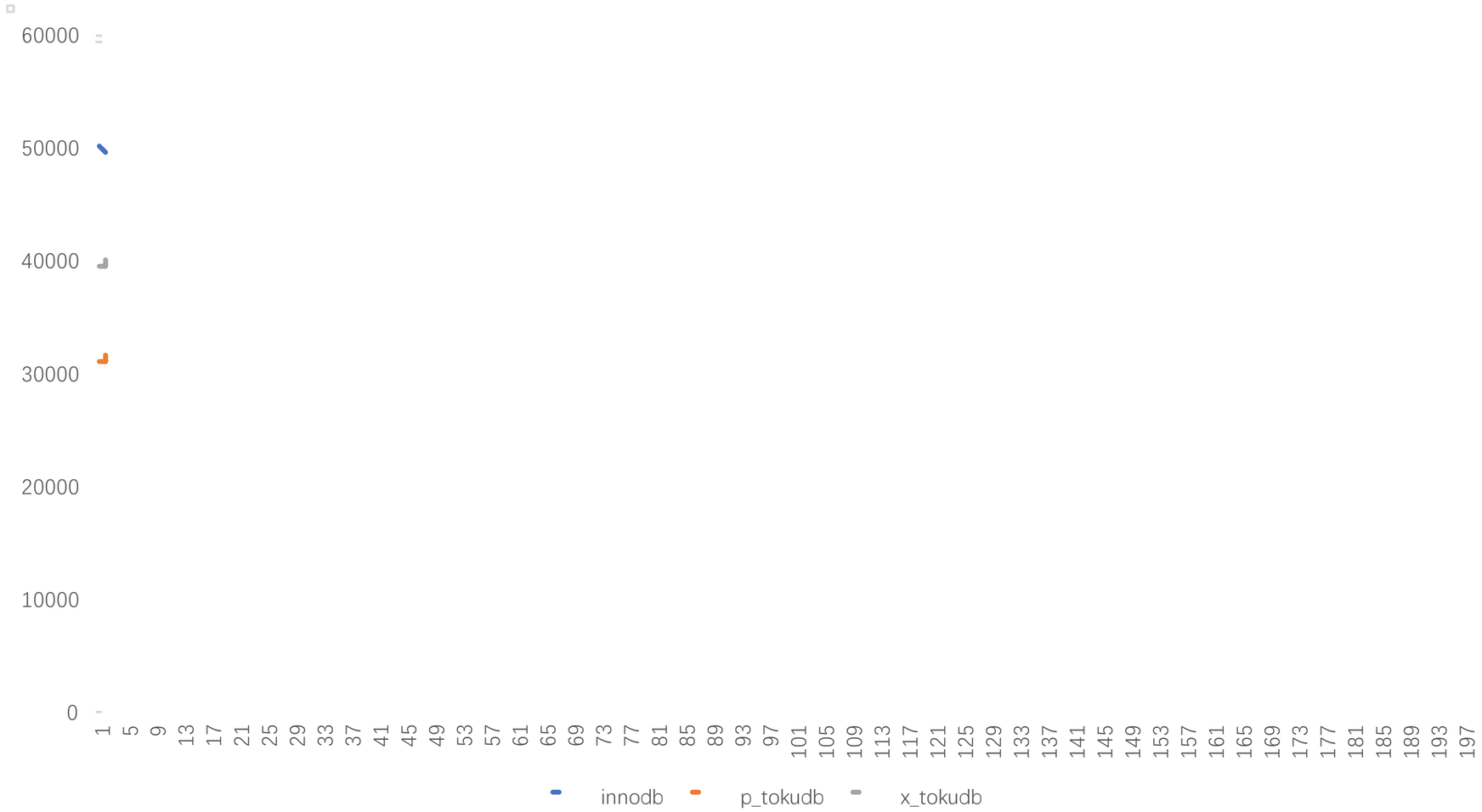

每10万行 Row/S
50000
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89 93 97 101 105 109 113 117 121 125 129 133 137 141 145 149 153 157 161 165 169 173 177 181 185 189 193 197
innodb
p_tokudb
x_tokudb

测试二 bp=200M

- innodb_buffer_pool_size=200M
- tokudb_cache_size = 200M

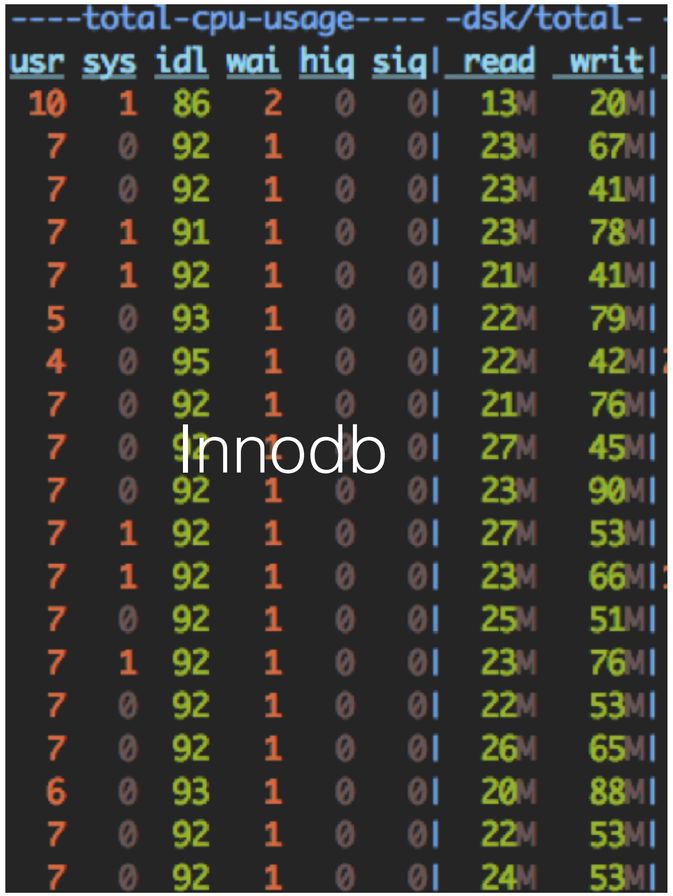
版本	运行时间（S）	row/s	大小
innodb	2250	8888.7	3.1G
p_tokudb	976	20491.8	1.48G
x_tokudb	810	24673.2	1.46G

对比


60000
50000
40000
30000
20000
10000
0
1
5
9
13
17
21
25
29
33
37
41
45
49
53
57
61
65
69
73
77
81
85
89
93
97
101
105
109
113
117
121
125
129
133
137
141
145
149
153
157
161
165
169
173
177
181
185
189
193
197
innodb
p_tokudb
x_tokudb

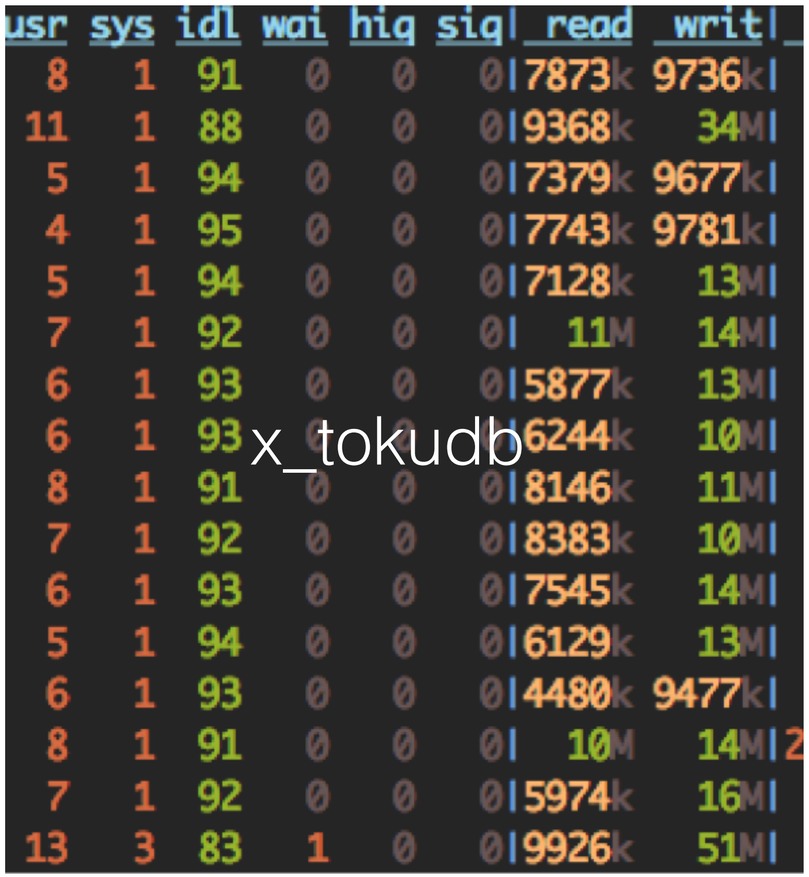
IO情况对比

Innodb



----total-cpu-usage----						-dsk/total-	
usr	sys	idl	wai	hiq	siq	read	writ
10	1	86	2	0	0	13M	20M
7	0	92	1	0	0	23M	67M
7	0	92	1	0	0	23M	41M
7	1	91	1	0	0	23M	78M
7	1	92	1	0	0	21M	41M
5	0	93	1	0	0	22M	79M
4	0	95	1	0	0	22M	42M
7	0	92	1	0	0	21M	76M
7	0	91	1	0	0	27M	45M
7	0	92	1	0	0	23M	90M
7	1	92	1	0	0	27M	53M
7	1	92	1	0	0	23M	66M
7	0	92	1	0	0	25M	51M
7	1	92	1	0	0	23M	76M
7	0	92	1	0	0	22M	53M
7	0	92	1	0	0	26M	65M
6	0	93	1	0	0	20M	88M
7	0	92	1	0	0	22M	53M
7	0	92	1	0	0	24M	53M

x_tokudb



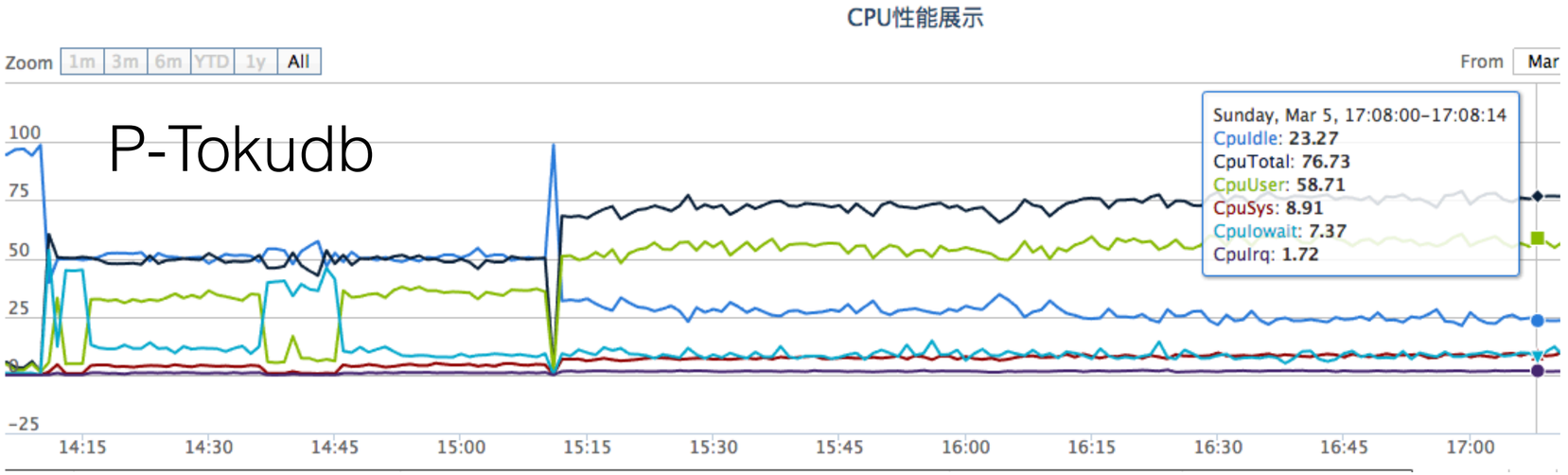
usr	sys	idl	wai	hiq	siq	read	writ
8	1	91	0	0	0	7873k	9736k
11	1	88	0	0	0	9368k	34M
5	1	94	0	0	0	7379k	9677k
4	1	95	0	0	0	7743k	9781k
5	1	94	0	0	0	7128k	13M
7	1	92	0	0	0	11M	14M
6	1	93	0	0	0	5877k	13M
6	1	93	0	0	0	6244k	10M
8	1	91	0	0	0	8146k	11M
7	1	92	0	0	0	8383k	10M
6	1	93	0	0	0	7545k	14M
5	1	94	0	0	0	6129k	13M
6	1	93	0	0	0	4480k	9477k
8	1	91	0	0	0	10M	14M
7	1	92	0	0	0	5974k	16M
13	3	83	1	0	0	9926k	51M

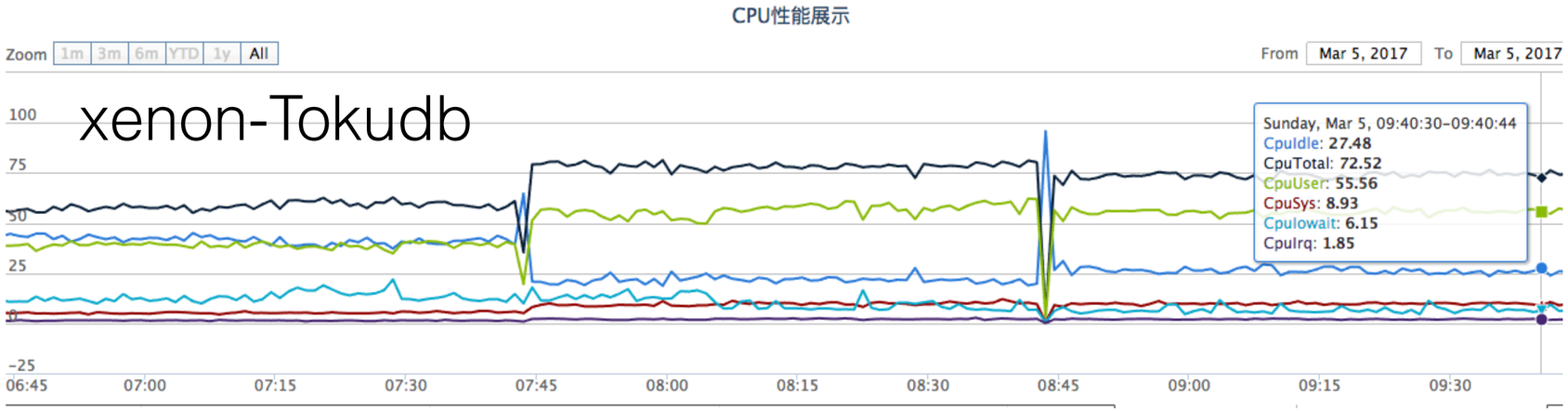
基于Sysbench测试

- 场景：
  - tokudb_cache_size = 8G
  - 8张表，每张表5千万数据 单表大小在 5.2G
  - 连接线程16，32，64三种情况

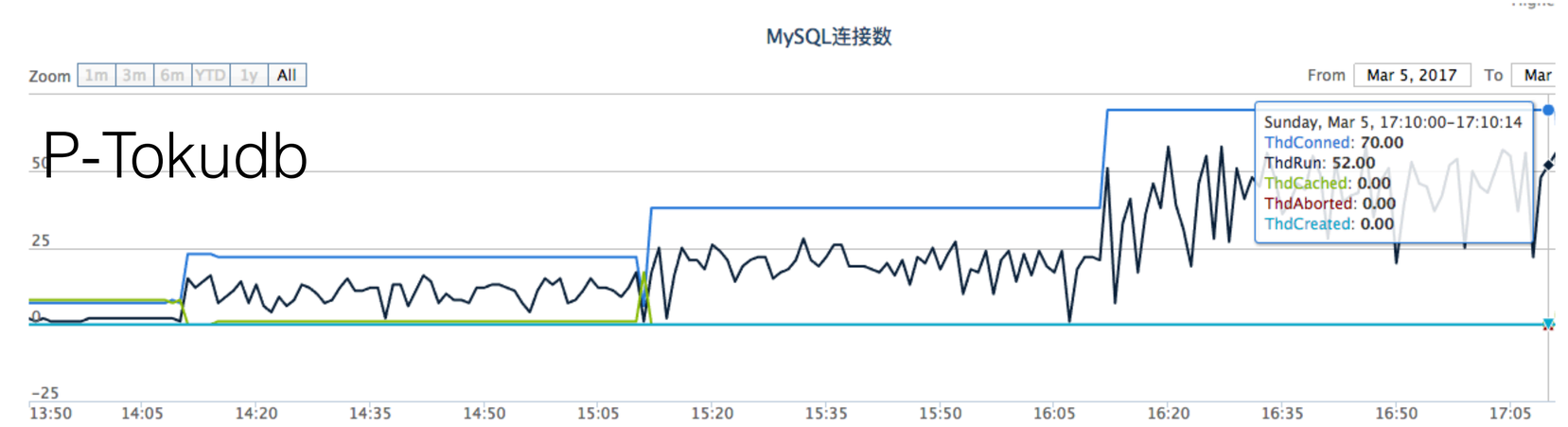
版本	thd	TPS	QPS
Percona tokudb	16	675.32	13506.43
	32	917.78	18355.52
	64	915.69	18313.77
Xenon tokudb	16	1159.10	23182
	32	1298.40	25968
	64	1220.09	24401.75

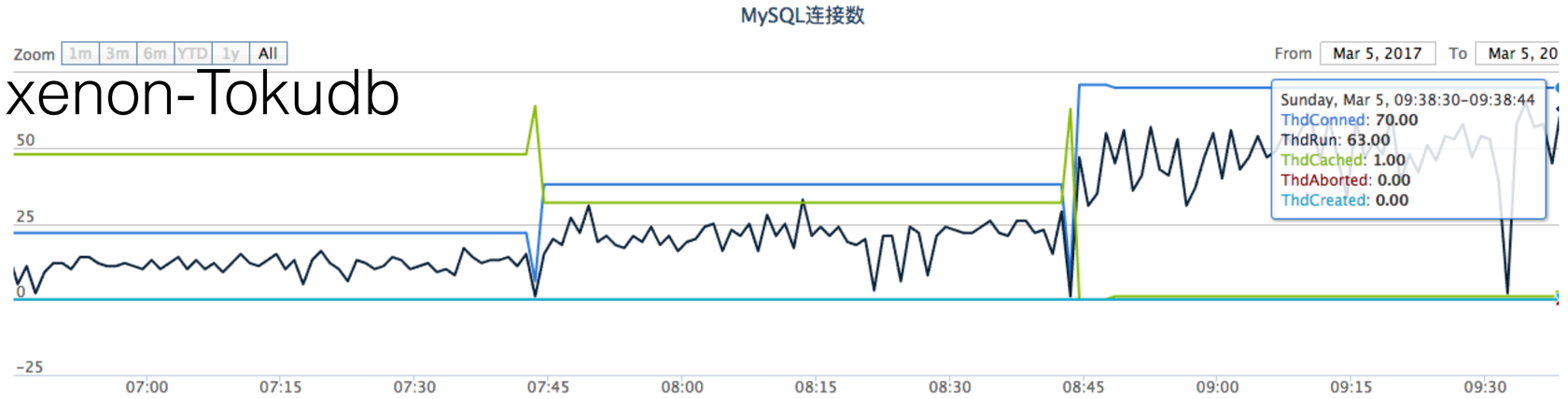
CPU 对比图


CPU性能展示
Zoom 1m 3m 6m YTD 1y All
From Mar
P-Tokudb
Sunday, Mar 5, 17:08:00-17:08:14
CpuIdle: 23.27
CpuTotal: 76.73
CpuUser: 58.71
CpuSys: 8.91
CpuIowait: 7.37
CpuIrq: 1.72
100
75
50
25
0
-25
14:15
14:30
14:45
15:00
15:15
15:30
15:45
16:00
16:15
16:30
16:45
17:00


CPU性能展示
Zoom 1m 3m 6m YTD 1y All
From Mar 5, 2017
To Mar 5, 2017
xenon-Tokudb
Sunday, Mar 5, 09:40:30-09:40:44
CpuIdle: 27.48
CpuTotal: 72.52
CpuUser: 55.56
CpuSys: 8.93
CpuIowait: 6.15
CpuIrq: 1.85
100
75
50
25
0
-25
06:45
07:00
07:15
07:30
07:45
08:00
08:15
08:30
08:45
09:00
09:15
09:30

连接数 对比图


MySQL连接数
Zoom 1m 3m 6m YTD 1y All
From Mar 5, 2017 To Mar
P-Tokudb
Sunday, Mar 5, 17:10:00-17:10:14
ThdConned: 70.00
ThdRun: 52.00
ThdCached: 0.00
ThdAborted: 0.00
ThdCreated: 0.00
50
25
0
-25
13:50 14:05 14:20 14:35 14:50 15:05 15:20 15:35 15:50 16:05 16:20 16:35 16:50 17:05


MySQL连接数
Zoom 1m 3m 6m YTD 1y All
From Mar 5, 2017 To Mar 5, 20
xenon-Tokudb
Sunday, Mar 5, 09:38:30-09:38:44
ThdConned: 70.00
ThdRun: 63.00
ThdCached: 1.00
ThdAborted: 0.00
ThdCreated: 0.00
50
25
0
-25
07:00 07:15 07:30 07:45 08:00 08:15 08:30 08:45 09:00 09:15 09:30

基于测试对比总结

- 从测试数据上看
  - 从结果上看32个连接是性能最高,xenon-tokudb是percona-tokudb的140%
  - Xenon-tokudb和Percona-tokudb 使用CPU接近的情况下，性能高出40%。

    长期来看，xenon-tokudb会比官方tokudb更节点一点CPU
  - 在IO方面， 从实际的测试中看同样的IO， xenon-tokudb更节点一点IO。 在硬件比较差的环境下，更显的有优势

高速写入环境MySQL优化

- 引擎选择
- 配置优化
- 硬件优化
- SQL优化

高速写入环境MySQL优化

- 配置优化
  - Buffer Cache size
  - Directio
  - 日志提交：双1
  - binary log：binlog_sync

高速写入环境MySQL优化

- 引擎选择
- 配置优化
- 硬件优化
- SQL优化

高速写入环境MySQL优化

- 硬件优化
  - SSD & PCI-E
  - STAT
  - Raid & No Raid
  - 高频 & more Core

高速写入环境MySQL优化

- 引擎选择
- 配置优化
- 硬件优化
- SQL优化

高速写入环境MySQL优化

- SQL优化
  - 批量写入
  - 导步队列入库
  - 基于分表的多表并发写入

agenda

- 现在行业数据特点
- MySQL在大数据面临的挑战
- 高速写入环境MySQL优化
- MySQL对于大表拆分设计
- TokuDB无痛上线架构
- 总结

对于大表拆分设计

- 基于并发度高的业务
  - 分库分表
- 基于数据量大并发不大的业务
  - 基于分区设计
- 考虑热数据和归档数据设计
  - 在线热数据在半年左右，其它可以归档设计

对于大表拆分设计

- **举例：**
  - 用户表，半年左右一张表，基于Range分表分库
  - 订单日志数据： 在线热库保存半年左右的数据，半年前的数据每期向归档库迁移
  - 分层写入数据：订单表： 天表， 旬表， 月表，年度归档分区表，分级处理，利用空间去换时间

agenda

- 现在行业数据特点
- MySQL在大数据面临的挑战
- 高速写入环境MySQL优化
- MySQL对于大表拆分设计
- TokuDB无痛上线架构
- 总结

TokuDB无痛上线

- 构建多个归档库
  - Innodb & TokuDB并存，进行观查 到最后只有TokuDB
- 利用TokuDB的Replication的Read Free Replication 处理写入量大，从库跟不上的问题
- 最终可以使用TokuDB最初直接替换生产库，充分利用SSD，PCI-E小硬盘，发挥高性能，提供更高的效应能力。

TokuDB上线方案

- TokuDB在Replication中支持： Read Free Replication

Master:
server-id=
log-bin=
sync_binlog=on
binlog_format=row
binlog_row_iamge=FULL

slave:
server-id=
log-bin=
read_only=ON
binlog_format=row
master_info_repository=TABLE
sync_master_info=1
relay_log_info_repository=TABLE
sync_relay_log_info=1
slave_parallel_workers

TokuDB上线方案

- TokuDB成为主库的配置：

  loose_tokudb_cache_size=内存一半

  loose_tokudb_directio=ON

  loose_tokudb_fsync_log_period=1000

  loose-tokudb_commit_sync=1

TokuDB上线方案

- 备份方案：
  - https://github.com/XeLabs/tokudb-xtrabackup
  - 和使用Percona官方的xtrabackup一样，只是多了备份xenon tokudb的功能
  - 同时支持备分Innodb

agenda

- 现在行业数据特点
- MySQL在大数据面临的挑战
- 高速写入环境MySQL优化
- MySQL对于大表拆分设计
- TokuDB无痛上线架构
- 总结

总结

- TokuDB现在整体上非常好用。大业环境中也出现交多，成功案例较多。可以放心使用。
- 如果你的数据库已经超过TB级别或是你的SSD盘较小，想获得高效的性能也可以考虑TokuDB。
- 如果你的硬件很烂，又想存放大数据也可以考虑TokuDB
- 如果你的定期归档的数据，也可以考虑TokuDB

参考

- **手册（重点）： https://github.com/percona/tokudb-engine/wiki**
- Xenon-TokuDB： https://github.com/XeLabs/tokudb
- RDS TokuDB小手册 http://mysql.taobao.org/monthly/2015/04/02/
- Percona TokuDB： https://www.percona.com/software/mysql-database/percona-tokudb
- 云数据库 PetaData https://www.aliyun.com/product/petadata

谢谢，希望有所帮助