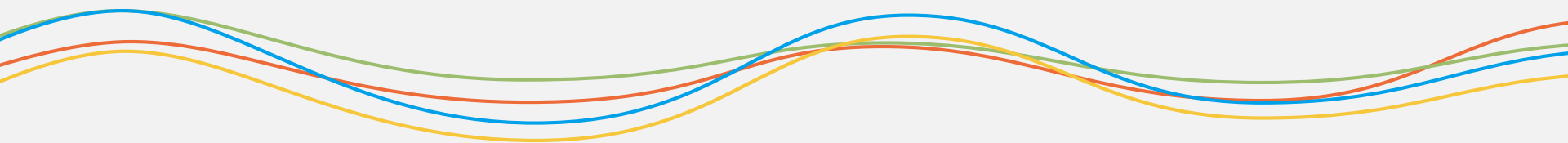


MySQL 5.7让优化更轻松

叶金荣



- 知数堂联合创始人
- Oracle MySQL ACE
- 公众号 老叶茶馆(imysql_wx)

MySQL 5.7搜索趋势



MySQL 5.5
Search term

MySQL 5.6
Search term

MySQL 5.7
Search term

+ Add comparison

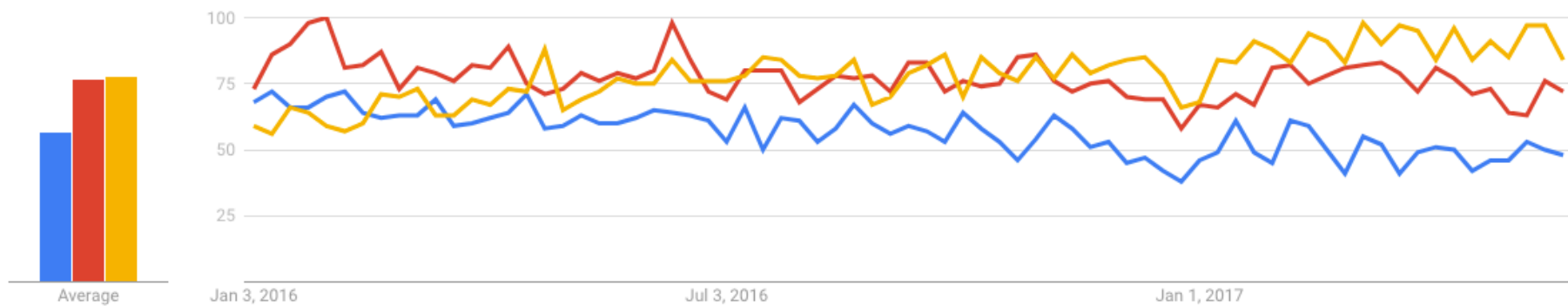
Worldwide ▼

1/1/16 - 5/27/17 ▼

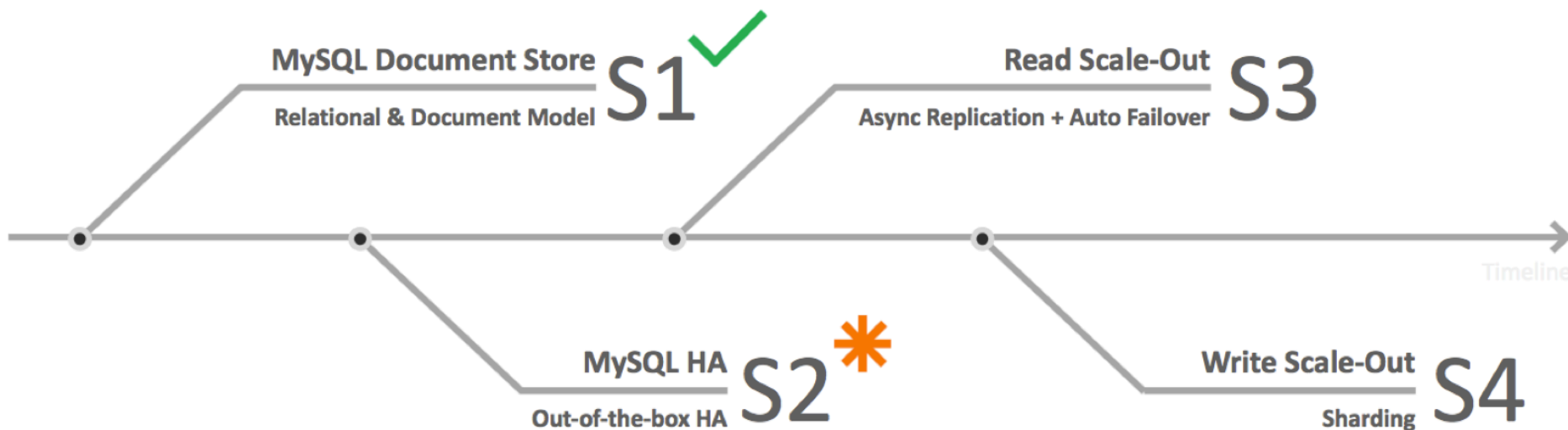
All categories ▼

Web Search ▼

Interest over time ?



The Road Ahead

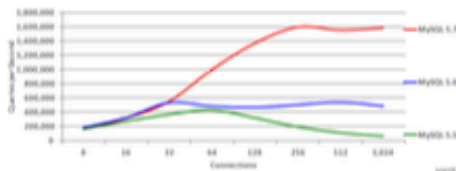


官方号称5.7比5.6性能提升3倍以上

MySQL 5.7 Sysbench Benchmark: **SQL** Point Selects / sec

3x Faster than MySQL 5.6

4x Faster than MySQL 5.5

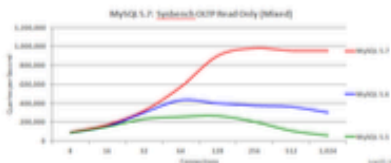


1,600,000 QPS

MySQL 5.7 Sysbench Benchmark: Mixed OLTP Read Only

3x Faster than MySQL 5.6

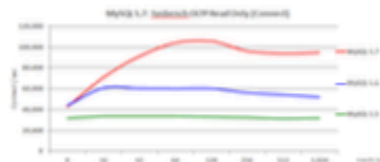
6x Faster than MySQL 5.5



Near 1M QPS

MySQL 5.7 Sysbench Benchmark: Connect / sec

82% Faster than MySQL 5.6



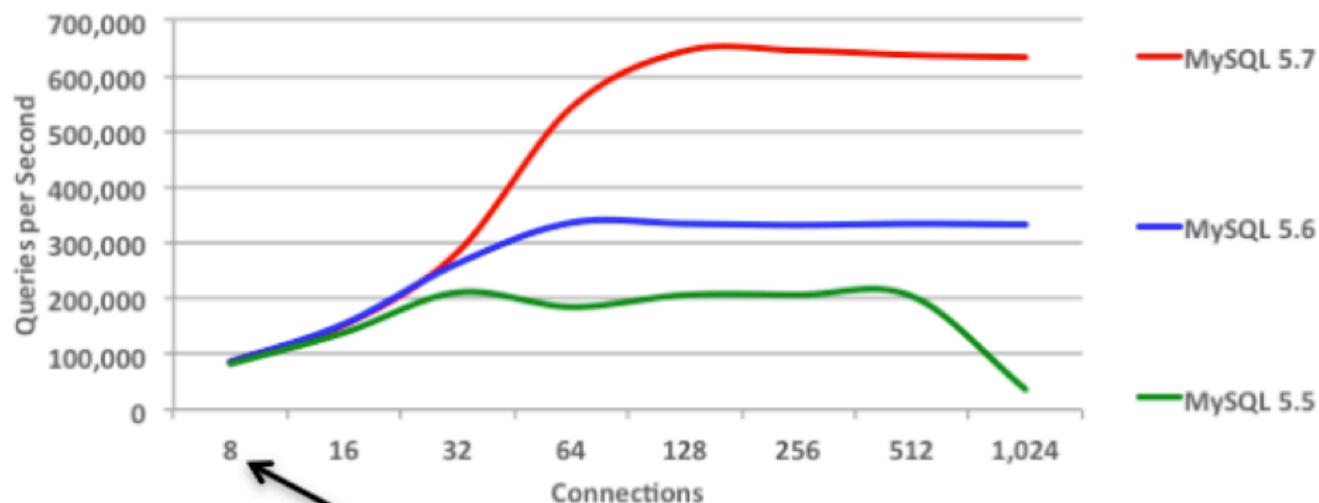
100K Connect / Sec

图片来源《Whats New in MySQL 5.7》by Geir Høydalsvik & Simon Mudd

645,000 QPS

2x Faster than MySQL 5.6
3x Faster than MySQL 5.5

MySQL 5.7: Sysbench Read Only (Point Select)



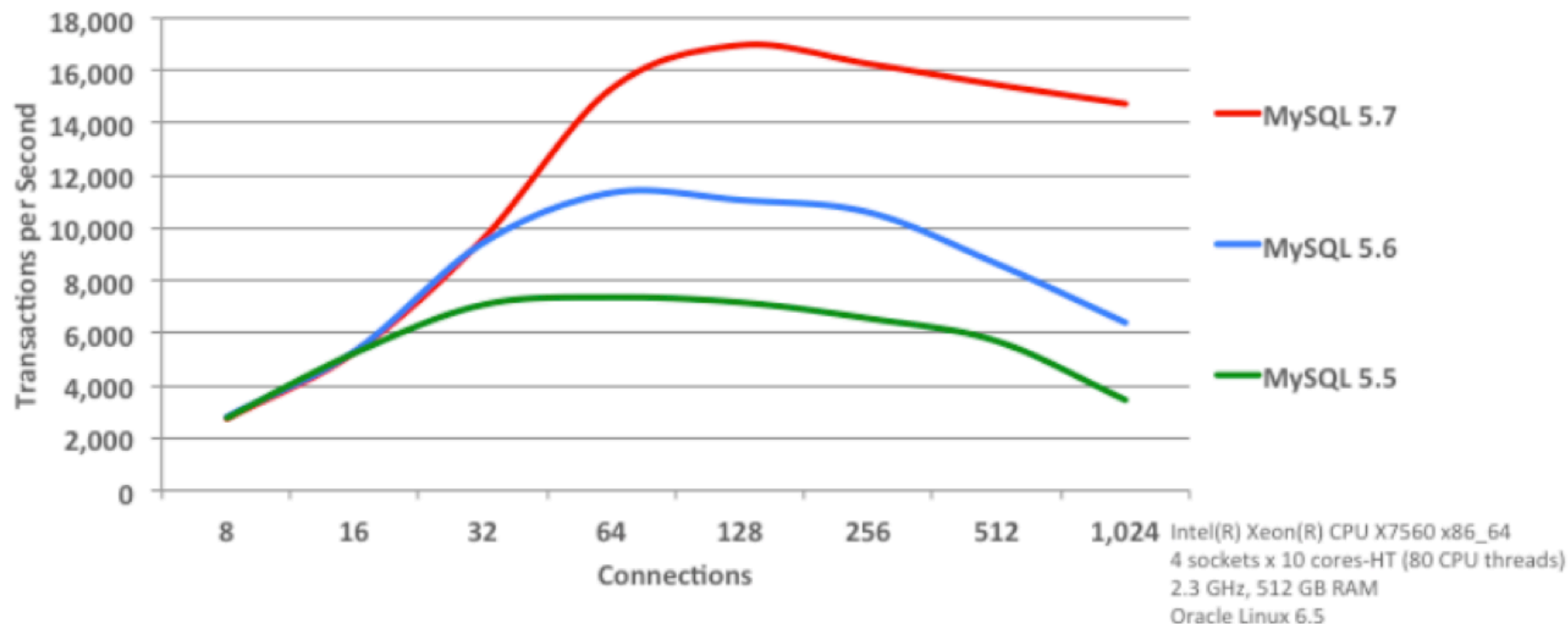
Starts with 8 Threads
What about 2-4 threads?

Intel(R) Xeon(R) CPU E7-4860 x86_64
4 sockets x 10 cores-HT (80 CPU threads)
2.3 GHz, 512 GB RAM
Oracle Linux 6.5

1.5x Faster than MySQL 5.6
2.5x Faster than MySQL 5.5

17,000 TPS

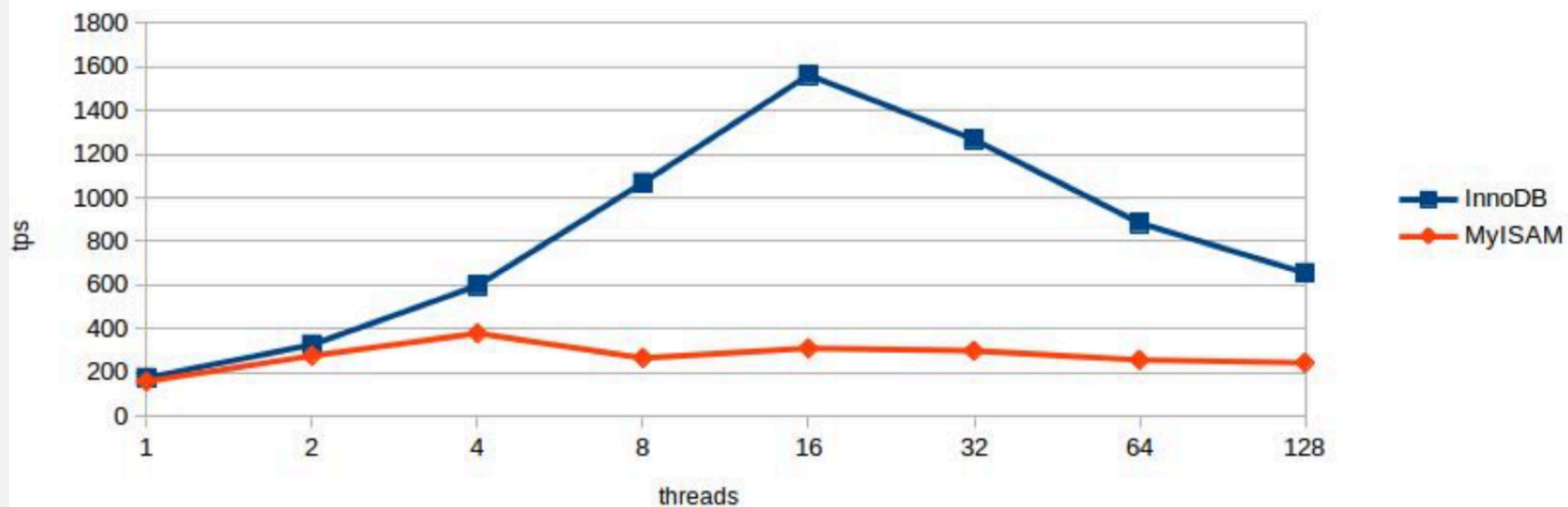
MySQL 5.7: Sysbench OLTP Read Write



是时候全面放弃MyISAM了

Feature	MyISAM	InnoDB
Full Text Indexes	yes	Since MySQL 5.6
Portable tables (tablespaces)	yes	Since MySQL 5.6
Spatial Indexes/RTREE (GIS)	yes	<i>Since MySQL 5.7</i>
Last update for table	yes	<i>Since MySQL 5.7</i> <i>(http://dev.mysql.com/worklog/task/?id=6658)</i>
Suitable for temp tables	yes	<i>Since MySQL 5.7</i> <i>Also complex selects uses InnoDB ondisk temp tables</i>
Fast count(*)	yes	<i>*Faster in MySQL 5.7 but does not store counter</i>

InnoDB vs MyISAM (as Optimizer SE)



InnoDB增强

1、Online buffer pool resize

- 由小改大几乎无影响；
- 由大改小需要释放部分内存，有一定代价。

resize innodb buffer pool from 64G to 16G

```
09:22:55.591669Z 0 [Note] InnoDB: Resizing buffer pool from 17179869184 to 1073741824.  
(unit=134217728)  
09:22:55.680836Z 0 [Note] InnoDB: disabled adaptive hash index.  
09:22:55.680864Z 0 [Note] InnoDB: buffer pool 0 : start to withdraw the last 491511  
blocks.  
09:22:55.765778Z 0 [Note] InnoDB: buffer pool 0 : withdrew 489812 blocks from free  
list. Tried to relocate 1698 pages (491510/491511).  
09:22:55.774492Z 0 [Note] InnoDB: buffer pool 0 : withdrew 0 blocks from free list.  
Tried to relocate 1 pages (491511/491511).  
...  
09:22:56.308997Z 0 [Note] InnoDB: buffer pool 1 : 60 chunks (491520 blocks) were freed.  
09:22:56.316258Z 0 [Note] InnoDB: buffer pool 0 : hash tables were resized.  
...  
09:22:56.393589Z 0 [Note] InnoDB: Resized hash tables at lock_sys, adaptive hash  
index, dictionary.  
09:22:56.393616Z 0 [Note] InnoDB: Completed to resize buffer pool from 17179869184 to  
1073741824.  
09:22:56.393628Z 0 [Note] InnoDB: Re-enabled adaptive hash index.
```

2、 VARCHAR in-place enlarge

- 255字节长度是个门槛；
- 不跨越255长度门槛即可在线调大；
- 没必要再为VARCHAR列预留更大长度；
- 缩短长度不能in-place。

- 成功

#enlarge c1 length from 30 to 50

```
mysql> ALTER TABLE t1 ALGORITHM=INPLACE,      modify  
email VARCHAR(50);
```

Query OK, 0 rows affected (0.68 sec)

- 失败

#enlarge c1 length from 30 to 300

```
mysql> alter table t1 ALGORITHM=INPLACE, modify  
char_col varchar(300) not null default '';
```

ERROR 1846 (0A000): ALGORITHM=INPLACE is not supported. Reason: Cannot change column type INPLACE. Try ALGORITHM=COPY.

3、临时表空间增强

- 使用独立表空间，不写redo、无change buffer；

4、支持在线清除undo log

- 避免undo log积累过多；

复制增强

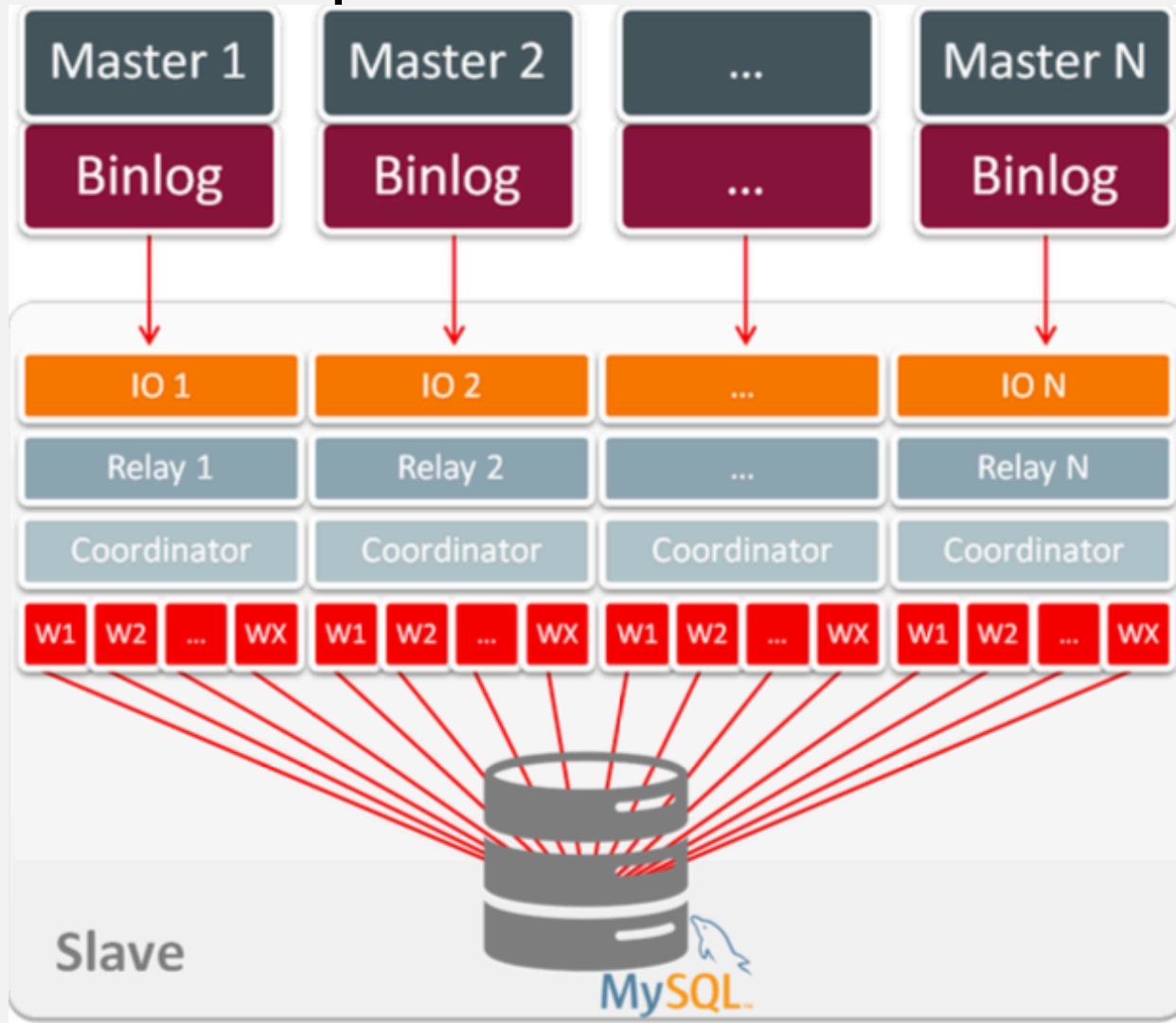
1、multi source replication/多源复制

- 支持多主一从，从多个master汇聚到一个slave；
- 特别适合数据集中分析、集中备份、异地容灾。

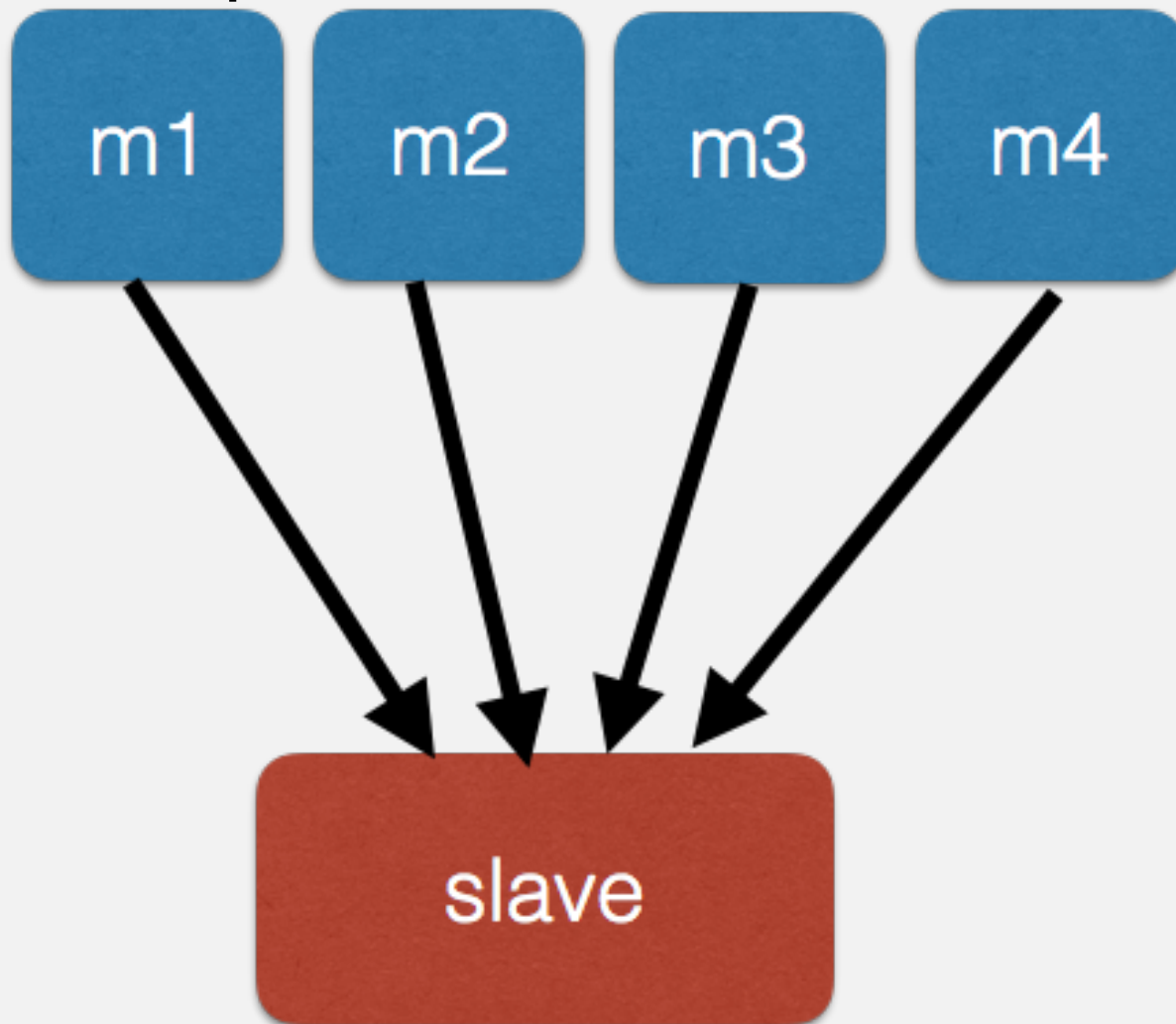
2、multi-threaded replication/并行复制

- 基于LOGICAL_CLOCK时序组提交的并行复制，同时进入PREPARE状态的事务都可以在SLAVE并行应用；
- 可解决除大事务外的大部分复制延迟问题。

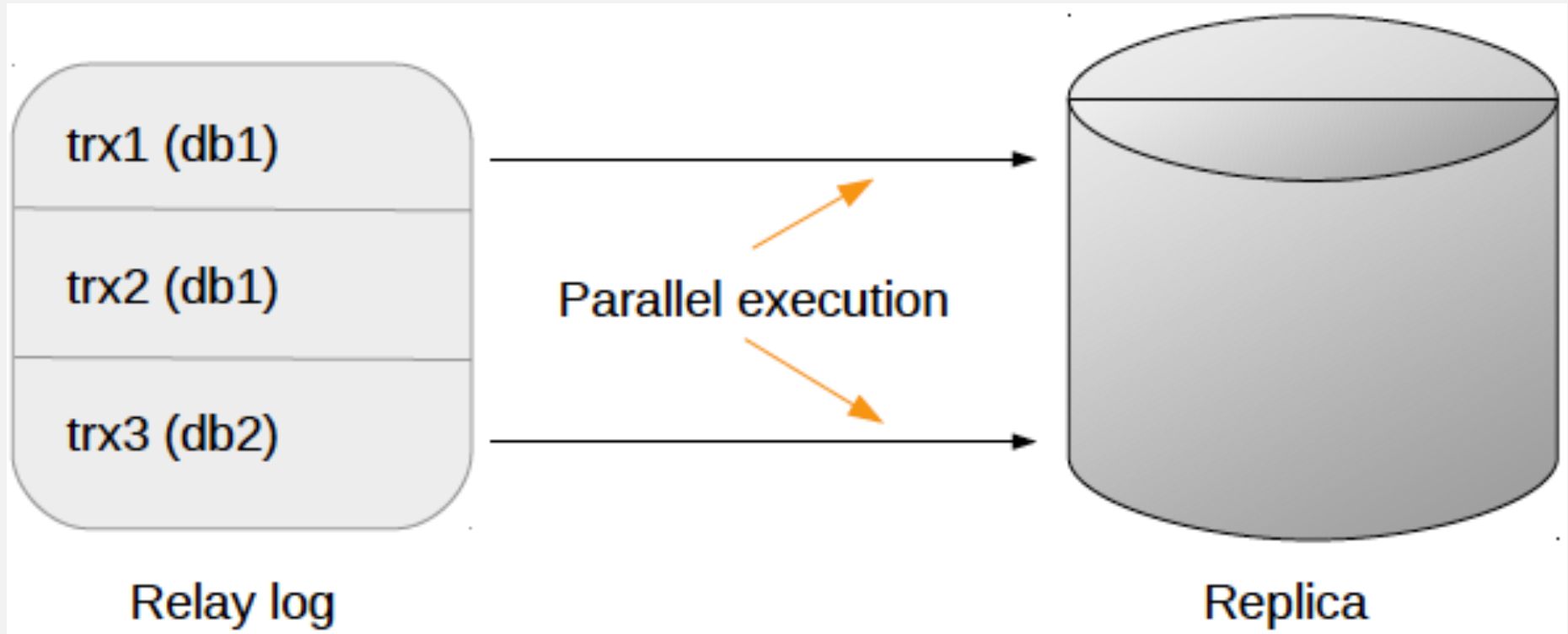
multi-threaded replication



multi source replication

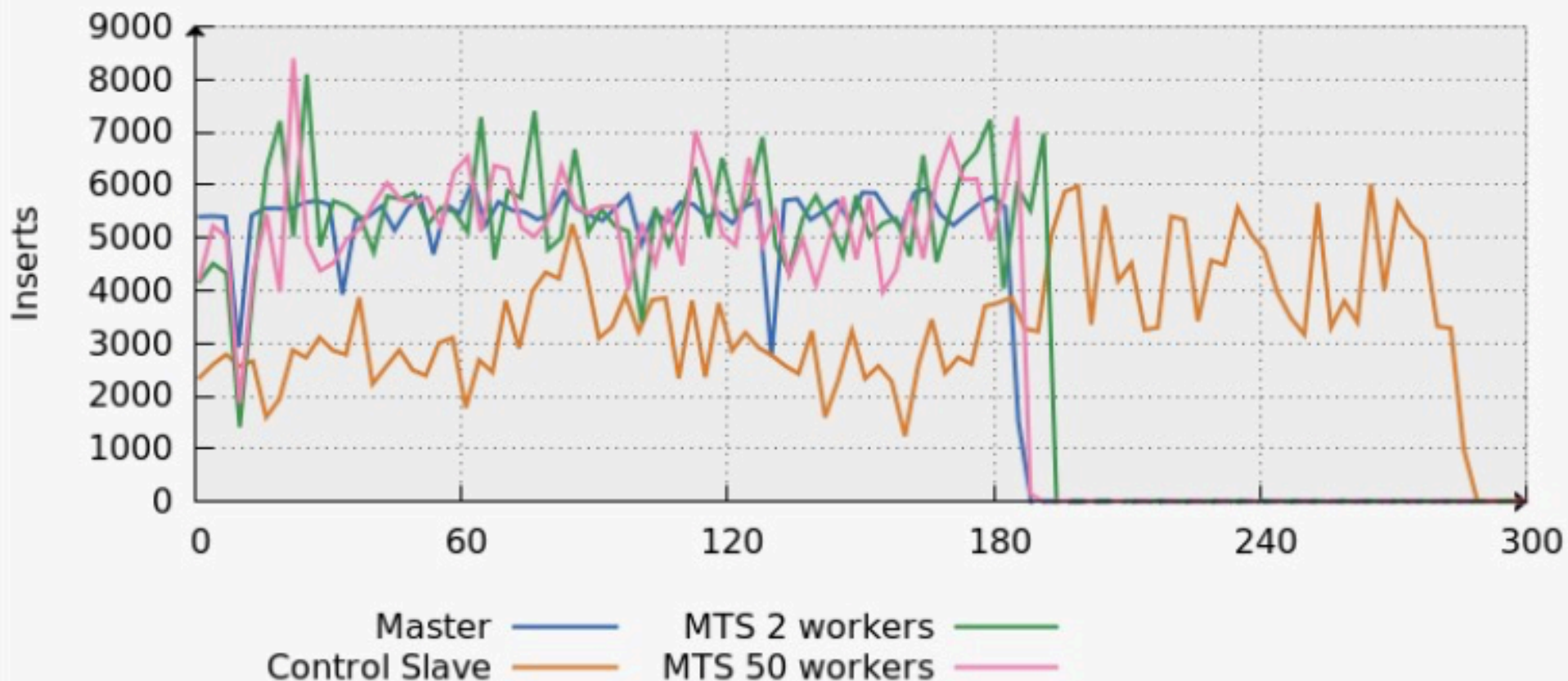


multi-threaded replication



multi-threaded replication

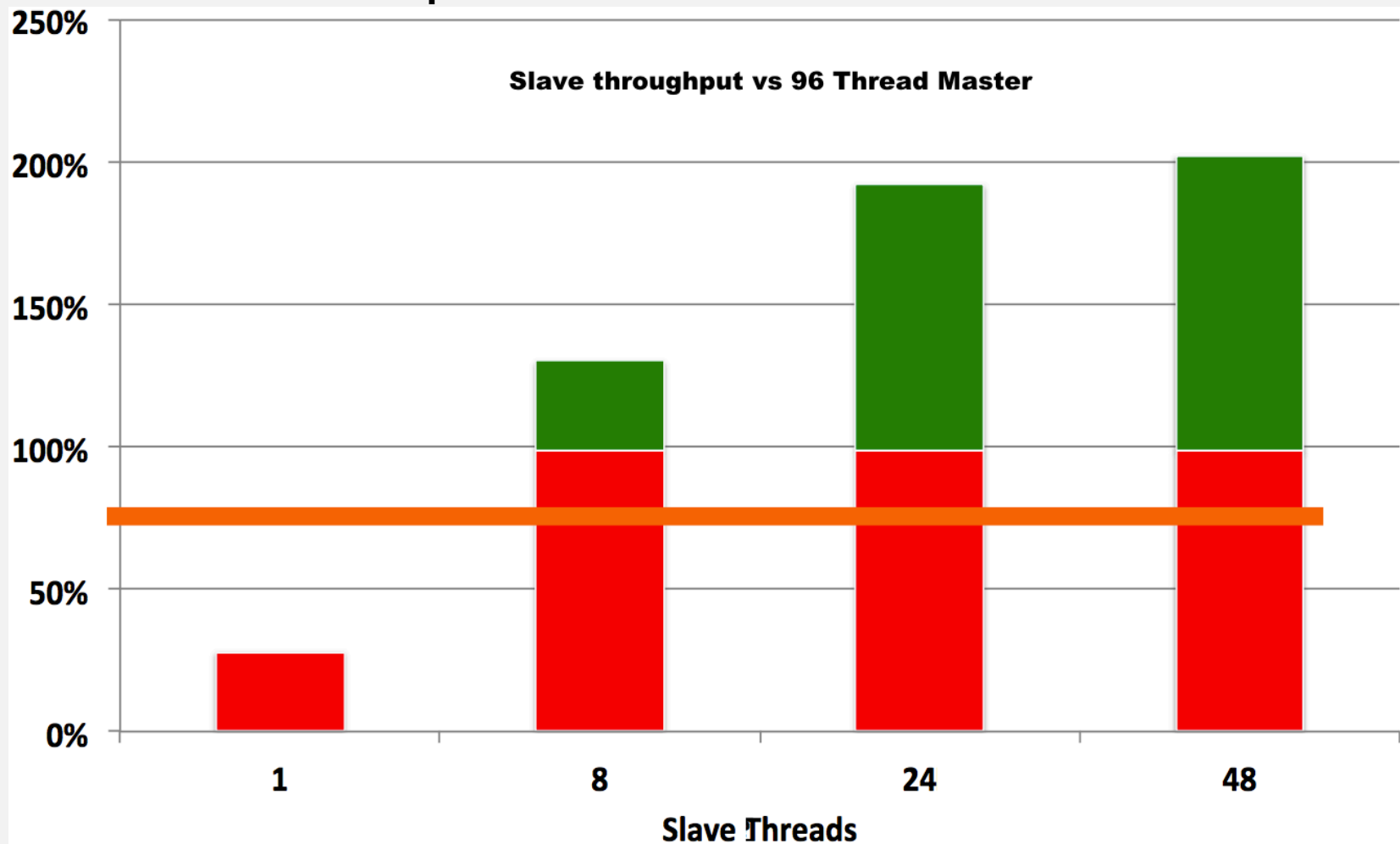
Insert Rate - 50%/50% write distribution



multi-threaded replication



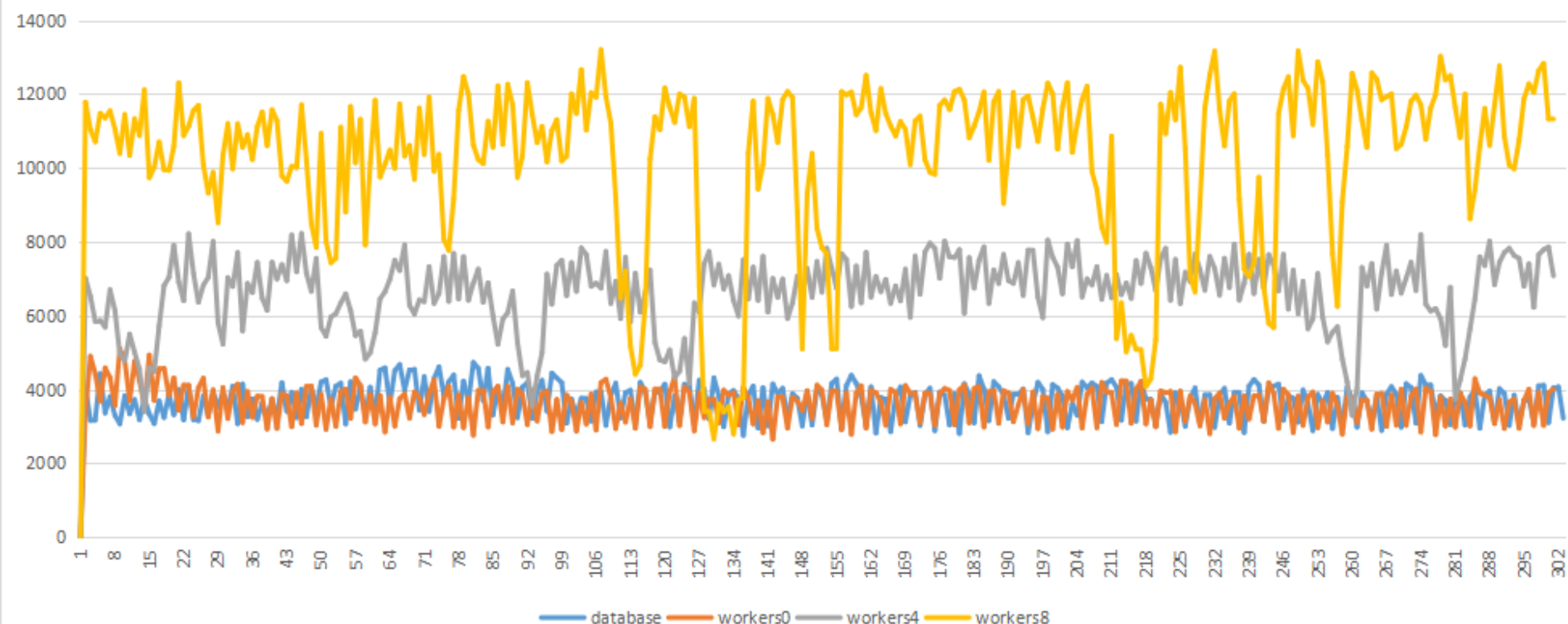
multi-threaded replication



图片源自《Whats New in MySQL 5.7》by Geir Høydaalsvik & Simon Mudd

multi-threaded replication

mysql并行复制性能对比图



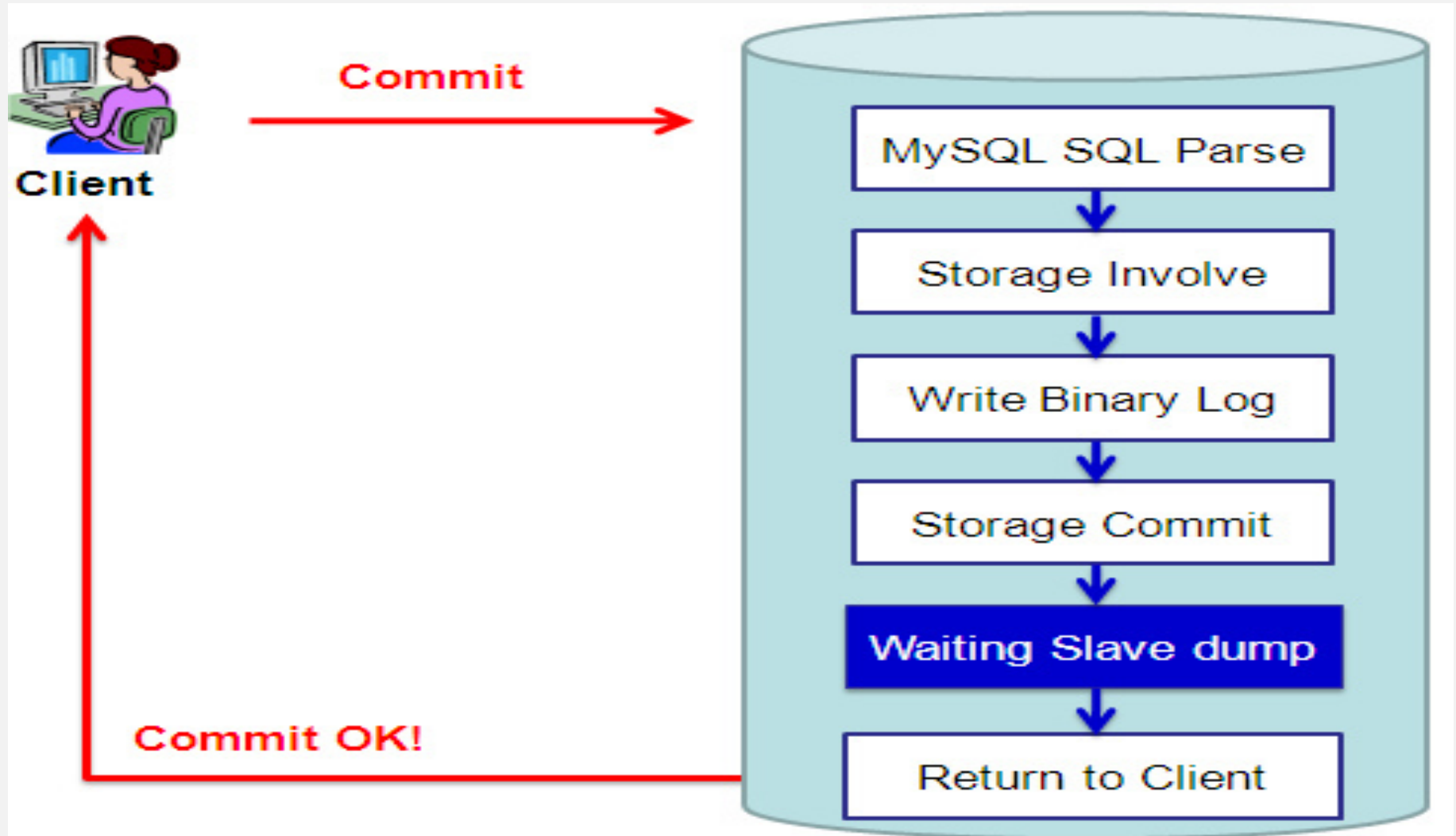
3、lossless semi-sync replication/无损半同步复制

- 设置master事务提交点AFTER_SYNC 或 AFTER_COMMIT，提高复制可靠性；
- 接收、发送信号线程分离（串行变并行），提高复制效率。

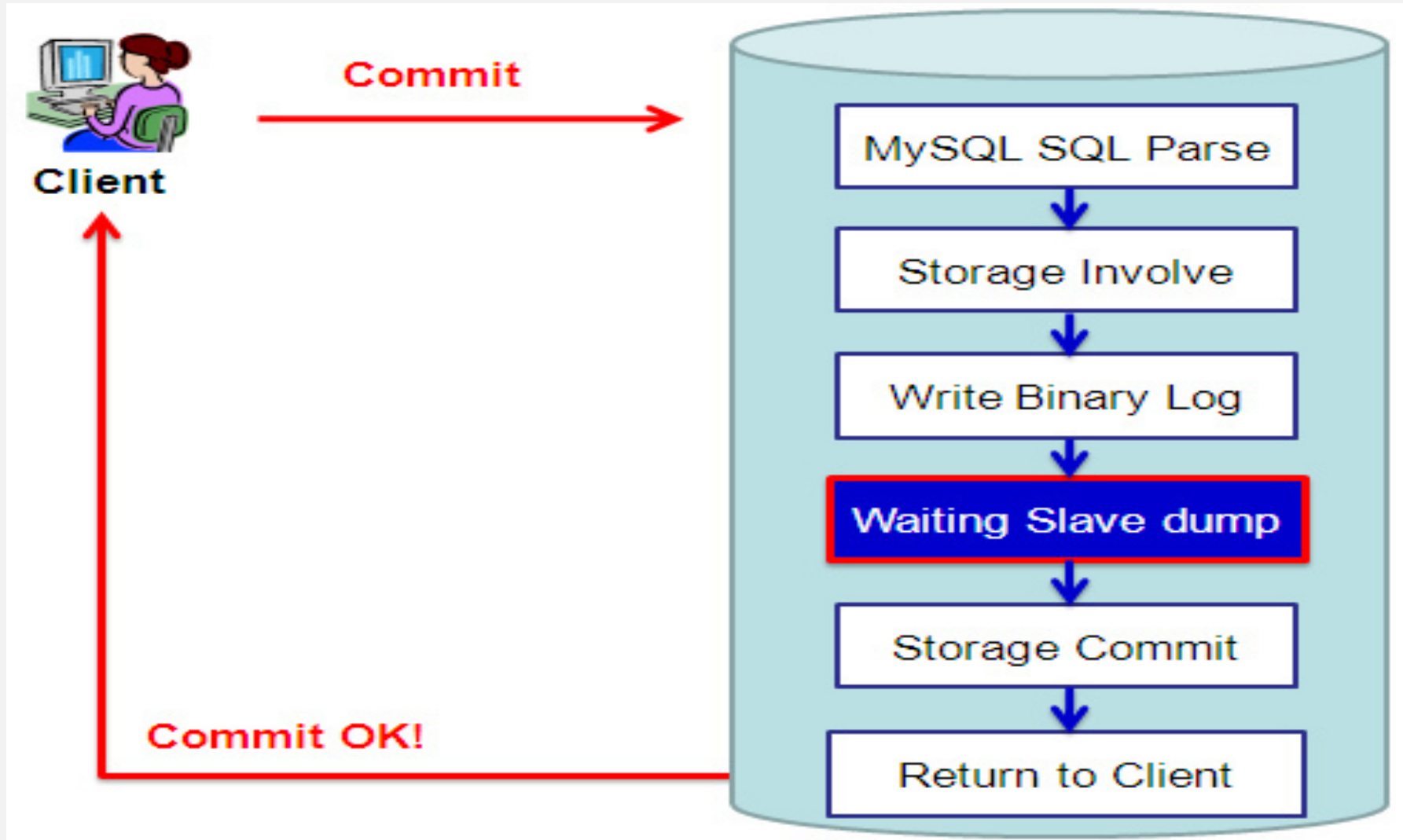
4、group replication/组复制

- 类似Galera Cluster（PXC）架构，支持多点并行写入，同时提供读负载均衡；
- 只支持InnoDB引擎，表必须有主键，且基于RBR。

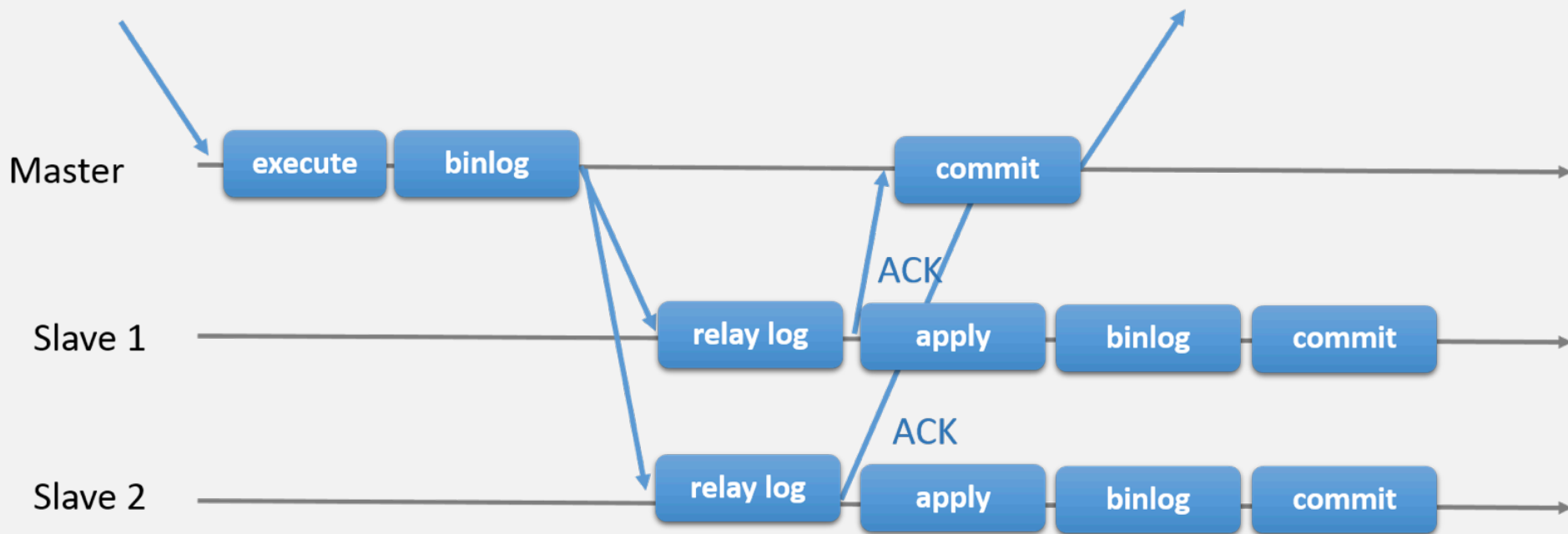
semi-sync replication



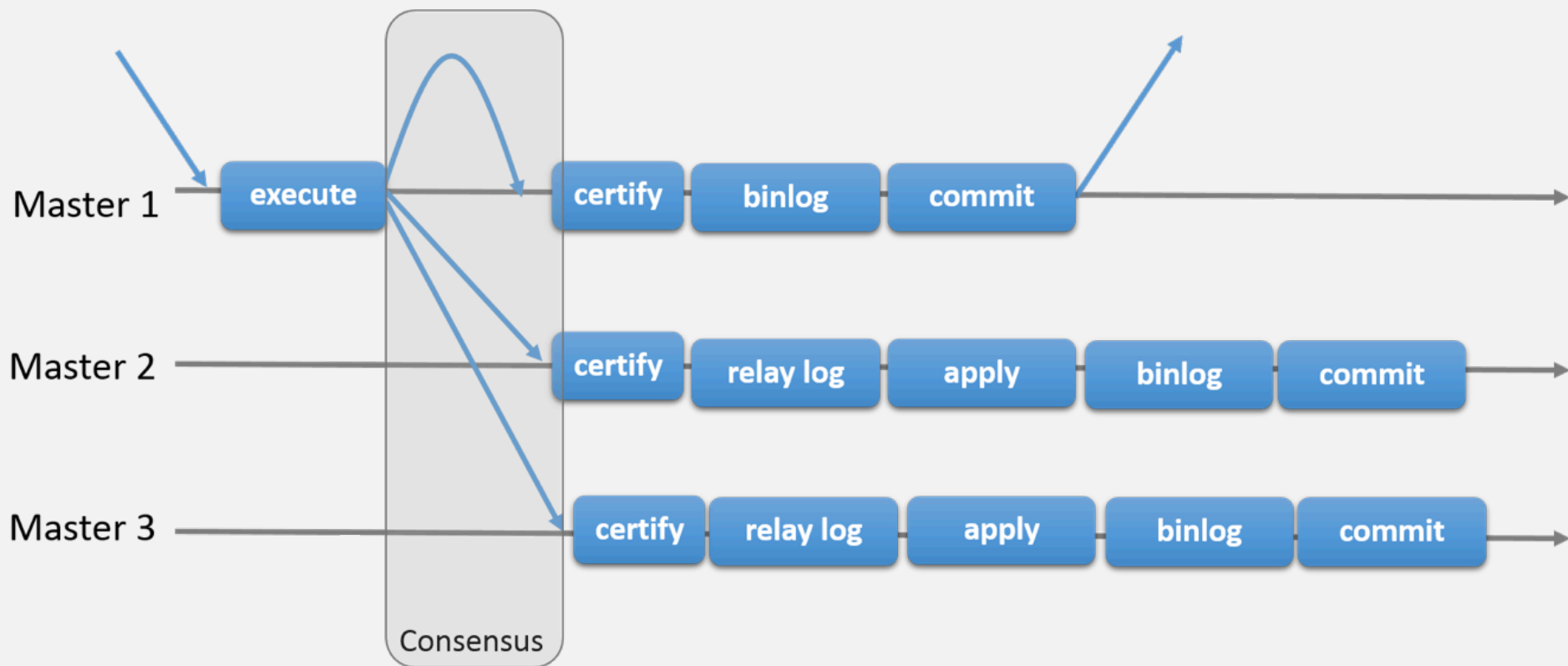
lossless semi-sync replication



增强semi-sync replication (ack异步且独立线程处理)

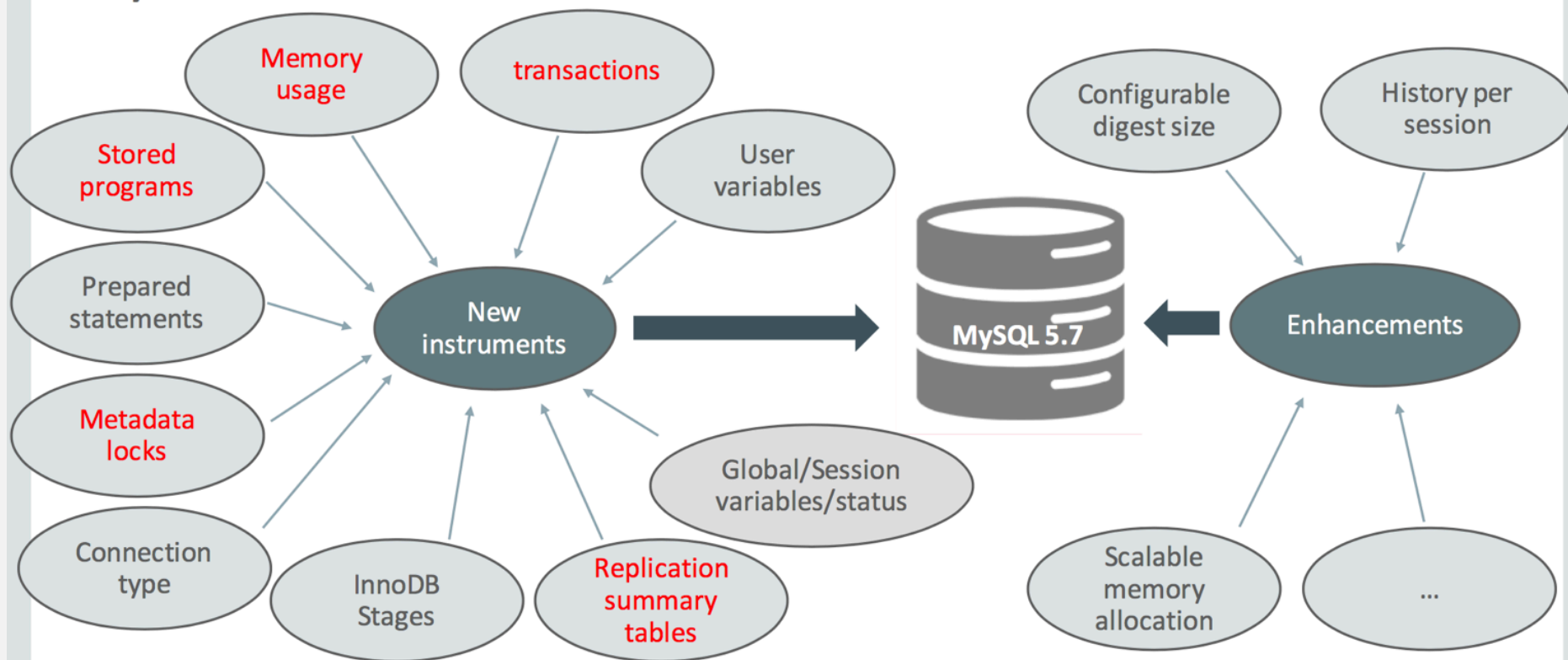


group replication



p_s、sys增强

MySQL 5.7 : Performance Schema



sys schema中， schema、table相关统计视图

```
yejr@imysql.com[sys]> show tables like 'schema%';
```

```
+-----+
| Tables_in_sys (schema%) |
+-----+
| schema_auto_increment_columns |
| schema_index_statistics |
| schema_object_overview |
| schema_redundant_indexes |
| schema_table_lock_waits |
| schema_table_statistics |
| schema_table_statistics_with_buffer |
| schema_tables_with_full_table_scans |
| schema_unused_indexes |
+-----+
```

table对象DML统计

```
mysql> select table_schema,table_name,rows_fetched,rows_inserted,
rows_updated,rows_deleted,io_read_requests,io_read,io_write_requests,
io_write from schema_table_statistics order by rows_inserted desc limit 2\G
***** 1. row *****
    table_schema: test
      table_name: t1
    rows_fetched: 1187841
   rows_inserted: 102400
    rows_updated: 0
    rows_deleted: 0
   io_read_requests: 278
         io_read: 95.62 KiB
io_write_requests: 849
         io_write: 12.30 MiB
***** 2. row *****
    table_schema: yejr
      table_name: tt2
    rows_fetched: 512003
   rows_inserted: 102400
    rows_updated: 0
    rows_deleted: 0
   io_read_requests: 26
         io_read: 2.91 KiB
io_write_requests: 839
         io_write: 12.71 MiB
```

table对象DML统计

```
mysql> select rows_fetched,rows_inserted,rows_updated,rows_deleted,io_read_requests,io_read,io_write_requests,io_write from  
schema_table_statistics where table_schema = 'test' and table_name = 't1';
```

rows_fetched	rows_inserted	rows_updated	rows_deleted	io_read_requests	io_read	io_write_requests	io_write
1187841	102400	0	0	278	95.62 KiB	849	12.30 MiB

index对象usage统计

```
mysql> select index_name, rows_selected, rows_inserted, rows_updated, rows_deleted  
       from schema_index_statistics where table_schema = 'test' and table_name = 't1';
```

index_name	rows_selected	rows_inserted	rows_updated	rows_deleted
u1	163841	0	0	0
PRIMARY	0	0	0	0

redundant indexes

```
mysql> select * from schema_redundant_indexes limit 1\G
***** 1. row *****
      table_schema: yejr
      table_name: t3
      redundant_index_name: u1
      redundant_index_columns: u1
      redundant_index_non_unique: 1
      dominant_index_name: u1_2
      dominant_index_columns: u1,u2
      dominant_index_non_unique: 1
      subpart_exists: 0
      sql_drop_index: ALTER TABLE `yejr`.`t3` DROP INDEX `u1`
```

unused indexes

```
mysql> select * from schema_unused_indexes;
```

object_schema	object_name	index_name
yejr	t1	int_col
yejr	t1	char_col
yejr	t2	char_col
yejr	t3	u1
yejr	t3	u1_2
yejr	t3	u3
yejr	tt2	int_col
yejr	tt2	char_col

row lock waits

```
yejr@imysql.com [sys]>select * from sys.innodb_lock_waits\G
***** 1. row *****
      wait_started: 2017-02-13 15:32:54
      wait_age: 00:00:30
      wait_age_secs: 30
      locked_table: `test`.`t1`
      locked_index: PRIMARY
      locked_type: RECORD
      waiting_trx_id: 248632
      waiting_trx_started: 2017-02-13 15:32:54
      waiting_trx_age: 00:00:30
      waiting_trx_rows_locked: 1
      waiting_trx_rows_modified: 0
      waiting_pid: 24
      waiting_query: select * from t1 where c1=2 for update
      waiting_lock_id: 248632:293:3:3
      waiting_lock_mode: X
      blocking_trx_id: 248631
      blocking_pid: 22
      blocking_query: NULL
      blocking_lock_id: 248631:293:3:3
      blocking_lock_mode: X
      blocking_trx_started: 2017-02-13 15:32:37
      blocking_trx_age: 00:00:47
      blocking_trx_rows_locked: 1
      blocking_trx_rows_modified: 0
      sql_kill_blocking_query: KILL QUERY 22
      sql_kill_blocking_connection: KILL 22
```


mdl lock waits

```
mysql> select * from schema_table_lock_waits\G
...
***** 2. row *****
      object_schema: test
      object_name: t1
      waiting_thread_id: 628
      waiting_pid: 595
      waiting_account: root@localhost
      waiting_lock_type: EXCLUSIVE
      waiting_lock_duration: TRANSACTION
      waiting_query: alter table t1 drop index u1
      waiting_query_secs: 29
      waiting_query_rows_affected: 0
      waiting_query_rows_examined: 0
      blocking_thread_id: 629
      blocking_pid: 596
      blocking_account: root@localhost
      blocking_lock_type: SHARED_WRITE
      blocking_lock_duration: TRANSACTION
      sql_kill_blocking_query: KILL QUERY 596
      sql_kill_blocking_connection: KILL 596
```

```
mysql> show processlist;
```

Id	User	Host	db	Command	Time	State	Info
595	root	localhost	test	Query	4	Waiting for table metadata lock	alter table t1 drop index u1
596	root	localhost	test	Query	0	starting	show processlist

I/O写最多的文件

```
yejr@imysql.com [sys]>select * from x$io_global_by_file_by_bytes
-> order by total_written desc limit 1\G
***** 1. row *****
      file: /Volumes/DATA/mysql/test/sid.ibd
count_read: 3046
total_read: 49954816
  avg_read: 16400.1366
count_write: 4054
total_written: 95322112
  avg_write: 23513.1011
      total: 145276928
write_pct: 65.61
```

热门SQL TOP10

```
mysql>select db,exec_count,query from statement_analysis order by exec_count desc limit 10;|
```

db	exec_count	query
yejr	299993	SET @? = `round` (`rand` () * ?) ;
yejr	299987	SET @? = `round` (`rand` () * ?) ;
test	15997	SELECT @@`version_comment` LIMIT ?
test	14004	UPDATE `t1` SET `c4` = ? WHERE `c1` = ?
test	2000	INSERT INTO `t1` SELECT ?, ...
information_schema	92	SHOW ENGINE `innodb` STATUS
test	54	SELECT ?
sys	37	SELECT IF (`isnull` (`perfor ... _host_by_event_name` GROUP BY
NULL	32	SHOW PROCESSLIST
test	29	SHOW TABLES

查看实例内存消耗

```
mysql> select *from memory_global_total;  
+-----+  
| total_allocated |  
+-----+  
| 3.49 GiB        |  
+-----+
```

查看内部对象内存消耗

```
yejr@imysql> select event_name,SUM_NUMBER_OF_BYTES_ALLOC from  
memory_summary_global_by_event_name  
order by SUM_NUMBER_OF_BYTES_ALLOC desc LIMIT 10;
```

event_name	SUM_NUMBER_OF_BYTES_ALLOC
memory/innodb/mem0mem	23839472166
memory/sql/Filesort_buffer::sort_k	2754080008
memory/memory/HP_PTRS	2632950064
memory/sql/thd::main_mem_root	1830295888
memory/mysys/IO_CACHE	977629336
memory/sql/String::value	946131984
memory/sql/TABLE	409615108
memory/mysys/MY_DIR	300412224
memory/sql/test_quick_select	178511200
memory/sql/QUICK_RANGE_SELECT	173118272

查看线程内存消耗

```
yejr@imysql>select event_name, SUM_NUMBER_OF_BYTES_ALLOC from  
memory_summary_by_thread_by_event_name  
order by SUM_NUMBER_OF_BYTES_ALLOC desc limit 20;
```

thread_id	event_name	SUM_NUMBER_OF_BYTES_ALLOC
1	memory/innodb/buf_buf_pool	139722752
1	memory/sql/Log_event	41576778
1	memory/sql/THD::Session_tracker	35384895
1	memory/sql/thd::main_mem_root	33832032
1	memory/sql/NET::buff	33402726
1239	memory/innodb/mem0mem	30407948
2327	memory/innodb/mem0mem	30407948
1220	memory/innodb/mem0mem	30407948

其他增强

1、EXPLAIN增强

- 查看当前活跃SESSION的SQL执行计划
EXPLAIN FOR CONNECTION 3306
- JSON格式输出结果中，可见执行计划代价信息
- 总是启用PARTITIONS/EXTENDED选项

2、JSON & Generated Columns

```
Create Table: CREATE TABLE `json_test` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `data` json DEFAULT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8
```

```
yejr@imysql.com [test]>select * from json_test where data->'$.type' = 'line';
```

```
+-----+-----+  
| id | data |  
+-----+-----+  
| 2 | {"type": "line", "coordinates": [-87.9101245, 41.7585879]} |  
+-----+-----+  
1 row in set (0.00 sec)
```

```
yejr@imysql.com [test]>select * from json_test where json_extract(data,'$.type') = 'line';
```

```
+-----+-----+  
| id | data |  
+-----+-----+  
| 2 | {"type": "line", "coordinates": [-87.9101245, 41.7585879]} |  
+-----+-----+
```

2、JSON & Generated Columns

```
yejr@imysql.com [test]>explain select * from json_test where data->'$.type' = 'line'\G
***** 1. row *****
      id: 1
select_type: SIMPLE
      table: json_test
  partitions: NULL
        type: ALL
possible_keys: NULL
          key: NULL
       key_len: NULL
         ref: NULL
        rows: 2
   filtered: 100.00
      Extra: Using where
```

2、JSON & Generated Columns

```
yejr@imysql.com [test]>alter table json_test add data_type varchar(255) GENERATED ALWAYS AS (data->'$.type') VIRTUAL;  
Query OK, 0 rows affected (0.04 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

```
yejr@imysql.com [test]>alter table json_test add key (data_type);  
Query OK, 0 rows affected (0.03 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

```
yejr@imysql.com [test]>explain select * from json_test where data_type = 'line';
```

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	json_test	NULL	ref	data_type	data_type	768	const	1	100.00	NULL

2、JSON & Generated Columns

```
Create Table: CREATE TABLE `sid` (  
  `id` int(10) unsigned NOT NULL DEFAULT '0',  
  `name` varchar(50) NOT NULL DEFAULT '',  
  `aid` int(10) unsigned NOT NULL AUTO_INCREMENT,  
  `nid` int(11) unsigned GENERATED ALWAYS AS ((`id` + 1)) VIRTUAL NOT NULL,  
  `nnid` int(11) unsigned GENERATED ALWAYS AS ((`id` + 1)) STORED NOT NULL,  
  PRIMARY KEY (`aid`),  
  KEY `name` (`name`),  
  KEY `id` (`id`),  
  KEY `nid` (`nid`)  
) ENGINE=InnoDB AUTO_INCREMENT=893210 DEFAULT CHARSET=utf8
```

2、JSON & Generated Columns

```
yejr@imysql.com [test]>desc select * from sid where id+1=1024\G
***** 1. row *****
```

```
      id: 1
select_type: SIMPLE
      table: sid
partitions: NULL
      type: ref
possible_keys: nid
         key: nid
      key_len: 4
         ref: const
        rows: 8
  filtered: 100.00
      Extra: NULL
```

3、SELECT中设置超时阈值

- SQL Hint
 - `SELECT /*+ MAX_EXECUTION_TIME(10000) */ ...`
- 也可设定 `max_execution_time` 选项
- 可有效避免某个垃圾SQL导致雪崩
- 参考
 - MariaDB/Percona选项 `innodb_kill_idle_transaction`

1、支持invisible index

- 无需重建索引，快速完成
- in-place设置visible/invisible
- force index也不可用
- 索引统计信息仍旧更新
- 主键不能invisible

2、descending index

- 有了倒序索引，倒序扫描效率更高了
- 也支持虚拟列（VIRTUAL、STORED）
- 只支持B+TREE，不支持HASH、FULLTEXT、SPATIAL索引
- 只支持InnoDB

- SELECT增加NOWAIT、SKIP LOCKED选项
- InnoDB memcached支持mget指令
- InnoDB表自增列最大值持久化
- 直方图 (Histogram)

THANK YOU

