


ORACLE®

ORACLE®

# MySQL Server: Performance and Scalability

MySQL Global Business Unit  
Sales Consulting Manager, JAPAC  
梶山 隆輔 / Ryusuke Kajiyama





以下の事項は、弊社の一般的な製品の方向性に関する概要を説明するものです。また、情報提供を唯一の目的とするものであり、いかなる契約にも組み込むことはできません。以下の事項は、マテリアルやコード、機能を提供することをコミットメント(確約)するものではないため、購買決定を行う際の判断材料になさらないで下さい。オラクル製品に関して記載されている機能の開発、リリースおよび時期については、弊社の裁量により決定されます。

OracleとJavaは、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。文中の社名、商品名等は各社の商標または登録商標である場合があります。

ORACLE



# Program Agenda

- **MySQL 5.6**
- **MySQL Enterprise Edition**
- **MySQL 5.7**
- **Performance Schema**
- **MySQLベンチマーク vs 他製品**



# MySQL 5.6 GA



ORACLE

## MySQL 5.6 GA

New!

- **オプティマイザ**: パフォーマンス&スケーラビリティ
- **パフォーマンス・スキーマ**: より詳細な統計情報
- **InnoDB**: トランザクション・スループットの向上
- **レプリケーション**: さらなる可用性とデータの整合性
- **「NotOnlySQL」オプション**: さらなる柔軟性

- ダウンロードはこちらから！

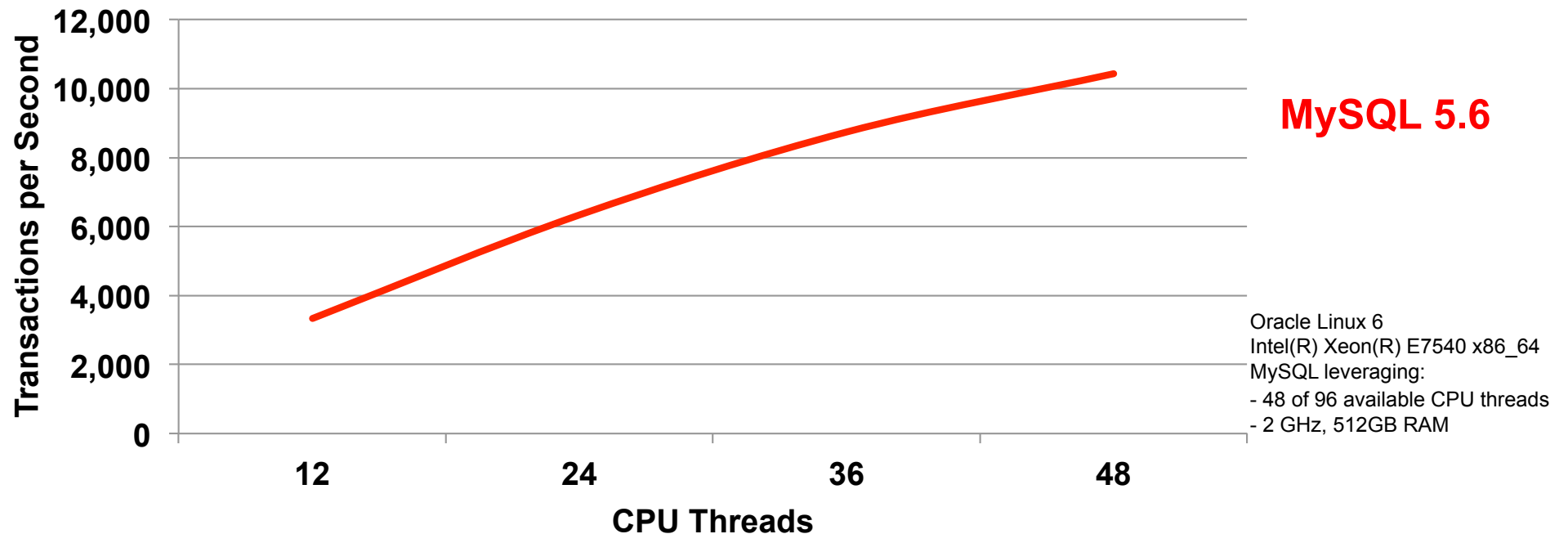
[dev.mysql.com/downloads/mysql/](http://dev.mysql.com/downloads/mysql/)



ORACLE

# MySQL 5.6: Scalability

## MySQL 5.6 Read Write (Linux)

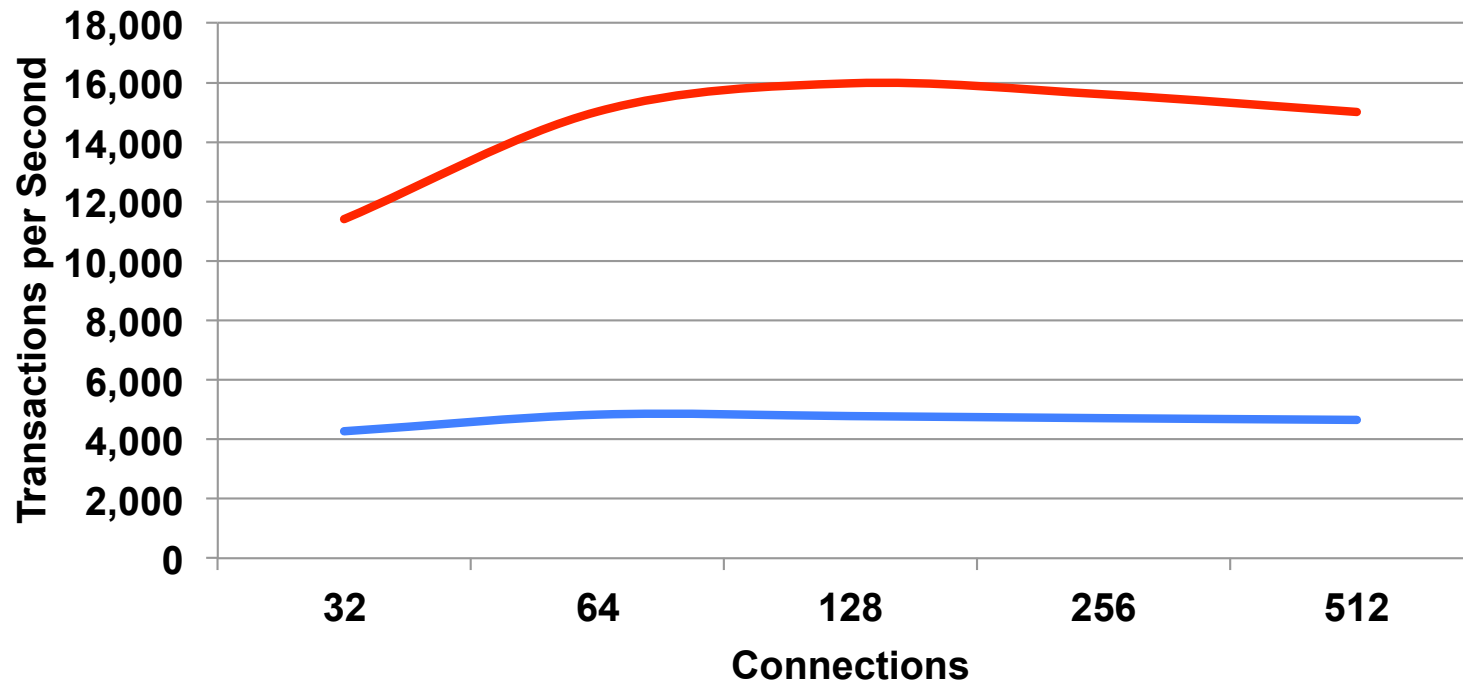


- 最新のハードウェアやOSに対応した性能拡張性

ORACLE

# MySQL 5.6 SysBench Benchmarks

## MySQL 5.6 vs. 5.5 - Read Only (Linux)



MySQL 5.6

MySQL 5.5

Oracle Linux 6  
Intel(R) Xeon(R) E7540 x86\_64  
MySQL leveraging:  
- 48 of 96 available CPU threads  
- 2 GHz, 512GB RAM

**Up to 234% Performance Gain**

ORACLE



# MySQL 5.6: InnoDB

## Better Performance, Scalability

- 複数の内部実装の改良 (例: カーネルミューテックスの分割、バッファプールのフラッシュの効率改善など)
- 参照専用トランザクションの実装
- オプティマイザ統計の永続化
  - 安定して正確な実行計画
  - ユーザから制御、自動/手動
- SSDへの最適化
  - 4, 8kページサイズ
  - .ibdファイルをデータディレクトリ以外へ
  - UNDOログ表領域を分離



# MySQL 5.6: InnoDB

## 参照処理の性能向上

- 参照処理の同時実行が多いWebアプリケーションなどで効果大
- 開発者が参照専用トランザクションを選択することでオーバーヘッド削減

```
SET autocommit = 1;  
SELECT c FROM sbtest WHERE id=N;
```

デフォルト

```
SET autocommit = 0;  
START TRANSACTION READ ONLY;  
SELECT c FROM sbtest WHERE id=N;  
COMMIT;
```

参照専用トランザクション開始

<http://dev.mysql.com/doc/refman/5.6/en/innodb-performance.html#innodb-performance-ro-txn>

ORACLE

# MySQL 5.6: InnoDB

## バッファプールのダンプ&リストア

- 起動直後からバッファプールにデータがキャッシュされた状態に
- シャットダウン/起動時に自動で、または手動で
- ディスク上にはテーブルスペースのページIDのみを書き出す

シャットダウン時に自動的にバッファプールの内容をダンプ:

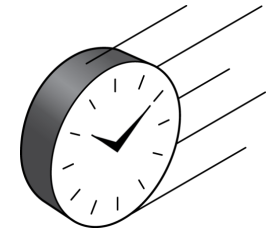
```
mysql> SET innodb_buffer_pool_dump_at_shutdown=ON;
```

起動時にダンプされた内容をバッファプールにロード:

```
mysql> SET innodb_buffer_pool_load_at_startup=ON;
```

- 再起動直後や新しいサーバの起動直後でも性能劣化しない
- クラウド、ホスティング、SaaSなどの環境でもメリット

# MySQL 5.6: オプティマイザ



- サブクエリ的高速化
- LIMIT句で少数のレコードを取得する際のファイル・ソートを最適化
  - 4倍高速化 - 40秒から10秒に短縮
- インデックス条件のプッシュダウン
  - 160倍高速化 - 15秒から90ミリ秒に短縮
- FROMからのビュー／サブクエリの実データ取得を遅延
  - EXPLAINが240倍高速化 - 8分から2秒に短縮
- バッチ・キー・アクセスと複数範囲の読み取り
  - 280倍高速化 - 2800秒から10秒に短縮
- オプティマイザの統計情報の永続化

# 構造化されたEXPLAIN

## MySQL 5.6の新機能

```
EXPLAIN FORMAT=JSON
SELECT  l_returnflag,  l_linestatus,
        SUM(l_quantity) AS sum_qty
FROM    lineitem
WHERE   l_shipdate <=
        DATE_SUB('1998-12-01',
        INTERVAL '118' DAY)
GROUP BY l_returnflag,  l_linestatus
ORDER BY l_returnflag,  l_linestatus;
```



### EXPLAIN

```
{ "query_block": {
  "select_id": 1,
  "ordering_operation": {
    "using_filesort": false,
    "grouping_operation": {
      "using_temporary_table": true,
      "using_filesort": true,
      "table": {
        "table_name": "lineitem",
        "access_type": "ALL",
        "possible_keys": [
          "i_l_shipdate"
        ],
        "rows": 2829575,
        "filtered": 50,
        "attached_condition":
          "(`dbt3`.`lineitem`.`l_shipDATE` <=
          <cache>(('1998-12-01' - interval '118' day)))"
      } /* table */
    } /* grouping_operation */
  } /* ordering_operation */
} /* query_block */ }
```

ORACLE

# Optimizer Traces

## MySQL 5.6の新機能

### クエリの実行計画の最適化を確認

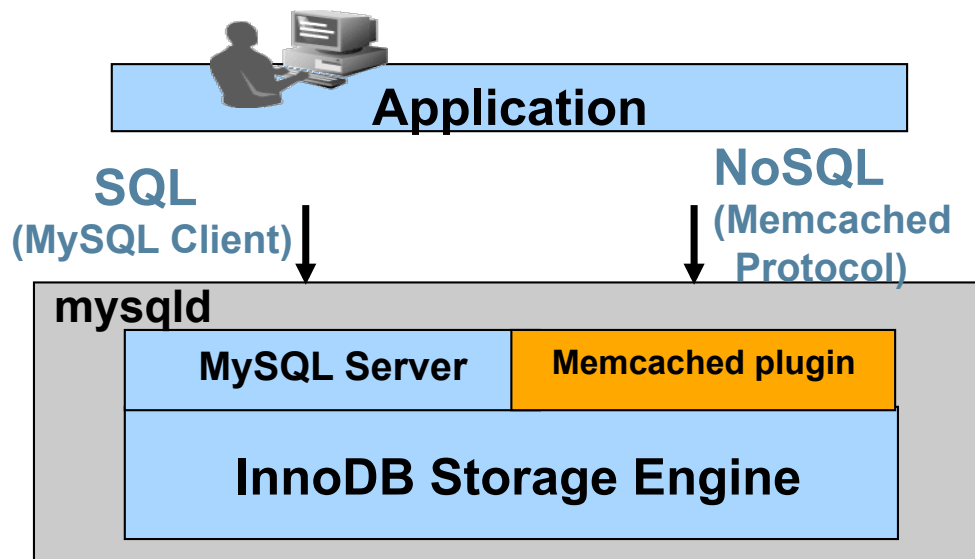
- オプティマイザによる主なステップと最適化の流れを追跡
- 実行例:

```
> SET optimizer_trace="enabled=on";  
> SELECT a, b FROM t1 WHERE a > 10;  
> SELECT * FROM  
    INFORMATION_SCHEMA.OPTIMIZER_TRACE;
```

```
"table": "`t1`",  
"range_analysis": {  
  "table_scan": {  
    "rows": 54,  
    "cost": 13.9  
  },  
  "best_covering_index_scan": {  
    "index": "idx_a_b",  
    "cost": 11.903,  
    "chosen": true  
  },  
  "analyzing_range_alternatives": {  
    "range_scan_alternatives": [  
      {  
        "index": "idx_a_b",  
        "ranges": [  
          "10 < a"  
        ],  
        "rowid_ordered": false,  
        "using_mrr": false,  
        "index_only": true,  
        "rows": 12,  
        "cost": 3.4314,  
        "chosen": true  
      }  
    ]  
  }  
}
```

# MySQL 5.6: InnoDB

## RDBMSとNoSQLの両立

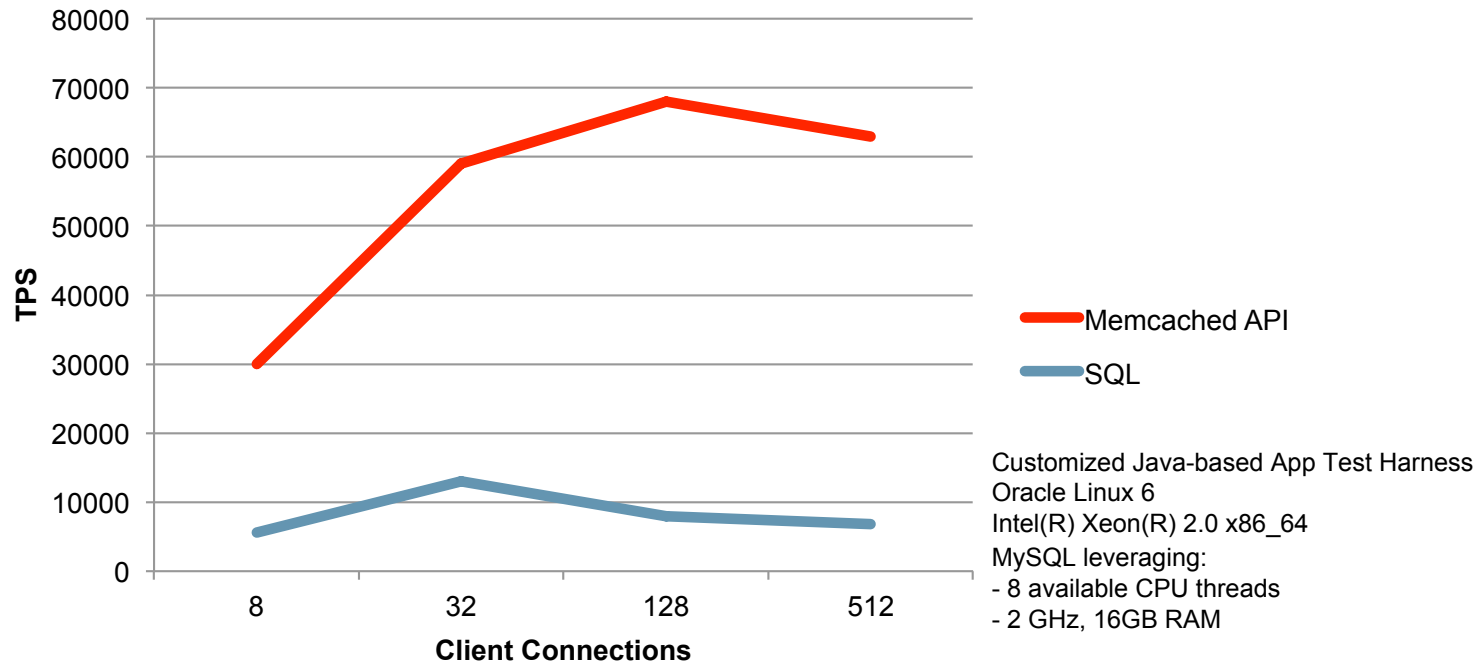


- InnoDBに素早く、簡単にアクセス
  - Memcached API経由のアクセス
  - 既存のMemcachedクライアントを使用
  - SQL変換をバイパス
- NotOnlySQLアクセス
  - キー・バリュー操作
  - 複雑なクエリやJOIN、FKにはSQLを使用
- 実装
  - mysqlにMemcachedをデーモン・プラグインとして統合
  - ネイティブInnoDB APIをmemcachedプロトコルにマッピング
  - 超低レイテンシ用の共有プロセス・スペース

ORACLE

# NoSQL APIによる性能

## MySQL 5.6: NoSQL Benchmarking



**Up to 9x Higher “SET / INSERT” Throughput**

[blogs.oracle.com/mysqlinnodb/entry/new\\_enhancements\\_for\\_innodb\\_memcached](https://blogs.oracle.com/mysqlinnodb/entry/new_enhancements_for_innodb_memcached)

ORACLE





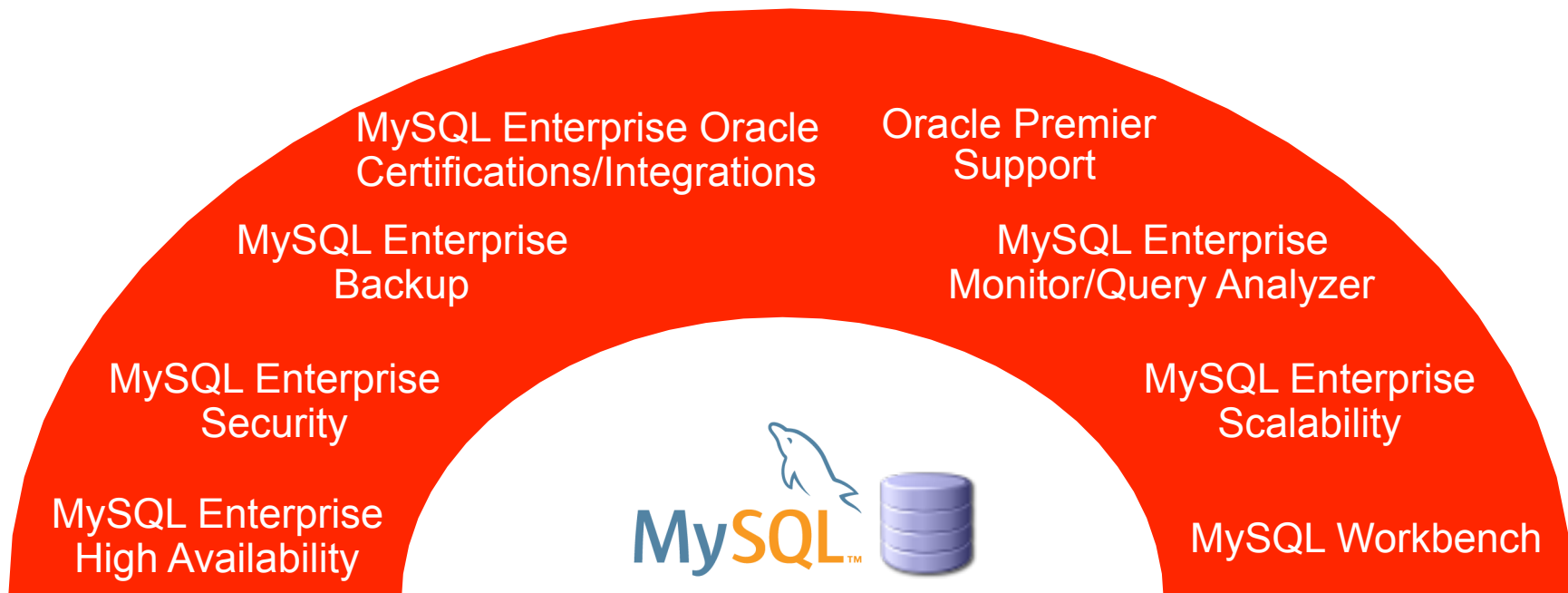
# MySQL Enterprise Monitor 3.0



ORACLE®

# MySQL Enterprise Edition

最高レベルのMySQLスケーラビリティ、セキュリティおよび稼働時間

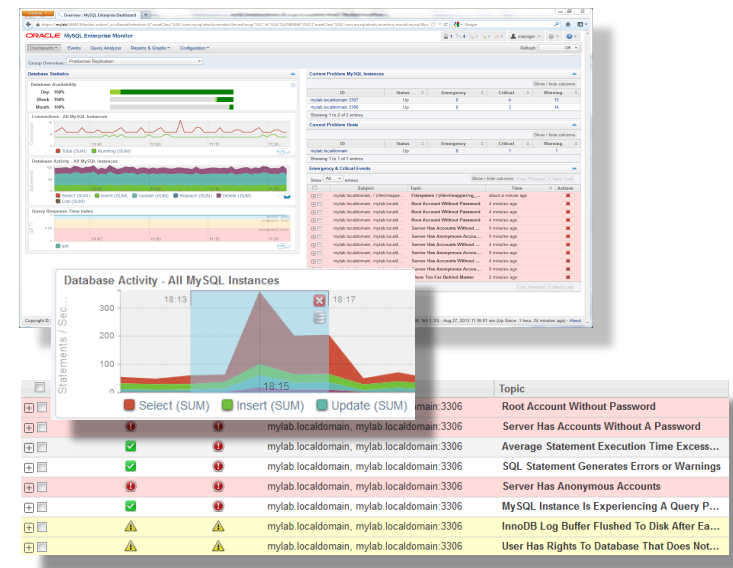


ORACLE

# MySQL Enterprise Monitor

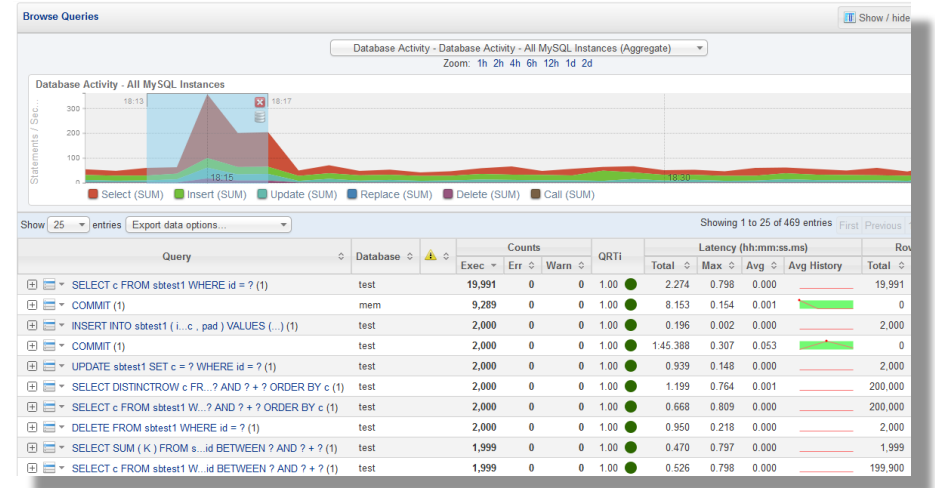
リアルタイムでMySQLの稼働状況とパフォーマンスを監視

- クエリーの問題を視覚的に把握し、修正
- キャパシティプランニングのために、ディスク使用量を監視
- クラウド環境と親和性の高いアーキテクチャ (エージェントレス)
- MySQLの監視を10分で開始
- OS監視のためのリモートエージェントオプション



# MySQL Query Analyzer

- クエリーのパフォーマンスをリアルタイムで確認
- グラフにて相関関係の確認
- 高コストなクエリーを発見し修正
- クエリーの実行計画や詳細情報を確認
- Query Response Time index (QRTi)



「MySQL Query Analyzer を使用することで、問題のある SQLコードを特定および解析して、データベースパフォーマンスを3倍に改善することができました。さらに重要なことに、これは、何週間もかからずに、わずか3日で実現できました」

Big Fish Games 社  
ソフトウェア開発エンジニア  
キース・ソーラダ氏 (Keith Souhrada)



ORACLE



# MySQL Enterprise Scalability

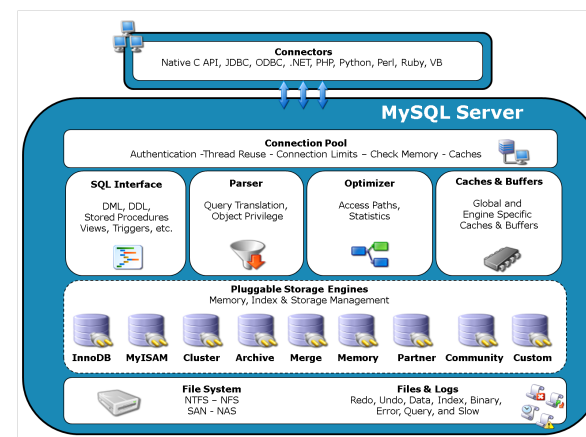


ORACLE®

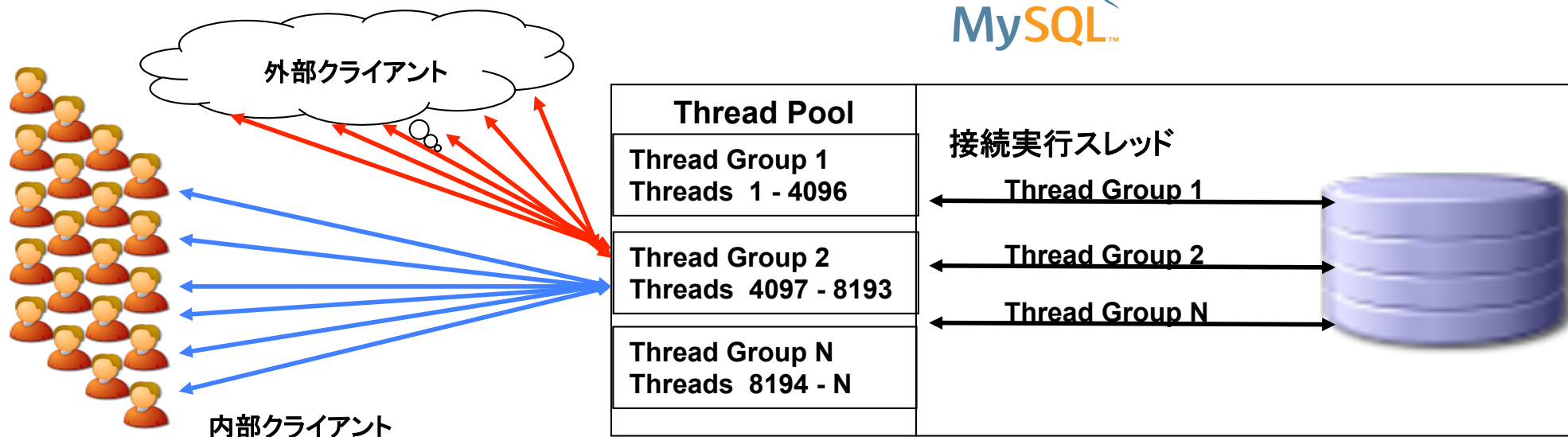
# MySQL Enterprise Scalability

## Thread Pool

- MySQLデフォルト・スレッド処理はパフォーマンスは高いが、接続数が増加するとスケーラビリティに制約が出る可能性がある
- MySQL Thread Pool  
ユーザ接続数の増加に対応し、パフォーマンスとスケーラビリティを維持
- Thread Pool API

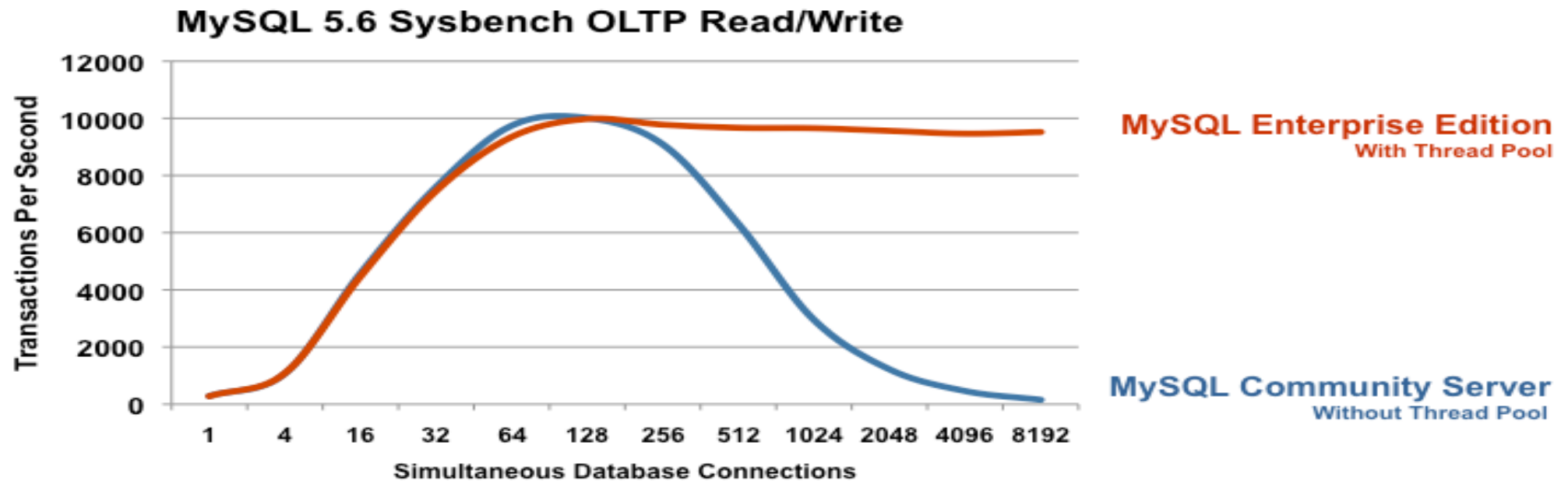


# Thread Poolの有効化



- スレッド・グループ数を設定可能(デフォルト = 16)、4096スレッド
- ラウンド・ロビンによって各接続をスレッド・グループに割り当てる
- スレッドは優先付けされる。ステートメントはキューに挿入することで同時 実行を制限し、サーバの負荷や接続増加に対応したスケールビリティを確保

# MySQL Enterprise Scalability



## Configuration

MySQL 5.6.11

Oracle Linux 6.3, Unbreakable Kernel 2.6.32

4 sockets, 24 cores, 48 Threads

Intel(R) Xeon® E7540 2GHz CPUs

512GB DDR RAM






# MySQL 5.7 DMR



ORACLE®



## MySQL 5.7: DMR 4

MySQL 5.7 builds on MySQL 5.6 by improving:

- **InnoDB** for better transactional throughput, availability, IO
- **Replication** for better scalability and availability
- **Utilities** for dev/ops automation
- **Performance Schema** for better performance metrics
- **Optimizer** for better EXPLAINing, query performance, enhanced buffering and partition optimization
- **Connecting** at higher rates, improve session efficiency

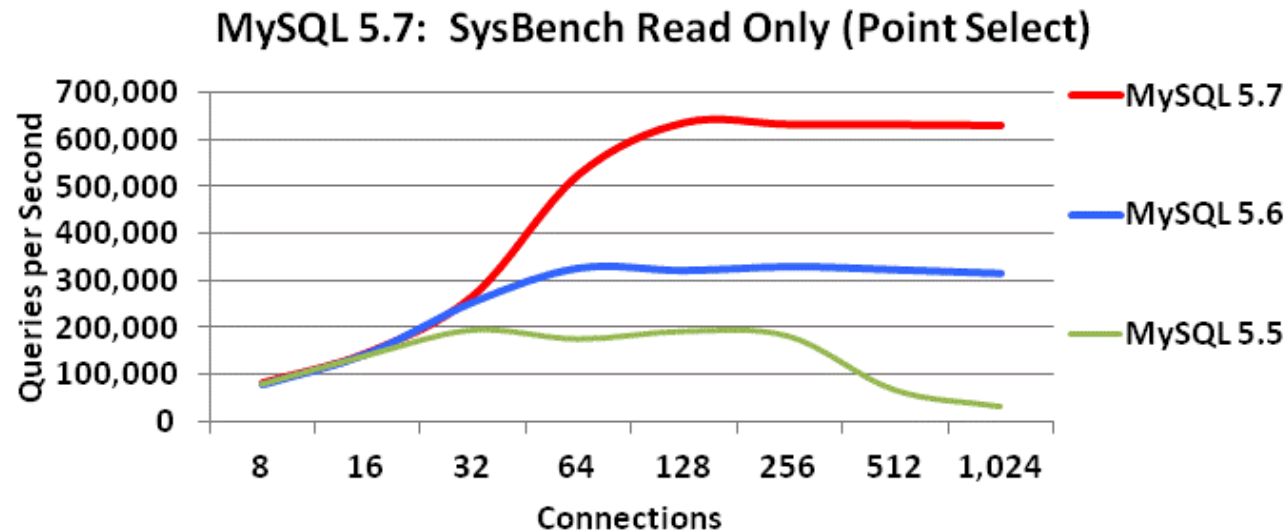
Available Now! Get it here: [dev.mysql.com/downloads/mysql/](http://dev.mysql.com/downloads/mysql/)



# MySQL 5.7 Sysbench Benchmark

Sysbench Point Select

**630,000 QPS**



Intel(R) Xeon(R) CPU X7560 x86\_64  
5 sockets x 8 cores-HT (80 CPU threads)  
2.27GHz, 256G RAM  
Oracle Linux 6.5

**2X Faster than MySQL 5.6**  
**Over 3X Faster than MySQL 5.5**

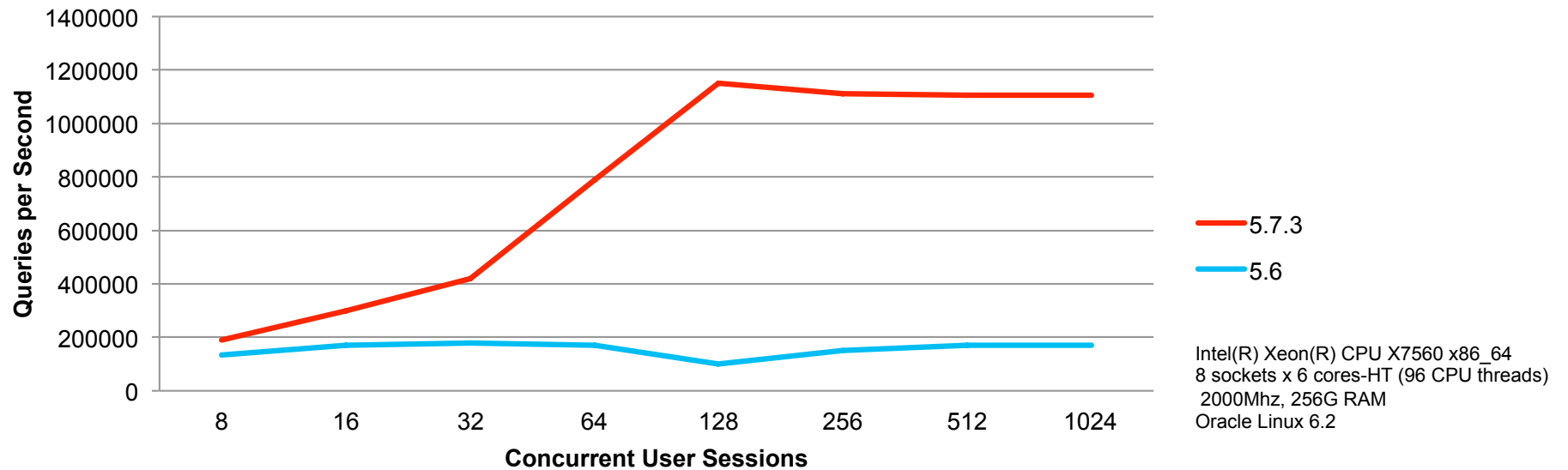
ORACLE



# MySQL 5.7: InnoDB Memcached

Thank you Yoshinori (Facebook)

**1,150,000 QPS**



**6x Faster than MySQL 5.6**

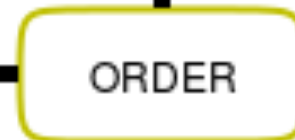
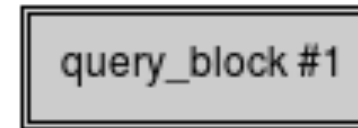


# Visual EXPLAIN

MySQL Workbench



Query cost: 4252434.00



tmp table, filesort

1251827.0 6.00M rows



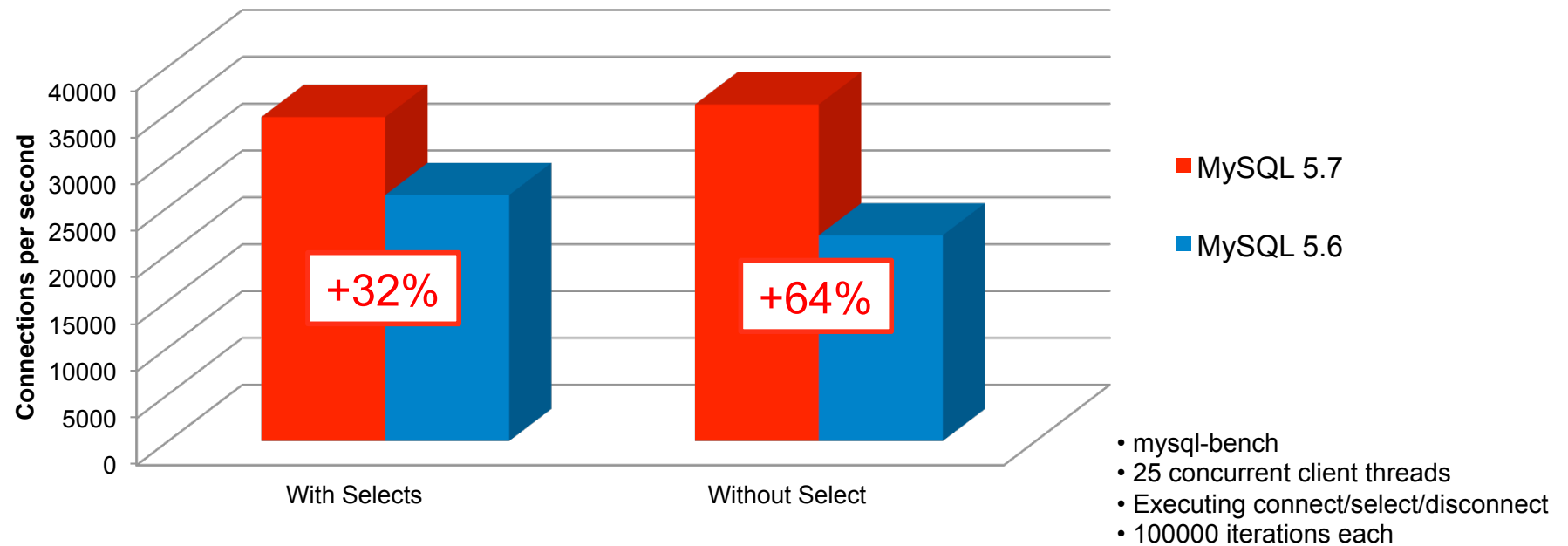
lineitem

Cost estimate  
(MySQL 5.7)



# MySQL 5.7: Connections / Second

Faster processing of new connections



Built with input from Facebook



# MySQL 5.7: Optimizer Cost Model

- New cost model API
- Allows storage engines to provide accurate and dynamic cost estimates for key lookups, table scans, range scans, etc...
  - Enables future support of additional factors
    - Whether the data is in RAM, SSD, HDD
- Lays the groundwork for making costs configurable
  - Based on your hardware performance characteristics
- Improves records per key estimates
- Cost values included in JSON Explain output



# MySQL 5.7: InnoDB

## Many Improvements

- Improved InnoDB Online Alter Table
  - Online Rename Index, Online Change Varchar
- Enhanced FusionIO Integration
  - Doublewrite buffer automatically disabled when DirectFS detected
- Parallel “Dirty Page” Flushing
  - Higher throughput, performance, and scalability
- Partitions – support for Transportable Tablespaces (TTS)
  - TTS support for individual partitions





# MySQL 5.7: InnoDB Compression

labs.mysql.com

Thank you, Fusion-io

- Transparent Page Level Compression
  - Happens transparently in background threads
  - Managed entirely within the IO layer
  - Uses sparse file and "hole punching" in OS kernels and File Systems
- Reduces IO
  - Improves performance
  - Reduces write cycles, thus increasing SSD lifespan
- Applies to all tables, including the system tablespace and UNDO logs

ORACLE



# MySQL 5.7: Improved MDL locking

- Removes bottlenecks around DML access to a single table
  - 10% increased throughput in OLTP\_RO/POINT\_SELECT sysbench tests on higher core counts
  - Optimized for typical DML heavy workloads
- Implemented fast-path for DML locks
- Implemented lock-free DML lock acquisition
- Implemented a lock-free hash
  - Now uses MurmurHash library



# Performance Schema



# パフォーマンススキーマとは？

- MySQL 5.5から実装
- ストレージエンジンの一種で、稼働統計を記録
  - PERFORMANCE\_SCHEMA
  - Lock-freeハッシュを利用したメモリ内のバッファ管理
  - INFORMATION\_SCHEMAとは異なり、ストレージエンジンとして実装
- データベーススキーマとして、稼働統計を表示
  - performance\_schema
- インターフェース群として、MySQLサーバの内部コールをトレース
  - 例: pthread\_mutex\_lock() -> mysql\_mutex\_lock()



## パフォーマンススキーマとは? (続き)

- サーバ内で発生するイベントのレイテンシを記録
- 全てのレイテンシをピコ秒単位で表示
  - ただし、内部で記録される精度は異なる場合も
- その他、必要となる情報を記録
  - バイト数、ソースでの位置、オブジェクトのメタデータなど

# パフォーマンススキーマの設定

- パフォーマンススキーマの有効/無効

```
[mysqld]
performance_schema=on
```

- 個別のInstrumentsの設定:

```
[mysqld]
--performance_schema_instrument='wait/synch/cond/%=counted'
- off/false/0 = 無効
- on/true/1 = 有効 & 時間計測する
- counted = 有効 & 回数計測する、時間計測しない
```

<http://dev.mysql.com/doc/refman/5.6/en/performance-schema.html>

# パフォーマンススキーマの設定

## setup\_instruments

- 処理時間や待機時間を収集するMySQLサーバのソースコード内に設けられたinstrumentの設定
- 個別に有効無効や時間の記録の有無を設定可能
- UPDATE文にて動的に変更可能

```
mysql> SELECT * FROM setup_instruments;
```

NAME	ENABLED	TIMED
wait/synch/mutex/sql/TC_LOG_MMAP::LOCK_tc	NO	NO
wait/synch/mutex/sql/LOCK_des_key_file	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_commit	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_commit_queue	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_done	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_flush_queue	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_index	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_log	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_binlog_end_pos	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_sync	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_sync_queue	NO	NO
wait/synch/mutex/sql/MYSQL_BIN_LOG::LOCK_xids	NO	NO
wait/synch/mutex/sql/MYSQL_RELAY_LOG::LOCK_commit	NO	NO
wait/synch/mutex/sql/MYSQL_RELAY_LOG::LOCK_commit_queue	NO	NO
wait/synch/mutex/sql/MYSQL_RELAY_LOG::LOCK_done	NO	NO
wait/synch/mutex/sql/MYSQL_RELAY_LOG::LOCK_flush_queue	NO	NO
wait/synch/mutex/sql/MYSQL_RELAY_LOG::LOCK_index	NO	NO
wait/synch/mutex/sql/MYSQL_RELAY_LOG::LOCK_log	NO	NO
wait/synch/mutex/sql/MYSQL_RELAY_LOG::LOCK_sync	NO	NO
wait/synch/mutex/sql/MYSQL_RELAY_LOG::LOCK_sync_queue	NO	NO

# パフォーマンススキーマの設定

## setup\_consumers

- どれだけの情報を取得するかの設定(現在のみ、履歴を含むなど)
- UPDATE文にて動的に変更可能
- イベントテーブルで各SQL文を示すのDigest(MD5ハッシュ値)が有効となっている

```
mysql> SELECT * FROM setup_consumers;
+-----+-----+
| NAME                                | ENABLED |
+-----+-----+
| events_stages_current               | NO      |
| events_stages_history               | NO      |
| events_stages_history_long          | NO      |
| events_statements_current           | YES     |
| events_statements_history           | NO      |
| events_statements_history_long      | NO      |
| events_transactions_current         | YES     |
| events_transactions_history         | NO      |
| events_transactions_history_long    | NO      |
| events_waits_current                | NO      |
| events_waits_history                | NO      |
| events_waits_history_long           | NO      |
| global_instrumentation              | YES     |
| thread_instrumentation              | YES     |
| statements_digest                   | YES     |
+-----+-----+
15 rows in set (0.00 sec)
```



# パフォーマンススキーマの設定

## setup\_objects

- 監視対象とするテーブルなどのオブジェクトを設定
- システム系のスキーマなどはデフォルトで監視対象外としている
- UPDATE文にて動的に変更可能
- setup\_actorsでは監視対象とするユーザアカウントを設定可能

```
mysql> SELECT * FROM setup_objects;
```

OBJECT_TYPE	OBJECT_SCHEMA	OBJECT_NAME	ENABLED	TIMED
EVENT	mysql	%	NO	NO
EVENT	performance_schema	%	NO	NO
EVENT	information_schema	%	NO	NO
EVENT	%	%	YES	YES
FUNCTION	mysql	%	NO	NO
FUNCTION	performance_schema	%	NO	NO
FUNCTION	information_schema	%	NO	NO
FUNCTION	%	%	YES	YES
PROCEDURE	mysql	%	NO	NO
PROCEDURE	performance_schema	%	NO	NO
PROCEDURE	information_schema	%	NO	NO
PROCEDURE	%	%	YES	YES
TABLE	mysql	%	NO	NO
TABLE	performance_schema	%	NO	NO
TABLE	information_schema	%	NO	NO
TABLE	%	%	YES	YES
TRIGGER	mysql	%	NO	NO
TRIGGER	performance_schema	%	NO	NO
TRIGGER	information_schema	%	NO	NO
TRIGGER	%	%	YES	YES

```
20 rows in set (0.00 sec)
```



## MySQL 5.5のパフォーマンススキーマ

- 稼働統計情報を取得するフレームワークとして設計
- まずは低レベルな稼働統計情報の収集に重点
  - 負荷に耐えうるかの確認および新規に開発されたため
- 性能へのオーバーヘッドに課題があり、デフォルトではオフ

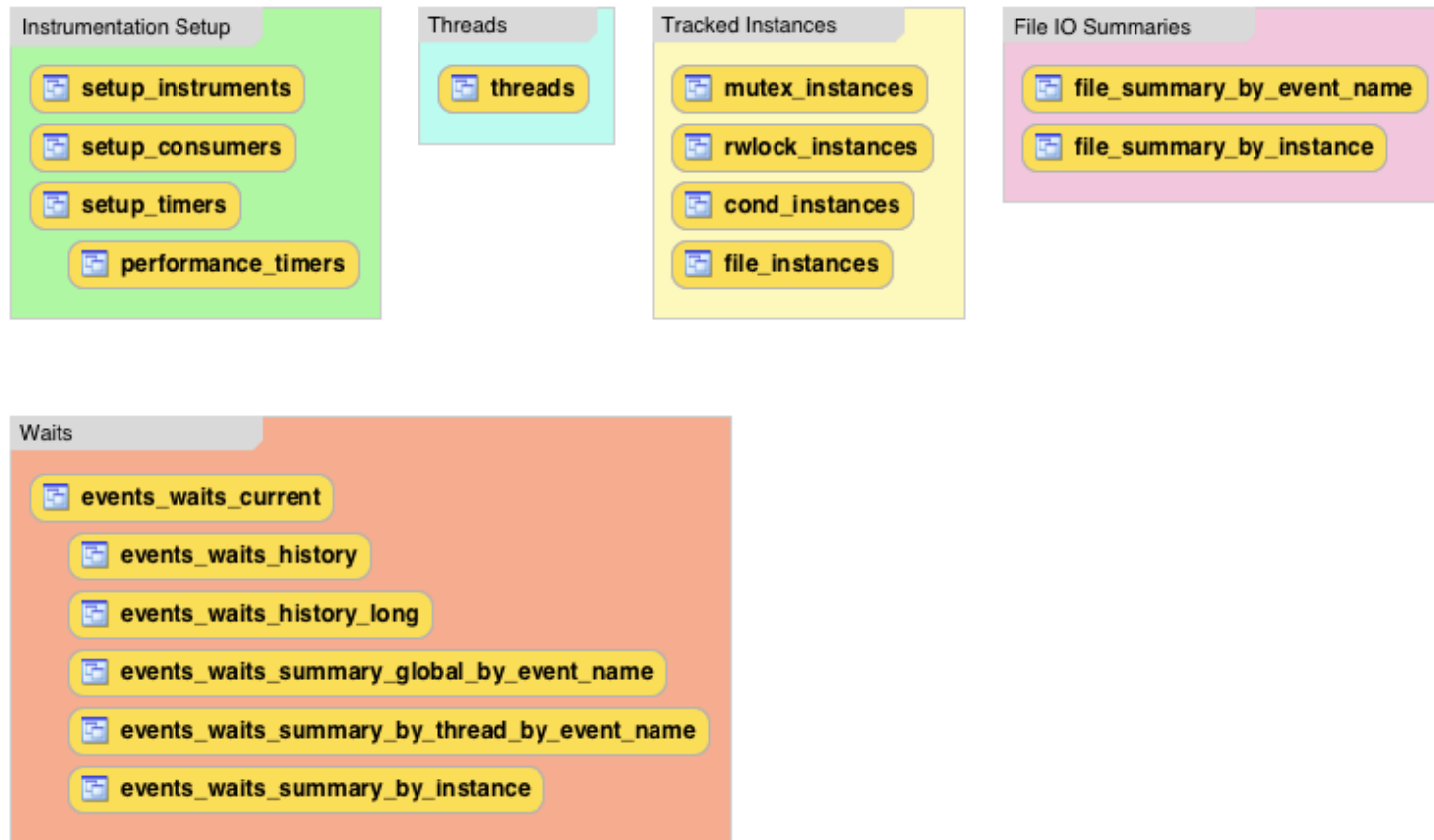
# MySQL 5.5のパフォーマンススキーマ

- 17 Tables
- 222 Instruments



Instrument	Event Class
File IO	wait/io/file/%
Mutexes	wait/synch/mutex/%
Read/Write Locks	wait/synch/rwlock/%
Conditions	wait/synch/cond/%

# MySQL 5.5のパフォーマンススキーマ





# MySQL 5.6のパフォーマンススキーマ

- 性能に関する課題を多数改善
- データベース管理者やアプリ開発者に役立つ情報の追加に重点
  - SQL文 / SQL文ダイジェスト
  - テーブルIO、インデックスIO、テーブルロック
  - ネットワークIO
- デフォルトでオン、ただしいくつかの詳細項目はオフ

# MySQL 5.6のパフォーマンススキーマ

- 52 Tables (+35)
- 545 Instruments (+323)



Instrument Type	Event Class
Statements	statement/%
Stages	stage/%
Table IO	wait/io/table/%
Table Locks	wait/lock/table/%
Network IO	wait/io/socket/%
Idle Timing	idle

# MySQL 5.6のパフォーマンススキーマ



- メモリ利用状況
- メタデータロック
- レプリケーション設定 & 状況
- プリペアドステートメント
- トランザクション
- ストアドプロシージャ & ファンクション





# MySQL 5.7.4 DMRのパフォーマンススキーマ

- 75 Tables (+23)
- 784 Instruments (+239)



Instrument Type	Event Class
Transactions	transaction
Memory	memory/%
Metadata Locks	wait/lock/metadata/%

# MySQL 5.7.4 DMRのパフォーマンススキーマ





# メモリ関連のInstruments

- 212のメモリ関連Instrumentsを追加 (5.7.4時点)
- 現在の利用量やそれまでの最高/最低利用量などを記録
- メモリ割り当てのレイテンシは記録しない
- 5つの新しいテーブル
  - memory\_summary\_by\_account\_by\_event\_name
  - memory\_summary\_by\_host\_by\_event\_name
  - memory\_summary\_by\_thread\_by\_event\_name
  - memory\_summary\_by\_user\_by\_event\_name
  - memory\_summary\_global\_by\_event\_name

## メモリ関連のInstruments – 全体サマリ

```
mysql> SELECT * FROM sys.memory_global_by_current_allocated\G
***** 1. row *****
      event_name: memory/performance_schema/internal_buffers
      current_count: 60
      current_alloc: 497.00 MiB
current_avg_alloc: 8.28 MiB
      high_count: 60
      high_alloc: 497.00 MiB
      high_avg_alloc: 8.28 MiB
***** 2. row *****
      event_name: memory/mysys/KEY_CACHE
      current_count: 3
      current_alloc: 8.00 MiB
current_avg_alloc: 2.67 MiB
      high_count: 3
      high_alloc: 8.00 MiB
      high_avg_alloc: 2.67 MiB
```

## メモリ関連のInstruments – スレッド別サマリ

```
mysql> select * from memory_by_thread_by_current_allocated\G
*****
1. row *****
      user: sql/main
      current_count: 2407
      current_alloc: 10.89 MiB
      current_avg_alloc: 4.63 KiB
      current_max_alloc: 8.00 MiB
      total_allocated: 30.55 MiB
      thread_id: 1
*****
2. row *****
      user: mem@localhost
      current_count: 1914
      current_alloc: 1.50 MiB
      current_avg_alloc: 824 bytes
      current_max_alloc: 816.67 KiB
      total_allocated: 9.25 GiB
      thread_id: 4336
```

# メモリ関連のInstruments – スレッド別詳細

```
mysql> SELECT event_name,  
->         sys.format_bytes(current_number_of_bytes_used) AS current_used  
-> FROM performance_schema.memory_summary_by_thread_by_event_name  
-> WHERE thread_id = 24  
-> ORDER BY current_number_of_bytes_used DESC;
```

event_name	current_used
memory/sql/Filesort_buffer::sort_keys	255.94 KiB
memory/sql/sp_head::main_mem_root	103.64 KiB
memory/mysys/IO_CACHE	64.05 KiB
memory/mysys/lf_dynarray	46.17 KiB
memory/mysys/array_buffer	24.20 KiB
memory/sql/thd::main_mem_root	23.95 KiB
memory/sql/String::value	16.13 KiB
memory/sql/TABLE	9.44 KiB
memory/sql/TABLE_SHARE::mem_root	8.70 KiB
memory/myisam/MI_INFO	7.07 KiB
memory/sql/THD::transactions::mem_root	4.02 KiB
memory/myisam/MYISAM_SHARE	3.29 KiB

# MySQL SYS Schema

パフォーマンススキーマとインフォメーションスキーマをシンプルなビューに

- データベース管理者のタスクを支援
  - 稼働統計や成長率などの主要な統計値の監視
  - 性能問題の検出、診断および改善
- 状況の詳細の確認をシンプルに
  - IO量の高いファイルや処理
  - コストの高いSQL文
  - テーブル、インデックス、スキーマの統計
  - レイテンシや待ち時間の分析
  - ロック
  - InnoDBの稼働統計





# MySQL SYS Schema

- ps\_helperから得られたフィードバックから改良
  - 80以上のビュー、自動更新、サーバーバージョン別
  - MySQL 5.5, 5.6, 5.7対応
- 他のデータベースにおけるSYS類似機能:
  - Oracle V\$表 (動的パフォーマンスビュー)
  - Microsoft SQL Server DMV (Dynamic Management Views)
  - IBM DB2 SYSIBMカタログ
- Workbench 6.1から設定、またはGitHubからダウンロード可能
  - Workbenchには簡単に利用可能なレポート機能あり



The screenshot shows the MySQL Workbench interface with the Performance Reports tool open. The left sidebar contains navigation menus for Management, Instance, Performance, and Information. The main window displays a tree view of performance reports, with 'Top I/O by File by Bytes' selected. The right pane shows a table of the top global I/O consumers by bytes usage.

### Top I/O by File by Bytes

Show the top global I/O consumers by bytes usage by file

File	Re...	Total Read	Avg Read	Writes	Total Written	Avg Writ...	Total	Write %
@@datadir/mysql/backup_history...	144044	559.35 MB	3.98 KB	704	490.52 KB	713 bytes	559.83 MB	0.09
@@datadir/mysql/backup_historyf...	3332	29.77 MB	9.15 KB	0	0 bytes	0 bytes	29.77 MB	0.00
@@datadir/mysql/backup_progres...	3332	11.76 MB	3.61 KB	0	0 bytes	0 bytes	11.76 MB	0.00
@@datadir/mysql/backup_history...	476	16.27 KB	35 bytes	950	32.47 KB	35 bytes	48.74 KB	66.62
@@datadir/mysql/backup_progres...	476	16.27 KB	35 bytes	951	32.50 KB	35 bytes	48.77 KB	66.64
@@datadir/mysql/proc.MyD	324	163.53 KB	517 bytes	52	76.30 KB	1.47 KB	239.82 KB	31.81
@@datadir/performance_schemas...	28	2.83 KB	104 bytes	0	0 bytes	0 bytes	2.83 KB	0.00
@@datadir/performance_schemaf...	28	15.24 KB	557 bytes	0	0 bytes	0 bytes	15.24 KB	0.00
@@datadir/mysql/proc.frm	28	12.27 KB	449 bytes	0	0 bytes	0 bytes	12.27 KB	0.00
@@datadir/performance_schemas...	28	3.72 KB	136 bytes	0	0 bytes	0 bytes	3.72 KB	0.00
@@datadir/mysql/user.frm	28	12.59 KB	460 bytes	0	0 bytes	0 bytes	12.59 KB	0.00
@@datadir/performance_schemat...	21	10.53 KB	513 bytes	0	0 bytes	0 bytes	10.53 KB	0.00
@@datadir/performance_schemat...	21	9.70 KB	473 bytes	0	0 bytes	0 bytes	9.70 KB	0.00
@@datadir/performance_schemaf...	21	2.58 KB	126 bytes	0	0 bytes	0 bytes	2.58 KB	0.00
@@datadir/performance_schemaf...	21	3.57 KB	174 bytes	0	0 bytes	0 bytes	3.57 KB	0.00
@@datadir/performance_schemaf...	21	3.66 KB	179 bytes	0	0 bytes	0 bytes	3.66 KB	0.00
@@datadir/mysql/procs_priv.frm	21	5.07 KB	247 bytes	0	0 bytes	0 bytes	5.07 KB	0.00
@@datadir/performance_schemaf...	21	7.46 KB	364 bytes	0	0 bytes	0 bytes	7.46 KB	0.00
@@datadir/mysql/proxies_priv.frm	21	4.36 KB	213 bytes	0	0 bytes	0 bytes	4.36 KB	0.00
@@datadir/performance_schemaf...	21	7.49 KB	365 bytes	0	0 bytes	0 bytes	7.49 KB	0.00
@@datadir/performance_schemaf...	21	13.44 KB	655 bytes	0	0 bytes	0 bytes	13.44 KB	0.00
@@datadir/performance_schemaf...	21	12.33 KB	601 bytes	0	0 bytes	0 bytes	12.33 KB	0.00
@@datadir/performance_schemaf...	21	12.33 KB	601 bytes	0	0 bytes	0 bytes	12.33 KB	0.00
@@datadir/performance_schemas...	21	2.41 KB	118 bytes	0	0 bytes	0 bytes	2.41 KB	0.00
@@datadir/performance_schemat...	21	7.21 KB	352 bytes	0	0 bytes	0 bytes	7.21 KB	0.00
@@datadir/performance_schemas...	21	3.13 KB	153 bytes	0	0 bytes	0 bytes	3.13 KB	0.00
@@datadir/performance_schemaf...	21	3.43 KB	167 bytes	0	0 bytes	0 bytes	3.43 KB	0.00



The screenshot shows the MySQL Workbench interface. The main window is titled "Performance Schema - Setup" for the instance "MyFirstConnection". The left sidebar contains a "Navigator" pane with categories: MANAGEMENT (Server Status, Client Connections, Users and Privileges, Status and System Variables, Data Export, Data Import/Restore), INSTANCE (Startup / Shutdown, Server Logs, Options File), and PERFORMANCE (Dashboard, Performance Reports, Performance Schema Setup). Below the Navigator are tabs for "Management" and "Schemas", and a section for "Information" showing "No object selected". The main content area features a large stopwatch icon and the text "Performance Schema Fully Enabled". Below this is a toggle switch currently set to "YES", with "NO" and "YES" labels. A "Show Advanced" button is in the top right. At the bottom, there are buttons for "Clear Event Tables" and "Reset to Factory Defaults". A descriptive text block explains that the Performance Schema allows instrumenting MySQL to collect statistics and performance data, and logging collected events into tables for analysis.



MySQL Workbench

Localhost x

File Edit View Query Database Server Tools Scripting Help

Navigator Query 1 Administration - Performance Sc... x

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

PERFORMANCE

- Dashboard
- Performance Reports
- Performance Schema Setup

Management Schemas

Information

No object selected

Object Info Session

Localhost Performance Schema - Setup

Hide Advanced

Easy Setup Introduction Instruments Consumers Actors & Objects Threads Options

### Performance Schema Basics

The performance schema collects data from various aspects of MySQL performance and gives very detailed information about what exactly is happening inside your MySQL database server. For each statement executed, the PS instruments will gather various statistics and timing information in different levels of granularity and from different subsystems, from network to disk storage, and keep them in the performance\_schema.events\_\* tables.

**Actors**

User User User

**Instruments**

Statements Stages Waits

**Database Objects**

DB object DB object DB object

**Consumers**

events\_statements tables  
events\_stages tables  
events\_waits tables

### Configuring Performance Schema

To control the trade-off between data collected and overhead, the performance schema gives you a few fine grained configuration options.

You can configure what, when and how much will be instrumented by the Performance Schema by tweaking three option categories:

- \* Actors - filters the users, hosts and DB objects to collect data for the performance\_schema. This was introduced in MySQL 5.6.
- \* Instruments - allow fine-tuning of what kind of stats are gathered for whatever is being monitored
- \* Consumers - toggle which of the performance\_schema.event\_\* tables should be filled

You can also use the simplified configuration interface for one-click setup of the performance schema for some common use cases.

Clear Event Tables Reset to Factory Defaults Revert Apply

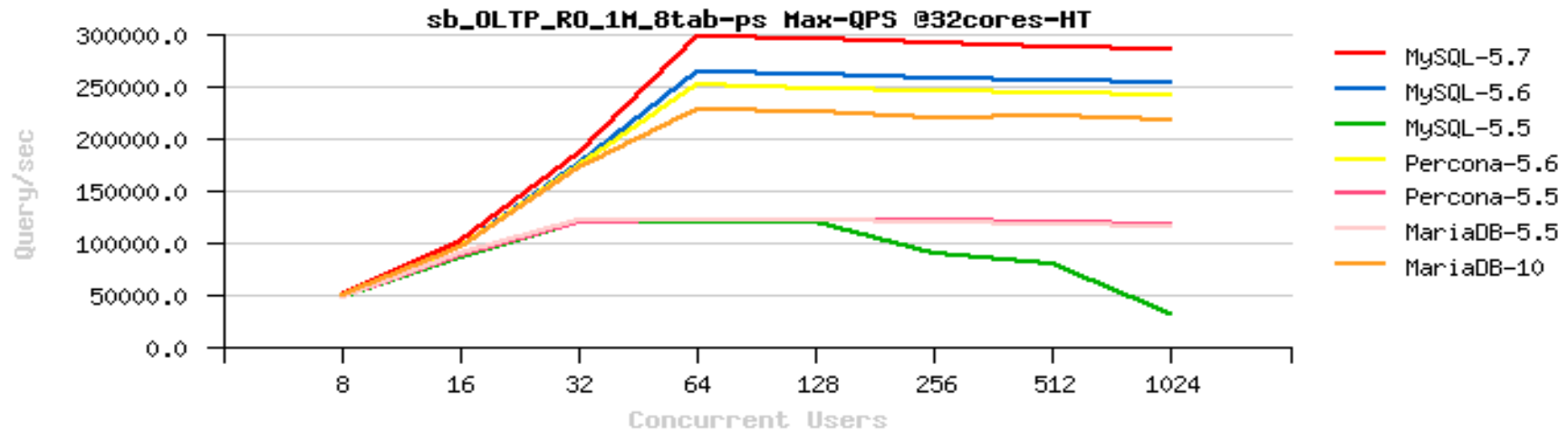


# MySQLベンチマーク vs 他製品

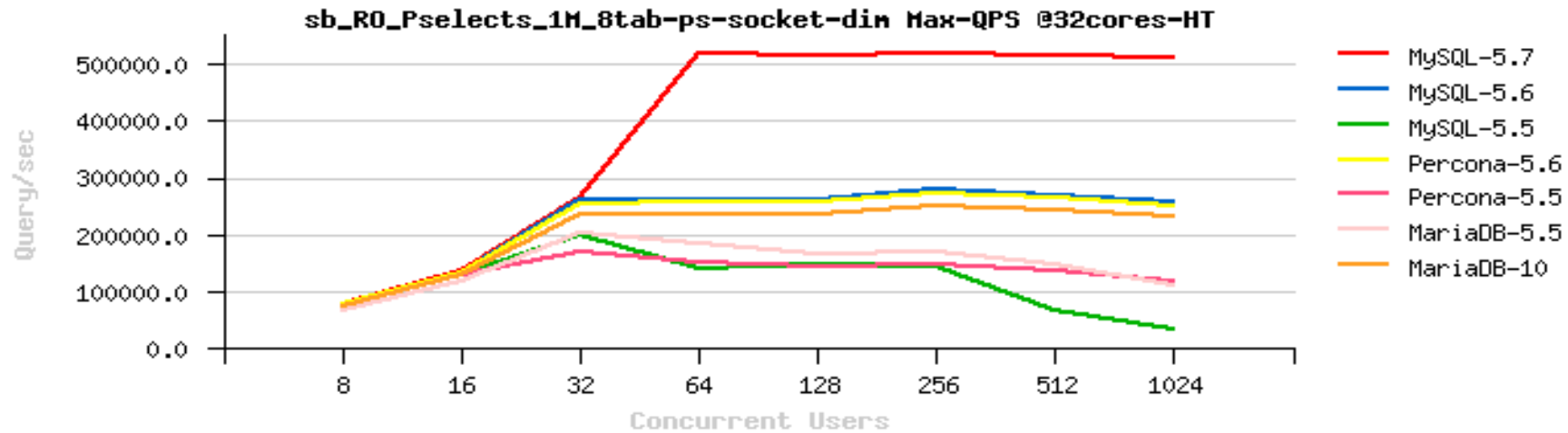


ORACLE®

# RO In-Memory @MySQL 5.7



# RO In-Memory @MySQL 5.7





**Hardware and Software**

**ORACLE®**

**Engineered to Work Together**



ORACLE®