



MySQL GUIツール Performance Report, Query Analyzer

MySQL Global Business Unit

Shinya Sugiyama / 杉山真也

MySQL Principal Sales Consult, MySQL Global Business Unit

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

MySQL GUIパフォーマンスツール: Agenda

1: データベース安定運用における課題

2: MySQL Workbench

MySQL Workbench概要

Performance Reportによるモニタリング

3: MySQL Enterprise Monitor

MySQL Enterprise Monitor概要

Query AnalyzerとQuery改善プロセス

4 補足情報

A high-quality photograph of a dolphin leaping from the surface of the ocean. The dolphin is captured mid-air, its body arched as it moves from the bottom right towards the top left. The water is a deep, clear blue, and the dolphin's sleek, dark grey skin is glistening with water droplets. The background is a vast, open expanse of the same blue water, creating a sense of depth and tranquility. The overall composition is dynamic and visually appealing.

データベース安定運用における課題

MySQL DBA チェックリスト

1. 本番データベースが使用可能であることを確認。 ✓
2. 24x365 MySQLのパフォーマンスを監視 ✓
3. MySQLのレプリケーションが正常に動作していることを確認 ✓
4. バックアップが正常に完了したことを確認 ✓
5. MySQLのディスク容量が不足しないように監視 ✓
6. 定期的に監視し、ブロッキングの問題を特定 ✓
7. データベーススキーマ変更の有無を確認 ✓
8. OSにおける、異常なイベントを確認 ✓
9. セキュリティの脆弱性をチェックする ✓
10. メモリ使用状況を監視して分析 ✓

MySQL DBA チャレンジ

- 「データベースが遅いです。どのようなチューニングが必要ですか? 」
- 「最もコストが高いクエリはどれでしょうか? 」
- 「インデックスは、最適化されていますか? 」
- 「レプリケーションの遅延が問題になっていませんか? 」
- 「最後のバックアップは成功しましたか? 」
- 「いつ頃、ディスクがいっぱいになりそうでしょうか? 」
- 「いつ頃、スケールアウトに追加のハードウェアが必要になりますか? 」
- 「データベーススキーマは変わりましたか? 」
- 「何か、対応しなければならない、セキュリティの脆弱性がありますか? 」



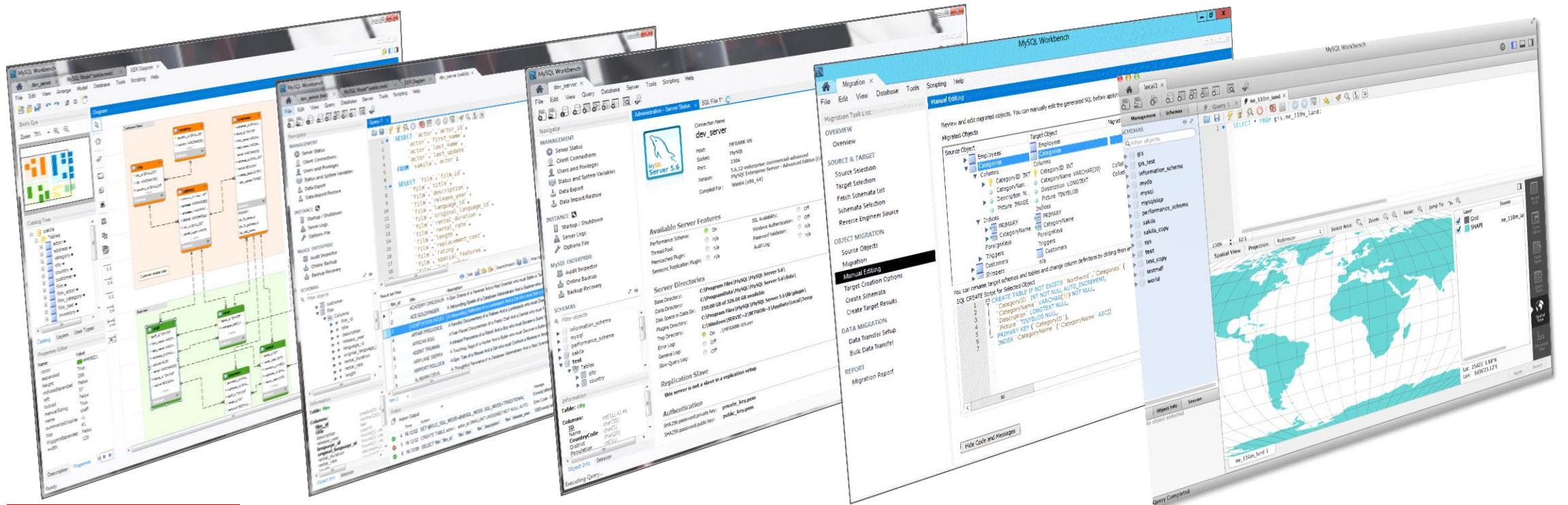
MySQL Workbench



MySQL Workbench

- MySQLの公式GUIツール
- MySQL Databaseの統合開発環境
- Windows, OS X, Linux 対応

MySQL Workbench は、データベースアーキテクト、開発者、DBA のための統合GUIツールです。



MySQL Workbenchの機能 (コミュニティ版&商用版共通)

• 管理

- 起動/停止、システム変数/ステータス変数確認、ログ確認、ユーザ管理、セッション管理、オブジェクト管理、データ編集、パフォーマンスダッシュボード&レポート、GIS Viewer, Dump/Import

• 設計

- E-R図作成、フォワードエンジニアリング、リバースエンジニアリング

• 開発

- SQLエディタ、SQL整形、SQLコード補完、SQLシンタックスハイライト、SQL Snippets(ステートメント再利用)、Visual Explain

• マイグレーション

- 各種RDBMSからのマイグレーション
- PostgreSQL, MS SQL Server, MS Access, SQLite, SQL Anywhere, Sybase ASE

MySQL Workbenchの機能 (商用版限定機能)

- DBドキュメント出力

- データベーススキーマの情報をドキュメント化(テーブル定義書を自動作成)

- MySQL Enterprise Backup用GUI

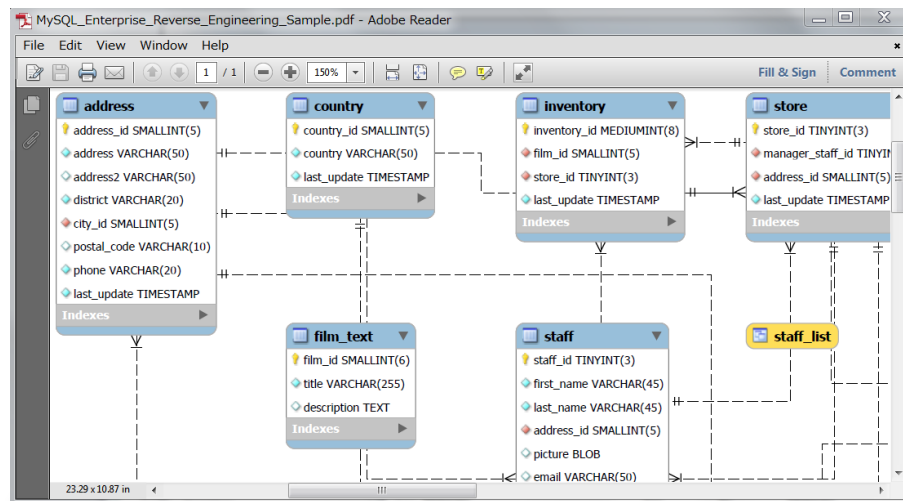
- バックアップジョブの管理

- データモデルの検証

- DB設計上の間違いや懸念事項を提示
- RDBMS全般/MySQL独自の観点でチェック

- MySQL Enterprise Audit用GUI

- 監査ログの分析、フィルタリング



The screenshot shows the 'Table customer' details in MySQL Workbench. The table is located in the 'Schema sakila' database. The columns and their properties are as follows:

Key	Column Name	Datatype	Not Null	Default	Comment
PK	customer_id	SMALLINT(5)	Yes		
	store_id	TINYINT(3)	Yes		
	first_name	VARCHAR(45)	Yes		
	last_name	VARCHAR(45)	Yes		
	email	VARCHAR(50)	No	NULL	
	address_id	SMALLINT(5)	Yes		
	active	TINYINT(1)	Yes	'1'	
	create_date	DATETIME	Yes		
	last_update	TIMESTAMP	Yes	CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	

The indexes for the table are:

Index Name	Columns	Primary	Unique	Type	Kind	Comment
PRIMARY	customer_id	Yes	No	PRIMARY		
idx_fk_store_id	store_id	No	No	INDEX		
idx_fk_address_id	address_id	No	No	INDEX		
idx_last_name	last_name	No	No	INDEX		

The screenshot shows the MySQL Workbench Administration - Server tab. The left sidebar has a red box around the 'PERFORMANCE' section, which includes 'Dashboard', 'Performance Reports', and 'Performance Schema Setup'. The main area displays server information for 'Misc' (MySQL Server 5.6), including connection details, server status (Running), load (0.0), and connections (5). A callout box with Japanese text points to the 'PERFORMANCE' section. The right sidebar shows a dashboard with various performance metrics: Traffic (0.0 B/s), Key Efficiency (91.7%), Selects per Second (0), InnoDB Buffer Usage (5.2%), InnoDB Reads per Second (0), and InnoDB Writes per Second (0). The bottom of the main area shows server directories and logs.

MySQL Workbench を利用したGUIベースの Performanceの確認

Server Status: Running
Load: 0.0
Connections: 5
Traffic: 0.0 B/s
Key Efficiency: 91.7%
Selects per Second: 0
InnoDB Buffer Usage: 5.2%
InnoDB Reads per Second: 0
InnoDB Writes per Second: 0

Available Server Features:
Performance Schema: On
Thread Pool: n/a
Memcached Plugin: On
Semisync Replication Plugin: n/a
SSL Availability: Off
PAM Authentication: Off
Password Validation: On (Policy: STRONG)
Audit Log: n/a

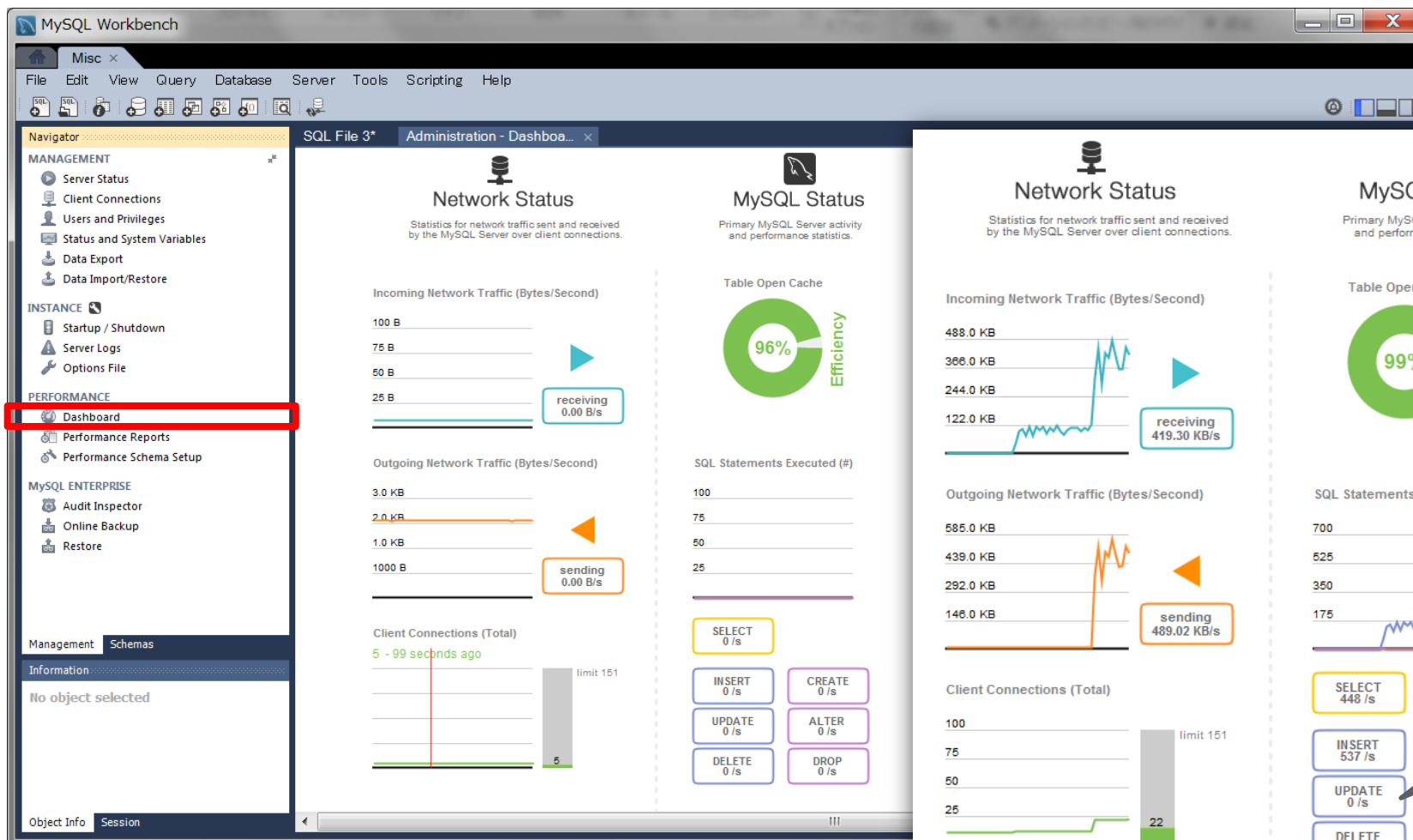
Server Directories:
Base Directory: /usr/local/mysql
Data Directory: /usr/local/mysql/data/
Disk Space in Data Dir: 71M of 6.7G available
Plugins Directory: /usr/local/mysql/lib/plugin/
Tmp Directory: /tmp
Error Log: On /usr/local/mysql/data/misc.err
General Log: On [Stored in database]
Slow Query Log: On [Stored in database]

this server is not a slave in a replication setup

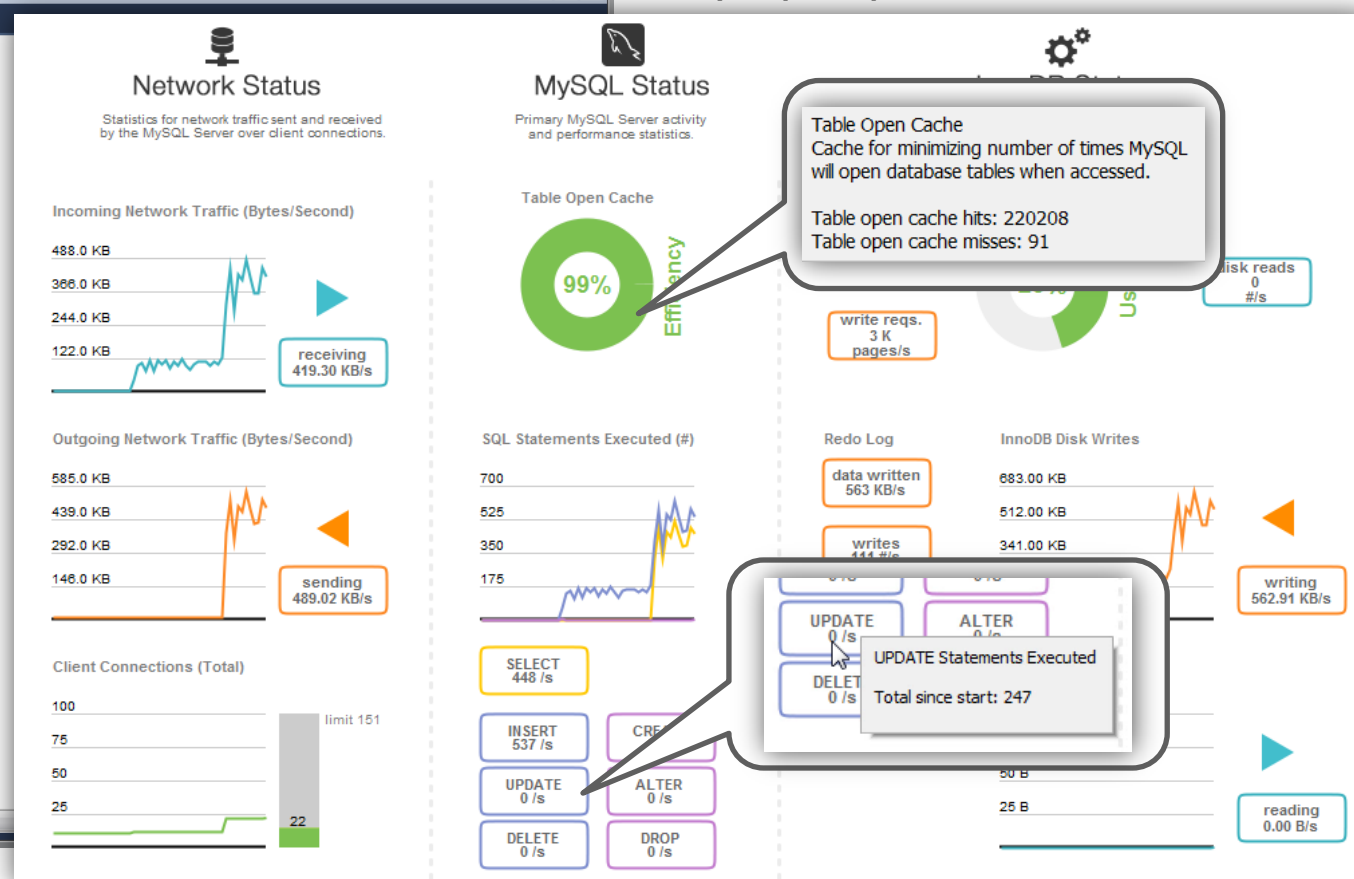
SHA256 password private key: private_key.pem

General Dashboard

ネットワーク, サーバ, InnoDBの状況を可視化出来、マウスオーバーすると詳細と説明を確認する事が可能
 一時的なデータで最大2.5分間のデータを表示可能。長期的なモニタリングはMySQL Enterprise Monitorがサポート

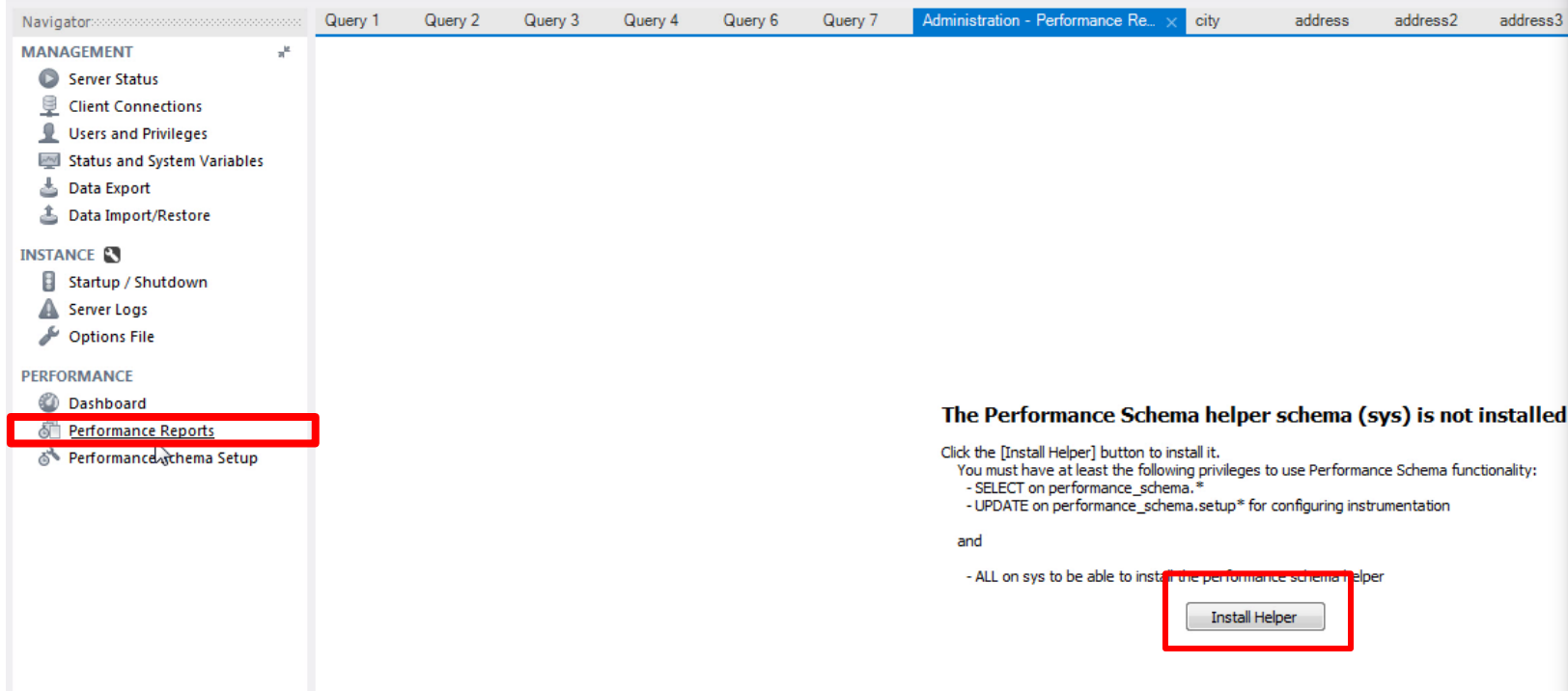


mysqlslapにて負荷をかけた状態



Performance Reports

Installs the SYS schema (Views, SPs, Functions)



The screenshot shows the MySQL Workbench Administration console. The left sidebar has three main sections: MANAGEMENT, INSTANCE, and PERFORMANCE. Under the PERFORMANCE section, 'Performance Reports' is highlighted with a red box. The main area displays a message: 'The Performance Schema helper schema (sys) is not installed'. Below this message, it lists the privileges required for installation: SELECT on performance_schema.*, UPDATE on performance_schema.setup*, and ALL on sys. An 'Install Helper' button is highlighted with a red box.

The Performance Schema helper schema (sys) is not installed

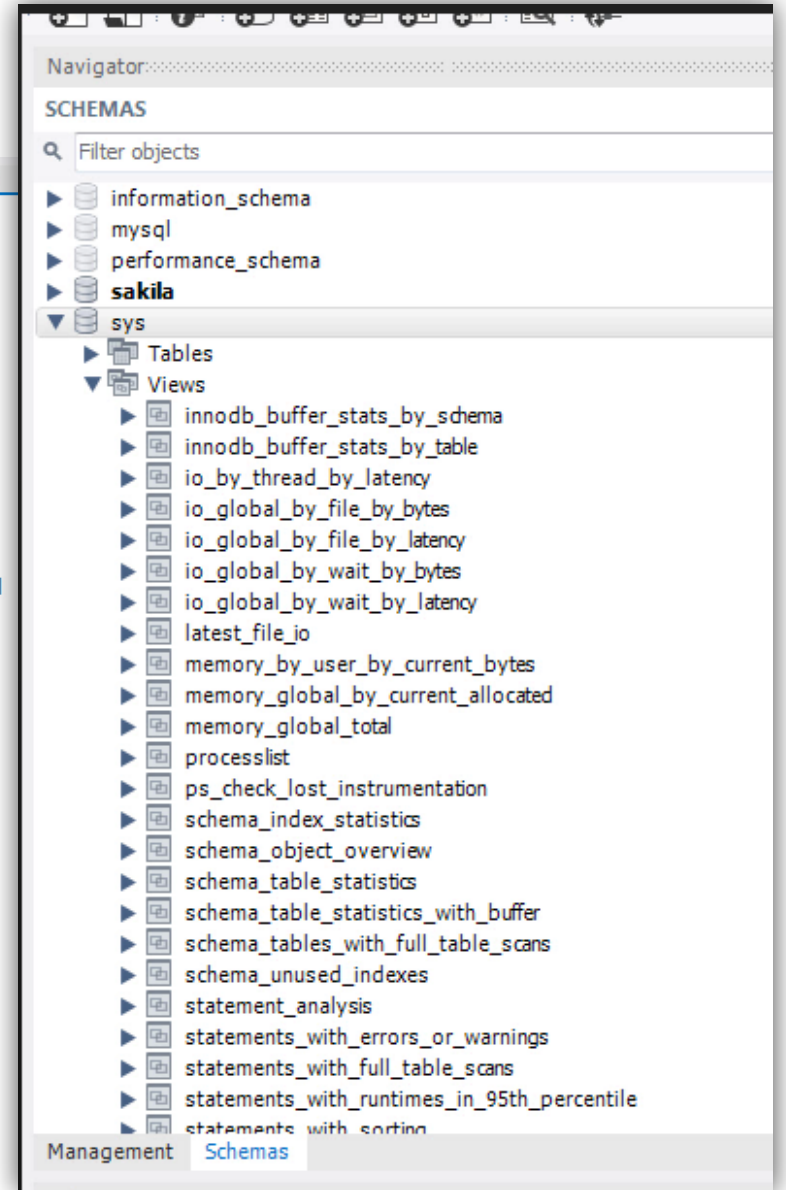
Click the [Install Helper] button to install it.

You must have at least the following privileges to use Performance Schema functionality:

- SELECT on performance_schema.*
- UPDATE on performance_schema.setup* for configuring instrumentation

and



- ALL on sys to be able to install the performance schema helper



The screenshot shows the MySQL Workbench Navigator. The 'SCHEMAS' section is expanded to show the 'sys' schema. Under 'sys', there are 'Tables' and 'Views'. The 'Views' section is expanded to show a list of views including innodb_buffer_stats_by_schema, innodb_buffer_stats_by_table, io_by_thread_by_latency, io_global_by_file_by_bytes, io_global_by_file_by_latency, io_global_by_wait_by_bytes, io_global_by_wait_by_latency, latest_file_io, memory_by_user_by_current_bytes, memory_global_by_current_allocated, memory_global_total, processlist, ps_check_lost_instrumentation, schema_index_statistics, schema_object_overview, schema_table_statistics, schema_table_statistics_with_buffer, schema_tables_with_full_table_scans, schema_unused_indexes, statement_analysis, statements_with_errors_or_warnings, statements_with_full_table_scans, statements_with_runtimes_in_95th_percentile, and statements with sorting.

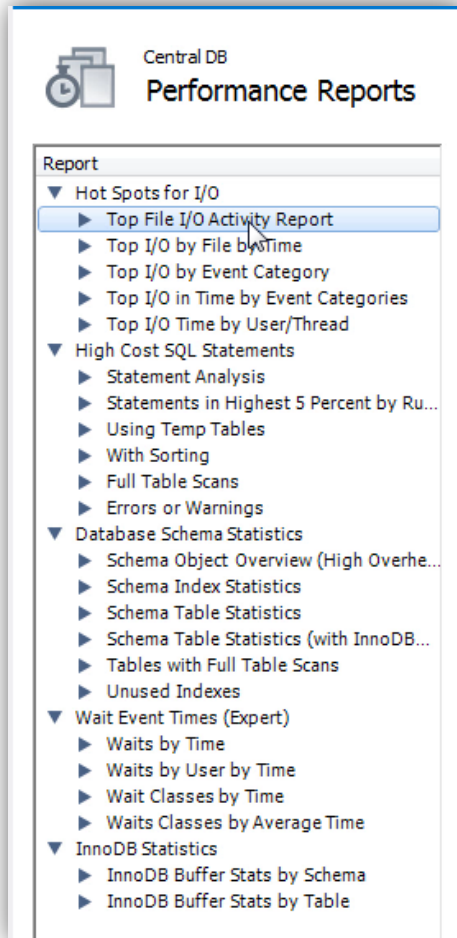
GUI: <http://www-jp.mysql.com/products/workbench/>

SCRIPT: <https://github.com/MarkLeith/mysql-sys>

 sys_56.sql	Adding the view innodb_lock_waits that displays a snapshot of the tra...	2 months ago
 sys_57.sql	Merge pull request #31 from JesperWisborgKrogh/dev/20141021_innodb_lo...	2 months ago

Performance Reports

5 カテゴリーのレポート



IO レポート

- For Files, Event Categories, User/Thread
- Quickly View, Sort
 - By #, Time, Reads, Writes, Percentages
- Look for Hot Spots
 - Use to determine system requirements, tuning, etc.

実行コストの高いSQL

- Look at query statements and statistics
- Look for Full Table Scans
- Frequency of execution
- Errors, Warnings -
- Long Runtimes – Total, Max, Ave
- Large numbers of Rows – Total, Max, Ave
- Usage of Temp Tables

データベーススキーマ統計

- Quickly Review Various Stats
 - Counts, Rows, Timing, Paging, Buffering, IOs
- Easily find Full Scans
- Spot Unused Indexes

待機イベント (For Experts)

- These reports show collected data
- Show the queries used to collect
- Provides Statistics
- Breaks out events by Users and Classes

InnoDBの統計

- Quick View
 - Aggregated
 - By Schema, By Table
- Easy to sort by
 - Allocation, Data
 - Pages, Pages hashed, Old Pages
 - Rows cached

Performance Reports Samples

InnoDB Buffer Stats by Schema

Summarizes the output of the INFORMATION_SCHEMA.INNODB_BUFFER_PAGE table, aggregating by schema

Schema	Allocated	Data	Pages	Pages Has...	Pages Old	Rows Cac...
sakila	4915200	3456342	300	300	300	9573
InnoDB System	180224	30923	11	11	11	86
mysql	114688	18154	7	7	7	193
innodb_memcache	49152	168	3	3	3	4
test	16384	288	1	1	1	6

Export...

Copy Selected

Copy Query

Refresh

Performance Reports

サーバ全体を見て問題点を発見

Queries, IO and Files, Buffers, Threads, Connections, Etc

MySQL Workbench Performance Reports interface. The 'Performance Reports' menu item is highlighted in red. The 'Top File I/O Activity Report' is displayed, showing a list of files and their I/O statistics. At the bottom, the 'Export...', 'Copy Selected', and 'Copy Query' buttons are also highlighted in red.

	A	B	C	D	E	F	G	H	I	J
1	Statement Analysis									
2										
3	Query	Full Table	Executed	Errors (#)	Warnings	Total Time	Max Time	Avg Time	Rows Sen	Avg. Row
4										
5	INSERT INTO t1 VALUES (...)		64549	0	0	1121682833	1016256	17377.23	0	0
6	SHOW GLOBAL STATUS	*	4963	0	0	16935549.5	30470.32	3412.36	1697346	342
7	SELECT @@ performance_sc		55	0	0	23089.63	942.85	419.81	55	1
8	SHOW SESSION VARIABLES L	*	21	0	0	59077.93	39723.6	2813.23	21	1
9	SHOW SESSION VARIABLES L	*	20	0	0	15428.35	1930.11	771.42	20	1
10	SET 'autocommit' = ?		16	0	0	23666.18	19438.85	1479.14	0	0
11	SELECT CURRENT_USER ()		16	0	0	16450.74	10947.53	1028.17	16	1
12	SET SESSION TRANSACTION		16	0	0	5084.78	1191.52	317.8	0	0
13	SELECT COUNT (*) FROM `	*	12	0	0	4695.76	786.54	391.31	12	1
14	USE `test`		10	0	0	2710.51	1030.13	271.05	0	0

Query, Full Tabl Rows Sorted (#), Sort Merge Passes (#), Digest
 INSERT INTO t1 VALUES (...), , 64549, 0, 0, 1121682833.32,
 1016256.34, 17377.23, 0, 0.0, 0, 0.0, 0, 0, 0, 0,
 a0583512c4eb718088979fe23a35a893

```
select * from sys.`x$io_global_by_file_by_latency`
select * from sys.`statements_with_temp_tables`
select * from sys.`schema_unused_indexes`
.....
```

Performance Schema Setup

MySQL Workbench

Misc (test) x Misc (test) x

File Edit View Query Database Server Tools Scripting Help

Navigator

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**

MySQL ENTERPRISE

- Audit Inspector
- Online Backup
- Restore

Management Schemas

Information

No object selected

Object Info Session

Ready

SQL File 3* Administration - Performa... x

Misc

Performance Schema - Setup

Easy Setup

performance_schemaの細かい設定を行う事が可能。

Performance Schema

Fully Enabled
Custom
Server Default
Disabled

Higher Performance Overhead
Lower / No Performance Overhead

The MySQL Performance Schema allows to:

- instrument MySQL to collect statistics and performance data
- log collected events into tables, so they can be analyzed

Use the switch above to change Performance Schema instrumentation or disable it.

Clear Event Tables

Full Reset to Factory Defaults

Performance Schema Setup

パフォーマンス収集に関するフラグをまとめて設定する事が可能。

Misc
Performance Schema - Setup

Hide Advanced

Easy Setup Introduction Instruments Consumers Actors & Objects Threads Options

Performance Schema

Fully Enabled
Custom
Server Default
Disabled

Higher Performance
Lower Performance

MySQL Workbench

Performance Warning

While enabling all performance_schema instrumentation allows collecting a lot of information from MySQL, it will also impose a significant performance and memory overhead on it. Do not enable this option if your server is a production server under heavy load, unless you know what you're doing.

Leave Unchanged
Enable Everything

setup_actors:	Determines the initial monitoring state for new foreground threads
setup_consumers:	Which event information can be stored and which are enabled
setup_instruments:	The classes of instrumented objects for which events can be collected
setup_objects:	Controls whether the Performance Schema monitors particular table objects.
setup_timers:	The current event timer

参照 : [22.9.2 Performance Schema Setup Tables](#)

参照 : [MySQL Performance: Why Performance Schema Overhead?..](#)

Performance Schema Setup

概要紹介: performance_schema.events_*テーブルへパフォーマンス関連情報を集約等



Misc

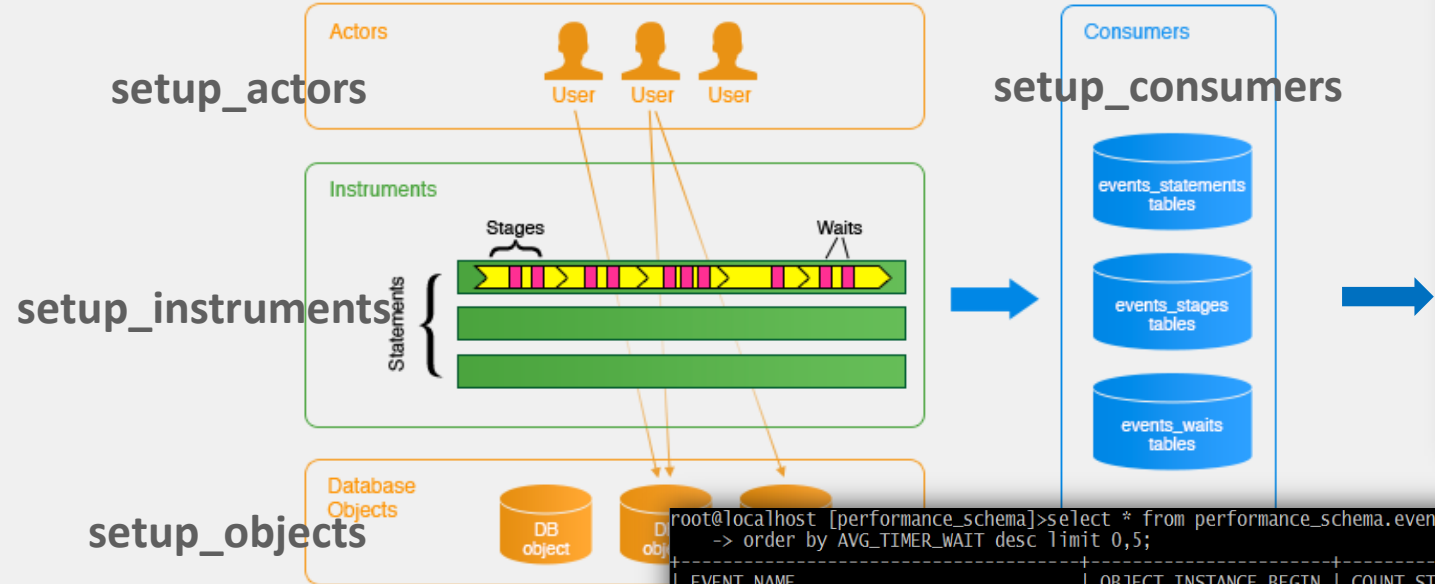
Performance Schema - Setup

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.



TABLE_SCHEMA	TABLE_NAME
performance_schema	events_stages_current
performance_schema	events_stages_history
performance_schema	events_stages_history_long
performance_schema	events_stages_summary_by_account_by_event_name
performance_schema	events_stages_summary_by_host_by_event_name
performance_schema	events_stages_summary_by_thread_by_event_name
performance_schema	events_stages_summary_by_user_by_event_name
performance_schema	events_stages_summary_global_by_event_name
performance_schema	events_statements_current
performance_schema	events_statements_history
performance_schema	events_statements_history_long
performance_schema	events_statements_summary_by_account_by_event_name
performance_schema	events_statements_summary_by_digest
performance_schema	events_statements_summary_by_host_by_event_name
performance_schema	events_statements_summary_by_thread_by_event_name
performance_schema	events_statements_summary_by_user_by_event_name
performance_schema	events_statements_summary_global_by_event_name
performance_schema	events_waits_current
performance_schema	events_waits_history
performance_schema	events_waits_history_long
performance_schema	events_waits_summary_by_account_by_event_name
performance_schema	events_waits_summary_by_host_by_event_name
performance_schema	events_waits_summary_by_instance
performance_schema	events_waits_summary_by_thread_by_event_name
performance_schema	events_waits_summary_by_user_by_event_name
performance_schema	events_waits_summary_global_by_event_name

```
root@localhost [performance_schema]>select * from performance_schema.events_waits_summary_by_instance
-> order by AVG_TIMER_WAIT desc limit 0,5;
```

EVENT_NAME	OBJECT_INSTANCE_BEGIN	COUNT_STAR	SUM_TIMER_WAIT	MIN_TIMER_WAIT	AVG_TIMER_WAIT	MAX_TIMER_WAIT
wait/io/file/innodb/innodb_data_file	140377561247488	3	72725448510	13894140	24241816170	72641370360
wait/io/file/sql/slow_log	140377561251008	4	96633499950	551460	24158374890	96600784410
wait/io/file/innodb/innodb_data_file	140377561248192	9	171773780490	18338190	19085975610	147748395600
wait/io/file/innodb/innodb_data_file	140377561250304	3	52512466110	9375600	17504155110	52472409600
wait/io/file/innodb/innodb_data_file	140377561243264	15	204108590010	10313940	13607239230	132510151020

Configuring Performance Schema



Performance Schema Setup

performance_schema.setup_instruments テーブルの設定をGUIから設定変更可能

Misc
Performance Schema - Setup

Easy Setup | Introduction | **Instruments** | Consumers | Actors & Objects | Threads | Options

Select Enabled Instrumentation Points

An instrument refers to a specific section of the MySQL code that has been enabled for monitoring. When an instrument is enabled and is executed on the server, it will generate events that can be used for monitoring.

The top level elements on the table below represent the "Event Type" generated by the instrument.

The enabled column indicates whether the events for the instrument are generated. The timed column indicates whether information about the event duration is recorded.

Instrument	Enabled	Timed
idle	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▼ stage	<input type="checkbox"/>	<input type="checkbox"/>
mysys	<input type="checkbox"/>	<input type="checkbox"/>
sql	<input type="checkbox"/>	<input type="checkbox"/>
▼ statement	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
abstract	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
sql	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▼ wait	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
lock	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
io	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
synch	<input type="checkbox"/>	<input type="checkbox"/>

Hide Advanced

The instrument name space has a tree-like structure. The components of an instrument name from left to right provide a progression from more general to more specific.

Instrument	Enabled	Timed
idle	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ stage	<input type="checkbox"/>	<input type="checkbox"/>
▶ statement	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▼ wait	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ lock	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▼ io	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ table	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ socket	<input type="checkbox"/>	<input type="checkbox"/>
▼ file	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ myisammrg	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ mysys	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▼ innodb	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
innodb_temp_file	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
innodb_log_file	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
innodb_data_file	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ myisam	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ sql	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ csv	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ archive	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
▶ synch	<input type="checkbox"/>	<input type="checkbox"/>

イベント収集可能なインストルメントオブジェクトのリスト

```
root@localhost [sys]>select * from performance_schema.setup_instruments
-> where NAME like 'wait/io/file/innodb%';
```

NAME	ENABLED	TIMED
wait/io/file/innodb/innodb_data_file	YES	YES
wait/io/file/innodb/innodb_log_file	YES	YES
wait/io/file/innodb/innodb_temp_file	YES	YES

参照: [22.4 Performance Schema Instrument Naming Conventions](#)



Performance Schema Setup

performance_schema.setup_consumersテーブルの設定を変更可能。

Easy Setup | Introduction | Instruments | **Consumers** | Actors & Objects | Threads | Options

Select performance_schema Event Types to be Collected

Performance Schema generates execution events only if a consumer for such events is active. A consumer is considered active if it is enabled and the consumers it depends on are active based on the next hierarchy:

- Global Instrumentation is the top level consumer.
- Thread Instrumentation and Statement Digest depend on Global Instrumentation.
- The Current consumers for Statement, Stage and Wait events depend on Thread Instrumentation
- The History and History Long consumers depend on it's associated Current consumer

The options you toggle determine which of the performance_schema.events_* tables are fed with data

Global instrumentation

Thread instrumentation

Statement Events

Current events

History (5 events)

Long History (100 events)

Stage Events

Current events

History (5 events)

Long History (100 events)

Wait Events

Current events

History (5 events)

Long History (100 events)

Digesting normalizes statements in a way that permits grouping similar statements and collecting information about how often they occur.

Statements digest

```
global_instrumentation
thread_instrumentation
events_waits_current
events_waits_history
events_waits_history_long
events_stages_current
events_stages_history
events_stages_history_long
events_statements_current
events_statements_history
events_statements_history_long
statements_digest
```

階層

どのイベントを取得し保存するか定義

```
root@localhost [sys]>select * from performance_schema.setup_consumers;
+-----+-----+
| NAME                                | ENABLED |
+-----+-----+
| events_stages_current                | NO      |
| events_stages_history                | NO      |
| events_stages_history_long           | NO      |
| events_statements_current            | YES     |
| events_statements_history            | YES     |
| events_statements_history_long       | YES     |
| events_waits_current                 | YES     |
| events_waits_history                 | YES     |
| events_waits_history_long            | YES     |
| global_instrumentation               | YES     |
| thread_instrumentation               | YES     |
| statements_digest                    | YES     |
+-----+-----+
```


Performance Schema Setup

performance_schemasetup_actors, performance_schema.setup_objectsテーブル設定を変更可能
新規コネクションはsetup_actorsにてフィルタリングされ、threadのinstrumentedがYESになりモニタリング対象になります。

Misc Performance Schema - Setup

Hide Advanced

Easy Setup Introduction Instruments Consumers **Actors & Objects** Threads Options

Filter Users and Objects to be Monitored

Users

Performance Schema allows defining filters to determine the connections for which data will be collected. New connections having a user@host matching an entry below will be enabled for monitoring.

Use % to indicate either any user or any host.

User	Host
%	%

Add

Remove

Database Objects

Performance Schema allows defining filters to determine the objects for which data will be collected. Any schema/object matching a combination defined below will be enabled for monitoring.

Use % to indicate either any schema or any object.

The enabled column indicates whether events for the matching objects are instrumented. The timed column indicates whether information about the events duration is recorded.

Type	Schema	Object	Enab...	Timed
TABLE	mysql	%	<input type="checkbox"/>	<input type="checkbox"/>
TABLE	performance_schema	%	<input type="checkbox"/>	<input type="checkbox"/>

新規モニタリング開始対象スレッドの定義

Monitoring for all foreground threads is enabled by default

```
root@localhost [sys]>SELECT * FROM performance_schema.setup_actors;
+-----+-----+-----+
| HOST | USER | ROLE |
+-----+-----+-----+
| %    | %    | %    |
+-----+-----+-----+
1 row in set (0.00 sec)
```

どのObjectをモニタリングするかの定義可能

```
root@localhost [performance_schema]>SELECT * FROM performance_schema.setup_objects;
+-----+-----+-----+-----+-----+
| OBJECT_TYPE | OBJECT_SCHEMA | OBJECT_NAME | ENABLED | TIMED |
+-----+-----+-----+-----+-----+
| TABLE     | mysql        | %          | NO      | NO    |
| TABLE     | performance_schema | %        | NO      | NO    |
| TABLE     | information_schema | %        | NO      | NO    |
| TABLE     | %           | %          | YES     | YES   |
+-----+-----+-----+-----+-----+
```

Performance Schema Setup

performance_schema.threadsテーブルの設定を変更可能 (特定のスレッドのみ監視する事が可能)

Threads to Instrumentsの変更は、Consumersの値がONになっている場合にのみモニタリングが有効



Misc

Performance Schema - Setup

Easy Setup Introduction Instruments Consumers Actors & Objects **Threads** Options

Threads to Instrument

Performance Schema allows enabling/disabling the monitoring of events that occur on specific threads in the server.

Changes on the instrumentation status are only effective if Thread Instrumentation is enabled on the Consumers Tab.

Id	Name	Instrumented	Type	Process Id	Account	Command	Time
1	thread/sql/main	<input checked="" type="checkbox"/>	BG		<system>		9312
2	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
3	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
4	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
5	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
6	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
7	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
8	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
9	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
10	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
11	thread/innodb/io_handler_thread	<input checked="" type="checkbox"/>	BG		<system>		
13	thread/innodb/srv_error_monitor_thread	<input checked="" type="checkbox"/>	BG		<system>		
14	thread/innodb/srv_monitor_thread	<input checked="" type="checkbox"/>	BG		<system>		
15	thread/innodb/srv_master_thread	<input checked="" type="checkbox"/>	BG		<system>		
17	thread/innodb/srv_lock_timeout_thread	<input checked="" type="checkbox"/>	BG		<system>		
18	thread/innodb/srv_purge_thread	<input checked="" type="checkbox"/>	BG		<system>		
19	thread/innodb/page_cleaner_thread	<input checked="" type="checkbox"/>	BG		<system>		
20	thread/sql/signal_handler	<input checked="" type="checkbox"/>	BG		<system>		
23	thread/sql/one_connection	<input checked="" type="checkbox"/>	FG	3	root@localhost	Sleep	744
24	thread/sql/one_connection	<input checked="" type="checkbox"/>	FG	4	admin@192...	Sleep	230

Instrument All Instrument None

```
root@localhost [performance_schema]>desc performance_schema.threads;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| THREAD_ID | bigint(20) unsigned | NO | | NULL | |
| NAME | varchar(128) | NO | | NULL | |
| TYPE | varchar(10) | NO | | NULL | |
| PROCESSLIST_ID | bigint(20) unsigned | YES | | NULL | |
| PROCESSLIST_USER | varchar(16) | YES | | NULL | |
| PROCESSLIST_HOST | varchar(60) | YES | | NULL | |
| PROCESSLIST_DB | varchar(64) | YES | | NULL | |
| PROCESSLIST_COMMAND | varchar(16) | YES | | NULL | |
| PROCESSLIST_TIME | bigint(20) | YES | | NULL | |
| PROCESSLIST_STATE | varchar(64) | YES | | NULL | |
| PROCESSLIST_INFO | longtext | YES | | NULL | |
| PARENT_THREAD_ID | bigint(20) unsigned | YES | | NULL | |
| ROLE | varchar(64) | YES | | NULL | |
| INSTRUMENTED | enum('YES','NO') | NO | | NULL | |
+-----+-----+-----+-----+-----+-----+
```

```
root@localhost [performance_schema]>SELECT * FROM threads limit 1\G
***** 1. row *****
      THREAD_ID: 1
      NAME: thread/sql/main
      TYPE: BACKGROUND
      PROCESSLIST_ID: NULL
      PROCESSLIST_USER: NULL
      PROCESSLIST_HOST: NULL
      PROCESSLIST_DB: NULL
      PROCESSLIST_COMMAND: NULL
      PROCESSLIST_TIME: 9951
      PROCESSLIST_STATE: System lock
      PROCESSLIST_INFO: INTERNAL DDL LOG RECOVER IN PROGRESS
      PARENT_THREAD_ID: NULL
      ROLE: NULL
      INSTRUMENTED: YES
1 row in set (0.01 sec)
```



Performance Schema Setup

performance_schema.setup_timersテーブルの設定をGUIから設定変更が可能。

Instrumentsのタイプ毎に異なるタイマーを設定可能



Misc

Performance Schema - Setup

Easy Setup Introduction Instruments Consumers Actors & Objects Threads **Options**

Event Timers

Instruments measure the duration of events, for that they can use different timers. A timer has characteristics that need to be considered when setting up the timer to be used on the different instruments:

- Frequency: Indicates the number of timer units per second.
- Resolution: Indicates the size used to increase a timer value at a time.
- Overhead: Minimal number of cycles of overhead to obtain one timing.

Here you can configure which timer will be used for each instrument type.

Idle Events	Microseconds	(Occurs a million times in a second - frequency: 1000000, resolution: 1, overhead: 76.)	▼
Stage Events	Nanoseconds	(Occurs a billion times in a second - frequency: 1000000000, resolution: 1, overhead: 76.)	▼
Statement Events	Nanoseconds	(Occurs a billion times in a second - frequency: 1000000000, resolution: 1, overhead: 76.)	▼
Wait Events	Cycles	(Occurs based on the CPU speed - frequency: 2622626865, resolution: 1, overhead: 24)	▼

```
root@localhost [performance_schema]>desc performance_schema.setup_timers;
```

Field	Type	Null	Key	Default	Extra
NAME	varchar(64)	NO		NULL	
TIMER_NAME	enum('CYCLE','NANOSECOND','MICROSECOND','MILLISECOND','TICK')	NO		NULL	

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

```
root@localhost [performance_schema]>select * from performance_schema.setup_timers;
```

NAME	TIMER_NAME
idle	MICROSECOND
wait	CYCLE
stage	NANOSECOND
statement	NANOSECOND

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

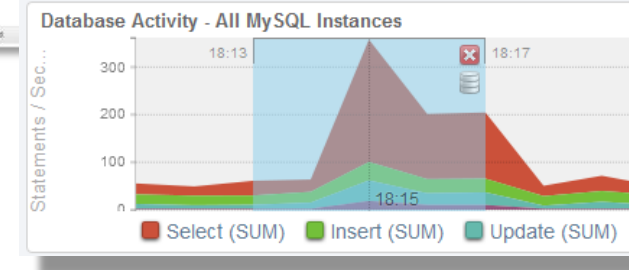
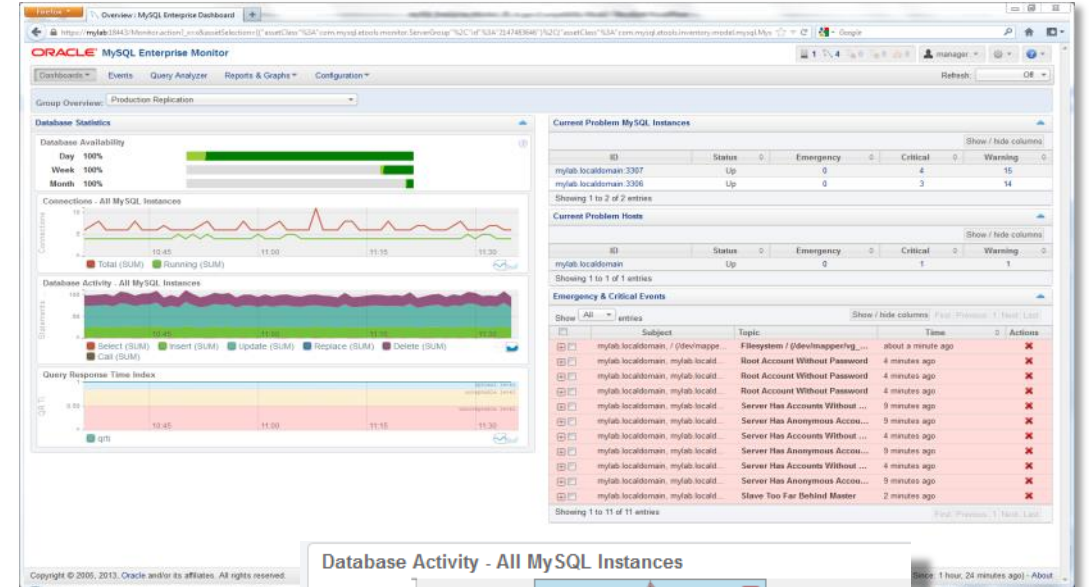

MySQL Enterprise Monitor Query Analyzer



MySQL Enterprise Monitor

- パフォーマンスと可用性の監視
- 問題のあるSQL文の検知
- ディスク監視と容量プランニング
- クラウド対応アーキテクチャ
 - ポリシーベースの設定
 - エージェント導入不要
- MySQL監視を10分以内で開始可能

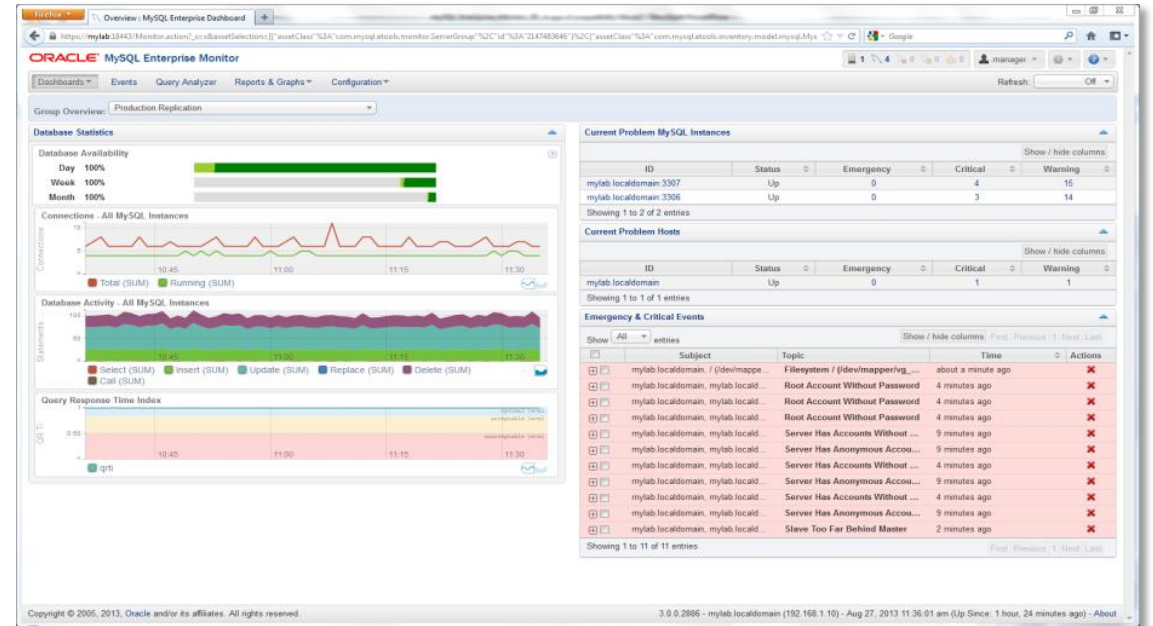
参照: [MySQL Enterprise Monitor](#)



	Current	Worst	Subject	Topic
<input type="checkbox"/>	🚫	🚫	mylab.localdomain, mylab.localdomain:3306	Root Account Without Password
<input type="checkbox"/>	🚫	🚫	mylab.localdomain, mylab.localdomain:3306	Server Has Accounts Without A Password
<input type="checkbox"/>	✅	🚫	mylab.localdomain, mylab.localdomain:3306	Average Statement Execution Time Excess...
<input type="checkbox"/>	✅	🚫	mylab.localdomain, mylab.localdomain:3306	SQL Statement Generates Errors or Warnings
<input type="checkbox"/>	🚫	🚫	mylab.localdomain, mylab.localdomain:3306	Server Has Anonymous Accounts
<input type="checkbox"/>	✅	🚫	mylab.localdomain, mylab.localdomain:3306	MySQL Instance Is Experiencing A Query P...
<input type="checkbox"/>	⚠️	⚠️	mylab.localdomain, mylab.localdomain:3306	InnoDB Log Buffer Flushed To Disk After Ea...
<input type="checkbox"/>	⚠️	⚠️	mylab.localdomain, mylab.localdomain:3306	User Has Rights To Database That Does Not...

Enterprise Monitor Dashboard

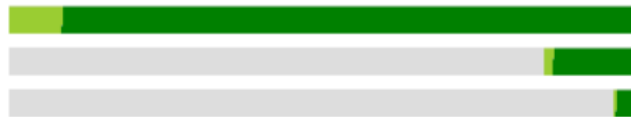
- サービスレベルのモニタリング
- リアルタイムパフォーマンス監視
- 警告と通知による迅速な対応
- ベストプラクティスアドバイザー
- 全MySQL サーバを視覚的に管理



Database Statistics

Database Availability

Day 100%
Week 100%
Month 100%



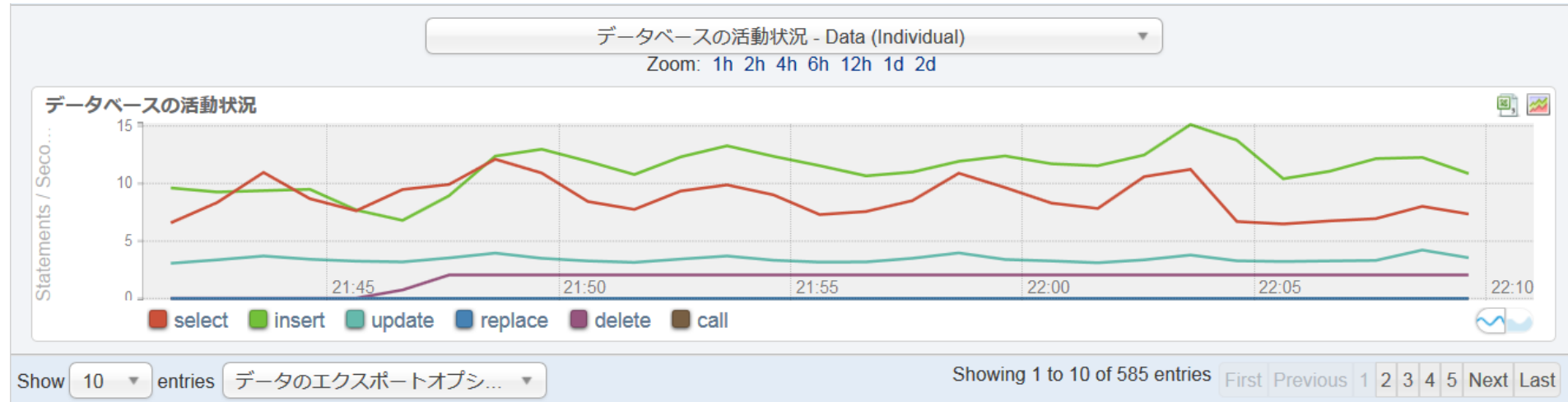
"The MySQL Enterprise Monitor is an absolute must for any DBA who takes his work seriously."

- Adrian Baumann, System Specialist
Federal Office of Information Technology &
Telecommunications

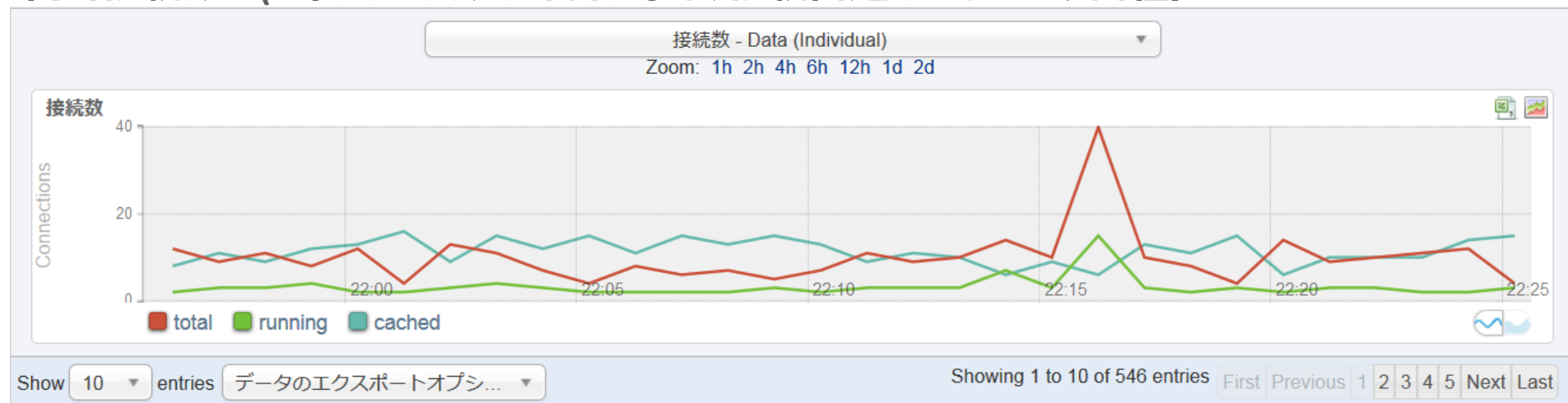


接続状況

データベースのスループットと活動状況の確認 (全体的な傾向確認)



同時接続数 (コネクション数の確認による、接続関連のパラメータ調性)



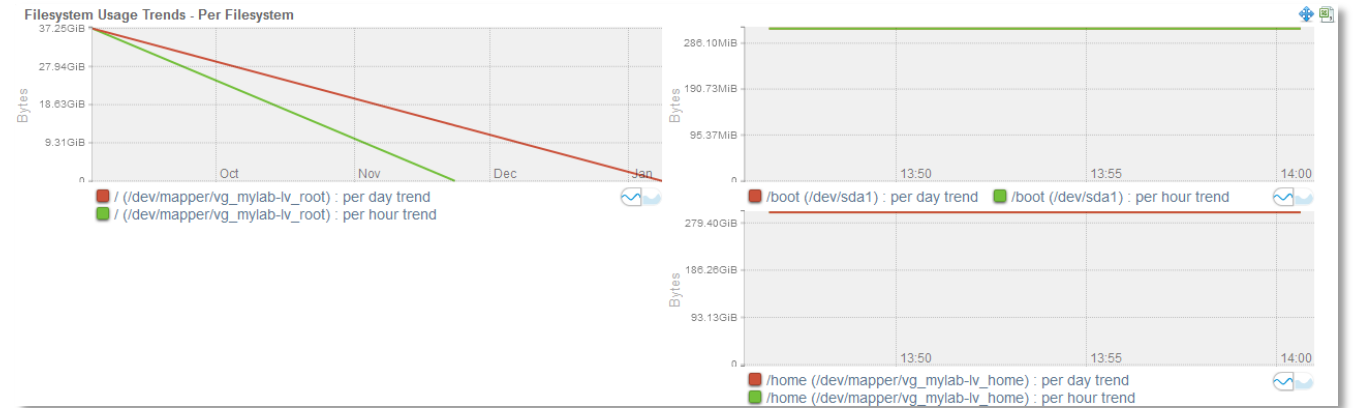
InnoDB Monitoring

- 主要なパフォーマンス・メトリックを監視
- ロック発生状況の把握
- 設定のアドバイスを取得
- バッファプールの使用状況を確認



Trends & Predictive Problem Detection

- ビジュアルトレンド分析
- 異常値の特定
- 問題発生を回避
- 差し迫った容量の問題に対処



Topic: Filesystem / (/dev/mapper/vg_mylab-lv_root) Running Out Of Space In about 13 days

Categories: Operating System

Advisor: Filesystem Free Space

Current State: Open

Current Status: Critical

Worst Status: Critical

Auto-Closes by Default: Yes

Last Checked: Sep 3, 2013 2:10:39 PM

Worst Alarm Time: Sep 3, 2013 2:08:06 PM

Notes:

No notes provided.

Details:

Problem Description

Databases use disks and filesystems to store data, indexes, logs, and other artifacts. When space gets low, it can adversely affect the performance of your system, and in extreme cases may cause your application to halt or crash.

Advice

Investigate why filesystem / (/dev/mapper/vg_mylab-lv_root) has a 24-hour average growth rate of 2.8 GiB. At that rate, the 28 GiB remaining (out of 49 GiB) will run out in about 13 days, around September 16, 2013.

Consider archiving and deleting large files that are no longer needed, as well as temporary files (e.g. files in /tmp on Linux and %TEMP% on Windows). Look for files that are growing rapidly and consider alternatives (e.g. rotating logs, moving those files to another filesystem, etc).

Links and Further Reading

- [MySQL Manual: Managing Disk I/O and File Space for InnoDB Tables](#)
- [Article: Managing Disk Space in Linux](#)
- [Oracle Solaris Administration: Displaying Information About Files and Disk Space](#)

Agentを導入する事で、リモートホストのCPU, Memory, Disk等の情報も取得しモニタリングする事が可能。

<input type="checkbox"/>	現在 <input type="checkbox"/>	最重要 <input type="checkbox"/>	サーバー	説明	時刻	アクション
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	misc:3306	MyISAM の同時挿入が適切に設定されていない可能性があります	about an hour ago	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	misc:3306	名前の大文字小文字を区別するためデータベースの互換性が低いと考えられます	about an hour ago	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	misc:3306	max_prepared_stmt_countにデフォルト値が使用されています	about an hour ago	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	misc:3306	シンボリックリンクが有効です	about an hour ago	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	misc:3306	Slave SQL Processing Not Multi-Threaded	about an hour ago	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	misc:3306	Binary Log Debug Information Disabled	about an hour ago	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	misc:3306	一般クエリログが有効になっています	about an hour ago	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	misc:3306	InnoDBログバッファがトランザクション後毎回ディスクにフラッシュされています	about an hour ago	<input checked="" type="checkbox"/>

Topic: InnoDBログバッファがトランザクション後毎回ディスクにフラッシュされています

Categories: パフォーマンス

Advisor: InnoDBログバッファがトランザクション後毎回ディスクにフラッシュされています

Current State: オープン

Closed By: Closed:

ワーストステータス Warning

Auto-Closes by Default: Yes

Worst Alarm Time: 2015/01/24 14:25:17

Notes:

メモの入力はありません。

Details:

問題の説明

デフォルトでは、InnoDBのログバッファはログファイルにトランザクションのコミットごとに書き出され、ディスクへのフラッシュ操作がログファイルに対して実行されます。この仕組みによりACIDに準拠しています。クラッシュ時、1秒程度のトランザクションをロストすることが許容できるなら、**innodb flush log at trx commit**を0または2に設定することでパフォーマンスを向上できます。この値を2に設定した場合、オペレーティングシステムがクラッシュするか電源異常が発生したときのみ、最後の1秒分のトランザクションが消去される場合があります。これはスレーブサーバーでは有効です。1秒程度のデータロストは、必要に応じてマスタサーバーから復元することができるからです。

アドバイス

my.cnf/my.iniファイルにおいて **innodb_flush_log_at_trx_commit=2** と設定し、サーバーを再起動してください。

警告: ACIDに準拠する場合は値は1にする必要があります。値を2に設定した場合、オペレーティングシステムがクラッシュするか電源異常が発生したとき、最後の1秒分のトランザクションが消失する場合があります。これは使用中のアプリケーションや環境によっては重大ではない場合もあります。特にスレーブサーバーでは、1秒間分のデータをロストしたとしても、マスタから復元することができるからです。

カスタマイズ & 拡張性

- カスタマイズ
 - グループ
 - アドバイザー
 - グラフ
 - フィルター
 - イベントハンドラ(SMTP,SNMP等)
- and more ...

The screenshot displays the 'Edit Event Handler' configuration interface in the Oracle MySQL Enterprise Monitor. The main window shows the 'Rule Definition' section with fields for 'Rule Name', 'Advisor Category', and 'Version'. Below this is the 'Variable Assignment' section, which includes a table for assigning variables to data items. The 'Rule Info' section provides details for email notifications. The 'Event Handling' section is expanded, showing options for SMTP Notification Groups (DBA Team), SMTP Notification Policy (Notify on event escalation), SMTP Rate Limit (10), Send SNMP Traps (No), SNMP Notification Policy (Notify on any status change), and Auto-Close Events (No). The interface includes a 'Save Event Handler' button and a 'Cancel' button.

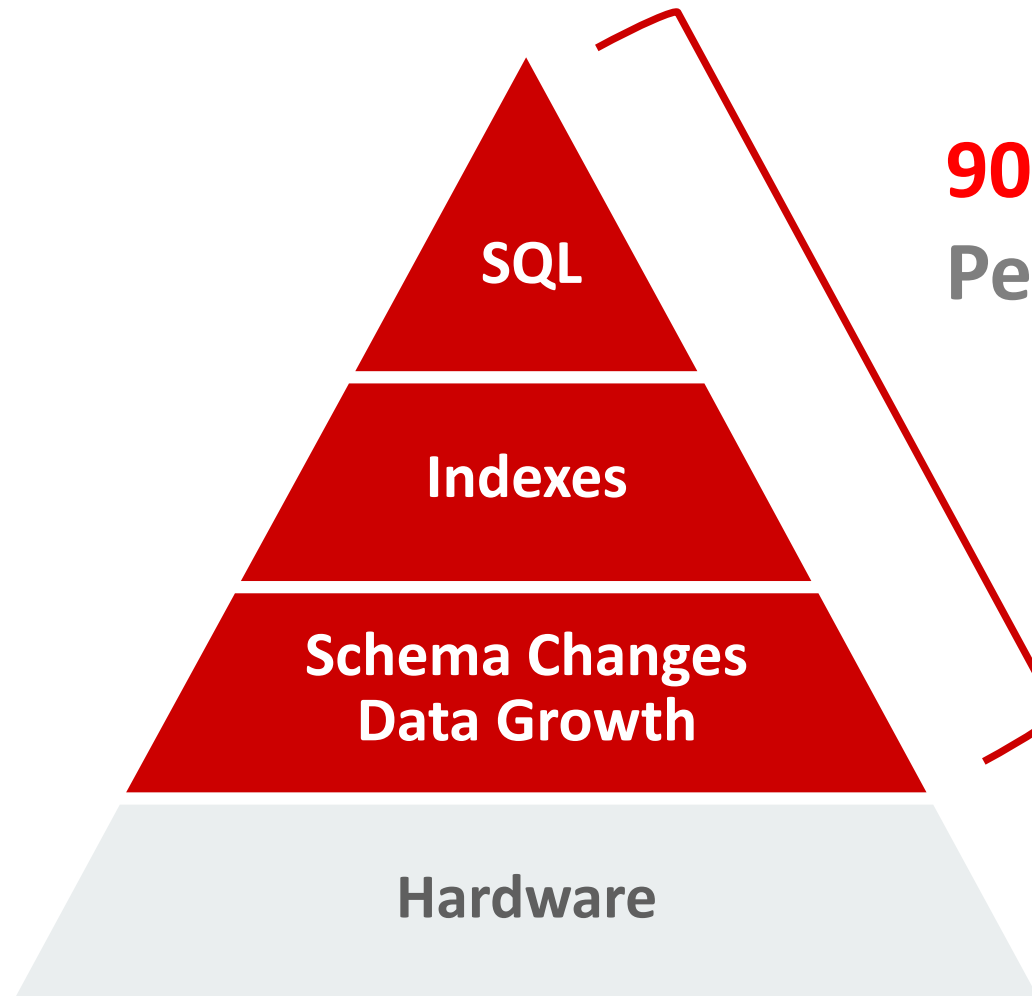
Custom Data Collection

<http://dev.mysql.com/doc/mysql-monitor/3.0/en/memcustomdata-collection.html>

Overview of Advisor Creation

<http://dev.mysql.com/doc/mysql-monitor/3.0/en/memcreating-advisors-overview.html>

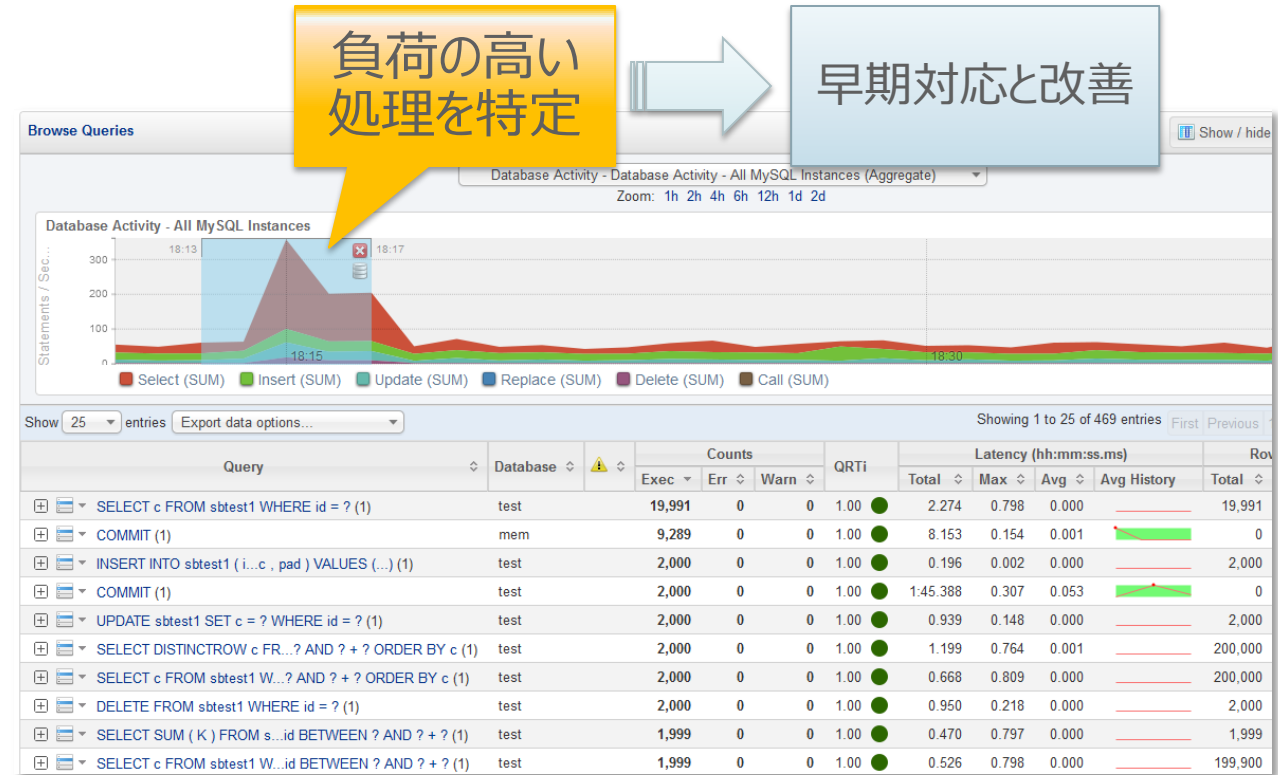
データベース・パフォーマンスの問題の原因 Query Analyzerが解決サポート



90% of
Performance Problems

Enterprise Query Analyzer

- 全クエリーのリアルタイム統合監視
- パフォーマンスの可視化
- コストの大きいクエリーの特定
- クエリー統計詳細の確認
- Query Response Time index (QRTi)
 - クエリーサービスレベル指針
 - サーバー、インスタンスのサービスレベル
 - クエリーパフォーマンス指標



Advantages of the Query Analyzer **over** Slow Query Log

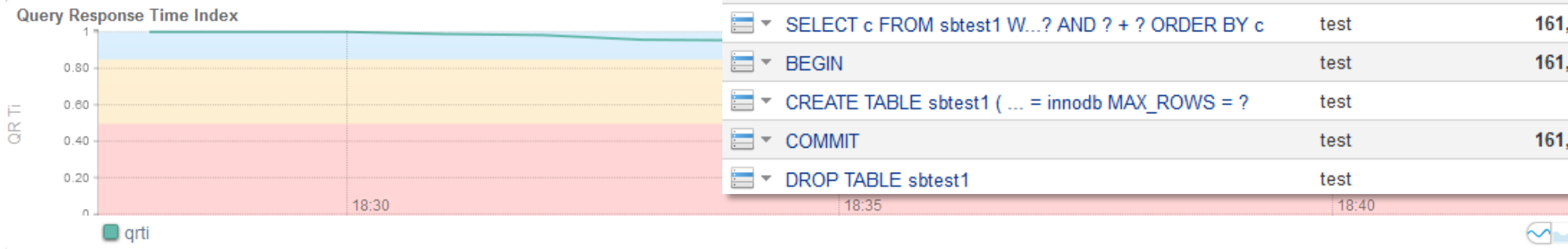
- 実行統計を参照
- 原因となるアプリケーションの特定
- 一定期間の間、全体的なクエリーのパフォーマンスを確認
- 該当のクエリーが発生した時間軸で確認可能
- 実行計画を参照可能
- 特定クエリー、特定ホスト、期間に焦点を置いて確認可能
- 他のパフォーマンスグラフと特定クエリーの相関性を確認可能

Query Response Time Index (QRTi)

- 各クエリの「サービス品質」(QoS) を測定
- サーバ、グループ、またはすべてのインスタンスのQoS測定
- クエリパフォーマンス確認の為の単一測定基準

Query Response Time	
Green (Optimum)	< 100ms
Yellow (Acceptable)	100ms < 400ms
Red (Unacceptable)	400ms <

Query	Database	Counts			QRTi
		Exec	Err	Warn	
INSERT INTO sbtest1 (i...c , pad) VALUES (...)	test	161,133	0	0	1.00 ●
INSERT INTO sbtest1 (k...LUES (...)/* , ... */	test	20	0	0	0.50 ●
SELECT SUM (K) FROM s...id BETWEEN ? AND ? + ?	test	161,156	0	0	1.00 ●
SELECT DISTINCTROW c FR...? AND ? + ? ORDER BY c	test	161,160	0	0	1.00 ●
CREATE INDEX k_1 ON sbtest1 (k)	test	1	0	0	0.00 ●
SELECT c FROM sbtest1 W...? AND ? + ? ORDER BY c	test	161,157	0	0	1.00 ●
BEGIN	test	161,139	0	0	1.00 ●
CREATE TABLE sbtest1 (... = innodb MAX_ROWS = ?	test	1	0	0	0.50 ●
COMMIT	test	161,091	0	0	0.69 ●
DROP TABLE sbtest1	test	1	0	0	0.50 ●



参照 : Query Response Time index (QRTi)

<http://dev.mysql.com/doc/mysql-monitor/3.0/en/mem-features-qrti.html>

Missing Indexes

Indexが無い為、テーブル全体のデータを処理している。
メモリーでソート処理出来ない場合は、
ディスクのTemp Tableで処理する為パフォーマンスが落ちる原因になります。

Temp Tables	
Total	Disk %
30	100
36	100

クエリを参照

Show 10 entries データのエキスポートオプション

Showing 1 to 10 of 647 entries First Previous 1 2 3 4 5 Next Last

クエリ	データベース	警告	カウント			QRTI	待ち時間 (hh:mm:ss.ms)				行数	
			実行	エラー	警告		合計	最高	平均	平均値の履歴	合計	平均
SELECT COUNT (*) AS ...RE `state` = ? LIMIT ? (1)	mysql		22	0	0	0.86	2.384	7.688	0.108		22	
SELECT `plugin_name` FR...ORDER BY `plugin_name` (1)	mysql		26	0	0	0.79	3.498	6.254	0.135		754	
SELECT `plugin_status` ...ugin_name` = ? LIMIT ? (1)	mysql		2	0	0	0.00	6.862	5.589	3.431		1	
COMMIT (1)	mem		15,423	0	0	1.00	1:33.520	4.259	0.006		0	
UPDATE `mem__inventory` ...e` = ? WHERE `hid` = ? (1)	mem		897	0	0	0.99	14.739	4.136	0.016		897	
SELECT GROUP_CONCAT (`...in_status` = ? LIMIT ? (1)	mysql		1	0	0	0.00	4.022	4.022	4.022		1	
UPDATE `mem__inventory` ...p` = ? WHERE `hid` = ? (1)	mem		25	0	0	0.92	4.029	3.949	0.161		25	
DELETE FROM `mem__quan`...timestamp` < ? LIMIT ? (1)	mem		22	0	0	0.95	3.954	3.949	0.180		0	
UPDATE `mem__inventory` ...p` = ? WHERE `hid` = ? (1)	mem		1	0	0	0.00	3.942	3.942	3.942		1	
UPDATE `mem__inventory` ...p` = ? WHERE `hid` = ? (1)	mem		93	0	0	0.97	0.977	3.942	0.011		93	

データのエキスポートオプション

First Previous 1 2 3 4 5 Next Last

New, Un-optimized Statements

新規追加若しくは実行されたクエリーの為、まだ最適化されていない状態

クエリを参照 Show / hide columns  

Show 10 entries データのエキスポートオプション... Showing 1 to 10 of 450 entries First Previous 1 2 3 4 5 Next Last

クエリ	データベース	警告	カウント			QRTI	待ち時間 (hh:mm:ss.ms)				行数		Temp Tables		初回実行
			実行	エラー	警告		合計	最高	平均	平均値の履歴	合計	平均	Total	Disk %	
BY TIMESTAMP (1)	mem		1	0	0	1.00 	0.001	0.001	0.001		28	28	3	0	10:23:59
BY TIMESTAMP (1)	mem		1	0	0	1.00 	0.001	0.001	0.001		28	28	3	0	10:23:59
BY TIMESTAMP (1)	mem		1	0	0	1.00 	0.001	0.001	0.001		28	28	3	0	10:23:59
BY TIMESTAMP (1)	mem		1	0	0	1.00 	0.001	0.001	0.001		28	28	3	0	10:23:59
er_id' = ? (1)	mem		3	0	0	1.00 	0.006	0.005	0.002		3	1	0	0	10:15:04
63_', ... (1)	mem		4	0	0	1.00 	0.010	0.013	0.002		8	2	0	0	2015/03/01 22:03:10
= ?) (1)	mem		1	0	0	1.00 	0.001	0.017	0.001		2	2	0	0	2015/03/01 22:03:09
es` AS ... (1)	mem		2	0	0	1.00 	0.002	0.180	0.001		44	22	0	0	2015/03/01 21:48:21
... (1)	mem		6	0	0	0.92 	0.220	0.461	0.037		6	1	0	0	2015/03/01 21:42:33
ctions` ... (1)	mem		6	0	0	0.92 	0.227	0.672	0.038		6	1	0	0	2015/03/01 21:42:33

データのエクスポートオプション... First Previous 1 2 3 4 5 Next Last

High Rate of Low Latency Statements

高い確率で遅延が発生しているクエリーの設定

クエリを参照 Show / hide columns

Show 10 entries データのEXPORTオプション... Showing 1 to 10 of 448 entries First Previous 1 2 3 4 5 Next Last

クエリ	データベース	警告	カウント			QRTi	待ち時間 (hh:mm:ss.ms)				行数		Temp
			実行	エラー	警告		合計	最高	平均	平均値の履歴	合計	平均	
...UES (...)(1)	mem		8,476	0	0	0.95	4:28.917	1.660	0.032		8,545	1	0
	mem		23,397	0	0	1.00	1:39.170	4.259	0.004		0	0	0
...een`) (1)	mem		8,491	5	0	1.00	59.801	1.150	0.007		16,989	2	0
, ... (1)	mem		493	0	0	0.90	19.218	1.071	0.039		493	1	0
dTotal`) AS ... (1)	mem		48	0	0	0.50	12.150	2.220	0.253		480	10	96
DM (SELECT ... (1)	mem		48	0	0	0.50	10.050	0.542	0.209		480	10	96
E `hid` = ? (1)	mem		1,334	0	0	1.00	9.737	4.136	0.007		1,334	1	0
E `id` = ? (1)	mem		662	0	0	0.98	8.684	3.277	0.013		663	1	0
ble`) (1)	mem		29	0	0	0.71	5.071	0.778	0.175		29	1	0
_` ... (1)	mem		66	0	0	0.89	3.538	1.248	0.054		66	1	0

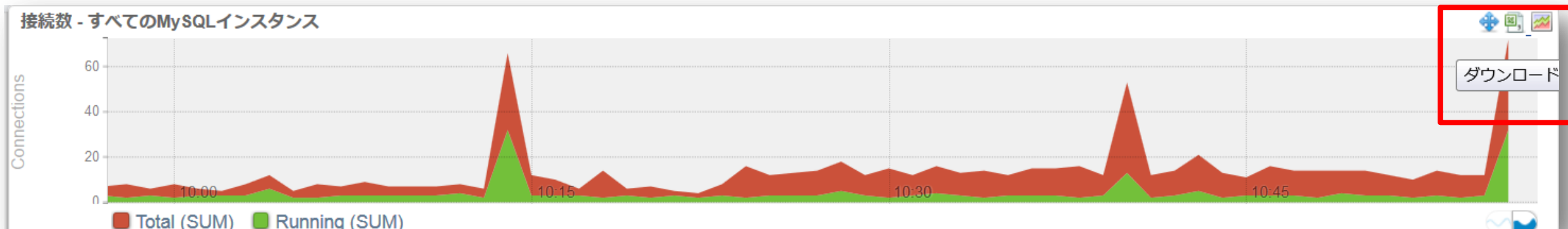
データのEXPORTオプション... First Previous 1 2 3 4 5 Next Last

長期的にデータをアーカイブ

コマンドラインクライアントによる定期的なデータの収集

MEMは、標準的なWebアプリケーションの為、wgetやCurlを用いてデータを/取り出す事も可能。

MEMは、HTTP基本認証やSSLに対応しています。



例) `curl --user user:pass --insecure https://etoolsstable.no.oracle.com:30000/v2/rest/[asset-selection]?format=CSV -o throughput.csv`

Query改善プロセス



クエリパフォーマンスの問題の解決ステップ

•Query Analyzer , Workbench

- 視覚的にスロークエリを特定
- 相関グラフ
- クエリー応答時間指数 (QRTi)
- 実行統計

Step1

Step2

Step3

Step4

- クエリーチューニング
- インデックスの追加
- スキーマのチューニング
- キャッシュヒット率の改善

- MySQL 実行プラン(Explain)
- サンプルクエリ
- クエリグラフ

パフォーマンスの向上

SQLチューニングに関する確認

- **EXPLAIN文でオプティマイザの解析結果を確認**
- スロークエリログを有効に
 - log-slow-queries / long-query-time=2 / log-queries-not-using-indexes
 - mysql_explain_log - ログの内容を表示するコマンド(5.0のみ)
 - mysqldumpslow – スローログ内容を集計するコマンド
- 開発環境では一般ログ(general query log)も利用
 - クエリの重複やクエリが多すぎないか
- “SHOW [FULL] PROCESSLIST”
 - 遅いクエリ、終わらないクエリなどを検出 (PS, SYS, Workbench, MEM)
- クエリが実際に行っている内容の確認
 - FLUSH STATUS; <run query>; SHOW STATUS; (PS, SYS, Workbench, MEM)

問題のあるクエリを解決 – EXPLAIN;

```
EXPLAIN SELECT part_num
FROM `inventory`.`parts`
WHERE (`ven` = "foo")
ORDER BY `delivery_datetime`
DESC LIMIT 100;¥G
```

```
***** 1. row *****
      ID: 1
  select_type: SIMPLE
        table: parts
         type: ref
possible_keys: ven, part#
          key: ven
        key_len: 3
           ref: null
          rows: 872
     Extra: Using WHERE
1 row in set (0.00 sec)
```

解析

- インデックスがどのように使用されているか?
- ファイルソートが必要だったか?
- どのテーブル、カラムがクエリで使用されているか?

修正/チューニング – 以下の作業を繰り返し行う:

- インデックスを追加/変更
- テーブル定義、データ型など変更
- クエリ構造を変更

As of MySQL 5.6.3, permitted explainable statements for [EXPLAIN](#) are [SELECT](#), [DELETE](#), [INSERT](#), [REPLACE](#), and [UPDATE](#). Before MySQL 5.6.3, [SELECT](#) is the only explainable statement.

EXPLAINの各項目

- テーブルごとに1行出力される。
 - id... クエリのID (テーブルのIDではないので注意)
 - select_type... クエリの種類を表す
 - table... 対象のテーブル
 - type... レコードアクセスタイプ。どのようにテーブルにアクセスされるかを示す。
 - possible_keys... 利用可能なキー。
 - key/key_len... 選択されたキーとその長さ。
 - rows... 行数の概算見積もり。
 - Extra... オプティマイザヒント。

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	PRIMARY	demo	ref	idx_fk_country_id	idx_fk_country_id	2	const	31	Using where
2	SUBQUERY	country	ALL	NULL	NULL	NULL	NULL	109	Using where

Full table scan

Chosen index (none)

Number of rows to be read

MySQL 5.6 + MySQL Workbench (Visual Explain)

JOINの順番も一目で確認可能
オブジェクトへのアクセスパターンを色で識別

- 赤色 (要 : 改善検討)

- ALL(full table scan)

- オレンジ色 (要 : 改善検討)

- Full index scan

- Full Text Index Search

- 緑色

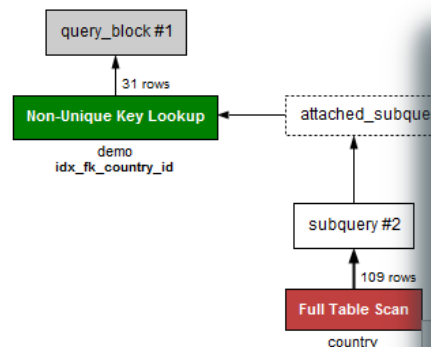
- Range (>,<,...)

- Reference

- 青色(Good)

- EQ_REF

Visual Explain | Display Info: Read + Eval cc | Overview: | View Source:



country

Access Type: ALL
Full Table Scan
Cost Hint: Very High - very costly for large tables (not so much for small ones).
No usable indexes were found for the table and the optimizer must search every row.
This could also mean the search range is so broad that the index would be useless.

Key/Index: -
Attached Condition:
(`sakila`.`country`.`country` = 'Japan')

Rows Examined per Scan: 109
Rows Produced per Join: 109
Filtered (ratio of rows produced per rows examined): 100%
Hint: 100% is best, <= 1% is worst
A low value means the query examines a lot of rows that are not returned.

Rows Examined per Scan: 109
Rows Produced per Join: 109
Filtered (ratio of rows produced per rows examined): 100%
Hint: 100% is best, <= 1% is worst
A low value means the query examines a lot of rows that are not returned.

demo 5 x



MySQL 5.7 + MySQL Workbench (Visual Explain)

Visual Explain | Display Info: Read + Eval cc | Overview: | View Source:

Query cost: 15.20
query_block#1

15.2 | 31 rows
Non-Unique Key Lookup
demo
idx_fk_country_id

attached_subqueries

Query cost: 22.80
subquery #2

22.8 | 109 rows
Full Table Scan
country

Cost estimate (MySQL 5.7)

NEW in 5.7

```
country
Access Type: ALL
Full Table Scan
Cost Hint: Very High - very costly for large tables (not so much for small ones).
No usable indexes were found for the table and the optimizer must search every row.
This could also mean the search range is so broad that the index would be useless.
Used Columns: country_id,
country

Key/Index: -

Attached Condition:
(`sakila`.`country`.`country` = 'Japan')

Rows Examined per Scan: 109
Rows Produced per Join: 0
Filtered (ratio of rows produced per rows examined): 0.9174%
Hint: 100% is best, <= 1% is worst
A low value means the query examines a lot of rows that are not returned.

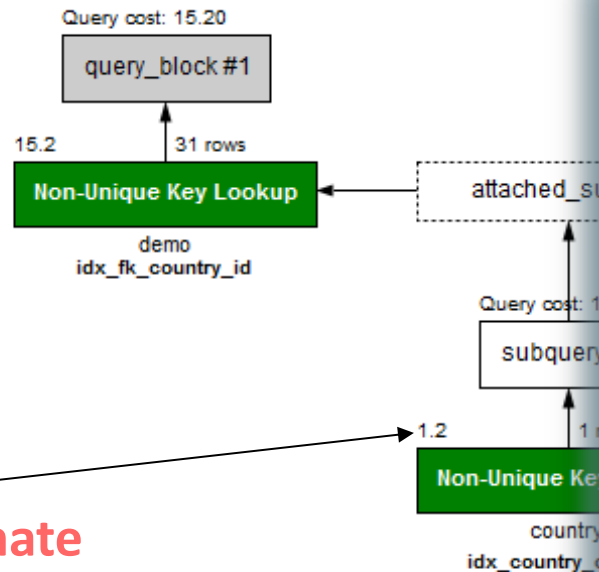
Cost Info
Read: 22.60
Eval: 0.20
Prefix: 22.80
Data Read: 159
```

例) 対象テーブルにIndexを追加後

alter table country add index idx_country_country(country);

Visual Explain | Display Info: Read + Eval cc | Overview: | View Source:

Result Grid



country

Access Type: ref
Non-Unique Key Lookup
Cost Hint: Low-medium - Low if number of matching rows is small, higher as the number of rows increases.
Used Columns: country_id, country

Key/Index: idx_country_country
Ref.: const
Used Key Parts: country
Possible Keys: idx_country_country

Rows Examined per Scan: 1
Rows Produced per Join: 1
Filtered (ratio of rows produced per rows examined): 100%
Hint: 100% is best, <= 1% is worst
A low value means the query examines a lot of rows that are not returned.

Cost Info
Read: 1.00
Eval: 0.20
Prefix: 1.20
Data Read: 160

Cost estimate
22.8 -> 1.2

参考) MySQL 5.6と5.7のEXPLAIN FORMAT=JSON

```
***** 1. row *****
EXPLAIN: {
  "query_block": {
    "select_id": 1,
    "table": {
      "table_name": "demo",
      "access_type": "ref",
      "possible_keys": [
        "idx_fk_country_id"
      ],
      "key": "idx_fk_country_id",
      "used_key_parts": [
        "country_id"
      ],
      "key_length": "2",
      "ref": [
        "const"
      ],
      "rows": 31,
      "filtered": 100,
      "attached_condition": "(`sakila`.`demo`.`country_id` = ("
    },
    "attached_subqueries": [
      {
        "dependent": false,
        "cacheable": true,

```

```
***** 1. row *****
EXPLAIN: {
  "query_block": {
    "select_id": 1,
    "cost_info": {
      "query_cost": "15.20"
    },
    "table": {
      "table_name": "demo",
      "access_type": "ref",
      "possible_keys": [
        "idx_fk_country_id"
      ],
      "key": "idx_fk_country_id",
      "used_key_parts": [
        "country_id"
      ],
      "key_length": "2",
      "ref": [
        "const"
      ],
      "rows_examined_per_scan": 31,
      "rows_produced_per_join": 31,
      "filtered": 100,
      "cost_info": {
        "read_cost": "9.00",
        "eval_cost": "6.20",
        "prefix_cost": "15.20",
        "data_read_per_join": "4K"
      },
      "used_columns": [
        "city_id",

```

NEW in 5.7

まとめ

- 1 MySQL Workbench **Dashboard**ではパフォーマンスに関するデータを確認可能で、短期的なネットワーク, サーバ, InnoDBの状況を可視化する事が出来ます。
- 2 **Performance Report**では、SYSスキーマをベースとした、IO, SQL Statement, Schema 統計, 待機イベント, InnoDB統計等の稼働状況を確認する事が可能です。
- 3 **Performance Report Setup**では、パフォーマンスデータ取得オプションをGUIを利用して詳細に設定する事が可能です。 (setup_*テーブルの値変更)
- 4 **MySQL Enterprise Monitor**では、中長期的なパフォーマンスを確認する事が可能。また、Query Analyzerを利用してQueryの最適化に必要な情報を取得可能です。
- 5 **MySQL Visual Explain**を利用して、SQLの実行プランを確認する事が可能です。MySQL5.7では、より細かいクエリーコストが確認可能です。

補足情報



クエリの種類を把握しよう。

- select_typeでクエリの構造が分かる！

- JOIN

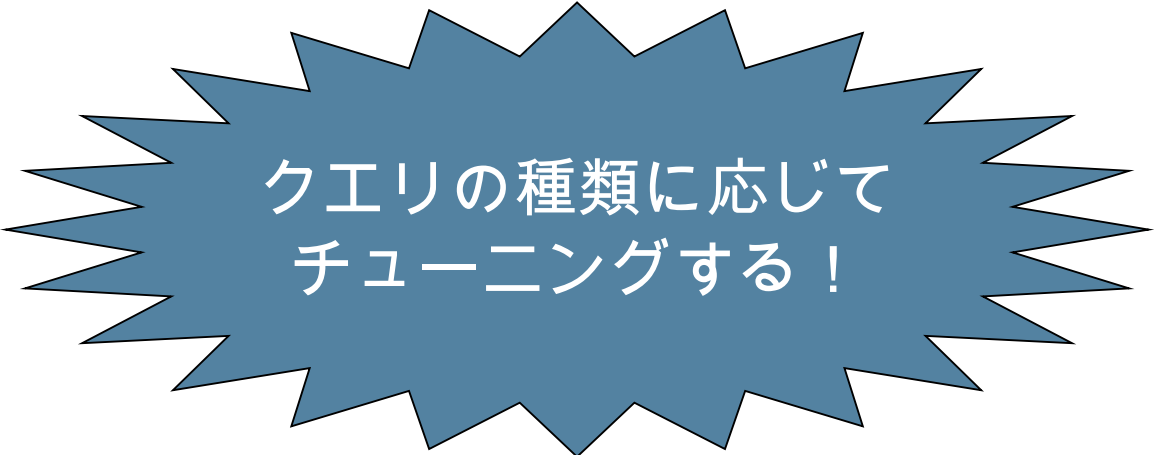
- SIMPLE

- UNION

- UNION, UNION RESULT

- サブクエリ

- FROM句のサブクエリ... PRIMARY, DERIVED
- その他... PRIMARY, SUBQUERY, DEPENDENT SUBQUERY, UNCACHEABLE SUBQUERYなど。
- サブクエリがUNIONになっている場合... DEPENDENT UNION, UNCACHEABLE UNION



クエリの種類に応じて
チューニングする！

Record Access Type

- const・・・PRIMARY/UNIQUEによるルックアップ
- system・・・レコードか1行のテーブル
- ALL・・・テーブルスキャン
- index・・・インデックススキャン
- eq_ref・・・PRIMARY/UNIQUEによるJOIN
- ref・・・ユニークでないインデックスによる等価比較
- ref_or_null・・・ユニークでないインデックスによる等価比較とIS NULLのOR
- range・・・範囲検索
- fulltext・・・全文検索
- index_merge・・・2つの異なるインデックスをAND/OR
- unique_subquery・・・サブクエリ内でPRIMARY/UNIQUEでルックアップ
- index_subquery・・・サブクエリ内でref

Extra: オプティマイザヒント

- Using where... テーブルから行がフェッチされた後、WHERE句の条件によってさらに絞り込みが行われることを示す
- Using index... インデックスだけを使ってクエリを解決出来ることを示す
- Using filesort... ファイルソート（クイックソート）によってソートをする
- Using temporary... クエリを解決するのにテンポラリテーブルが必要
- Using where with pushed condition... engine condition pushdownを利用
- Using index for group-by... MIN()またはMAX()によって集計を行う際、クエリがインデックスだけを用いて解決できる
- Distinct... JOINにおいてDISTINCTによって一意な結果を生成しなければならない場合、内部表のキーでDISTINCTを解決出来る
- Range checked for each record (index map: N)... 遅いJOIN。ほぼフルJOINに近い
- Not exists... LEFT JOINにおいて、内部表にマッチする行が存在しないレコードだけを探したいことを示す
- Using join buffer... JOINバッファが利用されていることを示す

インデックス

- インデックスは参照時の性能は向上するが、更新時はオーバーヘッド
 - 小さなインデックス、プレフィックス index(name(8))
- MySQLはインデックス内で順序が先の列のみ利用可能
 - key (a,b) where b=5 はインデックスを使わない
- インデックスは必要最小限に留めること
 - 例) 性別に対するインデックス (ビットマップ・インデックス)
- ユニークなインデックスにはUNIQUE キーワードをつける
- 重複するようなインデックスの利用は避けるべき
 - key(a, b) があるなら key(a) は削除
- BTREE インデックスはソートされた結果を返す
 - select * from t where b=5 order by c ... key(b,c) optimal
- “covering indexes”は高速、行のデータのフェッチが不要
 - select c from t where b=5 ...の場合、key(b,c) が良い
- OPTIMIZE TABLE ... でインデックスのソートと最適化

SQL オプティマイザの制御

- SELECT STRAIGHT_JOIN * from tbl1,tbl2 ...
 - SQL文に書かれたテーブルの順に処理を行う
- USE INDEX / FORCE INDEX / IGNORE INDEX
 - SELECT * FROM Country USE INDEX(PRIMARY)
 - ヒント句、MySQLでの利用ケースはあまり多くない
 - インデックスを強制的に使わせるケースはある
- ANALYZE TABLE ...
 - 通常はあまり必要とされないが、大量にデータの更新があった後などに実行することも

データ型選択時における注意点

- 出来るだけ小さいサイズの型を使う。
 - 本当にその文字列長は必要か？
 - 文字コードを工夫（日本語が不要ならlatin1やascii、binaryを利用する）
 - ENUMの活用。文字列だが数値として格納される。例) ENUM('YES','NO')
- PROCEDURE ANALYSE()の活用
 - 格納されているデータを分析
 - 最適なデータ型を提案

マルチカラムインデックスの活用

- MySQLのオプティマイザが同時に利用出来るインデックスはひとつだけ
 - インデックスマージが効く場面は限定的
 - インデックスマージより遙かに高速
- 複数の検索条件、ソート条件がある場合はマルチカラムインデックスが役立つ！！

Covering Index

- インデックスだけでクエリを解決出来るようにすること
 - データへアクセスしないのでとても高速
 - EXPLAINコマンドのExtraフィールドは「Using Index」
 - これを目的として多くのインデックスを作成しすぎると更新性能は劣化するため注意が必要
- `SELECT col1, col2 WHERE col1 = x ORDER BY col2;`

(col1, col2)というインデックスがあればデータへのアクセスは不要

MySQLにおけるパーティショニング

- パーティショニングとは？
 - データを特定のカラムの値によって分類
 - それぞれにデータ、インデックスを持つ
- 5.1から導入
 - 現行は同一のストレージエンジンを使用したもののみをサポート
 - 同じテーブル内に違うストレージエンジンのパーティションを持つことは出来ない。
- パーティションのタイプは4つ
 - Range ... カラム値の範囲を指定。
 - List ... カラム値をリストアップ。
 - Hash ... 数値カラムのハッシュ値で分ける。
 - Key ... 文字列カラムのハッシュ値で分ける。

スキーマのデザイン

- **正規化**

- OLTPや書き込み多い環境
- データの冗長性を削減
- JOINのオーバーヘッド
- トータルのデータサイズは小さくなる
- E/R図とスムーズに連携

- **非正規化**

- OLAPやレポーティング
- データの冗長性が高い
- JOINを削減できる

- **データ型**

- tinyint, smallint, mediumint, などを使いデータ量を削減
- JOINに使う列は同じデータ型に
- char(64)ではなくvarchar(64)
- 可能なところはNOT NULLを宣言
- varchar(255)ではなくvarchar(64)

- **インデックス**

- 複数列
- プレフィックス
- “covering index”

ステータス変数(稼働統計)の確認

- MySQLサーバーの動作を監視するために**ステータス変数**を確認する

```
mysql> show status like 'innodb_buf%';
```

```
+-----+-----+
| Variable_name          | Value |
+-----+-----+
| InnoDB_buffer_pool_pages_data | 142   |
| InnoDB_buffer_pool_pages_dirty | 0     |
```

```
shell> mysqladmin -uroot -S /tmp/mysql.sock extended
```

```
+-----+-----+-----+
| Variable_name          | Value |
+-----+-----+-----+
| Aborted_clients       | 0     |
| Aborted_connects     | 0     |
```

- 特定のクエリ(SQL)について調査する場合

```
mysql> FLUSH STATUS; <クエリ実行>; SHOW STATUS;
```

- 定期的に確認する例(15秒間隔で、ステータス変数の差分のみ表示)

```
shell> mysqladmin -u -p ... ex -i 15 -r | grep -v `0`
```

<http://dev.mysql.com/doc/refman/5.6/en/server-status-variables.html>

サーバのコネクション & スレッド

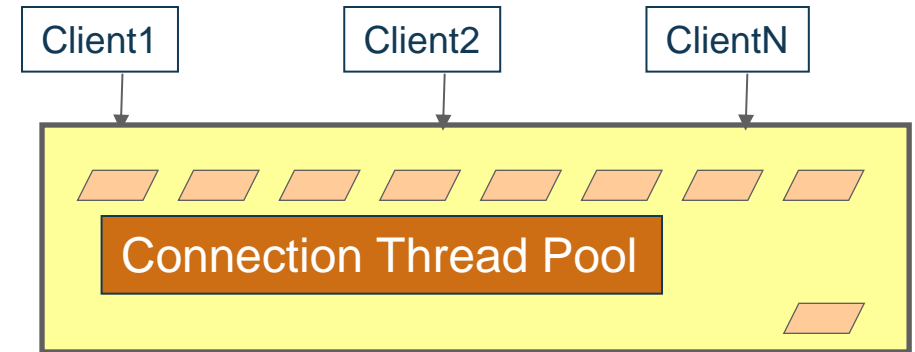
設定ファイル my.cnf :

- **max_connections (151)**
 - サーバが許容可能なコネクション数。
多すぎるとメモリを消費しきる可能性あり
- **thread_cache_size (9) ※**
 - スレッドをコネクションの切断後にもキャッシュしておく数一般的には $\text{max_connections} / 3$

※以下計算式により自動計算

$$8 + (\text{max_connections} / 100)$$

参照 : http://dev.mysql.com/doc/refman/5.6/en/server-system-variables.html#sysvar_thread_cache_size



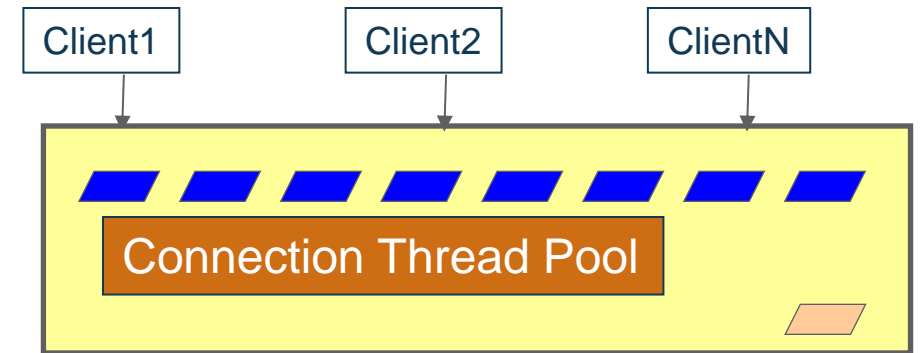
mysql> show status;

- **Max_used_connections**
 - max_connections とあわせてチェック
- **Threads_created**
 - thread_cache ミス
 - 低い数値であるべき

コネクションスレッド毎のバッファ

設定ファイル my.cnf:

- **sort_buffer_size (256KB)**
 - ソート用のメモリサイズ。このサイズを超えるソートはディスクを利用。
MySQL 5.6でデフォルト値が縮小された。(2MB⇒256KB)
- その他のread, read_rnd, etc... バッファはデフォルトで問題ないケースも多い
- バッチ処理などの場合、処理実行前に動的にこれらのパラメタを変更する



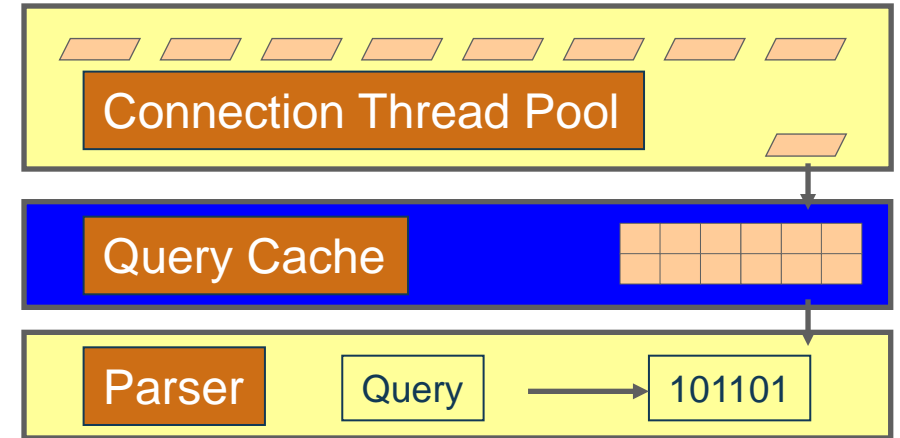
mysql> show status;

- **Sort_merge_passes** -
 - ファイルを利用したマージソートのパス数
 - ソートがメモリ上だけで収まらない場合には要確認
 - インデックスの利用を検討

クエリキャッシュ

設定ファイル my.cnf:

- **query_cache_size (1MB)**
 - クエリキャッシュに割り当てるメモリサイズ
 - 一般的には32MでOK
- **query_cache_type (OFF)**
 - 最悪のケースではパフォーマンスのオーバーヘッドが約15%
 - SELECTの比率が高いサーバで有効
 - DEMANDに設定すると、クエリ実行時にSQL_CACHE句を付けたクエリだけキャッシュ可能



mysql> show status;

- **Qcache_hits, Qcache_inserts**
 - キャッシュヒット/登録件数、ヒット率が小さければクエリキャッシュを無効にすることも検討
- **Qcache_lowmem_prunes**
 - メモリ不足のためにキャッシュが削除された回数。

InnoDB パフォーマンス Tips

- **innodb_buffer_pool_size**

- MySQL&InnoDBのみを利用していれば、メインメモリの80%程度を割り当てる
- データとインデックスの両方をキャッシュ

- **innodb_log_file_size**

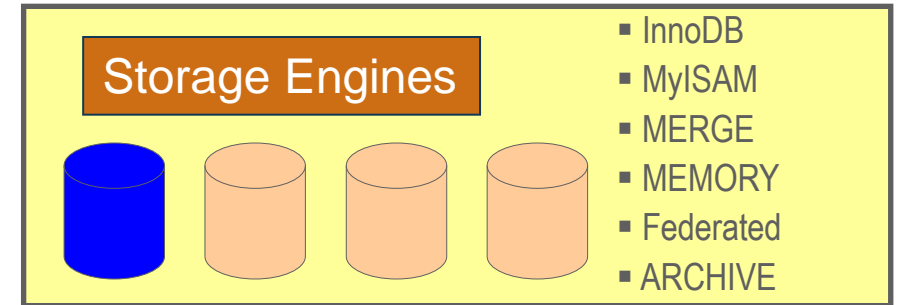
- innodb_buffer_pool_sizeの25%~100%
- ログファイルがどの程度頻繁に切り替わっているかをチェック
- 値を大きくするとクラッシュ後のリカバリ時間が長くなる

- **innodb_flush_log_at_trx_commit**

- 1 (遅い) コミット時にログをフラッシュ。真のACID
- 2 (速い) コミット時にはOSのキャッシュにログをフラッシュ、ディスクとのシンクは毎秒1回
- 0 (最速) ログを毎秒1回(またはそれ以下)フラッシュ

- **innodb_file_per_table**

- MySQL 5.6からデフォルト"ON"
- ディスクI/Oの分散やibdataファイルの肥大化を防ぐために"ON"を推奨



```
mysql> SHOW ENGINE INNODB STATUS;
```

InnoDBの内部での稼働情報

- ファイル I/O
- バッファプール
- ログ情報
- 行/ロック情報



Thank You!