



Introduction to MySQL Sys Schema

Daniel Guzmán Burgos
July 29th 2015

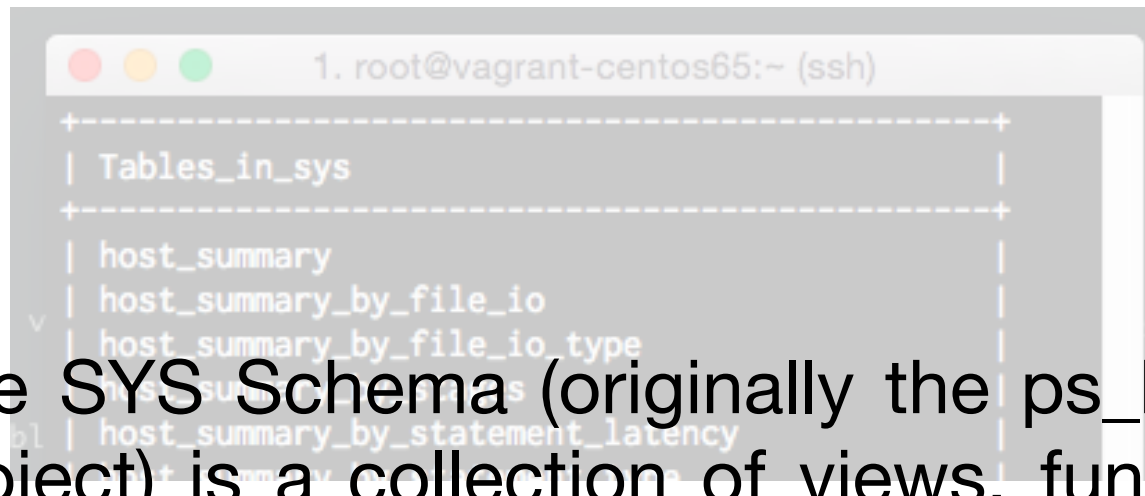
Sys Schema

- Introduction to SYS Schema
- Review of Performance Schema
- Installation
- Overview of objects
- Examples

Sys Schema

- Introduction to SYS Schema
- Review of Performance Schema
- Installation
- Overview of objects
- Examples

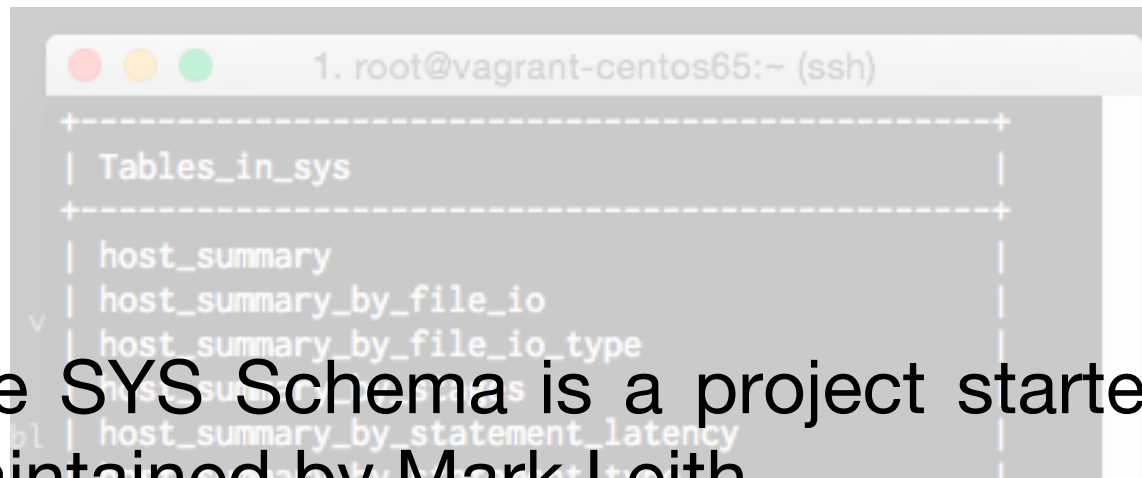
Introduction to SYS Schema



```
1. root@vagrant-centos65:~ (ssh)
+-----+
| Tables_in_sys |
+-----+
| host_summary |
| host_summary_by_file_io |
| host_summary_by_file_io_type |
| host_summary_by_statement_latency |
```

The SYS Schema (originally the ps_helper project) is a collection of views, functions and procedures to help MySQL administrators get insight in to MySQL Database usage.

Introduction to SYS Schema

A terminal window with a title bar showing '1. root@vagrant-centos65:~ (ssh)'. The terminal displays a list of tables in the SYS schema, enclosed in a dashed box. The tables listed are: Tables_in_sys, host_summary, host_summary_by_file_io, host_summary_by_file_io_type, host_summary_by_statement_latency, and host_summary_by_statement_latency.

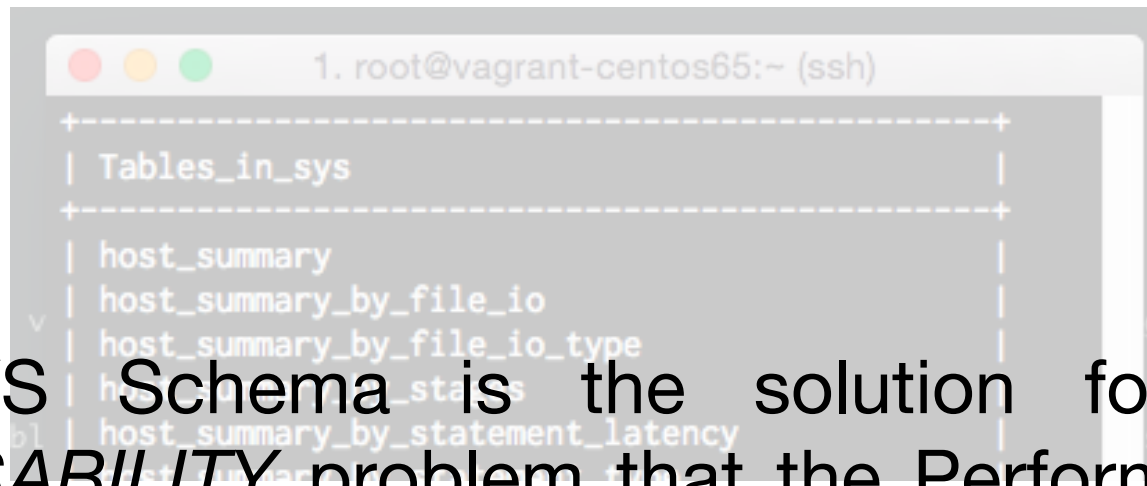
```
1. root@vagrant-centos65:~ (ssh)
+-----+
| Tables_in_sys |
+-----+
| host_summary  |
| host_summary_by_file_io |
| host_summary_by_file_io_type |
| host_summary_by_statement_latency |
| host_summary_by_statement_latency |
+-----+
```

The SYS Schema is a project started and maintained by Mark Leith.

http://www.markleith.co.uk/ps_helper/

Introduction to SYS Schema

SYS Schema is the solution for the *USABILITY* problem that the Performance Schema have.

A terminal window with a title bar showing '1. root@vagrant-centos65:~ (ssh)'. The terminal displays a list of tables in the SYS schema, enclosed in a dashed box. The tables listed are: Tables_in_sys, host_summary, host_summary_by_file_io, host_summary_by_file_io_type, host_summary_by_statement_latency, and host_summary_by_statement_type. The text 'SYS Schema is the solution for the USABILITY problem that the Performance Schema have.' is overlaid on the terminal output.

```
1. root@vagrant-centos65:~ (ssh)
+-----+
| Tables_in_sys |
+-----+
| host_summary  |
| host_summary_by_file_io |
| host_summary_by_file_io_type |
| host_summary_by_statement_latency |
| host_summary_by_statement_type |
+-----+
```

Introduction to SYS Schema

What usability problem?

Sys Schema

- Introduction to SYS Schema
- Review of Performance Schema
- Installation
- Overview of objects
- Examples

Review of Performance Schema

The Performance_schema is the name of both the *storage engine* and the *database* itself, initially implemented in MySQL 5.5, which records runtime statistics from the MySQL database.

The performance_schema is intended to provide **access to useful information** about server execution while having minimal impact on server performance.

Review of Performance Schema

- Platform independent (not like iostat, vmstat which depends of the platform: linux, bsd, etc...)
- Monitor server events (Statements, stages, waits)
- Focus on low overhead/Fast collection:
 - Time measured in picoseconds
 - Operates in fixed memory
 - Per Thread event IDs

Review of Performance Schema

Before P_S, the performance analysis has to be done by using:

- Status variables (SESSION and GLOBAL)
- Show innodb status
- Show profiles
- MySQL Slow query log

We were “gifted” with the PERFORMANCE_SCHEMA tables.

Review of Performance Schema

However...

There is such a wealth of information in those tables that it is intimidating to dive into them: 52 tables to query and 31 configuration variables is enough to scare people.

Review of Performance Schema

This is where Sys Schema comes to rescue!

Sys Schema

- Introduction to SYS Schema
- Review of Performance Schema
- **Installation**
- Overview of objects
- Examples

Installation of Sys Schema

Installation is easy:

- The first step is to get a copy of the SYS SCHEMA files:
 - ◆ *git clone <https://github.com/MarkLeith/mysql-sys>*
 - ◆ If no git:
wget <https://github.com/MarkLeith/mysql-sys/archive/master.zip>

Installation of Sys Schema

- Next, actually install the SYS Schema:
 - ◆ *For instance if you download to /tmp/mysql-sys/, and want to install the 5.6 version you should:*

```
cd /tmp/mysql-sys/  
mysql -u root -p < ./sys_56.sql
```

And that's it! :)

Installation of Sys Schema

Starting from MySQL 5.7.7, Sys Schema comes included by default.

So, when you initialize your MySQL 5.7 server for the first time, with either `mysql_install_db`, or the new `mysqld --initialize` option, the **sys** schema is added alongside the other standard schemas.

Nothing further for you to do, just initialize your database instance as you normally would.

Installation of Sys Schema

After the import, you will have a new “sys” schema with some very descriptive table names.

Do you want to know what tables are using most of our InnoDB buffer memory? Easy:

```
mysql> select * from sys.innodb_buffer_stats_by_table;
```

object_schema	object_name	allocated	data	pages	pages_hashed	pages_old	rows_cached
InnoDB System	SYS_FOREIGN	32.00 KiB	0 bytes	2	2	2	0
InnoDB System	SYS_TABLES	32.00 KiB	1.33 KiB	2	2	2	12
InnoDB System	SYS_COLUMNS	16.00 KiB	5.25 KiB	1	1	1	80
InnoDB System	SYS_DATAFILES	16.00 KiB	426 bytes	1	1	1	8
InnoDB System	SYS_FIELDS	16.00 KiB	855 bytes	1	1	1	20
InnoDB System	SYS_INDEXES	16.00 KiB	962 bytes	1	1	1	14
InnoDB System	SYS_TABLESPACES	16.00 KiB	420 bytes	1	1	1	8
mysql	innodb_index_stats	16.00 KiB	273 bytes	1	1	1	3
mysql	innodb_table_stats	16.00 KiB	61 bytes	1	1	1	1
sys	sys_config	16.00 KiB	49 bytes	1	1	1	1
tmp	#sql6a_8_b	16.00 KiB	540 bytes	1	1	1	20

```
11 rows in set (0.03 sec)
```

Installation of Sys Schema

And how it looks after creating a 1M rows table with sysbench?

```
mysql> select * from sys.innodb_buffer_stats_by_table;
```

object_schema	object_name	allocated	data	pages	pages_hashed	pages_old	rows_cached
test	sbtest1	110.59 MiB	98.62 MiB	7078	7078	7078	726357
mysql	innodb_index_stats	16.00 KiB	856 bytes	1	1	1	10
mysql	innodb_table_stats	16.00 KiB	120 bytes	1	1	1	2

```
3 rows in set (0.52 sec)
```

The ridiculous small buffer pool (128MB) is completely filled with the new table “sbtest1”. How cool is to have this info that easy?

Installation of Sys Schema

What's behind the VIEW *innodb_buffer_stats_by_table* ?
This query:

```
SELECT IF(LOCATE('.', ibp.table_name) = 0, 'InnoDB System', REPLACE(SUBSTRING_INDEX(ibp.table_name, '.', 1), '.', ''))  
       REPLACE(SUBSTRING_INDEX(ibp.table_name, '.', -1), '.', '') AS object_name,  
       sys.format_bytes(SUM(IF(ibp.compressed_size = 0, 16384, compressed_size))) AS allocated,  
       sys.format_bytes(SUM(ibp.data_size)) AS data,  
       COUNT(ibp.page_number) AS pages,  
       COUNT(IF(ibp.is_hashed = 'YES', 1, 0)) AS pages_hashed,  
       COUNT(IF(ibp.is_old = 'YES', 1, 0)) AS pages_old,  
       ROUND(SUM(ibp.number_records)/COUNT(DISTINCT ibp.index_name)) AS rows_cached  
FROM information_schema.innodb_buffer_page ibp  
WHERE table_name IS NOT NULL  
GROUP BY object_schema, object_name  
ORDER BY SUM(IF(ibp.compressed_size = 0, 16384, compressed_size)) DESC;
```

Sys Schema

- Introduction to SYS Schema
- Review of Performance Schema
- Installation
- Overview of objects
- Examples

Objects

Sys Schema is a collection of views, procedures and functions (and 1 table with a couple of triggers).

Those are the objects.

Objects

Procedures:

```
mysql> pager grep Name
PAGER set to 'grep Name'
mysql> SHOW PROCEDURE STATUS\G
      Name: create_synonym_db
      Name: ps_setup_disable_background_threads
      Name: ps_setup_disable_consumer
      Name: ps_setup_disable_instrument
      Name: ps_setup_disable_thread
      Name: ps_setup_enable_background_threads
      Name: ps_setup_enable_consumer
      Name: ps_setup_enable_instrument
      Name: ps_setup_enable_thread
      Name: ps_setup_reload_saved
      Name: ps_setup_reset_to_default
      Name: ps_setup_save
      Name: ps_setup_show_disabled
      Name: ps_setup_show_disabled_consumers
      Name: ps_setup_show_disabled_instruments
      Name: ps_setup_show_enabled
      Name: ps_setup_show_enabled_consumers
      Name: ps_setup_show_enabled_instruments
      Name: ps_statement_avg_latency_histogram
      Name: ps_trace_statement_digest
      Name: ps_trace_thread
      Name: ps_truncate_all_tables
22 rows in set (0.06 sec)
```

Objects

Functions:

```
mysql> SHOW FUNCTION STATUS\G
      Name: extract_schema_from_file_name
      Name: extract_table_from_file_name
      Name: format_bytes
      Name: format_path
      Name: format_statement
      Name: format_time
      Name: ps_is_account_enabled
      Name: ps_is_consumer_enabled
      Name: ps_is_instrument_default_enabled
      Name: ps_is_instrument_default_timed
      Name: ps_is_thread_instrumented
      Name: ps_thread_id
      Name: ps_thread_stack
      Name: sys_get_config
14 rows in set (0.04 sec)
```


Objects

The table:

```
mysql> show create table sys.sys_config\G
***** 1. row *****
      Table: sys_config
Create Table: CREATE TABLE `sys_config` (
  `variable` varchar(128) NOT NULL,
  `value` varchar(128) DEFAULT NULL,
  `set_time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  `set_by` varchar(128) DEFAULT NULL,
  PRIMARY KEY (`variable`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
1 row in set (0.06 sec)
```

Objects

The Triggers:

```
mysql> show triggers\G
        Trigger: sys_config_insert_set_user
        Trigger: sys_config_update_set_user
2 rows in set (0.00 sec)
```

Objects

Views:

...There are 45 amazing views.

Objects

Views:

- Build upon both performance_schema and information_schema
- Both formatter and raw views are available (All raw views are prefixed with x\$)
- And you can add your own.

Objects: Views

User / Host summary views

host_summary
host_summary_by_file_io
host_summary_by_file_io_type
host_summary_by_stages
host_summary_by_statement_latency
host_summary_by_statement_type
user_summary
user_summary_by_file_io
user_summary_by_file_io_type
user_summary_by_stages
user_summary_by_statement_latency
user_summary_by_statement_type

Objects: Views

IO Summary views

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

Objects: Views

Schema Analysis Views

schema_index_statistics
schema_object_overview
schema_table_statistics
schema_table_statistics_with_buffer
schema_tables_with_full_table_scans
schema_unused_indexes

Objects: Views

Wait Analysis Views

wait_classes_global_by_avg_latency
wait_classes_global_by_latency
waits_by_user_by_latency
waits_by_host_by_latency
waits_global_by_latency

Objects: Views

Statement Analysis Views

statement_analysis
statements_with_errors_or_warnings
statements_with_full_table_scans
statements_with_runtimes_in_95th_percentile
statements_with_sorting
statements_with_temp_tables

Objects: Views

Miscellaneous Views

processlist

Objects

For more detail (and source code) check:

<https://github.com/MarkLeith/mysql-sys#overview-of-objects>

Sys Schema

- Introduction to SYS Schema
- Review of Performance Schema
- Installation
- Overview of objects
- **Examples**

Examples

```
mysql> select * from user_summary_by_statement_type;
```

user	statement	total	total_latency	max_latency	lock_latency	rows_sent	rows_examined	rows_affected	full_scans
root	insert	101613	8.68 m	1.96 s	19.52 s	0	0	1101240	0
root	call_procedure	9	19.52 s	10.05 s	2.39 ms	0	0	0	0
root	select	43	1.60 s	522.97 ms	413.41 ms	351	363637	0	27
root	create_view	81	605.87 ms	80.22 ms	73.17 ms	0	0	0	0
root	Field List	450	384.95 ms	14.57 ms	3.48 ms	0	0	0	0
root	create_table	2	227.36 ms	120.38 ms	0 ps	0	0	0	0
root	show_triggers	8	133.26 ms	77.83 ms	13.72 ms	12	12	0	8
root	show_table_status	3	110.45 ms	68.36 ms	365.00 us	246	246	0	3
root	create_index	1	97.99 ms	97.99 ms	16.18 ms	0	0	0	0
root	show_create_table	2	93.79 ms	59.55 ms	0 ps	0	0	0	0
root	show_tables	10	93.17 ms	53.00 ms	935.00 us	736	736	0	10
root	create_trigger	2	69.14 ms	63.47 ms	22.84 ms	0	0	0	0
root	show_procedure_status	4	67.24 ms	56.29 ms	43.35 ms	88	88	0	4
root	create_udf	3	52.45 ms	51.98 ms	283.00 us	0	0	0	0
root	drop_function	14	51.34 ms	49.42 ms	958.00 us	0	0	0	0
root	show_function_status	2	42.71 ms	38.86 ms	1.98 ms	28	28	0	2
root	set_option	6	38.91 ms	37.79 ms	0 ps	0	0	0	0
root	create_function	14	15.98 ms	9.82 ms	12.81 ms	0	0	0	0
root	drop_trigger	2	13.46 ms	13.25 ms	0 ps	0	0	0	0
root	create_procedure	22	8.71 ms	1.55 ms	5.17 ms	0	0	0	0
root	error	13	4.24 ms	3.22 ms	0 ps	0	0	0	0
root	drop_procedure	22	3.95 ms	588.72 us	871.00 us	0	0	0	0
root	show_databases	9	3.06 ms	650.37 us	848.00 us	44	44	0	9
root	show_fields	1	1.25 ms	1.25 ms	683.00 us	6	6	0	1
root	show_storage_engines	5	1.12 ms	245.84 us	498.00 us	45	45	0	5
root	show_variables	1	633.36 us	633.36 us	127.00 us	4	4	0	1
root	create_db	1	595.40 us	595.40 us	0 ps	0	0	1	0
root	Init DB	7	354.32 us	83.10 us	0 ps	0	0	0	0
root	Quit	8	138.85 us	24.40 us	0 ps	0	0	0	0
root	kill	1	65.77 us	65.77 us	0 ps	0	0	0	0

```
30 rows in set (0.01 sec)
```

Examples

The view **user_summary_by_statement_type** is based on the P_S table:

events_statements_summary_by_user_by_event_name

And make use of the function **format_time**.

It's a great view to have an insight of what's going on inside your database in terms of events.

Examples

```
mysql> select * from io_global_by_file_by_latency limit 10;
```

file	total	total_latency	count_read	read_latency	count_write	write_latency	count_misc	misc_latency
@@datadir/ib_logfile1	181553	5.01 m	0	0 ps	90795	2.18 m	90758	2.83 m
@@datadir/ib_logfile0	17499	37.44 s	6	32.77 us	8747	8.95 s	8746	28.49 s
@@datadir/test/sbtest1.ibd	441	4.82 s	0	0 ps	309	816.05 ms	132	4.01 s
@@datadir/ibdata1	505	414.73 ms	418	87.78 ms	79	45.17 ms	8	281.78 ms
@@datadir/sys/sys_config.TRG	24	61.17 ms	8	60.98 ms	0	0 ps	16	187.46 us
@@datadir/sys/version.frm~	6	53.97 ms	0	0 ps	1	21.90 us	5	53.95 ms
@@datadir/Innodb Merge Temp File	8	52.79 ms	1	1.07 ms	1	148.38 us	6	51.57 ms
@@datadir/sys/x@0024host_summary_by_file_io.frm~	6	48.31 ms	0	0 ps	1	17.13 us	5	48.30 ms
@@datadir/sys/x@0024user_summary_by_statement_type.frm~	6	32.58 ms	0	0 ps	1	89.19 us	5	32.49 ms
@@datadir/sys/sys_config.ibd	14	32.11 ms	0	0 ps	7	672.17 us	7	31.44 ms

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

Examples

The view **io_global_by_file_by_latency** is based on the P_S table:

file_summary_by_instance

And make use of the function **format_time** and **format_path**.

This view shows the top global IO consumers by latency by file.

Examples

```
mysql> select * from schema_table_statistics limit 1\G
***** 1. row *****
  table_schema: test
    table_name: sbtest1
  total_latency: 1.49 m
    rows_fetched: 0
  fetch_latency: 0 ps
  rows_inserted: 1101239
  insert_latency: 1.49 m
    rows_updated: 0
  update_latency: 0 ps
    rows_deleted: 0
  delete_latency: 0 ps
  io_read_requests: 21
        io_read: 2.25 KiB
  io_read_latency: 108.46 us
  io_write_requests: 332
        io_write: 272.02 MiB
  io_write_latency: 816.16 ms
  io_misc_requests: 157
    io_misc_latency: 4.02 s
1 row in set (0.09 sec)
```

```
mysql> select * from schema_index_statistics limit 1\G
***** 1. row *****
  table_schema: test
    table_name: sbtest1
    index_name: k_1
  rows_selected: 0
  select_latency: 0 ps
  rows_inserted: 0
  insert_latency: 0 ps
    rows_updated: 0
  update_latency: 0 ps
    rows_deleted: 0
  delete_latency: 0 ps
1 row in set (0.01 sec)
```

```
mysql> select * from schema_tables_with_full_table_scans limit 1\G
***** 1. row *****
  object_schema: sys
    object_name: sys_config
  rows_full_scanned: 4
        latency: 118.67 us
1 row in set (0.01 sec)
```

Examples

The views **schema_table_statistics**, **schema_index_statistics** and **schema_tables_with_full_table_scans** are based on the P_S tables:

table_io_waits_summary_by_table

table_io_waits_summary_by_index_usage

Want to know the most used tables? What are the hot indexes?
Ever wonder if you have tables that are being accessed by full table scans?

This are the views for you!

Examples

```
mysql> select * from statement_analysis limit 1\G
***** 1. row *****
      query: INSERT INTO sbtest1 ( id , k , c , pad ) VALUES (...)
        db: test
      full_scan:
    exec_count: 101239
      err_count: 0
    warn_count: 0
  total_latency: 7.43 m
    max_latency: 1.96 s
    avg_latency: 4.41 ms
   lock_latency: 16.50 s
      rows_sent: 0
  rows_sent_avg: 0
  rows_examined: 0
rows_examined_avg: 0
   rows_affected: 101239
rows_affected_avg: 1
      tmp_tables: 0
  tmp_disk_tables: 0
      rows_sorted: 0
sort_merge_passes: 0
      digest: 49ea91ed4bf76581f51d7bf9f68c5b5a
    first_seen: 2015-07-28 03:04:17
    last_seen: 2015-07-28 04:14:14
1 row in set (0.01 sec)
```

Examples

The view **statement_analysis** is based on the P_S table:

events_statements_summary_by_digest

This table records statement events summarized by schema and digest.

The view aims to emulate the behavior that one can find by using tools like pt-query-digest or the MySQL Enterprise Monitor Query Analysis view.

This view lists a normalized statement view with aggregated statistics, ordered by the total execution time per normalized statement.

Examples

```
mysql> select thd_id, conn_id, user, db, command, state, time, current_statement, lock_latency, tmp_tables, tmp_disk_tables, full_scan from processlist;
```

thd_id	conn_id	user	db	command	state	time	current_statement	lock_latency	tmp_tables	tmp_disk_tables	full_scan
1	NULL	sql/main	NULL	NULL	NULL	366471	NULL	NULL	NULL	NULL	NO
28	8	root@localhost	sys	Query	Sending data	0	select thd_id, conn_id, user, ... es, full_scan from processlist	594.00 us	2	1	YES
2	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
3	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
4	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
5	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
6	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
7	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
8	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
9	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
10	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
11	NULL	innodb/io_handler_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
12	NULL	innodb/srv_monitor_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
13	NULL	innodb/srv_error_monitor_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
14	NULL	innodb/srv_master_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
15	NULL	innodb/srv_purge_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
16	NULL	innodb/srv_lock_timeout_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
18	NULL	innodb/lru_manager_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
19	NULL	innodb/page_cleaner_thread	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO
20	NULL	sql/signal_handler	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NO

```
20 rows in set (0.01 sec)
```

Examples

The view **processlist** is based on the P_S tables:

threads

events_waits_current

events_statements_current

This view is a detailed non-blocking processlist view to replace
[INFORMATION_SCHEMA. | SHOW FULL] PROCESSLIST



Join us at:

Percona Live Europe
September 21-23, 2015. Amsterdam.

<https://www.percona.com/live/europe-amsterdam-2015/>

Daniel Guzman Burgos - Consultant
daniel.guzman.burgos@percona.com

Thank you!