Performance Schema

for MySQL Troubleshooting

March, 1, 2018

Sveta Smirnova

Table of Contents

PERCONA

Overview and Configuration

PERCONA

What is Inside?

5.6	5.7	8.0
• 52 tables	• 87 tables	• 101 tables
• 554 instrs	• 1019 instrs	• 1193 instrs
• 31 variables	• 42 variables	• 44 variables

PERCONA

What Can be Found?

- Which statements are less optimal
- Which operations take most of the time
- Which locks and mutexes taken most often
- What happens inside a session
- How much memory was allocated
- Why users cannot connect from a host
- More

PERCONA

How it Works?

- Internally
  - Instrumented instruction called
  - Corresponding field updated
    - Counter incremented
    - Time recorded
    - Query recorded
    - ...
  - More often you call instruction - higher overhead!

PERCONA

How it Works?

- Internally
- From the user point of view
  - Database `performance_schema`
  - Set of tables with performance statistics
    - `events_NAME_[current|history[_long]]`
    - Few tables with unique names
  - Setup tables
    - `SETUP_*`
  - Option variables
  - Schema `sys`

PERCONA

Performance Schema Defaults

- ON by default
- Only global, thread, statements and transactions instrumentation enabled
- All other consumers are disabled

PERCONA

How to Configure

- Use pattern

```
update performance_schema.setup_consumers set enabled='yes' \
where name like 'OUR_REQUIREMENT_%';

update performance_schema.setup_instruments set enabled='yes', \
timed='yes' where name like 'OUR_REQUIREMENT_%';
```

PERCONA

How to Configure

- Use pattern
- Or easier

```
call sys.ps_setup_enable_consumer(YOUR_CONSUMER);
```

  - Requires sys schema

```
call sys.ps_setup_enable_instrument(YOUR_INSTRUMENT);
```

  - Needs separate install before 5.7

PERCONA

How to Configure

- Use pattern
- Or easier
- Be careful!
  - They are memory and CPU intensive
  - Do not turn them all ON until needed

PERCONA

Statements

Statements Instrumentation

- For regular SQL statements
- Prepared statements
- Stored routines
- Stages of statements execution

PERCONA

What Can We Discover?

- Why statements are slow?
  - Per-query statistics
  - Most evolving stages

PERCONA

What Can We Discover?

- Why statements are slow?
  - Per-query statistics
  - Most evolving stages
- What was executed inside stored routine?

PERCONA

Why Statements are Slow?

- `events_statements_*` and `prepared_statements_instances` tables
  - Important field names
    - `CREATED_TMP_DISK_TABLES`
    - `CREATED_TMP_TABLES`
    - `SELECT_FULL_JOIN`
    - `SELECT_RANGE_CHECK`
    - `SELECT_SCAN`
    - `SORT_MERGE_PASSES`
    - `SORT_SCAN`

PERCONA

Why Statements are Slow?

- `events_statements_*` and `prepared_statements_instances` tables
- Views in `sys` schema
  - Important view names
    - `statement_analysis`
    - `statements_with_full_table_scans`
    - `statements_with_runtimes_in_95th_percentile`
    - `statements_with_sorting`
    - `statements_with_temp_tables`
    - `statements_with_errors_or_warnings`

PERCONA

Which Queries Do Not Use Indexes?

```
mysql> SELECT THREAD_ID TID, SUBSTR(SQL_TEXT, 1, 50) SQL_TEXT, ROWS_SENT RS,
    -> ROWS_EXAMINED RE,CREATED_TMP_TABLES,NO_INDEX_USED,NO_GOOD_INDEX_USED
    -> FROM performance_schema.events_statements_history
    -> WHERE NO_INDEX_USED=1 OR NO_GOOD_INDEX_USED=1\G
*********************** 1. row ***********************
               TID: 10124
          SQL_TEXT: select emp_no, first_name, last_name from employee
                RS: 97750
                RE: 397774
CREATED_TMP_TABLES: 0
     NO_INDEX_USED: 1
NO_GOOD_INDEX_USED: 0
...
```

PERCONA

Take it Easy: Index Usage with sys Schema

```
mysql> SELECT query, total_latency, no_index_used_count, rows_sent,
    -> rows_examined
    -> FROM sys.statements_with_full_table_scans
    -> WHERE db='employees' AND query NOT LIKE '%performance_schema%'\G
*********************** 1. row ***********************
              query: SELECT COUNT ( `emp_no` ) FROM ...  `emp_no` )
                     WHERE `title` = ?
      total_latency: 805.37 ms
no_index_used_count: 1
          rows_sent: 1
      rows_examined: 397774
...
```

PERCONA

Prepared Statements Diagnostics

```
mysql> prepare stmt from 'select dept_no, sum(salary) from employees e ...

mysql> set @d1='d001', @d2='d002', @d3='d003', @d4='d004';

mysql> execute stmt using @d1, @d2, @d3, @d4;
+---------+-------------+
| dept_no | sum(salary) |
+---------+-------------+
| d001    | 13725425266 |
| d002    | 11650834677 |
| d003    |  9363811425 |
| d004    | 41554438942 |
| d005    |  2494260927 |
...
```

PERCONA

Prepared Statements Diagnostics

```
mysql> select * from performance_schema.prepared_statements_instances\G
*************************** 1. row ***************************
      OBJECT_INSTANCE_BEGIN: 139956274327632
              STATEMENT_ID: 1
            STATEMENT_NAME: stmt
                  SQL_TEXT: select dept_no, sum(salary) from employees e...
            OWNER_THREAD_ID: 28
                        ...
            COUNT_REPREPARE: 0
              COUNT_EXECUTE: 1
                        ...
```

Prepared Statements Diagnostics

```
                           ...
              SUM_ROWS_SENT: 9
          SUM_ROWS_EXAMINED: 2011495
SUM_CREATED_TMP_DISK_TABLES: 0
     SUM_CREATED_TMP_TABLES: 1
                           ...
            SUM_SELECT_SCAN: 1
                           ...
              SUM_SORT_ROWS: 9
              SUM_SORT_SCAN: 1
```

Stored Routines Instrumentation

- What happens inside a routine
- Queries, called from the routine
  - `statement/sp/statement`

PERCONA

Stored Routines: example

- We will use this procedure

```
CREATE DEFINER=‘root‘@‘localhost‘ PROCEDURE ‘sp_test‘(val int)
BEGIN
  DECLARE CONTINUE HANDLER FOR 1364, 1048, 1366
  BEGIN
    INSERT IGNORE INTO t1 VALUES(’Some string’);
    GET STACKED DIAGNOSTICS CONDITION 1 @st_state = RETURNED_SQLSTATE;
    GET STACKED DIAGNOSTICS CONDITION 1 @stacked_msg = MESSAGE_TEXT;
  END;
  INSERT INTO t1 VALUES(val);
END
```

- When `HANDLER` called?

PERCONA

Correct Value

```
mysql> call sp_test(1);
Query OK, 1 row affected (0.07 sec)

mysql> select thread_id, event_name, sql_text from events_statements_history
    -> where event_name like 'statement/sp%';
+-----------+-------------------------+---------------------------+
| thread_id | event_name              | sql_text                  |
+-----------+-------------------------+---------------------------+
|        24 | statement/sp/hpush_jump | NULL                      |
|        24 | statement/sp/stmt       | INSERT INTO t1 VALUES(val) |
|        24 | statement/sp/hpop       | NULL                      |
+-----------+-------------------------+---------------------------+
3 rows in set (0.00 sec)
```

PERCONA

HANDLER call

```
mysql> call sp_test(NULL);
Query OK, 1 row affected (0.07 sec)
mysql> select thread_id, event_name, sql_text from events_statements_history
    -> where event_name like 'statement/sp%';
+-----------+-------------------------+----------------------------------------+
| thread_id | event_name              | sql_text                               |
+-----------+-------------------------+----------------------------------------+
|        24 | statement/sp/hpush_jump | NULL                                   |
|        24 | statement/sp/stmt       | INSERT INTO t1 VALUES(val)             |
|        24 | statement/sp/stmt       | INSERT IGNORE INTO t1 VALUES('Som...   |
|        24 | statement/sp/stmt       | GET STACKED DIAGNOSTICS CONDITION...   |
|        24 | statement/sp/stmt       | GET STACKED DIAGNOSTICS CONDITION...   |
|        24 | statement/sp/hreturn    | NULL                                   |
|        24 | statement/sp/hpop       | NULL                                   |
+-----------+-------------------------+
7 rows in set (0.00 sec)
```

Statements Deep Dive

- `events_stages_*` tables

Statements Deep Dive

- `events_stages_*` tables
- Same information as in table `INFORMATION_SCHEMA.PROCESSLIST` or `SHOW PROCESSLIST` output
  - `init`
  - `executing`
  - `Opening tables`

PERCONA

Statements Deep Dive

- events_stages_* tables
- Same information as in table INFORMATION_SCHEMA.PROCESSLIST or SHOW PROCESSLIST output
  - init
  - executing
  - Opening tables
- Replacement for SHOW PROFILE

PERCONA

Statements Deep Dive

- `events_stages_*` tables
- Same information as in table `INFORMATION_SCHEMA.PROCESSLIST` or `SHOW PROCESSLIST` output
  - `init`
  - `executing`
  - `Opening tables`
- Replacement for `SHOW PROFILE`
- Only server-level
  - No storage engine information!

Stages Shortcuts

- Everything, related to temporary tables
  - `EVENT_NAME LIKE 'stage/sql/%tmp%'`
- Everything, related to locks
  - `EVENT_NAME LIKE 'stage/sql/%lock%'`
- Everything in state "`Waiting for`"
  - `EVENT_NAME LIKE 'stage/%/Waiting for%'`
- Frequently met issues

PERCONA

Stages Shortcuts

- Everything, related to temporary tables
- Everything, related to locks
- Everything in state "`Waiting for`"
- Frequently met issues
  - `EVENT_NAME='stage/sql/freeing items'`
  - `EVENT_NAME='stage/sql/Sending data'`
  - `EVENT_NAME='stage/sql/cleaning up'`
  - `EVENT_NAME='stage/sql/closing tables'`
  - `EVENT_NAME='stage/sql/end'`

PERCONA

Stages Example: Which Stage Run Critically Long?

```
mysql> SELECT eshl.event_name, sql_text, eshl.timer_wait/1000000000000 w_s
    -> FROM performance_schema.events_stages_history_long eshl
    -> JOIN performance_schema.events_statements_history_long esthl
    -> ON (eshl.nesting_event_id = esthl.event_id)
    -> WHERE eshl.timer_wait > 1*10000000000\G
*************************** 1. row ***************************
event_name: stage/sql/Sending data
  sql_text: SELECT COUNT(emp_no) FROM employees JOIN salaries USING(emp_no)
            WHERE hire_date=from_date
       w_s: 0.8170
1 row in set (0.00 sec)
```

PERCONA

Stages with sys Schema

- How to prepare

```
mysql> select count(*) from employees.employees join employees.titles using(emp_no)
    -> where title='Senior Engineer';
+----------+
| count(*) |
+----------+
|    97750 |
+----------+
1 row in set (4,57 sec)
mysql> select digest from performance_schema.events_statements_history_long
    -> where sql_text like 'select count(*) from employees%';
+----------------------------------+
| digest                           |
+----------------------------------+
| 059933329986f4b8af007e85f63d6ea4 |
+----------------------------------+
1 row in set (0,00 sec)
```

PERCONA

Stages with sys Schema

- CALL sys.ps_trace_statement_digest

```
mysql> call sys.ps_trace_statement_digest('059933329986f4b8af007e85f63d6ea4',
    -> 10, 0.1, false, true);
+--------------------+
| summary            |
+--------------------+
| Disabled 1 thread |
+--------------------+
1 row in set (0,01 sec)

+--------------------+
| SUMMARY STATISTICS |
+--------------------+
| SUMMARY STATISTICS |
+--------------------+
1 row in set (9,29 sec)
```

PERCONA

Stages with sys Schema

- Overall statistics

```
+------------+-----------+-----------+-----------+---------------+---------------+ ***
| executions | exec_time | lock_time | rows_sent | rows_affected | rows_examined | ***
+------------+-----------+-----------+-----------+---------------+---------------+ ***
|          1 | 4.57 s    | 970.00 us |         1 |             0 |        541058 | ***
+------------+-----------+-----------+-----------+---------------+---------------+ ***

*** +------------+------------+
*** | tmp_tables | full_scans |
*** +------------+------------+
*** |          0 |          0 |
*** +------------+------------+
1 row in set (9,29 sec)
```

PERCONA

Stages with sys Schema

- Stages

event_name	count	latency
stage/sql/Sending data	1	4.57 s
stage/sql/Opening tables	1	732.39 us
stage/sql/starting	1	156.01 us
stage/sql/freeing items	1	92.60 us
stage/sql/init	1	72.87 us
stage/sql/statistics	1	47.47 us
stage/sql/preparing	1	28.28 us
stage/sql/query end	1	19.60 us
stage/sql/optimizing	1	18.66 us
stage/sql/closing tables	1	18.59 us
stage/sql/System lock	1	16.72 us
stage/sql/checking permissions	2	6.93 us
stage/sql/end	1	4.33 us
stage/sql/cleaning up	1	2.36 us
stage/sql/executing	1	1.40 us

PERCONA

Stages with sys Schema

- Additional information
  - Statistics for a sample and slowest query
  - Output of `EXPLAIN FORMAT=JSON`

PERCONA

Memory Usage

Memory Instrumentation

- Since version 5.7
- Answers on question: how memory used?
- Overall and detailed statistics

PERCONA

Memory Instrumentation

- Since version 5.7
- Answers on question: how memory used?
- Overall and detailed statistics
  - Memory usage per server

```
mysql> select * from sys.memory_global_total;
+-----------------+
| total_allocated |
+-----------------+
| 319.69 MiB      |
+-----------------+
1 row in set (0.05 sec)
```

PERCONA

Detailed Memory Statistics

```
mysql> select thread_id tid, user, current_allocated ca, total_allocated
    -> from sys.memory_by_thread_by_current_bytes;
+-----+-------------------------+-------------+-----------------+
| tid | user                    |          ca | total_allocated |
+-----+-------------------------+-------------+-----------------+
|   1 | sql/main                | 2.53 GiB    | 2.69 GiB        |
| 150 | root@127.0.0.1          | 4.06 MiB    | 32.17 MiB       |
| 146 | sql/slave_sql           | 1.31 MiB    | 1.44 MiB        |
| 145 | sql/slave_io            | 1.08 MiB    | 2.79 MiB        |
...
|   4 | innodb/io_log_thread    | -2880 bytes | 132.38 KiB      |
|  72 | innodb/io_write_thread  | -7632 bytes | 1.10 MiB        |
+-----+-------------------------+-------------+-----------------+
145 rows in set (2.65 sec)
```

RAW Performance Schema Tables

- 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
- sys schema includes user account

PERCONA

Users in sys.memory_* Tables

- `NAME@HOST` - regular user
- System users
  - `sql/main`
  - `innodb/*`
  - ...
- Data comes from table `THREADS`

PERCONA

Locks Diagnostics

What can Block Your Queries?

- Metadata locks
  - Table `METADATA_LOCKS`
- Table-level locks
  - Table `TABLE_HANDLES`
- Engine-dependent locks, transactions
  - Tables `EVENTS_TRANSACTIONS_*`
  - This is not exactly lock information!

PERCONA

Metadata Locks

- Table `METADATA_LOCKS`
- Which thread is waiting for a lock
- Which thread holds the lock
- Not only for tables:

```
GLOBAL, SCHEMA, TABLE, FUNCTION, PROCEDURE, EVENT, COMMIT, USER LEVEL LOCK, TABLESPACE
```

PERCONA

METADATA_LOCKS: example

```
mysql> select processlist_id, object_type, lock_type, lock_status, source
    -> from metadata_locks join threads on (owner_thread_id=thread_id)
    -> where object_schema='employees' and object_name='titles'\G
*************************** 1. row ***************************
processlist_id: 4
   object_type: TABLE
     lock_type: EXCLUSIVE
   lock_status: PENDING -- waits
        source: mdl.cc:3263
*************************** 2. row ***************************
processlist_id: 5
   object_type: TABLE
     lock_type: SHARED_READ
   lock_status: GRANTED -- holds
        source: sql_parse.cc:5707
```

PERCONA

Table Locks

- Table `TABLE_HANDLES`
- Not only locks, but all open tables
  - `FLUSH TABLES` removes data from this table

PERCONA

Table Locks: example

```
mysql> select * from table_handles\G
*************************** 1. row ***************************
          OBJECT_TYPE: TABLE
        OBJECT_SCHEMA: employees
          OBJECT_NAME: titles
OBJECT_INSTANCE_BEGIN: 140663937105248
      OWNER_THREAD_ID: 28
       OWNER_EVENT_ID: 951
        INTERNAL_LOCK: NULL
        EXTERNAL_LOCK: READ EXTERNAL  - Read lock
                                        (I run LOCK TABLE ... READ)
```

Table Locks: example

```
*************************** 2. row ***************************
          OBJECT_TYPE: TABLE
        OBJECT_SCHEMA: employees
          OBJECT_NAME: emp
OBJECT_INSTANCE_BEGIN: 140663879605856
      OWNER_THREAD_ID: 26
       OWNER_EVENT_ID: 10419193
        INTERNAL_LOCK: WRITE          - Table lock for MyISAM table
        EXTERNAL_LOCK: WRITE EXTERNAL - Write lock
2 rows in set (0.00 sec)
```

PERCONA

Transactions at Server Level

- Tables `EVENTS_TRANSACTIONS_*`
- Transaction information even if engine is not transactional - One more tool to hunt MDL
- GTIDs
- Background transactions

PERCONA

Background Transaction

```
mysql> select processlist_ID, STATE, GTID, SOURCE, ACCESS_MODE,
     -> ISOLATION_LEVEL, AUTOCOMMIT
     -> from events_transactions_current join threads using(thread_id)\G
*********************** 1. row ***********************
 processlist_ID: NULL
          STATE: COMMITTED
           GTID: NULL
         SOURCE: handler.cc:1248
    ACCESS_MODE: READ WRITE
ISOLATION_LEVEL: REPEATABLE READ
     AUTOCOMMIT: YES
...
```

PERCONA

Transaction in BEGIN ... COMMIT Block

```
mysql> select processlist_ID, STATE, GTID, SOURCE,
    -> ACCESS_MODE, ISOLATION_LEVEL, AUTOCOMMIT
    -> from events_transactions_current join threads using(thread_id)\G
*********************** 2. row ***********************
 processlist_ID: 4
          STATE: COMMITTED
           GTID: AUTOMATIC  - GTID information here
         SOURCE: transaction.cc:150
    ACCESS_MODE: READ WRITE
ISOLATION_LEVEL: REPEATABLE READ
     AUTOCOMMIT: NO
```

Autocommitted Transaction

```
mysql> select processlist_ID, STATE, GTID, SOURCE,
    -> ACCESS_MODE, ISOLATION_LEVEL, AUTOCOMMIT
    -> from events_transactions_current join threads using(thread_id)\G
 ********************** 3. row **********************
  processlist_ID: 5
           STATE: COMMITTED
            GTID: NULL
          SOURCE: handler.cc:1248
     ACCESS_MODE: READ WRITE
 ISOLATION_LEVEL: REPEATABLE READ
      AUTOCOMMIT: YES
...
```

Variables and Status

Variables and Status

- Now in Performance Schema
  - Global
  - Session
  - User-defined - First time ever!

PERCONA

Variables and Status

- Now in Performance Schema
  - Global
  - Session
  - User-defined - First time ever!
- Tables in Information Schema
  - Deprecated in 5.7
  - Removed in 8.0.1

PERCONA

Groupped by

- Variables
  - Global
  - Session
  - By thread
- User variables
  - By thread

- Status variables
  - Global
  - Session
  - Account
  - Host
  - User
  - Thread

PERCONA

Variables and Status: example

- Variables

```
mysql> select * from variables_by_thread
    -> where variable_name='tx_isolation';
+-----------+---------------+-----------------+
| THREAD_ID | VARIABLE_NAME | VARIABLE_VALUE  |
+-----------+---------------+-----------------+
|        71 | tx_isolation  | REPEATABLE-READ |
|        83 | tx_isolation  | REPEATABLE-READ |
|        84 | tx_isolation  | SERIALIZABLE    |
+-----------+---------------+-----------------+
3 rows in set, 3 warnings (0.00 sec)
```

PERCONA

Variables and Status: example

- Variables
- User variables

```
mysql> select * from user_variables_by_thread;
+-----------+---------------+----------------+
| THREAD_ID | VARIABLE_NAME | VARIABLE_VALUE |
+-----------+---------------+----------------+
|        71 | baz           | boo            |
|        84 | foo           | bar            |
+-----------+---------------+----------------+
2 rows in set (0.00 sec)
```

PERCONA

Variables and Status: example

- Variables
- User variables
- Status variables

```
mysql> select * from status_by_thread
    -> where variable_name='Handler_write';
+-----------+---------------+----------------+
| THREAD_ID | VARIABLE_NAME | VARIABLE_VALUE |
+-----------+---------------+----------------+
|        71 | Handler_write | 94             |
|        83 | Handler_write | 477            | -- Most writes
|        84 | Handler_write | 101            |
+-----------+---------------+----------------+
3 rows in set (0.00 sec)
```

PERCONA

Variables Info

- `VARIABLES_INFO` in 8.0+
  - Source of variable
    - `COMPILED`
    - `EXPLICIT`
    - `COMMAND_LINE`
    - `DYNAMIC`
  - Path of option file if specified
  - Minimum and maximum values

PERCONA

Variables Info

- VARIABLES_INFO in 8.0+

```
mysql> select * from variables_info \G
*************************** 1. row ***************************
  VARIABLE_NAME: auto_increment_increment
VARIABLE_SOURCE: COMPILED
  VARIABLE_PATH:
      MIN_VALUE: 1
      MAX_VALUE: 65535
*************************** 2. row ***************************
  VARIABLE_NAME: basedir
VARIABLE_SOURCE: EXPLICIT
  VARIABLE_PATH: /home/sveta/build/mysql-8.0/mysql-test/var/my.cnf
      MIN_VALUE: 0
      MAX_VALUE: 0
      ...
```

PERCONA

Variables Info

- `VARIABLES_INFO` in 8.0+
  - Source of variable
    - `COMPILED`
    - `EXPLICIT`
    - `COMMAND_LINE`
    - `DYNAMIC`
  - Path of option file if specified
  - Minimum and maximum values
- No variable values in this table!

PERCONA

Errors Summary

Errors Summary Tables in 8.0

- Traditionally aggregated
  - `events_errors_summary_by_account_by_error`
  - `events_errors_summary_by_host_by_error`
  - `events_errors_summary_by_thread_by_error`
  - `events_errors_summary_by_user_by_error`
  - `events_errors_summary_global_by_error`
- All tables have similar structure

PERCONA

Errors Summary Tables in 8.0

- Traditionally aggregated
- All tables have similar structure

```
mysql> DESC events_errors_summary_global_by_error;
+-------------------+---------------------+------+-----+---------------------+
| Field             | Type                | Null | Key | Default             |
+-------------------+---------------------+------+-----+---------------------+
| ERROR_NUMBER      | int(11)             | YES  | UNI | NULL                |
| ERROR_NAME        | varchar(64)         | YES  |     | NULL                |
| SQL_STATE         | varchar(5)          | YES  |     | NULL                |
| SUM_ERROR_RAISED  | bigint(20) unsigned | NO   |     | NULL                |
| SUM_ERROR_HANDLED | bigint(20) unsigned | NO   |     | NULL                |
| FIRST_SEEN        | timestamp           | YES  |     | 0000-00-00 00:00:00 |
| LAST_SEEN         | timestamp           | YES  |     | 0000-00-00 00:00:00 |
+-------------------+---------------------+------+-----+---------------------+
7 rows in set (0,03 sec)
```

Errors Summary: Which Accounts Raise More Errors?

```
mysql> select * from events_errors_summary_by_account_by_error
    -> where SUM_ERROR_RAISED > 100\G
*************** 1. row ***************
            USER: root
            HOST: localhost
    ERROR_NUMBER: 1213
      ERROR_NAME: ER_LOCK_DEADLOCK
       SQL_STATE: 40001
 SUM_ERROR_RAISED: 221
SUM_ERROR_HANDLED: 0
      FIRST_SEEN: 2016-09-28 01:45:09
       LAST_SEEN: 2016-09-28 01:47:02
*************** 2. row ***************
            USER: root
            HOST: localhost
    ERROR_NUMBER: 1287
      ERROR_NAME: ER_WARN_DEPRECATED_SYNTAX
       SQL_STATE: HY000
 SUM_ERROR_RAISED: 279
SUM_ERROR_HANDLED: 0
      FIRST_SEEN: 2016-09-27 23:59:49
       LAST_SEEN: 2016-09-28 01:47:05
```

Connection Diagnostics

PERCONA

Connection Diagnostics

- Tables accounts, users, hosts

```
mysql> select user, host, current_connections as cur,
    -> total_connections as total from accounts;
+------+-----------+-----+-------+
| user | host      | cur | total |
+------+-----------+-----+-------+
| foo  | localhost | 0   | 3     |
| root | localhost | 1   | 3     |
| NULL | NULL      | 14  | 17    |
+------+-----------+-----+-------+
3 rows in set (0.01 sec)
```

- Connection attributes

PERCONA

Connection Diagnostics

- Connection attributes

```
mysql> SELECT ATTR_NAME, ATTR_VALUE FROM session_account_connect_attrs
    -> WHERE processlist_id != @@pseudo_thread_id;
+-----------------+--------------------------+
| ATTR_NAME       | ATTR_VALUE               |
+-----------------+--------------------------+
| _os             | Linux                    |
| _client_name    | libmysql                 |
| _pid            | 4729                     |
| program_name    | Webinar                  |
| _platform       | x86_64                   |
| session         | MySQL Performance Schema |
| author          | Sveta Smirnova           |
| _client_version | 5.7.17-13                |
+-----------------+--------------------------+
```

PERCONA

Host Cache

- Content of DNS cache
- Errors from:
  - Name Server
  - Connection
  - Authentication
  - `max_connect_errors`, `max_user_errors`, etc.
- Your first assistant in case of connection issue

Replication

Replication Diagnostics

- Data from `SHOW SLAVE STATUS` available in `replication_*` tables
  - Not full replacement
  - Binary, relay log names and positions are in `mysql.slave_* tables`
  - GTID information for single-threaded slave is in `gtid_executed variable`

PERCONA

Replication Diagnostics

- Data from `SHOW SLAVE STATUS` available in `replication_*` tables
  - Not full replacement
  - Binary, relay log names and positions are in `mysql.slave_* tables`
  - GTID information for single-threaded slave is in `gtid_executed variable`
- Support of Replication Channels (Multi-threaded slave)

PERCONA

Replication Diagnostics

- Data from `SHOW SLAVE STATUS` available in `replication_*` tables
  - Not full replacement
  - Binary, relay log names and positions are in `mysql.slave_* tables`
  - GTID information for single-threaded slave is in `gtid_executed variable`
- Support of Replication Channels (Multi-threaded slave)
- GTID instrumentation

PERCONA

SLAVE STATUS

- No need to parse `SHOW` output

PERCONA

SLAVE STATUS

- No need to parse `SHOW` output
- Configuration
  - `replication_connection_configuration`
  - `replication_applier_configuration`

PERCONA

SLAVE STATUS

- No need to parse `SHOW` output
- Configuration
- IO thread
  - `replication_connection_status`

PERCONA

SLAVE STATUS

- No need to parse `SHOW` output
- Configuration
- IO thread
- SQL thread
  - `replication_applier_status`
  - `replication_applier_status_by_coordinator`
  - `replication_applier_status_by_worker` - MTS only

PERCONA

SLAVE STATUS

- Configuration

```
mysql> select * from replication_connection_configuration
    -> join replication_applier_configuration using(channel_name)\G
*************************** 1. row ***************************
             CHANNEL_NAME:
                     HOST: 127.0.0.1
                     PORT: 13000
                     USER: root
        NETWORK_INTERFACE:
            AUTO_POSITION: 1
              SSL_ALLOWED: NO
              SSL_CA_FILE:
...
```

PERCONA

SLAVE STATUS

- Configuration

```
...
CONNECTION_RETRY_INTERVAL: 60
   CONNECTION_RETRY_COUNT: 10
       HEARTBEAT_INTERVAL: 60.000
             CHANNEL_NAME:
            DESIRED_DELAY: 0
1 row in set (0.00 sec)
```

PERCONA

SLAVE STATUS

- State of IO Thread

```
mysql> select * from replication_connection_status\G
*************************** 1. row ***************************
             CHANNEL_NAME:
               GROUP_NAME:
              SOURCE_UUID: d0753e78-14ec-11e5-b3fb-28b2bd7442fd
                THREAD_ID: 21
             SERVICE_STATE: ON
COUNT_RECEIVED_HEARTBEATS: 17
 LAST_HEARTBEAT_TIMESTAMP: 2015-06-17 15:49:08
 RECEIVED_TRANSACTION_SET:
        LAST_ERROR_NUMBER: 0
       LAST_ERROR_MESSAGE:
     LAST_ERROR_TIMESTAMP: 0000-00-00 00:00:00
1 row in set (0.00 sec)
```

PERCONA

SLAVE STATUS

- State of SQL Thread

```
mysql> select * from replication_applier_status join
    -> replication_applier_status_by_coordinator using(channel_name)\G
*************************** 1. row ***************************
              CHANNEL_NAME:
             SERVICE_STATE: ON
         REMAINING_DELAY: NULL
COUNT_TRANSACTIONS_RETRIES: 0
                 THREAD_ID: 22
             SERVICE_STATE: ON
       LAST_ERROR_NUMBER: 0
      LAST_ERROR_MESSAGE:
    LAST_ERROR_TIMESTAMP: 0000-00-00 00:00:00
1 row in set (0.00 sec)
```

PERCONA

Server Internals

PERCONA

Server Internals

- EVENTS_WAITS_* tables
  - EVENT_NAME
    - wait/synch/rwlock/innodb/dict_operation_lock
  - SOURCE
    - Line of the source code
  - OPERATION
    - Kind of operation: read, lock, write

PERCONA

Which Kind of Events Can We Examine?

- `wait/lock` - Metadata/table locks
- `wait/synch/cond` - InnoDB, MyISAM, sql
- `wait/synch/mutex` - sql, mysys, engines
- `wait/synch/rwlock` - sql, InnoDB, MyISAM
- `wait/synch/sxlock` - InnoDB global locks
- `wait/io/file` - Operations with files
- `wait/io/socket`
- `wait/io/table/sql/handler`

PERCONA

*_INSTANCES Tables

- file_instances - Opened files
- socket_instances - Connections
- cond_instances
- rwlock_instances

```
select * from rwlock_instances where READ_LOCKED_BY_COUNT > 0;
select * from rwlock_instances where WRITE_LOCKED_BY_THREAD_ID > 0;
```

- mutex_instances - LOCKED_BY_THREAD_ID

Easier!

- Tables in sys schema
  - io_*
  - host_summary_*
  - user_summary_*
  - waits_*

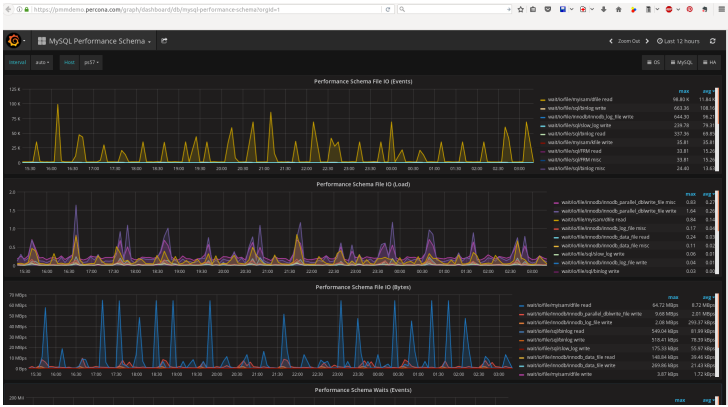
PERCONA

Easier!

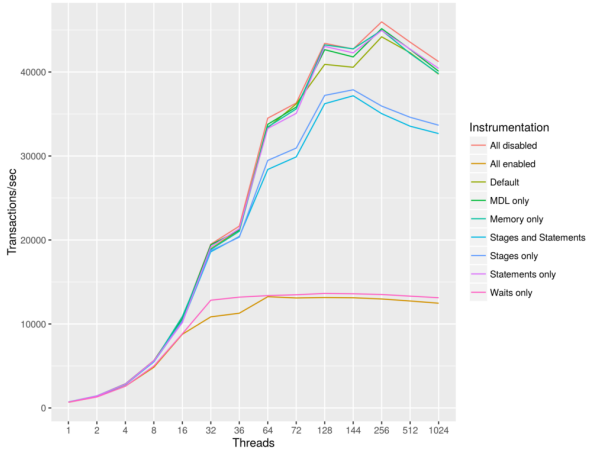
- Tables in sys schema
- Digests events_waits_summary_*
  - *by_account_by_event_name
  - *by_host_by_event_name
  - *by_instance
  - *by_thread_by_event_name
  - *by_user_by_event_name
  - *by_event_name

PERCONA

In Graphs: PMM



Performance Overview



PERCONA

More informaiton

- Blog of developers team
- Blog of Mark Leith: author of sys schema
- Official reference manual
- PMM Demo
- Benchmarks of instruments

PERCONA

Thank you!

http://www.slideshare.net/SvetaSmirnova

https://twitter.com/svetsmirnova