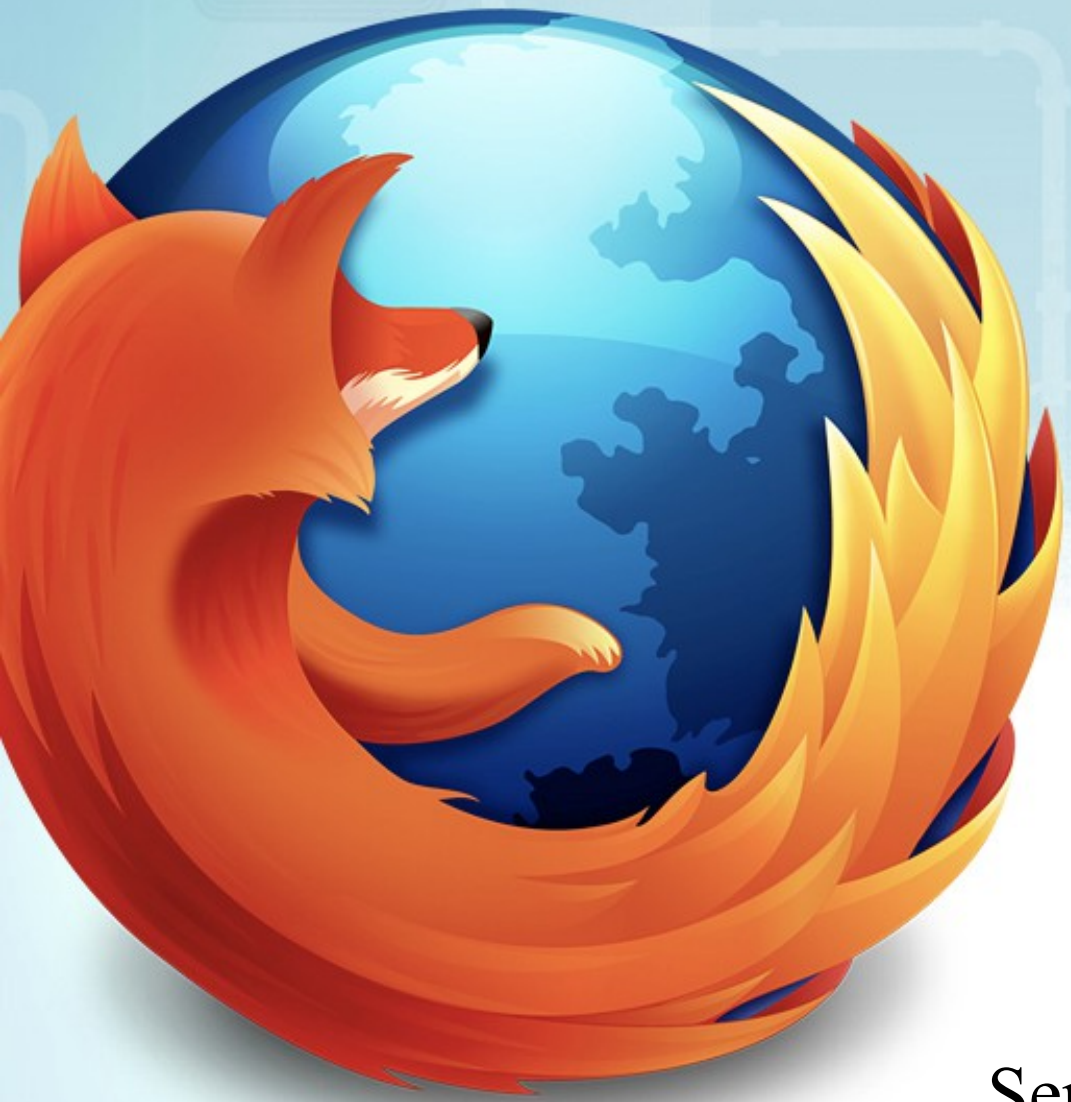
Get Rid of Cron Scripts Using Events



http://bit.ly/mysqlevents

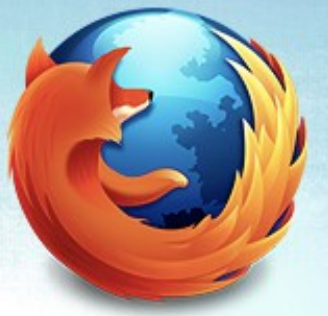
Sheeri Cabral

Senior DB Admin/Architect, Mozilla

@sheeri www.sheeri.com

Northeast PHP 2012

MySQL Events



- Scheduled SQL statements
- Recurring (like “cron”)
  - Scheduled times
  - At intervals
- One-time (like “at”)

- Platform-independent
  - Runs on all platforms the same, 5.1 and up

Why use MySQL events?

- Put db maintenance inside the db
  - Cache table
- Backed up with the db
- No need to store authentication information
  - More secure than a shell script

About the MySQL Event Scheduler

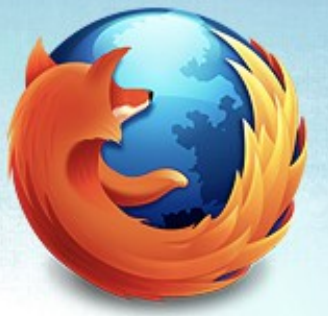
- Separate MySQL thread for event scheduler
- Each event also has its own thread
- Events can be done in parallel
- Server attempts to run events as close to the time scheduled as possible, but there is no guarantee

MySQL Event Process

- What it looks like in SHOW PROCESSLIST\G

```
     Id: 88332
   User: event_scheduler
   Host: localhost
     db: NULL
Command: Daemon
   Time: 2
  State: Waiting for next activation
   Info: NULL
```

About MySQL Events



- Database objects

- Can be defined regardless of whether or not the event scheduler is on

- If the event scheduler is turned off, no new events will be run.
  - Currently executing events will finish normally.

Turning the event scheduler on

- SET GLOBAL event_scheduler=1;

or

- SET GLOBAL event_scheduler='ON';

- Need the SUPER privilege (as with setting any GLOBAL system variable)

- Can check the PROCESSLIST to see if it's running

Disabling the Event Scheduler

- Is not dynamic

- Change option file (/etc/my.cnf) or mysqld command line:

```
mysqld –event_scheduler=DISABLED
```

or

```
[mysqld]
event_scheduler=DISABLED
```

Disabling the Event Scheduler

- If the event scheduler is DISABLED, you cannot turn on the event scheduler:

```
mysql> SET GLOBAL event_scheduler='ON';
ERROR 1290 (HY000): The MySQL server is
running with the --event_scheduler=DISABLED
or --skip-grant-tables option so it cannot
execute this statement
```

Events and Disabled Event Scheduler

- The only change is that events will not run
- When the event scheduler is disabled, events can still be:
  - Created
  - Dropped
  - Modified

Minimal CREATE EVENT statement

- Requires the EVENT privilege

```
CREATE EVENT event_name
ON SCHEDULE schedule
DO statement;
```

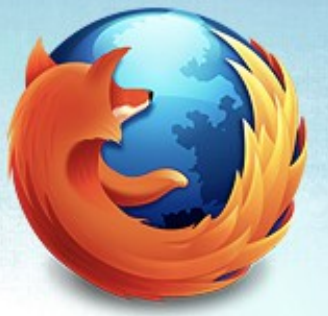
- The statement can be one SQL statement or a BEGIN...END block with many SQL statements
- BEGIN...END block requires the DELIMITER to be changed while creating the event.

Sample Event

- Simple test: insert the time into a table

```
USE test;
CREATE TABLE my_log (
my_time DATETIME NOT NULL
) ENGINE=InnoDB;
```

Sample Event

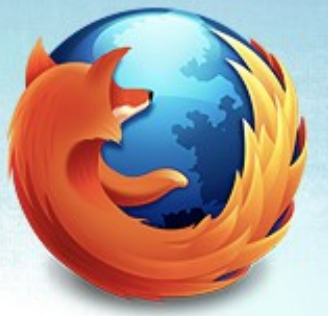


```
CREATE EVENT event_scheduler_test
ON SCHEDULE EVERY 5 SECOND
DO INSERT INTO my_log (my_time)
SELECT NOW();
```

CREATE EVENT / DROP EVENT

- If an event with the same name exists, an error is raised
- Can use CREATE EVENT IF NOT EXISTS event_name to raise a warning instead of an error
- To drop an event:
  - DROP EVENT event_name;
  - DROP EVENT IF EXISTS event_name;

Starting / ending times



- By default, an event's period starts immediately
- By default, an event's period never ends
- Can set STARTS and/or ENDS clauses in CREATE EVENT statement as part of ON SCHEDULE
- Can set STARTS and ENDS relative or absolute:
  - STARTS '2012-07-05 00:00:00'
  - ENDS NOW() + INTERVAL 1 YEAR

Sample Event with Start Time

```
DROP EVENT IF EXISTS
event_scheduler_test;

CREATE EVENT event_scheduler_test

ON SCHEDULE EVERY 1 SECOND

ENDS NOW() + INTERVAL 10 second

DO INSERT INTO my_log (my_time)

SELECT NOW();
```

Specifying starting / ending times

- By default, an event's period starts immediately
- By default, an event's period never ends
- If you set the start time but not the end time:
  - Event's period starts at the start time
  - Event's period never ends
- If you set the end time but not the start time:
  - Event's period starts immediately
  - Event's period ends at the end time

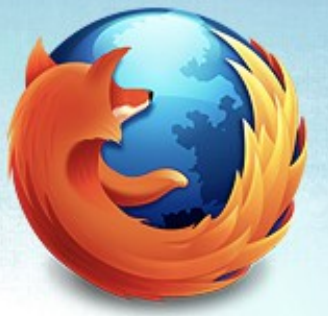
Seeing Events

- SHOW EVENTS command
  - like SHOW TABLES
  - SHOW EVENTS FROM db_name
  - SHOW EVENTS LIKE 'text%'
- INFORMATION_SCHEMA.EVENTS table
- Stored in the mysql.event table
  - Do not change this table!

SHOW CREATE EVENT

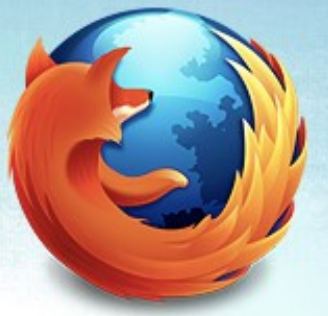
```
mysql> SHOW CREATE EVENT event_scheduler_test\G
*************************** 1. row ***************************
                    Event: event_scheduler_test
                 sql_mode:
                time_zone: SYSTEM
           Create Event: CREATE EVENT
`event_scheduler_test` ON SCHEDULE EVERY 1
SECOND ENDS NOW() + INTERVAL 10 SECOND ON
COMPLETION NOT PRESERVE ENABLE DO INSERT INTO
my_log (my_time) SELECT NOW();
character_set_client: latin1
collation_connection: latin1_swedish_ci
  Database Collation: latin1_swedish_ci
1 row in set (0.00 sec)
```

ON COMPLETION



- When the event's period is over (after ENDS), should you save the event or not?
- Default is to drop the event
  - ON COMPLETION NOT PRESERVE
- Change by using the ON COMPLETION clause:
  - ON COMPLETION PRESERVE

Sample One-time Event

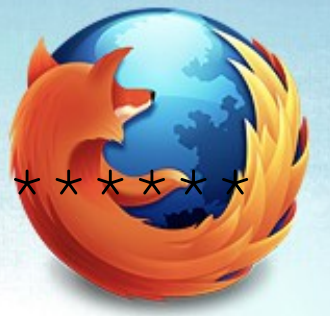


```
DROP EVENT IF EXISTS
event_scheduler_test;

CREATE EVENT event_scheduler_test

ON SCHEDULE AT NOW()+INTERVAL 5 SECOND

DO INSERT INTO my_log (my_time)

SELECT NOW();
```

Preserving a one-time event

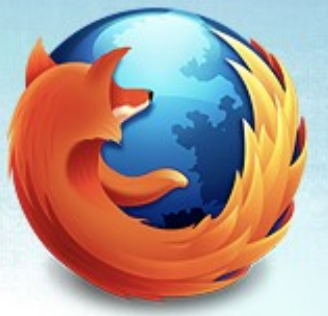
```
DROP EVENT IF EXISTS
event_scheduler_test;

CREATE EVENT event_scheduler_test

ON SCHEDULE AT NOW()+INTERVAL 5 SECOND

ON COMPLETION PRESERVE

DO INSERT INTO my_log (my_time)

SELECT NOW();
```

SHOW EVENTS



```
*************************** 1. row ***************************
                    Db: test
                  Name: event_scheduler_test
               Definer: root@localhost
             Time Zone: SYSTEM
                  Type: ONE TIME
            Execute at: '2012-07-05 19:32:11'
        Interval value: NULL
        Interval field: NULL
               Starts: NULL
                 Ends: NULL
               Status: ENABLED
          Originator: 0
character_set_client: latin1
collation_connection: latin1_swedish_ci
  Database Collation: latin1_swedish_ci
1 row in set (0.00 sec)
```

ALTER EVENT



- Like ALTER TABLE
- Be mindful of clauses
- Incorrect: ALTER EVENT event_scheduler_test ENDS NOW() + INTERVAL 1 MINUTE;
- Correct: ALTER EVENT event_scheduler_test ON SCHEDULE EVERY 1 SECOND ENDS NOW() + INTERVAL 1 MINUTE

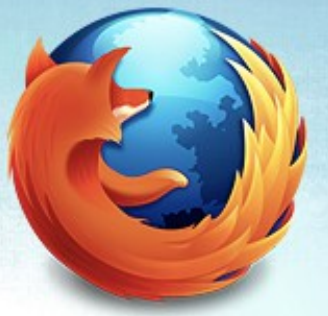
How to rename an event

- ALTER EVENT event_name RENAME new_event_name;

- There is no RENAME EVENT command

- RENAME is the only extra clause ALTER EVENT has compared to CREATE EVENT

Full CREATE syntax

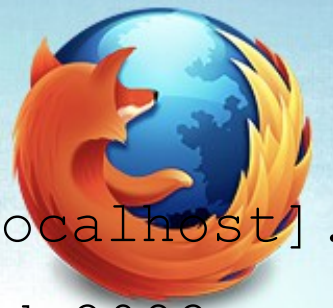


```
CREATE
    [DEFINER = { user | CURRENT_USER }]
    EVENT
    [IF NOT EXISTS]
    event_name
    ON SCHEDULE schedule
    [ON COMPLETION [NOT] PRESERVE]
    [ENABLE | DISABLE | DISABLE ON SLAVE]
    [COMMENT 'comment']
    DO sql_statement;
```

Full CREATE syntax

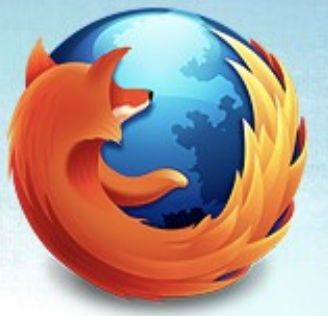
```
schedule:
 AT timestamp [+ INTERVAL interval] ...
 | EVERY interval
   [STARTS timestamp [+ INTERVAL interval] ...]
   [ENDS timestamp [+ INTERVAL interval] ...]


interval:
 quantity {YEAR | QUARTER | MONTH | DAY | HOUR |
MINUTE | WEEK | SECOND | YEAR_MONTH | DAY_HOUR |
DAY_MINUTE | DAY_SECOND | HOUR_MINUTE |
HOUR_SECOND | MINUTE_SECOND}
```

Event logging



```
090109 11:19:54 [Note] Event Scheduler: [root@localhost].
   [test.event_scheduler_test] started in thread 2082.
090109 11:19:54 [Note] Event Scheduler: [root@localhost].
   [test.event_scheduler_test] executed successfully in
thread 2082.
090109 11:19:55 [Note] Event Scheduler: [root@localhost].
   [test.event_scheduler_test] started in thread 2083.
090109 11:19:55 [Note] Event Scheduler: [root@localhost].
   [test.event_scheduler_test] executed successfully in
thread 2083.
090109 11:19:56 [Note] Event Scheduler: Last execution of
test.event_scheduler_test.  Dropping.
090109 11:19:54 [Note] Event Scheduler: Dropping
test.event_scheduler_test.
```

Event error logging



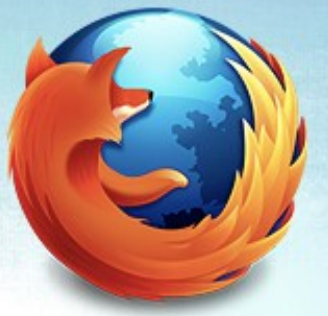
```
CREATE EVENT bad_event

ON SCHEDULE AT NOW()+INTERVAL 1 SECOND

DO INSERT INTO table_not_exists SELECT 1;
```

```
090109 11:19:54 [ERROR] Event Scheduler:
[root@localhost]. [test.bad_event] Table
'test.table_not_exists' doesn't exist.

090109 11:19:54 [Note] Event Scheduler: Last execution of
test.event_scheduler_test.  Dropping.

090109 11:19:54 [Note] Event Scheduler: Dropping
test.bad_event.
```

Event errors

- Like stored procedures and triggers, event errors are not always easy to debug
- Do not ignore [ERROR] Event Scheduler
- You can ignore [Note] Event Scheduler
  - Perhaps except for “Dropping”

Event runtime behavior



- sql_mode
  - No way to explicitly set it
  - SET @@sql_mode=[string]
  - CREATE EVENT....
  - If desired, set the sql_mode back
  - If the event already exists, drop and re-create it.

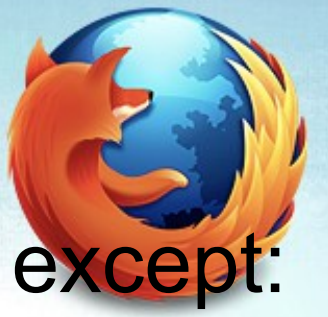
- Character set and collation are the same way
  - SET CHARACTER SET charset
  - SET COLLATION_CONNECTION=collation

Changing whom an event is invoked as

- DEFINER is the invoker for an event
- Default is CURRENT_USER()
- `CREATE DEFINER=user@host event_name`
  - `ALTER DEFINER=user@host event_name`
  - Only if you have the SUPER privilege

- Warning if does not exist at CREATE/ALTER time
  - No warning if the user is dropped later
  - Error when the event is run

Event limitations



- Almost all SQL can be used in DO clause, except:
  - ALTER VIEW
  - LOCK TABLES
  - UNLOCK TABLES
  - LOAD DATA
  - INSERT DELAYED
    - Statement is allowed, but inserts are not actually delayed

Get Rid of Cron Scripts Using Events

http://bit.ly/mysqlevents

Sheeri Cabral
Senior DB Admin/Architect, Mozilla
@sheeri www.sheeri.com
Northeast PHP 2012

MySQL Events

- Scheduled SQL statements
- Recurring (like “cron”)
  - Scheduled times
  - At intervals
- One-time (like “at”)

- Platform-independent
  - Runs on all platforms the same, 5.1 and up

Why use MySQL events?

- Put db maintenance inside the db
  - Cache table
- Backed up with the db
- No need to store authentication information
  - More secure than a shell script

About the MySQL Event Scheduler

- Separate MySQL thread for event scheduler
- Each event also has its own thread
- Events can be done in parallel
- Server attempts to run events as close to the time scheduled as possible, but there is no guarantee

MySQL Event Process

- What it looks like in SHOW PROCESSLIST\G

```
     Id: 88332
   User: event_scheduler
   Host: localhost
     db: NULL
Command: Daemon
   Time: 2
  State: Waiting for next activation
   Info: NULL
```

About MySQL Events

- Database objects
- Can be defined regardless of whether or not the event scheduler is on
- If the event scheduler is turned off, no new events will be run.
  - Currently executing events will finish normally.

Turning the event scheduler on

- SET GLOBAL event_scheduler=1;

or

- SET GLOBAL event_scheduler='ON';

- Need the SUPER privilege (as with setting any GLOBAL system variable)

- Can check the PROCESSLIST to see if it's running

Disabling the Event Scheduler

- Is not dynamic

- Change option file (/etc/my.cnf) or mysqld command line:

```
mysqld –event_scheduler=DISABLED
```

or

```
[mysqld]
event_scheduler=DISABLED
```

Disabling the Event Scheduler

- If the event scheduler is DISABLED, you cannot turn on the event scheduler:

```
mysql> SET GLOBAL event_scheduler='ON';
ERROR 1290 (HY000): The MySQL server is
running with the --event_scheduler=DISABLED
or --skip-grant-tables option so it cannot
execute this statement
```

Events and Disabled Event Scheduler

- The only change is that events will not run
- When the event scheduler is disabled, events can still be:
  - Created
  - Dropped
  - Modified

Minimal CREATE EVENT statement

- Requires the EVENT privilege

```
CREATE EVENT event_name
ON SCHEDULE schedule
DO statement;
```

- The statement can be one SQL statement or a BEGIN...END block with many SQL statements
- BEGIN...END block requires the DELIMITER to be changed while creating the event.

Sample Event

- Simple test: insert the time into a table

```
USE test;
CREATE TABLE my_log (
my_time DATETIME NOT NULL
) ENGINE=InnoDB;
```

Sample Event

```
CREATE EVENT event_scheduler_test
ON SCHEDULE EVERY 5 SECOND
DO INSERT INTO my_log (my_time)
SELECT NOW();
```

CREATE EVENT / DROP EVENT

- If an event with the same name exists, an error is raised
- Can use CREATE EVENT IF NOT EXISTS event_name to raise a warning instead of an error
- To drop an event:
  - DROP EVENT event_name;
  - DROP EVENT IF EXISTS event_name;

Starting / ending times

- By default, an event's period starts immediately
- By default, an event's period never ends
- Can set STARTS and/or ENDS clauses in CREATE EVENT statement as part of ON SCHEDULE
- Can set STARTS and ENDS relative or absolute:
  - STARTS '2012-07-05 00:00:00'
  - ENDS NOW() + INTERVAL 1 YEAR

Sample Event with Start Time

```
DROP EVENT IF EXISTS
event_scheduler_test;
CREATE EVENT event_scheduler_test
ON SCHEDULE EVERY 1 SECOND
ENDS NOW() + INTERVAL 10 second
DO INSERT INTO my_log (my_time)
SELECT NOW();
```

Specifying starting / ending times

- By default, an event's period starts immediately
- By default, an event's period never ends
- If you set the start time but not the end time:
  - Event's period starts at the start time
  - Event's period never ends
- If you set the end time but not the start time:
  - Event's period starts immediately
  - Event's period ends at the end time

Seeing Events

- SHOW EVENTS command
  - like SHOW TABLES
  - SHOW EVENTS FROM db_name
  - SHOW EVENTS LIKE 'text%'
- INFORMATION_SCHEMA.EVENTS table
- Stored in the mysql.event table
  - Do not change this table!

SHOW CREATE EVENT

```
mysql> SHOW CREATE EVENT event_scheduler_test\G
*************************** 1. row ***************************
               Event: event_scheduler_test
            sql_mode:
           time_zone: SYSTEM
        Create Event: CREATE EVENT
`event_scheduler_test` ON SCHEDULE EVERY 1
SECOND ENDS NOW() + INTERVAL 10 SECOND ON
COMPLETION NOT PRESERVE ENABLE DO INSERT INTO
my_log (my_time) SELECT NOW();
character_set_client: latin1
collation_connection: latin1_swedish_ci
  Database Collation: latin1_swedish_ci
1 row in set (0.00 sec)
```

ON COMPLETION

- When the event's period is over (after ENDS), should you save the event or not?
- Default is to drop the event
  - ON COMPLETION NOT PRESERVE
- Change by using the ON COMPLETION clause:
  - ON COMPLETION PRESERVE

Sample One-time Event

```
DROP EVENT IF EXISTS
event_scheduler_test;
CREATE EVENT event_scheduler_test
ON SCHEDULE AT NOW()+INTERVAL 5 SECOND
DO INSERT INTO my_log (my_time)
SELECT NOW();
```

Preserving a one-time event

```
DROP EVENT IF EXISTS
event_scheduler_test;
CREATE EVENT event_scheduler_test
ON SCHEDULE AT NOW()+INTERVAL 5 SECOND
ON COMPLETION PRESERVE
DO INSERT INTO my_log (my_time)
SELECT NOW();
```

SHOW EVENTS

```
*************************** 1. row ***************************
                    Db: test
                  Name: event_scheduler_test
               Definer: root@localhost
             Time Zone: SYSTEM
                  Type: ONE TIME
            Execute at: '2012-07-05 19:32:11'
        Interval value: NULL
        Interval field: NULL
               Starts: NULL
                 Ends: NULL
               Status: ENABLED
           Originator: 0
character_set_client: latin1
collation_connection: latin1_swedish_ci
  Database Collation: latin1_swedish_ci
1 row in set (0.00 sec)
```

ALTER EVENT

- Like ALTER TABLE
- Be mindful of clauses
- Incorrect: ALTER EVENT event_scheduler_test ENDS NOW() + INTERVAL 1 MINUTE;
- Correct: ALTER EVENT event_scheduler_test ON SCHEDULE EVERY 1 SECOND ENDS NOW() + INTERVAL 1 MINUTE

How to rename an event

- ALTER EVENT event_name RENAME new_event_name;

- There is no RENAME EVENT command

- RENAME is the only extra clause ALTER EVENT has compared to CREATE EVENT

Full CREATE syntax

```
CREATE
    [DEFINER = { user | CURRENT_USER }]
    EVENT
    [IF NOT EXISTS]
    event_name
    ON SCHEDULE schedule
    [ON COMPLETION [NOT] PRESERVE]
    [ENABLE | DISABLE | DISABLE ON SLAVE]
    [COMMENT 'comment']
    DO sql_statement;
```

Full CREATE syntax

```
schedule:
 AT timestamp [+ INTERVAL interval] ...
 | EVERY interval
   [STARTS timestamp [+ INTERVAL interval] ...]
   [ENDS timestamp [+ INTERVAL interval] ...]


interval:
 quantity {YEAR | QUARTER | MONTH | DAY | HOUR |
MINUTE | WEEK | SECOND | YEAR_MONTH | DAY_HOUR |
DAY_MINUTE | DAY_SECOND | HOUR_MINUTE |
HOUR_SECOND | MINUTE_SECOND}
```

Event logging

```
090109 11:19:54 [Note] Event Scheduler: [root@localhost].
   [test.event_scheduler_test] started in thread 2082.
090109 11:19:54 [Note] Event Scheduler: [root@localhost].
   [test.event_scheduler_test] executed successfully in
thread 2082.
090109 11:19:55 [Note] Event Scheduler: [root@localhost].
   [test.event_scheduler_test] started in thread 2083.
090109 11:19:55 [Note] Event Scheduler: [root@localhost].
   [test.event_scheduler_test] executed successfully in
thread 2083.
090109 11:19:56 [Note] Event Scheduler: Last execution of
test.event_scheduler_test.  Dropping.
090109 11:19:54 [Note] Event Scheduler: Dropping
test.event_scheduler_test.
```

Event error logging

```
CREATE EVENT bad_event
ON SCHEDULE AT NOW()+INTERVAL 1 SECOND
DO INSERT INTO table_not_exists SELECT 1;
```

```
090109 11:19:54 [ERROR] Event Scheduler:
[root@localhost]. [test.bad_event] Table
'test.table_not_exists' doesn't exist.
090109 11:19:54 [Note] Event Scheduler: Last execution of
test.event_scheduler_test.  Dropping.
090109 11:19:54 [Note] Event Scheduler: Dropping
test.bad_event.
```

Event errors

- Like stored procedures and triggers, event errors are not always easy to debug
- Do not ignore [ERROR] Event Scheduler
- You can ignore [Note] Event Scheduler
  - Perhaps except for “Dropping”

Event runtime behavior

- sql_mode
  - No way to explicitly set it
  - SET @@sql_mode=[string]
  - CREATE EVENT....
  - If desired, set the sql_mode back
  - If the event already exists, drop and re-create it.
- Character set and collation are the same way
  - SET CHARACTER SET charset
  - SET COLLATION_CONNECTION=collation

Changing whom an event is invoked as

- DEFINER is the invoker for an event
- Default is CURRENT_USER()
- `CREATE DEFINER=user@host event_name`
  - `ALTER DEFINER=user@host event_name`
  - Only if you have the SUPER privilege
- Warning if does not exist at CREATE/ALTER time
  - No warning if the user is dropped later
  - Error when the event is run

Event limitations

- Almost all SQL can be used in DO clause, except:
  - ALTER VIEW
  - LOCK TABLES
  - UNLOCK TABLES
  - LOAD DATA
  - INSERT DELAYED
    - Statement is allowed, but inserts are not actually delayed