



ORACLE®

Getting ready for the new MySQL

Giuseppe Maxia

MySQL Community Team Leader

Contents

- Future: MySQL 5.5
 - ✦ performance
 - ★ InnoDB plugin 1.2 default engine
 - ★ Performance schema
 - ✦ ease of use (partitioning, SIGNAL)
 - ✦ reliability (semisynch replication)
- Current: MySQL 5.1
 - ✦ performance: InnoDB plugin 1.0.8

Contents

- **Future: MySQL 5.5**
 - ◆ performance
 - ★ InnoDB plugin 1.1 default engine
 - ★ Performance schema
 - ◆ ease of use (partitioning, SIGNAL)
 - ◆ reliability (semisynch replication)
- **Current: MySQL 5.1**
 - ◆ performance: InnoDB plugin 1.0.8

MySQL 5.5

- MySQL 5.5.5-m3 released on July 18th
- Default storage engine is now InnoDB
- Lots of goodies

Contents

- Future: MySQL 5.5
 - ✦ **performance**
 - ★ **InnoDB plugin 1.1 default engine**
 - ★ Performance schema
 - ✦ ease of use (partitioning, SIGNAL)
 - ✦ reliability (semisynch replication)
- Current: MySQL 5.1
 - ✦ performance: InnoDB plugin 1.0.8

default engine

```
select @@version, @@storage_engine;
```

```
+-----+-----+
| @@version | @@storage_engine |
+-----+-----+
| 5.1.48    | MyISAM           |
+-----+-----+
```

```
select @@version, @@storage_engine;
```

```
+-----+-----+
| @@version | @@storage_engine |
+-----+-----+
| 5.5.5-m3  | InnoDB           |
+-----+-----+
```

Plugin version

```
select @@version, @@innodb_version;
```

@@version	@@innodb_version
5.5.5-m3	1.1.1

InnoDB at a glance

- Performance Improvements
 - ✦ Improved recovery performance
 - ✦ Multiple buffer pool instances
 - ✦ Multiple rollback segments
 - ✦ Native asynchronous I/O for Linux
 - ✦ Extended change buffering

Faster!

- How fast?
- In **my** benchmarks, **10% to 50%** faster.
- Others have reported much higher gains.
- **You** decide.
- Test it under your load.

Contents

- Future: MySQL 5.5
 - ✦ **performance**
 - ★ InnoDB plugin 1.2 default engine
 - ★ **Performance schema**
 - ✦ ease of use (partitioning, SIGNAL)
 - ✦ reliability (semisynch replication)
- Current: MySQL 5.1
 - ✦ performance: InnoDB plugin 1.0.8

PERFORMANCE SCHEMA

- PERFORMANCE_SCHEMA presents low level MySQL performance information
- Data can be cleared
- Filters with WHERE are allowed
- Must be enabled with --performance_schema

performance schema

```
SELECT EVENT_ID, EVENT_NAME, TIMER_WAIT
FROM EVENTS_WAITS_HISTORY
WHERE THREAD_ID = 13
ORDER BY EVENT_ID;
```

EVENT_ID	EVENT_NAME	TIMER_WAIT
86	wait/synch/mutex/mysys/THR_LOCK::mutex	686322
87	wait/synch/mutex/mysys/THR_LOCK_malloc	320535
88	wait/synch/mutex/mysys/THR_LOCK_malloc	339390
89	wait/synch/mutex/mysys/THR_LOCK_malloc	377100
90	wait/synch/mutex/sql/LOCK_plugin	614673
91	wait/synch/mutex/sql/LOCK_open	659925
92	wait/synch/mutex/sql/THD::LOCK_thd_data	494001
93	wait/synch/mutex/mysys/THR_LOCK_malloc	222489
94	wait/synch/mutex/mysys/THR_LOCK_malloc	214947
95	wait/synch/mutex/mysys/LOCK_alarm	312993

performance schema

```
mysql> UPDATE SETUP_INSTRUMENTS
        SET ENABLED = 'NO'
        WHERE NAME =
'wait/synch/mutex/myisammrg/MYRG_INFO::mutex';
```

```
mysql> UPDATE SETUP_CONSUMERS
        SET ENABLED = 'NO'
        WHERE NAME = 'file_summary_by_instance';
```

Contents

- Future: MySQL 5.5
 - ✦ **performance**
 - ★ InnoDB plugin 1.2 default engine
 - ★ Performance schema
 - ✦ **ease of use (partitioning, SIGNAL)**
 - ✦ reliability (semisynch replication)
- Current: MySQL 5.1
 - ✦ performance: InnoDB plugin 1.0.8

MySQL 5.5 enhancements

- PARTITION BY RANGE COLUMNS
- PARTITION BY LIST COLUMNS
- TO_SECONDS

MySQL 5.5 enhancements

```
CREATE TABLE t (  
  dt date  
)  
PARTITION BY RANGE (TO_DAYS(dt))  
(  
  PARTITION p01 VALUES LESS THAN  
(TO_DAYS('2007-01-01')),  
  PARTITION p02 VALUES LESS THAN  
(TO_DAYS('2008-01-01')),  
  PARTITION p03 VALUES LESS THAN  
(TO_DAYS('2009-01-01')),  
  PARTITION p04 VALUES LESS THAN  
(MAXVALUE));
```

BEFORE

5.1

MySQL 5.5 enhancements

BEFORE

5.1

```
SHOW CREATE TABLE t \G
      Table: t
Create Table: CREATE TABLE `t` (
  `dt` date DEFAULT NULL
) ENGINE=MyISAM DEFAULT CHARSET=latin1
/*!50100 PARTITION BY RANGE (TO_DAYS
(dt))
(PARTITION p01 VALUES LESS THAN (733042)
ENGINE = MyISAM,
[...]
```

MySQL 5.5 enhancements

```
CREATE TABLE t (  
  dt date  
)
```

PARTITION BY RANGE COLUMNS (dt)

```
(  
  PARTITION p01 VALUES LESS THAN  
( '2007-01-01' ),  
  PARTITION p02 VALUES LESS THAN  
( '2008-01-01' ),  
  PARTITION p03 VALUES LESS THAN  
( '2009-01-01' ),  
  PARTITION p04 VALUES LESS THAN  
(MAXVALUE) );
```

AFTER

5.5

MySQL 5.5 enhancements

AFTER

5.5

```
SHOW CREATE TABLE t
```

```
Table: t
```

```
Create Table: CREATE TABLE `t` (  
  `dt` date DEFAULT NULL  
) ENGINE=MyISAM DEFAULT CHARSET=latin1  
/*!50500 PARTITION BY RANGE COLUMNS  
(dt)  
(PARTITION p01 VALUES LESS THAN  
( '2007-01-01' ) ENGINE = MyISAM,  
[...]
```

MySQL 5.5 - Multiple columns

```
CREATE TABLE t (  
  a int,  
  b int  
) PARTITION BY RANGE COLUMNS (a,b)  
(  
  PARTITION p01 VALUES LESS THAN  
(10,1),  
  PARTITION p02 VALUES LESS THAN  
(10,10),  
  PARTITION p03 VALUES LESS THAN  
(10,20),  
  PARTITION p04 VALUES LESS THAN  
(MAXVALUE, MAXVALUE) );
```

records	
a	b
1	10
10	9
10	10
10	11

partition definition		
by range (a,b)		
partition	LESS THAN	
p01	10	10
p02	10	20
p03	10	30
p04	10	MAXVALUE
p05	MAXVALUE	MAXVALUE

records	
a	b
1	10
10	9
10	10
10	11

partition definition		
by range (a,b)		
partition	LESS THAN	
p01	10	10
p02	10	20
p03	10	30
p04	10	MAXVALUE
p05	MAXVALUE	MAXVALUE

$(1, 10) < (10, 10) ?$

$(a < 10)$
 OR
 $((a = 10) \text{ AND } (b < 10))$

$(1 < 10)$
 OR
 $((1 = 10) \text{ AND } (10 < 10))$

TRUE

records	
a	b
1	10
10	9
10	10
10	11

partition definition by range (a,b)		
partition	LESS THAN	
p01	10	10
p02	10	20
p03	10	30
p04	10	MAXVALUE
p05	MAXVALUE	MAXVALUE

$(10,9) < (10,10) ?$

$(a < 10)$
OR
 $((a = 10) \text{ AND } (b < 10))$

$(10 < 10)$
OR
 $((10 = 10) \text{ AND } (9 < 10))$

TRUE

records	
a	b
1	10
10	9
10	10
10	11

partition definition by range (a,b)		
partition	LESS THAN	
p01	10	10
p02	10	20
p03	10	30
p04	10	MAXVALUE
p05	MAXVALUE	MAXVALUE

$(10,10) < (10,10) ?$

$(a < 10)$
 OR
 $((a = 10) \text{ AND } (b < 10))$

$(10 < 10)$
 OR
 $((10 = 10) \text{ AND } (10 < 10))$

FALSE

records	
a	b
1	10
10	9
10	10
10	11

partition definition by range (a,b)		
partition	LESS THAN	
p01	10	10
p02	10	20
p03	10	30
p04	10	MAXVALUE
p05	MAXVALUE	MAXVALUE

$(10,10) < (10,20) ?$

$(a < 10)$
 OR
 $((a = 10) \text{ AND } (b < 20))$

$(10 < 10)$
 OR
 $((10 = 10) \text{ AND } (10 < 20))$

TRUE

```

CREATE TABLE employees (
  emp_no int(11) NOT NULL,
  birth_date date NOT NULL,
  first_name varchar(14) NOT NULL,
  last_name varchar(16) NOT NULL,
  gender char(1) DEFAULT NULL,
  hire_date date NOT NULL
) ENGINE=MyISAM
PARTITION BY RANGE COLUMNS (gender, hire_date)
(PARTITION p01 VALUES LESS THAN ('F', '1990-01-01'),
PARTITION p02 VALUES LESS THAN ('F', '2000-01-01'),
PARTITION p03 VALUES LESS THAN ('F', MAXVALUE),
PARTITION p04 VALUES LESS THAN ('M', '1990-01-01'),
PARTITION p05 VALUES LESS THAN ('M', '2000-01-01'),
PARTITION p06 VALUES LESS THAN ('M', MAXVALUE),
PARTITION p07 VALUES LESS THAN (MAXVALUE, MAXVALUE))

```

MySQL 5.5 enhancements

- TRUNCATE PARTITION
- TO_SECONDS()

Contents

- Future: MySQL 5.5
 - ✦ **performance**
 - ★ InnoDB plugin 1.2 default engine
 - ★ Performance schema
 - ✦ **ease of use** (partitioning, **SIGNAL**)
 - ✦ reliability (semisynch replication)
- Current: MySQL 5.1
 - ✦ performance: InnoDB plugin 1.0.8

SIGNAL and RESIGNAL

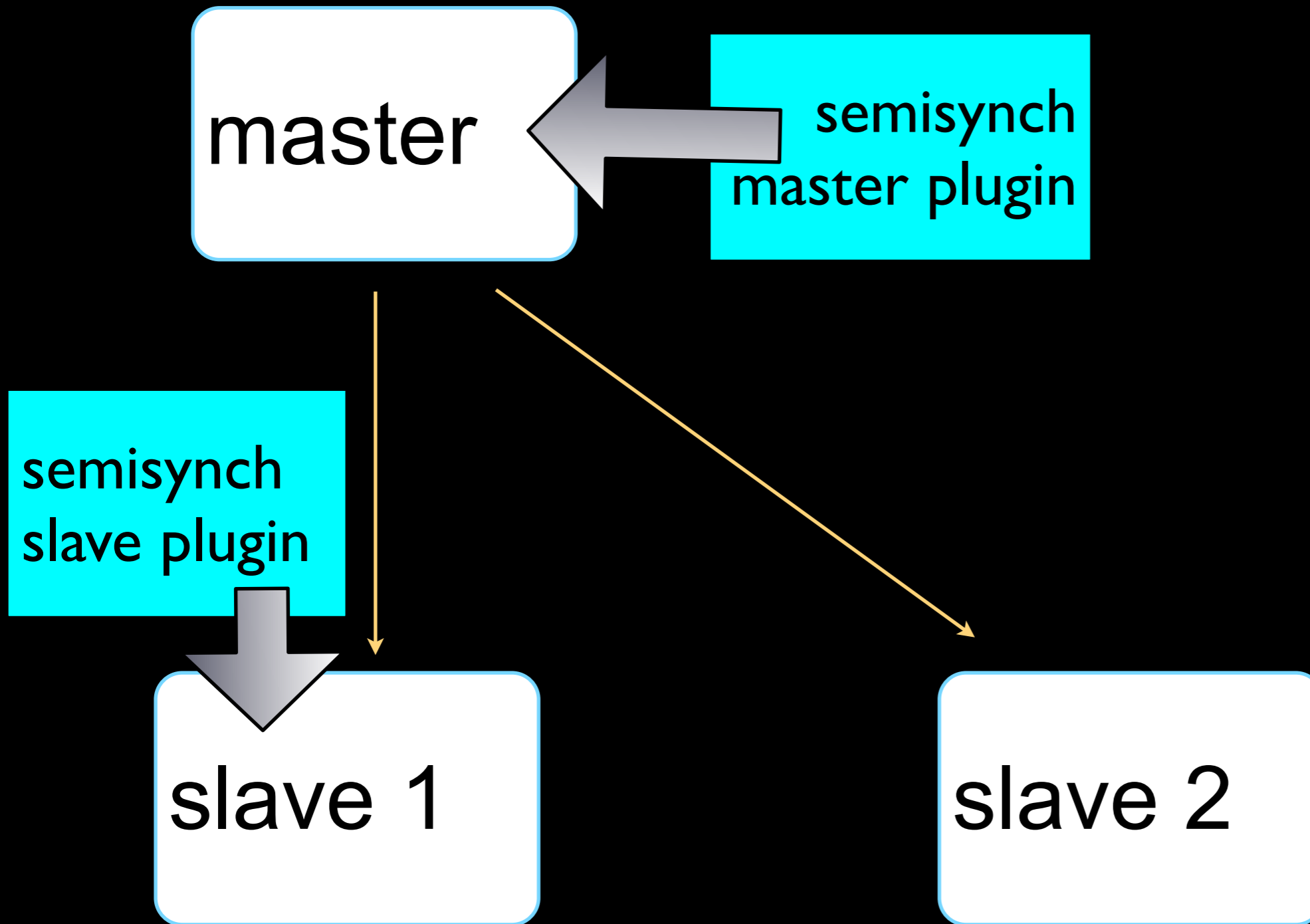
- Allow error handling in stored routines
- The execution is passed to an error handler
- Accessible error values are SQLSTATE, MESSAGE_TEXT and MYSQL_ERRNO
- RESIGNAL can pass along the original error or a new information



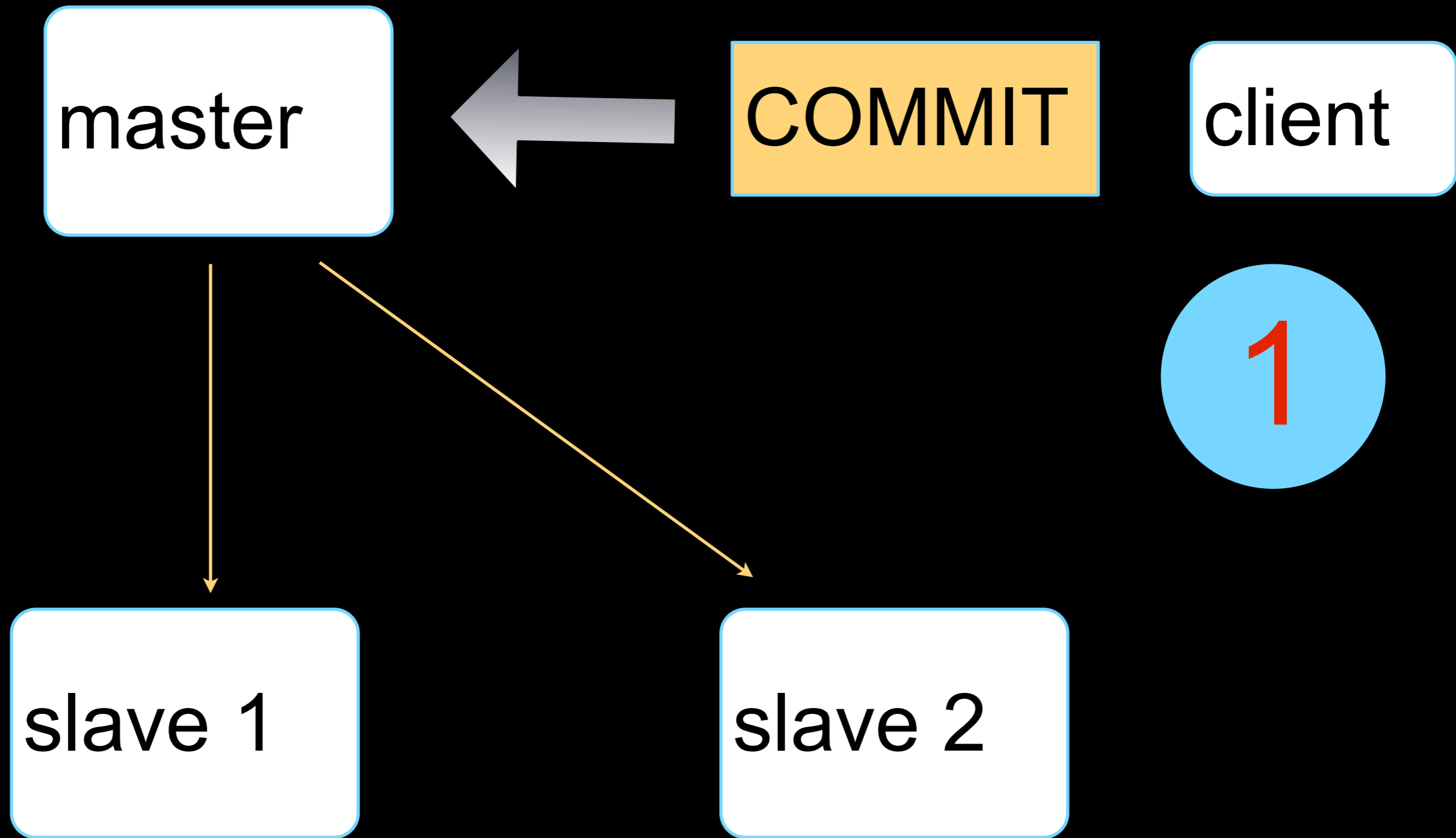
Contents

- Future: MySQL 5.5
 - ✦ **performance**
 - ★ InnoDB plugin 1.2 default engine
 - ★ Performance schema
 - ✦ ease of use (partitioning, SIGNAL)
 - ✦ **reliability (semisynch replication)**
- Current: MySQL 5.1
 - ✦ performance: InnoDB plugin 1.0.8

semi-synchronous replication



semi-synchronous replication



semi-synchronous replication

master

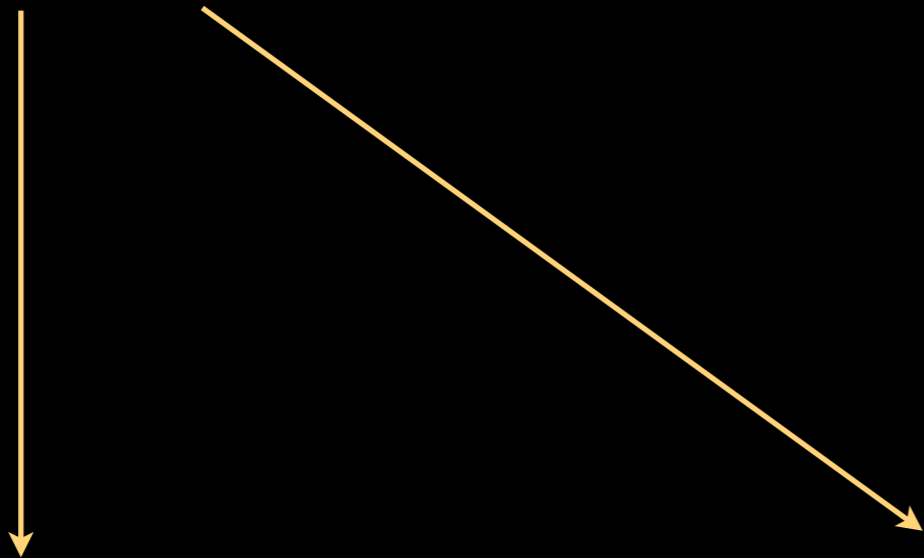
client

COMMIT
binary log

2

slave 1

slave 2

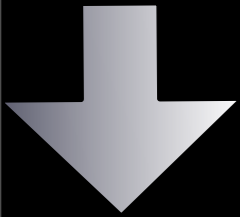


semi-synchronous replication

master

client

COMMIT
binary log



relay log

slave 1

slave 2

3

semi-synchronous replication

master

client

COMMIT
binary log

confirm log
reception

4

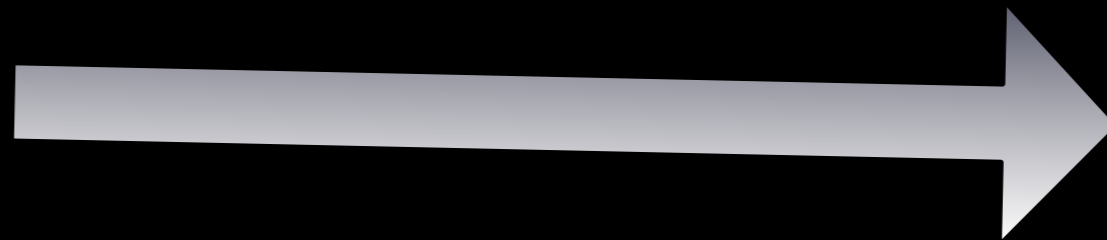
relay log

slave 1

slave 2

semi-synchronous replication

master



client

COMMIT
binary log

relay log

slave 1

slave 2

5

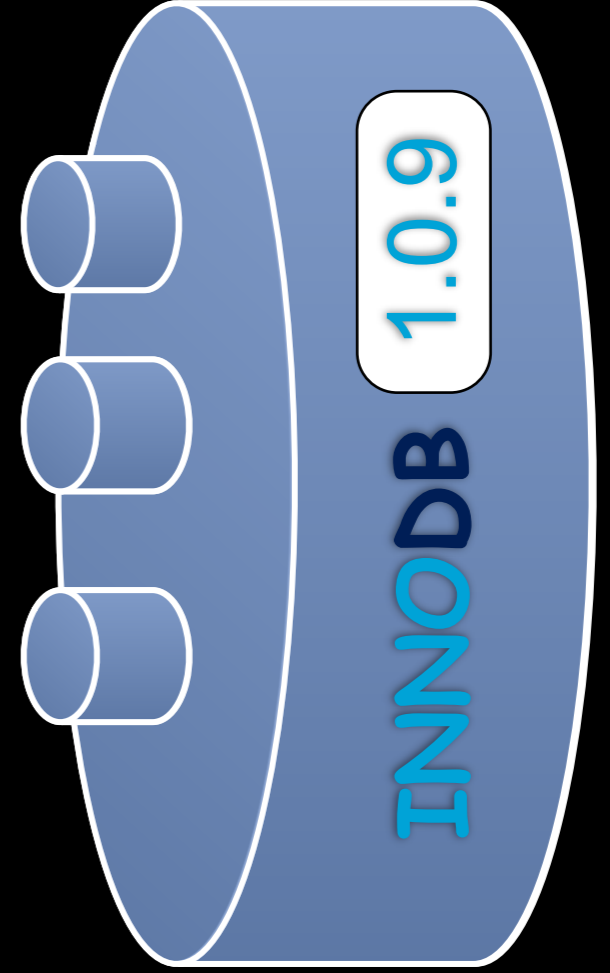
DEMO

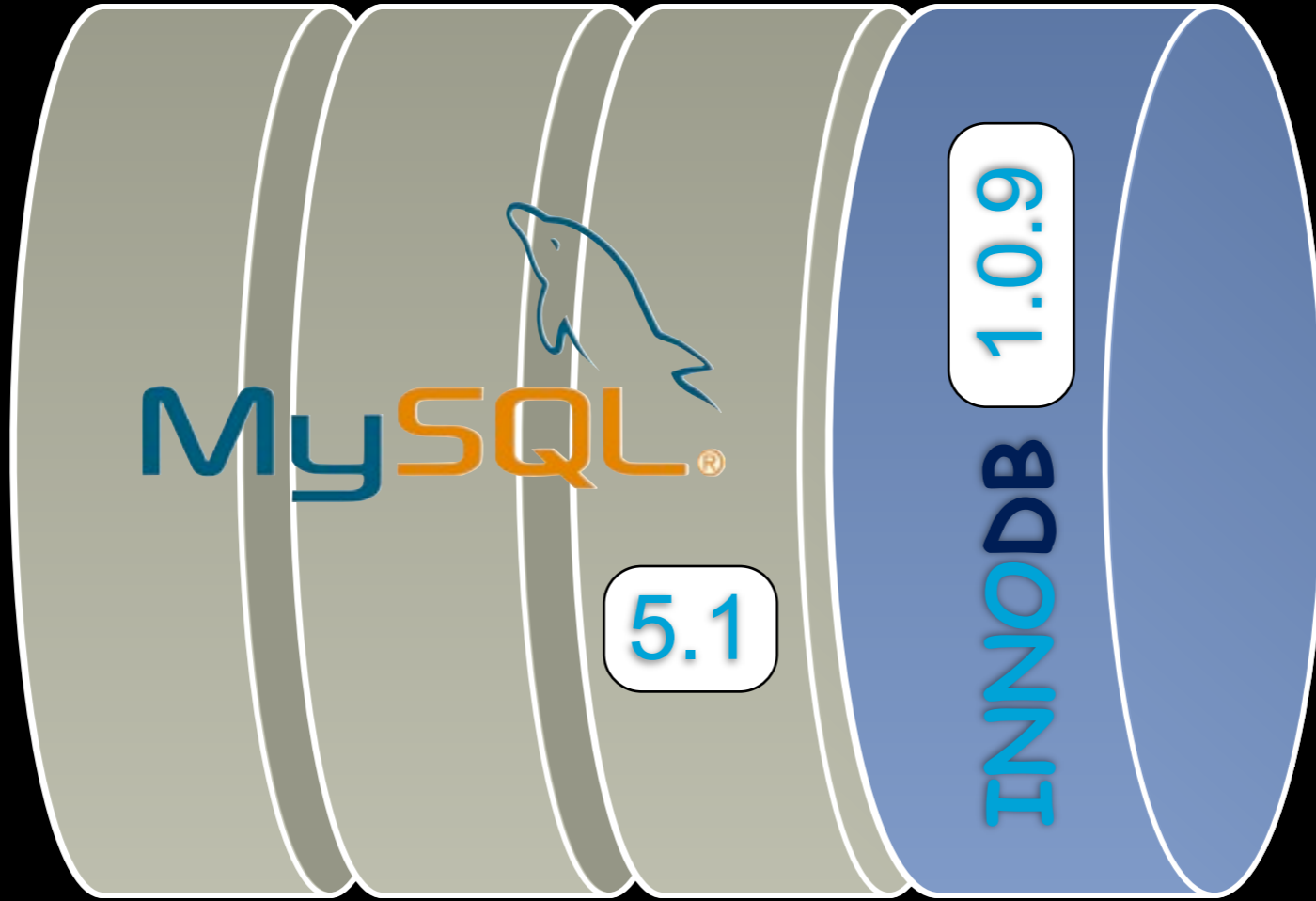
Contents

- Future: MySQL 5.5
 - ◆ **performance**
 - ★ InnoDB plugin 1.2 default engine
 - ★ Performance schema
 - ◆ ease of use (partitioning, SIGNAL)
 - ◆ reliability (semisynch replication)
- **Current: MySQL 5.1**
 - ◆ **performance: InnoDB plugin 1.0.8**

Missed announcement

- A GA release
- As of MySQL 5.1.47
- The InnoDB plugin is **GA**
- Ready to use for immediate gains

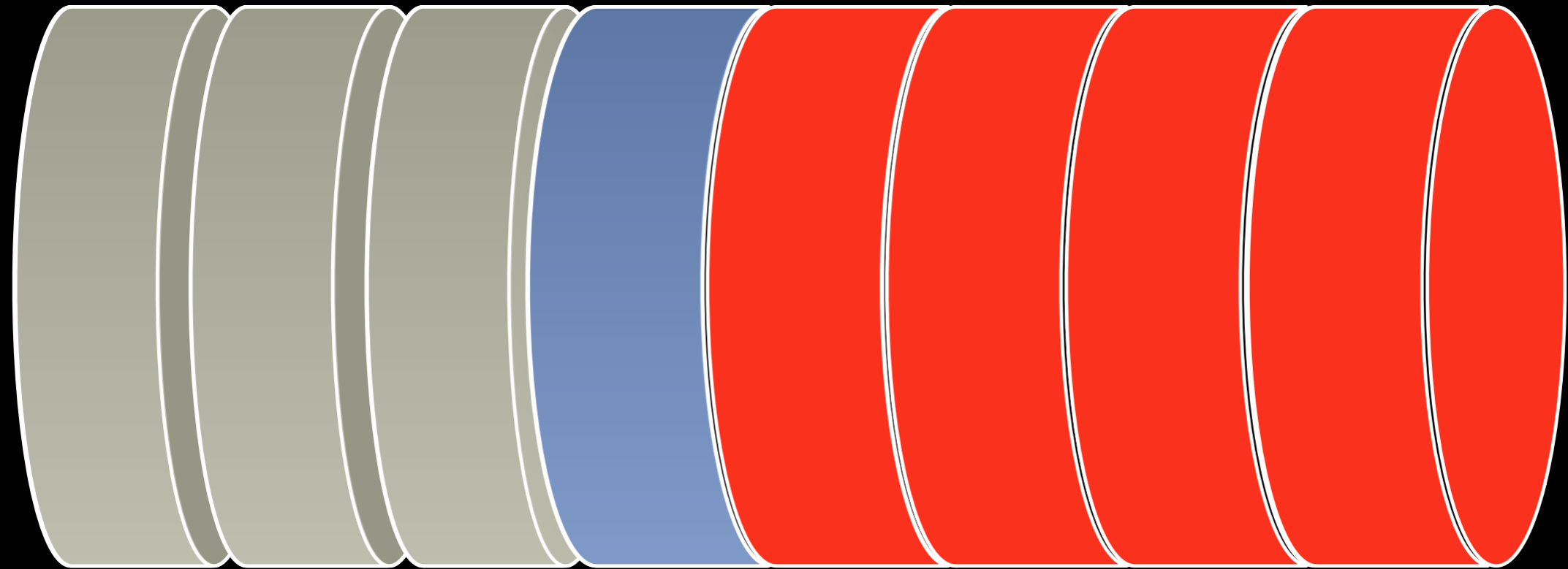




MySQL
Server

InnoDB plugin

INFORMATION
SCHEMA table plugin



Installation (1)

my.cnf

```
[mysqld]
```

```
plugin_dir = /usr/local/mysql/lib/plugin  
ignore_builtin_innodb  
plugin-load=innodb=ha_innodb_plugin.so  
default-storage-engine=InnoDB  
innodb_file_per_table=1  
innodb_file_format=barracuda  
innodb_strict_mode=1
```

Installation (1a)

my.cnf

[mysqld]

```
plugin-load=innodb=ha_innodb_plugin.so;  
innodb_trx=ha_innodb_plugin.so;  
innodb_locks=ha_innodb_plugin.so;  
innodb_lock_waits=ha_innodb_plugin.so;  
innodb_cmp=ha_innodb_plugin.so;  
innodb_cmp_reset=ha_innodb_plugin.so;  
innodb_cmpmem=ha_innodb_plugin.so;  
innodb_cmpmem_reset=ha_innodb_plugin.so  
#(all in one line with no spaces)
```

Installation (2)

```
SET GLOBAL innodb_fast_shutdown=0;
```

```
RESTART the server
```

Installation - 2nd method (1)

```
my.cnf
```

```
[mysqld]
```

```
ignore_builtin_innodb
```

Installation - 2nd method (2)

```
SET GLOBAL innodb_fast_shutdown=0;
```

```
RESTART the server
```

Installation - 2nd method (3)

```
mysql
```

```
INSTALL PLUGIN INNODB SONAME 'ha_innodb_plugin.so';  
INSTALL PLUGIN INNODB_TRX SONAME  
'ha_innodb_plugin.so';  
INSTALL PLUGIN INNODB_LOCKS SONAME  
'ha_innodb_plugin.so';  
INSTALL PLUGIN INNODB_LOCK_WAITS SONAME  
'ha_innodb_plugin.so';  
INSTALL PLUGIN INNODB_CMP SONAME  
'ha_innodb_plugin.so';  
INSTALL PLUGIN INNODB_CMP_RESET SONAME  
'ha_innodb_plugin.so';  
INSTALL PLUGIN INNODB_CMPMEM SONAME  
'ha_innodb_plugin.so';  
INSTALL PLUGIN INNODB_CMPMEM_RESET SONAME  
'ha_innodb_plugin.so';
```

Installation - 2nd method (4)

```
my.cnf
```

```
[mysqld]
```

```
default-storage-engine=InnoDB  
innodb_file_per_table=1  
innodb_file_format=barracuda  
innodb_strict_mode=1
```

Installation - 2nd method (5)

```
SET GLOBAL innodb_fast_shutdown=0;
```

```
RESTART the server
```

Installation differences

- Method 1 (plugin-load in my.cnf)
 - ✦ Only one operation
 - ✦ But error prone (one loooooong command)
 - ✦ plugins not stored in mysql.plugin table
- Method 2 (install plugin)
 - ✦ plugin info saved to mysql.plugin table
 - ✦ Easier to write
 - ✦ 2 restarts required

CAVEAT

- If you uninstall the InnoDB plugin, remember:
 - ✦ The tables are not backward compatible
 - ✦ You must uninstall all the INFORMATION_SCHEMA plugin tables BEFORE removing the InnoDB plugin
 - ✦ If the plugin is busy, it may not be removed until you restart the server

hands on

Checking installation

```
select @@version, @@innodb_version;
```

@@version	@@innodb_version
5.1.48	1.0.9



Detecting locks

```
session1> select c from t1 for update;
```

```
+-----+  
|  c  |  
+-----+  
|  aaa  |  
|  bbb  |  
|  ccc  |  
+-----+
```

Detecting locks

```
session2> select c from t1 for update;
```

```
[... waiting]
```

Detecting locks

```
session3> select i from t1 for update;
```

```
[... waiting]
```

getting locks information

SELECT

```
r.trx_id waiting_trx_id,  
r.trx_mysql_thread_id waiting_thread,  
r.trx_query waiting_query,  
b.trx_id blocking_trx_id,  
b.trx_mysql_thread_id blocking_thread,  
b.trx_query blocking_query
```

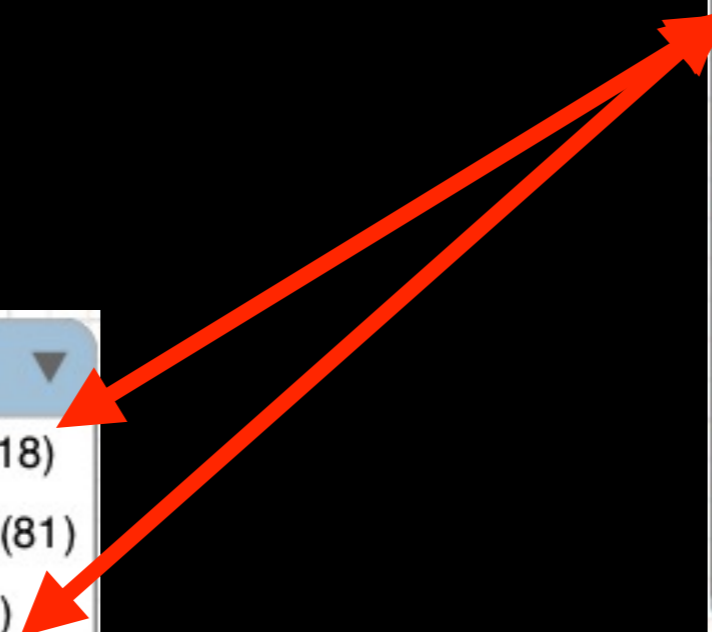
FROM

```
innodb_lock_waits w  
INNER JOIN innodb_trx b  
    ON b.trx_id = w.blocking_trx_id  
INNER JOIN innodb_trx r  
    ON r.trx_id = w.requesting_trx_id
```

getting locks information

INNODB_LOCK_WAITS	
requesting_trx_id	VARCHAR(18)
requested_lock_id	VARCHAR(81)
blocking_trx_id	VARCHAR(18)
blocking_lock_id	VARCHAR(81)

INNODB_TRX	
trx_id	VARCHAR(18)
trx_state	VARCHAR(13)
trx_started	DATETIME
trx_requested_lock_id	VARCHAR(81)
trx_wait_started	DATETIME
trx_weight	BIGINT(21)
trx_mysql_thread_id	BIGINT(21)
trx_query	VARCHAR(1024)



getting locks information


```
***** 1. row *****
waiting_trx_id: 711
waiting_thread: 3
waiting_query: select c from t1 for
update
blocking_trx_id: 710
blocking_thread: 2
blocking_query: select i from t1 for
update
```

getting locks information

```
***** 2. row *****
waiting_trx_id: 711
waiting_thread: 3
waiting_query: select c from t1 for
update
blocking_trx_id: 70F
blocking_thread: 1
blocking_query: NULL
```

getting locks information

```
***** 3. row *****
waiting_trx_id: 710
waiting_thread: 2
waiting_query: select i from t1 for
update
blocking_trx_id: 70F
blocking_thread: 1
blocking_query: NULL
```



The preceding 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.

THANKS

Let's talk!



This work is licensed under the Creative Commons Attribution-Share Alike 3.0 Unported License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/3.0/> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.