How to get MySQL to fail

Daniël van Eeden

Snow B.V.

Feb 3, 2013

Introduction

We all know we shouldn't press the button...

Introduction

We all know we shouldn't press the button... but we all want to try.

Do you know?

Do you know what happens if you would actually press the button?

Do you know?

Do you know what happens if you would actually press the button?
Does it work?

Warning

Don't try this at home!

Warning

Don't try this at home! or at work!

Warning

Don't try this at home! or at work!
But if you do: MySQL Sandbox is a great tool for experimentation.

Warning

Don't try this at home! or at work!
But if you do: MySQL Sandbox is a great tool for experimentation.
Virtual Machines can also be very usefull.

Animal testing

No animals were harmed during the experiments.

Enterprise file management

In the enterprise everything should be managed

Enterprise file management

In the enterprise everything should be managed, even files.

System tables

We all know we should not mess with the system tables (mysql.*)

System tables

We all know we should not mess with the system tables (mysql.*)
So let's try!

System tables

We all know we should not mess with the system tables (mysql.*)
So let's try! And break something!

System tables

```
mysql> ALTER TABLE mysql.user ENGINE=InnoDB;
```

System tables

```
mysql> ALTER TABLE mysql.user ENGINE=InnoDB;

ERROR 1726 (HY000): Storage engine 'InnoDB' does not support
system tables. [mysql.user]
```

System tables

```
mysql> USE mysql
Reading table information for completion of table and column
names You can turn off this feature to get a quicker startup
with -A
```

System tables

```
mysql> USE mysql
Reading table information for completion of table and column
names You can turn off this feature to get a quicker startup
with -A

Database changed
mysql> CREATE TABLE user2 SELECT * FROM user;
Query OK, 8 rows affected (0.02 sec)
Records: 8  Duplicates: 0  Warnings: 0
```

System tables

```
mysql> USE mysql
Reading table information for completion of table and column
names You can turn off this feature to get a quicker startup
with -A

Database changed
mysql> CREATE TABLE user2 SELECT * FROM user;
Query OK, 8 rows affected (0.02 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE user2 ENGINE=InnoDB;
Query OK, 8 rows affected (0.02 sec)
Records: 8  Duplicates: 0  Warnings: 0
```

System tables

```
mysql> USE mysql
Reading table information for completion of table and column
names You can turn off this feature to get a quicker startup
with -A

Database changed
mysql> CREATE TABLE user2 SELECT * FROM user;
Query OK, 8 rows affected (0.02 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE user2 ENGINE=InnoDB;
Query OK, 8 rows affected (0.02 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql> DROP TABLE user;
Query OK, 0 rows affected (0.00 sec)
```

System tables

```
mysql> RENAME TABLE user2 TO user;
ERROR 1025 (HY000): Error on rename of './mysql/user2' to
'./mysql/user' (errno: -1)
```

System tables

```
mysql> CREATE TABLE user(id int);
ERROR 1005 (HY000): Can't create table 'mysql.user' (errno: -1)
```

System tables

```
mysql> CREATE TABLE user(id int);
ERROR 1005 (HY000): Can't create table 'mysql.user' (errno: -1)

mysql> CREATE TABLE user(id int) ENGINE=MyISAM;
Query OK, 0 rows affected (0.01 sec)
```

System tables

```
mysql> CREATE TABLE user(id int);
ERROR 1005 (HY000): Can't create table 'mysql.user' (errno: -1)

mysql> CREATE TABLE user(id int) ENGINE=MyISAM;
Query OK, 0 rows affected (0.01 sec)

mysql> FLUSH PRIVILEGES;
ERROR 2013 (HY000): Lost connection to MySQL server during query
```

System tables

```
mysql> \s
ERROR 2006 (HY000): MySQL server has gone away
No connection. Trying to reconnect...
ERROR 2002 (HY000): Can't connect to local MySQL server through
socket '/tmp/mysql_sandbox5529.sock' (2)
ERROR:
Can't connect to the server
```

System tables

And now we need to fix it again.

```
$ pkill -9 mysqld_safe
```

System tables

And now we need to fix it again.

```
$ pkill -9 mysqld_safe
```

```
$ ./mysqld --skip-grant-tables &
```

And now restore the mysql.user table from backup or in any other way you like. And then restart MySQL to make sure it works properly.

System tables

Lessons learned:

System tables

Lessons learned:

1. There is protection against changing system tables to other engines than MyISAM (or NDB).

System tables

Lessons learned:

1. There is protection against changing system tables to other engines than MyISAM (or NDB).
2. Issues with system tables can cause MySQL to crash and keep crashing.

System tables

What about the general_log table?

```
mysql> ALTER TABLE general_log ENGINE=InnoDB;
ERROR 1579 (HY000): This storage engine cannot be used for log
tables"
```

System tables

What about the general_log table?

```
mysql> ALTER TABLE general_log ENGINE=InnoDB;
ERROR 1579 (HY000): This storage engine cannot be used for log
tables"

mysql> ALTER TABLE mysql.general_log ENGINE=MyISAM;
ERROR 1580 (HY000): You cannot 'ALTER' a log table if logging
is enabled
```

System tables

What about the general_log table?

```
mysql> ALTER TABLE general_log ENGINE=InnoDB;
ERROR 1579 (HY000): This storage engine cannot be used for log
tables"

mysql> ALTER TABLE mysql.general_log ENGINE=MyISAM;
ERROR 1580 (HY000): You cannot 'ALTER' a log table if logging
is enabled

mysql> SET GLOBAL general_log=OFF;
Query OK, 0 rows affected (0.01 sec)
```

System tables

What about the general_log table?

```
mysql> ALTER TABLE general_log ENGINE=InnoDB;
ERROR 1579 (HY000): This storage engine cannot be used for log
tables"

mysql> ALTER TABLE mysql.general_log ENGINE=MyISAM;
ERROR 1580 (HY000): You cannot 'ALTER' a log table if logging
is enabled

mysql> SET GLOBAL general_log=OFF;
Query OK, 0 rows affected (0.01 sec)

mysql> ALTER TABLE mysql.general_log ENGINE=MyISAM;
Query OK, 7 rows affected (0.01 sec)
Records: 7  Duplicates: 0  Warnings: 0
```

System tables

What about the general_log table?

```
mysql> ALTER TABLE general_log ENGINE=InnoDB;
ERROR 1579 (HY000): This storage engine cannot be used for log
tables"

mysql> ALTER TABLE mysql.general_log ENGINE=MyISAM;
ERROR 1580 (HY000): You cannot 'ALTER' a log table if logging
is enabled

mysql> SET GLOBAL general_log=OFF;
Query OK, 0 rows affected (0.01 sec)

mysql> ALTER TABLE mysql.general_log ENGINE=MyISAM;
Query OK, 7 rows affected (0.01 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql> SET GLOBAL general_log=ON;
Query OK, 0 rows affected (0.00 sec)
```

System tables

So InnoDB isn't suitable for logging, but MyISAM is.

System tables

So InnoDB isn't suitable for logging, but MyISAM is.
What about replication?

System tables

So InnoDB isn't suitable for logging, but MyISAM is.
What about replication?
The logging tables are not being replicated.

System tables

What about adding a column?

```
mysql> alter table mysql.user add column fosdem varchar(100);
Query OK, 9 rows affected (0.01 sec)
Records: 9  Duplicates: 0  Warnings: 0
```

System tables

What about adding a column?

```
mysql> alter table mysql.user add column fosdem varchar(100);
Query OK, 9 rows affected (0.01 sec)
Records: 9  Duplicates: 0  Warnings: 0

mysql> update mysql.user set fosdem='Hello there!';
Query OK, 9 rows affected (0.00 sec)
Rows matched: 9  Changed: 9  Warnings: 0
```

System tables

What about adding a column?

```
mysql> alter table mysql.user add column fosdem varchar(100);
Query OK, 9 rows affected (0.01 sec)
Records: 9  Duplicates: 0  Warnings: 0

mysql> update mysql.user set fosdem='Hello there!';
Query OK, 9 rows affected (0.00 sec)
Rows matched: 9  Changed: 9  Warnings: 0

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

System tables

Whether a storage engine can be used for system tables or logging tables is not listed in the "SHOW ENGINES" output.

```
Whether a
*************************** 8. row ***************************
      Engine: InnoDB
     Support: DEFAULT
     Comment: Supports transactions, row-level locking, and
      foreign keys
Transactions: YES
          XA: YES
  Savepoints: YES
```

Kill the angel!

What happens if you kill the angel process (mysqld_safe)?

Kill the angel!

What happens if you kill the angel process (mysqld_safe)?

```
$ pkill -9 mysqld_safe
```

Kill the angel!

What happens if you kill the angel process (mysqld_safe)?

```
$ pkill -9 mysqld_safe
```

The angel process will exit and the mysqld process will have to survive on it's own.

Kill the angel!

What happens if you kill the angel process (mysqld_safe)?

```
$ pkill -9 mysqld_safe
```

The angel process will exit and the mysqld process will have to survive on it's own.
You must stop mysqld to get the protection of an angel again.

Put mysqld on standby

What happens when you STOP the mysqld process?

```
mysql> \! kill -STOP 28055
mysql> SELECT COUNT(*) FROM DUAL;
+----------+
| COUNT(*) |
+----------+
|        1 |
+----------+
```

Put mysqld on standby

What happens when you STOP the mysqld process?

```
mysql> \! kill -STOP 28055
mysql> SELECT COUNT(*) FROM DUAL;
+----------+
| COUNT(*) |
+----------+
|        1 |
+----------+

$ kill -CONT 28055
```

Put mysqld on standby

What happens when you STOP the mysqld process?

```
mysql> \! kill -STOP 28055
mysql> SELECT COUNT(*) FROM DUAL;
+----------+
| COUNT(*) |
+----------+
|        1 |
+----------+

$ kill -CONT 28055

1 row in set (9.12 sec)
```

Backup the (too) easy way

What happens if we would just use tar to create a backup and just forget about snapshot tools, "FLUSH TABLES WITH READ LOCK" and all the other special backup tools.

```
$ tar cf data.tar data
tar: data/ib_logfile1: file changed as we read it

$ mv data data.old
$ tar xf data.tar

rm data/mysql_sandbox5529.pid

./start
```

Backup the (too) easy way

```
InnoDB: Log scan progressed past the checkpoint lsn 13283340
130201 16:59:21  InnoDB: Database was not shut down normally!
InnoDB: Starting crash recovery.
InnoDB: Reading tablespace information from the .ibd files...
InnoDB: Restoring possible half-written data pages from the
InnoDB: doublewrite buffer...
InnoDB: Doing recovery: scanned up to log sequence
        number 18526208
InnoDB: Doing recovery: scanned up to log sequence
        number 18814312
130201 16:59:21  InnoDB: Starting an apply batch of log records
                         to the database...
```

Backup the (too) easy way

```
InnoDB: Progress in percents: 9 10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61
62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82
83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
InnoDB: Apply batch completed
130201 16:59:21  InnoDB: Waiting for the background threads to
                        start
.130201 16:59:22 InnoDB: 1.1.8 started; log sequence
                        number 18814312
```

Backup the (too) easy way

So you might be able to recover from a 'generic' filesystem backup.

Backup the (too) easy way

So you might be able to recover from a 'generic' filesystem backup. Only do this if this is your only option.

In-flight removal of InnoDB files

Remove ib*

In-flight removal of InnoDB files

Remove ib* and nothing happens...

In-flight removal of InnoDB files

Remove ib* and nothing happens... until restart.

In-flight removal of InnoDB files

Remove ib* and nothing happens... until restart.

```
130201 18:35:38 [ERROR] Cannot find or open table test/test2
from the internal data dictionary of InnoDB though the .frm
file for the table exists. Maybe you have deleted and
recreated InnoDB data files but have forgotten to delete the
corresponding .frm files of InnoDB tables, or you have moved
.frm files to another database? or, the table contains
indexes that this version of the engine doesn't support.
See
http://dev.mysql.com/doc/refman/5.5/en/innodb-troubleshooting.ht
how you can resolve the problem.
```

In-flight removal of InnoDB files

Remove ib_logfile's and restart...

In-flight removal of InnoDB files

Remove ib_logfile's and restart...

```
130201 17:05:30  InnoDB: Log file ./ib_logfile0 did not
                         exist: new to be created
InnoDB: Setting log file ./ib_logfile0 size to 5 MB
InnoDB: Database physically writes the file full: wait...
130201 17:05:30  InnoDB: Log file ./ib_logfile1 did not
                         exist: new to be created
InnoDB: Setting log file ./ib_logfile1 size to 5 MB
InnoDB: Database physically writes the file full: wait...
130201 17:05:30 InnoDB: highest supported file format is
                        Barracuda.
InnoDB: The log sequence number in ibdata files does not match
InnoDB: the log sequence number in the ib_logfiles!
130201 17:05:30  InnoDB: Database was not shut down normally!
InnoDB: Starting crash recovery.
```

In-flight removal of InnoDB files

If you copy back old ib_logfiles.

```
130203 11:48:36  InnoDB: Error: page 1001 log sequence number
InnoDB: 41465405 is in the future! Current system log sequence
InnoDB: number 41427061. Your database may be corrupt or you
InnoDB: may have copied the InnoDB tablespace but not the
InnoDB: InnoDB log files. See
InnoDB:
http://dev.mysql.com/doc/refman/5.5/en/forcing-innodb-recovery.h
InnoDB: for more information.
```

In-flight removal of InnoDB files

If the InnoDB files are somehow gone, Don't stop the server!

In-flight removal of InnoDB files

If the InnoDB files are somehow gone, Don't stop the server! Try to make a backup with mysqldump first. Tools which copy the data files probably won't work.

In-flight removal of InnoDB files

If the InnoDB files are somehow gone, Don't stop the server! Try to make a backup with mysqldump first. Tools which copy the data files probably won't work. This is also true if your suddenly find out you're using a diskless storage system.

Funny views

```
mysql> create view view_test as select * from mysql.user;
Query OK, 0 rows affected (0.04 sec)
```

Funny views

```
mysql> create view view_test as select * from mysql.user;
Query OK, 0 rows affected (0.04 sec)

$ mv test/view_test.frm test/view_test2.frm
```

Funny views

```
mysql> create view view_test as select * from mysql.user;
Query OK, 0 rows affected (0.04 sec)

$ mv test/view_test.frm test/view_test2.frm

mysql> select * from view_test;
ERROR 13 (HY000): Can't get stat of './test/view_test.frm'
(Errcode: 2)
```

Funny views

```
mysql> create view view_test as select * from mysql.user;
Query OK, 0 rows affected (0.04 sec)

$ mv test/view_test.frm test/view_test2.frm

mysql> select * from view_test;
ERROR 13 (HY000): Can't get stat of './test/view_test.frm'
(Errcode: 2)

mysql> flush table view_test;
Query OK, 0 rows affected (0.01 sec)
```

Funny views

```
mysql> create view view_test as select * from mysql.user;
Query OK, 0 rows affected (0.04 sec)

$ mv test/view_test.frm test/view_test2.frm

mysql> select * from view_test;
ERROR 13 (HY000): Can't get stat of './test/view_test.frm'
(Errcode: 2)

mysql> flush table view_test;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from view_test;
ERROR 1146 (42S02): Table 'test.view_test' doesn't exist
```

Funny views

```
mysql> select COUNT(*) from view_test2;
+----------+
| COUNT(*) |
+----------+
|        8 |
+----------+
1 row in set (0.00 sec)
```

Funny views

```
mysql> select COUNT(*) from view_test2;
+----------+
| COUNT(*) |
+----------+
|        8 |
+----------+
1 row in set (0.00 sec)
```

You can edit a view and change the definer!

Funny views

```
echo bla > bla.frm
```

Funny views

```
echo bla > bla.frm

mysql> show table status like 'bla'\G
130201 17:31:53 [ERROR] /home/dveeden/opt/mysql/5.5.29/bin/mysql
                  Incorrect information in file: './test/bla.frm'
*************************** 1. row ***************************
           Name: bla
         Engine: NULL
        Version: NULL
     Row_format: NULL
           Rows: NULL
 Avg_row_length: NULL
    Data_length: NULL
Max_data_length: NULL
   Index_length: NULL
      Data_free: NULL
 Auto_increment: NULL
    Create_time: NULL
    Update_time: NULL
     Check_time: NULL
```

Funny views

```
mysql> show warnings;
+---------+------+-------------------------------+
| Level   | Code | Message                       |
+---------+------+-------------------------------+
| Warning | 1033 | Incorrect information in file:|
                                 './test/bla.frm' |
+---------+------+-------------------------------+
1 row in set (0.00 sec)
```

Funny views

It's not possible to remove the files created with SELECT..INTO DUMPFILE.

Funny views

It's not possible to remove the files created with SELECT..INTO DUMPFILE. Or is it?

Funny views

It's not possible to remove the files created with SELECT..INTO DUMPFILE. Or is it?

```
mysql> SELECT '0000000000000000000000000000000000000000
    '> 0000000000000000000000000000000000000000 Hello!'
    -> INTO DUMPFILE 'test/aaa.frm';
Query OK, 1 row affected (0.00 sec)
```

Funny views

It's not possible to remove the files created with SELECT..INTO DUMPFILE. Or is it?

```
mysql> SELECT '0000000000000000000000000000000000000000
    '> 0000000000000000000000000000000000000000 Hello!'
    -> INTO DUMPFILE 'test/aaa.frm';
Query OK, 1 row affected (0.00 sec)

mysql> DROP TABLE aaa;
Query OK, 0 rows affected (0.00 sec)
```

The mysqldump utility will stop when encountering the first broken table.

Funny views

It's not possible to remove the files created with SELECT..INTO DUMPFILE. Or is it?

```
mysql> SELECT '0000000000000000000000000000000000000000
    '> 0000000000000000000000000000000000000000 Hello!'
    -> INTO DUMPFILE 'test/aaa.frm';
Query OK, 1 row affected (0.00 sec)

mysql> DROP TABLE aaa;
Query OK, 0 rows affected (0.00 sec)
```

The mysqldump utility will stop when encountering the first broken table. So you could create aaa.frm and break the backup!

FAT MySQL

Let's mount a FAT filesystem on /var/lib/mysql and have a try.

FAT MySQL

Let's mount a FAT filesystem on /var/lib/mysql and have a try.

```
130201 18:11:55 [Warning] Setting lower_case_table_names=2
because file system for /var/lib/mysql/ is case insensitive
```

FAT MySQL

Let's mount a FAT filesystem on /var/lib/mysql and have a try.

```
130201 18:11:55 [Warning] Setting lower_case_table_names=2
because file system for /var/lib/mysql/ is case insensitive
```

That nice.

FAT MySQL

Let's mount a FAT filesystem on /var/lib/mysql and have a try.

```
130201 18:11:55 [Warning] Setting lower_case_table_names=2
because file system for /var/lib/mysql/ is case insensitive
```

That nice. We just found an easy way to test if our applications might have issues when deployed on windows.

MySQL and full disks

Myth: MySQL doesn't handle full disk situations.

MySQL and full disks

Myth: MySQL doesn't handle full disk situations.

```
mysql> INSERT INTO test(name) SELECT name FROM test;
Query OK, 96 rows affected (0.09 sec)
Records: 96  Duplicates: 0  Warnings: 0
```

MySQL and full disks

Myth: MySQL doesn't handle full disk situations.

```
mysql> INSERT INTO test(name) SELECT name FROM test;
Query OK, 96 rows affected (0.09 sec)
Records: 96  Duplicates: 0  Warnings: 0

mysql> INSERT INTO test(name) SELECT name FROM test;
Query OK, 192 rows affected (0.13 sec)
Records: 192  Duplicates: 0  Warnings: 0
```

MySQL and full disks

Myth: MySQL doesn't handle full disk situations.

```
mysql> INSERT INTO test(name) SELECT name FROM test;
Query OK, 96 rows affected (0.09 sec)
Records: 96  Duplicates: 0  Warnings: 0

mysql> INSERT INTO test(name) SELECT name FROM test;
Query OK, 192 rows affected (0.13 sec)
Records: 192  Duplicates: 0  Warnings: 0

mysql> INSERT INTO test(name) SELECT name FROM test;
ERROR 1114 (HY000): The table 'test' is full
```

MySQL and full disks

Myth: MySQL doesn't handle full disk situations.

```
mysql> INSERT INTO test(name) SELECT name FROM test;
Query OK, 96 rows affected (0.09 sec)
Records: 96  Duplicates: 0  Warnings: 0

mysql> INSERT INTO test(name) SELECT name FROM test;
Query OK, 192 rows affected (0.13 sec)
Records: 192  Duplicates: 0  Warnings: 0

mysql> INSERT INTO test(name) SELECT name FROM test;
ERROR 1114 (HY000): The table 'test' is full

mysql> alter table test engine=MyISAM;
ERROR 3 (HY000): Error writing file
'./test/#sql-18df_25.frm' (Errcode: 28)
```

MySQL and full disks

Myth: MySQL doesn't handle full disk situations.

```
mysql> INSERT INTO test(name) SELECT name FROM test;
Query OK, 96 rows affected (0.09 sec)
Records: 96  Duplicates: 0  Warnings: 0

mysql> INSERT INTO test(name) SELECT name FROM test;
Query OK, 192 rows affected (0.13 sec)
Records: 192  Duplicates: 0  Warnings: 0

mysql> INSERT INTO test(name) SELECT name FROM test;
ERROR 1114 (HY000): The table 'test' is full

mysql> alter table test engine=MyISAM;
ERROR 3 (HY000): Error writing file
'./test/#sql-18df_25.frm' (Errcode: 28)

$ perror 28
OS error code  28:  No space left on device
```

MySQL and full disks

```
mysql> optimize table test;
+-----------+----------+----------+----------------------------+
| Table     | Op       | Msg_type | Msg_text                   |
+-----------+----------+----------+----------------------------+
| test.test | optimize | note     | Table does not support     |
                                  | optimize, doing recreate   |
                                  | + analyze instead          |
| test.test | optimize | error    | Error writing file         |
                        './test/#sql-18df_25.frm' (Errcode: 28)|
| test.test | optimize | status   | Operation failed           |
+-----------+----------+----------+----------------------------+
3 rows in set, 1 warning (0.00 sec)
```

MySQL and full disks

Let's remove the ib_logfiles and enlarge them, while the disk is full.

```
130201 18:27:17  InnoDB: Log file ./ib_logfile0 did not exist:
                                                 new to be created
InnoDB: Setting log file ./ib_logfile0 size to 10 MB
InnoDB: Database physically writes the file full: wait...
130201 18:27:17  InnoDB: Log file ./ib_logfile1 did not exist:
                                                 new to be created
InnoDB: Setting log file ./ib_logfile1 size to 10 MB
InnoDB: Database physically writes the file full: wait...
130201 18:27:17  InnoDB: Error: Write to file ./ib_logfile1
                                failed at offset 0 0.
InnoDB: 1048576 bytes should have been written, only -1 were
InnoDB: written.
InnoDB: Operating system error number 28.
```

MySQL and full disks

```
InnoDB: Check that your OS and file system support files of
InnoDB: this size.
InnoDB: Check also that the disk is not full or a disk quota
InnoDB: exceeded.
InnoDB: Error number 28 means 'No space left on device'.
InnoDB: Some operating system error numbers are described at
InnoDB: http://dev.mysql.com/doc/refman/5.5/en
                              /operating-system-error-codes.html
InnoDB: Error in creating ./ib_logfile1: probably out of
InnoDB: disk space
```

Intentional corruption

Let's truncate the tablespace for table t1.

```
$ > t1.ibd
```

Nothing seems to happen. We can still SELECT and INSERT. But when we restart:

```
InnoDB: Error: tablespace id is 2 in the data dictionary
InnoDB: but in file ./test/t1.ibd it is 0!
130202 15:52:35  InnoDB: Assertion failure in thread
              139652071266048 in file fil0fil.c line 768
InnoDB: We intentionally generate a memory trap.
```

Intentional corruption

Let's truncate the tablespace for table t1.

```
$ > t1.ibd
```

Nothing seems to happen. We can still SELECT and INSERT. But when we restart:

```
InnoDB: Error: tablespace id is 2 in the data dictionary
InnoDB: but in file ./test/t1.ibd it is 0!
130202 15:52:35  InnoDB: Assertion failure in thread
              139652071266048 in file fil0fil.c line 768
InnoDB: We intentionally generate a memory trap.
```

And if we copy the ibd file from an identical table

Intentional corruption

Let's truncate the tablespace for table t1.

```
$ > t1.ibd
```

Nothing seems to happen. We can still SELECT and INSERT. But when we restart:

```
InnoDB: Error: tablespace id is 2 in the data dictionary
InnoDB: but in file ./test/t1.ibd it is 0!
130202 15:52:35  InnoDB: Assertion failure in thread
              139652071266048 in file fil0fil.c line 768
InnoDB: We intentionally generate a memory trap.
```

And if we copy the ibd file from an identical table

```
nnoDB: Error: tablespace id is 2 in the data dictionary
InnoDB: but in file ./test/t1.ibd it is 3!
```

Triggers on system tables

```
mysql> CREATE TRIGGER mu_ins BEFORE INSERT ON mysql.user
    -> FOR EACH ROW INSERT INTO test.foo VALUES('test');
ERROR 1465 (HY000): Triggers can not be created on system tables
```

Just copying and modifying a .TRG file works. Make sure the permissions are correct otherwise an error will occur and prevent MySQL from starting.

Triggers on system tables

```
mysql> CREATE TRIGGER mu_ins BEFORE INSERT ON mysql.user
    -> FOR EACH ROW INSERT INTO test.foo VALUES('test');
ERROR 1465 (HY000): Triggers can not be created on system tables
```

Just copying and modifying a .TRG file works. Make sure the permissions are correct otherwise an error will occur and prevent MySQL from starting.

```
130202 16:29:43 [ERROR] Fatal error: Can't open and lock
privilege tables: File './mysql/user.TRG' not found
(Errcode: 13)
```

Triggers on system tables

```
mysql> CREATE TRIGGER mu_ins BEFORE INSERT ON mysql.user
    -> FOR EACH ROW INSERT INTO test.foo VALUES('test');
ERROR 1465 (HY000): Triggers can not be created on system tables
```

Just copying and modifying a .TRG file works. Make sure the permissions are correct otherwise an error will occur and prevent MySQL from starting.

```
130202 16:29:43 [ERROR] Fatal error: Can't open and lock
privilege tables: File './mysql/user.TRG' not found
(Errcode: 13)
```

```
mysql> alter table mysql.general_log engine=myisam;
ERROR 29 (HY000): File './mysql/general_log.TRG' not found
(Errcode: 13)
```

Triggers on system tables

```
mysql> INSERT INTO mysql.user(Host,User,Password)
    -> VALUES('localhost','user','pass');
ERROR 1146 (42S02): Table 'test.foo' doesn't exist
```

Triggers on system tables

```
mysql> INSERT INTO mysql.user(Host,User,Password)
    -> VALUES('localhost','user','pass');
ERROR 1146 (42S02): Table 'test.foo' doesn't exist

mysql> CREATE USER test@test;
Query OK, 0 rows affected (0.00 sec)
```

Shared management of config files

You want everybody to help you create the best my.cnf possible?

Shared management of config files

You want everybody to help you create the best my.cnf possible?
Then MySQL will ignore you!

```
Warning: World-writable config file
'/etc/mysql/my.cnf' is ignored
```

Shared datadir

```
root@ubuntu-mysql01:~# pgrep -fl mysqld
11614 mysqld --defaults-file=/etc/mysql/my.cnf
11731 mysqld --defaults-file=/etc/mysql/my-2.cnf
root@ubuntu-mysql01:~# mysql -ptest \
> -S /var/run/mysqld/mysqld.sock -e 'SELECT @@datadir'
+-----------------+
| @@datadir       |
+-----------------+
| /var/lib/mysql/ |
+-----------------+
root@ubuntu-mysql01:~# mysql -ptest \
> -S /var/run/mysqld/mysqld2.sock -e 'SELECT @@datadir'
+-----------------+
| @@datadir       |
+-----------------+
| /var/lib/mysql/ |
+-----------------+
```

Shared datadir

This only works if you need to

1. Give each instance a different port

Shared datadir

This only works if you need to

1. Give each instance a different port
2. Give each instance a different socket

Shared datadir

This only works if you need to

1. Give each instance a different port
2. Give each instance a different socket
3. Add skip-innodb

Shared datadir

This only works if you need to

1. Give each instance a different port
2. Give each instance a different socket
3. Add skip-innodb
4. Add default-storage-engine=MyISAM

Shared datadir

This only works if you need to

1. Give each instance a different port
2. Give each instance a different socket
3. Add skip-innodb
4. Add default-storage-engine=MyISAM
5. Remove skip-external-locking

Other things you must not do

1. Use long distance SQL

Other things you must not do

1. Use long distance SQL
2. Use MySQL's rollback features

Other things you must not do

1. Use long distance SQL
2. Use MySQL's rollback features
3. Remove binlogs with rm

Other things you must not do

1. Use long distance SQL
2. Use MySQL's rollback features
3. Remove binlogs with rm
4. Run MySQL with default security

Other things you must not do

1. Use long distance SQL
2. Use MySQL's rollback features
3. Remove binlogs with rm
4. Run MySQL with default security
5. Run MySQL with default settings

Other things you must not do

1. Use long distance SQL
2. Use MySQL's rollback features
3. Remove binlogs with rm
4. Run MySQL with default security
5. Run MySQL with default settings or try to tune everything at once.

Conclusion

Experimenting can be a great way to learn about how a product behaves when something (or someone) makes and accidental mistake. It also shows that security is imporant.

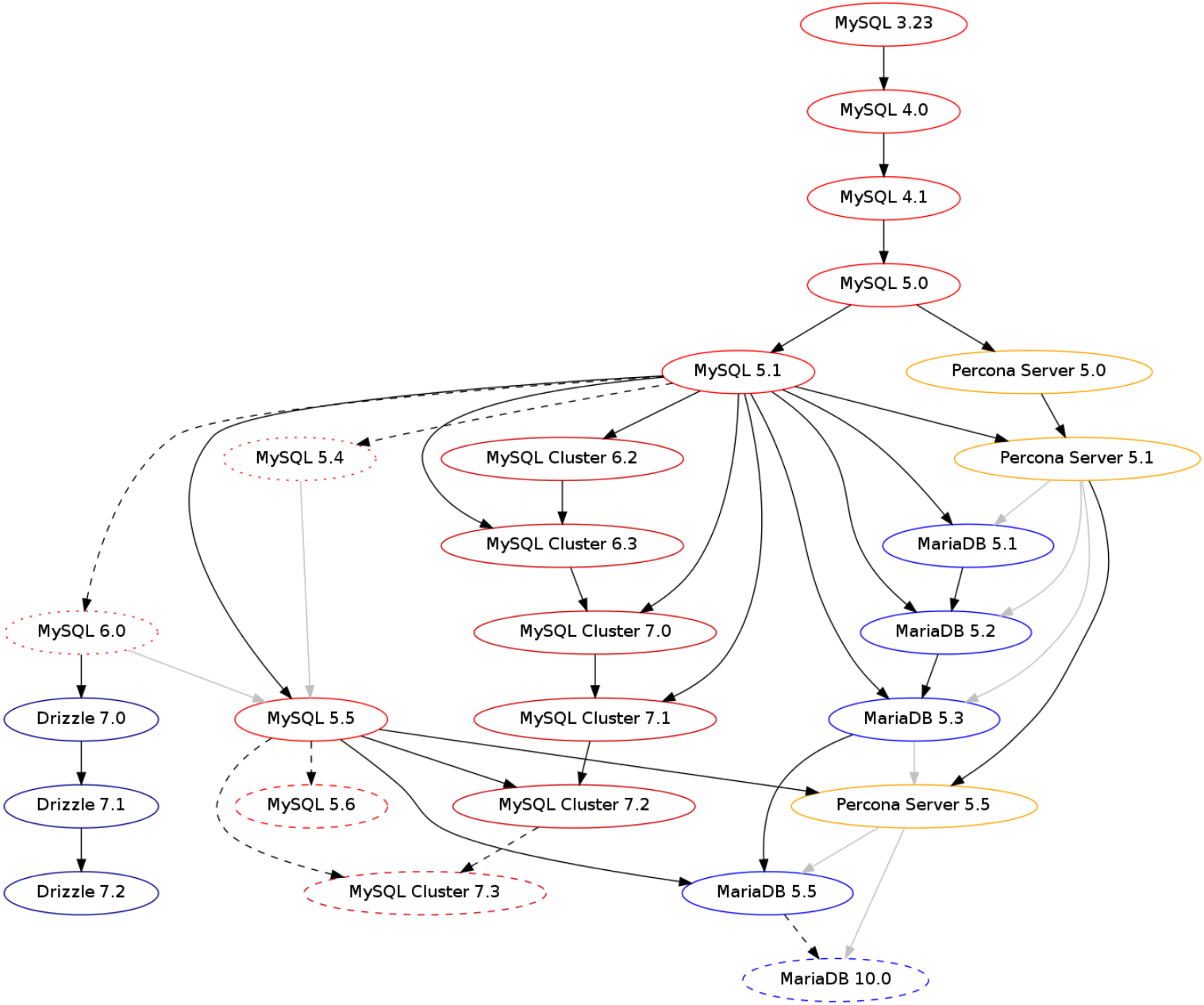
Conclusion

Experimenting can be a great way to learn about how a product behaves when something (or someone) makes and accidental mistake. It also shows that security is imporant. MyISAM allowed you to move files around, use share datadirs and more.

Conclusion

Experimenting can be a great way to learn about how a product behaves when something (or someone) makes and accidental mistake. It also shows that security is imporant. MyISAM allowed you to move files around, use share datadirs and more. MySQL 5.6 will make it easier to do those things with InnoDB.

History


MySQL 3.23
MySQL 4.0
MySQL 4.1
MySQL 5.0
MySQL 5.1
Percona Server 5.0
MySQL 5.4
MySQL Cluster 6.2
Percona Server 5.1
MySQL Cluster 6.3
MariaDB 5.1
MySQL 6.0
MySQL Cluster 7.0
MariaDB 5.2
Drizzle 7.0
MySQL 5.5
MySQL Cluster 7.1
MariaDB 5.3
Drizzle 7.1
MySQL 5.6
MySQL Cluster 7.2
Percona Server 5.5
Drizzle 7.2
MySQL Cluster 7.3
MariaDB 5.5
MariaDB 10.0

Daniël van Eeden

http://databaseblog.myname.nl

http://www.meetup.com/MySQL-User-Group-NL/

http://github.com/dveeden