Future Design Hurdles to Tackle in the MySQL Server

OR

The Future of MySQL (The Project)

Michael Widenius
MySQL Fellow
2008-04-17

What this talk is about

- MySQL server limitations or
  - Skeletons in the closet
  - “Official secrets” that no one dares to talk about
  - Embarrassing things in the server
- Why MySQL is 'not' an open source project

For each item, I will try to describe

- Nature of the problem
- Background why the problem exists
- How the problem affects our users
- How I think the problem should be fixed

All things, said, indicated or thought during this talk are my own are not necessary the opinions of MySQL or Sun

Why this talk

- MySQL and Sun needs to become more transparent in what they are doing
- It's easier to discuss and act when you have facts and don't have to rely on rumors or interpretations
- When users know the limitations they can go around them
- When developers know about the problem they can help us with coming up with better solutions and even help us fix the problems
- It's easier to trust someone if they acknowledge a problem and it's even better when they have a solution for it
- Open source/free software is based on transparency and trust

Threads

Problems

- One connection per thread doesn't work in all cases
- No way to give priority to thread
- No way to ensure that we have X active threads running

Symptoms

- Too many context switches
- We are not using the multi-core CPU efficiently
- MySQL doesn't scale that well after 4–8 cores when many active connections

Solution (one of many)

- --thread-handling=--pool-of-threads (MySQL 6.0)

Still lots of work to be done to

- Need to get in all InnoDB concurrency patches
- Spawn more threads when threads are blocked
- Removing overall mutex contention in the server
- Give higher/lower priority to some threads
- Allow 'super' user to login when all threads are in use

Memory as a resource

Problems

- No single memory allocator (server/engines)
- Sort buffer size (and others) are not flexible
- Memory engine needs work (no varchar/blob handling)

Symptoms

- Running out of memory with many active threads
- Not using all memory when running with few threads

Solutions

- Review and apply contributed patch for memory engine (varchar and blobs stored efficiently)
- Introduce an interface to check memory usage (internally and externally)
- Allocate large buffers flexibly depending on how much memory is in use
- Better communication between MySQL and storage engines on how to allocate dynamic memory

Metadata

Problems

- MySQL doesn't have any online operations (a few fast operations)
- Table information is generated from (.frm) files on demand
- No versioning of meta data (tables, stored procedure etc)
- Information schema tables materialized for each query
- Meta data operations are not ACID

Symptoms

- Slow information schema / SHOW commands
- No versioning makes it impossible to make online ALTER TABLE

Solutions

- Store meta data in transactional tables in the 'mysql' database
- If table doesn't exist in metadata tables, fall back to .frm files (allows easy table shipping)
- Work is in progress for data dictionary; We had an architecture meeting about this last week. Scheduled for 6.1

No solution yet for versioning of meta data

- Versioning of meta data will require notable changes in both MySQL level and the storage engines that supports this.

Privileges

Problems

- Not modular/pluggable (LDAP?)
- Hard to extend with other authentication methods
- No ROLES

Symptoms

- Hard to maintain lots of users
- Hard (impossible?) to use external authentication

(No) Solution

- Get the community to implement an authentication module using LDAP and then use this as a base to make it pluggable
- We have an old ROLE patch that we need someone to work upon it
- Contribute a patch for ROLE or tell MySQL/Sun to increase the priority for this. (preliminary scheduled for 7.x (2011 ?))

Pluggable storage engines

Problems

- Storage engines are depending on internal MySQL structures (TABLE, Field, ...)

Symptoms

- Storage engines can (mostly) only be used with the exact MySQL server version they where compiled against

Solution (?)

- We are constantly cleaning up the interface but some things may never be solved
- Think of this as a **loadable** storage engine interface

Items (expressions)

SELECT a,b FROM t1 WHERE a>10

Problems

- Item's hold temporary values during evaluation
- Item's are not re-entrant

Symptoms

- Hard to make parallel execution of statement
- We can't cache prepared statements and stored procedures between connections

Solutions

- Implement cloning of item's or evaluate items via pre-created buffer
  - item->initialize_buffer(&thd->eval_buffer), item->val(thd->eval_buffer)
- If clone, clone prepared statements both for prepare and evaluation
- Probably in MySQL 6.1 (?)

Parser

Problems

- State machine to large
- Not pluggable
- Not cacheable
- Still Bison
- Bad error messages (especially for stored procedures)

Symptoms

- Parsing has a high overhead for simple queries (12 % time spent in parser)
- Parser takes a lot of code space

(No) Solution

- We have wanted to have a new parser for 4 years
- Write and implement a new recursive decent (pluggable) parser.

Modularity

Problems

- Server is very monolithic
- Few defined interfaces (not often stable)
- Server and libraries are not documented
- Multiple Execution paths (UPDATE & SELECT)
- No rewrite state for optimizer

Symptoms

- Hard for change code without introducing bugs
- Hard for newcomers to understand the server

Solutions

- All new code that is added is well documented
- Make dbug, mysys and strings external libraries (under bsd license)
- We would need one year to do a 're-engineering' release (should have been done after 5.0!)
- Make more things pluggable (will enforce better interfaces)

Stored procedures/triggers

Problems

- Stored procedures are not cacheable across connections
- We only support SQL
- Pre-locking for all tables (deadlock-free algorithm)
- All cursors are materialized (no scrollable)
- Trigger code is not shared across opened tables
- No constraint of resources
- We don't support SP as table: **SELECT * FROM (CALL SP()) AS t**

Symptoms

- MySQL uses more memory than needed
- Hard to debug and profile

Solution

- Work is going on to support external languages (Antony Curtis & Eric Herman)
  - Perl is supported (in process)
  - Java and XML-callout? is supported
- Removing pre-locking will require serious rework (7.x)
- To fix caching we need a solution for the re-entrant items

Replication

Problems

- Replication is not fail safe
- No synchronous option
- No checking consistency option
- Setup and resync slave is complicated
- Single thread on the slave
- No Multi+Master
- Only InnoDB synchronizes with the replication (binary) log

Symptoms

- Slave can't catch up with master
- Hard to do clean fail overs
- We are dependent on InnoDB

(No)Solutions

- Use backup to setup slave
- Replicate CHECKSUM TABLE and do consistency checking on slave
- Most other things are hanging in the air

Client/server protocol

Problems

- Only one running query/connection (no async option)
- Possible to connect to wrong resource when using
  - mysql –socket=/tmp/mysql.sock –port=30 **without –**protocol=tcp
- Not all statements can be prepared (LOCK TABLES)
- Not all languages support prepared statement protocol
- Prepared statements do not support multi-statements/stored procedures with multi-results

Symptoms

- User's are not using prepared statements
- One must use many connections to handle concurrent queries

Solution (?)

- We plan to support prepare for all statements (6.1 ?)
- We should add support binding values to arrays (6.1 ?)
- No plan to support multi-statements or multi-result stored procedures
  - To support this, we have to change client interface to support multiple bindings at once (a bit complex)

Table names

Problems

- Tables are stored as files (name.frm)
- File system may be case sensitive (Unix) or not (Windows, Mac)
- Falcon has it's own interpretation of how things should be done

Symptoms

- **SELECT * from TableName** and **SELECT * from 'TABLEname'** MAY or MAY NOT refer to different tables depending on file system
- Hard to move applications between operating systems
- Doing ALTER TABLE of all tables to Falcon may delete data from tables on Unix for table names that only differ in case (Bug#22166)

Solutions

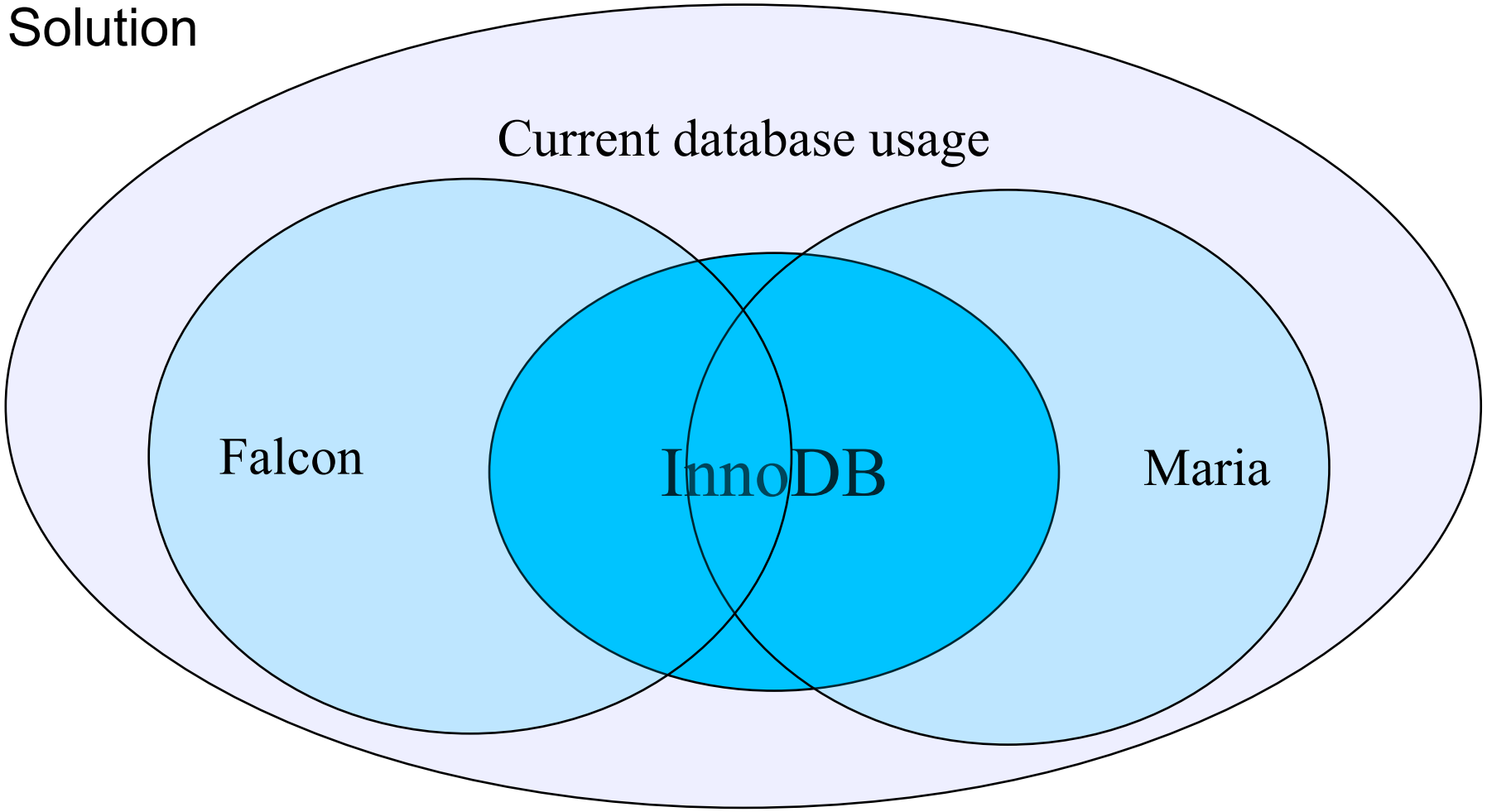
- Use --lower-case-table-names when running Windows and Mac
- Add modes to be backward compatible, ANSI compatible and PostgreSQL compatible. --table-quoting= one of
  - everything-is-always-quoted (Unix default now)
  - nothing-is-quoted-even-if-there-are-quotes (Windows default now)
  - not-quoted-table-names-converts-to-lower-case (PostgreSQL)
  - not-quoted-table-names-converts-to-upper-case (ANSI)

Why Falcon and Maria

Problem

- MySQL/Sun doesn't have it's own transactional storage engine

Solution



Phone home

Problems

- MySQL/Sun don't have a clue what features MySQL users are using

Symptoms

- Endless internal discussions of:
  - What features can be deprecated
  - Default values for future (strict mode, default engine, memory option)
  - What is the impact of a bug in a certain feature (affects bug fix priorities)

Solution

- Add 'Phone home functionally to the MySQL server
  - Off by default
  - Can **not** be used to identify users
  - Low impact; Would send data (basically show status + hardware info) to phone-home-server at startup and once every week
  - Fully documented (including source) of what gets sends to the phone-home-server
  - Phone home server should be free and available and installable locally for internal statistics purposes
  - **All** accumulated data should be accessible by community as aggregates

Open source project ?

- Very hard to contribute code / ideas
  - Current development plan aims for “at least 8 patches” this year
- Not many MySQL developers that are also using MySQL
- Customers are using their production systems as bug-finders
- Road map and decision making is not transparent
- Server components are discussed to be released as crippleware

Symptoms

- Community is not contributing to the MySQL server code base
- User base is growing slowly and linearly, not exponentially as the web
- Very few small features or usability fixes in 5.1
- No new 'innovative' features in 5.1 from real life applications

Solution

- Change MySQL development model to attract outside developers
- Give outside developers **commit** and **decision** rights to the MySQL server code base (similar constraints as for internal developers)
- Do this by aiming at a high target: “MySQL should within 2 years have as many outside core contributors as PostgreSQL”.
  - Yes, we have a lot to learn from how PostgreSQL is developed!

Release policy

Problems

- MySQL are constantly shipping releases before they are “ready”
- Benchmarks are given out with shows “partial truths”

Symptoms

- MySQL 5.1 was declared RC way too early
- Features are removed in “release candidate” releases
- Major code changes are done each month in RC code
- Users are not happy with the releases until 6 months after GA (see 5.0)
- Critical bugs are still open in 5.1 and not scheduled to be fixed before GA:
  - Bug #989, which allows anyone with rights to any database that is replicated to take down all slaves
  - Bug #30414, which shows that the new 5.1 feature “logging to tables” is to slow to be usable

Solution

- Wait to declare something GA until code stabilizes and critical bugs are fixed
- Create a **release policy** and independent **release policy board** that can't be manipulated by people in charge of server development (to not allow anyone to sacrifice quality to reach personal goals)

The good news

We know there are problems and have a good clue of how to fix (most) of them

Sun is more open source/free software friendly than MySQL AB has been lately and is driving MySQL in the right direction

Sun really understands developers and I am even more convinced that they will provide the best possible home for the MySQL project!