Welcome!
Performance 101 for Small Web Apps

Jens Bollmann
jens@skysql.com

Principal consultant and trainer within the Professional Services group at SkySQL Ab.
SkySQL Ab is the alternative source for software, services and support for MySQL related technologies.

- Founded by key people from MySQL AB
- Funded by OnCorps Capital, David Axmark, Open Ocean, Founders and SkySQL employees
- Backed by Product Engineering Monty Program Ab
- Most of the team are ex-MySQL AB employees, with more joining all the time!
- Operating in 13 countries so far...
Performance Tuning is about finding bottlenecks.

- Little point in tuning something if it is not the bottleneck
- Decide if your bottleneck is in the database or the client
- On a shared host, the problem could be someone else!

**Questions**

How many queries do you execute per page view?

Any _write_ queries that run arbitrarily on every hit?

Do you execute queries inside loops?

Do you use `SELECT *`?

ORDER BY RAND()?
Slow Queries
Performance 101 for Small Web Apps

Shared hosts make gathering statistics hard.

- Often no access to features: slow query log, show profile
- So, do it yourself:
  - Wrap mysql_query() or equivalent call on the application side
  - Log time stamp, SQL, and execution time somewhere (error log / table)
  - You may have to do it all on production! (real world load / traffic)

- Can you download a copy of your dataset, for testing?
  - If not, EXPLAIN can still be run from a script or web tool like phpMyAdmin

- Check slow queries with EXPLAIN
  - Look first at rows column: big number = problem
  - Look second at type column: ALL = problem (can you index?)

- Use SET SESSION
Are either of these patterns familiar?

SELECT COUNT(*) FROM ... WHERE ...
SELECT ... FROM ... JOIN ... WHERE ... ORDER ... LIMIT ...

or

SELECT SQL_CALC_FOUND_ROWS ... FROM ... JOIN ... ORDER ... LIMIT ...
SELECT FOUND_ROWS()

- First solution needs to process dataset twice
  - Depends on WHERE complexity. Exacerbated by user controlled filtering!

- Second solution is usually an improvement on the first
  - But SQL_CALC_FOUND_ROWS requires entire result set to be generated!

- In both cases, the ORDER / LIMIT step can become expensive as dataset grows
Option #1: Limit the size of the dataset

SELECT ... WHERE stamp > now() - interval 6 month

or

SELECT ... WHERE id > N

- Or archive old data to another table
- Remember to add an index to stamp!
- `id` is assumed to be a PRIMARY KEY
- N can be a periodically re-generated constant
Option #2: Reduce size of dataset in ORDER / LIMIT

```
SELECT id FROM ... WHERE ... ORDER ... LIMIT
SELECT <other_fields> FROM ...JOIN ... WHERE id IN (id_list)
```

- Again, `id` assumed to be a PRIMARY KEY
- The ORDER / LIMIT step often needs a temp table
- First SELECT retrieving only ids keeps temp table small
- Allows 'Using index', especially with InnoDB
- Second SELECT is a safe fixed size range access
Caching

Performance 101 for Small Web Apps

What to use for caching?

- Turn on MySQL Query Cache
- Memcached is easy to add if you have your own server
- For shared hosts, consider caching to MEMORY tables

What to cache?

- For pagination, the page count and the id=N
- Candidate: Data not specific to any user and not required to update in real time (static, or mostly static)
- Candidate: User specific data with a lifetime longer than one page request
Lie, Cheat, Steal
Performance 101 for Small Web Apps

Website counters!

N people viewing this page
User last seen on <date>

- The naive implementation uses a lot of query traffic
- Updates to track when folk last visited, on every page hit
- Selects to calculate the counter value, on every page hit

So why not lie about it...who really cares?

- Calculate counters once every few minutes, and cache
- Slightly randomize the result for each display
- Be vague: hundreds, thousands, today, last week ...
Query concurrency = How well queries work together.

- **SHOW PROCESSLIST** is a snapshot of MySQL activity
- **Shared hosts**: You can only see your own queries
  - So if someone else is slow, no real way to tell, or fix. Move to another server.
  - If you can see other folks' queries, move to another server too!
- **MySQL** is multi-threaded, running one thread per client connection (and hence per query/transaction)
  - For SHOW PROCESSLIST, a "process" is really a "thread"
- **MyISAM** uses table level locking - use InnoDB!

<table>
<thead>
<tr>
<th></th>
<th>Locked</th>
<th>Table lock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waiting for</td>
<td>Waiting for table</td>
<td>Copying to tmp table on disk</td>
</tr>
<tr>
<td>table</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>GROUP_CONCAT()</td>
<td>Concatenate grouped values as a string (reduces query counts!)</td>
<td></td>
</tr>
<tr>
<td>COALESCE()</td>
<td>Return first non-NULL argument (simpler and faster than IF/IFNULL constructs)</td>
<td></td>
</tr>
<tr>
<td>LEAST()</td>
<td>Return minimum and maximum arguments (MIN / MAX at the record level)</td>
<td></td>
</tr>
<tr>
<td>GREATEST()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEFAULT()</td>
<td>Return a column's default value (not precisely performance related, but...)</td>
<td></td>
</tr>
<tr>
<td>GET_LOCK()</td>
<td>Get and release named co-operative locks (useful for locking between background cron jobs)</td>
<td></td>
</tr>
<tr>
<td>RELEASE_LOCK()</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Thanks for attending!

Questions?

http://www.skysql.com/en/services/training

Administering MySQL
Performance Tuning for MySQL