Avoiding Common (but Deadly) MySQL Operations Mistakes

Bill Karwin

bill.karwin@percona.com

MySQL Operations Mistakes

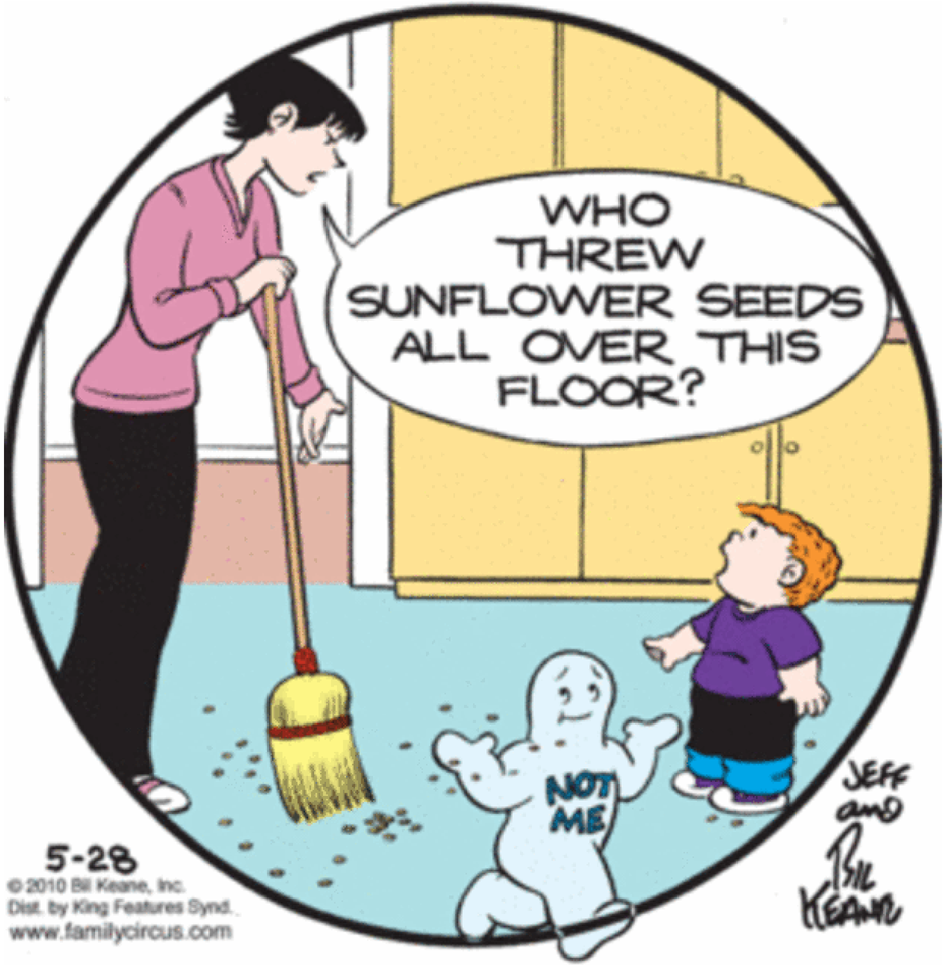
MYSTERY CONFIGURATION

Who Changed the Config?

- Database server restarted, but MySQL didn't start
- In the config file /etc/my.cnf, the log file size was commented out:

```
 [mysqld]
# innodb_log_file_size = 128M
```

- The default (5MB) didn't match the log file, so mysqld refused to start.
- No one remembered who had commented out the entry, or why, or even when.


WHO THREW SUNFLOWER SEEDS ALL OVER THIS FLOOR?
NOT ME
5-28
© 2010 Bil Keane, Inc.
Dist. by King Features Synd.
www.familycircus.com
JEFF and BIL KEANE

“Not me!”

Tracking It Down

- Keep `/etc/my.cnf` under source control
  - Who changed the file
  - When they made the change
  - Comment on the rationale
  - Include an issue tracker number
- You can find out answers to the above:

```
$ git blame my.cnf
```

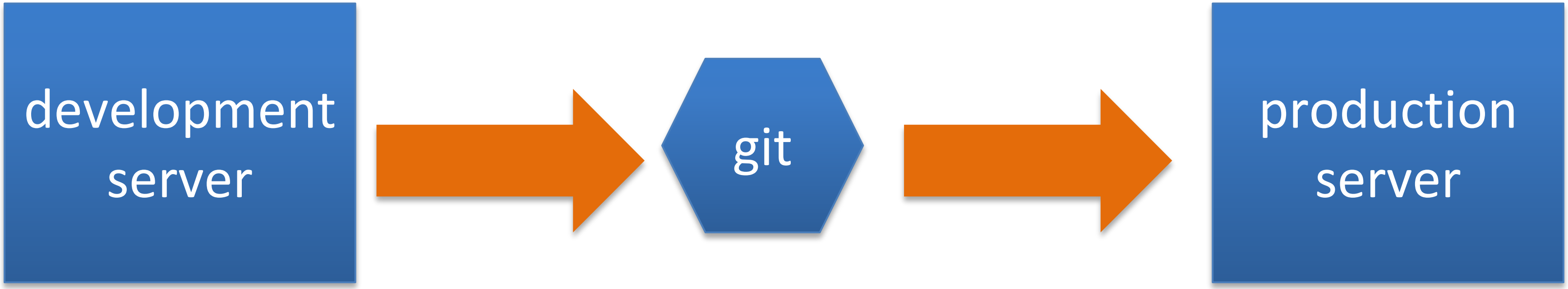
More Readable

- Your config file may also contain comments:

```
[mysqld]
# Test if mysqld auto-resizes the log
# (mike, 2013-12-31)
# innodb_log_file_size = 128M
```

Deployment Process

- Check into source control, and then deploy directly from source control.



- This is best practice for application deployment.
- Good even in IT – “we need to be more agile” doesn’t fly when you have a larger company.

MySQL Operations Mistakes

ABANDONED EXPERIMENTS

Change All the Configs!

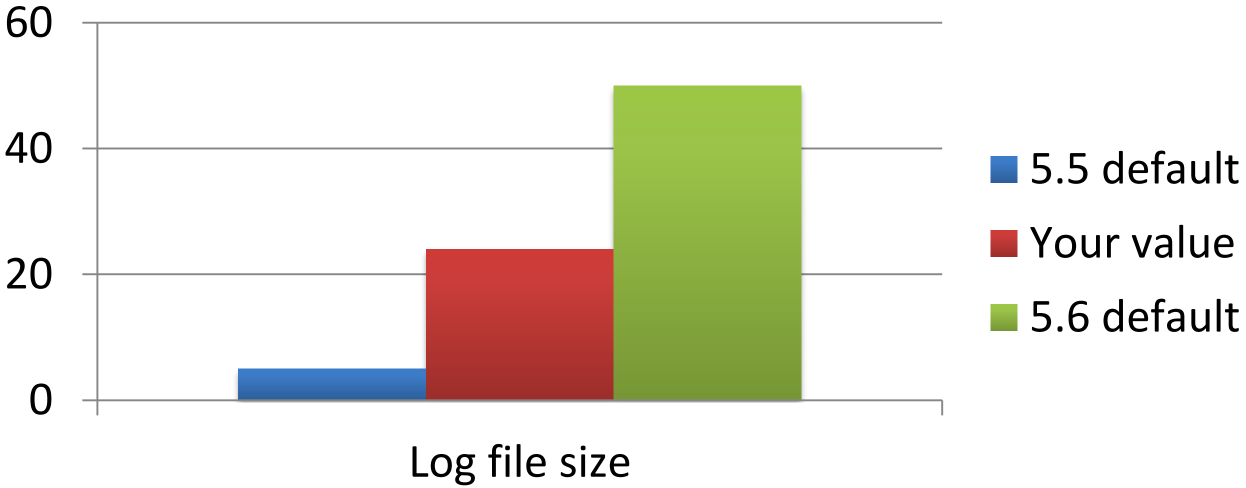
- Copy `my-innodb-heavy.cnf`
  - Keep every comment – 479 lines worth.
- If some buffers are good, more must be better!
- Don’t leave any variables at default values.
- Change early, change often.

Abandoned Experiments

- Change values willy-nilly, to see what will happen.
- Leave experimental changes in production systems.
- Keep no records of who made the change, what was the reason, or whether it made a difference.

Defaults

- New MySQL versions have new defaults.
- Overriden config values hide new defaults.

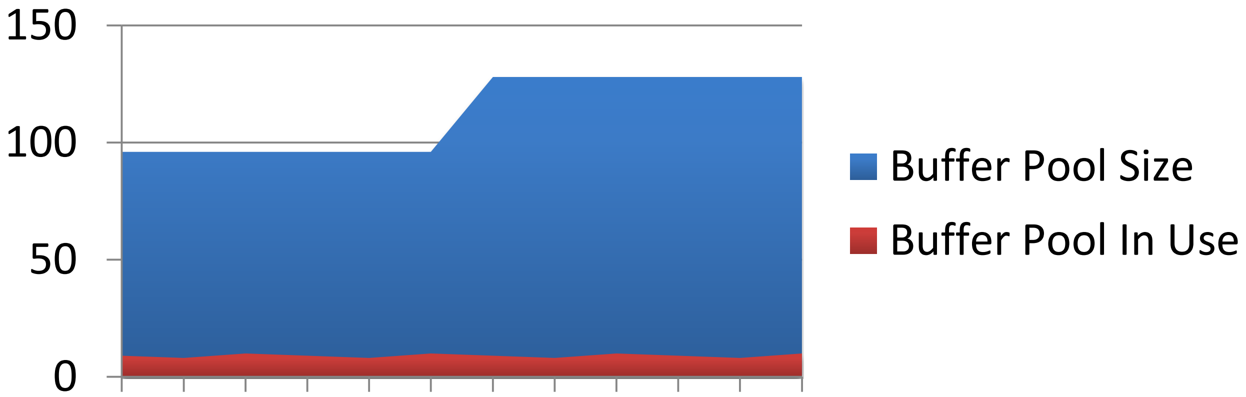


Leave It Alone

- Defaults are often well-chosen and perfectly good for most workloads.
  - Leave config values at their default
- Experiment, but do it methodically.
  - That means make results *testable*.
  - Show the difference in a staging environment.

Not Everything Needs Tuning

- A change might have no effect on a system where there is no bottleneck.



  - e.g. Increasing the buffer pool doesn’t help if your database is small and only using 3% of current BP.

MySQL Operations Mistakes

AIMLESS TUNING

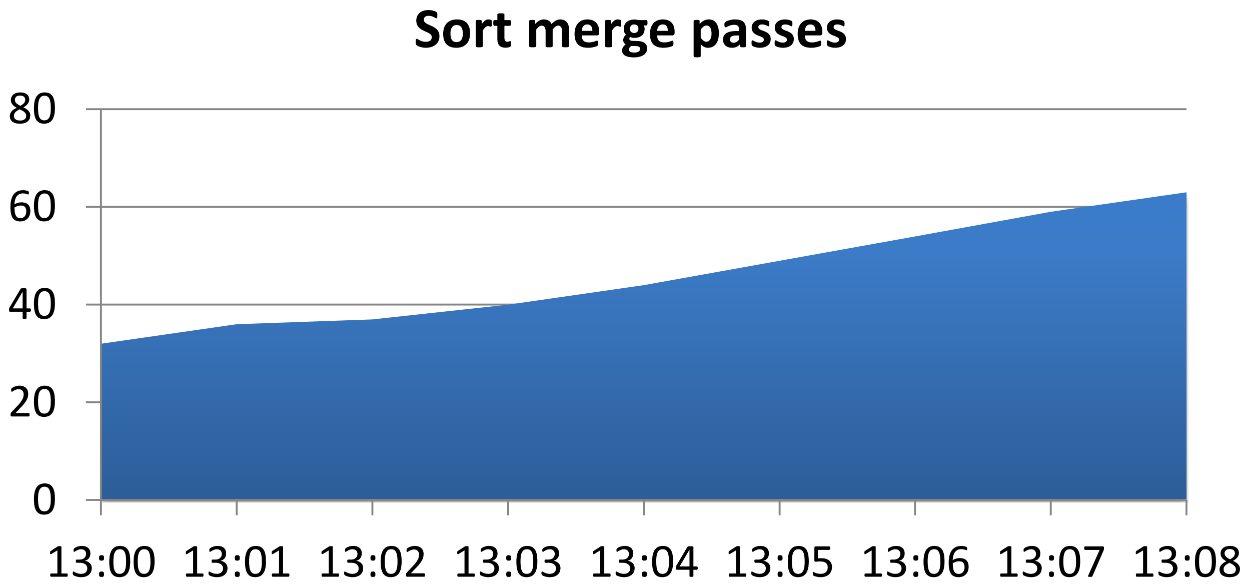
“Measure twice, cut once.”

Step 1

- Choose a measurable indicator of performance
  - e.g. `sort_buffer_size` effectiveness is measured by `sort_merge_passes`
- Measure the impact of performance before changing the configuration parameter.

Step 2

- Measure the rate of increase:



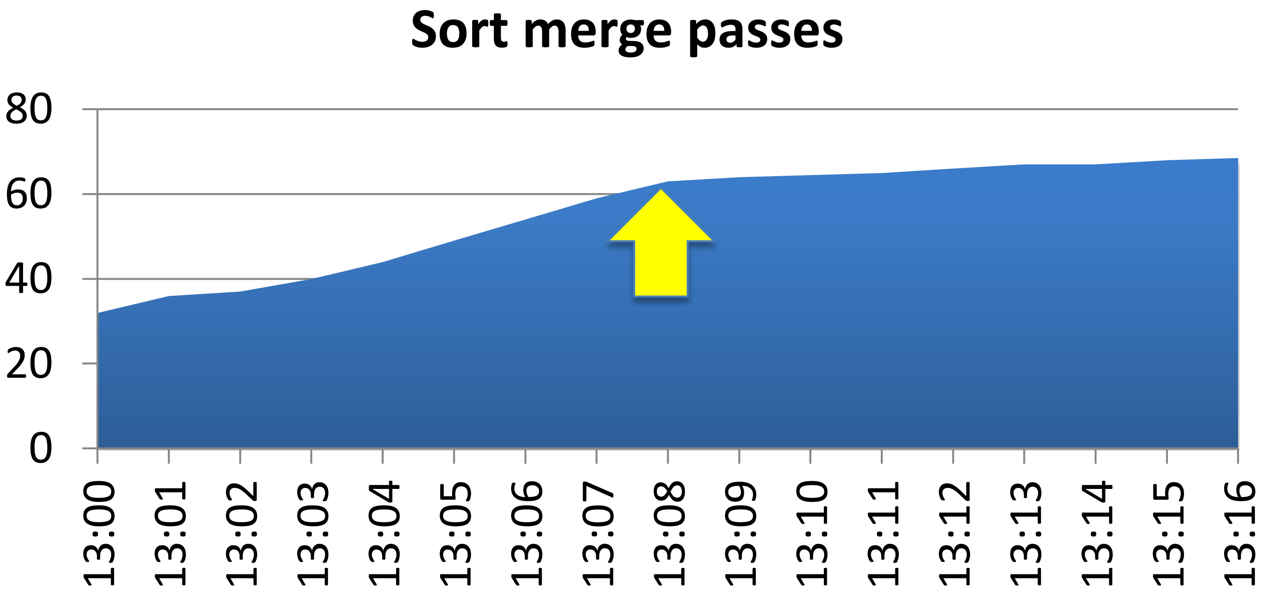
Step 3

- Research the range of reasonable values for the corresponding configuration variable.
- Make a *modest* change, for example, this variable was 256KB by default, let's raise it to 384KB.

```
 mysql> SET GLOBAL sort_buffer_size = 393216;
```

Step 4

- Re-measure the rate of increase to verify impact:



Tools

- Use `pt-mext` for ad hoc measurements in a command window.
  - http://www.percona.com/doc/percona-toolkit/pt-mext.html
- Use Percona Monitoring Plugins to produces trending graphs in Cacti or Zabbix.
  - http://www.percona.com/software/percona-monitoring-plugins

Don't Overallocate

- Some buffers are allocated globally:
  - innodb_buffer_pool_size
  - innodb_log_buffer_size
  - max_heap_table_size*
  - query_cache_size

- Some are allocated per SQL thread:
  - binlog_cache_size
  - innodb_sort_buffer_size
  - join_buffer_size*
  - read_buffer_size
  - read_rnd_buffer_size
  - thread_stack
  - tmp_table_size*

MySQL Operations Mistakes

BRITTLE BACKUPS

Brittle Backups

- Site had an emergency crash, and needed to restore from backup.
- They discovered the last six months of backups had been sent to a 100% full filesystem – none of the backups were restorable.
- It's hard for a business to recover from this kind of mistake – you may be turning off the lights

False Positives

- Reviewing a shell script to automate backups…
  - Back up database – OK.
  - Copy external assets – OK.
  - Email IT admin to notify of backup success – OK.
- The problem: the email reported success without checking if any of the commands in the script were successful.

Trust But Verify

- Test that your backups are restorable!
- Restore the backup to a staging/test server.
  - If you don’t have one, use virtual machines, or MySQL Sandbox.
- Implement error detection and error reporting into your backup scripts.

How to Verify?

- Restore completes with success.
- Check that all databases & tables are present.

```
mysqldump --no-data | diff - baseline.sql
```

- Run a few queries as a "smoke test."
  - E.g. a few representative queries from your app.
- Replay a sample of binary logs, collected from the production server right after the backup.
- `CHECKSUM TABLE`

Other Benefits of Testing

- Practice improves familiarity with recovery process.
- You have automation scripts for disaster recovery
- You can estimate the time to recover the current database more precisely.

Don’t think of a backup strategy – instead think of a *restore* strategy.

MySQL Operations Mistakes

DRIFT HAPPENS

READY TO ROCK

Replication Drift Happens

- Data on a replication slave may not be true.
  - Non-deterministic queries update the slave.
  - Someone may change data on the slave.
  - Data discrepancies tend to compound.
- MySQL has no built-in checks.

Impact of Data Drift

- Replication slave returns wrong query results.
- Replication slave is not a valid source of backups.
- Replication slave can’t be used as a failover server.

Solutions to Prevent Data Drift

- Block users from making illicit changes:

```
mysql> SET GLOBAL read_only=1;
```

- Row-based replication reduces the risk of non-deterministic queries:

```
[mysqld]
binlog-format = ROW
```

Solutions to Detect Data Drift

- Use `pt-table-checksum` to test integrity.
  - http://www.percona.com/doc/percona-toolkit/pt-table-checksum.html
- Automate this as a scheduled job via cron.
  - Recommend weekly – at the time of least load.
- Review the results ASAP.
  - Use an alerting tool to report discrepancies.
  - Percona Monitoring Plugins does this!

Solutions to Correct Data Drift

- Use `pt-table-sync` to re-insert data in chunks that contain discrepancies.
  - This is a no-op on the master.
  - The data is restored on the slaves.
  - http://www.percona.com/doc/percona-toolkit/pt-table-sync.html

MySQL Operations Mistakes

INDEX HOARDING

©danielathomas

Too Many Indexes

- Consumes space on disk:

```
mysql> show table status like 'title'\G
           Name: title
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 1565543
 Avg_row_length: 67
    Data_length: 105512960
Max_data_length: 0
   Index_length: 85164032
```

Too Many Indexes

- Consumes space in the buffer pool:

```
mysql> SELECT table_name, index_name, COUNT(*)
FROM INFORMATION_SCHEMA.innodb_buffer_page GROUP BY 1,2;
+-----------------+------------+----------+
| table_name      | index_name | count(*) |
+-----------------+------------+----------+
| NULL            | NULL       |     2202 |
| `imdb`.`title`  | PRIMARY    |     3412 |
| `imdb`.`title`  | ti         |     2559 |
| `imdb`.`title`  | title      |       14 |
| `SYS_FOREIGN`   | FOR_IND    |        1 |
| `SYS_FOREIGN`   | REF_IND    |        1 |
| `SYS_INDEXES`   | CLUST_IND  |        1 |
| `SYS_TABLES`    | CLUST_IND  |        1 |
+-----------------+------------+----------+
```

Too Many Indexes

- Makes more work for the query optimizer:

```
mysql> EXPLAIN SELECT * FROM imdb.title
WHERE title = 'Goldfinger'\G
           id: 1
  select_type: SIMPLE
        table: title
         type: ref
possible_keys: title,ti
          key: title
      key_len: 152
          ref: const
         rows: 4
        Extra: Using where
```

Duplicate Indexes

- MySQL allows more than one index covering the same columns.

```
mysql> CREATE TABLE Foo ( x INT, y INT );
mysql> ALTER TABLE Foo ADD INDEX (x);
mysql> ALTER TABLE Foo ADD INDEX (x, y);
```

makes index on *x* superfluous

- Virtually every database has some.
  - Example: 400GB of duplicate indexes in a 2TB database, i.e. 20% of total size.

Detecting Duplicate Indexes

- MySQL 5.6 generates a “Note” warning for an exact duplicate index.

```
mysql> ALTER TABLE Foo ADD INDEX (x);
mysql> show warnings\G
 Level: Note
   Code: 1831
Message: Duplicate index 'x_3' defined on the table
'test.Foo'. This is deprecated and will be disallowed
in a future release.
```

Detecting Duplicate Indexes

- MySQL 5.7 can be more strict.
  - Generates a “Warning” instead of a “Note.”
  - In strict mode, the warning becomes an error, and the index creation fails.

Detecting Duplicate Indexes

- Use pt-duplicate-key-checker to report:

```
# ####################################################################
# test.Foo
# ####################################################################

# x is a left-prefix of x_2
# Key definitions:
#   KEY `x` (`x`),
#   KEY `x_2` (`x`,`y`),
# Column types:
#      `x` int(11) default null
#      `y` int(11) default null
# To remove this duplicate index, execute:
ALTER TABLE `test`.`Foo` DROP INDEX `x`;
```

Unused Indexes

- Consumes disk space.
- Query optimizer considers all relevant indexes – during *every* query.

Detecting Unused Indexes

- Use `pt-index-usage` with your query log.
  - http://www.percona.com/doc/percona-toolkit/pt-index-usage.html

Detecting Unused Indexes

- Use PERFORMANCE_SCHEMA
  - Doesn't need logs, but costs overhead to enable the global statistics consumer.

```
SELECT OBJECT_SCHEMA, OBJECT_NAME, INDEX_NAME, COUNT_STAR
FROM performance_schema.table_io_waits_summary_by_index_usage
WHERE COUNT_STAR = 0 AND INDEX_NAME != 'PRIMARY'
AND OBJECT_SCHEMA NOT IN ('mysql', 'performance_schema');
```

Detecting Unused Indexes

- Use Percona Server user stats:
  - Negligible overhead.

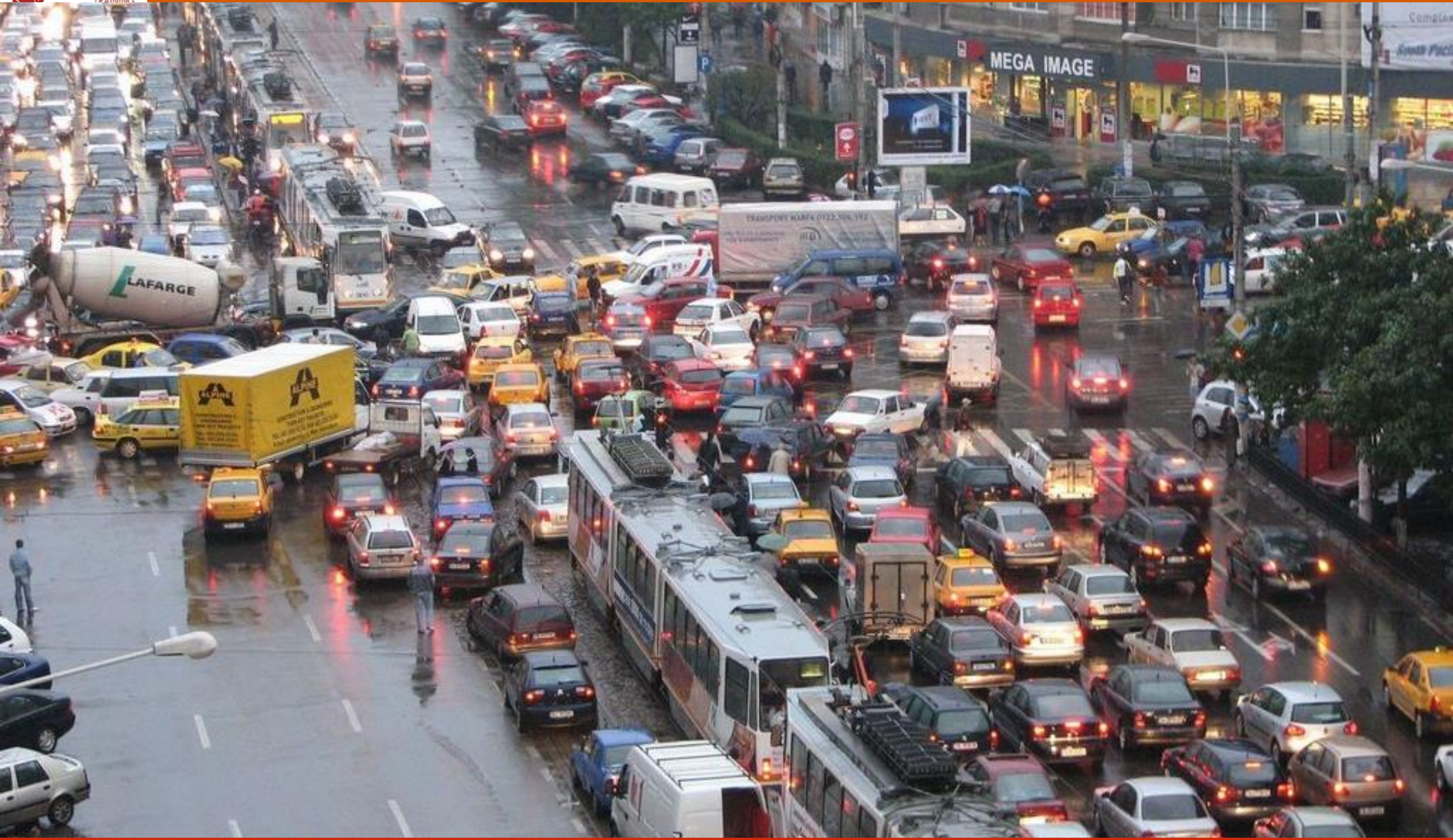
```
SELECT S.TABLE_SCHEMA, S.TABLE_NAME, S.INDEX_NAME
FROM INFORMATION_SCHEMA.STATISTICS S
LEFT OUTER JOIN INFORMATION_SCHEMA.INDEX_STATISTICS I
USING (TABLE_SCHEMA, TABLE_NAME, INDEX_NAME)
WHERE I.INDEX_NAME IS NULL
AND S.INDEX_NAME != 'PRIMARY'
AND S.TABLE_SCHEMA NOT IN ('mysql',
'performance_schema');
```

Clean Up Regularly

- Check for duplicates periodically:
  - After changes to the schema.
- Check for unused indexes periodically:
  - After changes to the schema.
  - After changes to application queries.
  - After changes to application traffic.
  - After changes to data.

MySQL Operations Mistakes

NO CAPACITY MONITORING


MEGA IMAGE
LAFARGE

Types of Bottlenecks

Disk-Bound I/O queueing disk space exhaustion	Network-Bound bandwidth exhaustion
Memory-Bound swapping to virtual memory	CPU-Bound multi-threaded contention

Disk Bottlenecks

- Greatest single causes of downtime:
  - Disk full
  - SAN failure
  - RAID failure

Measure Capacity

- Disk size capacity is easy (`df`).
- Run benchmarks on your storage hardware to estimate your capacity.
  - Use sysbench for system I/O benchmark, MySQL read/write benchmark, etc.
  - https://launchpad.net/sysbench

Measure Usage

- Monitor usage against your capacity.
  - Use `df` (again) for disk usage.
  - Use `iostat` for ad hoc monitoring, detecting queueing.
  - Use Percona Monitoring Plugins for monitoring resource use in Cacti or Zabbix.
  - Use alerting tools like Nagios (also PMP).

Test Disk Health

- Use `smartctl` for testing S.M.A.R.T. capable disks
- Use RAID controller utilities such as `arcconf` or `MegaCli64` to test health of volumes and disks in RAID arrays.

MySQL Operations Mistakes

SERVICE INTERRUPTIONS

Service Interruptions

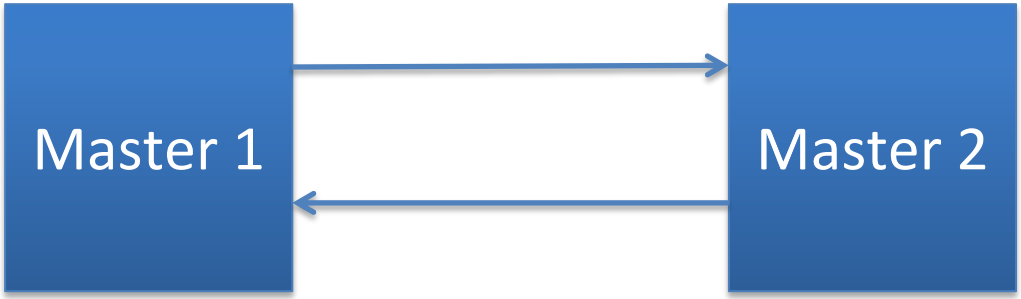
- `ALTER TABLE` interrupts traffic.
- Upgrading MySQL also interrupts traffic.
- Server failover can take too long.

Solutions for ALTER TABLE

- MySQL 5.5 has fast index creation/drop.
- MySQL 5.6 introduces many new cases for online `ALTER TABLE`.
  - http://dev.mysql.com/doc/refman/5.6/en/innodb-online-ddl.html
- Use pt-online-schema-change for other cases.
  - http://www.percona.com/doc/percona-toolkit/pt-online-schema-change.html

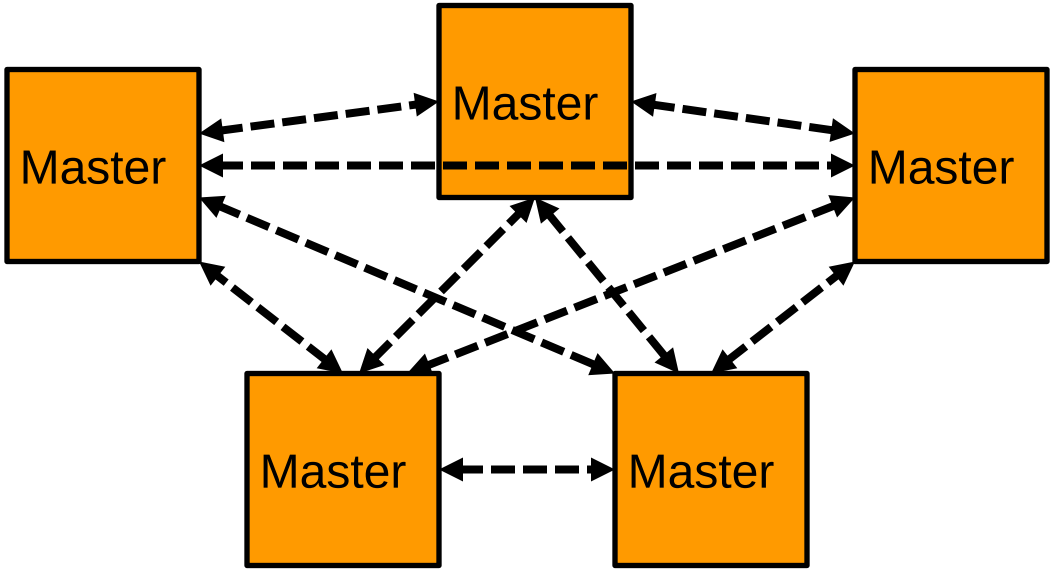
Solutions for Upgrades

- Use a pair of MySQL servers, replicating both ways.
- One is always the primary, while the secondary can be taken offline for upgrades.



Solutions for Failover

- Percona XtraDB Cluster—every node is writable.



http://www.mysqlperformanceblog.com/2012/12/04/a-closer-look-at-percona-xtradb-cluster-for-mysql/

PERCONA LIVE: MYSQL CONFERENCE AND EXPO 2014

Learn from leading MySQL experts. Santa Clara, CA. April 1 - 4, 2014.

Early Bird Rates End February 5th, 2014

http://www.percona.com/live/mysql-conference-2014/

PERCONA TRAINING

Senior Industry Experts

In-Person and Online Classes

Custom Onsite Training

http://percona.com/training

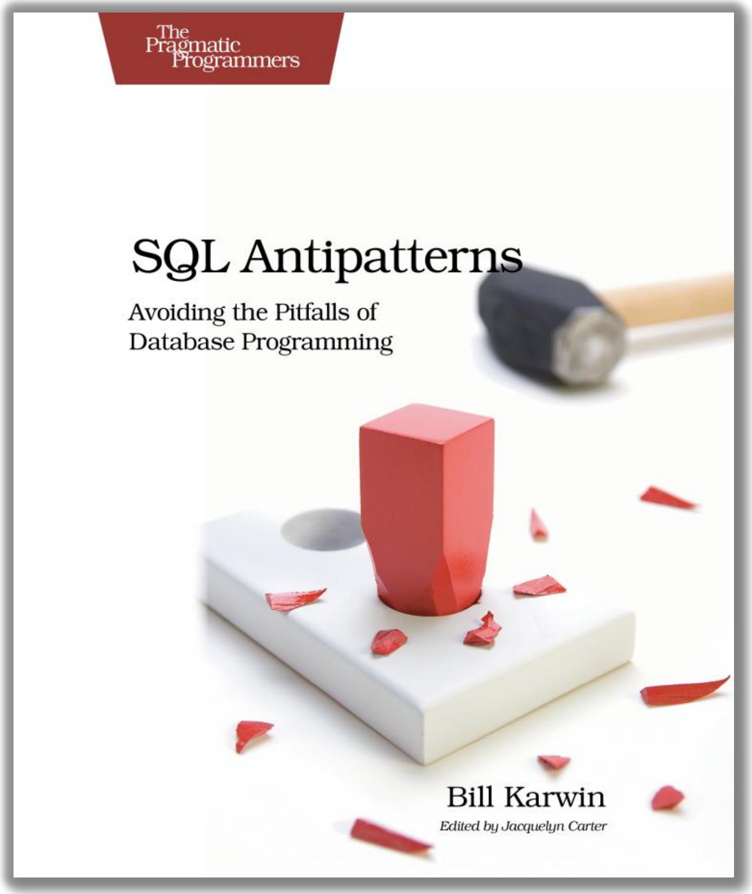
SQL Antipatterns:

Avoiding the Pitfalls of Database Programming

by Bill Karwin

Available in print, epub, mobi, pdf.
Delivery options for Kindle or Dropbox.

http://pragprog.com/book/bksqla/


The Pragmatic Programmers
SQL Antipatterns
Avoiding the Pitfalls of Database Programming
Bill Karwin
Edited by Jacquelyn Carter

License and Copyright

Copyright 2014 Bill Karwin

http://www.slideshare.net/billkarwin

Released under a Creative Commons 3.0 License:
http://creativecommons.org/licenses/by-nc-nd/3.0/

You are free to share—to copy, distribute and transmit this work, under the following conditions:

cc BY NC ND

Attribution.
You must attribute this work to Bill Karwin.

Noncommercial.
You may not use this work for commercial purposes.

No Derivative Works.
You may not alter, transform, or build upon this work.