The DB is Broken. Now what?
A global IT company that helps businesses leverage disruptive technologies to better compete.

Our services and software solutions unleash the power of cloud, data and analytics to drive better business outcomes for our clients.

Our 20 years in data, commitment to hiring the best talent, and our deep technical and business expertise allow us to meet our promise of using technology to deliver the best outcomes faster.
20 Years in Business

400+ Python Experts in 35 Countries

350+ Current Clients Globally
<table>
<thead>
<tr>
<th>Media/Information Services</th>
<th>Retail</th>
<th>E-commerce</th>
<th>SaaS</th>
<th>Financial Services</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FOX SPORTS</strong></td>
<td><strong>SONOS</strong></td>
<td><strong>Tech Data</strong></td>
<td><strong>Grasshopper</strong></td>
<td><strong>PIRAEUS BANK</strong></td>
</tr>
<tr>
<td><strong>NATIONAL GEOGRAPHIC</strong></td>
<td><strong>URBAN OUTFITTERS</strong></td>
<td><strong>freshdirect</strong></td>
<td></td>
<td><strong>COMMERZBANK</strong></td>
</tr>
<tr>
<td><strong>AEG</strong></td>
<td><strong>ARCTERYX</strong></td>
<td><strong>Camelot</strong></td>
<td><strong>Entrust Datacard</strong></td>
<td></td>
</tr>
<tr>
<td><strong>BBM</strong></td>
<td><strong>ALSAINTS</strong></td>
<td><strong>rockyou</strong></td>
<td></td>
<td><strong>Paymentus</strong></td>
</tr>
<tr>
<td><strong>medianet</strong></td>
<td><strong>FAST RETAILING</strong></td>
<td></td>
<td></td>
<td><strong>TaxAudit</strong></td>
</tr>
</tbody>
</table>

© Allan Mason, Carmen Mason, Ivan Groenewold 2019
THE WORLD OF LEARNING HAS CHANGED
VITALSOURCE CAN HELP

Student Affordability
Helping students get their critical content

Digital Distribution
Simplified, integrated access to learning materials

Personalized Learning
Using artificial intelligence to fuel learning impact
On our platforms in over 200 countries and territories—in higher education, adult learning, and K-12.

E-textbooks and courseware, all major publishers, all major markets; various durations, models and prices—including Inclusive Access, Equitable access, purchase, rental, others.

10+ MILLION STUDENTS

4500+ INTEGRATIONS
LMS / VLE, e-commerce, library portals, teaching portals, and other learning access points; global

20+ MILLION NEW UNITS
What we're going to talk about

Our talk is split into two main pieces.

- Tools and ideas to help diagnose many DB problems.
  - You should review how to use them.
- Our experience with a handful of issues. We'll get as deep as time allows with these.

- Not possible, in one talk, to cover everything.
- If you need more help, we have some resources at the end.
noun. [dey-tuh-beys, ad-min-uh-strey-ter]
Someone who does precision guesswork based on unreliable data...
Setup Your "Toolbox" before you need it

Quick Tips

1. Terminal Scrollback
2. Cheatsheet - Personal quick list of commands you already know.
3. Runbook/Wiki - Detailed how-to's, etc..
4. Monitoring and Alerting - Installed and setup
5. Ticketing System, like JIRA, or another place to store details
6. Consider installing additional tools like: percona-toolkit, htop, mytop
   - Can make life much easier when they are needed.
7. Learn how to use the tools we'll cover, and understand what the output means.
Diagnostics - Capture Initial State! Long Version

Quickly capture information (save it somewhere: JIRA, Atom, Evernote...)

- uptime
- mysql> \s
  - Quick status of MySQL.
- mysql> pager grep -v Sleep; SHOW PROCESSLIST;
- date ; mysql -e "SHOW ENGINE INNODB STATUS\G" | tee
  - Especially the transaction section
- free -m, df -h
  - memory/swapping/disk issues?
- top, vmstat, iostat
Diagnostics - Capture Initial State! Long Version, continued

If the previous slide didn't provide the complete smoking gun, start reviewing the output. Even if you think you know what's going on, seriously consider these commands as well.

- uptime
- dmesg | tail
- vmstat 1
- mpstat -P ALL 1
- pidstat 1
- iostat -xz 1
- free -m
- sar -n DEV 1
- sar -n TCP,ETCP 1
- top

**Diagnostics - If you want to go the quick route**

- Review monitoring tools.
- Quickly capture information (save it somewhere: JIRA, Atom, Evernote...)
  - Take the time to do this. It can be very useful.
- **pt-stalk --no-stalk --iterations 1**
  - Dumps the output to handful of commands and logs for you to review.
  - "--dest" to chose the output location. By default, this is `/var/lib/pt-stalk`
- **pt-sift** to review the information.
- **date ; uname -n ; uptime ; who**
  - Add or remove commands here to have something more "in front of your face", that you don't need to find in a file from pt-stalk.
- **mytop**
Diagnostics - Use the Buddy System When It's Serious

- Front Person
  - Point of contact leaving you free to troubleshoot.
- Sounding Board
- Reach out to someone and have them double check your commands before you run them.
  - DBA+1
- Fellow Bus Driver for Post Mortems
Handle an issue efficiently, and quickly, without making it worse.
FIT-ACER

- Focus
- Identify the Server / Database name, and time
- Type the Command (do not hit enter yet!)
- Assess the command/script/action for issues
- Check the Server / Database name again
- Execute the command/script/action
- Review and document the Results.
Slow Down!

- Uptime Institute: 70% of DC Outages Due to Human Error.
- 2019: Facebook - Facebook, Instagram, WhatsApp
  - 24 hours til all clear.
  - "Yesterday, as a result of a server configuration change, many people had trouble accessing our apps and services."
    - https://twitter.com/facebook/status/1106229690069442560?s=20
- 2017: Amazon - S3 service outage for us-east-1
  - Over 4 hours.
  - "Unfortunately, one of the inputs to the command was entered incorrectly and a larger set of servers was removed than intended."
    - https://aws.amazon.com/message/41926/
Random oddities
Storage in the cloud

EBS/GCP persistent disks
Where are my IOPS?

IOPS depend on

- Volume size
  - gp2 volumes < 1 TB can burst to higher IOPS (Amazon)

- Instance type
  - vCPU count
  - IO bandwidth cap
  - Network egress cap (Google Cloud)
    - https://cloud.google.com/compute/docs/disks/performance
Where are my IOPS?

Look at metrics

- iostat, pt-diskstats

<table>
<thead>
<tr>
<th>rd_s</th>
<th>rd_avkb</th>
<th>rd_mb_s</th>
<th>rd_mrg</th>
<th>rd_cnc</th>
<th>rd_rt</th>
<th>wr_s</th>
<th>wr_avkb</th>
<th>wr_mb_s</th>
<th>wr_mrg</th>
<th>wr_cnc</th>
<th>wr_rt</th>
<th>busy</th>
<th>in_pr</th>
<th>io_s</th>
<th>qtime</th>
<th>stime</th>
</tr>
</thead>
<tbody>
<tr>
<td>1875.1</td>
<td>19.9</td>
<td>36.4</td>
<td>0%</td>
<td>2.2</td>
<td>1.2</td>
<td>821.8</td>
<td>19.1</td>
<td>15.3</td>
<td>56%</td>
<td>1.2</td>
<td>0.6</td>
<td>94%</td>
<td>7</td>
<td>2696.8</td>
<td>0.7</td>
<td>0.3</td>
</tr>
</tbody>
</table>

What about latency?

- Cloud vendors doc mention single-digit ms latency but no guarantees
- Local SSD have lower latency but ephemeral data
EBS recommendations

- Provisioned IOPS (io1) volumes are expensive
- gp2 volumes can now scale up to 16000 IOPS @ 5 TB size
  - Upgrade older volumes with \textit{ModifyVolume} operation
- Use EBS-optimized instances
- “r” instance type has good cost/benefit
I still don’t get my IOPS!

- Kernel with indirect file descriptors
  - 3.8+ kernels
- Spread interrupt handling across CPU cores
  - Use irqbalance 1.0.6+
  - Enable receive packet steering (RPS)
- Look at block size of your IOs
  - docs are based on 16 K I/O size
Ok got my IOPS now, but...

- `innodb_io_capacity`
  - It does not limit all MySQL IOPS
  - Only affects background flushing activities
  - Rarely need to go higher than 2000 or so
  - Look at SHOW ENGINE INNODB STATUS sections
    - INSERT BUFFER AND ADAPTIVE HASH INDEX
    - INDIVIDUAL BUFFER POOL INFO

- Set `innodb_flush_neighbours=0` for SSD
I restored an EBS snapshot and now things are slow again!

- Blocks are lazy-loaded from S3
- Pre-warm required for good performance
  - `dd`, `fio`
- Can take a long time!
- Plan recovery strategy

SWAPPING

even with swappiness=0!
I see swapping but there shouldn’t be any!

Symptoms:
- Enough “freeable” memory
- Swappiness=1 or 0
- System is still swapping

- The swap insanity problem
  - Term coined by Jeremy Cole
  
The swap insanity problem

Uniform Memory Architecture
- Single processor - multiple cores
- All memory is equal

Non-Uniform Memory Architecture (NUMA)
- Multi-CPU systems
- Each processor has “local” memory
  - Latency differences
The swap insanity problem

```
$ numactl --hardware
available: 2 nodes (0-1)
node 0 cpus: 0 1 2 3 4 5 6 7 16 17 18 19 20 21 22 23
node 0 size: 124999 MB
node 0 free: 3707 MB
node 1 cpus: 8 9 10 11 12 13 14 15 24 25 26 27 28 29 30 31
node 1 size: 125000 MB
node 1 free: 54 MB
node distances:
node 0 1
  0: 10 20
  1: 20 10
```
The swap insanity problem

NUMA policies

- Inherited by parent process
- localalloc (default)
- preferred=node
- membind=node(s)
- interleaved=[nodes|all]

Once memory is allocated it will not be moved to other nodes!
The swap insanity problem

- localalloc policy not ideal for MySQL
- Buffer pool allocation uses all of node0 memory
- If another process on node0 needs memory -> swap

MySQL memory (mostly Buffer pool)
The swap insanity problem

- Set `innodb_numa_interleave` option
- What if OS already allocated memory to buffer/caches?
- Flush buffers/caches before starting MySQL
  - Percona Server adds `flush_caches` option
  - add to `[mysqld_safe]` section of my.cnf

https://www.percona.com/doc/percona-server/5.7/performance/innodb_numa_support.html
Transparent Huge Pages

THP good or bad?
I get random stalls!

- Database seems to be well tuned...
- Check `top` for:
  - Significant sys% usage
  - CPU spikes
  - `khugepaged`, `kswapd`, `kcompactd` activity

- Look at transparent huge pages (THP)
Transparent hugepages

- Increase the page size from the standard 4 KB
- Enabled by default (2.6.38+)
- Applications not designed for large pages can take advantage
- High number of pages -> fewer are maintained in TLB
- THP can be swapped out (need to be split)
Transparent hugepages

- Can be problematic for high-concurrency MySQL use case
  - Think many threads doing memory-intensive operations on “small” chunks
- THP improved in > 4.14 kernels
- Test in your environment
  ```bash
  cat /sys/kernel/mm/transparent_hugepage/enabled
  [always] madvise never
  ```

https://www.percona.com/blog/2019/03/06/settling-the-myth-of-transparent-hugepages-for-databases/
TASKSMAX

Why??
Starting Misleading Information

LEAD DEV:
"We're hitting Max Connections!"

SENTRY:
ActiveRecord::ConnectionTimeoutError
could not obtain a connection from the pool within 5.000 seconds (waited 5.005 seconds); all pooled connections were in use.

```
mysql> SELECT @@GLOBAL.max_connections;
+--------------------------+
| @@GLOBAL.max_connections |
+--------------------------+
|                     7500 |
+--------------------------+
1 row in set (0.00 sec)
```
First Hunch is Usually Correct… Usually

MySQL ERROR LOG:
[Note] Aborted connection … to db … (Got an error reading communication packets)
[Note] Got an error reading communication packets
[Note] Bad handshake
[ERROR] Error log throttle: 7 'Can't create thread to handle new connection'
error(s) suppressed
[ERROR] Can't create thread to handle new connection(errno= 11)

cat /proc/14953/limits | egrep "(processes|files)"
Max processes 418944 418944 processes
Max open files 1048576 1048576 files
systemctl status mysql

mysql.service - Percona Server
  Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
  Active: active (running) since Wed 2019-03-20 14:47:08 UTC; 1 months 11 days ago
  Main PID: 14953 (mysqld)
    Tasks: 4898 **(limit: 4915)**
    CGroup: /system.slice/mysql.service
      └─14953 /usr/sbin/mysqld --daemonize --pid-file=/var/run/mysqld/mysqld.pid
The Fix

**DMESG** error: 'cgroup: fork rejected by pids controller in /system.slice/mysql.service'

Create the Override
Add a `~/etc/systemd/system/mysql.service.d/` directory with an `override.conf` file setting TasksMax to the same as the ulimit.

```bash
cat ~/etc/systemd/system/mysql.service.d/override.conf
[Service]
TasksMax=418944
```

Confirm the Change
```bash
systemctl show mysql | grep -i task
TasksCurrent=219
TasksAccounting=yes
TasksMax=418944
```
What is TasksMax??

Simplified: The number of tasks that can be created within a unit (process).

```
systemd --version
systemd 237
```

```
systemctl show -p TasksMax mysql.service
- Ubuntu 16.04 xenial w/ 228: 512!!
- Ubuntu 18.04 bionic w/ 237: 4915 This is what we are running.
```
Too Many Open Files!

MySQL ERROR LOG:
'errno: 24 - Too many open files'
How Many?

How is the effective limit determined?

1) \( 10 + \max_{\text{connections}} + (\text{table\_open\_cache} \times 2) \)
2) \( \max_{\text{connections}} \times 5 \)
3) operating system limit if positive
4) if operating system limit is Infinity:
   open_files_limit value specified at startup, 5000 if none

https://dev.mysql.com/doc/refman/5.7/en/server-system-variables.html#sysvar_open_files_limit

/etc/systemd/system.conf
DefaultLimitNOFILE=infinity
What about my system limits?

- To see OS hard limits: `ulimit -aH`
- To see OS Soft limits: `ulimit -aS`
- To see limits specific to MySQL:
  
  ```
  cat /proc/{pid}/limits | egrep "(processes|files)"
  ```

What's MySQL's PID?

- `ps -ef|grep MySQL`

  ```
  mysql 1701 1090 37 Apr03 ? 64-18:12:10 /usr/sbin/mysqld ...
  ```
Too Many Open Files!

To change it dynamically:
ulimit -n {new number}

To persist the change:
Modify /etc/security/limits.conf
  <snip>
mysql hard nofile {new number}
mysql soft nofile {new number}
Max CPU
How it Presented

- CPU is suddenly at 100% on Master
  - "all CPUs are pegged at 100% and load jump from 1 to 100 in seconds"
- No new deploys
- Recently Scaled Workers
  - Reduced back down as soon as things were hot.
- Major Production Outage
  - Grabbed a buddy!
A Billion Little Queries

```
SELECT `cached_values`.*
FROM `cached_values``
WHERE (user_guid = '?'
   AND change_id = ?)
ORDER BY `cached_values`..id``
   ASC LIMIT 1
/*application:Notes,controller:changes,action:index*/
explain extended SELECT `cached_values`.* FROM `cached_values` WHERE (user_guid = '?' AND change_id = ?) ORDER BY `cached_values`.`id` ASC LIMIT 1

/*application:Notes,controller:changes,action:index*/;

<table>
<thead>
<tr>
<th>id</th>
<th>select_type</th>
<th>table</th>
<th>type</th>
<th>possible_keys</th>
<th>key</th>
<th>key_len</th>
<th>ref</th>
<th>rows</th>
<th>filtered</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SIMPLE</td>
<td>cached_values</td>
<td>index</td>
<td>NULL</td>
<td>PRIMARY</td>
<td>8</td>
<td>NULL</td>
<td>1</td>
<td>86086704.00</td>
<td>Using where</td>
</tr>
</tbody>
</table>

- Full Index Scan
CREATE TABLE `cached_values` (
    `id` bigint(20) NOT NULL AUTO_INCREMENT,
    `user_guid` varchar(40) CHARACTER SET latin1 NOT NULL DEFAULT '',
    `change_id` bigint(20) unsigned DEFAULT NULL,
    `value_json` longtext CHARACTER SET utf8mb4,
    `created_at` datetime NOT NULL,
    `updated_at` datetime NOT NULL,
    PRIMARY KEY (`id`),
    KEY `cached_user_change_id_index` (`user_guid`,`change_id`)
) ENGINE=InnoDB AUTO_INCREMENT=15771390 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
Relief
Cause

- Programmatically created table
  - Distinct statement (Not in a transaction)
  - Create table THEN Create index
  - Any single step has potential to fail
The Solution

Short Term:
- Add the missing index
  - KEY `cached_user_change_id_index` (`user_guid`, `change_id`)

Long Term:
- Fix the code to prevent this issue.
HISTORY LIST LENGTH (HLL)
Symptoms for an HLL Issue

Quiet killer of overall DB performance.

- DB seems to be slower.
- Processlist probably shows LRQs.
- Bad to really bad query performance.
Explanation for HLL

HLL and Multi-Version Concurrency Control (MVCC), brief overview

- Consistent (ACID) version of the data, while minimizing locking issues.
- Allows each thread to see a consistent snapshot of the data inside a transaction, even as the underlying data might be changed by other threads.
- Implemented as a linked list, starting with the most recent version of rows, which points to the previous version, and so on, back to the oldest version being saved in the list.
- The entire history isn't retained forever. The oldest items are purged when they've been closed.
- Balance between # of purge threads to clean the list and getting user work done.

When there's a problem...

- Long running transactions - preventing purge threads from cleaning up.
- Keeping too much unpurged transaction history = more versions of old rows that needs to be scanned for queries = bad performance
- Longer history list = more versions of old rows that needs to be scanned for queries = bad performance
Diagnosing the Issue

- If HLL starts getting anywhere near the millions, then it needs to be fixed.
- `SHOW ENGINE INNODB STATUS\G`
  
  ---------------
  TRANSACTIONS
  ---------------
  Trx id counter 141789232186
  Purge done for trx's n:o < 141789224445 undo n:o < 0 state: running but idle
  History list length 28

- Query I use to check it directly
  
  - `SELECT name, @@hostname, NOW(), COUNT FROM information_schema.INNODB_METRICS WHERE name like '%hist%';`

<table>
<thead>
<tr>
<th>name</th>
<th>@@hostname</th>
<th>NOW()</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>trx_rseg_history_len</td>
<td>vsc-dub-dbm-p12-use1c</td>
<td>2019-09-25 19:46:54</td>
<td>35</td>
</tr>
</tbody>
</table>

1 row in set (0.01 sec)
Short Term Fix

- Look for a long running query, maybe a couple.
- Identify and kill those transactions. I would stay away from the writes if possible.
- The offending query could be sleeping, or the transaction might be hung.
  - E.G. App started the TRX and then went to do other things before finishing the TRX.
  - `mysql -BN -e 'SELECT CONCAT("kill ", id,"; --"), host, time, time_ms, state, db, left(info, 100) FROM information_schema.processlist WHERE info NOT like "Master has sent all%" AND info like "select %" AND host !="localhost" AND time_ms>10000 ORDER BY time DESC'
    
    ```
    kill 3502756; -- 127.0.0.1:40752 17 16599 Sending data NULL select t.table_schema,t.table_name,c.column_name,t.auto_increment,c.column_type from information_sch
    ```
- `pt.kill` can also be used, (think idle time, etc)
- Keep checking `trx_rseg_history_len`.
  
  ```
  SELECT name, @@hostname, NOW(), COUNT FROM information_schema.INNODB_METRICS WHERE name like '%hist%';
  ```
- After problem Trx is killed, the HLL will drop quickly, and performance will recover with it.
Long Term HLL Solutions, Reduce and Prevent the Issue

- Write 'shorter' transactions that execute quickly, and commit as soon as possible.
- Work on query performance.
  - Hides the issue, killing queries isn't ideal.
- Track and fix the issues that come up.
  - Keep track of killed queries (in JIRA, etc).
  - Consider the slow query log to grab some samples of LRQs and improve them.
- Check the Transaction Isolation Level, ensure it matches your needs.
- Move read traffic to the slaves.
STATISTICS

Alternative Facts
Symptoms

- One or more queries are suddenly very slow.
- In my experience, the issues usually are on the tables with a lot of rows.
The query optimizer used sample data, not actual data from the table.

The data in a given indexed column isn't very consistent.

On top of this, the normal processes for sampling the data, sampled pages that didn't accurately represent the overall column data.

It is just a random sample. In MySQL the sampling is based on number of pages, as opposed to a percentage of the data size. Which leads to...

As the number of rows increases, statistically, the chances of getting an inaccurate sample (sample data that doesn't reflect the overall data??) increases.
Diagnosing Bad Statistics

- Look at the queries.
- Check the EXPLAIN Plans.
  - If you have this same query / schema on a server that still runs fine, grab the Explain from there and compare to the problem DB.
  - The new, bad explain plan will be skipping the index that it used to use.
- To be sure, check 'SHOW CREATE TABLE'. Verify the indexes.
- Check Monitoring, did query execution time jump?
Resolving Statistics Issues

- Personally not a fan of inserting an Index Hint. It seems to me like it just hides the issue.
- Couple of possibilities.
  - ANALYZE TABLE (Issue with Busy DB, with contention, what version fixed??)
  - For MySQL version before the fix, run a dummy alter. It will re-run the ANALYZE under the covers.
  - Bump the number of sample pages, etc, and run ANALYZE again.
  - Might need a couple of Analyze Tables. If so, consider bumping the # of sample pages again.
- Note: When increasing the sample pages, this can be done for the whole database, or just for specific tables. I wouldn't suggest increasing the sample size significantly for the entire DB, for large jumps, I would focus on the tables in need.
Resources

● The stackoverflow family of sites.
  ● [https://stackoverflow.com/questions/tagged/mysql](https://stackoverflow.com/questions/tagged/mysql)

● Pythian's blog
  ● [https://blog.pythian.com/](https://blog.pythian.com/)

● Percona's blog
  ● [https://www.percona.com/blog/](https://www.percona.com/blog/)

● Oracle's MySQL Blog
  ● [https://blogs.oracle.com/mysql/](https://blogs.oracle.com/mysql/)

● PlanetMySQL
  ● [https://planet.mysql.com/](https://planet.mysql.com/)

● MySQL Docs!
  ● [https://dev.mysql.com/doc/](https://dev.mysql.com/doc/)

● If all else doesn't seem to have the answers
  ● MySQL Source Code
  ● MySQL Bug Reports


● Two heads are better than one. For complicated, or ongoing issues, consider Pythian for help.
What we covered

- We covered a lot. But used correctly, you should have a steady path to follow to resolve many of the issues you might encounter.
- Prepare ahead to time to help ensure you succeed, especially when time is of the essence during an issue.
- You should have a basic set of commands you always run.
- The initial information gathered, should let you know where to go next, and which commands you might want to run, or at least hints where to look for the answers.
Keep In Touch!

Carmen Mason  
carmen.mason@vitalsource.com  
http://vitalsource.com  
@CarmenMasonCIC

Ivan Groenewold  
groenewold@pythian.com  
https://pythian.com/  
@igroenew

Allan Mason  
amason@pythian.com  
https://pythian.com/  
@_digitalknight
Thank You