

MYSQL TROUBLESHOOTING

Or, what to do when MySQL starts throwing a fit

ABOUT THE CLASS

- 24 hours over three days
- Very Short Lecture and **Lots of Labs**
- Hours:
 - 8:30am - 5:00pm
 - Lunch: 11:45am - 1:00pm

ABOUT THE INSTRUCTOR

- Nathan Isburgh
 - instructor@edgecloud.com
 - Unix user 15+ years
 - Teaching 10+ years
 - MySQL user 8+ years
 - RHCE, CISSP
 - Forgetful, goofy, patient :)

ABOUT THE COLLEGE

- Rackspace Parking Sticker = good to go
- Breaks when you need them
- Breakroom downstairs - labeled “Laundry”
- Sodas - bottles in machine (\$1.25) or cans in mini-fridge (\$0.50)
- Cafeteria
- Do not speed!
- No smoking anywhere. Can only smoke sitting in car.

ABOUT THE STUDENTS

- Name?
- Time served, I mean employed, at Rackspace?
- Department?
- Unix skill level?
- MySQL skill level?
- How would you teach someone to troubleshoot?

EXPECTATIONS OF STUDENTS

- Strong foundation in basic Linux use and administration
- Ask Questions!
- Complete the labs
- Email if you're going to be late/miss class
- Have fun
- Learn something

OVERVIEW

- Troubleshooting is a thorough methodology used to track down the cause of problem.
- Keywords: **thorough** and **methodology**
- Without a thorough and exhaustive approach, the issue might be overlooked
- Without a strong and methodical approach, the issue may be misdiagnosed

TROUBLESHOOTING KEYS

- Most Important: Only change one thing at a time
- Check #1 most likely cause: You
- Check logs for error messages
- After that, check configuration and permissions
- If all else fails, slowly, piece by piece, start removing complexity from the system to narrow down the problem area.
- DOCUMENT EVERYTHING

LOGS

- One of the easiest places to find the cause of a problem is in the log files.
- Log files store informational messages from software. The types of messages include debug information, status information, warnings, errors and more.
- MySQL manages all of its logging needs. If installed from package, many distributions configure MySQL to log to:
 - `/var/log/mysqld.log`

WHEN LOGS FAIL...

- Looking through logs is all fine and dandy, but really that's a best case scenario. Your software and hardware rarely come out and announce problems and solutions in the log files. No, it's not that easy!
- More often, users will encounter symptoms of a problem, and you, as the BOFH (hopefully not yet!), will be tasked with finding and fixing the issue.

TROUBLESHOOTING TOOLS

- Troubleshooting is part science, part mystical art.
- Hopefully, through this class, you will start to develop both sides of the equation.
- For now, a discussion of several tools to help the process of troubleshooting MySQL will get you started.

DOCUMENTATION

- Documentation.
- Documentation.
- DOCUMENTATION.
- `dev.mysql.com/doc`

TOP

- `top`: Self-updating tool displays combination summary at top, followed by ordered list of processes. Fully customizable.
- The summary includes uptime information, memory breakdowns, CPU utilization and process state summaries
- The process display can be customized and sorted to suit need

```
top - 16:39:32 up 682 days, 10:41, 2 users, load average: 0.01, 0.00, 0.00
Tasks: 118 total, 1 running, 116 sleeping, 1 stopped, 0 zombie
Cpu(s): 0.1%us, 0.0%sy, 0.0%ni, 99.8%id, 0.0%wa, 0.0%hi, 0.0%si, 0.1%st
Mem: 262316k total, 258024k used, 4292k free, 7380k buffers
Swap: 524280k total, 74564k used, 449716k free, 67808k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM     TIME+  COMMAND
    1 root        15   0 10316   648  592  S   0    0.2   0:06.24  init
    2 root        RT   0     0     0    0  S   0    0.0   0:04.88  migration/0
    3 root        34  19     0     0    0  S   0    0.0   0:00.19  ksoftirqd/0
```


DF

- `df`: lists filesystem utilization
 - Breaks down size and use information for each mounted filesystem
 - `-h` is useful option to display in “human-friendly” format

```
[root@dev1 ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1       9.4G  7.2G  1.8G  81% /
none           129M     0  129M   0% /dev/shm
[root@dev1 ~]#
```


ULIMIT

- ulimit: Sets resource limits
 - Can limit open files, memory use, cpu time, subprocesses and more.

```
[root@dev1 ~]# ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
max nice                (-e) 0
file size               (blocks, -f) unlimited
pending signals         (-i) 2112
max locked memory       (kbytes, -l) 32
max memory size         (kbytes, -m) unlimited
open files              (-n) 1024
pipe size               (512 bytes, -p) 8
POSIX message queues    (bytes, -q) 819200
max rt priority         (-r) 0
stack size              (kbytes, -s) 8192
cpu time                (seconds, -t) unlimited
max user processes      (-u) 2112
virtual memory          (kbytes, -v) unlimited
file locks              (-x) unlimited
[root@dev1 ~]#
```


STRACE

- `strace`: Traces each library call a process makes
 - Extremely useful to see what a process is doing
 - Can find errors, bugs, permission issues and more
 - Let's play with tracing MySQL for a few minutes...

ERROR MESSAGES

- MySQL error messages contain useful information, which should be reviewed prior to in-depth troubleshooting:
- `ERROR 1146 (42S02): Table 'blah' doesn't exist`
 - The MySQL specific error code: 1146. Stable across GA releases, can be looked up in documentation.
 - The five-character SQLSTATE value: 42S02. Standardized to ANSI SQL and ODBC. HY000 means a MySQL specific error not mappable to a SQLSTATE value.

PERROR

- Also note that MySQL error messages will sometimes include an additional error code in parenthesis. In this case, use `perror` to figure out what happened:
- `ERROR 1005 at line 20: Can't create table './test/test.frm' (errno: 150)`
 - `shell> perror 150`
 - MySQL error code 150: Foreign key constraint is incorrectly formed

REPLICATION FAILURES

- Generally, once replication is established and working, problems only arise when replicated queries fail on the slave. To remedy the problem (if you are certain the data integrity is solid):
- To skip just one query:
 - `SET GLOBAL SQL_SLAVE_SKIP_COUNTER=1; START SLAVE;`
- To skip all queries that are failing on an error code, add the following to `/etc/my.cnf`:
 - `slave-skip-errors = code`


```
slideshow.end();
```