抽丝剥茧之 - MySQL疑难杂症排查

叶金荣 - 知数堂培训联合创始人、MySQL专家

2016.10.28

PostgreSQL

网站打开怎么那么慢啊

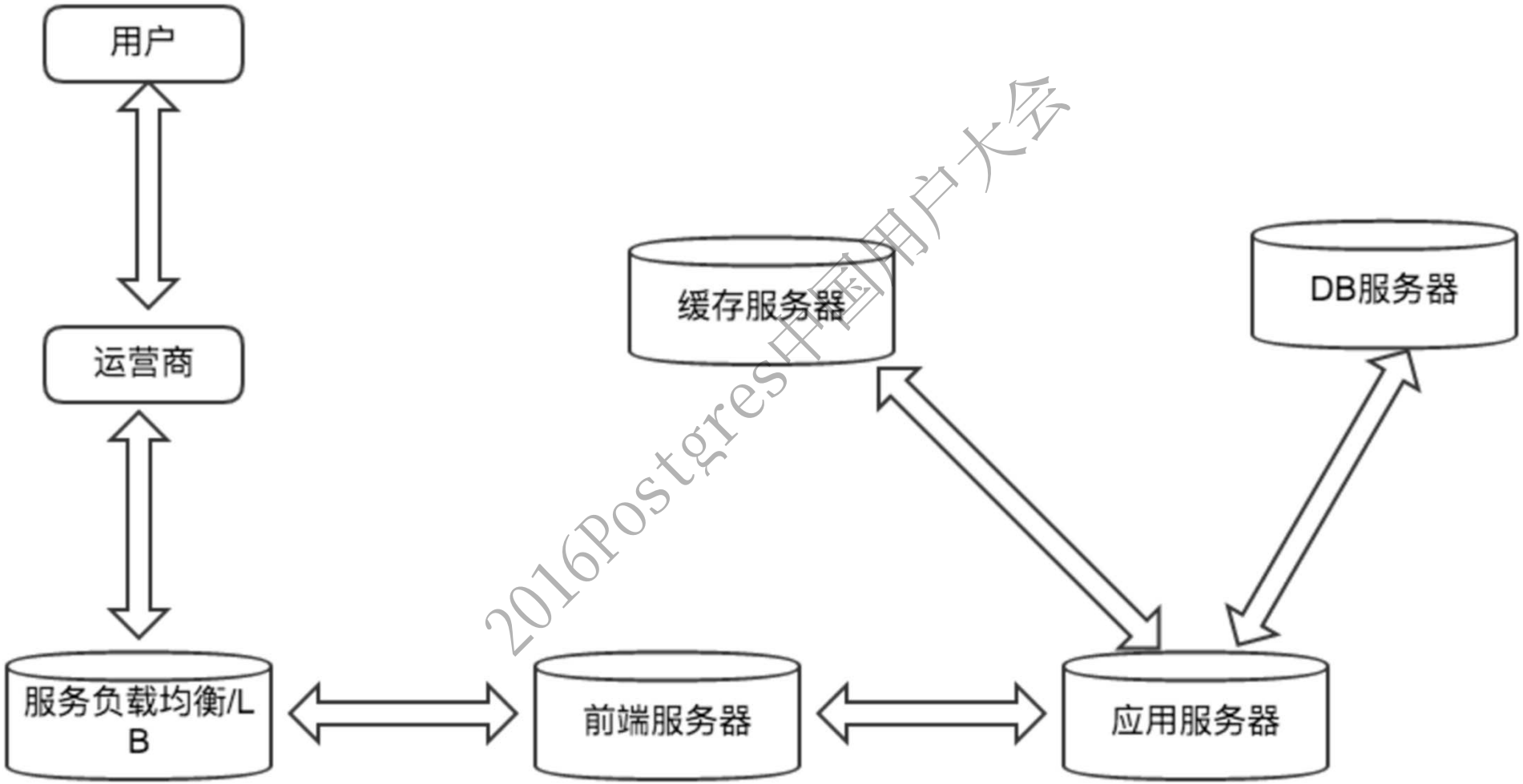
有个活动页面打不开啦

技术同学们赶紧看下嘛！！！

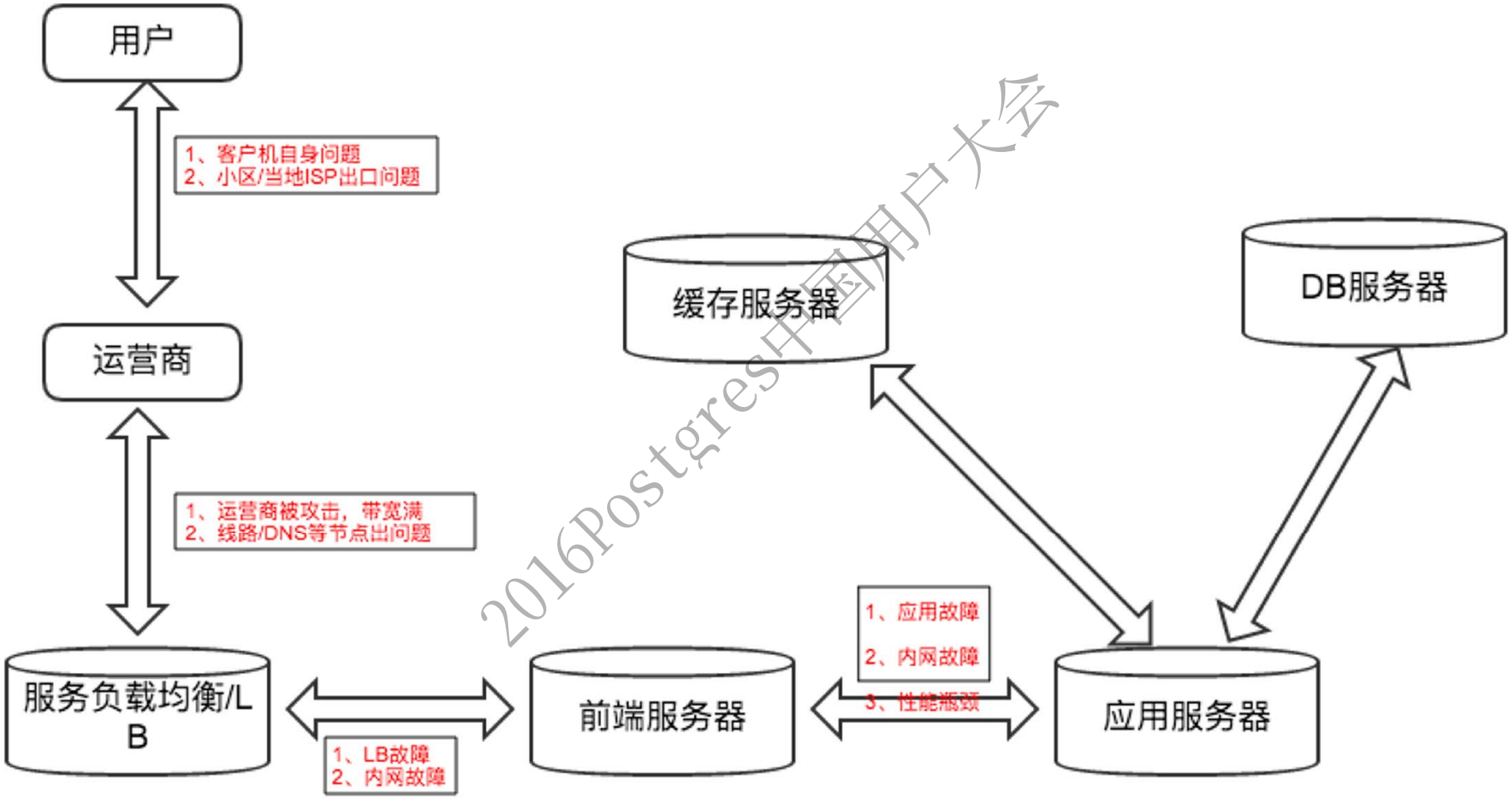
别急，先确认问题

- 全网用户打开都慢还是个别线路问题？
- 每次页面打开都慢还是客户端偶发现象？
- 整个页面打开都慢还是个别元素导致？
- 所有页面打开都慢还是只有动态页面慢？
- 页面打不开有什么提示代码吗，是404还是其他

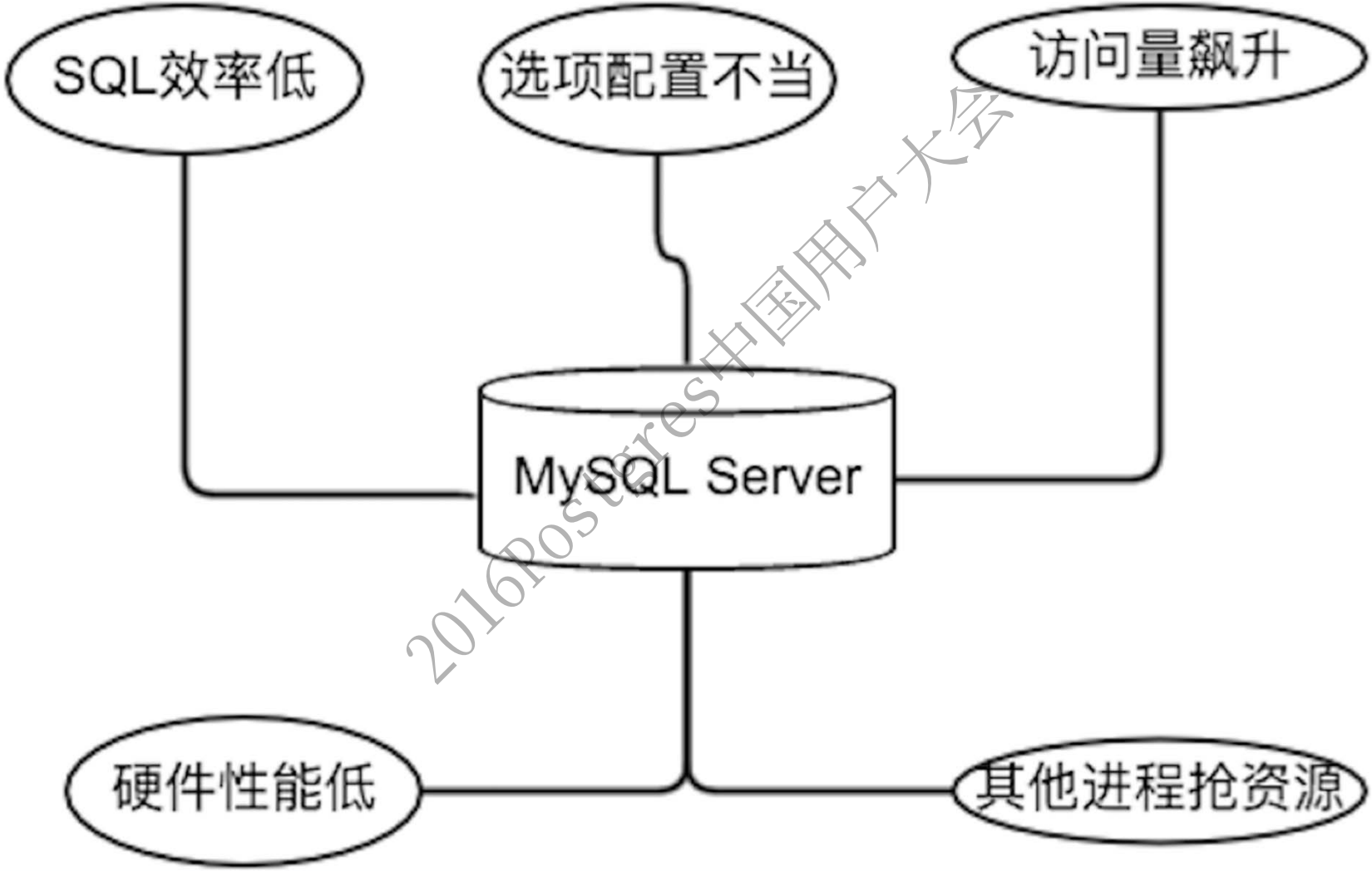
问题追踪



问题追踪



常见瓶颈



确认是MySQL存在瓶颈

- top/free/vmstat/sar/mpstat确认
  - 确认mysqld进程的CPU消耗占比
  - 确认mysqld进程的CPU消耗是%user，还是%sys高
  - 确认是否物理内存不够用了
  - 确认是否有swap产生
  - 确认CPU上是否有大量中断（或中断不均）

确认是MySQL存在瓶颈

- top

```
top - 23:52:22 up 44 days, 13:18,  2 users,  load average: 16.36, 11.54, 1.11
Tasks: 726 total,   1 running, 724 sleeping,   1 stopped,   0 zombie
%Cpu(s): 62.7 us,  5.6 sy,  0.0 ni, 27.8 id,  3.0 wa,  0.0 hi,  1.0 si,  0.0 st
KiB Mem : 98719944 total,   335312 free, 85283712 used, 13100916 buff/cache
KiB Swap: 20971516 total, 20908900 free,   62616 used. 12792824 avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
185880 mysql     20   0 15.784g 2.273g   7496 S 784.1  2.4   2:15.56 mysqld
   568 root      20   0       0      0      0 S   1.0  0.0   1:48.35 kswapd1
193905 mysql     20   0 80.413g 0.076t  12772 S   1.0 82.8 227:08.13 mysqld
   567 root      20   0       0      0      0 S   0.7  0.0   2:58.29 kswapd0
 19924 root      20   0  158224   2856   1536 R   0.7  0.0   0:00.13 top
```

– 哥很忙~

确认是MySQL存在瓶颈

- free

```
[root@imysql ~]# free -m
              total        used        free      shared  buff/cache   available
Mem:          96506       54265         452          79         417         417
Swap:         20479          21       20458
```

  - 嗯，看出来啥了没

确认是MySQL存在瓶颈

- vmstat -S m 1

```
vmstat -S m 1
procs -----------memory----------  ---swap--  -----io----  --system--  ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs      us sy id wa st
42  0     80    338      0  13400    0    0   536 112400 34657 221678 43  6 50  1  0
 0  1     80    358      0  13379    0    0   372  71421 33717 227494 43  6 50  1  0
 0  1     80    352      0  13383    0    0   524 106847 28957 202112 35  6 58  1  0
33  0     80    354      0  13385    0    0   540  73859 38159 259421 40  7 52  1  0
 0  1     80    347      0  13386    0    0   456 108401 65412 205543 33  6 60  1  0
31  1     81    329      0  13409    0    0   448  76707 37723 240639 45  7 48  1  0
27  0     81    344      0  13392    0    0   360 104701 40005 188579 34  6 59  1  0
35  0     81    344      0  13395    0    0   360  78647 37873 222284 39  7 54  1  0
```

  - CPU和I/O的压力都不算小

确认是MySQL存在瓶颈

- sar -u 1

```
Linux 3.10.0-327.el7.x86_64 (imysql)     10/09/2016     _x86_64_ (32 CPU)

11:57:57 PM     CPU     %user     %nice   %system   %iowait    %steal     %idle
11:57:58 PM     all     66.49      0.00      8.43      0.38      0.00     24.70
11:57:59 PM     all     72.19      0.00      9.26      0.16      0.00     18.39
11:58:00 PM     all     74.34      0.00      9.08      0.09      0.00     16.49
11:58:01 PM     all     72.09      0.00     10.03      0.13      0.00     17.76
11:58:02 PM     all     67.52      0.00     12.30      0.13      0.00     20.06
11:58:03 PM     all     66.55      0.00     10.54      0.22      0.00     22.69
11:58:04 PM     all     72.03      0.00      9.70      0.09      0.00     18.17
11:58:05 PM     all     72.08      0.00      9.77      0.09      0.00     18.06
11:58:06 PM     all     72.20      0.00      9.87      0.09      0.00     17.83
```


– CPU好忙的样子

确认是MySQL存在瓶颈

- sar -d 1

```
Linux 3.10.0-327.el7.x86_64 (imysql)      10/10/2016      _x86_64_ (32 CPU)

12:01:25 AM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
12:01:26 AM    dev8-16   2990.00      0.00 267137.00     89.34      0.20      0.07      0.05     14.60
12:01:27 AM    dev8-16   2949.00      0.00 248692.00     84.33      0.18      0.06      0.05     15.00
12:01:28 AM    dev8-16   3043.00      0.00 333020.00    109.44      0.24      0.08      0.07     20.10
12:01:29 AM    dev8-16   3199.00      0.00 285494.00     89.24      0.18      0.06      0.04     13.20
12:01:30 AM    dev8-16   3207.00      0.00 329196.00    102.65      0.17      0.05      0.04     12.80
12:01:31 AM    dev8-16   3125.00      0.00 304370.00     97.40      0.23      0.07      0.05     16.10
12:01:32 AM    dev8-16   3062.00      0.00 232723.00     76.00      0.12      0.04      0.04     11.50
12:01:33 AM    dev8-16   3060.00      0.00 266480.00     87.08      0.15      0.05      0.04     12.50
12:01:34 AM    dev8-16   2913.00      0.00 261099.00     89.63      0.20      0.07      0.06     18.50

Average:       dev8-16   3069.59      0.00 280986.34     91.54      0.19      0.06      0.05     15.02
```

  - I/O压力不小

确认是MySQL存在瓶颈

- mpstat -P ALL -I SUM 1 100

```
Average:      CPU    intr/s
Average:      all  50272.00
Average:        0   1815.00
Average:        1   1698.00
Average:        2   1704.33
......
Average:       27   5240.33
Average:       29   1287.33
Average:       30   1575.00
Average:       31   1279.33
```

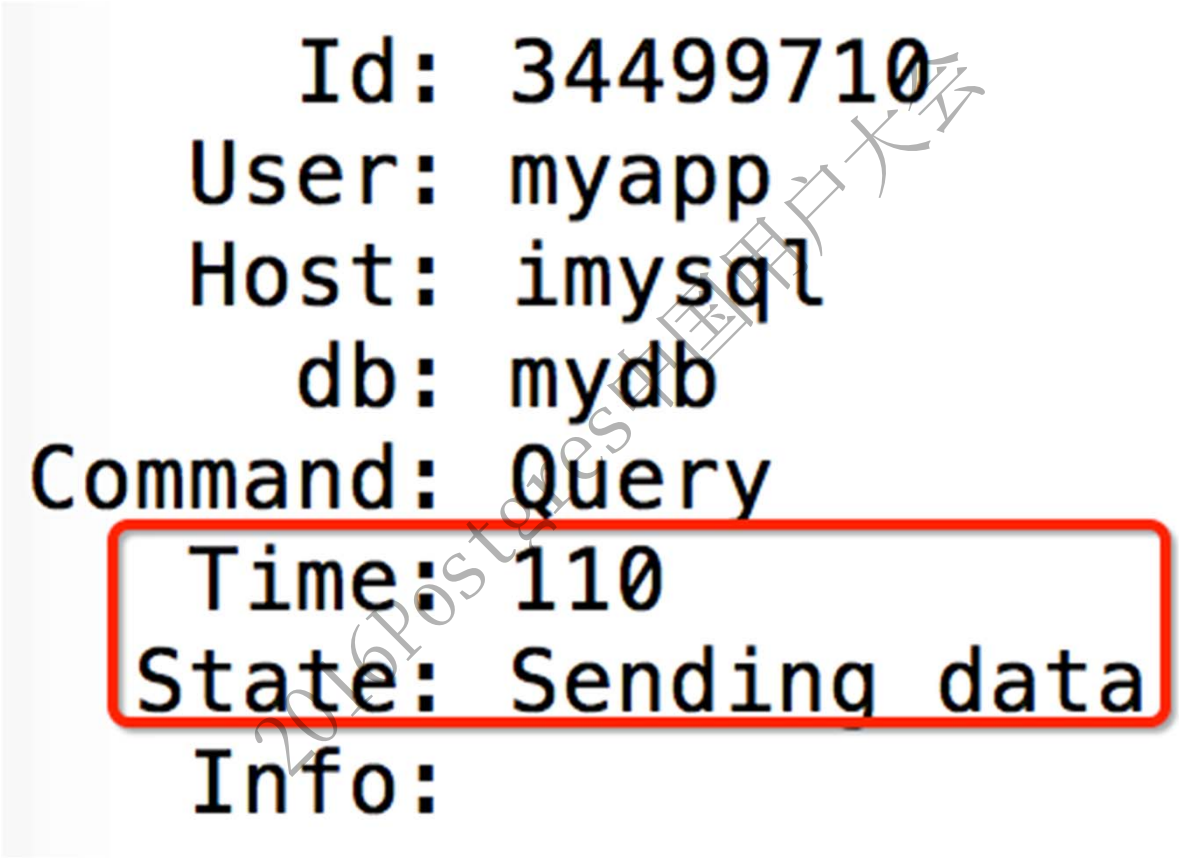
  - 中断不均衡

那就一个个解决吧

看现场：MySQL在干嘛

- show [full] processlist
- locks & lock waits
  - I_S.innodb_trx
  - I_S.innodb_locks
  - I_S.innodb_lock_waits
  - sys.innodb_lock_waits
- show engine innodb status
  - log flush
  - checkpoint

查看MySQL线程状态



长时间的Sending data

- 从引擎层读取数据返回给Server端的状态
- 长时间存在的原因
  - 没适当的索引，查询效率低
  - 读取大量数据，读取缓慢
  - 系统负载高，读取缓慢

长时间的Sending data

- 怎么办
  - 加上合适的索引
  - 或者改写SQL，提高效率
  - 增加LIMIT限制每次读取数据量
  - 检查&升级I/O设备性能

查看MySQL线程状态

```
     Id: 34399716
   User: myapp
   Host: imysql
     db: mydb
Command: Query
   Time: 15132
  State: Waiting for table metadata lock
   Info:
```

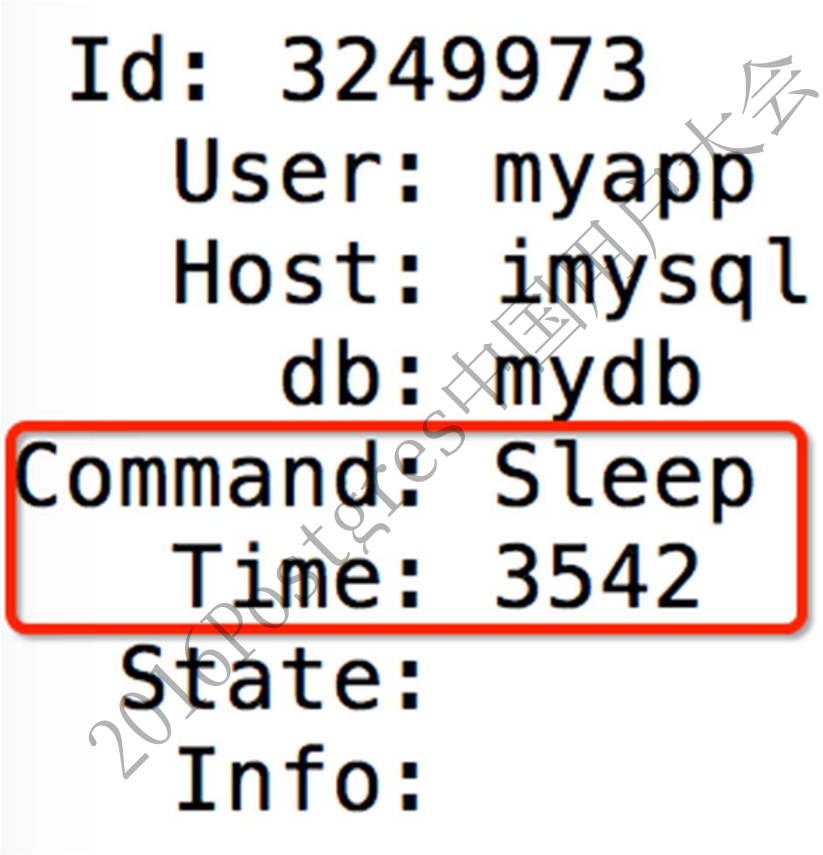
长时间等待MDL锁

- 原因
  - DDL被阻塞，进而阻塞其他后续SQL
  - DDL之前的SQL长时间未结束

长时间等待MDL锁

- 怎么办
  - 提高每个SQL的效率
  - 干掉长时间运行的SQL
  - 把DDL放在半夜等低谷时段
  - 采用pt-osc执行DDL

查看MySQL线程状态



长时间的Sleep

- 看似无害，实则可能是大害虫
  - 占用连接数
  - 消耗内存未释放
  - 可能有行锁（甚至是表锁）未释放

长时间的Sleep

- 怎么办
  - 适当调低timeout
  - 主动kill超时不活跃连接
  - 定期检查锁、锁等待
  - 可以利用pt-kill工具

还有哪些状态要关注的呢

- Copy to tmp table
  - 执行alter table修改表结构，需要生成临时表
  - 建议放在夜间低谷执行，或者用pt-osc

还有哪些状态要关注的呢

- Copying to tmp table [on disk]
- Creating tmp table
  - 常见于group by没有索引的情况
  - 需要拷贝数据到临时表[内存/磁盘上]
  - 执行计划中会出现Using temporary关键字
  - 建议创建合适的索引，消除临时表

还有哪些状态要关注的呢

- Creating sort index
  - 常见于order by没有索引的情况
  - 需要进行filesort排序
  - 执行计划中会出现Using filesort关键字
  - 建议创建排序索引

还有哪些状态要关注的呢

- Waiting for global read lock
- Waiting for query cache lock
- Waiting for table level lock
- Waiting for table metadata lock

还应该怎么关注

- slow query log

```
# Thread_id: 44  Schema: test Last_errno: 0  Killed: 0
# Query_time: 1.841461  Lock_time: 0.000051  Rows_sent: 1  Rows_examined:
13719413  Rows_affected: 0  Rows_read: 13719413
# Bytes_sent: 70  Tmp_tables: 0  Tmp_disk_tables: 0  Tmp_table_sizes: 0
# InnoDB_trx_id: 4A0B89B
# QC_Hit: No  Full_scan: Yes  Full_join: No  Tmp_table: No  Tmp_table_on_disk:
No
# Filesort: No  Filesort_on_disk: No  Merge_passes: 0
# InnoDB_IO_r_ops: 0  InnoDB_IO_r_bytes: 0  InnoDB_IO_r_wait: 0.000000
# InnoDB_rec_lock_wait: 0.000000  InnoDB_queue_wait: 0.000000
# InnoDB_pages_distinct: 12252
SET timestamp=1312970168;
select count(*) from t1_rnd;
```

还应该怎么关注

- innodb lock wait

```
(root@imysql)[information_schema]>select * from innodb_locks;
+-----------------+-------------+-----------+-----------+-------------+------------+------------+-----------+----------+-----------+
| lock_id         | lock_trx_id | lock_mode | lock_type | lock_table  | lock_index | lock_space | lock_page | lock_rec | lock_data |
+-----------------+-------------+-----------+-----------+-------------+------------+------------+-----------+----------+-----------+
| 30251:149:173:20 | 30251       | X         | RECORD    | `test`.`t1` | PRIMARY    |        149 |       173 |       20 | 49130     |
| 30250:149:173:20 | 30250       | X         | RECORD    | `test`.`t1` | PRIMARY    |        149 |       173 |       20 | 49130     |
+-----------------+-------------+-----------+-----------+-------------+------------+------------+-----------+----------+-----------+
```

```
(root@imysql)[information_schema]>select * from innodb_lock_waits;
+-------------------+-------------------+-------------------+-------------------+
| requesting_trx_id | requested_lock_id | blocking_trx_id | blocking_lock_id |
+-------------------+-------------------+-------------------+-------------------+
| 30251             | 30251:149:173:20  | 30250             | 30250:149:173:20 |
+-------------------+-------------------+-------------------+-------------------+
```

```
(root@imysql)[sys]>select * from innodb_lock_waits\G
*************************** 1. row ***************************
                wait_started: 2016-09-27 12:57:27
                    wait_age: 00:00:04
               wait_age_secs: 4
                locked_table: `test`.`t1`
                locked_index: PRIMARY
                 locked_type: RECORD
              waiting_trx_id: 30282
         waiting_trx_started: 2016-09-27 12:57:27
             waiting_trx_age: 00:00:04
     waiting_trx_rows_locked: 1
   waiting_trx_rows_modified: 0
                 waiting_pid: 32
               waiting_query: update t1 set c4 = 12345 where c1 >=10 and c1<=15
             waiting_lock_id: 30282:149:6:8
           waiting_lock_mode: X
             blocking_trx_id: 30281
                blocking_pid: 31
              blocking_query: NULL
            blocking_lock_id: 30281:149:6:8
          blocking_lock_mode: X
        blocking_trx_started: 2016-09-27 12:57:24
          blocking_trx_age: 00:00:07
    blocking_trx_rows_locked: 6
  blocking_trx_rows_modified: 5
     sql_kill_blocking_query: KILL QUERY 31
sql_kill_blocking_connection: KILL 31
```

还应该怎么关注

- show engine innodb status

```
---TRANSACTION 30260, ACTIVE 16 sec
80 lock struct(s), heap size 24784, 32833 row lock(s), undo log entries 32755
MySQL thread id 22, OS thread handle 123145316356096, query id 195 localhost root cleaning up
TABLE LOCK table `test`.`t1` trx id 30260 lock mode IX
RECORD LOCKS space id 149 page no 6 n bits 496 index PRIMARY of table `test`.`t1` trx id 30260 lock_mode X locks rec but not gap
Record lock, heap no 8 PHYSICAL RECORD: n_fields 6; compact format; info bits 0
```

还有什么可以预防的

- 业务上线前，提前消灭垃圾SQL
  - 在开发或压测环境中
  - 调低long_query_time的值，甚至设为0
  - 开启log_queries_not_using_indexes
  - 分析slow query log，并消除潜在隐患SQL

还有什么可以预防的

- 用更好的设备
  - CPU更快更多核
  - 内存更快更大
  - 用更快的I/O设备
  - 用更好的网络设备

还有什么可以预防的

- 让OS保持高效
  - 采用xfs/ext4文件系统
  - 采用noop/deadline io scheduler

回顾

- 根据业务架构，逐步排查问题所在
- 熟练使用各种工具定位瓶颈根源
- 根据各种不同情况应对解决

- 事实上，有经验的DBA，通常看到现场后就能很快判断出问题所在了

谢谢，希望有所帮助

Q & A

PostgreSQL