3.27 Percona Toolkit 使用场景分享

作者：杜金洋

更方便的数据归档 pt-archiver

某些存在时效性的数据，在达到一定条件后，需要进行归档和回收处理。

pt-archiver 工具可以帮助我们快速的进行数据行的归档和回收。

```
# 归档到其他数据库,并删除原表中对应的行
pt-archiver --source h=172.20.134.1,P=5722,u=repl,p=repl,D=sbtest,t=sbtest1,A=utf8  --where "id<=1000" --dest h=172.20.134.3,P=5722,u=dba,p=dba,D=sbtest,t=sbtest1,A=utf8

# 归档到其他数据库，不删除原表中对应的行
pt-archiver --source h=172.20.134.1,P=5722,u=repl,p=repl,D=sbtest,t=sbtest1,A=utf8 --no-delete --where "id<=1000" --dest h=172.20.134.3,P=5722,u=dba,p=dba,D=sbtest,t=sbtest1,A=utf8

# 归档到文件,并删除原表中对应的行
pt-archiver --source h=172.20.134.1,P=5722,u=repl,p=repl,D=sbtest,t=sbtest1,A=utf8 --file=/tmp/archive.save --where "id<=1000"

# 归档到文件，不删除原表中对应的行
pt-archiver --source h=172.20.134.1,P=5722,u=repl,p=repl,D=sbtest,t=sbtest1,A=utf8 --no-delete --file=/tmp/archive.save --where "id<=1000"

# 导入归档文件
mysql> load data infile "/tmp/archive.save" into table sbtest.sbtest1;
Query OK, 1000 rows affected (0.49 sec)
Records: 1000  Deleted: 0  Skipped: 0  Warnings: 0
```

更快速的配置对比 pt-config-diff

在我们日常工作中，大家一定遇到过以下场景：

- 若干套 MySQL 环境，只有一套：
  - 行为异常，怀疑触发 bug
  - 性能异常，比其他环境都要低

在这种场景下，我们一般的做法是首先控制变量，查看软硬件配置，以及 MySQL 的参数配置。

关于 MySQL 的参数配置对比，如果我们人工对比的话只会关注某些重点参数，而缺少了整体细节上的的对比。

在这里我们推荐给大家 Percona Toolkit 中的一个工具 pt-config-diff

```
# 指定 DSN，对比所有运行时参数
# 指定 --report-width 200,防止某些参数过长被截断
[root@172-20-134-1 /]#  pt-config-diff h=172.20.134.1,P=5722,u=repl,p=repl
    h=172.20.134.3,P=5722,u=dba,p=dba --report-width 200
4 config differences
Variable                  172-20-134-1                           172-20-134-3
========================= ====================================== ======================================
general_log_file          /data/mysql/data/5.7.22/172-20-134-1.log
    /data/mysql/data/5.7.22/172-20-134-3.log
gtid_executed             234303e2-20cb-11ea-a5a3-0242ac148601:1   2348904f-20cb-
    11ea-a565-0242ac148603:1
hostname                  172-20-134-1                           172-20-134-3
server_uuid               234303e2-20cb-11ea-a5a3-0242ac148601    2348904f-20cb-
    11ea-a565-0242ac148603

# 指定配置文件，对比配置文件的差异
[root@172-20-134-1 /]# pt-config-diff /data/mysql/etc/5.7.22.cnf /tmp/5.7.22.cnf --
    report-width 200
2 config differences
Variable                /data/mysql/etc/5.7.22.cnf /tmp/5.7.22.cnf
======================= ========================== ===============
max_allowed_packet      16777216                    67108864
relay_log_recovery      1                          0
```

更准确的复制延时 pt-heartbeat

在 MySQL 中，复制延迟可以理解为由两部分组成：

1. 主库已经生成了 BINLOG，但是还没有发送给从库 -- 我们在这里称之为：日志延迟
2. 从库已经接收到了 BINLOG，但是还没有应用完成 -- 我们在这里称之为：应用延迟

MySQL 原生的查看复制延迟的手段为：show slave status\G 中的 Seconds_Behind_Master 。

这种观测手法只能观测出应用延迟。在异步复制或降级的半同步复制下，误差较大，无法准确的反映出整体复制延时。

pt-heartbeat 提供了一种更为准确的观测手法，简要逻辑如下：

1. 在 Master 上循环插入：insert into database.heartbeat (master_now) values(NOW())
2. database.heartbeat 的变更会跟随主从复制流向从库

3. 系统当前时间 - 从库表中的时间 = 从库实际的复制延时

使用方式如下

```
# 向 Master 开启循环插入时间戳, interval 为每次插入的间隔时间
pt-heartbeat -D delay_checker --create-table --interval=5 --update -h
    172.20.134.1,P=5722,u=repl,p=repl

# 检查指定从库的复制延迟
pt-heartbeat -D delay_checker  --check 172.20.134.2,P=5722,u=repl,p=repl
# PS: 上面这条命令会使用当前系统的时间减去 delay_checker.heartbeat 表中的时间.
# 如果在从库上执行,需要保证从库的系统时间与主库一致.否则会导致延迟计算错误
```

更简单的参数配置建议 pt-variable-advisor

toolkit 中包含了一个简单的 MySQL 参数优化器，可以对参数配置做简单的优化建议。

```
[root@172-20-134-1 /]# pt-variable-advisor h=172.20.134.1,P=5722,u=repl,p=repl
# WARN delay_key_write: MyISAM index blocks are never flushed until necessary.
# WARN innodb_log_buffer_size: The InnoDB log buffer size generally should not be
    set larger than 16MB.
# NOTE innodb_max_dirty_pages_pct: The innodb_max_dirty_pages_pct is lower than the
    default.
# NOTE max_binlog_size: The max_binlog_size is smaller than the default of 1GB.
# NOTE port: The server is listening on a non-default port.
# NOTE sort_buffer_size-1: The sort_buffer_size variable should generally be left at
    its default unless an expert determines it is necessary to change it.
# WARN expire_logs_days: Binary logs are enabled, but automatic purging is not
    enabled.
# NOTE innodb_data_file_path: Auto-extending InnoDB files can consume a lot of disk
    space that is very difficult to reclaim later.
# WARN log_output: Directing log output to tables has a high performance impact.
# WARN myisam_recover_options: myisam_recover_options should be set to some value
    such as BA
```

更易用的调试工具 pt-pmp

在某些情况下，我们肯定会遇到某些故障无法从日志，以及状态命令中找到原因，需要深入到程序逻辑级别。

又或者我们需要立即通过非常规手段恢复故障数据库，但是又想保留足够多的故障信息。来避免我们事后复现问题的头疼。

pt-pmp 便是在这种场景下帮助我们的工具。它会使用 gdb 来打印 mysqld 的堆栈信息，并把调用链相同的线程堆栈合并。

堆栈合并的功能对于 MySQL 这种多线程的应用非常有帮助，会节省我们大量的时间。

```
# pt-pmp --binary /path/to/bin/mysqld --pid 11788
Sat Dec 21 23:14:06 CST 2019
#   第一列为线程数量  第二列为线程调用栈信息
    180
    poll(libc.so.6),vio_io_wait(viosocket.c:797),vio_socket_io_wait(viosocket.c:88),v
    io_read(viosocket.c:143),net_read_raw_loop(net_serv.cc:694),net_read_packet_heade
    r(net_serv.cc:778),net_read_packet(net_serv.cc:778),my_net_read(net_serv.cc:921),
    Protocol_classic::read_packet(protocol_classic.cc:815),Protocol_classic::get_comm
    and(protocol_classic.cc:972),do_command(sql_parse.cc:971),handle_connection(conne
    ction_handler_per_thread.cc:313),pfs_spawn_thread(pfs.cc:2197),start_thread(libpt
    hread.so.0),clone(libc.so.6)
     57
    pthread_cond_wait,native_cond_wait(thr_cond.h:147),my_cond_wait(thr_cond.h:147),i
    nline_mysql_cond_wait(thr_cond.h:147),Per_thread_connection_handler::block_until_
    new_connection(thr_cond.h:147),handle_connection(connection_handler_per_thread.cc
    :344),pfs_spawn_thread(pfs.cc:2197),start_thread(libpthread.so.0),clone(libc.so.6)
     30
    __io_getevents_0_4(libaio.so.1),LinuxAIOHandler::collect(os0file.cc:2513),LinuxAI
    OHandler::poll(os0file.cc:2673),os_aio_linux_handler(os0file.cc:2729),os_aio_hand
    ler(os0file.cc:2729),fil_aio_wait(fil0fil.cc:5862),io_handler_thread(srv0start.cc
    :319),start_thread(libpthread.so.0),clone(libc.so.6)
      9
    pthread_cond_wait,wait(os0event.cc:179),os_event::wait_low(os0event.cc:179),srv_w
    orker_thread(srv0srv.cc:2527),start_thread(libpthread.so.0),clone(libc.so.6)
      9
    pthread_cond_wait,wait(os0event.cc:179),os_event::wait_low(os0event.cc:179),buf_f
    lush_page_cleaner_worker(buf0flu.cc:3507),start_thread(libpthread.so.0),clone(lib
    c.so.6)
      3
    sigwait(libpthread.so.0),signal_hand(mysqld.cc:2132),pfs_spawn_thread(pfs.cc:2197
    ),start_thread(libpthread.so.0),clone(libc.so.6)
      3
    sigwaitinfo(libc.so.6),timer_notify_thread_func(posix_timers.c:89),pfs_spawn_thre
    ad(pfs.cc:2197),start_thread(libpthread.so.0),clone(libc.so.6)
      3
    pthread_cond_wait,wait(os0event.cc:179),os_event::wait_low(os0event.cc:179),srv_p
    urge_coordinator_suspend(srv0srv.cc:2683),srv_purge_coordinator_thread(srv0srv.cc
    :2683),start_thread(libpthread.so.0),clone(libc.so.6)
      3
    pthread_cond_wait,wait(os0event.cc:179),os_event::wait_low(os0event.cc:179),buf_r
    esize_thread(buf0buf.cc:3027),start_thread(libpthread.so.0),clone(libc.so.6)
```

```
  3
pthread_cond_wait,wait(os0event.cc:179),os_event::wait_low(os0event.cc:179),buf_d
ump_thread(buf0dump.cc:792),start_thread(libpthread.so.0),clone(libc.so.6)
  3
pthread_cond_wait,native_cond_wait(thr_cond.h:147),my_cond_wait(thr_cond.h:147),i
nline_mysql_cond_wait(thr_cond.h:147),compress_gtid_table(thr_cond.h:147),pfs_spa
wn_thread(pfs.cc:2197),start_thread(libpthread.so.0),clone(libc.so.6)
  3
pthread_cond_timedwait,os_event::timed_wait(os0event.cc:316),os_event::wait_time_
low(os0event.cc:488),srv_monitor_thread(srv0srv.cc:1592),start_thread(libpthread.
so.0),clone(libc.so.6)
  3
pthread_cond_timedwait,os_event::timed_wait(os0event.cc:316),os_event::wait_time_
low(os0event.cc:488),srv_error_monitor_thread(srv0srv.cc:1758),start_thread(libpt
hread.so.0),clone(libc.so.6)
  3
pthread_cond_timedwait,os_event::timed_wait(os0event.cc:316),os_event::wait_time_
low(os0event.cc:488),pc_sleep_if_needed(buf0flu.cc:2700),buf_flush_page_cleaner_c
oordinator(buf0flu.cc:2700),start_thread(libpthread.so.0),clone(libc.so.6)
  3
pthread_cond_timedwait,os_event::timed_wait(os0event.cc:316),os_event::wait_time_
low(os0event.cc:488),lock_wait_timeout_thread(lock0wait.cc:497),start_thread(libp
thread.so.0),clone(libc.so.6)
  3
pthread_cond_timedwait,os_event::timed_wait(os0event.cc:316),os_event::wait_time_
low(os0event.cc:488),ib_wqueue_timedwait(ut0wqueue.cc:168),fts_optimize_thread(ft
s0opt.cc:2910),start_thread(libpthread.so.0),clone(libc.so.6)
```

总结

Percona Toolkit 中集成了很多对于 DBA 来说非常有用的工具，合理利用它们在某种场景下会减轻一部分工作量。

愿这次的分享能帮助到大家节省时间，提高生活质量。

附录：Percona Toolkit 快速安装

```
wget https://www.percona.com/downloads/percona-toolkit/3.1.0/binary/tarball/percona-
    toolkit-3.1.0_x86_64.tar.gz
tar -xvf percona-toolkit-3.1.0_x86_64.tar.gz -C /usr/local
```