功能介绍

> 将 mysql 数据库中表的记录归档到另外一个表或者文件，也可以直接 进行记录的删除操作。

用法介绍

```
pt-archiver [OPTION...] --source DSN --where WHERE
```

常用参数

--limit 1000　　每次取1000行数据用pt-archive处理，Number of rows to fetch and archive per statement.

--txn-size 1000　设置1000行为一个事务提交一次，Number of rows pertransaction.

--where 'id<3000'　设置操作条件

--progress 5000　每处理5000行输出一次处理信息

--statistics　　输出执行过程及最后的操作统计。（只要不加上--quiet，默认情况下pt-archive都会输出执行过程的）

--charset=UTF8　指定字符集为UTF8

--bulk-delete　　批量删除source上的旧数据(例如每次1000行的批量删除操作)

--bulk-insert　　批量插入数据到dest主机（看dest的general log发现它是通过在dest主机上LOAD DATA LOCAL INFILE插入数据的）

--replace　　将insert into 语句改成replace写入到dest库

--sleep 120　　每次归档了limit个行记录后的休眠120秒（单位为秒）

--file'/root/test.txt'

--purge　　删除source数据库的相关匹配记录

--header　　输入列名称到首行（和--file一起使用）

--no-check-charset　不指定字符集

--check-columns　检验dest和source的表结构是否一致，不一致自动拒绝执行（不加这个参数也行。默认就是执行检查的）

--no-check-columns　不检验dest和source的表结构是否一致，不一致也执行（会导致dest上的无法与source匹配的列值被置为null或者0）

--check-interval　　默认1s检查一次

--local　　不把optimize或analyze操作写入到binlog里面（防止造成主从延迟巨大）

--retries　　超时或者出现死锁的话，pt-archiver进行重试的间隔（默认1s）

--no-version-check　目前为止，发现部分pt工具对阿里云RDS操作必须加这个参数

--analyze=ds　　操作结束后，优化表空间（d表示dest，s表示source）

默认情况下，pt-archiver操作结束后，不会对source、dest表执行analyze或optimize操作，因为这种操作费时间，并且需要你提前预估有足够的磁盘空间用于拷贝表。一般建议也是pt-archiver操作结束后，在业务低谷手动执行analyze table用以回收表空间。

—set-vars　可以设置mysql的变量,多个变量用逗号分割.
　　--set-vars wait_timeout=500

--why-quit　输出退出理由　　Print reason for exiting unless rows exhausted

使用示例

- 示例1 批量删除db1库中的表t1中date_time <'2017-06-01 00:00:00的数据

```
nohup pt-archiver --source h=127.0.0.1,u=root,p=root,P=3306,D=db1,t=t1,A=utf8mb4 --where
"date_time <'2017-06-01 00:00:00'" --statistics --progress 5000 --limit 5000 --txn-size 5000 --
bulk-delete --purge &>archiver.log &
```

- 示例2 将db2中表tb2中create_time<'2018-06-25 00:00:00'的数据归档到文件，不删除源数据

```
nohup pt-archiver --source h=127.0.0.1,u=root,p=root,P=3306,D=db2,t=tb2,A=utf8mb4 --where
"create_time <'2018-06-25 00:00:00'" --statistics --progress 5000 --limit 5000 --txn-size 5000 --
charset=UTF8 --no-delete --file '/data0/data_bak/archiver_%Y%m%d' &>archiver_to_file.log &
```

- 示例3 将db2中表tb2中create_time<'2018-06-25 00:00:00'的数据归档到表t2_20180807，不删除源数据

```
 nohup pt-archiver --source h=10.12.21.109,u=root,p=root,P=3306,D=db2,t=tb2,A=utf8mb4 --where
"create_time <'2018-06-25 00:00:00'" --statistics --progress 5000 --limit 5000 --txn-size 5000 --
charset=UTF8 --no-delete --dest h=10.12.21.109,u=root,p=root,P=3306,D=db2,t=t2_20180807,A=utf8mb4
&>archiver_to_table.log &
```

- 示例4 删除数据并不写入binlog，DSN中的 `b=1`

```
nohup pt-archiver --source h=127.0.0.1,u=root,p=root,P=3306,D=db1,t=t1,A=utf8mb4,b=1 --where
"date_time <'2017-07-01 00:00:00'" --statistics --progress 5000 --limit 1000 --txn-size 1000 --
bulk-delete --purge &>archiver_purge.log &
```

- 示例5 删除数据前设置参数 `wait_timeout`

```
nohup pt-archiver --source h=127.0.0.1,u=root,p=root,P=3306,D=db1,t=t1,A=utf8mb4 --where
"date_time <'2017-07-07 00:00:00'" --set-vars wait_timeout=1000 --statistics --progress 5000 --
limit 5000 --txn-size 1000 --bulk-delete --purge &>>archiver_20180808.log &
```

减小主从延迟的方法

- 业务低峰期操作
- 不使用批量删除操作 `bulk-delete`
- 增加参数检测主从延迟 `--max-lag=60  --check-interval=10` ,例子中为检测延迟到达60S时暂停10S
- 增加参数 `--sleep=2` ,例子中的为每次归档完Limit数据之后停顿2S
- 操作时设置 `sql_log_bin=0` ,分别在主库和从库进行操作

删除历史数据脚本

```
#! /bin/bash
#author:caopeng
#date:2018-08-13
#使用pt工具删除3个月前的历史数据并进行碎片整理

WORKPATH="/home/mysql/scripts/purge_history_data"
DATE=`date +%Y%m%d`
THREE_MONTH_AGO=`date -d "3 months ago" +%Y%m01`
PT_AC="/usr/bin/pt-archiver"
PT_OSC="/usr/bin/pt-online-schema-change"
PROD_MYSQL_USER="caopeng"
PROD_MYSQL_PASSWORD="xxxx"
PROD_MYSQL_HOST="127.0.0.1"
PROD_MYSQL_PORT="3306"
DUMP_DB="db1"   #需要删除的表所在的库
DUMP_TB_LIST=(t1 t2 t3)  #需要删除的表
MAIL_BIN="python /home/mysql/scripts/mail/mail.py"    # 调用邮件发送脚本
MAIL_TO="zhangsan@fangdd.com"
MAIL_SUB="archiver  at node1"
ANA_FILE=${WORKPATH}/maillog/"maillog"${DATE}_${PROD_MYSQL_PORT}.log
msg=""
##end set environment

#sent mail last day perfomance alert
my_sentmail()
{
   ${MAIL_BIN}  -s "${MAIL_SUB}" -t "${MAIL_TO}" -b "${msg}" -a "${ANA_FILE}"
}



cd $WORKPATH
if [ ! -d $WORKPATH/backup_tmp ]
then
        mkdir -p $WORKPATH/backup_tmp
        echo "create new dir backup_tmp." >>purge_history_data.log
fi

find $WORKPATH/backup_tmp/ -mtime +14 -type f -exec rm {}  \;



for DUMP_TB in ${DUMP_TB_LIST[@]}
do
    $PT_AC --source
u=$PROD_MYSQL_USER,p=$PROD_MYSQL_PASSWORD,h=$PROD_MYSQL_HOST,P=$PROD_MYSQL_PORT,D=$DUMP_DB,t=$DUMP
_TB,A=utf8mb4 --max-lag=60 --check-interval=10  --why-quit  --where "create_time
<'$THREE_MONTH_AGO'" --statistics --progress 5000 --limit 1000 --txn-size 1000  --purge
&>$WORKPATH/backup_tmp/ar_${DUMP_TB}_${DATE}.log && status=1 || status=0
    if (($status==1))
    then
        msg="$DUMP_TB pt-archiver is ok!"
        my_sentmail
        sleep 300
    else
        msg="$DUMP_TB pt-archiver has something wrong!"
        my_sentmail
        exit 110
    fi
done
```

```
for DUMP_TB in ${DUMP_TB_LIST[@]}
do
    $PT_OSC
u=$PROD_MYSQL_USER,p=$PROD_MYSQL_PASSWORD,h=$PROD_MYSQL_HOST,P=$PROD_MYSQL_PORT,D=$DUMP_DB,t=$DUMP
_TB,A=utf8mb4 --execute  --no-check-replication-filters --alter=engine=innodb
&>$WORKPATH/backup_tmp/osc_${DUMP_TB}_${DATE}.log && status=1 || status=0

    if (($status==1))
    then
        msg="$DUMP_TB pt-online-schema-change is ok!"
        my_sentmail
            sleep 300
    else
        msg="$DUMP_TB pt-online-schema-change has something wrong!"
        my_sentmail
        exit 110
    fi
done
```