MySQL5.7.16的TPCC压力测试笔记

# 一、 下载工具包

Tpcc-mysql是percona基于tpcc衍生出来的产品，专用于mysql基准测试，其源码放在bazaar(Bazaar是一个分布式的版本控制系统，采用 GPL 许可协议，可运行于 Windows、GNU/Linux、UNIX 以及 Mac OS 系统之上。Bazaar 由 Canonical 公司（Ubuntu母公司）赞助)上，因此还需要先安装bazaar客户端。

安装中科大epel源

rpm -ivh http://mirrors.ustc.edu.cn/fedora/epel/6/x86\_64/epel-release-6-8.noarch.rpm

然后就可以开始安装bzr客户端了：

yum install bzr

之后，就可以开始用bzr客户端下载tpcc-mysql源码了。

cd tmp

bzr branch lp:~percona-dev/perconatools/tpcc-mysql

# 二、编译安装

编译非常简单

cd /tmp/tpcc-mysql/src

make

然后就会在 /tmp/tpcc-mysql 下生成 tpcc 命令行工具 tpcc\_load 、 tpcc\_start

**tpcc\_load  提供初始化数据的功能**

**tpcc\_start 进行压力测试**

用法:

# ./tpcc\_load --help

tpcc\_load [server] [DB] [user] [pass] [warehouse]

server :    服务器名

DB     :    数据库名

user   :    用户名

pass   :    密码

warehouse: 仓库的数量

# ./tpcc\_start --help

tpcc\_start -h server\_host -P port -d database\_name -u mysql\_user -p mysql\_password -w warehouses -c connections -r warmup\_time -l running\_time -i report\_interval -f report\_file

介绍一下各个参数的用法

-h server\_host:       服务器名

-P port :             端口号，默认为3306

-d database\_name:     数据库名

-u mysql\_user :        用户名

-p mysql\_password :    密码

-w warehouses:        仓库的数量

-c connections :       线程数，默认为1

-r warmup\_time :       热身时间，单位:s，默认为10s ,热身是为了将数据加载到内存。

-l running\_time:      测试时间，单位:s，默认为20s

-i report\_interval     指定生成报告间隔时长

-f report\_file :       测试结果输出文件

# 三、开始加载测试数据

初始化测试库环境

cd /tmp/tpcc-mysql

mysqladmin –uroot –pjishu8cc create tpcc1000

mysql –uroot –pjishu8cc tpcc1000 < create\_table.sql

初始化完毕后，就可以开始加载测试数据了

./tpcc\_load localhost tpcc1000 root jishu8cc 1000

注意

 tpcc 默认会读取/var/lib/mysql/mysql.sock 这个socket位置，如果你的测试环境的mysql socket不在相应路径的话，就需要做个软连接，或者通过TCP/IP的方式连接测试服务器。

加载测试数据时长视仓库数量而定，可能会比较久。

加载测试数据的过程其实就是随机生成一堆电商订单数据。我这边直接生成1000个仓库了。

# 四、进行测试

现在我们来开启一个测试案例

由于我是4核8G的Vmware vShpere 虚拟化的主机，所以我测试用的压力不大。我用的下面这个参数:

tpcc\_start -hlocalhost -d tpcc1000 -u root -p jishu8cc -w 10 -c 64 -r 300 -l 1800 >/tmp/6G-tpcc.log

#使用tpcc\_start 进行64个线程的测试,热身时间为300秒, 测试时间为30分钟, 可根据自己需要调整

测试结果解读

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* ###easy### TPC-C Load Generator \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

option h with value '10.1.135.9'

option d with value 'tpcc1000'

option u with value 'root'

option p with value 'jishu8cc'

option w with value '10'

option c with value '64'

option r with value '300'

option l with value '1800'

<Parameters>

[server]: 10.1.135.9

[port]: 3306

[DBname]: tpcc1000

[user]: root

[pass]: jishu8cc

[warehouse]: 10

[connection]: 64

[rampup]: 300 (sec.)

[measure]: 1800 (sec.)

RAMP-UP TIME.(300 sec.)

MEASURING START.

10, 2405(0):1.936|3.378, 2409(0):0.430|1.258, 241(0):0.221|0.626, 239(0):2.515|3.066, 240(0):6.333|7.936

20, 2417(0):1.859|2.294, 2418(0):0.409|0.509, 242(0):0.212|0.272, 243(0):2.180|2.416, 242(0):5.515|5.727

30, 2420(0):1.862|2.215, 2409(0):0.406|0.478, 240(0):0.204|0.264, 242(0):2.202|2.272, 240(0):6.063|6.365

... ...

...

1780, 2419(0):1.744|2.124, 2415(0):0.372|0.458, 242(0):0.192|0.206, 240(0):2.056|2.224, 242(0):5.523|6.165

1790, 2353(0):1.706|2.005, 2351(0):0.389|0.496, 234(0):0.198|0.256, 235(0):2.226|2.412, 236(0):5.714|6.335

1800, 2370(0):1.864|2.231, 2375(0):0.386|0.487, 237(0):0.195|0.200, 237(0):2.257|2.531, 235(0):5.723|6.209

STOPPING THREADS................................................................

<Raw Results>

[0] sc:425277 lt:0 rt:0 fl:0

[1] sc:425313 lt:0 rt:0 fl:0

[2] sc:42530 lt:0 rt:0 fl:0

[3] sc:42533 lt:0 rt:0 fl:0

[4] sc:42532 lt:0 rt:0 fl:0

in 1800 sec.

<Raw Results2(sum ver.)>

[0] sc:425334 lt:0 rt:0 fl:0

[1] sc:425339 lt:0 rt:0 fl:0

[2] sc:42532 lt:0 rt:0 fl:0

[3] sc:42533 lt:0 rt:0 fl:0

[4] sc:42532 lt:0 rt:0 fl:0

<Constraint Check> (all must be [OK])

[transaction percentage]

Payment: 43.48% (>=43.0%) [OK]

Order-Status: 4.35% (>= 4.0%) [OK]

Delivery: 4.35% (>= 4.0%) [OK]

Stock-Level: 4.35% (>= 4.0%) [OK]

[response time (at least 90% passed)]

New-Order: 100.00% [OK]

Payment: 100.00% [OK]

Order-Status: 100.00% [OK]

Delivery: 100.00% [OK]

Stock-Level: 100.00% [OK]

<TpmC>

14175.900 TpmC

测试结果是服务器的每分钟事务数是14175

# 五、Troubleshoot

## 1. 执行tpcc\_load生成和载入测试数据时，报如下错误

./tpcc\_load: error while loading shared libraries: libmysqlclient.so.20: cannot open shared object file: No such file or directory

**解决方法:**

创建软链接可以解决这个问题。因为程序默认编译后找的lib库地址是/usr/lib64/

ln -s /opt/mysql/mysql-5.7.16-linux-glibc2.5-x86\_64/lib/libmysqlclient.so.20 /usr/lib64/libmysqlclient.so.20

## 2.执行tpcc\_load生成和载入测试数据时，报如下错误

1197, HY000, Multi-statement transaction required more than 'max\_binlog\_cache\_size' bytes of storage; increase this mysqld variable and try again

Retrying ...

.................................................. 5000

.................................................. 10000

...............................

**解决办法:**

root@localhost [(none)]>show variables like '%binlog\_cache\_size%';

+-----------------------+---------+

| Variable\_name | Value |

+-----------------------+---------+

| binlog\_cache\_size | 1048576 |

| max\_binlog\_cache\_size | 1048576 |

+-----------------------+---------+

2 rows in set (0.01 sec)

可见默认的max\_binlog\_cache\_size太小了，只有1MB，调大可以解决。反正当时把下面3个参数都调大了

binlog\_cache\_size = 1G

max\_binlog\_size = 1G

max\_binlog\_cache\_size = 1G

## 3.两次压测之间要重启服务器吗？

答: 可以选择重启服务器，也可以选择重启mysql并清除linux缓存：

service mysql restart

echo 3 > /proc/sys/vm/drop\_caches

# 六、最后:

性能测试、压力测试的目的:

\* 采购服务器，我们可能需要测试不同组合配置下的数据库性能，选取一个性价比更好的方案.

\* 对比不同系统参数/数据库参数 配置下的数据库性能

\* 对比不同的数据库产品

\* 对比数据库不同版本的差异

\* 一些新特性的试用，验证

\* 一些patch的验证

\* 对不不同的OS/文件系统/库的差异。

我性能测试的目的是调优，下次我将研究和尝试最大程度地榨取机器价值，提高Tpmc