# tpcc-mysql安装使用详解

**什么是TPC-C**

TPC-C是专门针对联机交易处理系统（OLTP系统场景下）的规范，一般情况我们也把这类系统称为业务处理系统。

TPC-C是TPC(Transaction Processing Performance Council)组织发布的一个测试规范，用于模拟测试复杂的在线事务处理系统。其测试结果包括每分钟事务数(tpmC)，以及每事务的成本(Price/tpmC)。

**什么是TPCC-MYSQL**

TPCC-MYSQL是percona基于TPC-C(下面简写成TPCC)衍生出来的产品，专用于MySQL基准测试。用来测试数据库的压力工具，模拟一个电商的业务，主要的业务有新增订单，库存查询，发货，支付等模块的测试。

-- **下载tpcc-mysql package**

[root@192 ~]# git clone https://github.com/Percona-Lab/tpcc-mysql

正克隆到 'tpcc-mysql'...

remote: Enumerating objects: 392, done.

remote: Total 392 (delta 0), reused 0 (delta 0), pack-reused 392

接收对象中: 100% (392/392), 202.81 KiB | 6.00 KiB/s, done.

处理 delta 中: 100% (216/216), done.

-- **编译安装**

[root@192 src]# cd tpcc-mysql/src

[root@192 src]# make

cc -w -O3 -g -I. `mysql\_config --include` -c load.c

cc -w -O3 -g -I. `mysql\_config --include` -c support.c

cc load.o support.o `mysql\_config --libs\_r` -lrt -o ../tpcc\_load

cc -w -O3 -g -I. `mysql\_config --include` -c main.c

cc -w -O3 -g -I. `mysql\_config --include` -c spt\_proc.c

cc -w -O3 -g -I. `mysql\_config --include` -c driver.c

cc -w -O3 -g -I. `mysql\_config --include` -c sequence.c

cc -w -O3 -g -I. `mysql\_config --include` -c rthist.c

cc -w -O3 -g -I. `mysql\_config --include` -c sb\_percentile.c

cc -w -O3 -g -I. `mysql\_config --include` -c neword.c

cc -w -O3 -g -I. `mysql\_config --include` -c payment.c

cc -w -O3 -g -I. `mysql\_config --include` -c ordstat.c

cc -w -O3 -g -I. `mysql\_config --include` -c delivery.c

cc -w -O3 -g -I. `mysql\_config --include` -c slev.c

cc main.o spt\_proc.o driver.o support.o sequence.o rthist.o sb\_percentile.o neword.o payment.o ordstat.o delivery.o slev.o `mysql\_config --libs\_r` -lrt -o ../tpcc\_start

[root@192 src]# echo $?

0

[root@192 tpcc-mysql]# ls -l

-rw-r--r--. 1 root root 1621 6月 15 17:07 add\_fkey\_idx.sql

-rw-r--r--. 1 root root 317 6月 15 17:07 count.sql

-rw-r--r--. 1 root root 3105 6月 15 17:07 create\_table.sql

-rw-r--r--. 1 root root 194 6月 15 17:07 Dockerfile

-rw-r--r--. 1 root root 763 6月 15 17:07 drop\_cons.sql

-rw-r--r--. 1 root root 1079 6月 15 17:07 load\_multi\_schema.sh

-rw-r--r--. 1 root root 573 6月 15 17:07 load.sh

-rw-r--r--. 1 root root 2302 6月 15 17:07 README.md

drwxr-xr-x. 2 root root 92 6月 15 17:07 schema2

drwxr-xr-x. 5 root root 4096 6月 15 17:07 scripts

drwxr-xr-x. 2 root root 4096 6月 15 17:24 src

-rwxr-xr-x. 1 root root 80928 6月 15 17:24 tpcc\_load

-rwxr-xr-x. 1 root root 188440 6月 15 17:24 tpcc\_start

-- 创建新库，并导入表及索引

[root@192 tpcc-mysql]# mysql -uroot -p -S /tmp/mysql\_3307.sock -e "create database tpcc"

Enter password:

[root@192 tpcc-mysql]# mysql -uroot -p -S /tmp/mysql\_3307.sock --default-character-set=utf8 tpcc <./create\_table.sql

Enter password:

[root@192 tpcc-mysql]# mysql -uroot -p -S /tmp/mysql\_3307.sock --default-character-set=utf8 tpcc <./add\_fkey\_idx.sql

Enter password:

**-- 查看库表**

mysql> show tables from tpcc;

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

| Tables\_in\_tpcc |

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

| customer |

| district |

| history | **-- 支付，主要对应orders、history表**

| item |

| new\_orders | **-- 新订单，一次完整的订单事务，几乎涉及到全部表**

| order\_line | **-- 发货，主要对应order\_line表**

| orders | **-- 支付，主要对应orders、history表**

| stock | **-- 库存，主要对应stock表**

| warehouse |

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

9 rows in set (0.00 sec)

**-- 单进程加载数据（不推荐）**

[root@192 tpcc-mysql]# ./tpcc\_load -h 127.0.0.1 -P 3307 -d tpcc -u root -p 123456 -w 10

**-- 并发加载数据（推荐）**

**-- 编辑脚本，根据实际情况修改。**

[root@192 tpcc-mysql]# cat load.sh

#export LD\_LIBRARY\_PATH=/usr/local/mysql/lib/mysql/

DBNAME=$1

WH=$2

HOST=127.0.0.1

PORT=3307

PASSWD="123456"

STEP=100

./tpcc\_load -h $HOST -d $DBNAME -u root -p $PASSWD -P $PORT -w $WH -l 1 -m 1 -n $WH >> 1.out &

x=1

while [ $x -le $WH ]

do

echo $x $(( $x + $STEP - 1 ))

./tpcc\_load -h $HOST -d $DBNAME -u root -p $PASSWD -P $PORT -w $WH -l 2 -m $x -n $(( $x + $STEP - 1 )) >> 2\_$x.out &

./tpcc\_load -h $HOST -d $DBNAME -u root -p $PASSWD -P $PORT -w $WH -l 3 -m $x -n $(( $x + $STEP - 1 )) >> 3\_$x.out &

./tpcc\_load -h $HOST -d $DBNAME -u root -p $PASSWD -P $PORT -w $WH -l 4 -m $x -n $(( $x + $STEP - 1 )) >> 4\_$x.out &

x=$(( $x + $STEP ))

done

[root@192 tpcc-mysql]# chmod +x load.sh

[root@192 tpcc-mysql]# ./load.sh tpcc 10

**-- 可查看进程**

[root@192 tpcc-mysql]# ps -ef|grep load |grep -v grep

root 3613 1 13 14:34 pts/2 00:00:07 ./tpcc\_load -h 127.0.0.1 -d tpcc -u root -p 123456 -P 3307 -w 10 -l 2 -m 1 -n 100

root 3614 1 7 14:34 pts/2 00:00:03 ./tpcc\_load -h 127.0.0.1 -d tpcc -u root -p 123456 -P 3307 -w 10 -l 3 -m 1 -n 100

root 3615 1 10 14:34 pts/2 00:00:06 ./tpcc\_load -h 127.0.0.1 -d tpcc -u root -p 123456 -P 3307 -w 10 -l 4 -m 1 -n 100

-- 完成后可查看输出文件

[root@192 tpcc-mysql]# tail -10 1.out

.................................................. 70000

.................................................. 75000

.................................................. 80000

.................................................. 85000

.................................................. 90000

.................................................. 95000

.................................................. 100000

Item Done.

...DATA LOADING COMPLETED SUCCESSFULLY.

**-- 真实测试场景中，建议预热时间不小于5分钟，持续压测时长不小于30分钟，每次测试结束后，待服务冷却。否则测试数据不具参考意义。**

**-- 模拟对10个仓库(-w 10),并发10个线程(-c 10),预热120s(-r 120),持续压测300s(-l 300)**

[root@192 tpcc-mysql]# ./tpcc\_start -h127.0.0.1 -P3307 -d TPCC -u root -p 123456 -w 10 -c 10 -r 120 -l 300 >> mysql\_tpcc\_20200617

[root@192 tpcc-mysql]# cat mysql\_tpcc\_20200617

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

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

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

option h with value '127.0.0.1'

option P with value '3307'

option d with value 'TPCC'

option u with value 'root'

option p with value '123456'

option w with value '10'

option c with value '10'

option r with value '120'

option l with value '300'

<Parameters>

[server]: 127.0.0.1

[port]: 3307

[DBname]: TPCC

[user]: root

[pass]: 123456

[warehouse]: 10

[connection]: 10

[rampup]: 120 (sec.)

[measure]: 300 (sec.)

RAMP-UP TIME.(120 sec.)

MEASURING START.

10, trx: 14, 95%: 4330.384, 99%: 6080.620, max\_rt: 6721.410, 12|4676.591, 1|1921.270, 0|15125.534, 1|34453.575

20, trx: 8, 95%: 2941.472, 99%: 2941.472, max\_rt: 2941.622, 12|3473.683, 1|1685.923, 1|20589.272, 2|33295.996

30, trx: 16, 95%: 5657.408, 99%: 6514.006, max\_rt: 6513.696, 11|2545.402, 1|868.354, 2|14683.738, 1|964.354

40, trx: 13, 95%: 2697.704, 99%: 3994.165, max\_rt: 3993.747, 16|2261.064, 2|1414.305, 2|11710.987, 2|27822.000

50, trx: 15, 95%: 4862.245, 99%: 4862.245, max\_rt: 4862.759, 14|4858.395, 1|1457.624, 0|0.000, 1|29468.685

60, trx: 14, 95%: 3673.040, 99%: 4114.302, max\_rt: 4114.063, 13|1745.711, 2|945.420, 2|14419.672, 1|231.804

70, trx: 15, 95%: 3847.475, 99%: 4451.296, max\_rt: 4450.881, 15|3605.003, 1|563.852, 1|13478.381, 2|28016.795

80, trx: 15, 95%: 2778.846, 99%: 3234.261, max\_rt: 3234.317, 16|1939.732, 2|356.616, 2|13358.957, 1|24505.221

90, trx: 16, 95%: 2386.128, 99%: 2754.003, max\_rt: 2753.697, 17|2199.863, 2|246.034, 2|12287.408, 3|22663.095

100, trx: 23, 95%: 2943.233, 99%: 3533.910, max\_rt: 3534.003, 25|3148.268, 2|1017.603, 1|10900.375, 3|21750.349

110, trx: 22, 95%: 2824.974, 99%: 3324.568, max\_rt: 3324.320, 17|1442.503, 2|640.927, 3|16588.578, 0|0.000

120, trx: 15, 95%: 2717.968, 99%: 2778.846, max\_rt: 2778.456, 15|1698.667, 1|302.327, 1|10946.037, 2|19905.533

130, trx: 15, 95%: 2570.006, 99%: 4004.940, max\_rt: 4005.489, 17|2610.050, 2|371.687, 2|13611.135, 1|12694.149

140, trx: 12, 95%: 3313.639, 99%: 4087.296, max\_rt: 4086.884, 11|2881.947, 1|371.333, 1|12677.826, 0|0.000

150, trx: 12, 95%: 4020.555, 99%: 4200.164, max\_rt: 4200.680, 12|3301.154, 1|1106.468, 2|15236.465, 3|27515.548

160, trx: 16, 95%: 2912.558, 99%: 3144.524, max\_rt: 3144.099, 17|2343.057, 2|268.451, 1|11810.461, 3|27571.882

170, trx: 20, 95%: 3162.460, 99%: 3588.274, max\_rt: 3588.524, 19|3523.185, 2|406.008, 2|10953.483, 1|13597.831

180, trx: 15, 95%: 2633.081, 99%: 3657.679, max\_rt: 3657.380, 14|2174.835, 2|868.959, 1|13376.481, 1|16321.603

190, trx: 18, 95%: 2696.090, 99%: 3441.006, max\_rt: 3441.414, 16|2216.698, 1|188.947, 2|12837.649, 2|18408.393

200, trx: 14, 95%: 3275.180, 99%: 3365.621, max\_rt: 3365.960, 15|2433.380, 2|361.566, 2|12258.444, 1|19218.267

210, trx: 20, 95%: 2492.723, 99%: 3055.448, max\_rt: 3055.156, 21|2554.645, 1|337.661, 2|12076.689, 4|17146.398

220, trx: 26, 95%: 2971.562, 99%: 4320.026, max\_rt: 4320.531, 24|1836.381, 3|674.214, 2|9370.373, 2|16295.460

230, trx: 27, 95%: 1701.854, 99%: 1848.429, max\_rt: 1848.164, 26|1754.689, 3|113.323, 2|8506.203, 2|13209.192

240, trx: 23, 95%: 2196.916, 99%: 2590.086, max\_rt: 2590.346, 26|1878.178, 2|574.608, 3|9012.565, 2|9878.651

250, trx: 21, 95%: 2152.005, 99%: 2493.469, max\_rt: 2493.661, 22|2099.949, 2|364.295, 2|9966.280, 4|10624.409

260, trx: 27, 95%: 2058.132, 99%: 2762.259, max\_rt: 2762.214, 27|2008.956, 3|366.781, 3|10109.748, 4|8088.025

270, trx: 38, 95%: 1456.101, 99%: 1999.830, max\_rt: 1999.971, 37|1643.105, 4|353.261, 3|8718.665, 3|229.547

280, trx: 36, 95%: 1512.975, 99%: 1732.696, max\_rt: 1732.840, 34|1750.481, 3|753.146, 4|7517.902, 3|258.714

290, trx: 23, 95%: 2277.947, 99%: 2310.222, max\_rt: 2309.961, 25|2062.126, 3|1101.806, 2|10218.299, 2|8919.934

300, trx: 22, 95%: 2271.138, 99%: 2404.771, max\_rt: 2404.726, 25|2060.381, 1|266.090, 3|10437.174, 1|5451.916

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

<Raw Results>

[0] sc:0 lt:571 rt:0 fl:0 avg\_rt: 2645.4 (5)

[1] sc:0 lt:571 rt:0 fl:0 avg\_rt: 1549.1 (5)

[2] sc:2 lt:54 rt:0 fl:0 avg\_rt: 695.0 (5)

[3] sc:0 lt:56 rt:0 fl:0 avg\_rt: 14643.1 (80)

[4] sc:2 lt:56 rt:0 fl:0 avg\_rt: 16391.7 (20)

in 300 sec.

<Raw Results2(sum ver.)>

[0] sc:0 lt:571 rt:0 fl:0

[1] sc:0 lt:573 rt:0 fl:0

[2] sc:2 lt:54 rt:0 fl:0

[3] sc:0 lt:56 rt:0 fl:0

[4] sc:2 lt:56 rt:0 fl:0

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

[transaction percentage]

Payment: 43.52% (>=43.0%) [OK]

Order-Status: 4.27% (>= 4.0%) [OK]

Delivery: 4.27% (>= 4.0%) [OK]

Stock-Level: 4.42% (>= 4.0%) [OK]

[response time (at least 90% passed)]

New-Order: 0.00% [NG] \*

Payment: 0.00% [NG] \*

Order-Status: 3.57% [NG] \*

Delivery: 0.00% [NG] \*

Stock-Level: 3.45% [NG] \*

<TpmC>

**114.200 TpmC -- 重要指标**

**-- 分析测试结果**

[root@192 scripts]# sh analyze.sh ../mysql\_tpcc\_20200617 >/tmp/mysql\_tpcc\_20200617.res

[root@192 scripts]# cat /tmp/mysql\_tpcc\_20200617.res

0 4330.384000

0 2941.472000

0 5657.408000

0 2697.704000

0 4862.245000

0 3673.040000

0 3847.475000

0 2778.846000

0 2386.128000

0 2943.233000

0 2824.974000

0 2717.968000

0 2570.006000

0 3313.639000

0 4020.555000

0 2912.558000

0 3162.460000

0 2633.081000

0 2696.090000

0 3275.180000

0 2492.723000

0 2971.562000

0 1701.854000

0 2196.916000

0 2152.005000

0 2058.132000

0 1456.101000

0 1512.975000

0 2277.947000

0 2271.138000

**-- 下面模拟MySQL不同配置参数对性能（TPS）的影响 （仅测试innodb\_flush\_log\_at\_trx\_commit 和 sync\_binlog 在0|1的不同)**

**-- 安装绘图软件**

[root@192 tpcc-mysql]# yum install gnuplot

**-- 编辑分析脚本**

[root@192 scripts]# cat tpcc\_analyze.sh

#!/bin/bash

TIMESLOT=1

if [ -n "$2" ]

then

TIMESLOT=$2

echo "Defined $2"

fi

cat $1 | grep -v HY000 | grep -v payment | grep -v neword | \

awk -v timeslot=$TIMESLOT ' BEGIN { FS="[,():]"; s=0; cntr=0; aggr=0 } \

/MEASURING START/ { s=1} /STOPPING THREADS/ {s=0} /0/ { if (s==1) { cntr++; aggr+=$2; } \

if ( cntr==timeslot ) { printf ("%d %3f\n",$1,$5) ; cntr=0; aggr=0 } } '

**-- 编辑绘图脚本**

[root@192 scripts]# cat tpcc\_graph.sh

#!/bin/bash

gnuplot << EOP

set style line 1 lt 1 lw 3

set style line 2 lt 5 lw 3

set style line 3 lt 7 lw 3

set terminal png size 960,480

set grid x y

set xlabel "Time(sec)"

set ylabel "Transactions"

set output "$2"

plot "$1" using 1:2 title "MySQL 5.7.28 tc:0-sb:0" ls 1 with lines,\

"$1" using 3:4 title "MySQL 5.7.28 tc:0-sb:1" ls 2 with lines,\

"$1" using 5:6 title "MySQL 5.7.28 tc:1-sb:0" ls 3 with lines,\

"$1" using 7:8 title "MySQL 5.7.28 tc:1-sb:1" ls 3 with lines axes x1y1

EOP

**-- 下面的每次测试均需要修改MySQL配置文件（生效）后，进行压测。**

./tpcc\_start -h127.0.0.1 -P3307 -d TPCC -u root -p 123456 -w 10 -c 10 -r 120 -l 180 >> mysql\_tpcc\_tc1\_sb1\_20200617

./tpcc\_start -h127.0.0.1 -P3307 -d TPCC -u root -p 123456 -w 10 -c 10 -r 120 -l 180 >> mysql\_tpcc\_tc1\_sb0\_20200617

./tpcc\_start -h127.0.0.1 -P3307 -d TPCC -u root -p 123456 -w 10 -c 10 -r 120 -l 180 >> mysql\_tpcc\_tc0\_sb1\_20200617

./tpcc\_start -h127.0.0.1 -P3307 -d TPCC -u root -p 123456 -w 10 -c 10 -r 120 -l 180 >> mysql\_tpcc\_tc0\_sb0\_20200617

**-- 生成绘图数据**

[root@192 tpcc-mysql]# cd scripts/

[root@192 tpcc-mysql]# sh tpcc\_analyze.sh ../mysql\_tpcc\_tc1\_sb1\_20200617 >/tmp/mysql\_tpcc\_tc1\_sb1\_20200617.data

[root@192 tpcc-mysql]# sh tpcc\_analyze.sh ../mysql\_tpcc\_tc1\_sb0\_20200617 >/tmp/mysql\_tpcc\_tc1\_sb0\_20200617.data

[root@192 tpcc-mysql]# sh tpcc\_analyze.sh ../mysql\_tpcc\_tc0\_sb1\_20200617 >/tmp/mysql\_tpcc\_tc0\_sb1\_20200617.data

[root@192 tpcc-mysql]# sh tpcc\_analyze.sh ../mysql\_tpcc\_tc0\_sb0\_20200617 >/tmp/mysql\_tpcc\_tc0\_sb0\_20200617.data

**-- 合并四组数据**

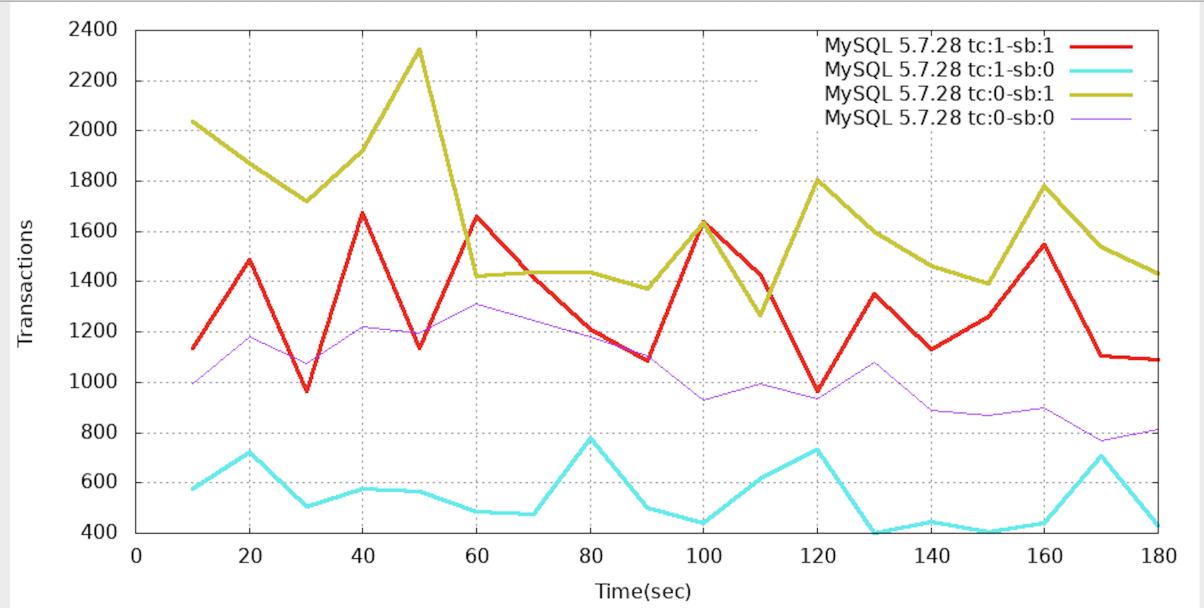
paste /tmp/mysql\_tpcc\_tc1\_sb1\_20200617.data /tmp/mysql\_tpcc\_tc1\_sb0\_20200617.data /tmp/mysql\_tpcc\_tc0\_sb1\_20200617.data /tmp/mysql\_tpcc\_tc0\_sb0\_20200617.data >/tmp/tpcc-merge.data

**-- 生成图片**

[root@192 scripts]# sh tpcc\_graph.sh /tmp/tpcc-merge.data tpcc\_merge.png

[root@192 scripts]# sz -be tpcc\_merge.png

**-- 查看图片**



**总结：**

**1.tpcc-mysql 可针对服务器的硬件配置，数据库的参数配置对比性能表现。**

**2.由于测试原因，压测过程中，服务缺少预热和冷却时间。测试过程和结果仅作为可行性演示，结果不具备参考意义。**