# 简介

Sysbench是一款开源的多线程性能测试工具，可以执行CPU/内存/线程/IO/数据库等方面的性能测试。

Sysbench支持以下几种测试模式：

1. CPU运算性能
2. 磁盘IO性能
3. 调度程序性能
4. 内存分配及传输速度
5. POSIX线程性能
6. 数据库性能（OLTP基准测试）

注：在GoldenDB分布式数据库中，使用sysbench的数据库性能（OLTP基准测试）测试模式来压测proxy性能。

参考：<https://blog.csdn.net/u012423685/article/details/108617102>

# 安装

参考：

[MySQL benchmark](https://brinnatt.com/projects/mysql_senior/%E7%AC%AC-d-%E7%AB%A0-mysql-benchmark/)

基本操作步骤：

1. 获取安装包sysbench-1.0.tar.gz

https://github.com/akopytov/sysbench

1. 解压：unzip sysbench-1.0.tar.gz
2. 进入解压后的目录：cd /home/sysbench-1.0/
3. 安装依赖：

yum -y install make automake libtool pkgconfig libaio-devel

yum -y install mariadb-devel openssl-devel

1. 编译：

./autogen.sh

./configure

make -j

make install

1. 创建自定义用户

useradd username

cp -r /home/sysbench-1.0 /home/username

chown

7、创建对应的库（即sysbench --mysql-db指定的库名）

# 测试模型

Sysbench通过脚本定义了若干个常用的压测模型，以下是几个常用模型：

|  |  |
| --- | --- |
| 压测模型 | 描述 |
| bulk\_insert.lua | 批量插入数据 |
| oltp\_delete.lua | 删除数据 |
| oltp\_insert.lua | 单值插入数据 |
| oltp\_point\_select.lua | 精准查询 |
| oltp\_read\_only.lua | 只读 |
| oltp\_read\_write.lua | 混合读写 |
| oltp\_update\_index.lua | 更新索引字段 |
| oltp\_update\_non\_index.lua | 更新非索引字段 |
| oltp\_write\_only.lua | 只写 |

# 使用

使用sysbench进行压测，通常分为三个步骤：

1. prepare：准备数据
2. run：运行测试模型
3. cleanup：清理测试数据

通常仅需要准备一次数据，在此数据基础上测试各种模型即可。

## 常用配置参数

Usage:

sysbench [options]... [testname] [command]

Commands implemented by most tests: prepare run cleanup help

General options:

--threads=N     number of threads to use [1]  # 创建测试线程的数量，默认是1

--events=N    limit for total number of events [0]   # 事件最大数量，0为不限制

--time=N limit for total execution time in seconds [10] # 最大执行时间，单位秒

--forced-shutdown=STRING     number of seconds to wait after the --time limit before forcing shutdown, or 'off' to disable [off]

--thread-stack-size=SIZE        size of stack per thread [64K]

--rate=N                        average transactions rate. 0 for unlimited rate [0]

--report-interval=N             periodically report intermediate statistics with a specified interval in seconds. 0 disables intermediate reports [0]

--report-checkpoints=[LIST,...] dump full statistics and reset all counters at specified points in time. The argument is a list of comma-separated values representing the amount of time in seconds elapsed from start of test when report checkpoint(s) must be performed. Report checkpoints are off by default. []

--debug[=on|off]                print more debugging info [off]

--validate[=on|off]             perform validation checks where possible [off]

--help[=on|off]                 print help and exit [off]

--version[=on|off]              print version and exit [off]

--config-file=FILENAME          File containing command line options

--tx-rate=N                     deprecated alias for --rate [0]

--max-requests=N                deprecated alias for --events [0]

--max-time=N                    deprecated alias for --time [0]

--num-threads=N                 deprecated alias for --threads [1]

Pseudo-Random Numbers Generator options:

--rand-type=STRING random numbers distribution {uniform,gaussian,special,pareto} [special]

--rand-spec-iter=N number of iterations used for numbers generation [12]

--rand-spec-pct=N  percentage of values to be treated as 'special' (for special distribution) [1]

--rand-spec-res=N  percentage of 'special' values to use (for special distribution) [75]

--rand-seed=N      seed for random number generator. When 0, the current time is used as a RNG seed. [0]

--rand-pareto-h=N  parameter h for pareto distribution [0.2]

Log options:

--verbosity=N verbosity level {5 - debug, 0 - only critical messages} [3]

--percentile=N       percentile to calculate in latency statistics (1-100). Use the special value of 0 to disable percentile calculations [95]

--histogram[=on|off] print latency histogram in report [off]

General database options:

--db-driver=STRING  specifies database driver to use ('help' to get list of available drivers) [mysql]

--db-ps-mode=STRING prepared statements usage mode {auto, disable} [auto]

--db-debug[=on|off] print database-specific debug information [off]

Compiled-in database drivers:

mysql - MySQL driver

mysql options:

--mysql-host=[LIST,...]          MySQL server host [localhost]

--mysql-port=[LIST,...]          MySQL server port [3306]

--mysql-socket=[LIST,...]        MySQL socket

--mysql-user=STRING              MySQL user [sbtest]

--mysql-password=STRING          MySQL password []

--mysql-db=STRING                MySQL database name [sbtest]

--mysql-ssl[=on|off]    use SSL connections, if available in the client library [off]

--mysql-ssl-cipher=STRING        use specific cipher for SSL connections []

--mysql-compression[=on|off]     use compression, if available in the client library [off]

--mysql-debug[=on|off]           trace all client library calls [off]

--mysql-ignore-errors=[LIST,...] list of errors to ignore, or "all" [1213,1020,1205]

--mysql-dry-run[=on|off]         Dry run, pretend that all MySQL client API calls are successful without executing them [off]

Compiled-in tests:

fileio - File I/O test

cpu - CPU performance test

memory - Memory functions speed test

threads - Threads subsystem performance test

mutex - Mutex performance test

## 装载数据相关参数

--mysql-host：proxy或者db的ip地址

--mysql-port： proxy或者db的port

--mysql-user：数据库用户名

--mysql-password：数据库密码

--mysql-db：database name

--tables ：装载表的个数

--table-size：每张表的行数

--db-driver：指定db driver类型为mysql（pg，mysql……）

--threads：用户连接数（客户端并发连接数）

noshard例子（oltp.lua）：在sysbench\_1数据库下创建10张表，每张表1W行

./sysbench ./src/lua/oltp.lua --mysql-host=xxxx --mysql-port=15151 --mysql-user=test1234 --mysql-password=test1234 --mysql-db=sysbench\_1 --tables=10 --table-size=10000 --db-driver=mysql --report-interval=10 --threads=10 --time=120 prepare

groupshard例子（oltp\_common.lua）：在sysbench\_1数据库下创建10张shard表，每张表1W行

./sysbench ./src/lua/oltp\_common.lua --mysql-host=xxxx--mysql-port=15584 --mysql-user=test1234 --mysql-password=test1234 --mysql-db=sysbench\_1 --tables=10 --table-size=10000 --db-driver=mysql --report-interval=10 --threads=10 --time=120 prepare

备注：groupshard，修改lua脚本在185行后面增加extra\_table\_options = extra\_table\_options .. " shardkey=id"

## 压测场景

--range\_size ：查询范围，默认值100行

--point\_selects =每个事务包含point select的个数，默认值1

--simple\_ranges ：每个事务包含range select的个数，默认值1

--sum\_ranges =每个事务包含sum() select的个数，默认值1

--order\_ranges =每个事务包含order by select的个数，默认值1

--distinct\_ranges =每个事务包含distinct select的个数，默认值1

--index\_updates =每个事务包含index select的个数，默认值1

--non\_index\_updates =每个事务包含更新非索引字段的个数，默认值1

--delete\_inserts =每个事务包含delete和insert的个数，默认值1

--range\_selects =开启或者关闭范围查询，默认值false

--time  运行时间设为0表示不限制时间  
--report-interval   运行期间日志，单位为秒

--events    最大请求数量，定义数量后可以不需要--time选项

写好上面的语句  
在上面的语句后面加上 prepare，执行  
在上面的语句后面加上 run，执行  
在上面的语句后面加上 cleanup，执行

prepare用于准备测试需要的数据，准备完后执行run来测试，测试完成后不要忘记执行cleanup来清除测试数据

混合读写场景

./sysbench --threads=20 ./src/lua/oltp\_read\_write.lua --table-size=10000 --tables=10 --point\_selects=2 --range\_selects=2 --index\_updates=2 --non\_index\_updates=1 --delete\_inserts=1 --report-interval=1 --mysql-host=xxxx --mysql-port=8080 --mysql-user=test1234 --mysql-password=test1234 --mysql-db=snow\_177 --time=10 --max-requests=0 run

普通查询

./sysbench --threads=20 ./src/lua/oltp\_point\_select.lua --table-size=10000 --tables=10 --report-interval=1 --mysql-host=xxxx --mysql-port=8080 --mysql-user=test1234 --mysql-password=test1234 --mysql-db=snow\_177 --time=10 --max-requests=0 run

仅查询（包含范围查询、sum、order by和distinct）

./sysbench --threads=20 ./src/lua/oltp\_read\_only.lua --table-size=10000 --tables=10 --report-interval=1 --mysql-host=xxxx --mysql-port=15151 --mysql-user=test1234 --mysql-password=test1234 --mysql-db=snow\_177 --time=10 --max-requests=0 run

索引更新

./sysbench --threads=20 ./src/lua/oltp\_update\_index.lua --table-size=10000 --tables=10 --report-interval=1 --mysql-host=xxxx --mysql-port=15151 --mysql-user=test1234 --mysql-password=test1234 --mysql-db=snow\_177 --time=10 --max-requests=0 run

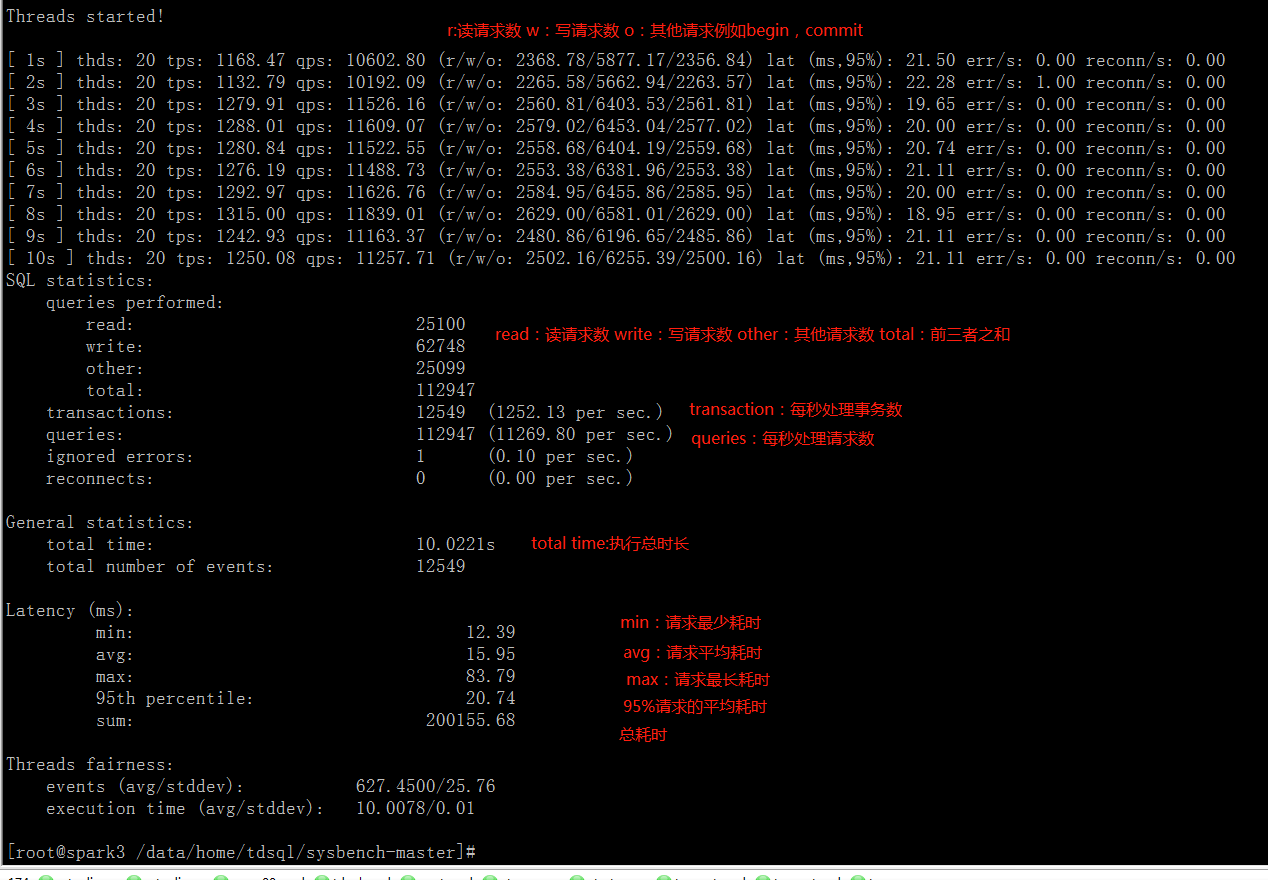
非索引更新

./sysbench --threads=20 ./src/lua/oltp\_update\_non\_index.lua --table-size=10000 --tables=10 --report-interval=1 --mysql-host=xxxx--mysql-port=15151 --mysql-user=test1234 --mysql-password=test1234 --mysql-db=snow\_177 --time=10 --max-requests=0 run

仅写(索引更新，非索引更新，delete和insert)

./sysbench --threads=20 ./src/lua/oltp\_write\_only.lua --table-size=10000 --tables=10 --point\_selects=2 --range\_selects=2 --index\_updates=2 --non\_index\_updates=1 --delete\_inserts=1 --report-interval=1 --mysql-host=xxxx --mysql-port=8080 --mysql-user=test1234 --mysql-password=test1234 --mysql-db=snow\_177 --time=10 --max-requests=0 run

压测结果：



read: 2368.78 //总select数量

write: 5877.17 //总写请求数update、insert、delete语句数量

other: 2356.84 //begin、commit、unlock tables以及其他mutex的数量

total: 10602.80 //总请求数（ r+w+o的总和）

transactions: 12549 总事务数 (1252.13 per sec.) //每秒处理完成事务数(TPS)

requests: 112947总请求数 (11269.80 per sec.) //每秒处理的请求数（qps）

other operations: 571864 ( 4764.21 per sec.)

ignored errors: 0 (0.00 per sec.) //每秒忽略的错误数

reconnects: 0 (0.00 per sec.) //每秒重连数据

General statistics:

total time: 120.0334s //即-times指定的压测时间

total number of events: 285932 //总的事件数，一般与transactions相同

Latency：response time:

min: 12.39ms

avg: 15.95ms //95%的语句的平均响应时间，后面的95%的大小可以通过--percentile=98的方式去更改

max: 83.79ms

95 percentile: 20.74ms //95%的语句的平均响应时间

sum:200155.68 //总耗时

Threads fairness:

events (avg/stddev): 627.4500/25.76

execution time (avg/stddev): 10.0078/0.01

## 操作步骤

参考：<https://www.cnblogs.com/ivictor/p/16955580.html>

1. 开始测试

开始测试前，还需要先连接proxy创建sysbenchtest数据库。

注意：sysbench内部采用prepare模式测试，因此需要开启数据库prepare模式。

1. 准备数据

sysbench /usr/local/share/sysbench/oltp\_insert.lua --mysql-host=\*\* --mysql-port=\*\* --mysql-user=\*\* --mysql-password=\*\* --mysql-db=\*\* --db-driver=mysql --mysql\_table\_options=’distributed by hash(id)(g1,g2,g3,g4)’ --tables=\*\* --table-size=\*\* --report-interval=\*\* --threads=\*\* --time=\*\* --skip\_trx=true --mysql-ignore-errors=all prepare

1. 执行测试

sysbench /usr/local/share/sysbench/oltp\_insert.lua --mysql-host=\*\* --mysql-port=\*\* --mysql-user=\*\* --mysql-password=\*\* --mysql-db=sysbenchtest --db-driver=mysql --mysql\_table\_options=’distributed by hash(id)(g1,g2,g3,g4)’ --tables=\*\* --table-size=\*\* --report-interval=\*\* --threads=\*\* --time=\*\* --skip\_trx=true --mysql-ignore-errors=all run

1. 清理环境

sysbench /usr/local/share/sysbench/oltp\_insert.lua --mysql-host=\*\* --mysql-port=\*\* --mysql-user=\*\* --mysql-password=\*\* --mysql-db=sysbenchtest --db-driver=mysql --mysql\_table\_options=’distributed by hash(id)(g1,g2,g3,g4)’ --tables=\*\* --table-size=\*\* --report-interval=\*\* --threads=\*\* --time=\*\* --skip\_trx=true --mysql-ignore-errors=all cleanup