# MyCAT 性能测试指南

**首先参考MyCAT性能调优，确保整个系统达到最优配置。**

**性能测试，建议先小规模压力预热10-20分钟，这是众所周知的Java的特性，越跑越快。**

**测试的硬件和网络条件：**

* **建议至少3台服务器：**
* **MyCAT Server一台**
* **Mysql 一台**
* **带宽应该是至少100M，建议千兆**
* **压力程序在另一台，压力程序的机器也可以由性能差的机器来代替。**

**有条件的话，分片库在不同的MYSQL实例上，如20个分片，每个MYSQL实例7个分片，而且最好有多台MYSQL物理机。**

**分片表的录入性能测试-T01**

**测试案例：分片表的并发录入性能测试，测试DEMO中的travelrecord表，此表的基准DDL语句：**create travelrecord: create table travelrecord (id bigint not null primary key,user varchar(100),traveldate DATE, fee decimal,days int);

**此表的标准分片方式为基于ID范围的自动分片策略。Schema.xml中配置如下：**

**<table name="travelrecord" dataNode="dn1,dn2,dn3" rule="auto-sharding-long" />**

**默认是3个分片，分片ID范围定义在autopartition-long.txt中，建议修改为以下或更大的数值范围分片，每个分片500万数据**

# range start-end ,data node index

0-2000000=0

2000001-4000000=1

4000001-6000000=2

根据自己的情况，可以每个分片放更多的数据，进行对比性能测试，当分片index增加时，注意dataNode也增加（**dataNode="dn1,dn2,dn3"）**。

测试的输入参数如下[jdbcurl] [user] [password] [threadpoolsize] [recordrange]：

Jdbcurl:连接mycat的地址，格式为jdbc:mysql://localhost:8066/TESTDB

User 连接Mycat的用户名

Password:密码

Threadpoolsize:并发线程请求，可以在50-2000左右调整，看看哪种情况下的性能最好

Recordrang:插入的分片系列以及对应的ID范围，minId-maxId然后逗号分开，对应多组分片的ID范围，如 0-200000,200001-400000,400001-600000，**跟分片配置保持一致**。

测试过程：

**每次测试，建议先执行重建表的操作，以保证测试环境的一致性：**

连接mycat 8066端口，在命令行执行下面的操作：

drop table travelrecord;

create table travelrecord (id bigint not null primary key,user varchar(100),traveldate DATE, fee decimal,days int);

先预测试：

执行命令：

test\_stand\_insert\_perf jdbc:mysql://localhost:8066/TESTDB test test 100 “0-100M,100M1-200M,200M1-400”

**MyCAT温馨提示**：并发线程数表明同时至少有多少个Mysql连接会被打开，当SQL不跨分片的时候，并发线程数=MYSQL连接数，在Mycat conf/schema.xml中，将minCon设置为>=并发连接数，这种情况下重启MYCAT，会初始建立minCon个连接，并发测试结果更好，另外，也可以验证是否当前内存设置，以及MYSQL是否支持开启这么多连接，若无法支持，则logs/mycat.log日志中会有告警错误信息，建议测试过程中tail –f logs/mycat.log 观察有无错误信息。另外，开启单独的Mycat管理窗口，mysql –utest –ptest –P9066 然后运行 show @@datasource 可以看到后端连接的使用情况。Show @@threadpool 可以看线程和SQL任务积压的情况。

**也可以同时启动多个测试程序,在不同的机器上，并发进行测试，每个测试程序写入一个分片的数据范围，对于1个亿的数据插入测试来说，可能效果更好，毕竟单机并发线程50个左右已经差不多极限：**

* test\_stand\_insert\_perf jdbc:mysql://localhost:8066/TESTDB test test 100 “0-100M”
* test\_stand\_insert\_perf jdbc:mysql://localhost:8066/TESTDB test test 100 100M1-200M”

**全局表的查询性能测试T02：**

全局表自动在多个节点上同步插入，因此其插入性能有所降低，这里的插入表为goods表，执行的命令类似T01的测试。温馨提示：全局表是同时往多个分片上写数据，因此所需并发MYSQL数连接为普通表的3倍，最好的模式是全局表分别在多个mysql实例上。

建表语句：

drop table goods;

create table goods(id int not null primary key,name varchar(200),good\_type tinyint,good\_img\_url varchar(200),good\_created date,good\_desc varchar(500), price double);

test\_globaltable\_insert\_perf.bat jdbc:mysql://localhost:8066/TESTDB test test 100 1000000

本机笔记本，4G内存，数据库与Mycat以及测试程序都在一起，跑出来每秒1000多的插入速度：



**分片表的查询性能测试T03：**

**此测试可以在T01的集成上运行，先生成大量**travelrecord记录，然后进行并发随机查询，

此测试是在分片库上，基于分片的主键ID进行随机查询，返回单条记录，多线程并发随机执行N此记录查询，每次查询的记录主键ID是随机选择，在maxID(参数)范围之内。

测试工具test\_stand\_select\_perf的参数如下

**[jdbcurl] [user] [password] [threadpoolsize] [executetimes] [maxId]**

**Executetimes：每个线程总共执行多少次随机查询，建议1000次以上**

**maxId：**travelrecord表的最大ID，可以执行select max(id) from travelrecord来获取。

Example:

test\_stand\_select\_perf.bat jdbc:mysql://localhost:8066/TESTDB test test 100 10000 50000

**分片表的汇聚性能测试T04：**

**此测试可以在T01的集成上运行，先生成大量**travelrecord记录，然后进行并发随机查询，

此测试执行分片库上的聚合、排序、分页的性能，SQL如下：

select sum(fee) total\_fee, days,count(id),max(fee),min(fee) from travelrecord group by days order by days desc limit ？

测试工具test\_stand\_merge\_sel\_perf的参数如下

**[jdbcurl] [user] [password] [threadpoolsize] [executetimes] [limit]**

**Executetimes：每个线程总共执行多少次随机查询，建议1000次以上**

**limit：**分页返回的记录个数，必须大于30

Example:

test\_stand\_merge\_sel\_perf.bat jdbc:mysql://localhost:8066/TESTDB test test 10 100 100

**分片表的更新性能测试T05：**

**此测试可以在T01的集成上运行，先生成大量**travelrecord记录，然后进行并发更新操作，

update travelrecord set user =? ,traveldate=?,fee=?,days=? where id=?

测试工具test\_stand\_update\_perf的参数如下

**[jdbcurl] [user] [password] [threadpoolsize] [record]**

**record：总共修改多少条记录， >5000**

Example:

test\_stand\_update\_perf.bat jdbc:mysql://localhost:8066/TESTDB test test 10 10000