Mycat+percona\_cluster

MyCat 1.2 支持percona\_cluster的集群模式，即多节点同时读写，牺牲了数据插入的性能而换来高可用性和读的性能，下面是测试报告。

软件版本:

Mycat: 1.2

Percona XtraDB Cluster: 5.6.15-25.3

Wsrep :25.3.r4034

系统:

服务器:华为Tecal RH2288 V2-8S

Cpu: Intel(R) Xeon(R) CPU E5-2620

processor : 24

Mem: 24G

Disk: Scsi Raid10

架构:



测试场景

1. 正常环境下三个数据库同时写入测试
2. 登录mycat

mysql -h10.20.38.68 -utest -ptest -P8066 TESTDB

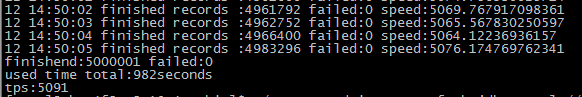
1. 创建表

create table travelrecord (id bigint not null ,user\_id varchar(100),traveldate DATETIME, fee decimal,days int);

C.1测试（400并发）

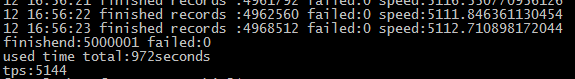
./test\_stand\_insert\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 400 0-500M

结果



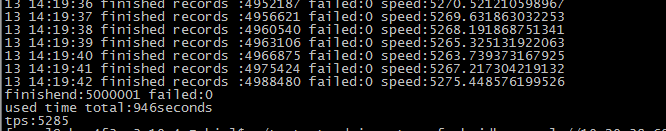
C.2 300并发

./test\_stand\_insert\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 300 0-500M



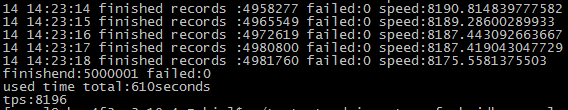
200并发

./test\_stand\_insert\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 200 0-500M



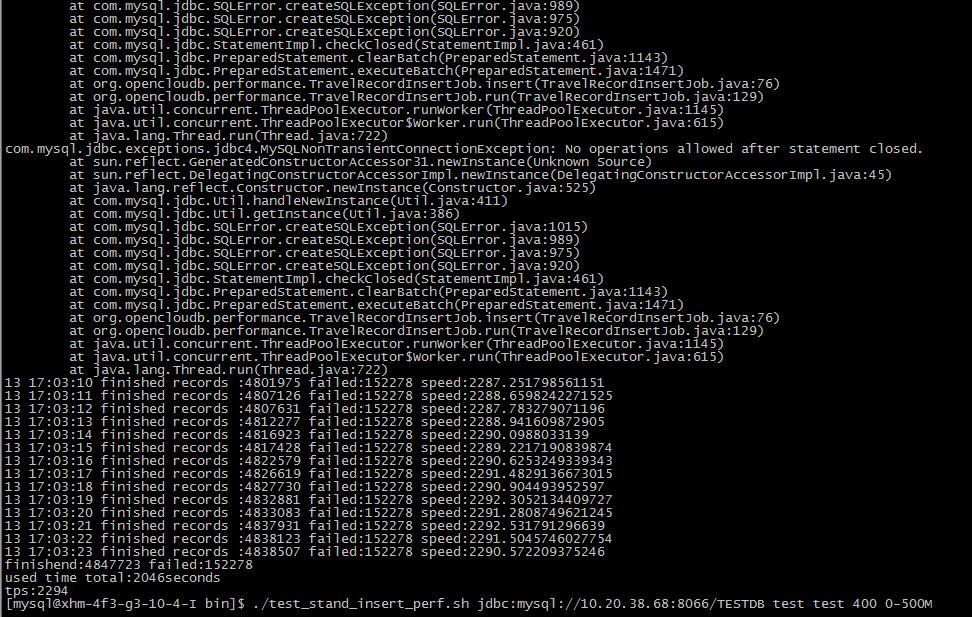
100并发

./test\_stand\_insert\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 100 0-500M



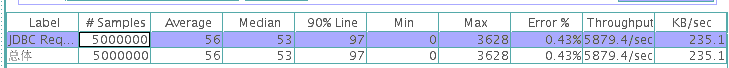
场景2，三个节点在高并发中宕掉一个节点

./test\_stand\_insert\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 400 0-500M

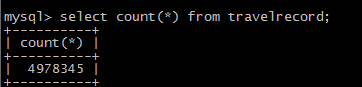


第二次测试

400并发



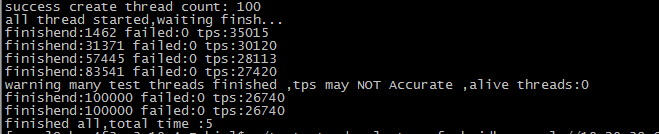
实际数据



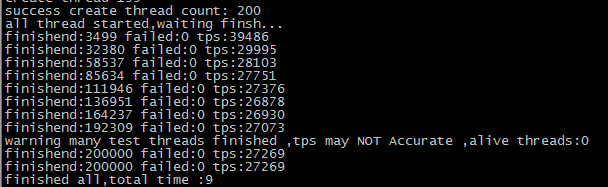
**存在部分数据没有写入的情况，此情况为检测节点失效过程中的时延导致，需要程序对SQL执行失败的数据进行处理，比如重新执行，或则写入文件中，事后分析和执行。**

查询测试

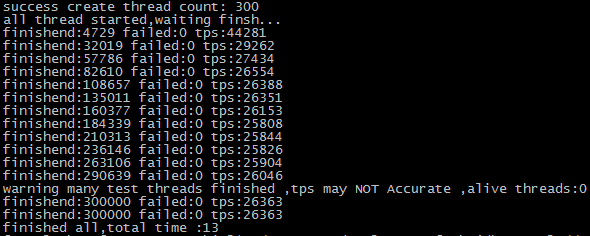
./test\_stand\_select\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 100 1000 5000000



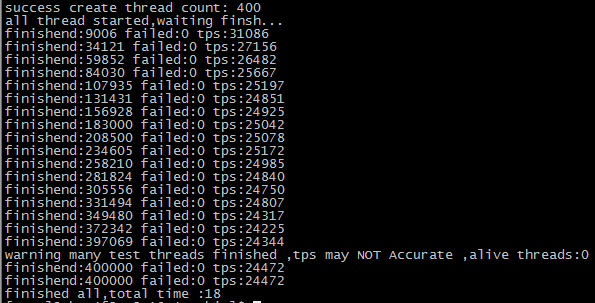
./test\_stand\_select\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 200 1000 5000000



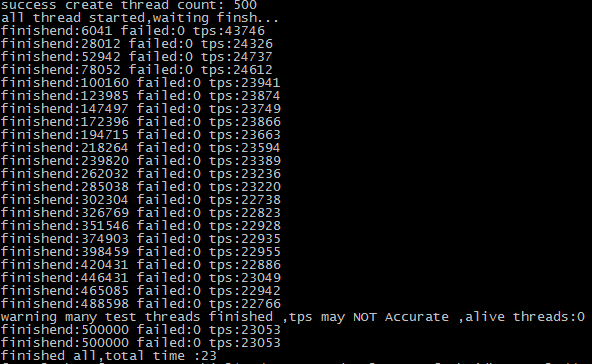
./test\_stand\_select\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 300 1000 5000000



./test\_stand\_select\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 400 1000 5000000



./test\_stand\_select\_perf.sh jdbc:mysql://10.20.38.68:8066/TESTDB test test 500 1000 5000000



Mycat 配置

#cat schema.xml

<?xml version="1.0"?>

<!DOCTYPE mycat:schema SYSTEM "schema.dtd">

<mycat:schema xmlns:mycat="http://org.opencloudb/">

<schema name="TESTDB" checkSQLschema="true">

<table name="travelrecord" primaryKey="ID" dataNode="dn1,dn2,dn3" rule="mod-long" />

</schema>

<dataNode name="dn1" dataHost="localhost1" database="db1" />

<dataNode name="dn2" dataHost="localhost1" database="db2" />

<dataNode name="dn3" dataHost="localhost1" database="db3" />

<dataHost name="localhost1" maxCon="1000" minCon="10" balance="0" writeType="1"

dbType="mysql" dbDriver="native">

<heartbeat>select user()</heartbeat> #检测mysql状态,自己可以定制sql

<writeHost host="hostM1" url="192.168.1.1:3306" user="mycat" password="mycat"> </writeHost>

<writeHost host="hostM2" url="192.168.1.2:3306" user="mycat" password="mycat"> </writeHost>

<writeHost host="hostM3" url="192.168.1.3:3306" user="mycat" password="mycat"> </writeHost>

</dataHost>

</mycat:schema>

注意: writeType是设置多主的,0为当主宕掉后会到第二个节点写入(默认为0),1为同时向三个节点写入数据.

Mysql\_cluster配置

cat my.cnf

[client]

socket= /usr/local/mysql/data/mysql.sock

[mysqld]

wait\_timeout=28800

skip-host-cache

skip-name-resolve

#innodb

socket= /usr/local/mysql /data/mysql.sock

datadir= /usr/local/mysql /data

user=mysql

wsrep\_provider= /usr/local/mysql/lib/libgalera\_smm.so

basedir= /usr/local/mysql

#wsrep\_cluster\_address=gcomm://192.168.1.1,192.168.1.2,192.168.1.3

binlog\_format=ROW

default\_storage\_engine=InnoDB

innodb\_locks\_unsafe\_for\_binlog=1

innodb\_autoinc\_lock\_mode=2

##add new parameters

innodb\_buffer\_pool\_size=15G

innodb\_buffer\_pool\_instances=4

innodb\_additional\_mem\_pool\_size=20M

innodb\_log\_buffer\_size=16M

innodb\_flush\_log\_at\_trx\_commit=0

innodb\_support\_xa=0

innodb\_file\_per\_table=1

innodb\_data\_file\_path=ibdata1:100M:autoextend

innodb\_read\_io\_threads=64

innodb\_write\_io\_threads=64

innodb\_io\_capacity=5000

innodb\_log\_file\_size=512M

innodb\_thread\_concurrency=0

innodb\_file\_format=Barracuda

innodb\_flush\_method=O\_DIRECT

innodb\_stats\_on\_metadata=0

#wsrep

###5.5.to update 5.6

#wsrep\_provider\_options="socket.checksum=1"

#log\_bin\_use\_v1\_row\_events=1

#gtid\_mode=0

#binlog\_checksum=NONE

#read\_only=ON

############################

wsrep\_certify\_nonPK=1

wsrep\_debug=0

wsrep\_convert\_LOCK\_to\_trx=0

wsrep\_max\_ws\_rows=131072

wsrep\_max\_ws\_size=1073741824

wsrep\_retry\_autocommit=1

wsrep\_auto\_increment\_control=1

wsrep\_node\_address=10.20.38.64

wsrep\_node\_name='node1'

#wsrep\_sst\_method=rsync

wsrep\_sst\_method=xtrabackup

wsrep\_drupal\_282555\_workaround=0

wsrep\_causal\_reads=0

wsrep\_slave\_threads=48

max\_connections=2000

wsrep\_cluster\_name=my\_centos\_cluster

#wsrep\_debug=on

#wsrep\_mysql\_replication\_bundle=200

wsrep\_provider\_options="socket.checksum = 1; pc.ignore\_quorum = true; pc.weight=0; gcache.size=8G; evs.keepalive\_period=PT3S; evs.inactive\_check\_period=PT10S; evs.suspect\_timeout=PT30S; evs.inactive\_timeout=PT1M; evs.consensus\_timeout=PT1M; evs.send\_window=1024; evs.user\_send\_window=512;"

wsrep\_sst\_auth="sstuser:s3cret"

配置和安装就不啰嗦,不懂的可以去百度或谷歌查询