## SQL 99 Specification

MyCAT supports SQL99 Specification, including DDL. Some of sql grammars in MySQL is not supported, however, MyCAT1.2 has made it with MySQL comments.

An example is below.

/\*!mycat select id from travelrecord where id=2\*/ select \* from travelrecord where id=2;

The part which starts with /\*!mycat xxxxx\*/ in the SQL above, is comment of mycat. A SQL meets SQL99 and replaced with xxxx, is used to notify MyCAT that it should be parsed, analyzed, and routed; MyCAT will send the real SQL after the comment to the routed database to execute and receive the result.

For example, MyCAT does not support the sql grammar such as **select into**, meanwhile, the table to be inserted is not sharding, so the SQL could be written as following:

/\*!mycat insert into B\*/ select \* from A insert into B

NOTE: The grammar of comment in newest version has been changed as below:

/\*!mycat: sql = select id from travelrecord where id=2\*/ select \* from travelrecord where id=2;

## GLOBAL TABLES IN MyCAT

Generally, there are many tables as dictionaries in a real scenario. These tables should be considered as **LABELS** for business tables, but not as primary tables for them. These tables are very seldom changed, so they could be cached based on their primary key.

The following diagram illustrates a typical "label Relations".

**Network device**

Due to business growth, the relation between sharding business tables and dictionary tables, become thorny issues. Considering the following features of dictionary tables:

* Changed infrequently
* Data quantity is not big
* Data scales is not large, there is little more than a few handrand thousand records

So, MyCAT defines a special table, named “global table”, which has following features:

* To insert, update, delete global tables, are executed on all nodes immediately, keeping data are consistency on each shard.
* To select records of global tables, queries on node only
* Global tables could be joined with any other tables

To define dictionary tables as global ones resolves the joining problems. Through global tables and sharding policy based on E-R relation, MyCAT can resolve most of developing problems with data sharding on RDBMS.

Global table configuration is easy. It need not sharding rule. You just add config as following:

<table name="company" primaryKey="ID" type="global" dataNode="dn1,dn2,dn3" />

需要注意的是，全局表每个分片节点上都要有运行创建表的DDL语句。

Note that, DDLs of each global tables must be executed on each DB node.

## HA and Read/Write splitting

Read/Write splitting policy of MyCAT are following:

* All SQLs will be executed on writing node, in a transaction, except sqlect SQLs which starts with /\*balance\*/
* **Auto commited** select SQL will be executed on **reading nodes**, with random balancing among them
* If one of masters is offline, all of its slaves is unavailable, because the new data could not be updated to reading nodes. MyCAT will load balance in using of another cluster with master/slaves.
* If all masters are unavailable, all auto committed select SQL will be sent to alive reading nodes. In the situation, most query accesses could receive datas, but insert/update/delete will fail.

following is configuration of read/write splitting



the attribute named “balance” of dataHost above has following values:

* 0，close read/write splitting
* 1，all **readHost**s and standing by **writeHost**s will load balance of select SQL. In the other hand, in double masters and double slaves mode(M1->S1, M2->S2), M2, S1, S2 will load balance of select SQL in common situations.
* 2，all **readHost**s and **writeHost**s will load balance select SQLs. In the other words, all nodes will load balance, if there are not too many accesses.

One dataHost element in config file, maps a group of database hosts with data synchronization. **DBA should guarantee this.** A writeHost maps a master of database server, meanwhile, each sub-node named readhost maps a slave database server which is synchronized with the master. When several **writeHost**s are configured, MyCAT could detect the offlined ones, and switch next available writeHost.

MyCAT supports HA, according to your application features, it could be configured as following:

* To configure database as one-master and multi-slaves, open read/write splitting
* To configure database as double-masters and double-slaves(multi-slaves), open read/write splitting
* To configure database as multi-masters and multi-slaves, open read/write spliting

The last two configurations, are with higher availabilities, if one of masters is failover, MyCAT will detect by heartbeating, and switch to another master node. **In any time, MyCAT will write to one node only.**

**Following is the** typical configuration of double-masters and double-slaves:

MyCAT

)

M2(writeHost)

)

M1(writeHost)

)

S2(readHost)

)

S1(readHost)

)

To set log level is debug in log4j.xml, MyCAT will output following logs.

16:37:21.660 DEBUG [Processor0-E3] (PhysicalDBPool.java:333) -select read source hostM1 for dataHost:localhost1

16:37:21.662 DEBUG [Processor0-E3] (PhysicalDBPool.java:333) -select read source hostM1 for dataHost:localhost1

Based on this information, each SQL could be determined which node is sending to, thus you can determine whether there has separation of read and write.

## MYCAT SEQUENCE

MyCAT sequence is a new feature of MyCAT, to ensure the primary key is unique in global with sharding. However, the auto increment feature of MySQL cannot achieve this. MyCAT sequence grammar meets SQL standard specification. You can use it like this:

next value for MYCATSEQ\_GLOBAL

MYCATSEQ\_GLOBAL is sequence name. MyCAT will create new sequence, to simplify the development complexity. In addition, MyCAT also provides a global sequence, named: MYCATSEQ\_GLOBAL.

**NOTE that, MYCATSEQ\_ must be UPPER CASE.**

NOTE again, in practice, it is suggested that to create a sequence for each table, and sequence name is like MYCATSEQ\_TableName\_ID\_SEQ.

sequence instraction in SQL

Sequence identifier is like MYCATSEQ\_XXX. Examples are following.

To use default global sequence:

insert into tb1(id,name) values(next value for MYCATSEQ\_GLOBAL,'micmiu.com');

To use custom sequence:

insert into tb2(id,name) values(next value for MYCATSEQ\_MY1,'micmiu.com');

To get a new value of sequence:

Select next value for MYCATSEQ\_XXX

MyCAT目前已经提供了一个本地配置版的实现，下面是配置说明：

MyCAT has provided a local config implementation. The following is configuration instruction.

Config file: sequence\_conf.properties

Config content:

XXX.HISIDS= 1-100,501-800,3001-5000 //used section in history

XXX.MINID=10001 //min value of current section

XXX.MAXID=20000 //max value of current section

XXX.CURID=10000 //current value in current section

Example of global sequence config:

GLOBAL.HISIDS=

GLOBAL.MINID=1

GLOBAL.MAXID=50000

GLOBAL.CURID=10000

Example of custom sequence config:

MY1.HISIDS=

MY1.MINID=101

MY1.MAXID=200

MY1.CURID=152

To config to store sequence in MySQL:  
Edit Serfver.xml, and add this config item   
<property name="sequnceHandlerType">1</property>  
• To create table and function about sequence in one data node. Sqls are stored in $MYCAT\_HOME/doc/sequence-sql.txt, and they should be executed on mysql, but not on MyCAT.   
• $MYCAT\_HOME/conf/quence\_db\_conf.properties contains configuration info of storing sequence database.   
#sequence stored in datanode  
GLOBAL=dn1  
COMPANY=dn1  
CUSTOMER=dn1  
• In sequence table, to insert a sequence record, confirm its initial value and step size. The suggestion is that step size is in a suitable range, such as 50-500. It should be executed on MySQL, but not on MyCAT. For example as following:  
INSERT INTO MYCAT\_SEQUENCE VALUES ('GLOBAL', 0, 100);  
• To update sequence current value as a new one, it should be executed on MySQL, but not on MyCAT.

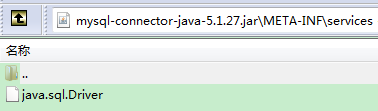
• To select sequence values with sql as following:  
SELECT mycat\_seq\_curval('GLOBAL');

NOTE that, step size depends on your TPS of data insertion. If it is 1000 per second, the step size is 1000\*10=10,000. A batch of sequence values will be selected every 10 seconds. It’s not so large.

## To support other databases on JDBC

From MyCATv1.2, it has supported to connect other databases besides MySQL in a common way, such as Oracle, SQL server, DB2, and so on. MyCAT is still a MySQL server to client.

To config it as following, at first to place database driver jar which meets JDBC4 under $MYCAT\_HOME/lib, and check the file which path is META-INF/services/java.sql.Driver in the driver jar file.



The file content is the class name of driver:



To config dataHost in schema.xml as following:

<dataHost name="jdbchost" maxCon="1000" minCon="10" balance="0"

dbType="mysql" **dbDriver**="**jdbc**">

<heartbeat>select user()</heartbeat>

<writeHost host="hostM1" **url**="jdbc:mysql://localhost:3306"

user="root" password="123456">

</writeHost>

</dataHost>

## CONFIGURATION OF MyCAT

* System arguments and user permissions of MyCAT could be defined in server.xml. Currently, MyCAT supports two kinds of permissions such as readonly and read/write.

Following is the configuration:

<user name=*"test"*>

<property name=*"password"*>test</property>

<property name=*"schemas"*>TESTDB</property>

<property name=*"readOnly"*>true</property>

</user>