一 mycat 介绍

数据库的集群,分布式高可用的结构,需要引入中间件的软件,实现中间状态的计算,连接后端数据库完成客户端的所有功能需求 sql 语句的执行.将客户端与服务端数据库切分开来.
mycat 是一个国内企业级别高性能数据库中间件.京东,ali 都在使用.

二 mycat 特点

1 高性能的读写分离（速度非常快）

对于数据库集群来讲,主节点可以写数据,同时也可以读,从节点不能写数据,只能读数据.
多个节点组成的一个主从结构,可以实现读和写的分离操作---读写分离.
读写分离的存在能够大大提升数据库主从集群的使用效率.
select 显然是读,一般大量分配给从节点
insert/drop/delete 显然是写,全部分配给主节点

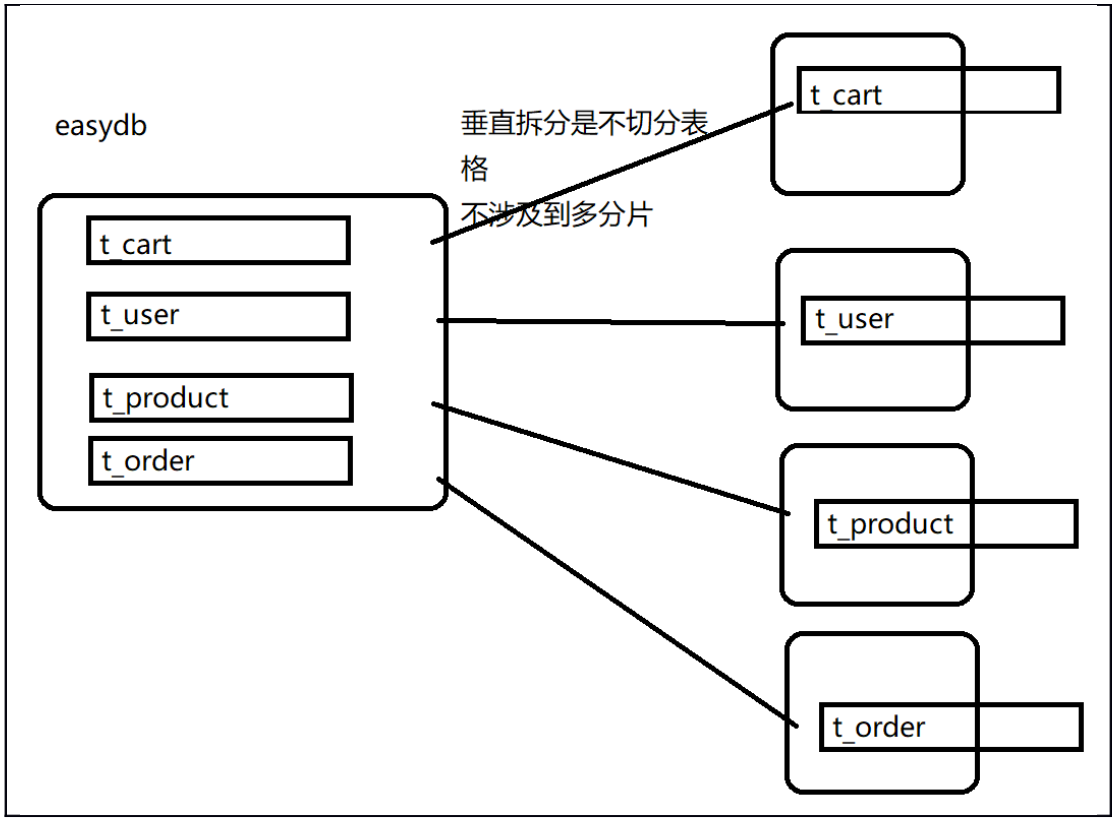
2 实现水平并行计算（水平拆分）

2.A 单机数据库结构：

一个库多个表格存在
多个库，存在多个表格

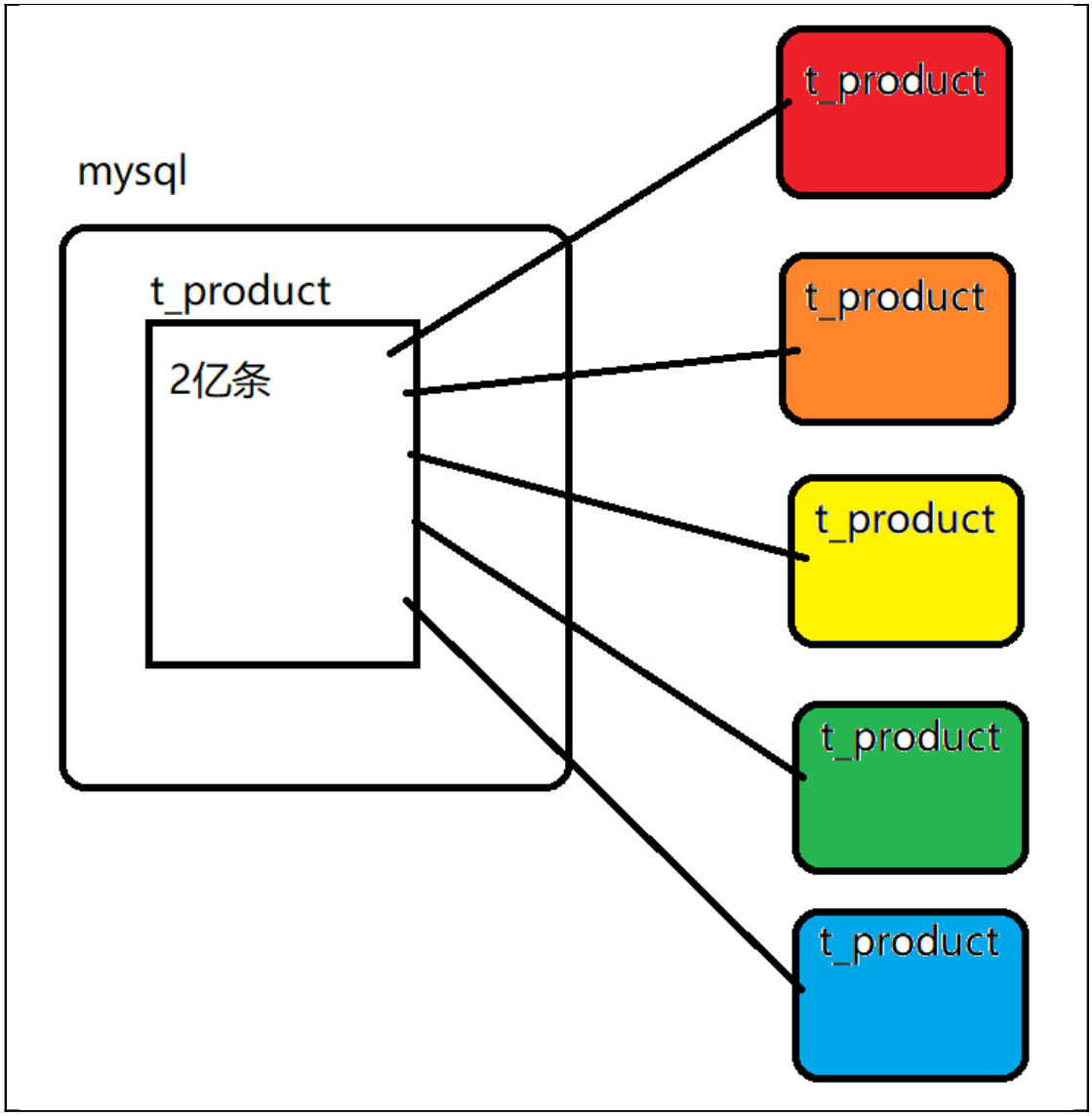
2.B 垂直拆分：

将不同的单机结构中的库数据，拆分到多个服务器中。由于微服务的存在，可以非常简单的实现垂直拆分（每一个拆分的结构中表格是完整的）



2.C水平拆分：

将同一个数据库中同一个表的数据切分到多个数据库中（也就是表记录的拆分，分到不同数据库存储），出现的数据分片（拆分表格），如果一个表格数据库需要特别大的量级，比如 30 亿条，没有任何一个数据库技术可以支持这么大的表格数据（oracle 单表数据量亿级别，mysql 单表数据量千万级别）,这样用水平拆分后逻辑上才能支持



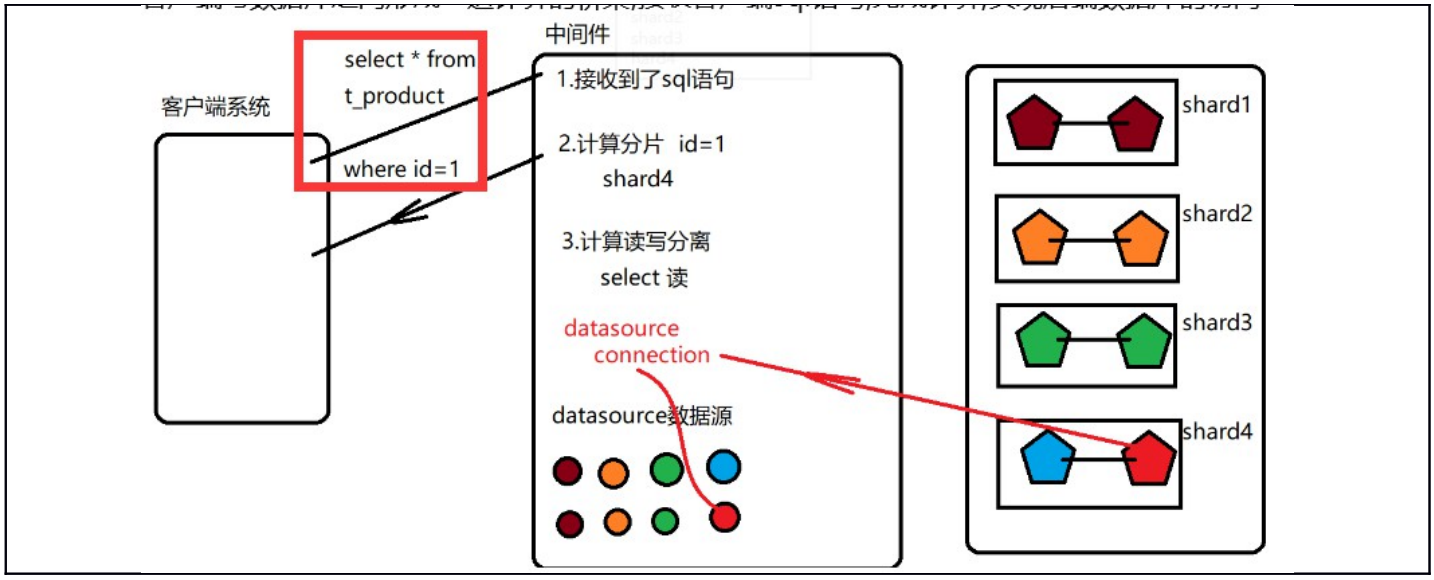
由于一个表格的整体数据被切分到了不同的真实数据库管理的一部分，那么实际运用中一定存在数据分片的计算。

mycat 可以实现水平分片，大表总数据量百亿，管理和计算后端非常庞大的数据库集群。

官方数据：mycat 的权威指南千亿级别。一般千亿级别的数据，都会采用大数据存储技术实现存储。

三　mycat 运行原理

mycay 在客户端与数据库之间形成一道计算的桥梁，接受客户端的 sql 语句，完成计算，实现后端数据库的访问



1. mycat 启动时，就生成了管理连接后端真实数据库的所有 datasource。
2. 客户端只能连接 mycat 发送 sql 语句。
3. mycat 接受客户端发送的 sql 语句，执行各种计算（包括计算 sql 对应哪个分片、计算读写分离等），拿到了数据源，获取连接发送 sql 语句。
4. 数据库拿到 mycat 发送的 sql 开始执行。
5. mycat 将数据库返回结果返回给客户端。
   客户端还以为是 mycat 执行的 sql 获取数据结果。

四　mycat 处理假死逻辑

cobar 是 mycat 的前身，是一个早期版本比较流行的中间件，没有解决后端连接的假死发生，2011 出现了 mycat 解决了后端假死问题。

mycat 不太需要关注假死的·现象，因为 mycat 可以非常高效的使用系统资源几乎不会出现假死现象。

1　什么是假死?

当中间件的线程资源在运行时被占满了，导致新的请求无法连接后端数据库（原因是没有线程可以分配），这时中间件错误的判断是数据库宕机导致的—这种情况称之为假死。

五　mycat 安装目录介绍

安装目录/home/software/mycat,目录结构如下:

```
18 22:20 bin
13  2015 catlet
18 22:21 conf
18 22:20 lib
13  2015 logs
30  2016 version.txt
```

1 bin

叫 bin 的文件夹一般都是运行命令脚本

```
-rwxr-xr-x 1 root root    593 Dec 15  2015 init_zk_data.sh
-rwxr-xr-x 1 root root  15714 Nov 30  2016 mycat
-rwxr-xr-x 1 root root   2947 Dec 13  2015 rehash.sh
-rwxr-xr-x 1 root root   2502 Dec 13  2015 startup_nowrap.sh
-rwxr-xr-x 1 root root 140198 Nov 30  2016 wrapper-linux-ppc-64
-rwxr-xr-x 1 root root  99401 Nov 30  2016 wrapper-linux-x86-32
-rwxr-xr-x 1 root root 111027 Nov 30  2016 wrapper-linux-x86-64
-rwxr-xr-x 1 root root    594 Jan 22  2016 xml_to_yaml.sh
```

可以在 bin 目录下执行（配置环境变量后可在任意地方运行）：

mycat console 控制台运行（一定 bin 文件夹下）

mycat start 后台运行

mycat stop 停止

mycat restart 重启

2 catlet

外部插件

3 conf

mycat 的配置文件

```
-rwxrwxrwx 1 root root   88 Nov 30  2016 autopartition-long.txt
-rwxrwxrwx 1 root root  340 Nov 30  2016 cacheservice.properties
-rwxrwxrwx 1 root root  439 Nov 30  2016 ehcache.xml
-rwxrwxrwx 1 root root  931 Nov 30  2016 index_to_charset.properties
-rwxrwxrwx 1 root root 1647 Dec 13  2015 log4j.xml
-rwxrwxrwx 1 root root   51 Nov 30  2016 myid.properties
-rwxrwxrwx 1 root root   15 Nov 30  2016 partition-hash-int.txt
-rwxrwxrwx 1 root root  102 Nov 30  2016 partition-range-mod.txt
-rwxrwxrwx 1 root root  943 Nov 30  2016 router.xml
-rwxrwxrwx 1 root root 4510 Nov 30  2016 rule.xml
-rwxr-xr-x 1 root root 4510 Feb 18 22:21 rule.xml.bak
-rwxrwxrwx 1 root root 4237 Nov 30  2016 schema.xml
-rwxr-xr-x 1 root root 4237 Feb 18 22:20 schema.xml.bak
-rwxrwxrwx 1 root root  413 Nov 30  2016 sequence_conf.properties
-rwxrwxrwx 1 root root   75 Nov 30  2016 sequence_db_conf.properties
-rwxrwxrwx 1 root root   51 Nov 30  2016 sequence_time_conf.properties
-rwxrwxrwx 1 root root 2346 Jun  3 10:46 server.xml
-rwxr-xr-x 1 root root 2507 Feb 18 22:21 server.xml.bak
-rwxrwxrwx 1 root root 4188 Nov 30  2016 wrapper.conf
-rwxrwxrwx 1 root root 5618 Nov 30  2016 zk-create.yaml
```

4 logs

运行日志（控制台运行可以直接输出日志）

```
[root@10-42-175-170 mycat]# cd logs
[root@10-42-175-170 logs]# ll
total 28
-rw-r--r-- 1 root root 13386 Jun  3 13:28 mycat.log
-rw-r--r-- 1 root root     5 Jun  3 13:27 mycat.pid
-rw-r--r-- 1 root root  4774 Jun  3 13:28 wrapper.log
[root@10-42-175-170 logs]#
```

六 mycat 的使用

1 登录（和 mysql 类型，端口 8066）

linux 执行 mysql 命令(mycat 支持默认环境 mysql)

```
[root@10-42-147-110 ~]# mysql -utest -ptest -P8066 -h10.42.175.170
```

-u 用户

-p 密码

-P 端口

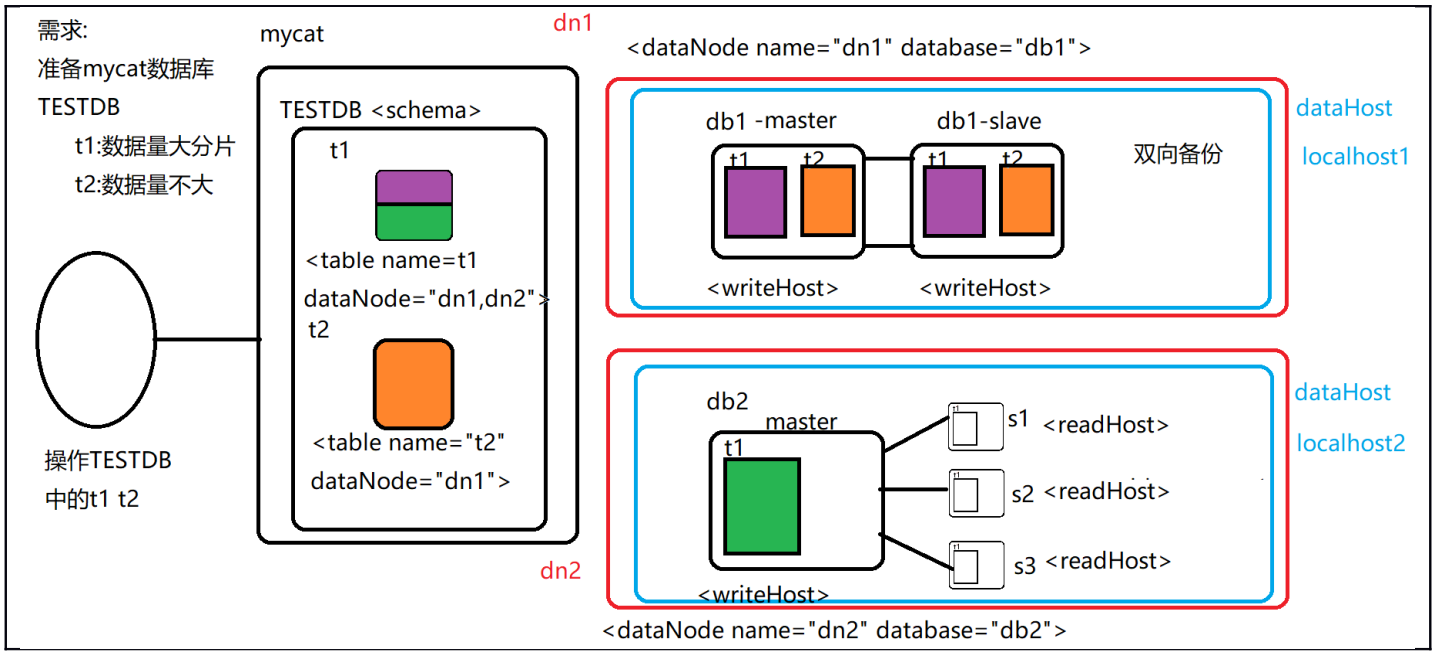
-h ip

- sqlYog 登录 mycat

没有配置好后端数据的配置文件之前.不要使用 sqlYog 会卡死.

只要在连接属性配置 ip username password port

七　mycat 的配置文件详解



在 mycat 根目录有 conf 文件夹，其中有三个 xml（server、schema、rule）配置，都会使用到。

一旦涉及到修改 xml，习惯备份原文件。

```
[root@10-42-175-170 mycat]# cd conf
[root@10-42-175-170 conf]# ll
total 104
-rwxrwxrwx 1 root root   88 Nov 30  2016 autopartition-long.txt
-rwxrwxrwx 1 root root  340 Nov 30  2016 cacheservice.properties
-rw-r--r-- 1 root root   66 Jun  3 17:04 dnindex.properties
-rwxrwxrwx 1 root root  439 Nov 30  2016 ehcache.xml
-rwxrwxrwx 1 root root  931 Nov 30  2016 index_to_charset.properties
-rwxrwxrwx 1 root root 1647 Dec 13  2015 log4j.xml
-rwxrwxrwx 1 root root   51 Nov 30  2016 myid.properties
-rwxrwxrwx 1 root root   15 Nov 30  2016 partition-hash-int.txt
-rwxrwxrwx 1 root root  102 Nov 30  2016 partition-range-mod.txt
-rwxrwxrwx 1 root root  943 Nov 30  2016 router.xml
-rwxrwxrwx 1 root root 4510 Nov 30  2016 rule.xml
-rwxr-xr-x 1 root root 4510 Feb 18 22:21 rule.xml.bak
-rwxrwxrwx 1 root root 1288 Jun  3 17:04 schema.xml
-rwxr-xr-x 1 root root 4237 Feb 18 22:20 schema.xml.bak
-rwxrwxrwx 1 root root  413 Nov 30  2016 sequence_conf.properties
-rwxrwxrwx 1 root root   75 Nov 30  2016 sequence_db_conf.properties
-rwxrwxrwx 1 root root   51 Nov 30  2016 sequence_time_conf.properties
-rwxrwxrwx 1 root root 2346 Jun  3 10:46 server.xml
-rwxr-xr-x 1 root root 2507 Feb 18 22:21 server.xml.bak
-rwxrwxrwx 1 root root 4188 Nov 30  2016 wrapper.conf
-rwxrwxrwx 1 root root 5618 Nov 30  2016 zk-create.yaml
[root@10-42-175-170 conf]#
```

1 server.xml

1.A 标签结构

system property user property quarantine whitehost blacklist

```
<mycat:server xmlns:mycat="http://org.opencloudb/">
        <system>
        <property name="defaultSqlParser">druidparser</property>
       <!--  <property name="useCompression">1</property>--> <!--1为开启mysql压缩协议-->
        <!-- <property name="processorBufferChunk">40960</property> -->
        <!--
        <property name="processors">1</property>
        <property name="processorExecutor">32</property>
         -->
                <!--默认是65535 64K 用于sql解析时最大文本长度 -->
                <!--<property name="maxStringLiteralLength">65535</property>-->
                <!--<property name="sequnceHandlerType">0</property>-->
                <!--<property name="backSocketNoDelay">1</property>-->
                <!--<property name="frontSocketNoDelay">1</property>-->
                <!--<property name="processorExecutor">16</property>-->
                <!--
                        <property name="mutiNodeLimitType">1</property> 0：开启小数量级（默认）
                <property name="mutiNodePatchSize">100</property> 亿级数量排序批量
                        <property name="processors">32</property> <property name="processorExecu
                        <property name="serverPort">8066</property> <property name="managerPort"
                        <property name="idleTimeout">300000</property> <property name="bindIp">0
                        <property name="frontWriteQueueSize">4096</property> <property name="pro
        </system>
        <user name="test">
                <property name="password">test</property>
                <property name="schemas">TESTDB</property>
        </user>

        <!--
        <quarantine>
           <whitehost>
              <host host="127.0.0.1" user="mycat"/>
              <host host="127.0.0.2" user="mycat"/>
           </whitehost>
         <blacklist check="false"></blacklist>
         </quarantine>
        -->

</mycat:server>
```

1.B 标签和属性

system：

内容与 mycat 进程启动时占用的资源，内部配置属性有关，见上图

i.1 ***property*** *标签*：可以配置压缩格式，配置端口号，配置占用系统的线程资源等等。

user：

mycat 登录的用户配置。权限配置等

下面是 user 下<propertiy>可以配置的 name

i.1 ***name***：用户名

i.2 ***password***：用户密码

i.3 ***schemas***：该用户可以访问的 mycat 中的数据库。可以访问多个数据库以，隔开，必须是 mycat 中存在的库(在 schema.xml 中配置才有)，如果有任何一个库不存在，将会报错

i.4 ***readOnly***：用户只读

例：test 用户的密码 test，可以访问 TESTDB 数据库

```
<user name="test">
    <property name="password">test</property>
     <property name="schemas">TESTDB</property>
 </user>
```

ii. **quarantine**：防火墙配置

i.1 ***whitehost***：ip 白名单

只有客户端 ip 地址在白名单范围内，才被允许访问 mycat

```
<host host="127.0.0.1" user="mycat"/>
```

上例中：只有客户端是从 127.0.0.1 的 ip 来访问 mycat，并且使用 mycat 用户登录才能登录成功

i.2 ***blacklist：*** **sql 黑名单** （规则见附录）

在黑名单中列出的 sql 规则，都不可以使用。例如：drop 不允许执行，没有 where 条件的 delete 不能执行。

```
<blacklist check="false"></blacklist>
```

标签中值是空的不检查

例如：限制客户端不能使用 select * 这种语句

```
<blacklist check="true">selectAllColumnAllow<blacklist>
```

2 schema.xml

2.A 标签结构

i. schema
   1. table
      1.a. childTable
         1.a.i. childTable

ii. dataNode

iii. dataHost
   1. heartbeat
   2. writeHost
      1.a. readHost

```
<?xml version="1.0"?>
<!DOCTYPE mycat:schema SYSTEM "schema.dtd">
<mycat:schema xmlns:mycat="http://org.opencloudb/" >

        <schema name="TESTDB" checkSQLschema="true" sqlMaxLimit="100">
                <table name="tb_user" dataNode="dn1" primaryKey="id"/>
                <table name="tb_student" dataNode="dn1,dn2" primaryKey="id" rule="auto-sharding-long"/>
        </schema>
        <dataNode name="dn1" dataHost="localhost01" database="mstest"/>
        <dataNode name="dn2" dataHost="localhost02" database="mstest"/>

        <dataHost name="localhost01" maxCon="1000" minCon="10" balance="1"
                writeType="0" dbType="mysql" dbDriver="native" switchType="1"  slaveThreshold="100">
                <heartbeat>select user()</heartbeat>
                <writeHost host="M1" url="10.42.175.170:3306" user="root" password="root">
                        <!--<readHost host="M2" url="10.42.147.110:3306" user="root" password="root"/>-->
                <!--<writeHost host="M2" url="10.42.147.110:3306" user="root" password="root">-->
                </writeHost>
        </dataHost>
        <dataHost name="localhost02" maxCon="1000" minCon="10" balance="1"
                writeType="0" dbType="mysql" dbDriver="native" switchType="1"  slaveThreshold="100">
                <heartbeat>select user()</heartbeat>
                <writeHost host="M1" url="10.42.147.110:3306" user="root" password="root">


                </writeHost>
        </dataHost>
</mycat:schema>
```

2.B 标签和属性

2.B.a schema 标签：

可以在一个 schema.xml 配置多个 schema 的标签，每一个都表示客户端可以看到一个数据库--不是真实的数据库，可以在 mycat 中 show databases 查询。

```
<schema name="TESTDB" checkSQLschema="true" sqlMaxLimit="100">
```

属性：

1 name：客户端可以看到和使用的数据库名称。sehema.xml 中配置的多个 schema 标签，对应 server.xml 用户互配置标签中 schemas 属性。

2 checkSQLschema：可以配置 true 和 false，表示所有 sql 语句是否需要添加表格的数据库名称。例如：表格 student,所在库 db1.配置 fasle，sql 发送到 mycat，select * from student.不会拼接 db1。但是如果是 true，变成 select * from db1.student.在 mycat 中存在多个库，多个表格，可以唯一定位表格的名称。

3 sqlMaxLimit：整数值，当 sql 语句做批量查询，mycat 防止性能浪费，在判断 sql 语句语句中没有 limit 关键字自动拼接 limit 0,100 查询前 100 条。

table 子标签：

在一个 schema 标签中，存在的数据库表格，可以有多个 table 标签，表示多个表格。

```
<table name="t1_student" primaryKey="s_id" dataNode="dn1"/>
```

属性

1 name：表格名称，所有表格数据都一定来自真实库，表格名称要和真实库的表格名一致，并且同一个 schema 标签中只能存在唯一一个名称 table

2 primaryKey：主键名称，对应表格的真实数据中的主键名字，默认是 id，当不是 id 时，需要配置这个属性

3 dataNode：table 表格数据，可以根据数据量大小，实现水平切分，对应的数据分片计算绑定 dataNode 标签，这里可以设置当前表格被切分到了几个数据分片中,可以对应一个分片，可以对应多个分片。 指定 dataNode 标签的 name。

4 rule：当表格需要进行对应多个分片数据切分时，指定切分数据数据分片计算逻辑，可以给配置 rule 规则，例如：auto-sharding-long 整数范围约束,表示以主键某个字段的整数做分片计算，0-500 万对应第一个分片，500 万-1000 万对应第二个分片，1000 万-1500 万对应第三个分片。字符串可以使用一致性 hash sharding-by-murmur. 指向 rule.xml 中<tableRule>的 name。

2.B.b dataNode 标签

指向真实的数据节点（可以理解为真的数据库）

在 mycat 管理一个主从数据库集群作为分片使用，需要将集群包装在一个数据分片对象中，一个 dataNode 标签标示一个数据分片--最主要的作用就是计算分片的，利用名称，下标实现分片计算。

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

属性

1 name：分片名称，按照配置顺序每个 dataNode 还有一个下标从 0 开始，可以做一致性 hash 计算。

2 dataHost：一个分片不负责数据库集群的管理，只是绑定数据库集群管理对象 dataHost，dataHost 使用名字来绑定。

3 database：当前分片绑定真实数据库集群使用，真实数据库中 database 可能有多个，当前分片用的是哪个库（垂直划分），指定真实数据库库名称

2.B.c dataHost 标签

负责管理一个真正的数据库主从集群。连接池，读写分离都是由这个标签的对象实现的计算。

```
<dataHost name="localhost1" maxCon="1000" minCon="10" balance="0"
          writeType="0" dbType="mysql" dbDriver="native" switchType="1"
          slaveThreshold="100">
```

属性

name：当前 dataHost 的名字，绑定 dataNode 时使用

maxCon：当前 dataHost 管理的数据主从集群每个节点的连接池中最大连接

minCon：最小连接

balance：读写分离的读逻辑，见 mycat 读写分离详解

writeType：读写分离的写逻辑，见 mycat 读写分离详解

dbType：默认 mysql，数据库软件类型

dbDriver：默认 mysql 叫做 native，如果是其他数据库给定 driver 全路径值

switchType：故障转移有关，见 mycat 故障转移详解

slaveThreshold：100 是毫秒数，表示当前主从集群，从节点 sql 延迟 100 毫秒以上时，将不会使用该节点处理读数据逻辑

heartbeat:

dataHost 连接使用后端数据时，实现心跳检测 sql 语句，一般有 2 中常用的，select user(),show slave status(只有使用该语句，slaveThreshold：100 才能生效)

```
<heartbeat>select user()</heartbeat>
```

writeHost:

写主机，表示在一个主从集群中的真实库连接标签。只能在其中配置主节点

```
<writeHost host="hostM1" url="10.202.4.39:3306" user="root"
password="sf123456">
```

属性

host：代号名称，相当于 name 一般会使用默认结构 hostM1 第一个主节点

url：ip：port 连接

user：登录数据用户名

password：登录密码

readHost:

读主机，表示主从集群中真实库连接，主从节点都可以在这里配置，一般都是使用从节点。

```
<readHost host="hostS2" url="192.168.1.200：3306" user="root"
password="xxx" />
```

属性

host：代号名称，相当于 name 一般会使用默认结构 hostM1S1 第一个主节点第一个从节点

url：ip：port 连接

user：登录数据用户名

password：登录密码

八　mycat 入门案例（实现代理中间件）

1　环境准备

一个后端数据库，这里用 mstest 来测试

一个 mycat 服务

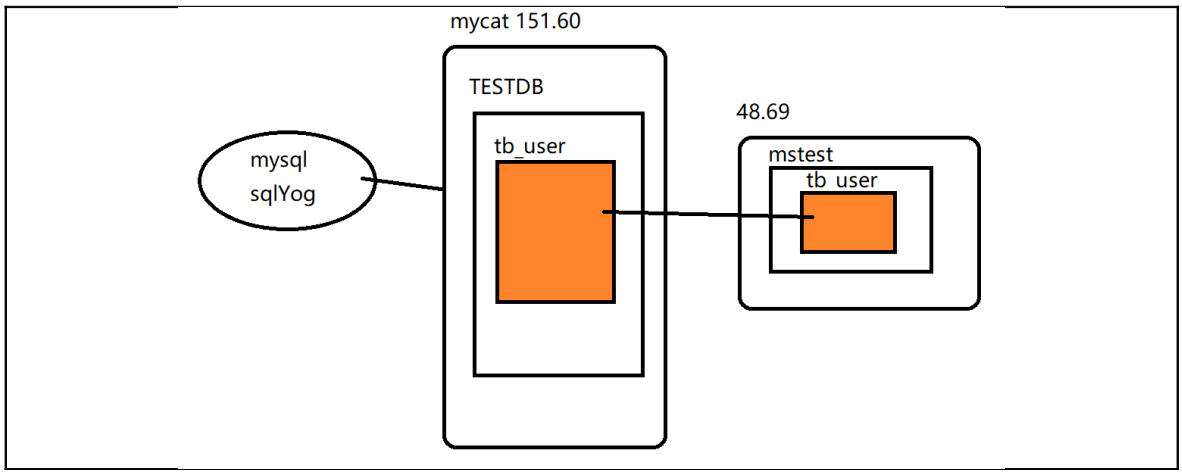
2　功能描述

通过客户端(mysql 登录,sqlYog)登录 mycat。

访问 mycat 的一个逻辑数据库 TESTDB。

看到使用一个表格 tb_user,不需要分片,查询所有数据都来自于后端一个真是库。

3　结构图



4　配置 xml 文件

4.A　Server.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mycat:server SYSTEM "server.dtd">
<mycat:server xmlns:mycat="http://org.opencloudb/">
        <system>
                <property name="defaultSqlParser">druidparser</property>
        </system>
        <user name="root">
                <property name="password">root</property>
                <property name="schemas">TESTDB</property>
        </user>
```

```
</mycat:server>
```

4.B Schema.xml

```
<?xml version="1.0"?>
<!DOCTYPE mycat:schema SYSTEM "schema.dtd">
<mycat:schema xmlns:mycat="http://org.opencloudb/" >
	<!--mycat only one logic database TESTDB-->
	<schema name="TESTDB" checkSQLschema="true" sqlMaxLimit="100">
		<!--tb_user-->
		<table name="tb_user" primaryKey="id" dataNode="dn1"/>
	</schema>
	<dataNode name="dn1" dataHost="localhost1" database="mstest"/>
	<dataHost	name="localhost1"	maxCon="1000"	minCon="10"
balance="0"
	writeType="0" dbType="mysql" dbDriver="native" switchType="1"
	slaveThreshold="100">
		<heartbeat>select user()</heartbeat>
		<writeHost host="M1" url="10.9.48.69:3306"
		user="root" password="root"/>
	</dataHost>
</mycat:schema>
```

5 测试

6 常见问题

- xml 的语法编写配置补熟练,有语法错误.

从 mycat 启动日志中,找到 Caused by 错误提示关键字，mycat console 可以直接从控制台看到日志

```
java:105)
jvm 1    |      at org.opencloudb.MycatServer.<clinit>(MycatServe
r.java:73)
jvm 1    |      ... 7 more
jvm 1    | Caused by: org.xml.sax.SAXParseException; lineNumber:
8; columnNumber: 4; The element type "table" must be terminated b
y the matching end-tag "</table>".
jvm 1    |      at com.sun.org.apache.xerces.internal.util.ErrorH
andlerWrapper.createSAXParseException(ErrorHandlerWrapper.java:20
3)
jvm 1    |      at com.sun.org.apache.xerces.internal.util.ErrorH
```

测试登录 mycat 访问后端真实库的表格资源,有一个常见的问题

- writeHost/readHost 提供的 datasource 创建的必要属性
  - url

- user
- password

给错了,所以 mycat 无法创建后端数据库的 datasource 导致你无法使用 mycat 执行操作表格增删查改的功能

ERROR 3009 (HY000): java.lang.IllegalArgumentException: Invalid DataSource:0

九　mycat 读写分离

数据某个分片中，实现的主从高可用备份,通过对主的写操作，对从的读操作将读写分离执行提升集群使用效率

读写分离和故障转移都是基于主从结构实现(mycat 至少管理一个后端主从结构)

- 之前主从复制课程,实现了一主一从的数据库结构,测试读写分离,故障转移,为了测试效果,把搭建好的主从给断开

1　准备环境

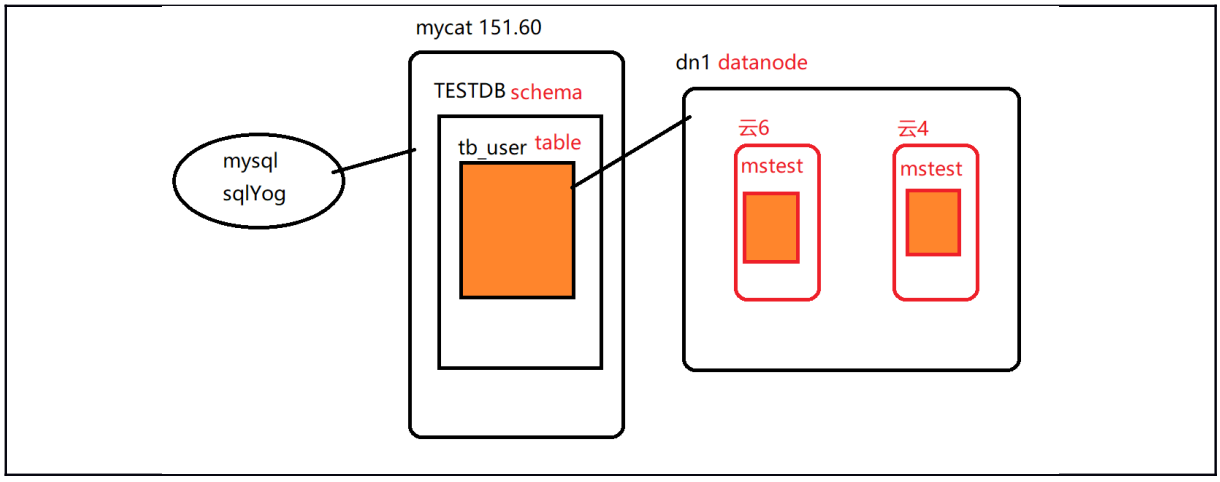
- 准备 2 个后端数据库,并且将已有的主从断开
  - 10.9.151.60:3306 10.9.48.69:3306
  - 到从节点 执行 stop slave
- 准备一个 mycat 的软件

  一个云主机运行了一个后端数据库

  令外一个云主机运行另另一个后端数据库

  同时运行了 mycat

2　结构图



3 配置文件 schema.xml

经过测试发现,配置了 2 个 writeHost 标签,但是读和写的操作都是在第一个 writeHost 进行的.
因为 balance 为 0，表示不开启读写分配

```
<?xml version="1.0"?>
<!DOCTYPE mycat:schema SYSTEM "schema.dtd">
<mycat:schema xmlns:mycat="http://org.opencloudb/" >
	<!--mycat only one logic database TESTDB-->
	<schema name="TESTDB" checkSQLschema="true" sqlMaxLimit="100">
		<!--tb_user-->
		<table name="tb_user" primaryKey="id" dataNode="dn1"/>
	</schema>
	<dataNode name="dn1" dataHost="localhost1" database="mstest"/>
	<dataHost name="localhost1"  maxCon="1000" minCon="10" balance="0"
	writeType="0" dbType="mysql" dbDriver="native" switchType="1"
	slaveThreshold="100">
	<heartbeat>select user()</heartbeat>
		<writeHost host="M1" url="10.9.48.69:3306"
		user="root" password="root">
		</writeHost>
		<writeHost host="M2" url="10.9.151.60:3306"
		user="root" password="root">
		</writeHost>
	</dataHost>
</mycat:schema>
```

4 读写分离逻辑属性

一个 dataHost 标签中,由 balance 和 writeType 控制读写逻辑.

writeType：实现写逻辑控制，二个值

0：默认值,在 index 下标为 0 的 writeHost 数据库中写数据

1：mycat1.5 版本以上的已经不推荐使用了,随机的在所有的 writeHost 进行写操作,覆盖 balance 的读逻辑,在所有**Host 进行随机的读

balance： 单独控制一个主从结构的读写分离的读逻辑,想让 balance 值生效,writeType 不能是 1

0：默认值,不开启读的分离,只会在 index=0 的 writeHost 进行读

1：读的操作,除了 index=0 的 writeHost,其他的所有 Host 都随机读.当读并发超级高时,所有后端是数据库读都承受很大压力时,第一个 writeHost 才会参与一部分读的分离.

2：随机在所有**Host 进行读

3：随机在所有的 ReadHost 进行读,没有 readHost 时,只会在第一个 writeHost 进行读

十 Mycat 故障转移

一个数据分片的读写工作，由于正在使用的某个节点出现故障，连接转向一个备份的节点实现读写

读写分离和故障转移都是基于主从结构实现(mycat 至少管理一个后端主从结构)

- 之前主从复制课程,实现了一主一从的数据库结构,测试读写分离,故障转移,为了测试效果,把搭建好的主从给断开
- 目的:通过高可用来提升一个数据分片的可靠性.

1 switchType 属性

dataHost 管理的主从数据库结构,可以通过 switchType 决定故障转移逻辑.

1：默认值,当正在通过写功能的 host(index=0 的 writeHost)故障,开启故障转移,将 index=1 的 writeHost 顶替(替换 index 下标).

-1：不开启故障转移

2 测试

通过前面读写分离案例,对第一个 writeHost 实现宕机,观察读写是否正常.

十一 Mycat 一致性 hash 算法

基础:hash 环

数据映射: 分片对象 dataNode 名字 字段值

对应关系:字段值的整数顺时针寻找最近 dataNode 整数

分片计算完成,后续还有读写分离计算.

十二　Mycat 跨分片的表格设计

数据库中很多个表格的时候,可以实现关联查询,比如商品表格和商品分类就可以设计成关联的表格.像这样的表格如果涉及到分片表格的配置,需要考虑底层数据是否跨分片.

mycat 不支持数据跨分片的,必须在业务层或者 mycat 配置中解决这个问题.

1　全局表

1.A　　应用场景

- tb_product：商品详情表
  - 保存了一个项目的所有商品数据，大表（数据非常多），在 mycat 应该配置成分片表格 。
- tb_category：商品分类表
  - 保存了当前电商项目的所有商品分类数据(京东大概 1000 多个商品分类)，不需要设计成分片表格，可以设计成非分片。
- 两张表格的关系

tb_product

id	product_name	c_id(分类 id 属于哪个分类)
1	乐事薯片	1
2	可口可乐	1
3	美年达	1
600w	海尔电视	2
601w	海信电视	2
602w	康佳彩电	2

tb_category

id	cat_name
1	食品饮料
2	电视

关联查询

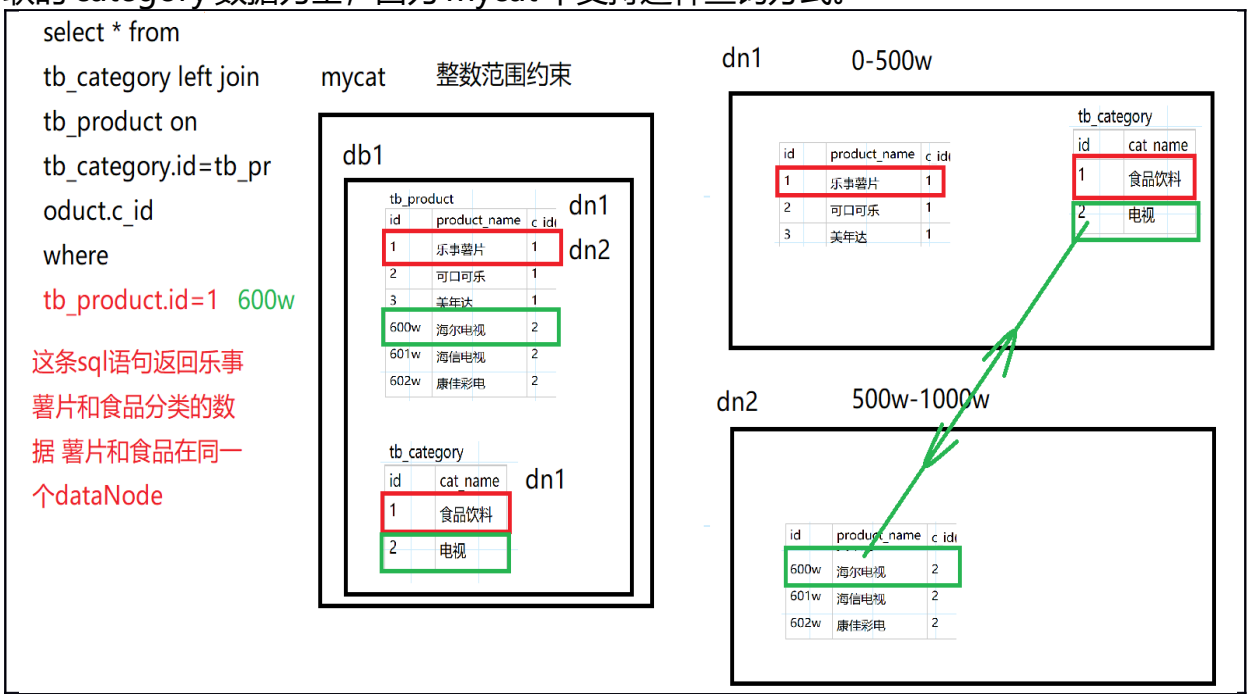
select * from tb_category left join tb_product on tb_category.id=tb_product.c_id

id	cat_name	id	product_name	c_id
1	食品饮料	1	乐事薯片	1

1	食品饮料	2	可口可乐	1
1	食品饮料	3	美年达	1
2	电视	600w	海尔电视	2
2	电视	601w	海信电视	2
2	电视	602w	康佳彩电	2

1.B 跨分片问题

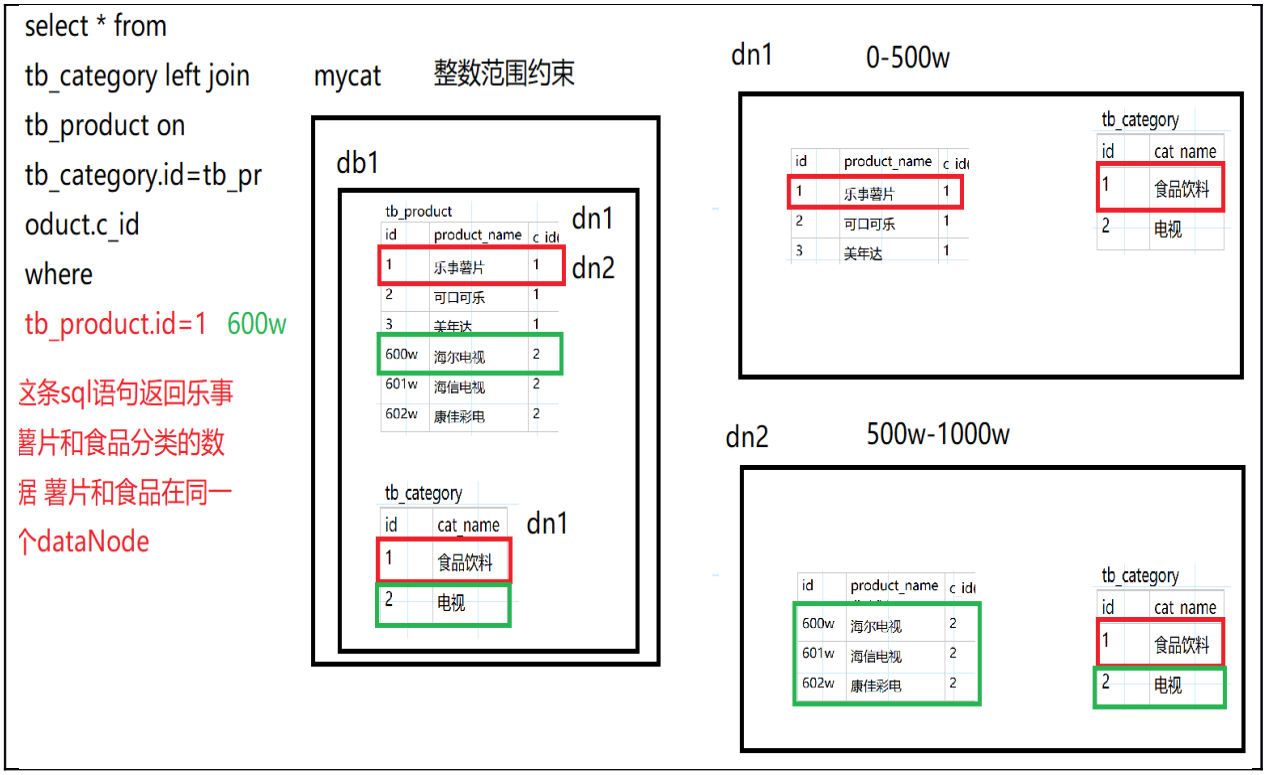
Category 表无需分片，那么他的数据写到第一个 writehost， 而 product 表数据太多需要分片，Mycat 计算数据分片时，根据 ID 计算，假如分到了二个数据库中，那么在 mycat 关联查询时，而关联的 category 数据在另一个数据库中，这样查询出来关联的 category 数据为空，因为 mycat 不支持这种查询方式。



由于 mycat 底层不支持跨分片的查询,所以对于上述一些数据 600w 601w 602w 这 3 个商品关联查询时没有返回的分类数据.

1.C解决办法

既然原因是 category 表只在一个分片中，那么将 category 表复制到所有分片不就行了吗？这就是全局表。



全局表配置：下列中 tb_category 就是全局表，type 的值为 global，数据复制到 dn1 和 dn2，而且没有分片既没有 rule 属性。

```
<table name="tb_product" primaryKey="id" dataNode="dn1,dn2"
rule="auto-sharding-long"/>
<table name="tb_category" primaryKey="id" dataNode="dn1,dn2"
type="global"/>
```

一张表格在配置 table 标签时,给指定了多个分片,但是没有指定 rule 计算分片规则,默认就会将表格新增数据同步到所有分片.

1.D 企业的全局表格

- 企业中使用 mycat 为了不出现跨分片的查询,将一些工具字典表设计成全局表格的形式,这样一大批业务表格关联查询就不需要考虑解决跨分片的问题
- 工具字典表:解释业务中一些数据的数据,特点是数据量稳定,变化不到,数据量不大.
- 比如:
  - 电信业务中,有记录日志字段 111119980707982556,这是业务表格的一个字段值,要想解释这个字段必须从 2 张工具字典表关联查询比如
    - 1111:地市代号-->江苏-南京
    - 982556:员工代号-->山东籍-济南电信分公司-56 号员工-刘首付

2 ER 分片表

2.A 应用场景

- tb_order:每一行数据都表示一个用户的某个订单
  - 保存了所有用户的所有订单数据
  - 订单数据量非常大--分片表格
- tb_order_item:每一行数据都表示一个订单中某个商品
  - 保存了电商项目的所有订单商品数据
  - 数据量更大--分片表格
- 两张表格的关系
  - 订单一行数据对应多个订单商品行数据(1 对多的关系)

tb_order

id	money	user_id
500w	800	a
800w	1500	b

tb_order_item

id	o_id	product_name
1	800w	海尔电视
2	800w	乐事薯片
3	800w	田格本(10 个)
601w	500w	面包
602w	500w	烤箱

- 关联查询 2 个表格

select * from tb_order left join tb_order_item on tb_order.id=tb_order_item.o_id where tb_order.id=500w

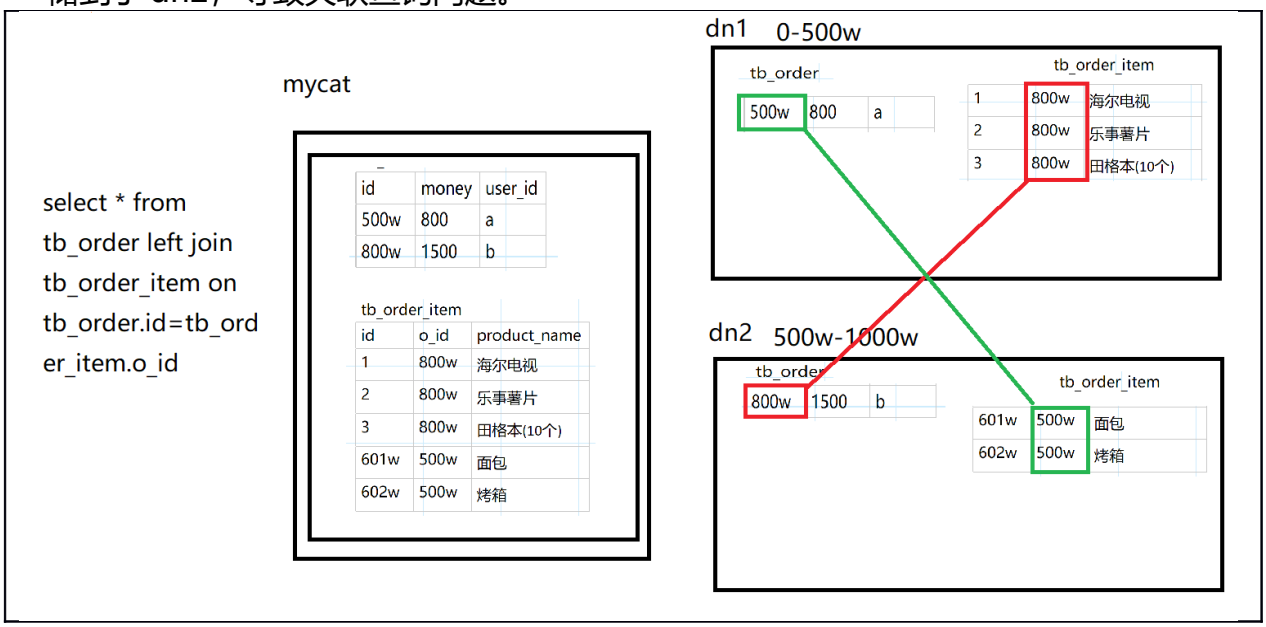
id	money	user_id	id	o_id	product_name
500w	800	a	601w	500w	面包
500w	800	a	602w	500w	烤箱

2.B 存在的问题

假如 mycat 分片计算逻辑为 id 为 1-500 存储到分片 dn1，最后二个关联表最后相关联的数据存放的数据分片不一致，导致了查询关联数据查询不到的问题。

见下图：mycat 计算时 tb_orderid 为 1-500W 存储到 dn1，同时 tb_order_item 的 id 为 1-3 也存储到 dn1，但是外键是 800W，对应 tb_orderid 存

储到了 dn2，导致关联查询问题。

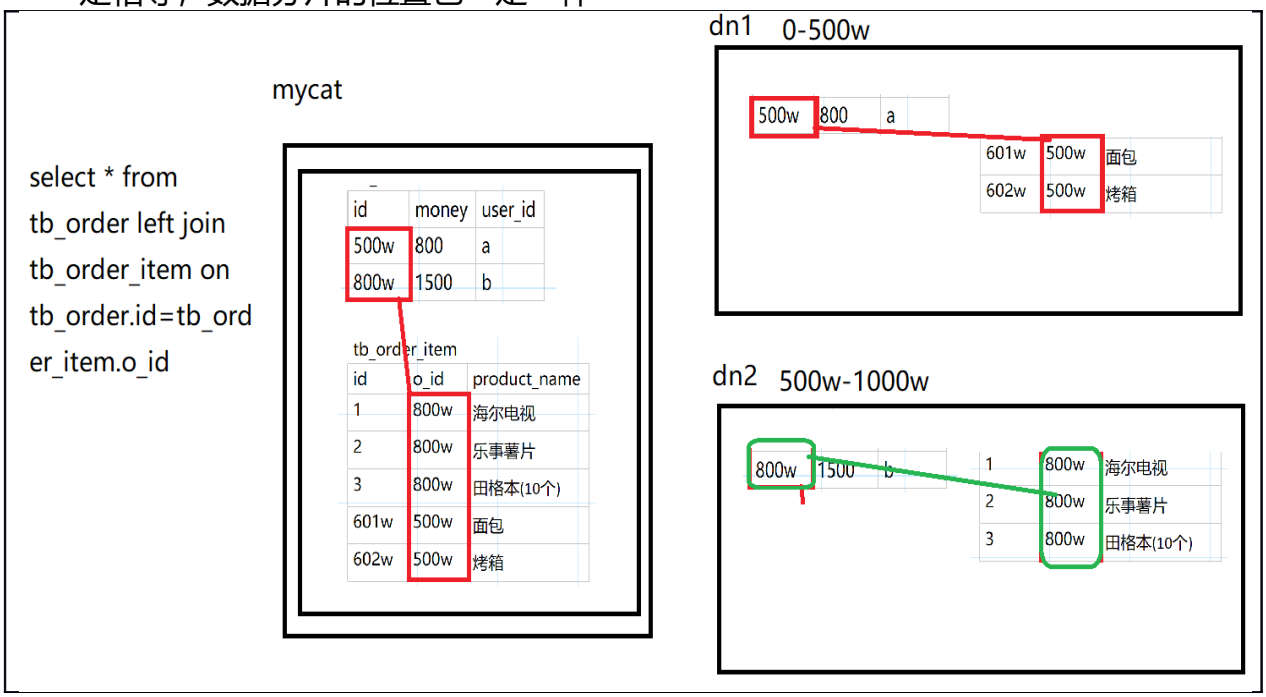


2.C解决办法

上面的二个表为一对多关系，一个订单（order）包括多个商品（order_item），在 mycat 中可认为是 order 为主表，order_item 为子表。

既然有关联关系的表记录被数据分片到了不同的 dn，那我想办法让他们在同一个 dn 不就行了吗?

主表用 id 计算分片，而子表用外键值计算分片，有关联关系的记录 id 值和外键一定相等，数据分片的位置也一定一样



- 关联表格的主子表关系
  - 由主表来定义分片计算规则,包括:分片个数,分片是谁,分片计算规则 rule
  - 子表跟随进行不再自定义这些内容

主子表区分:

tb_order 主表(父表)

tb_order_item 从表(子表)

父表的数据单独存在可以有意义

子表的数据单独存在没有意义

选择的计算分片的字段:订单 id(子表外键字段)

2.D ER 分片表配置

以上例中的 t_order 表和 t_order_item 表为例，t_order 主键为 order_id。t_order_item 表主键为 id，外键为 t_order 的 order_id，名字也叫 order_id。

- 定义计算分片字段：order_id
- 定义计算分片的方法：一致性 hash

2.D.a 配置 schema.xml

1 在 schema.xml 中添加一个 table 作为主表的配置即可实现 ER 分片表的使用,table 下需要包含一个子表的 childTable 标签.

```
<table name="t_order" primaryKey="order_id" dataNode="dn3,dn4"
rule="easymall-order-hash">
        <!--t_order_item childTable-->
        <childTable name="t_order_item" primaryKey="id"
joinKey="order_id"
        parentKey="order_id"/>
</table>
```

属性 **joinKey**：属于子表的外键关联字段(t_order_item 外键 order_id)

属性 **parentKey**：属于子表外键关联字段对应主表的字段名称(t_order order_id)

ER 分片表个配置中 2 个属性在 childTable 必须成对出现 ， 二个必须设置的意思？

2 添加 2 个 dataNode dn3 dn4

```
<dataNode name="dn3" dataHost="localhost1" database="easydb"/>
<dataNode name="dn4" dataHost="localhost2" database="easydb"/>
```

2.D.b 配置 rule.xml

- 添加一个专门为订单表格设计的一致性 hash 计算的 tableRule

```
<tableRule name="easymall-order-hash">
    <rule>
        <columns>order_id</columns>
        <algorithm>murmur</algorithm>
    </rule>
</tableRule>
```

以 order_id 计算

murmur:一致性 hash

3 局限性

购物车表格,用户表格,商品表格(订单,订单商品,商品,用户表格);

这些表格的总体结构都数据多对多,mycat 不能解决海量数据的多对多的关系.

以购物车,用户,商品表格为例;

2 个主表用户,商品

一个子表购物车(多对多外键关联的表格一样,同时存在 2 个主表的字段关联)

如果数据是海量,多对多在 mycat 中无法直接处理跨分片.

解决办法:

- 可以使用大数据管理数据关系,做数据清洗
- 从业务逻辑多次进行查询

十三 附录 A--sql 黑名单含义

<blacklist check="true">selelctAllow</blacklist>

配置项 缺省值 描述

selelctAllow true 是否允许执行 SELECT 语句

selectAllColumnAllow true 是否允许执行 SELECT * FROM T 这样的语句。如果设置为 false，不允许执行 select * from t，但可以 select * from (select id, name from t) a。这个选项是防御程序通过调用 select * 获得数据表的结构信息。

selectIntoAllow true SELECT 查询中是否允许 INTO 字句

deleteAllow true 是否允许执行 DELETE 语句

updateAllow true 是否允许执行 UPDATE 语句

insertAllow true 是否允许执行 INSERT 语句

replaceAllow true 是否允许执行 REPLACE 语句

mergeAllow true 是否允许执行 MERGE 语句，这个只在 Oracle 中有用

callAllow true 是否允许通过 jdbc 的 call 语法调用存储过程

setAllow true 是否允许使用 SET 语法

truncateAllow true truncate 语句是危险，缺省打开，若需要自行关闭
createTableAllow true 是否允许创建表
alterTableAllow true 是否允许执行 Alter Table 语句
dropTableAllow true 是否允许修改表
commentAllow false 是否允许语句中存在注释，Oracle 的用户不用担心，Wall 能够识别 hints 和注释的区别
noneBaseStatementAllow false 是否允许非以上基本语句的其他语句，缺省关闭，通过这个选项就能够屏蔽 DDL。
multiStatementAllow false 是否允许一次执行多条语句，缺省关闭
useAllow true 是否允许执行 mysql 的 use 语句，缺省打开
describeAllow true 是否允许执行 mysql 的 describe 语句，缺省打开
showAllow true 是否允许执行 mysql 的 show 语句，缺省打开
commitAllow true 是否允许执行 commit 操作
rollbackAllow true 是否允许执行 roll back 操作
##如果把
selectIntoAllow、deleteAllow、updateAllow、insertAllow、mergeAllow 都设置为 false，这就是一个只读数据源了。##
拦截配置 - 永真条件
selectWhereAlwayTrueCheck true 检查 SELECT 语句的 WHERE 子句是否是一个永真条件
selectHavingAlwayTrueCheck true 检查 SELECT 语句的 HAVING 子句是否是一个永真条件
deleteWhereAlwayTrueCheck true 检查 DELETE 语句的 WHERE 子句是否是一个永真条件
deleteWhereNoneCheck false 检查 DELETE 语句是否无 where 条件，这是有风险的，但不是 SQL 注入类型的风险
updateWhereAlayTrueCheck true 检查 UPDATE 语句的 WHERE 子句是否是一个永真条件
updateWhereNoneCheck false 检查 UPDATE 语句是否无 where 条件，这是有风险的，但不是 SQL 注入类型的风险
conditionAndAlwayTrueAllow false 检查查询条件(WHERE/HAVING 子句)中是否包含 AND 永真条件

conditionAndAlwayFalseAllow false 检查查询条件(WHERE/HAVING 子句)中是否包含 AND 永假条件

conditionLikeTrueAllow true 检查查询条件(WHERE/HAVING 子句)中是否包含 LIKE 永真条件

其他拦截配置

selectIntoOutfileAllow false SELECT ... INTO OUTFILE 是否允许，这个是 mysql 注入攻击的常见手段，缺省是禁止的

selectUnionCheck true 检测 SELECT UNION

selectMinusCheck true 检测 SELECT MINUS

selectExceptCheck true 检测 SELECT EXCEPT

selectIntersectCheck true 检测 SELECT INTERSECT

mustParameterized false 是否必须参数化，如果为 True，则不允许类似 WHERE ID = 1 这种不参数化的 SQL

strictSyntaxCheck true 是否进行严格的语法检测，Druid SQL Parser 在某些场景不能覆盖所有的 SQL 语法，出现解析 SQL 出错，可以临时把这个选项设置为 false，同时把 SQL 反馈给 Druid 的开发者。

conditionOpXorAllow false 查询条件中是否允许有 XOR 条件。XOR 不常用，很难判断永真或者永假，缺省不允许。

conditionOpBitwseAllow true 查询条件中是否允许有"&"、"~"、"|"、"^"运算符。

conditionDoubleConstAllow false 查询条件中是否允许连续两个常量运算表达式

minusAllow true 是否允许 SELECT * FROM A MINUS SELECT * FROM B 这样的语句

intersectAllow true 是否允许 SELECT * FROM A INTERSECT SELECT * FROM B 这样的语句

constArithmeticAllow true 拦截常量运算的条件，比如说 WHERE FID = 3 - 1，其中"3 - 1"是常量运算表达式。

limitZeroAllow false 是否允许 limit 0 这样的语句

禁用对象检测配置

tableCheck true 检测是否使用了禁用的表

schemaCheck true 检测是否使用了禁用的 Schema

functionCheck true 检测是否使用了禁用的函数
objectCheck true 检测是否使用了“禁用对对象”
variantCheck true 检测是否使用了“禁用的变量”
readOnlyTables 空 指定的表只读，不能够在 SELECT INTO、DELETE、UPDATE、INSERT、MERGE 中作