Mycat快速入門教程

JAVA葵花寶典 今天

作者：sf4y
来源：https://segmentfault.com/a/1190000022237732

基本原理

mycat是一個把自己偽裝成mysql服務的中間件,推薦閱讀Mycat權威指南官方下載[：下載地址]：https://github.com/MyCATApache/Mycat-Server/blob/4135f25df8239d52d220529cbf7cb697ede40e12/mycat-definitive-guide.pdf

安裝

下載安裝包解壓即用[點擊我下載]：http://dl.mycat.io/ ,下載對應的版本：Mycat-server-1.6-RELEASE-20161028204710-linux.tar.gz,建議安裝在usr/local/mycat,解壓到當前目錄，目錄結構：

```
bin catlet conf lib logs version.txt
```

環境變量配置MYCAT_HOME，vim /etc/profile添加MYCAT_HOME=/usr/local/mycat

運行

linux：

./mycat start 啟動

./mycat stop 停止

./mycat console 前台運行

./mycat restart 重啟服務

./mycat pause 暫停

./mycat status 查看啟動狀態

windows：

startup_nowrap.bat

基本配置：

启动需要配置一些jvm 参数，这个根据系统的需要来设置，打开conf/wrapper.conf文件，里面有一些我们平时常用的jvm参数,一般只要调整Xmx、Xms、MaxDirectMemorySize 内存大小就可以, JVM 参数，必须设置- XX:MaxDirectMemorySize 和 -Xmx 例如:-Xmx1024m -Xmn512m -XX:MaxDirectMemorySize=2048m -Xss256K -XX:+UseParallelGC了, MaxDirectMemorySize 尽可能设置大些，可以加快结果集处理时间

```
# Java Additional Parameters
wrapper.java.additional.1=-DMYCAT_HOME=. wrapper.java.additional.2=-server wrapper.java.additional.3=-XX:MaxPermSize=64M wrapp
```

规则配置

Mycat最重要的3大配置文件：

- server.xml

主要有user 和 system 标签。这个user标签主要用于定义登录 mycat的用户和权限。例如上面的例子中，我定 义了一个用户，用户名为 mycat、密码也为 mycat，可访问的 schema 也只有 TESTDB 一个。如果我在 schema.xml 中定义了多个 schema，那么这个用户是无法访问其他的 schema。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE mycat:server SYSTEM "server.dtd">
<mycat:server xmlns:mycat="http://org.opencloudb/">
<user name="mycat">
    <property name="password">mycat</property>
    <property name="schemas">TESTDB</property>
</user>
</mycat:server>
```

同时访问多个 schema 的话使用 , 隔开，例如:

```
<property name="schemas">TESTDB,db1,db2</property>
```

system 这个标签内嵌套的所有 property 标签都与系统配置有关，如果没有特殊需求默认即可

```
<system>
<property name="charset">utf8</property>
<!--这个属性主要用于指定系统可用的线程数，默认值为机器 CPU 核心线程数。-->
<property name="processors">1</property>
<!--这个属性主要用于指定 NIOProcessor 上共享的 businessExecutor 固定线程池大小。
mycat 在需要处理一 些异步逻辑的时候会把任务提交到这个线程池中。
新版本中这个连接池的使用频率不是很大了，可以设置一个较 小的值。-->
<property name="processorExecutor">32</property>
<!--分布式事务开关，0为不过滤分布式事务，1为过滤分布式事务
（如果分布式事务内只涉及全局表，则不过滤），2为不过滤分布式事务,
但是记录分布式事务日志-->
<property name="handleDistributedTransactions">0</property>
...
</system>
```

- schema.xml

  文件地址 mycat/conf/schema.xml，这个xml 是mycat配置的重头戏，里面有几个很关键的标签 schema table dataNode dataHost 等等。首先配置schema 标签:

  table : schema标签中包含了table 属性，mycat中table有2种类型，一种是全局广播表，一种是分片表，如果没有在这里面配置的表呢，schema 标签有个 dataNode 属性，没有配置默认读写在这个库里面, table中还有一个属性 subTables，是指把一个表拆分成多个子表，例如下面表示有3个表 t_order1，t_order2，t_order3

```
subTables="t_order$1-2,t_order3"
```

目前分表 1.6 以后开始支持 并且 dataNode 在分表条件下只能配置一个，分表条件下不支持各种条件的 join 语句, 这种适合单个数据库需要分表的情况。

dataNode：标签定义了 MyCat 中的数据节点，也就是我们通常说所的数据分片 。一个 dataNode 标签就是 一个独立的数据分片

dataHost：该标签在 mycat 逻辑库中也是作为最底层的标签存在，直接定义了具 体的数据库实例、读写分离配置和心跳语句。writeHost 标签、readHost 标签, writeHost 指 定写实例、readHost 指定读实例, 用来做读写分离。

```
<!-- 配置默认的name逻辑库名称，checkSQLschema属性为false的时候，sql查询是会带上逻辑库的名称TESTDB.tableName，
    如果不想带上逻辑库名称，设置为true，sqlMaxLimit表示分页最大limit，dataNode表示没有分片的表默认使用这个库-->
    <schema name="TESTDB" checkSQLschema="false" sqlMaxLimit="100" dataNode="dn1">
        <!-- mycat中table有2种类型，一种是全局广播表，一种是分片表，type="global" 表示全局表，会同步到所有的库，一般是用于数据字典表，
        方便join 操作，分片表只type不用指定，默认就是分片，需要配置分片规则， primaryKey对应逻辑表对应真实表的主键，
        例如:分片的规则是使用非主键进行分片的，那么在使用主键查询的时候，就会发送查询语句到所有配置的 DN 上，如果使用
        该属性配置真实表的主键。那么 MyCat 会缓存主键与具体 DN 的 信息，那么再次使用非主键进行查询的时候就不会进行广播式的查询，
        -->
        <table name="travelrecord" dataNode="dn1,dn2" rule="auto-sharding-long" />
        <table name="goods" primaryKey="ID" type="global" dataNode="dn1,dn2" />

        <!-- 设置dataNode 对应的数据库,及 mycat 连接的地址dataHost 对应dataHost 标签上定义的 name 属性，使用名字为 dh01 数据库实例上的 db1 
        <dataNode name="dn01" dataHost="dh01" database="db01" />
        <dataNode name="dn02" dataHost="dh02" database="db02" />
        <!-- balance为0 不开启读写分离机制，writeType="0"所有写操作发送到配置的第一个 writeHost,switchType=1 默认值，自动切换 -->
        <dataHost name="dh01" writeType="0" switchType="1" balance="0" dbType="mysql" maxCon="1000" minCon="10" dbDriver="nati
            <heartbeat>select user()</heartbeat>
            <writeHost host="hostM1" url="127.0.0.1:3306" user="root" password="root" >
            <!-- 配置读写分离， can have multi read hosts -->
            <readHost host="hostS1" url="192.168.1.200:3306" user="root" password="123456" />
            </writeHost>
        </dataHost>
        <!-- 如果是用多台数据库实例，需要配置多个dataHost，如果只是单实例多个数据库，只要配置一个就可以了 -->
        <dataHost name="dh02" writeType="0" switchType="1" balance="0" dbType="mysql" maxCon="1000" minCon="10" dbDriver="nati
            <heartbeat>select user()</heartbeat>
            <writeHost host="hostM2" url="192.168.11.123:3306" user="root" password="root" >
            </writeHost>
        </dataHost>
    </schema>
```

- rule.xml

/usr/local/mycat/conf/rule.xml

这个标签定义表规则，这个文件里面主要有 tableRule 和 function 这两个标签。tableRule标签中的 name 对应上面table标签中的rule属性，columns 内指定要拆分的列名字。algorithm 使用 function 标签中的 name 属性。

```
<tableRule name="rule1"> <rule>
<columns>id</columns>
<algorithm>func1</algorithm> </rule>
</tableRule>
```

name 指定算法的名字。class 制定路由算法具体的类名字。

```
<function name="hash-int"class="io.mycat.route.function.PartitionByFileMap">
    <property name="mapFile">partition-hash-int.txt</property>
</function>
```

Mycat 常用的分片规则

这个是本次分库分表的核心所在，直接影响到后续的sql执行效率，所以在分片规则的选择上，需要对表有比较清楚的认识

1. 分片枚举

通过在配置文件中配置可能的枚举 id，自己配置分片，本规则适用于特定的场景，比如有些业务需要按照省 份或区县来做保存，而全国省份区县固定的，这类业务使用本条规则

```
<tableRule name="sharding-by-intfile"> <rule>
<columns>user_id</columns> <algorithm>hash-int</algorithm> </rule>
</tableRule>
<function name="hash-int"class="io.mycat.route.function.PartitionByFileMap">
<property name="mapFile">partition-hash-int.txt</property> <property name="type">0</property>
<property name="defaultNode">0</property>
</function>
```

```
partition-hash-int.txt 配置:
10000=0
10010=1
DEFAULT_NODE=1
```

其中分片函数配置中，mapFile 标识配置文件名称，type 默认值为 0，0 表示 Integer，非零表示 String， 所有的节点配置都是从 0 开始，及 0 代表节点 1 /**

- defaultNode 默认节点:小于 0 表示不设置默认节点，大于等于 0 表示设置默认节点 * 默认节点的作用:枚举分片时，如果碰到不识别的枚举值，就让它路由到默认节点
- 如果不配置默认节点(defaultNode 值小于 0 表示不配置默认节点)，碰到
- 不识别的枚举值就会报错，
- like this:can't find datanode for sharding column:column_nameval:ffffffff */

2. 固定分片 hash 算法

本条规则类似于十进制的求模运算，区别在于是二进制的操作,是取 id 的二进制低 10 位，即 id 二进制 &1111111111。此算法的优点在于如果按照 10 进制取模运算，在连续插入 1-10 时候 1-10 会被分到 1-10 个分片，增 大了插入的事务控制难度，而此算法根据二进制则可能会分到连续的分片，减少插入事务事务控制难度。

```
<tableRule name="rule1"> <rule> <columns>user_id</columns> <algorithm>func1</algorithm> </rule>
</tableRule>
<function name="func1"class="io.mycat.route.function.PartitionByLong">
<!-- dataNote 个数，这边表示3个-->
<property name="partitionCount">2,1</property>
<property name="partitionLength">256,512</property>
</function>
```

分区长度:默认为最大 2^n=1024 ,即最大支持 1024 分区 约束: count,length 两个数组的长度必须是一致的。1024 = sum((count[i]*length[i])). count 和 length 两个向量的点积恒等于 1024 ，例如: 2*256 + 1*512 = 1024。

例如：*以上分为二个分区:0-512,512-1023*

1023的二进制&1111111111运算后为1023，故落入第二个分区

1024的二进制&1111111111运算后为0，故落入第一个分区

0266 的二进制&1111111111运算后为266，故落入第一个分区内

如果只需要平局分配，配置如下

```
<!--平均分为 4 分片，partitionCount*partitionLength=1024 -->
```

```
<function name="func1"class="io.mycat.route.function.PartitionByLong"> <property name="partitionCount">4</property>
<property name="partitionLength">256</property>
</function>
```

3. 范围约定

此分片适用于，提前规划好分片字段某个范围属于哪个分片

```
<tableRule name="auto-sharding-long"> <rule>
<columns>user_id</columns>
<algorithm>rang-long</algorithm>
</rule>
</tableRule>
<function name="rang-long"class="io.mycat.route.function.AutoPartitionByLong"> <property name="mapFile">autopartition-long.txt
<property name="defaultNode">0</property> </function>
```

rang-long 函数中 mapFile 代表配置文件路径 defaultNode 超过范围后的默认节点。所有的节点配置都是从 0 开始，及 0 代表节点 1，此配置非常简单，即预先制定可能的 id 范围到某个分片

```
0-500M=0
500M-1000M=1
1000M-1500M=2
或
0-10000000=0
10000001-20000000=1
```

4. 取 模

此种配置非常明确即根据 id 进行十进制求模预算，相比固定分片 hash，此种在批量插入时可能存在批量插入单 事务插入多数据分片，增大事务一致性难度。

```
<tableRule name="mod-long"> <rule>
<columns>user_id</columns> <algorithm>mod-long</algorithm> </rule>
</tableRule>
<function name="mod-long"class="io.mycat.route.function.PartitionByMod"> <!-- how many data nodes -->
<property name="count">2</property>
</function>
```

1. 按日期(天)分片

按天分片

```
<tableRule name="sharding-by-date"> <rule>
<columns>create_time</columns>
<algorithm>sharding-by-date</algorithm>
</rule>
</tableRule>
<function name="sharding-by-date"class="io.mycat.route.function.PartitionByDate">
<property name="dateFormat">yyyy-MM-dd</property>
<property name="sBeginDate">2014-01-01</property>
<property name="sEndDate">2014-01-02</property>
<property name="sPartionDay">10</property> </function>
```

配置说明: dateFormat :日期格式

sBeginDate :開始日期

sEndDate:結束日期

sPartionDay :分區天數，即默認從開始日期算起，分隔10 天一個分區如果配置了sEndDate 則代表數據達到了這個日期的分片後後循環從開始分片插入

1. 取模範圍約束
2. 截取數字做hash 求模範圍約束
3. 應用指定
4. 截取數字hash 解析
5. 一致性hash
6. 按單月小時拆分

....

日誌排查

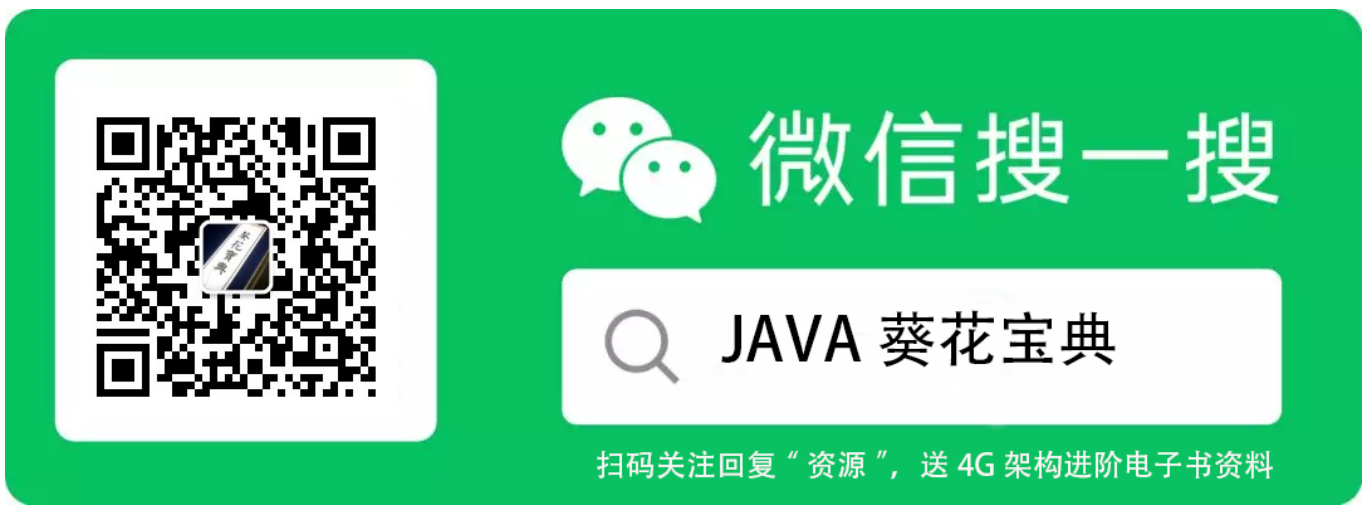
設置conf/log4j2.xml,設置mycat日誌地址

```
<RollingFile name="RollingFile" fileName="${sys:MYCAT_HOME}/logs/mycat.log"
                     filePattern="${sys:MYCAT_HOME}/logs/${date:yyyy-MM}/mycat-%d{MM-dd}-%i.log.gz">
```

後面可以在logs 中查看mycat.log 文件

推薦閱讀

日誌規範多重要，這篇文章告訴你！
IDEA依賴衝突分析神器—Maven Helper
我把SpringBoot的banner換成了美女，老闆說工作不飽和，建議安排加班
哪些數據可以放進緩存？記錄生產環境一次緩存評估的過程



點個在看少個 bug

閱讀原文