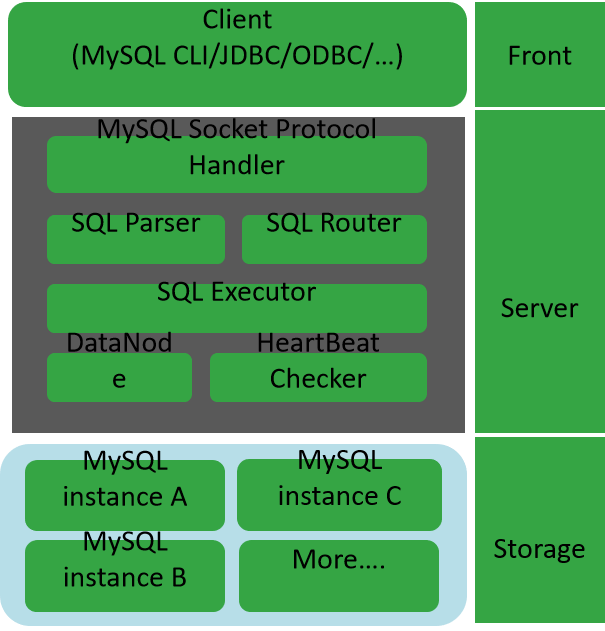
数据库中间件

需求：

1. 实现分库分表，读写分离
2. 提升数据库性能，应用层透明

## 一、Mycat

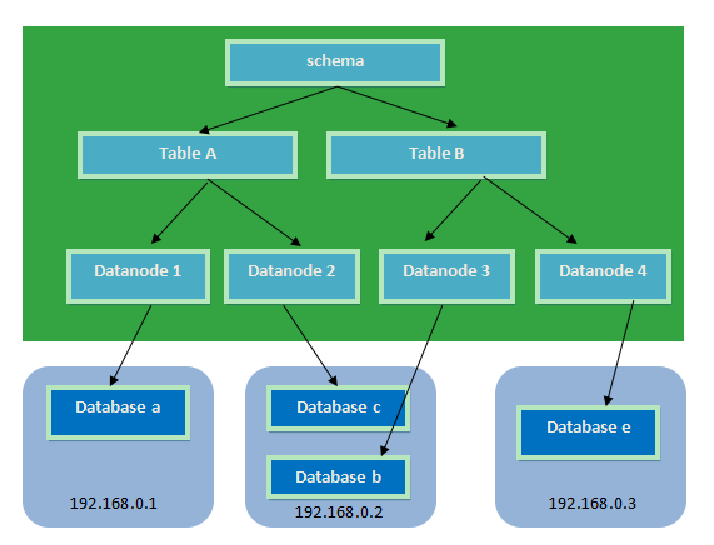
### 1.Mycat架构



### 2.工作原理

Mycat的原理中最重要的一个动词是“拦截”，它拦截了用户发送过来的SQL语句，首先对SQL语句做了一些特定的分析：如分片分析、路由分析、读写分离分析、缓存分析等，然后将此SQL发往后端的真实数据库，并将返回的结果做适当的处理，最终再返回给用户。

### 3.分表分库



备注：

MyCAT通过定义表的分片规则来实现分片，每个表格可以捆绑一个分片规则，每个分片规则指定一个分片字段并绑定一个函数，来实现动态分片算法。

1、Schema：逻辑库，与MySQL中的Database（数据库）对应，一个逻辑库中定义了所包括的Table。

2、Table：表，即物理数据库中存储的某一张表，与传统数据库不同，这里的表格需要声明其所存储的逻辑数据节点DataNode。在此可以指定表的分片规则。

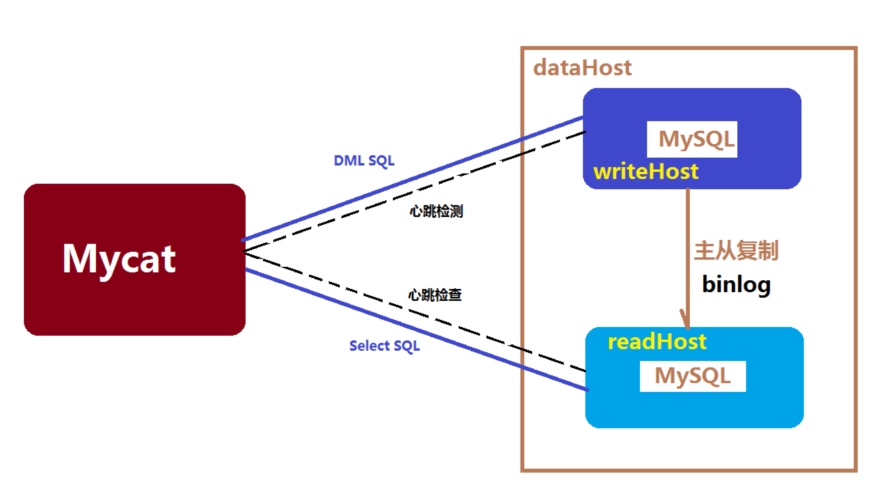
3、DataNode：MyCAT的逻辑数据节点，是存放table的具体物理节点，也称之为分片节点，通过DataSource来关联到后端某个具体数据库上

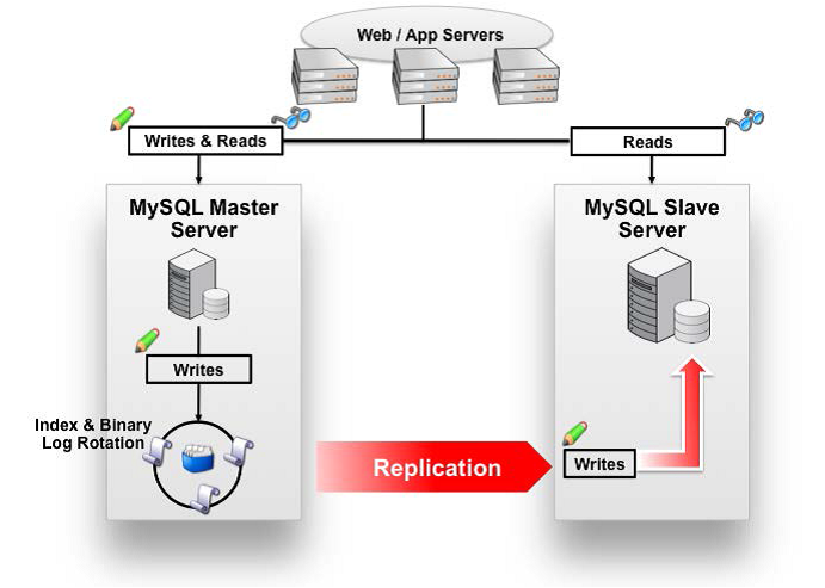
4、DataSource：定义某个物理库的访问地址，用于捆绑到Datanode上

### 4.分片规则

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.
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。
3. 按日期分片 此规则为按天分片。 按单月小时拆分 此规则是单月内按照小时拆分，最小粒度是小时，可以一天最多 24 个分片，最少 1 个分片，一个月完后下月从头开始循环。每个月月尾，需要手工清理数据。
4. 截取数字 hash 解析 此规则是截取字符串中的 int 数值 hash 分片。
5. 日期范围 hash 分片 思想与范围求模一致，当由于日期在取模会有数据集中问题，所以改成 hash 方法。 先根据日期分组，再根据时间 hash 使得短期内数据分布的更均匀。 优点可以避免扩容时的数据迁移，又可以一定程度上避免范围分片的热点问题。要求日期格式尽量精确些，不然达不到局部均匀的目的
6. 一致性 hash 一致性哈希主要应用于分布式集群对机器添加、删除的管理

### 5.读写分离





备注： 需要mysql自身的主从复制机制配合

### 6.mycat分布式事务解决方案

支持分布式事务（弱xa）

### 7.关键特性

1. 遵守Mysql原生协议，跨语言，跨平台，跨数据库的通用中间件代理。
2. 基于心跳的自动故障切换，支持读写分离，支持MySQL主从，以及galera cluster集群。
3. 支持Galera for MySQL集群，Percona Cluster或者MariaDB cluster
4. 基于Nio实现，有效管理线程，高并发问题。
5. 支持数据的多片自动路由与聚合，支持sum,count,max等常用的聚合函数。
6. 支持单库内部任意join，支持跨库2表join，甚至基于caltlet的多表join。
7. 支持通过全局表，ER关系的分片策略，实现了高效的多表join查询。
8. 支持多租户方案。
9. 支持分布式事务（弱xa）。
10. 支持全局序列号，解决分布式下的主键生成问题。
11. 分片规则丰富，插件化开发，易于扩展。
12. 强大的web，命令行监控。
13. 支持前端作为mysq通用代理，后端JDBC方式支持Oracle、DB2、SQL Server 、 mongodb 、巨杉。
14. 支持密码加密
15. 支持服务降级
16. 支持IP白名单
17. 支持SQL黑名单、sql注入攻击拦截
18. 支持分表（1.6）
19. 集群基于ZooKeeper管理，在线升级，扩容，智能优化，大数据处理（2.0开发版）。

### 8.mycat不适用场景

1. 非分片字段查询 如果该分片字段选择度高，也是业务常用的查询维度，一般只有一个或极少数个DB节点命中（返回结果集）。示例中只有3个DB节点，而实际应用中的DB节点数远超过这个，假如有50个，那么前端的一个查询，落到MySQL数据库上则变成50个查询，会极大消耗Mycat和MySQL数据库资源。
2. 分页排序 但Mycat向应用返回的结果集取决于哪个DB节点最先返回结果给Mycat。如果Mycat最先收到DB1节点的结果集，那么Mycat返回给应用端的结果集为 [0,1]，如果Mycat最先收到DB2节点的结果集，那么返回给应用端的结果集为 [5,6]。也就是说，相同情况下，同一个SQL，在Mycat上执行时会有不同的返回结果。
3. 任意表JOIN 无法跨库join
4. 分布式事务 Mycat并没有根据二阶段提交协议实现 XA事务，而是只保证 prepare 阶段数据一致性的 弱XA事务

### 9.mycat生产应用

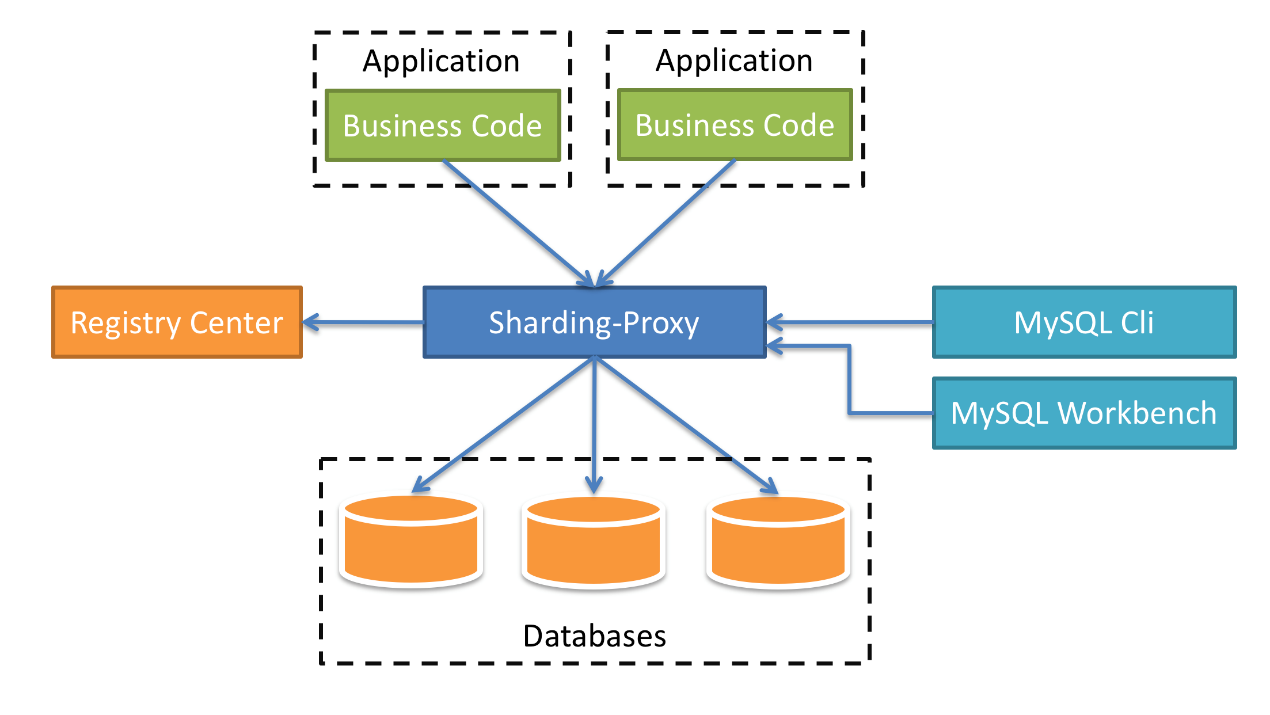
#### IMG_256

Mysql 的 proxy 中间件有比较多的工具，例如，mysql-proxy, atlas , cobar, mycat, tddl, tinnydbrouter等等。相比较而言，Mycat并不是最快的，但是功能比较丰富，对读写分离和分库分表都有支持，而且比较易用，对原有的应用系统侵入比较小，系统改造比较易于实现。

，由于Mycat本身需要解析sql，也需要合并各个数据库返回的结果，本身的CPU消耗就会比较高，在数据库较多的情况下，CPU可能不堪重负。

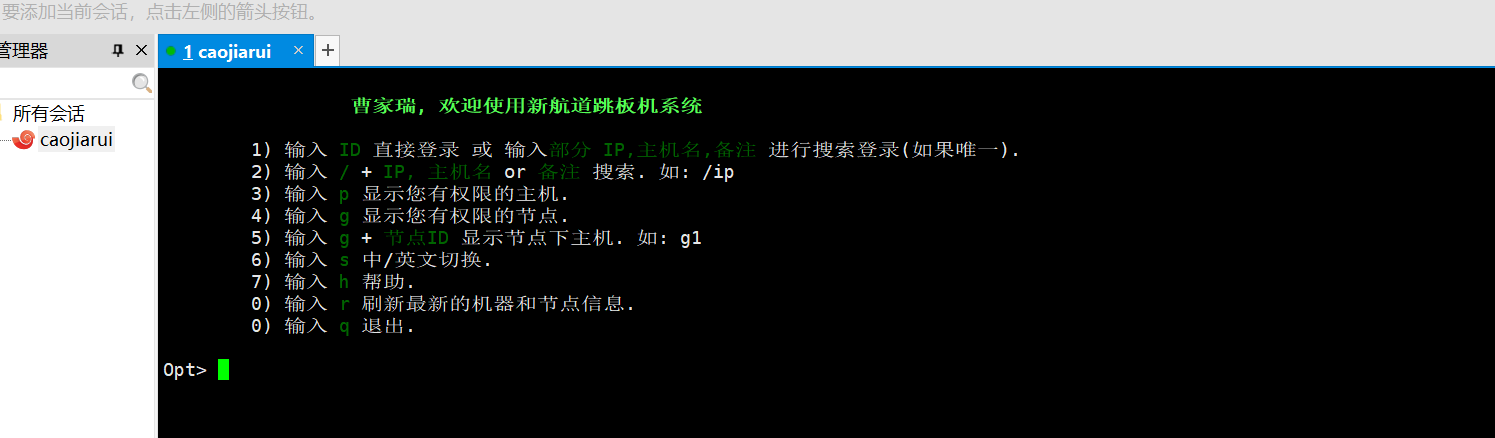
## 二、Sharding-proxy

### 1.应用部署

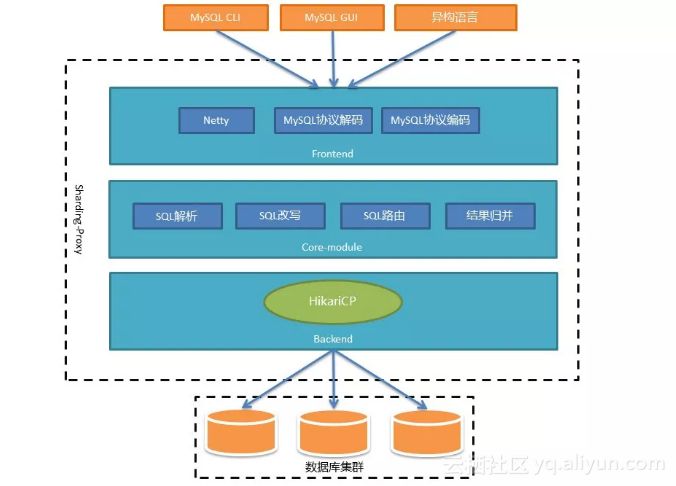


IMG_256对应用程序完全透明，可直接当做MySQL使用；

IMG_257适用于任何兼容MySQL协议的客户端。



### 系统架构



整个架构可以分为前端、后端和核心组件三部分来看：

IMG_256前端（Frontend）负责与客户端进行网络通信，采用的是基于NIO的客户端/服务器框架，在Windows和Mac操作系统下采用NIO模型，Linux系统自动适配为Epoll模型，在通信的过程中完成对MySQL协议的编解码；  
IMG_257核心组件（Core-module）得到解码的MySQL命令后，开始调用Sharding-Core对SQL进行解析、改写、路由、归并等核心功能；

IMG_258后端（Backend）与真实数据库的交互暂时借助基于BIO的Hikari连接池。BIO的方式在数据库集群规模很大，或者一主多从的情况下，性能会有所下降。所以未来我们还会提供NIO的方式连接真实数据库。

## 三、sharding-proxy、mycat比较

目前从调研的两个中间件mycat 和sharding-proxy来看,基本上都能实现读写分离和分库分表功能。mycat整体功能要比sharding-proxy更全面一些，但也存在一些问题，如：不支持分页，分布式事务等

|  |  |  |
| --- | --- | --- |
|  | **sharding-proxy** | **mycat** |
| 数据源 | Mysql | MySQL，Oracle，SQLServer，PostgreSQL等多种数据源 |
| 代理方式 | db-proxy | db-proxy |
| 集成方式 | 独立部署，对应用层透明 | 独立部署，对应用层透明 |
| 分布式事务 | 支持LOCAL,BASE,XA事务 | 支持弱XA事务 |
| SQL支持 | Insert…select; having; union; 不支持跨库关联 | 不支持跨分片多表join，不支持分页 |
| 分片规则 | 路由策略和分片算法可自定义 | 不支持二维分片路由 |
| 其他特点 | 扩展性较强，数据脱敏，弹性伸缩，Sharding-Scaling数据迁移 | 高可用，主从切换，安全机制，多租户应用、支持NOSQL |
| 监控UI | Sharding-ui | Mycat-eay |
| 社区 | 社区比较活跃，更新版本较快 | 目前不更新 |

目前从调研的两个中间件mycat 和sharding-proxy来看,基本上都能实现读写分离和分库分表功能。Mycat相对实现的功能比较全面一些，可以提供负载均衡和主从切换等功能，安全机制也提供了相应的配置，但目前无法支持二维分片，分布式事务和路由分页等功能。

## mycat测试

### mysql部署

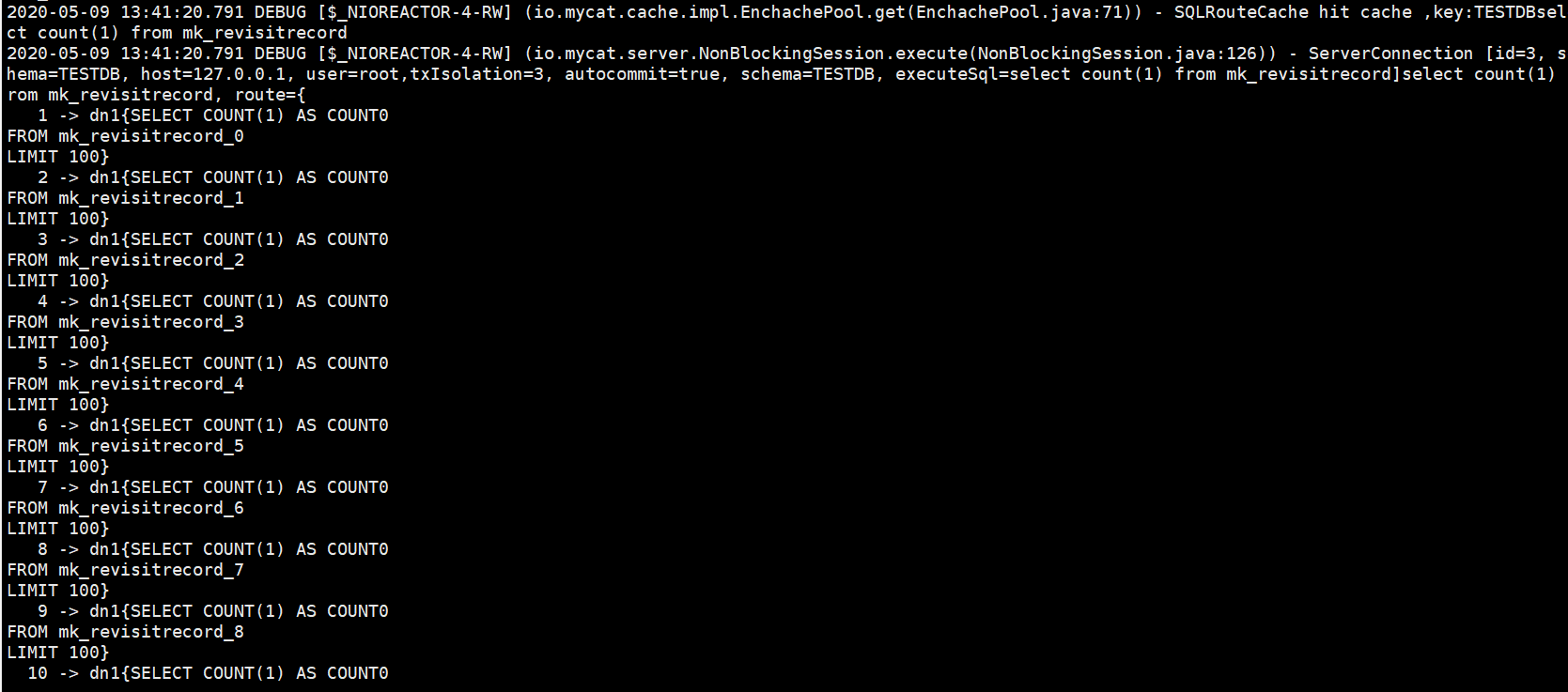
mysql架构：

master： 192.168.1.21

slave1: 192.168.1.19

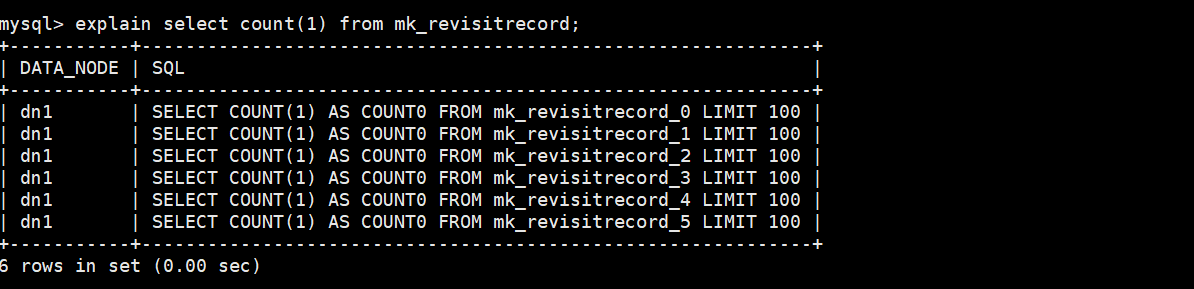
slave2: 192.168.1.20

### Mycat 路由



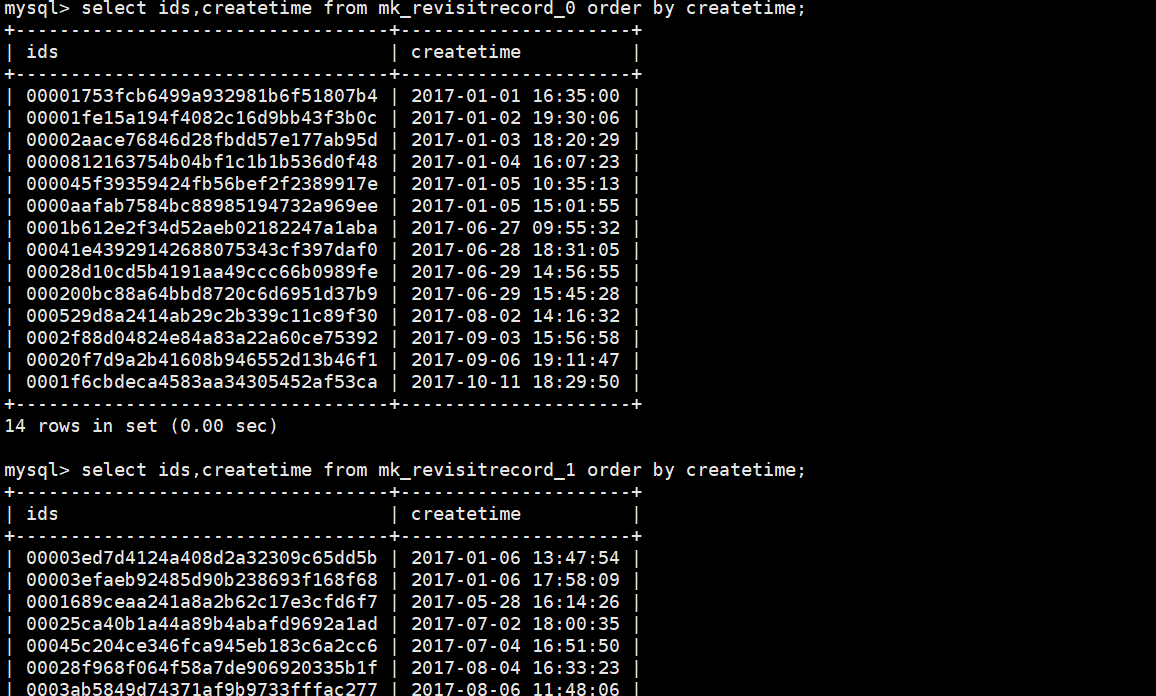
select count(1) from mk\_revisitrecord;

--needAddLimit: mycat 默认limit100;



### mycat分片

1. 按天分片

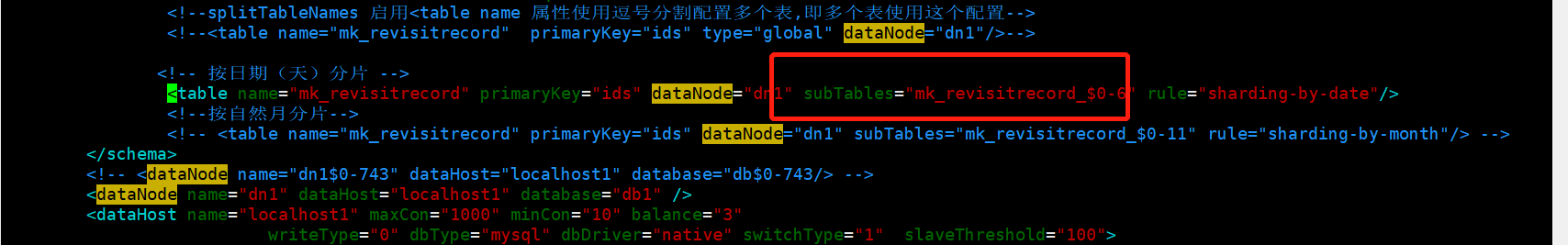


踩坑：

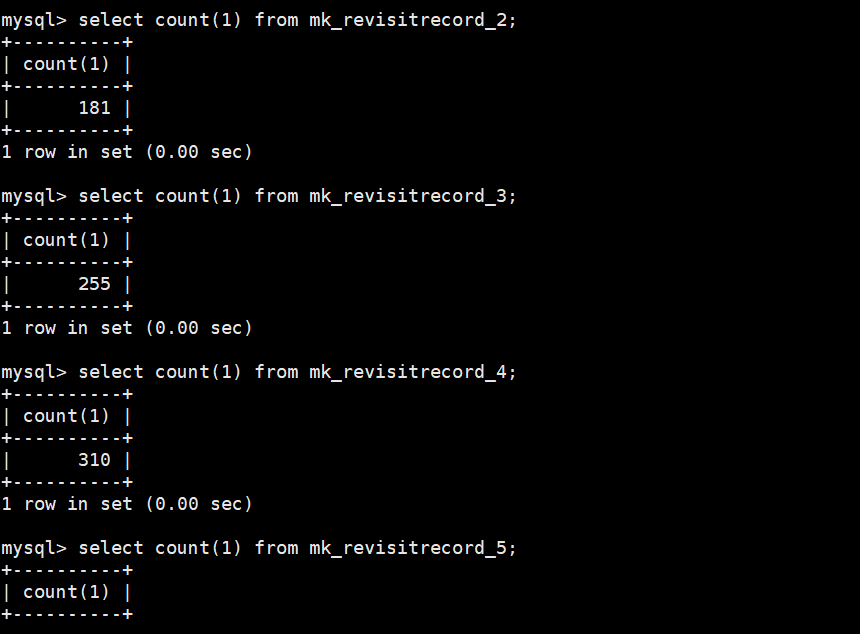
table [ MK\_REVISITRECORD ] rule function [ partbyday ] partition size : 7 > table datanode size : 1, please make sure table datanode size = function partition size

rule.xml

|  |
| --- |
| <function name="partbyday"  class="io.mycat.route.function.PartitionByDate">  <property name="dateFormat">yyyy-MM-dd</property>  <property name="sNaturalDay">0</property>  <property name="sBeginDate">2014-01-01</property>  <property name="sEndDate">2014-01-31</property>  <property name="sPartionDay">10</property>  </function> |

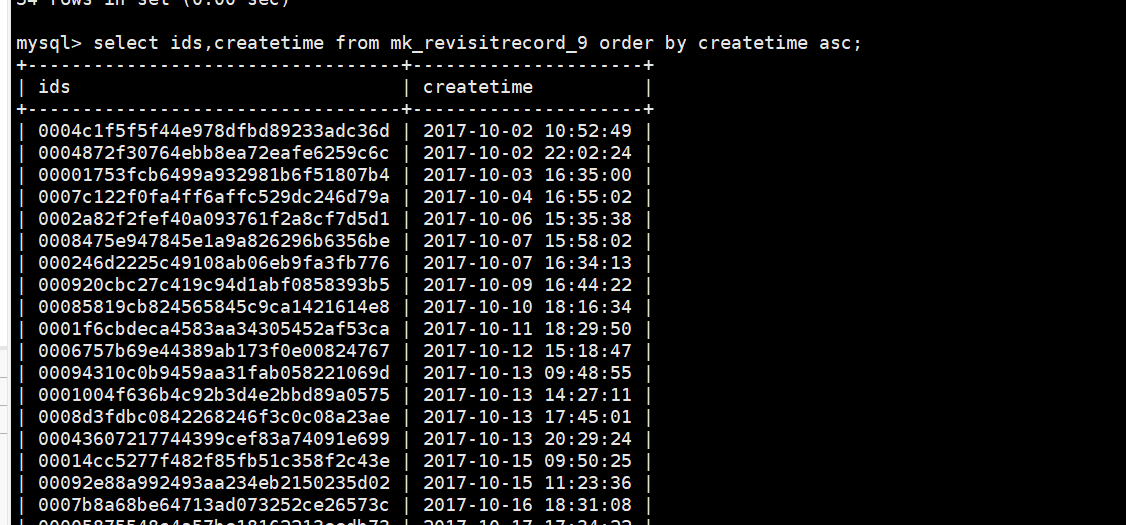


1. subTables 个数 =（sEndDate - sBeginDate）/ sPartionDay
2. 设置sEndDate结束日期可以重复利用分片

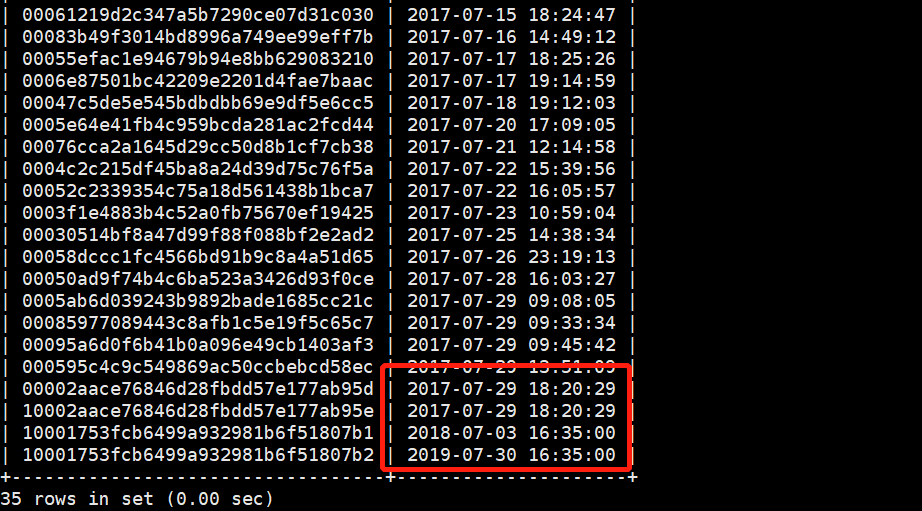


1. 按自然月分片

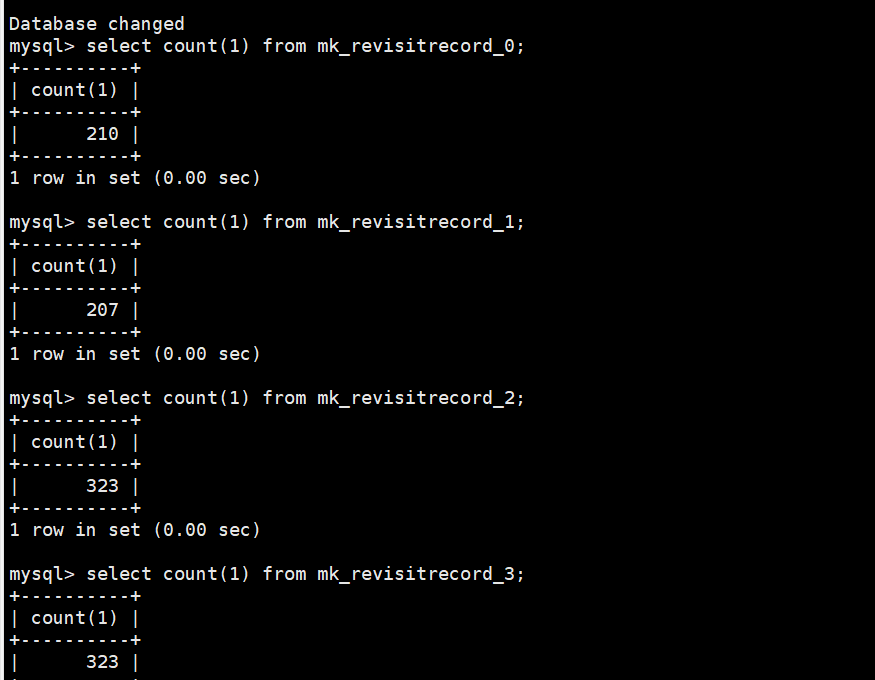
rule = “sharding-by-month” 每个自然月一个分片,需要提前将分片数规划好，建好，否则有可能日期超出实际配置分片数

 如果按月分片数量对，及报异常：ERROR 1064 (HY000): Index: 33, Size: 12

解决方案：在rule.xml中配置sEndDate结束日期，分片即可重复利用



1. 日期范围去取模hash 算法



优点：方便日后扩容，数据迁移

缺点：范围查询可能会路由多表