# 9.数据库高可用

## 9.1.数据库设计

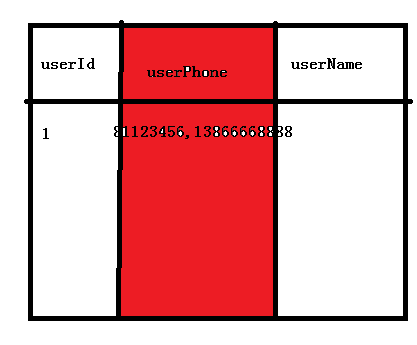
订单业务，需要数据库的高并发支持，不能采用传统的单数据库节点模式，需要数据库集群

### 9.1.1.三范式设计

早期数据库使用，需要节省空间，通过对表结构设计进行节省空间操作，需要利用三范式设计

三范式可以最大程度节省空间（浪费时间）

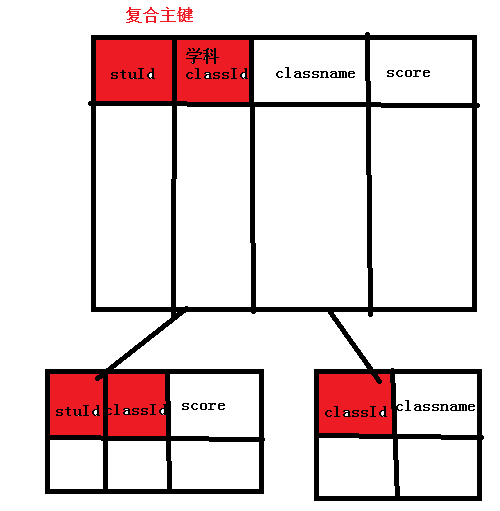
第一范式：数据库字段不可拆分



userPhone违反了第一范式，因为电话号码可以分为座机和手机两个字段

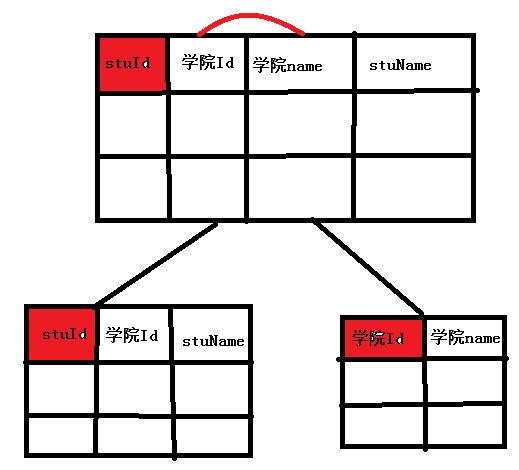
第二范式：满足第一范式前提，所有非主键字段，必须依赖全部的注解（复合主键）

easymall的购物车就违反了此范式



上图违反了三范式的第二范式，classname和stuId无关，可以拆分成两个表

第三范式：第二范式前提下，非主键字段必须全部依赖主键字段，不能依赖其他非主键字段



上图违反了第三范式设计，因为学院name依赖的是学院Id，依赖的不是主键

需要进行拆分

一旦严格按照三范式设计，每个表维护全局唯一的一批内容，节省了空间

### 9.1.2.反范式

随着时代发展，系统对响应速度要求越来越高，空间的成本越来越低，设计时需要用空间换时间，必不可少会出现冗余字段--反范式设计

例如easymall的购物车表

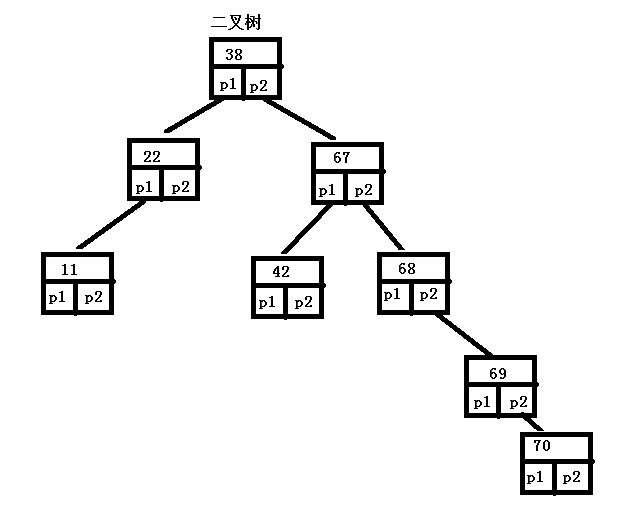
### 9.1.3.索引

定义：一种具有固定规则（排序），具有数据结构的，存储在磁盘上的，一批可以快速定位数据的文件

mysql的索引数据结构B+Tree

### 9.1.4.数据结构

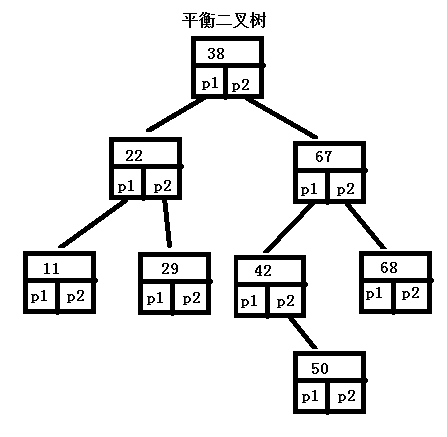
**（1）二叉树**



问题：如果跟节点每选好，有可能数据向一边倾斜，链式查询

**（2）平衡二叉树**

叶子节点的高度偏差不能超过1



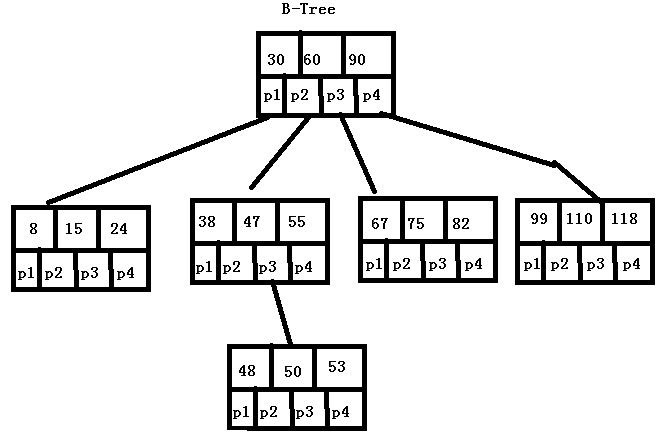
如果高度差超过1，可以通过左旋、右旋（自己去普及）达到平衡

问题一：效率低，数据量很大时，数变得很高，磁盘io次数很多

问题二：查询不稳定，有可能一次，也有可能很多次

问题三：节点数据太少，没哟充分利用磁盘io（默认一次io传输4k数据）

**（3）B-Tree**

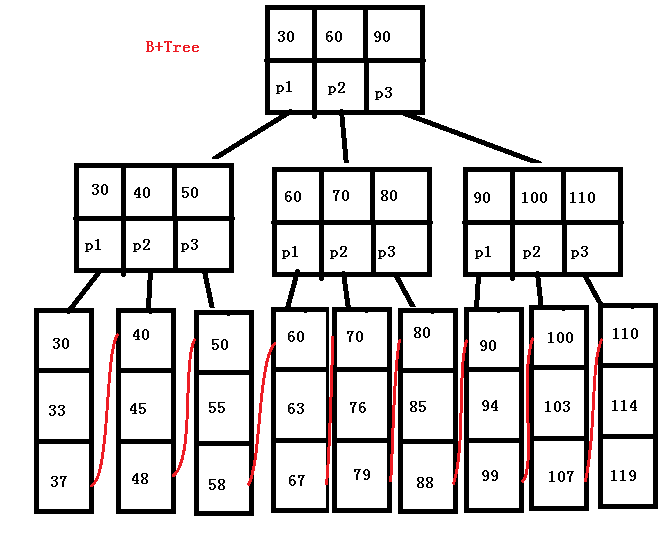


每个节点可以保存多个关键字，每次io默认16k

相比二叉树：

树高大大减小，充分利用磁盘（每个节点放满16k）

**（4）B+Tree**



与B-Tree类似

B+Tree特点：

特点1：半闭合的区间

特点2：根与支节点只保存关键字和子节点引用，不保存具体数据的位置信息

特点3：叶子节点的每个元素都有下个元素的引用，所有子节点形成链表

与B-Tree比较

优势一：全表扫描快，只需扫描叶子即可，叶子有数据的引用

优势二：排序功能强，因为叶子天然排序

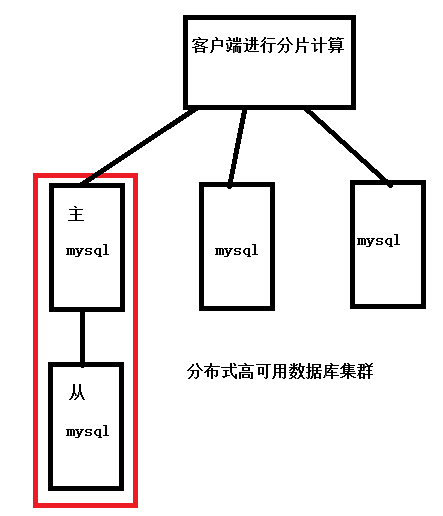
优势三：一次io获取的关键字更多（由于根和支节点只保存关键字），一次读的跟多，树高小

优势四：查询效率更稳定（看似b-Tree效率高，但是由于树高小，磁盘io少）

## 9.2.数据库集群

单机数据库受限物理瓶颈，难以应对高并发场景（oracle单表数据量亿级别，mysql千万级）

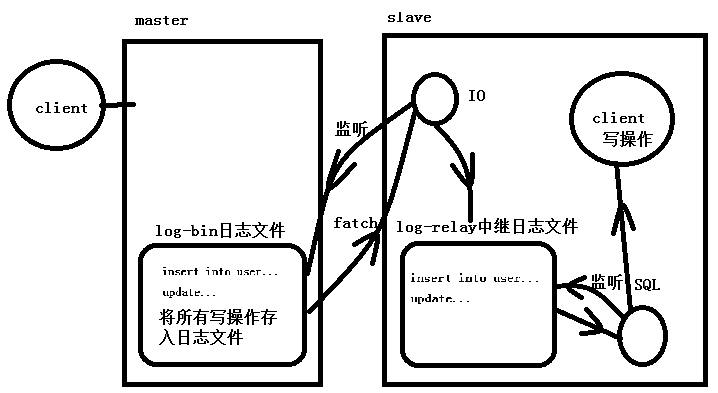
### 9.2.1.架构



### 9.2.2.主从复制原理

mysql支持一主多从、多级主从结构

主从原理：



在一个主从结构中，主从各自有各自的逻辑

**主节点：**

必须开启一个二进制的日志文件，以二进制格式记录所有写操作命令，并且，会根据数据的写入，增加一个position值，使得从节点通过position值判断和抓取数据

**从节点：**

开启IO线程、SQL线程、本地中继日志

IO线程：负责监听和抓取主节点更新内容（通过position的值判断是否变化）

中继日志：IO线程将更新的内容存入本地的一个文件-中继日志

SQL线程：负责监听本地中继日志，判断是否更新（position的值），如果有更新，则抓取更新内容，并在内部启动一个client客户端执行这些命令

### 9.2.3.mysql的主从复制搭建（一主一从）

1.安装2个主机的mysql数据库(Percona 5.6mysql)（关防火墙）

1.1获取安装包解压

/home/resources/Percona\*\*.tar

* 维护一个解压文件的目录

[root@10-9-48-69 ~]# mkdir /home/software/percona

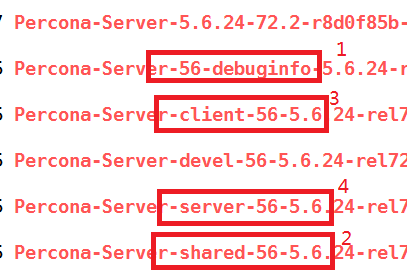
* 将资源tar包拷贝到创建的文件夹中

[root@10-9-48-69 ~]# cp /home/resources/Percona-Server-5.6.24-72.2-r8d0f85b-el6-x86\_64-bundle.tar /home/software/percona/

* 解压Percona的tar包

[root@10-9-48-69 percona]# tar -xf Percona-Server-5.6.24-72.2-r8d0f85b-el6-x86\_64-bundle.tar

1.2rpm安装包按顺序安装



* 安装debuginfo的rpm

[root@10-9-48-69 percona]# rpm -ivh Percona-Server-56-debuginfo-5.6.24-rel72.2.el6.x86\_64.rpm

* 安装shared的rpm

[root@10-9-48-69 percona]# rpm -ivh Percona-Server-shared-56-5.6.24-rel72.2.el6.x86\_64.rpm

* 安装client的rpm

[root@10-9-48-69 percona]# rpm -ivh Percona-Server-client-56-5.6.24-rel72.2.el6.x86\_64.rpm

* 安装server的rpm

[root@10-9-48-69 percona]# rpm -ivh Percona-Server-server-56-5.6.24-rel72.2.el6.x86\_64.rpm

* 出现mysql的各种问题之后,可以执行重装

注意:在percona意外的其他位置执行删除命令.

#yum -y remove Percona\*

2.调整mysql的环境

2.1启动mysql的服务

[root@10-9-48-69 percona]# service mysql start

[root@10-9-48-69 percona]# service mysql stop

[root@10-9-48-69 percona]# service mysql restart

2.2开启远程的访问权限

* 登录到mysql客户端

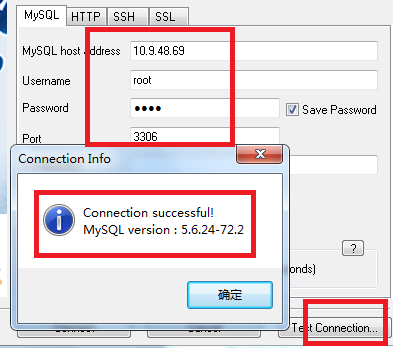
[root@10-9-48-69 percona]# mysql

* 在客户端执行命令

grant 权限 on 库.表 to 登录用户名@'允许登录ip' identified by '用户登录时携带的密码'

mysql> grant all on \*.\* to 'root'@'%' identified by 'root';

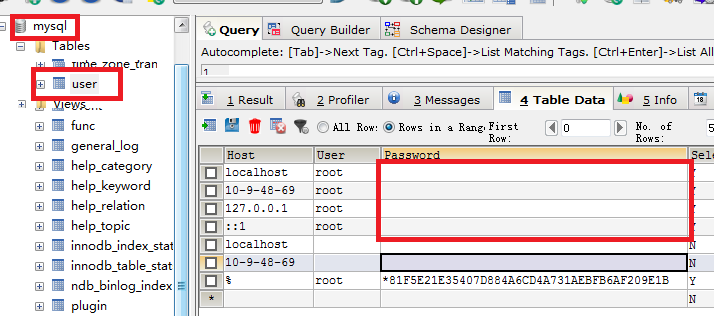
* 远程权限开启完毕,可以测试sqlYog从windows登录服务器的mysql数据库



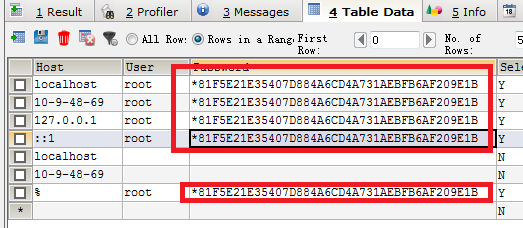
2.3将所有root用户名都修改成以'root',密码登录的方式

对于mysql在云服务器的percona版本,保证稳定的被mycat连接

需要修改mysql.user表格的数据



修改成如下内容,并且保存



2.4本地忽略localhost 空user 空密码的登录使用

[root@10-9-17-153 percona]# mysqladmin -uroot password 'root'

最终要确定使用mysql命令在本地登录时,不能再空密码,空用户名的登录了

3.数据库的主从复制搭建

3.1主节点相关配置

开启一个二进制日志文件,将所有的写操作记录在日志文件中

* 在/etc/my.cnf,当前服务器为mysql准备的一个配置文件

问题:没找到my.cnf,可以通过远程拷贝,将出现的my.cnf拷贝到/etc下

[root@10-9-17-153 percona]# scp /etc/my.cnf root@10.9.104.184:/etc

远程登录选择:

Are you sure you want to continue connecting (yes/no)? yes

盲敲登录密码:tarena2017Up;

root@10.9.104.184's password:

my.cnf 100% 251 0.3KB/s 00:00

如果密码输入失败,只能重新完成(免密登录)

* 打开my.cnf 添加集群id和开启二进制日志文件的配置内容

在中间空白处添加:

server-id=1 :定义的是集群中mysql的id值,不能和其他mysql节点的id重复.

log-bin=mysql-bin:定义了一个开启的二进制日志文件的名称

* 重启主节点的mysql服务

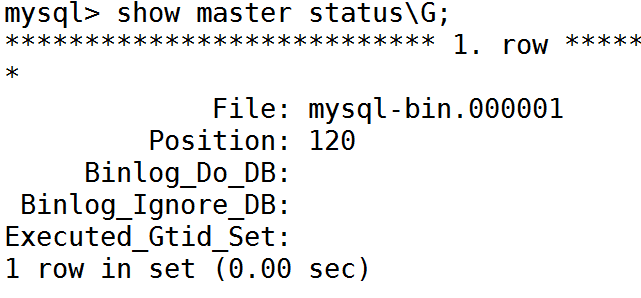
[root@10-9-48-69 percona]# service mysql restart

需要重启重新加载修改后的my.cnf配置文件,重启完成的mysql就会根据my.cnf内容进行加载修改,就会出现可以使用的二进制日志文件

* 登录到主节点客户端,执行查看命令

mysql> show master status\G;

\G是在mysql的linux客户端可以添加的配置参数,将查询的结果竖着展示

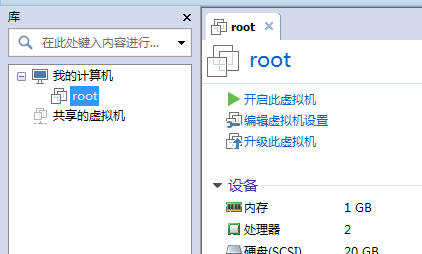


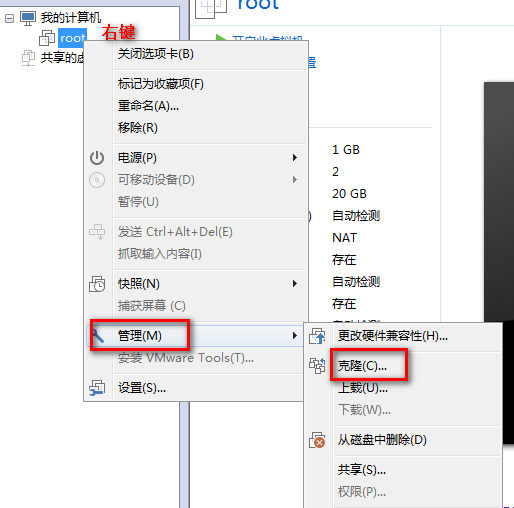
当前查询的内容,就是主节点开启的二进制文件的记录信息,包含文件名称mysql-bin.000001,后缀是一个自增的数字,从1开始,当服务器重启时,当文件数据量超过阈值64M/128M,会滚动,数字自增1,指针position的值可以用来判断更新,mysql-bin的上一个滚动文件的指针指向下一个新文件的开始的;

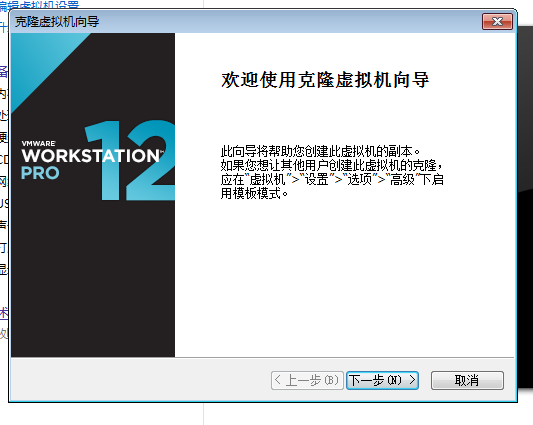
3.2从节点的配置

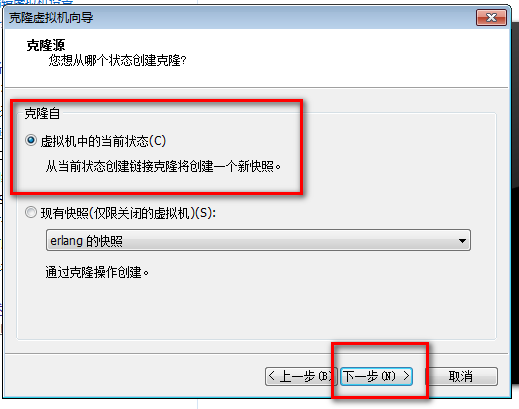
将主节点的虚拟机克隆一个（主节点配置前可以克隆）

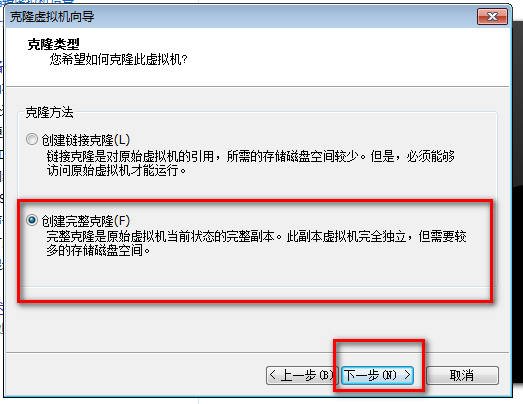
关机->虚拟机名称右键->管理->克隆

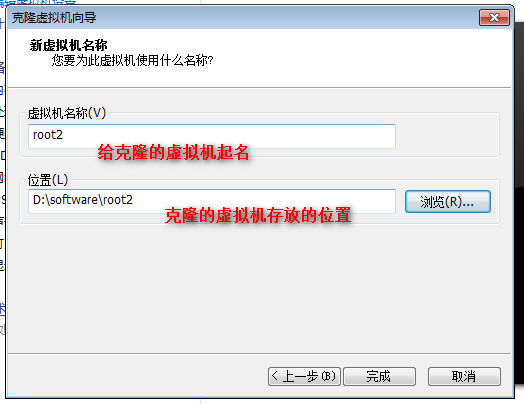


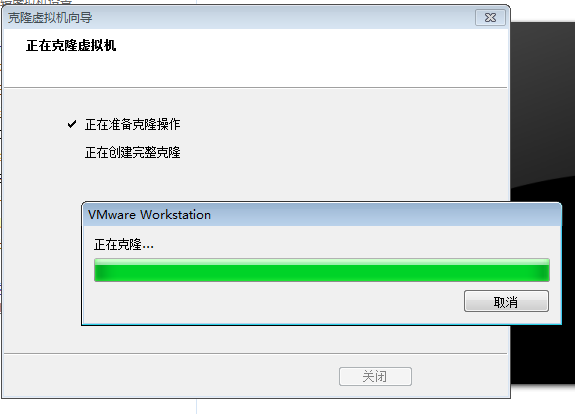


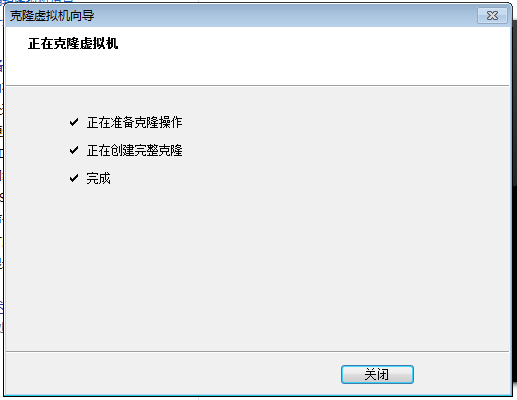












### 9.2.4.从节点搭建

注意：需要关闭防火墙

* 修改配置文件my.cnf

server-id=2

log-bin=mysql-bin (双机热备做准备)

* 重启从节点重新加载my.cnf

[root@10-9-17-153 percona]# service mysql restart

* 查看从节点状态

mysql> show slave status\G;

Empty set (0.00 sec)

说明当前从节点没有挂接任何主节点,一旦挂接,show slave status 返回所有从节点状态信息;在从节点客户端执行挂接命令

mysql>

CHANGE MASTER TO MASTER\_HOST="10.9.48.69",

MASTER\_PORT=3306,

MASTER\_USER="root",

MASTER\_PASSWORD="root",

MASTER\_LOG\_FILE="mysql-bin.000002",

MASTER\_LOG\_POS=120

查看从节点的信息status

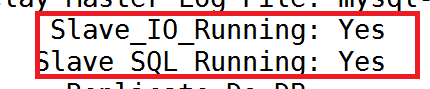
Slave\_IO\_Running: No

Slave\_SQL\_Running: No\

* 将从节点的io和sql线程同时开启

mysql>start slave/stop slave

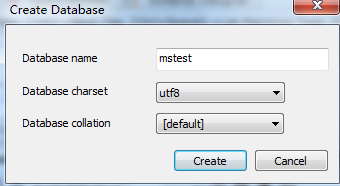
观察从节点status发现两个线程都工作



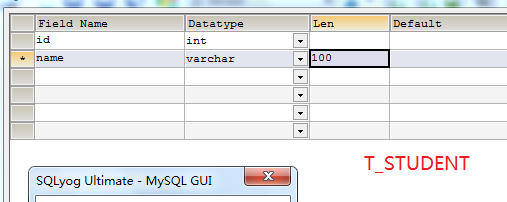
### 9.2.5.测试主从关系

* 在主节点写入数据,观察从节点的变化是否同步

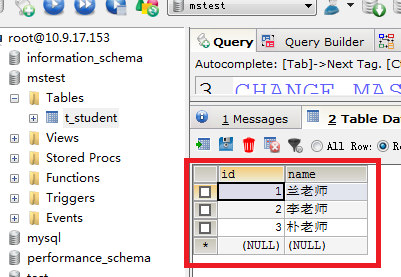
在主节点创建数据库



创建t\_student表格（id主键）



在表格中插入一些数据



* 在从节点写入数据.然后继续在主节点写入数据

INSERT INTO t\_student (NAME) VALUES

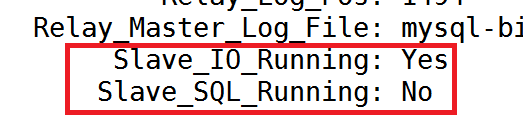
('Mr.Li')

我在主节点继续写数据（insert）;

在从节点观察，数据并未同步

出现报错:Duplicate entry '4' for key 'PRIMARY'

* 查看从节点状态status



确保从节点不能写数据;

* 重新挂接主从(历史数据不同步,保证历史数据完全一致)
  + 保证数据一致(数据比较简单)

修改数据一致

* 停掉IO SQL线程,停止从节点

STOP SLAVE

* 重新根据master的status挂接主从

CHANGE MASTER TO MASTER\_HOST="10.9.48.69",

MASTER\_PORT=3306,

MASTER\_USER="root",

MASTER\_PASSWORD="root",

MASTER\_LOG\_FILE="mysql-bin.000002",

MASTER\_LOG\_POS=2228

* 启动从节点
* 分析主从替换

可以通过spring aop spring框架管理多个datasource对象,通过在异常或者其他逻辑中,实现多个datasource的分别调用;性能极低.

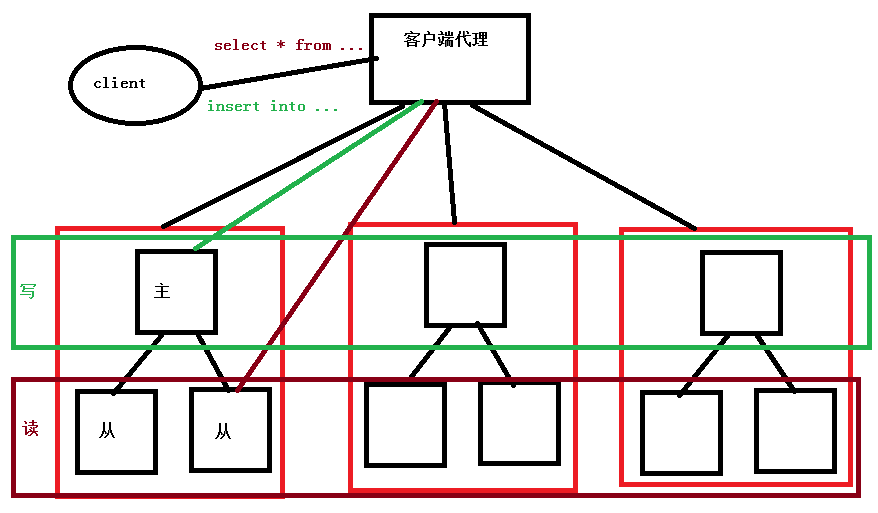
主从故障转移,是通过多种数据库集群的代理客户端技术实现(mycat)

## 9.3.数据库集群高可用

### 9.3.1.主从故障转移

一旦集群引入高可用主从复制结构，可以自定义配置，实现主节点宕机，从节点完成之前主节点的所有功能--故障转移

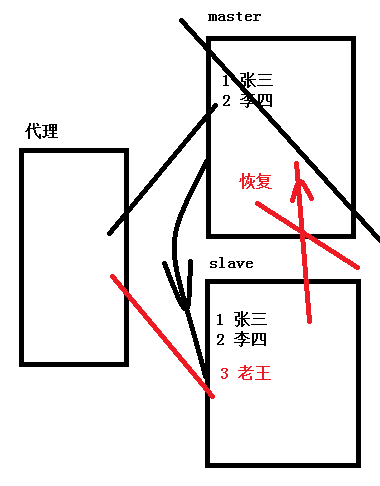
### 9.3.2.读写分离



读写分离：将写的请求全部交给主节点，但是读可以交给从节点，来分担主节点的压力

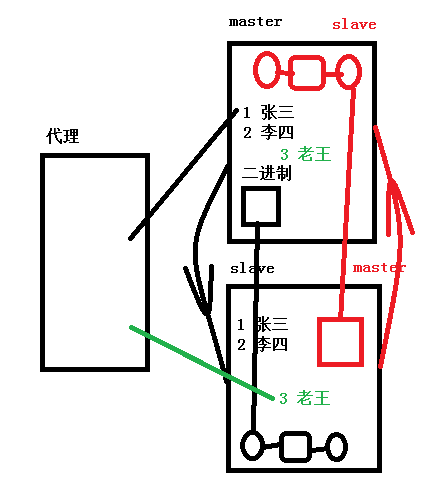
### 9.3.3.单机热备

备份是一次性的，只能从节点备份主节点，单方向备份，一旦主节点挂掉，即使重启，也无法拿到从节点数据。



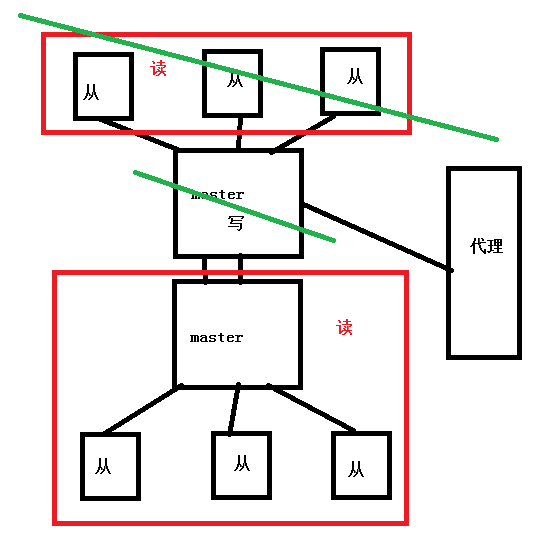
### 9.3.4.双机热备

备份是多次循环，双向主从复制实现故障转移



是否可以三机热备？（不推荐）

可以如下配置



实现双机热备：从挂主，主挂从

在主节点也执行从挂主的逻辑即可