查询注意点

禁用select \*，一是为了方便阅读，二是确实\*会慢一点点

使用select count(\*) 统计行数，效率上count(1)=count(0)=count(常量)=count(\*)，最后都是count(\*)，所以直接用count(\*)，在这里并不是全表扫描，count(列名)这个列字段如果长的话，效率较低。

尽量少运算

尽量避免对字段进行表达式计算

尽量避免对字段进行函数操作

where条件

尽量避免在where子句对字段进行null判断

尽量避免在where子句使用!= 或者<>

尽量避免在where子句使用or连接

尽量避免在where子句中使用in,not in或者having，使用exists,not exists代替

尽量早过滤（mysql与oracl不同）

多表查询

尽量少join，生成笛卡尔积

尽量用join代替子查询，这可以从执行顺序得出，子查询又从头到尾走了一边流程。

尽量用union all代替union

索引

尽量避免使用不是复合索引的前缀列进行过滤连接

尽量避免两端模糊匹配 like ‘%呵呵%’，这也是跟索引有关，索引依靠前缀去找，%在前面就不会用索引，所以慢。

尽量避免全表扫描，如果可以，在过滤列建立索引

不能用null作索引，任何包含null值的列都将不会被包含在索引中，复合索引任意一列包含null值，索引无效。

EXPLAIN

MySql提供了EXPLAIN语法用来进行查询分析，在SQL语句前加一个"EXPLAIN"即可。

其他

避免类型转换

优先优化高并发sql，而不是频率低的大sql

尽可能对每一条sql进行explain

尽可能从全局出发

尽量批量insert

1. 计算存储原理

数据库 先存内存 最终数据提交写入 磁盘里面（永久保存）

数据库量小，就不会出现（数据库性能的问题）

影响软件性能的第一大要素（I/O 吞吐量 读/写的速度）

固态硬盘(SSD) 机械硬盘

1. 内存

影响软件性能第二个要素 内存

1. CPU

i5(3.1GHZ) 每秒=3.1\*1204\*1024\*1024=次

1. 软件2种

内存型(存储量大) 计算型(大批量的运算)

数据库(内存型的)

1. 数据库优化
2. 、硬件方面

通过 换SSD硬盘， 增大内存容量(来提升速度) (有点感觉)

1. 、软性方面

例如： 1+10000000的和 循环 (增加硬件 I5,I7,I9)

50\*101=? 算法(p2+128MB)

1. 、表结构开始
   1. 、通过建立索引 达到优化目的

不要使用in ,not in

不要使用 like

不要使用函数对索引列进行查询

* 1. 、什么列该建立索引，什么列不要建立索引?

经验 经常查询的列

表频繁更新，这个表不要建立索引.

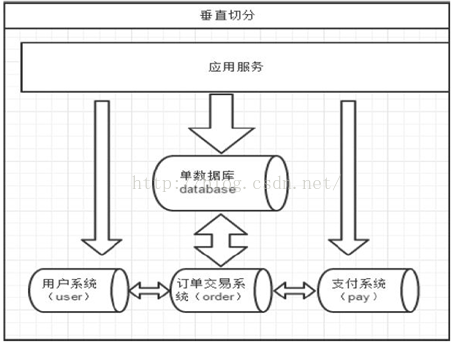
数据库小的表不要建立索引(万以下的数据)

2)、建立索引后还是慢

分库分表， 分法 2 种拆分规则

垂直拆分 (某些功能的用户访问量太大)

      一个数据库由很多表的构成，每个表对应着不同的业务，垂直切分是指按照业务将表进行分类，分布到不同的数据库上面，这样也就将数据或者说压力分担到不同的库上面，如下图：



优点：

        1. 拆分后业务清晰，拆分规则明确。

        2. 系统之间整合或扩展容易。

        3. 数据维护简单。

缺点：

        1. 部分业务表无法join，只能通过接口方式解决，提高了系统复杂度。

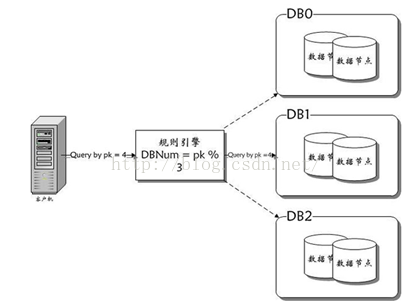
        2. 受每种业务不同的限制存在单库性能瓶颈，不易数据扩展跟性能提高。

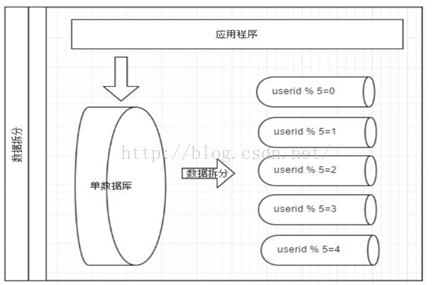
        3. 事务处理复杂(分布式事务[很难解决的问题])。

**水平拆分(单表性能解决方案)**

 垂直拆分后遇到单机瓶颈，可以使用水平拆分。相对于垂直拆分的区别是：垂直拆分是把不同的表拆到不同的数据库中，而水平拆分是把同一个表拆到不同的数据库中。

        相对于垂直拆分，水平拆分不是将表的数据做分类，而是按照某个字段的某种规则来分散到多个库之中，每个表中包含一部分数据。简单来说，我们可以将数据的水平切分理解为是按照数据行的切分，就是将表中 的某些行切分到一个数据库，而另外的某些行又切分到其他的数据库中，主要有分表，分库两种模式，如图：





优点：

        1. **不存在单库大数据，**高并发的性能瓶颈。

        2. 对应用透明，应用端改造较少。

        3. 按照合理拆分规则拆分，join操作基本避免跨库。

        4. 提高了系统的稳定性跟负载能力。

缺点：

        1. 拆分规则难以抽象。

        2. 分片事务一致性难以解决。

        3. 数据多次扩展难度跟维护量极大。

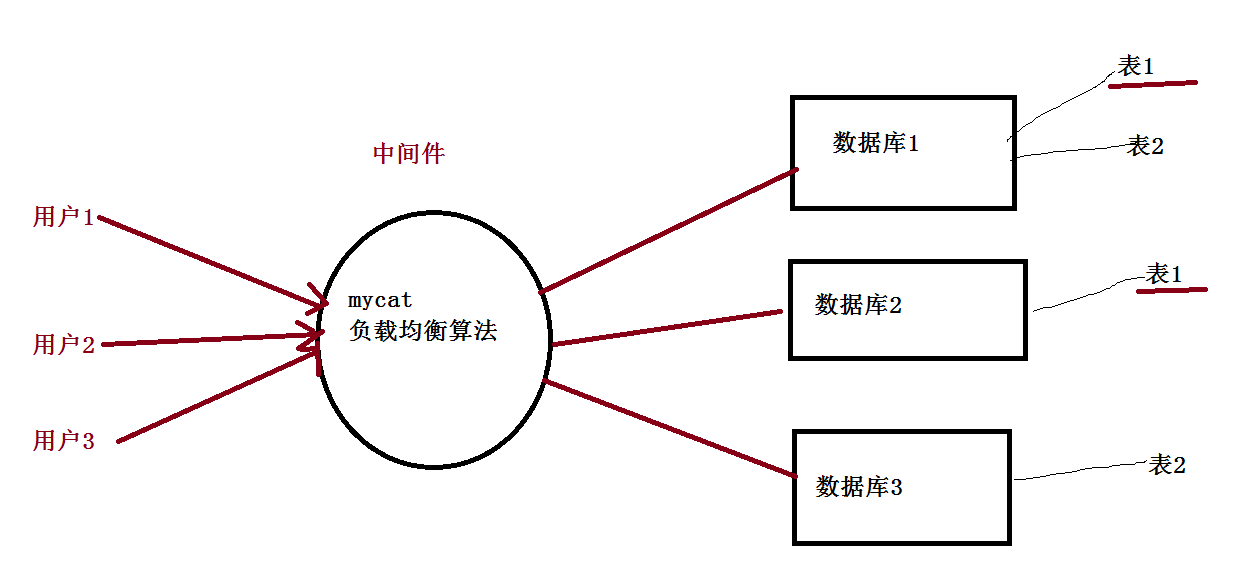
        4. 跨库join性能较差。

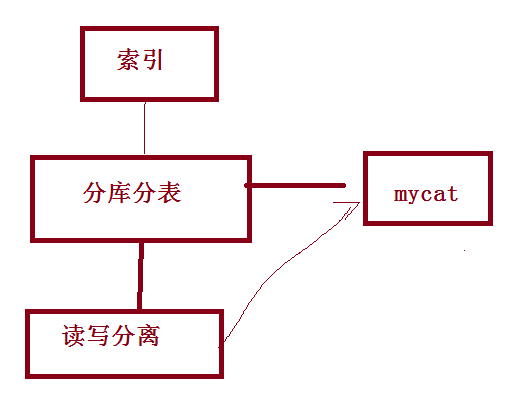
1. 分库分表 哪个软件能够来管理这些数据库？

mycat (阿里开源软件) 做配置

<http://www.mycat.io/>

<https://blog.csdn.net/vbirdbest/article/details/83448757> 简单案例





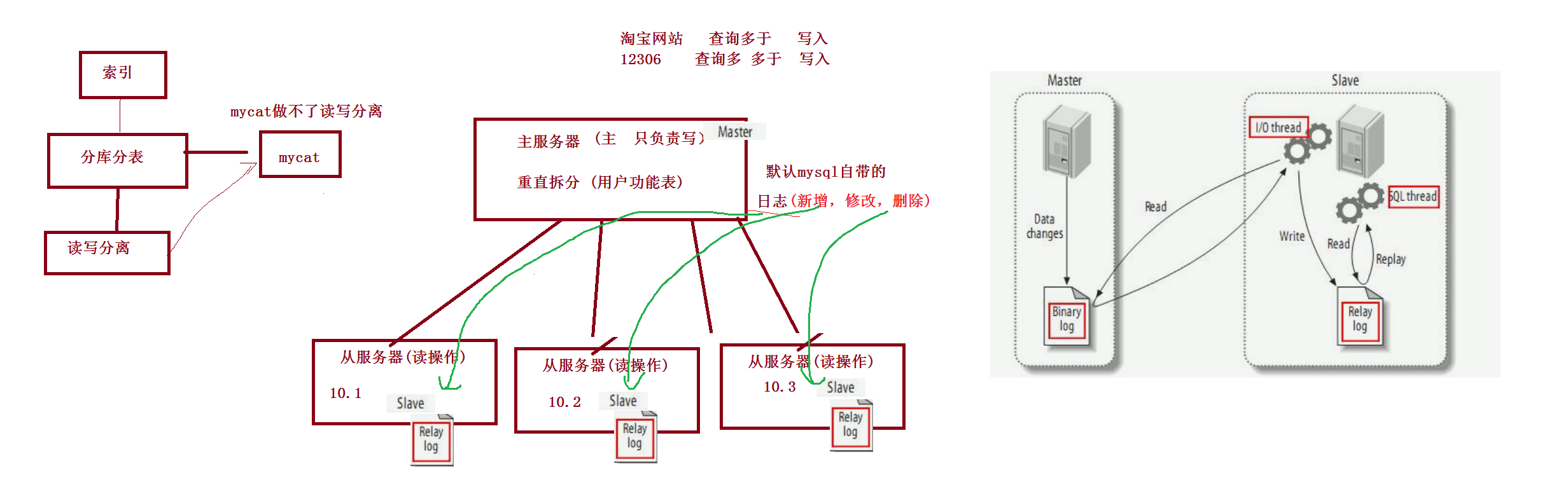
并发量太大

1. 读写分离配置

mysql自带的

<https://www.cnblogs.com/codehome/p/9356496.html>

mysql.conf.d 中配置



做完以后 问题越来越多：

1. 、分布式 （事务 怎么办？）

<https://www.cnblogs.com/jiangyu666/p/8522547.html>

1. 、服务维护 (请专业的数据库管理员)
2. 、程序代码要求提高
3. 能用缓存的就不要去用数据库的数据

CPU 运算最快的（内存 3MB） --> 内存(第二快) --> 硬盘

使用 redis做数据缓存