1. mysql优化方向：字段设计、索引、集群、负载均衡、读写分离、语句优化(优化效果小)
2. NotNull优于Null：Null需要额外空间存储，运算也需要通过is not null 和 is null。
3. 索引使用场景：
   1. where：最左前缀原则
   2. order by：对无索引字段进行排序时，mysql 会使用外部排序，即将数据读到内存进行内部排序，最后合并排序结果。外部排序开销很大，因此要对排序字段建索引。
   3. join：on上的字段建索引可以提高查询性能。

# Mysql分区

1. Mysql一般是一组存储文件对应一个表，InnoDB的存储文件为.idb文件及.frm(表结构)文件。
2. 分区：数据量大时单独的存储文件会导致性能下降，因此要将数据分散成多组存储文件，以保证单个存储文件的执行效率。
3. 分区要求：分区字段必须是主键的一部分。如以创建时间分区，PRIMARY KEY(id, created\_time)。
4. 分区语法：PARTITION BY RANGE(created\_time)(

PARTITION p201808 VALUES less than (1535731199), -- select UNIX\_TIMESTAMP('2018-8-31 23:59:59')

PARTITION p201809 VALUES less than (1538323199), -- 2018-9-30 23:59:59

PARTITION p201810 VALUES less than (1541001599) -- 2018-10-31 23:59:59

)

1. 分区算法：Range(常用)、hash、key(适用于字符串)
2. 分区可以增加也可以删除，删除分区会连带数据一起删除。
3. 分区一般用于单机，且分区使用少于分表。

# Mysql分表和分库

1. 垂直划分：将高频数据与低频数据分开存储。热门高频的放在主要表，冷门低频的放在次要表，主要表和次要表的关系一般为一对一。
2. 水平划分：把相同数据结构的表。水平划分方法：根据哈希取模、根据时间划分。
3. 分库原因：主从复制可以满足读取要求，但在写入速度持续升高的情况下，单库并不能处理高并发的写入。