项目难点及问题：

1. 当一个人多次点击，导致redis缓存已经小于0了，这样其他人也没法秒杀了，商品也只卖了。

这个叫幂等性，即同一个操作，无论重复多少次，其结果应该是一致的。
一个人发了10次秒杀，应该等同于发一次秒杀。那么你在redis应该记录一下这个用户的唯一标识，标识其是否进行过秒杀活动。秒杀成功，同样mysql也要做记录，避免服务重启等原因造成缓存丢失，重复秒杀的情况。

2. 使用JMeter做压测的时候开启5000个线程，系统跑不起来，出现异常

原因：修改配置文件中redis的配置项poolMaxTotal 将其设置成1000。

```
#redis配置项
redis.poolMaxTotal=1000
redis.poolMaxldle=500
redis.poolMaxWait=500
```

3. 使用了大量缓存，那么就存在缓存击穿和缓存雪崩以及缓存一致性等问题？

缓存穿透指的是对某个一定不存在的数据进行请求，该请求将会穿透缓存到达数据库。
解决方案：对这些不存在的数据缓存一个空数据，对这类请求进行过滤。

缓存雪崩指的是由于数据没有被加载到缓存中，或者缓存数据在同一时间大面积失效（过期），又或者缓存服务器宕机，导致大量的请求都到达数据库。
解决方案：
为了防止缓存在同一时间大面积过期导致的缓存雪崩，可以通过观察用户行为，合理设置缓存过期时间来实现；
为了防止缓存服务器宕机出现的缓存雪崩，可以使用分布式缓存，分布式缓存中每一个节点只缓存部分的数据，当某个节点宕机时可以保证其它节点的缓存仍然可用。
也可以进行缓存预热，避免在系统刚启动不久由于还未将大量数据进行缓存而导致缓存雪崩。
例如：首先针对不同的缓存设置不同的过期时间，比如session缓存，在userKey这个前缀中，设置是30分钟过期，并且每次用户响应的话更新缓存时间。这样每次取session,都会延长30分钟，相对来说，就减少了缓存过期的几率

缓存一致性要求数据更新的同时缓存数据也能够实时更新。

解决方案：
在数据更新的同时立即去更新缓存，首先尝试从缓存读取，读到数据则直接返回；如果读不到，就读数据库，并将数据会写到缓存，并返回。
在读缓存之前先判断缓存是否是最新的，如果不是最新的先进行更新，需要更新数据时，先更新数据库，然后把缓存里对应的数据失效掉（删掉）。

4. 大量的使用缓存，对于缓存服务器，也有很大的压力，思考如何减少redis的访问？

在redis预减库存的时候，内存中维护一个isOvermap作为一个内存标记，当没有库存的时候，将其置为true。每次秒杀业务访问redis之前，查一下map标记，如果true说明没有库存，就直接返回失败，无需再去请求redis服务器。

5. 在高并发请求的业务场景，大量请求来不及处理，甚至出现请求堆积时候？

消息队列，用来异步处理请求。每次请求过来，先不去处理请求，而是放入消息队列，然后在后台布置一个监听器，分别监听不同业务的消息队列，有消息来的时候，才进行秒杀业务逻辑。这样防止多个请求同时操作的时候，数据库连接过多的异常。

6. 怎么保证一个用户不能重复下单？

解决：秒杀订单表中建立一个唯一索引（所引是用户Id与商品goodsId），使得第一个记录可以插入，第二个则出错，然后通过事务回滚，防止一个用户同时发出多个请求的处理，秒杀到多个商品。

唯一索引，即是唯一的意思，在数据库表结构中对字段添加唯一索引后进行数据库进行存储操作时数据库会判断库中是否已经存在此数据，不存在此数据时才能进行插入操作。

这虽然是个小技能，但实际上在业务开发中是个很实用的技能，比如在高并发业务中，数据库如何杜绝数据并发插入两条相同的订单号呢？添加一个唯一索引当然是最快捷的方法之一，当然是添加索引还是通过业务代码去解决因公司业务而定

7. 怎么解决超卖现象？

超卖场景：不同用户在读请求的时候，发现商品库存足够，然后同时发起请求，进行秒杀操作，减库存，导致库存减为负数。

最简单的方法，更新数据库减库存的时候，进行库存限制条件，在reduceStock(GoodsVo goodsvo)这个方法里，sql要多加一个stock_count > 0 ，使用数据库特性来保证超卖的问题，只有stock_count还大于0的时候才去读stock_count然后减1操作

```
@Update("update miaosha_goods set stock_count=stock_count-1 where goods_id=#{goodsId}
and stock_count>0")
    public void reduceStock(MiaoshaGoods goods);
```

8. 假如减了库存用户没有支付，库存怎么还原继续参加抢购?

设定一个最长付款时间，比如30分钟，后台有个定时任务（使用定时器Timer），轮训超过30分钟的待付款订单(数据库里面判定订单状态)，然后关闭订单，恢复库存。