# 高并发

使用Synchronized 解决，给生成ID 的代码加上同步代码块，成功解决问题； 作用是：同一时刻，只有一个线程可以执行该代码块

使用Lock 锁解决问题：给生成ID 的代码加上Lock 锁，成功解决问题；

悲观锁---在修改数据的时候,采用锁定状态,排斥外部请求的修改,遇到加锁的状态,就必须等待

弊端: 在高并发下,每个请求都需要等待’锁’,某些线程可能永远都没有机会抢到这个锁,请求就会死在那里,这种请求会很多,瞬间增系统的平均响应时间,结果时可用链接数被耗尽,系统陷入异常

FIFO 队列----采用FIFO(First Input First Output，先进先出),这样就不会导致某些请求永远获取不到锁

弊端:请求很多,很有可能一瞬间将队列内存”撑爆”,系统陷入到异常状态,或者设计一个极大的内存队列,但是系统处理完一个队列,内请求的速度根本无法和疯狂涌入队列中的数目相比,也就是说,队列内的请求会越积累越多,最终WEB 系统平均响应时候还是会大幅下降，系统还是陷入异常。

乐观锁---相对于”悲观锁”采用更为宽松的加锁机制,大都是采用带版本号更新,实现就是,这个数据所有请求都有资格去修改,但会获得一个该数据的版本号,只有版本号符合才能更新成功,其他的返回抢购失败,这样就不用了考虑队列的问题,会增大CPU 的计算开销,