1. 并发编程的挑战
2. 概述

并发编程的目的是让程序运行地更快，通过多线程执行任务可以让程序运行地更快，但是会有很多的挑战，如上下文切换问题，死锁的问题，以及受限于硬件和软件的资源限制问题。

1. 上下文切换
2. 概述

即时是单核CPU也支持多线程执行代码，CPU通过给每个线程分配CPU时间片来实现这个机制，CPU通过不断地切换线程执行，让我们感觉多个线程是同时执行的。上下文切换指的就是CPU在切换时间片时，会保存上一个任务的状态，以便下一次直接加载，任务从保存到加载的过程就是一次上下文切换，会影响线程的执行速度。

1. 减少上下文切换

减少上下文切换的方式有无锁并发编程，CAS算法，使用最少线程和使用协程。

1. 无锁并发

避免使用锁，如将数据的ID按照Hash算法取模分段，不同的线程处理不同段的数据。

1. CAS

Java的Atomic包使用CAS算法来更新数据，不需要加锁。

1. 使用最少线程

避免创建不需要的线程，不然会有很多线程处于等待状态，减少上下文切换次数，因为每次从等待状态到RUNNABLE状态都会进行一次上下文切换。

1. 协程

在单线程里实现多任务的调度，在单线程里维持多个任务间的切换。

1. jstack使用

使用自带的jstack命令来dump线程信息，可以直接查询某个进程中的线程的状态。如发现其中WAITING状态的线程过多，可以通过减少该进程的线程池数量来减少上下文切换次数。

1. 死锁

避免死锁的方法：

1. 避免一个线程同时获取多个锁；
2. 避免一个线程在锁内同时占用多个资源；
3. 尝试使用定时锁，lock.tryLock(timeout)；
4. 对于数据库锁，加锁和解锁必须在一个数据库连接中，否则会出现解锁失败的问题；
5. 资源限制问题
6. 硬件资源限制

带宽的上传和下载速度，硬盘读写速度和CPU的处理速度，可以考虑使用集群并行执行程序，不同的机器处理不同的数据。

1. 软件资源限制

数据库的连接数和socket连接数问题，考虑使用资源池将资源复用，如数据库连接池和socket连接复用，不需要每次连接时再创建和关闭连接。