* 什么是Daemon线程？它有什么意义？  
         答：所谓后台(daemon)线程，是指在程序运行的时候在后台提供一种通用服务的线程，并且这个线程并不属于程序中不可或缺的部分。因此，当所有的非后台线程介绍时，程序也就终止了，同时会杀死进程中的所有后台线程。反过来说，只要有任何非后台线程还在运行，程序就不会终止。必须在线程启动之前调用setDaemon()方法，才能把它设置为后台线程。注意：后台进程在不执行finally子句的情况下就会终止其run()方法。
* java如何实现多线程之间的通讯和协作？  
        答： Java提供了3个非常重要的方法来巧妙地解决线程间的通信问题。这3个方法分别是：wait()、notify()和notifyAll()。它们都是Object类的最终方法，因此每一个类都默认拥有它们。虽然所有的类都默认拥有这3个方法，但是只有在synchronized关键字作用的范围内，并且是同一个同步问题中搭配使用这3个方法时才有实际的意义
* 上述两个正确吗？待确认

在java 1.5后提供了 volatile 保证读线程一定在写线程完成之后.---所以又衍生出volatile的问题。

缓存一致性协议。最出名的就是Intel 的MESI协议，MESI协议保证了每个缓存中使用的共享变量的副本是一致的。它核心的思想是：当CPU写数据时，如果发现操作的变量是共享变量，即在其他CPU中也存在该变量的副本，会发出信号通知其他CPU将该变量的缓存行置为无效状态，因此当其他CPU需要读取这个变量时，发现自己缓存中缓存该变量的缓存行是无效的，那么它就会从内存重新读取。

当一个共享变量被volatile修饰时，它会保证修改的值会立即被更新到主存，当有其他线程需要读取时，它会去内存中读取新值。

通过synchronized和Lock也能够保证可见性，synchronized和Lock能保证同一时刻只有一个线程获取锁然后执行同步代码，并且在释放锁之前会将对变量的修改刷新到主存当中。因此可以保证可见性。

### Thread 类中的start() 和 run() 方法有什么区别？

start()方法被用来启动新创建的线程，而且start()内部调用了run()方法，这和直接调用run()方法的效果不一样。当你调用run()方法的时候，只会是在原来的线程中调用，没有新的线程启动，start()方法才会启动新线程。

Java中Runnable和Callable有什么不同？

Runnable和Callable都代表那些要在不同的线程中执行的任务。Runnable从JDK1.0开始就有了，Callable是在JDK1.5增加的。它们的主要区别是Callable的 call() 方法可以返回值和抛出异常，而Runnable的run()方法没有这些功能。Callable可以返回装载有计算结果的Future对象。

 DB优化

        a、在数据库设计的时候就要考虑到后期的维护，数据库三范式是我们设计数据库索要遵循的原则。

        b、索引的建立：建立索引要适当，如果一个表经常用来被查询，对于增加和修改很少被用到，我们就可以为这个表建立索引，因为对于增加和修改和删除操作时，我们对索引的维护要大大超过索引给我们带来的效率。

        c、表字段的类型选择要恰当

包括字段的长度、类型等，要根据实际存储的数据进行选择，长度不要过长，否则会影响效率。

        d、外键要慎用，因为主键代表这一张表，而外键代表一群表，对表之间进行了关联，在删除修改等需要我们关联。

        e、在数据库操作上

                尽量使用prepareStatement，少用Statement，因为PrepareStatement是进行预编译的。

                connection设置为readOnly，Connection是对书库连接，属于重量级，我们使用即可。

                连接池的使用，我们可以修改数据库默认的连接数。

设计模式和编程模型，讲一下传统一个请求一个线程的模式和selector模式的区别，讲一下监听模式和广播模式的异同等等，都是多线程编程常见的概念

### Java中什么是竞态条件？

竞态条件会导致程序在并发情况下出现一些bugs。多线程对一些资源的竞争的时候就会产生竞态条件，如果首先要执行的程序竞争失败排到后面执行了，那么整个程序就会出现一些不确定的bugs。这种bugs很难发现而且会重复出现，因为线程间的随机竞争。

1. 使用 interrupt 方法终止线程

使用Thread提供的interrupt()方法，因为该方法虽然不会中断一个正在运行的线程，但是它可以使一个被阻塞的线程抛出一个中断异常，从而使线程提前结束阻塞状态，退出堵塞代码。

注意：在Thread类中有两个方法可以判断线程是否通过interrupt方法被终止。一个是静态的方法interrupted（），一个是非静态的方法isInterrupted（），这两个方法的区别是interrupted用来判断当前线是否被中断，而isInterrupted可以用来判断其他线程是否被中断。

OOP: Object Oriented Programming,[面向对象](https://www.baidu.com/s?wd=面向对象&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)的程序设计