事务的隔离级别

1.查看当前会话隔离级别

select @@tx_isolation;

2.查看系统当前隔离级别

select @@global.tx_isolation;

3.设置当前会话隔离级别

set session transaction isolatin level repeatable read;

4.设置系统当前隔离级别

set global transaction isolation level repeatable read;

5.命令行，开始事务时

set autocommit=off 或者 start transaction

关于隔离级别的理解

1.read uncommitted

可以看到未提交的数据（脏读），举个例子：别人说的话你都相信了，但是可能他只是说说，并不实际做。

2.read committed

读取提交的数据。但是，可能多次读取的数据结果不一致（不可重复读，幻读）。用读写的观点就是：读取的行数据，可以写。

3.repeatable read(MySQL默认隔离级别)

可以重复读取，但有幻读。读写观点：读取的数据行不可写，但是可以往表中新增数据。在MySQL中，其他事务新增的数据，看不到，不会产生幻读。采用多版本并发控制（MVCC）机制解决幻读问题。

4.serializable

可读，不可写。像java中的锁，写数据必须等待另一个事务结束。

此方法可能不太完美，但是能在一定程度上解决高并发问题。本文以高并发抢单来举例。
首先科普一下mysql隔离级别
mysql有四个隔离级别：
Read Uncommitted（读取未提交内容）
Read Committed（读取提交内容）
Repeatable Read（可重读）
Serializable（可串行化）
每个隔离级别的具体特点，大家可以再具体百度。我只知道如何使用，没有深入研究，就不误导大家了。
php高并发解决方案：
网上搜了好多，大概有以下几种方案
1 请求消息队列（MemcacheQ消息队列等），先来的请求先处理，后来的直接告诉用户订单已被抢（单个商品）或引导到商品已被抢完的页面（多个商品）。
2 Memcache锁，实现类似java的同步代码块的效果
3 文件锁实现代码同步
4 mysql事务隔离锁表。
本文讨论第四种
这种方法，肯定要对数据库开启事务的。如果把mysql的事务隔离级别设置为 Serializable（SET SESSION TRANSACTION ISOLATION LEVEL serializable），此隔离级别强制事务排序，需要等一个事务处理完再处理下一个，而在事务中，我们可以判断一下订单的状态，这样当第一个事务提交后，第二个事务处理时订单状态已变，直接回滚。