## 一 事务

事务(Transaction)是访问并可能更新数据库中各种数据项的一个程序执行单元(unit)

## 二 事务的特性

原子性：指事务是一个不可分割的工作单位，事务中的操作要么都发生，要么都不发生。

一致性：事务必须使数据库从一个一致性状态变换到另外一个一致性状态。转账前和转账后的总金额不变。

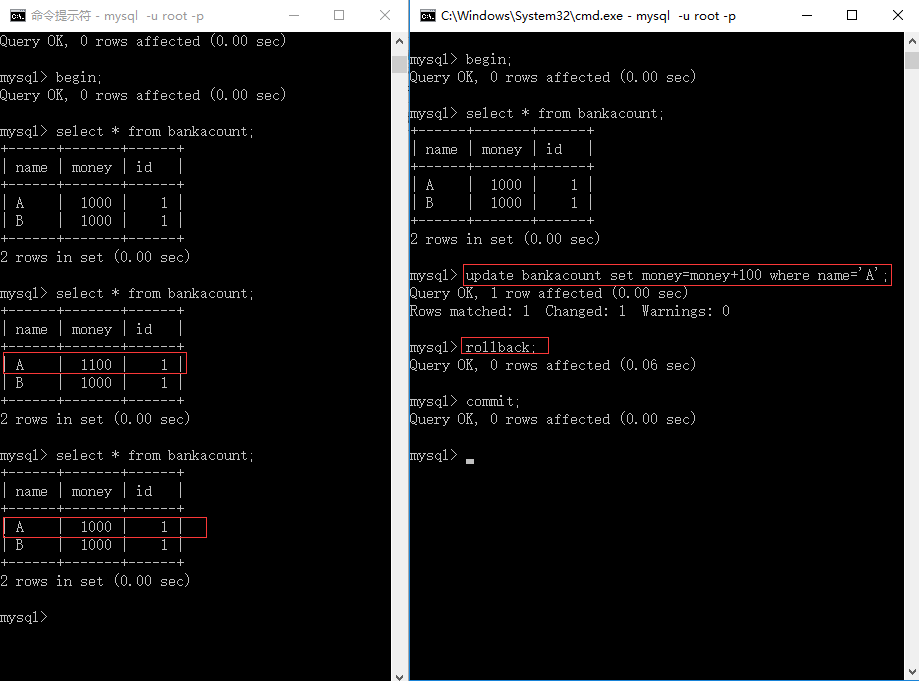
隔离性：事务的隔离性是多个用户并发访问数据库时，数据库为每一个用户开启的事务，不能被其他事务的操作数据所干扰，多个并发事务之间要相互隔离。

持久性：指一个事务一旦被提交，它对数据库中数据的改变就是永久性的，接下来即使数据库发生故障也不应该对其有任何影响。

## 三 事务的隔离级别

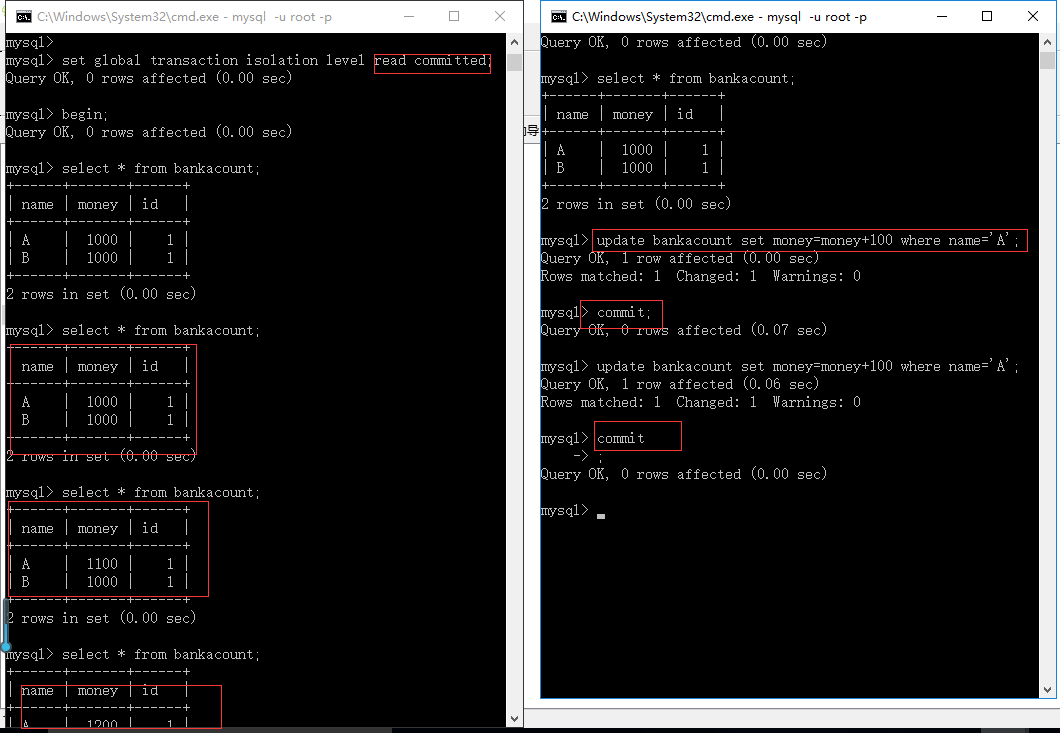
数据库通过设置事务的隔离级别防止以上情况的发生：

READ UNCOMMITTED: 赃读、不可重复读、虚读都有可能发生。



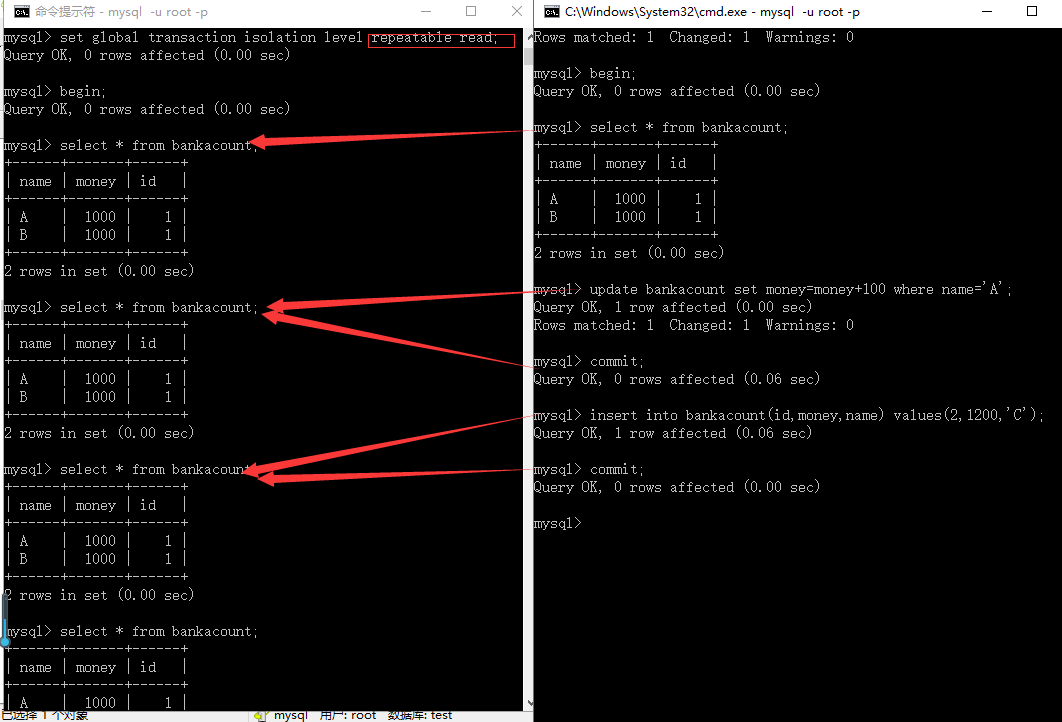
如上图所示开启两个会话，模拟两个线程同时对数据库的bankacount的表进行操作,首先我们改变数据库的隔离级别为READ UNCOMMITTED。此时在两个线程都开启事务，也就相当与两个人同时在柜台服务，都可以查看到A,B两个账户分别有1000元，此时B用户向A用户转账100元，如右图更新语句。然后找A用户确认，A用户查询一看，确是1100元。已经确认了。但是由于B用户还没有提交事务，此时B用户执行回滚操作，并且提交了。A用户再查询的时候，发现变成了1000元，这就是脏读引起的错误。也就是我们说的读未提交。

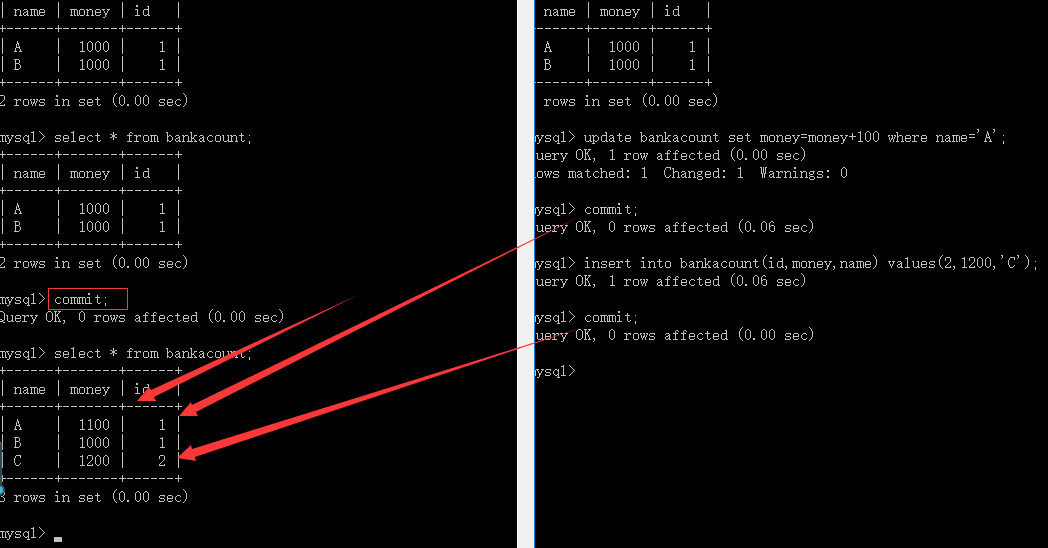
READ COMMITTED: 避免赃读。不可重复读、虚读都有可能发生。（oracle默认的）



如上图所示，为了避免脏读的发生，我们把数据库的事务隔离级别再修改高一个层次，叫做读提交，然而同样的操作，当B用户执行update更新以后，A用户再去查询的时候，发现金额并没有增加，避免了上例脏读的发生，可惜当B提交以后，A又看到了B已提交的数据，一会儿1100，一会儿1200，这样就造成了A账户在自己操作的时候，不断的被干扰，造成A用户的误操作，也就是我们说的读提交，产生了A的数据不可重复读的错误。

REPEATABLE READ:避免赃读、不可重复读。虚读有可能发生。（mysql默认）



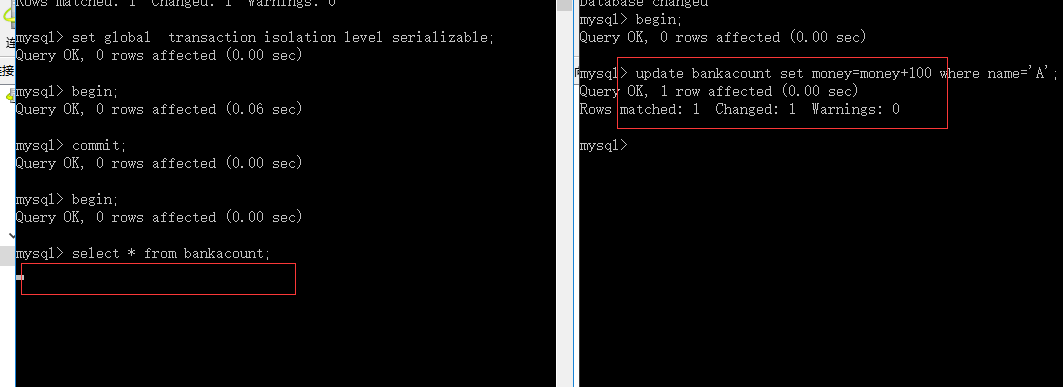


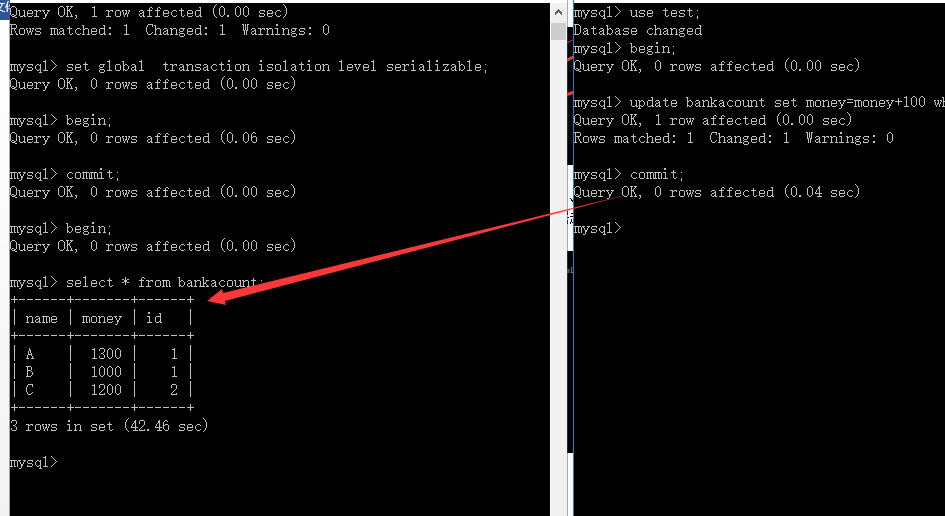
如上图所示，同样的事务操作，此时我们更改它的数据库隔离级别为REPEATABLE READ

当B账户更新，并提交以后A账户查询数据是不变了，保证了A操作不受外界打扰，可以读到重复的数据，这种隔离级别避免了脏读，不可重复读，但是却避免不了幻读，一般可以认为是insert语句，当然也不一定会发生，如上图就没发生幻读，此时当A用户提交查询事务，再查寻的时候，就可以查询到B账户所操作的提交。这样都不影响A,B用户同时操作，又能满足数据的一致性，当然insert语句也有可能产生幻读的现象。

SERIALIZABLE: 避免赃读、不可重复读、虚读。

级别越高，性能越低，数据越安全





如上图所示，为来更进一步的提高并发操作的事务的安全性，我们还可以设置为串行的读，如上图所示B账户在没有提交更新事务的时候，A账户开启来一个事务，然而此时A账户执行查询操作却要等待B用户提交，如图当B用户提交完更新操作以后，A用户查询的数据才能自动出来，这样的操作虽然很安全，但是却影响来性能不能做到并发操作。