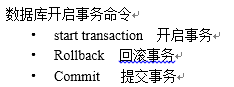
# 事务

## 事务的概念

事务指逻辑上的一组操作，组成这组操作的各个单元，要不全部成功，要不全部不成功。



在开发中，有事务的存在，可以保证数据完整性。

## 性质ACID

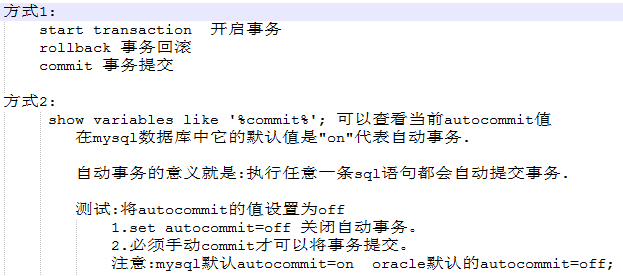
**原子性（Atomicity）**  
 原子性是指事务是一个不可分割的工作单位，事务中的操作要么都发生，要么都不发生。

**一致性（Consistency）**  
 事务前后数据的完整性必须保持一致。(数据库数据完整性约束)

**隔离性（Isolation）--事务与事务之间**  
 事务的隔离性是指多个用户并发访问数据库时，一个用户的事务不能被其它用户的事务所干扰，多个并发事务之间数据要相互隔离。

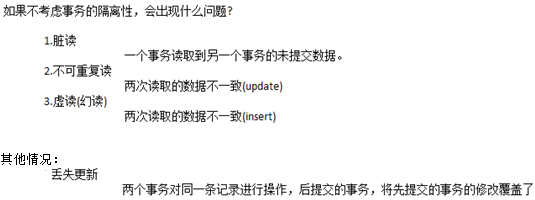
**持久性（Durability）**  
 持久性是指一个事务一旦被提交，它对数据库中数据的改变就是永久性的，接下来即使数据库发生故障也不应该对其有任何影响。

## mysql下提交（事务默认开启）



# 并发问题

## 概述



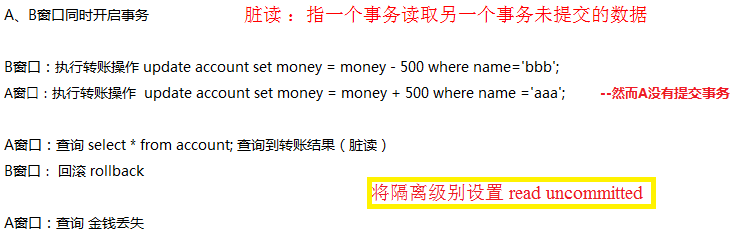
## 不考虑事务隔离性

### 脏读（update中select）

脏读 ：指一个事务读取另一个事务未提交的数据

当一个事务正在访问数据并且对数据进行了修改，而这种修改还没有提交到数据库中，这时另外一个事务也访问这个数据，然后使用了这个数据。因为这个数据是还没有提交的数据，那么另外一个事务读到的这个数据是脏数据，依据脏数据所做的操作可能是不正确的。

事务T1：更新一条数据  
          -->事务T2：读取事务T1更新的记录  
 事务T1：调用commit进行提交  
 此时事务T2读取到的数据是保存在数据库内存中的数据，称为脏数据，这个过程称为脏读。



### 不可重复读（select中update）

不可重复读：在一个事务先后两次读取发生数据不一致情况，第二次读取到另一个事务已经提交数据 （强调数据更新 update）

在同一事务中，两次读取同一数据，得到内容不同。

事务T1：查询一条记录  
         -->事务T2：更新事务T1查询的记录  
         -->事务T2：调用commit进行提交  
事务T1：再次查询上次的记录  
              
此时事务T1对同一数据查询了两次，可得到的内容不同，称为不可重复读。



### 虚读(幻读) (select中insert/delete)

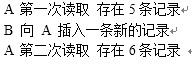
幻读是指当事务不是独立执行时发生的一种现象，例如第一个事务对一个表中的数据进行了修改，这种修改涉及到表中的全部数据行。同时，第二个事务也修改这个表中的数据，这种修改是向表中插入一行新数据。那么，以后就会发生操作第一个事务的用户发现表中还有没有修改的数据行，就好象发生了幻觉一样。

事务T1：查询表中所有记录  
          -->事务T2：插入一条记录  
          -->事务T2：调用commit进行提交  
事务T1：再次查询表中所有记录  
              
此时事务T1两次查询到的记录是不一样的，称为幻读。

注意：幻读重点在新增或删除。

幻读发生在当两个完全相同的查询执行时，第二次查询所返回的结果集跟第一个查询不相同。

在一个事务中，第二次读取发生数据记录数的不同 ，读取到另一个事务已经提交数据 （强调数据记录变化 insert ）

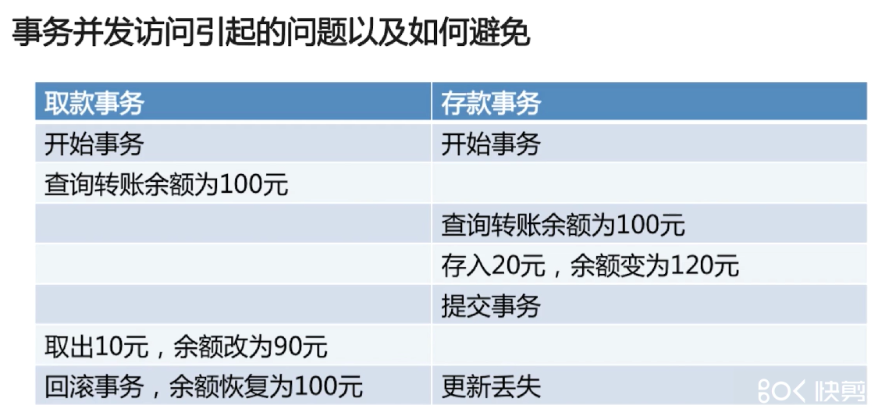


解决幻读问题：采用的是范围锁RangeS RangeS\_S模式，锁定检索范围为只读，这样就避免了幻读问题。

## 其他

### 丢失更新

多个事务对同一条记录进行了操作，后提交的事务将先提交的事务操作覆盖了。



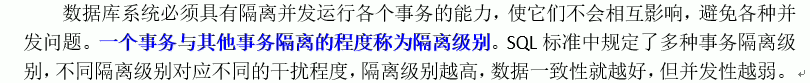
# 解决并发问题

## 事务的隔离级别

### 作用

数据库内部定义了四种隔离级别，用于解决三种隔离问题

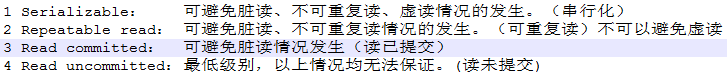
### 隔离级别

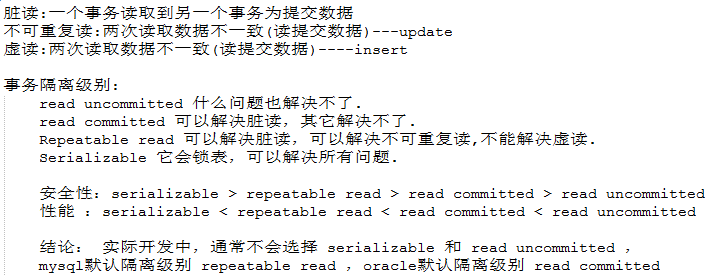


connection.setTransactionIsolation(Connection.TRANSACTION\_READ\_COMMITTED);

V:可能出现，X:不会出现



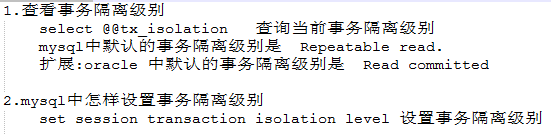




### 各种数据库对隔离级别支持程度



### mysql中设置隔离级别



## 锁（解决丢失更新）

### 悲观锁（共享锁和排他锁）



允许一张数据表中数据记录，添加多个共享锁，添加共享锁记录，对于其他事务可读不可写的

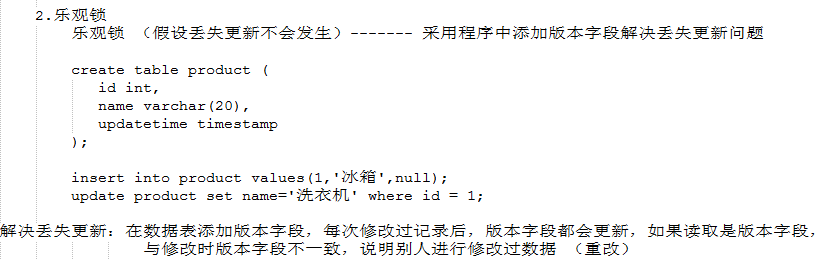
一张数据表中数据记录，只能添加一个排它锁，在添加排它锁的数据 不能再添加其他共享锁和排它锁的 ，对于其他事务可读不可写的

锁必须在事务中添加 ，如果事务结束了 锁就释放了

解决丢失更新：事务在修改记录过程中，锁定记录，别的事务无法并发修改

### 乐观锁





timestamp 在插入和修改时 都会自动更新为当前时间

解决丢失更新：在数据表添加版本字段，每次修改过记录后，版本字段都会更新，如果读取是版本字段，与修改时版本字段不一致，说明别人进行修改过数据 （重改）