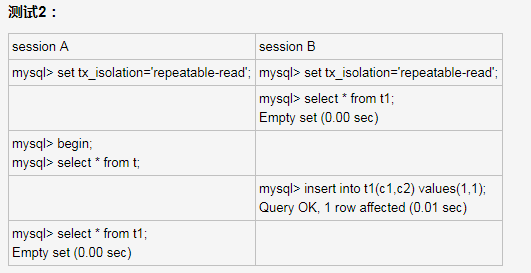


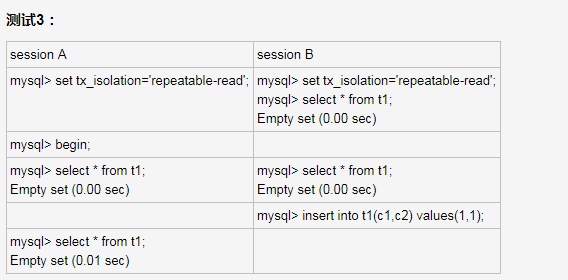
**结论1：RR隔离级别下的一致性读，不是以begin开始的时间点作为snapshot建立时间点，因为测试看出再begin一个事务的时候，表是没有数据的，sessionB写入数据之后，却能看到数据。**

**注意会话b中的insert是自动提交的，没有设置begin，所以a读取的是提交的数据**

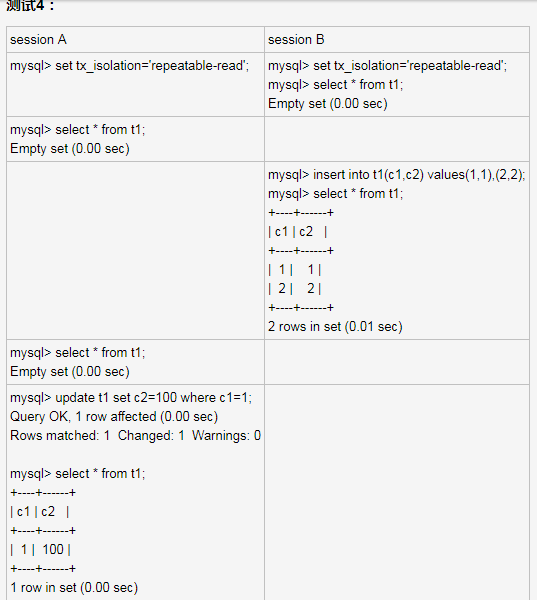


**结论2：RR隔离级别下的一致性读，是以第一条select语句的执行点作为snapshot建立的时间点的，即使是不同表的select语句。这里因为session A在insert之前对 t 表执行了select，所以建立了snapshot，所以后面的select \* from t1 不能读取到insert的插入的值(snapshot的时候t1表没有数据)。**

这个就是验证了，快照是在第一个select开始执行的时候建立的。



这个测试就是之前的验证补充



测试4说明的是，update语句是当前读，获取的是当前的数据，能更新并且在更新后，读取的不是之前的快照中的数据了，而是最新的数据

**通过上面的几个测试得出的结论：对同一个表或者不同表进行的第一次select语句建立了该事务中一致性读的snapshot，在snapshot建立之后提交的数据，一致性读就读不到，之前提交的数据就可以读到。事务一致性读的起始点其实是以执行的第一条语句为起始点的，而不是以begin作为事务的起始点的。**

一般begin/start transaction是开始一个事务的标志，但不是事务开始的时间点，也就是说执行了start transaction之后的第一个语句（任何语句），事务才真正的开始。但是如果要达到将 start transaction作为事务开始的时间点，那么必须使用：

**START TRANSACTION WITH consistent snapshot** **###mysqldump中的快照就是用这个实现的**

**这样开启事务效果等价于： start transaction 之后，马上执行一条 select 语句(此时会建立一致性读的snapshot)**

