华南理工大学

《数据库》课程实验报告

实验题目： 实验四：事务的管理

|  |
| --- |
| **实验概述** |
| 【实验目的及要求】  实验目的：   1. 通过编写事务隔离级别的测试程序了解事务并发控制机制 2. 学会进行数据库的备份与恢复。   实验要求：   1. 采用实验一的建库脚本和数据插入脚本创建Student数据库。 2. 测试事务隔离级别，要求：   分别设置不同的隔离级别, 让两个并发事务交错执行的程序或事务，能分别显示每种隔离级别下，是否出现丢失更新，脏读，读值不可复现以及幻象记录四种情况。  提示：  **Oracle**  SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;  SET TRANSACTION ISOLATION LEVEL READ COMMITTED;  SET TRANSACTION ISOLATION LEVEL READ ONLY;  **SQL Server**  SET TRANSACTION ISOLATION LEVEL SERIALIZABLE  SET TRANSACTION ISOLATION LEVEL REPEATABLE READ  SET TRANSACTION ISOLATION LEVEL READ COMMITTED  SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED   1. 备份与恢复 2. 备份数据库 3. 删除sc表 4. 恢复到删除之前 5. 提交实验报告：   实验报告包括实验过程和实验结果截屏，各种SQL脚本，测试过程的详细说明与分析。  【实验环境】  PC机，WINDOWS操作系统，Oracle 或Microsoft SQL Server 数据库 |
| **实验内容** |
| 【实验过程】   1. 实验步骤： 2. 创建数据库 3. 测试事务隔离级别 4. 实验主要过程：   1.事务的基本要素（ACID）  　　（1）原子性（Atomicity）：事务开始后所有操作，要么全部做完，要么全部不做，不可能停滞在中间环节。事务执行过程中出错，会回滚到事务开始前的状态，所有的操作就像没有发生一样。也就是说事务是一个不可分割的整体，就像化学中学过的原子，是物质构成的基本单位。  （2）一致性（Consistency）：事务开始前和结束后，数据库的完整性约束没有被破坏。  　　（3）隔离性（Isolation）：同一时间，只允许一个事务请求同一数据，不同的事务之间彼此没有任何干扰。比如A正在从一张银行卡中取钱，在A取钱的过程结束前，B不能向这张卡转账。  　　 （4）持久性（Durability）：事务完成后，事务对数据库的所有更新将被保存到数据库，不能回滚。  2.事务的并发问题  　　（1）脏读：事务A读取了事务B更新的数据，然后B回滚操作，那么A读取到的数据是脏数据  　　（2）不可重复读：事务 A 多次读取同一数据，事务 B 在事务A多次读取的过程中，对数据作了更新并提交，导致事务A多次读取同一数据时，结果 不一致。  　　（3）幻读：系统管理员A将数据库中所有学生的成绩从具体分数改为ABCDE等级，但是系统管理员B就在这个时候插入了一条具体分数的记录，当系统管理员A改结束后发现还有一条记录没有改过来，就好像发生了幻觉一样，这就叫幻读。  　　小结：不可重复读的和幻读很容易混淆，不可重复读侧重于修改，幻读侧重于新增或删除。解决不可重复读的问题只需锁住满足条件的行，解决幻读需要锁表。   1. MySQL事务隔离级别  |  |  |  |  | | --- | --- | --- | --- | | 事务隔离级别 | 脏读 | 不可重复读 | 幻读 | | 读未提交（read-uncommitted） | Y | Y | Y | | 不可重复读（read-committed） | N | Y | Y | | 可重复读（repeatable-read） | N | N | Y | | 串行化（serializable） | N | N | N |   4.测试事务隔离级别，要求：  分别设置不同的隔离级别, 让两个并发事务交错执行的程序或事务，能分别显示每种隔离级别下，是否出现丢失更新，脏读，读值不可复现以及幻象记录四种情况。   1. 读未提交   (1)设置事务T1，模式为read uncommitted（未提交读），查询表SC的初始值：  ../../../屏幕快照%202019-01-19%20上午1.08.42.png  SC表格初始值：  ../../../屏幕快照%202019-01-19%20上午1.15.06.png  (2)设置事务 T2，模式为read uncommitted（未提交读），更新 SC 表中数据：  ../../../屏幕快照%202019-01-19%20上午1.09.07.png  ../../../屏幕快照%202019-01-19%20上午1.30.11.png  （3）T1再次查询 SC 表：  ../../../屏幕快照%202019-01-19%20上午1.32.51.png  （4）T2回滚  ../../../屏幕快照%202019-01-19%20上午1.37.21.png  结论：虽然T2还没提交，但是T1已经可以查询到T2更新的数据。一旦客户端T2因为某种原因回滚，所有的操作都将会被撤销，那T1查询到的数据其实就是脏数据。   1. 读已提交   （1）设置事务 T1模式为read committed（未提交读），查询表sc的初始记录：  ../../../屏幕快照%202019-01-19%20上午1.41.58.png  （2）设置事务 T2模式为read committed（未提交读），更新表sc：  未更新前：  ../../../屏幕快照%202019-01-19%20上午1.48.00.png  更新后：  ../../../屏幕快照%202019-01-19%20上午1.49.11.png  （3）T1 再次查询sc表  ../../../屏幕快照%202019-01-19%20上午1.49.33.png  T1无法读出 T2已经更新的数据，避免了读脏数据的可能  （4）T2提交事务  ../../../屏幕快照%202019-01-19%20上午1.56.30.png  （5）T1查询：  ../../../屏幕快照%202019-01-19%20上午1.57.15.png  T1执行与上一步相同的查询，结果却得到了与上一步不一致的结果，即产生了不可重复读的问题。   1. 可重复读   （1）T1事务模式为repeatable read，查询sc 表的所有记录  ../../../屏幕快照%202019-01-19%20上午2.03.23.png  （2）T1事务模式为repeatable read，更新 sc 表：  ../../../屏幕快照%202019-01-19%20上午2.06.05.png  （3）T1查询sc表：  ../../../屏幕快照%202019-01-19%20上午2.09.55.png  结果与（1）一致，没有出现不可重复读的问题  （4）T1更新 sc：  ../../../屏幕快照%202019-01-19%20上午11.34.56.png  grade没有变成99-20=79，grade值用的是根据步骤（2）中的88来算的，所以是68，数据的一致性没有被破坏。可重复读的隔离级别下使用了MVCC机制，select操作不会更新版本号，是快照读（历史版本）；insert、update和delete会更新版本号，是当前读（当前版本）。  T2查询：  ../../../屏幕快照%202019-01-19%20上午11.34.45.png  查询结果与上次一致，可重复读   1. 串行化   （1）设置 T1事务模式为serializable，查询表sc的初始值：  ../../../屏幕快照%202019-01-19%20上午11.39.54.png  （2）设置 T2事务模式为serializable，插入一条数据  ../../../屏幕快照%202019-01-19%20上午11.52.33.png  由于表被锁所以插入失败。当mysql中事务隔离级别为serializable时会锁表，因此不会出现幻读的情况，这种隔离级别并发性极低，开发中很少会用到。   1. 数据库备份   使用软件：  屏幕快照 2018-12-10 上午9.23.49   1. 选择左侧菜单栏中的备份，点击“备份”按钮   屏幕快照 2018-12-10 上午9.19.59   1. 进入备份界面，点击“开始”进行备份   屏幕快照 2018-12-10 上午9.20.19屏幕快照 2018-12-10 上午9.20.30屏幕快照 2018-12-10 上午9.20.44   1. 备份成功，关闭界面   ~~屏幕快照 2018-12-10 上午9.21.13~~  屏幕快照 2018-12-10 上午9.21.26  （4）恢复备份：右键选择备份文件，点击“还原备份”  屏幕快照 2018-12-10 上午9.21.40  屏幕快照 2018-12-10 上午9.22.02  （5）点击“开始”、“确定”  屏幕快照 2018-12-10 上午9.22.19  （6）还原成功  屏幕快照 2018-12-10 上午9.23.05 |