1.事务概念

(1).Transaction其实指的是一组操作, 里面包含许多个单一的逻辑。只要有一个逻辑没有执行成功, 那么都算失败。所有的数据都回归到最初的状态。

2.为什么要有事务

(1).为了确保逻辑的成功。例子: 银行的转账。

3.事务的特性

(1).原子性: 指的是事务中包含的逻辑, 不可分割。

(2).一致性: 指的是事务执行前后, 数据完整性。

(3).隔离性: 指的是事务在执行期间不应该受到其他事务的影响。

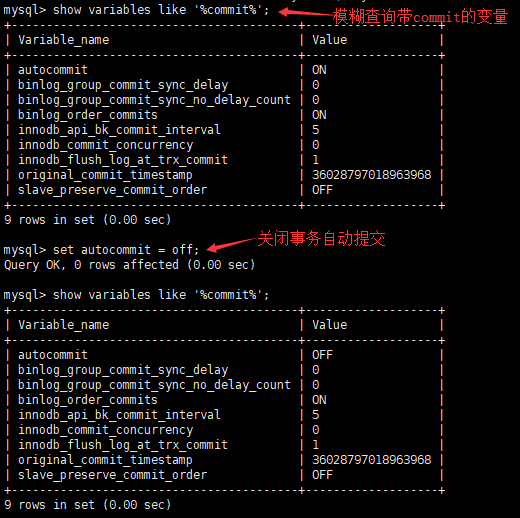
(4).持久性: 指的是事务执行成功, 那么数据应该持久保存到磁盘上。

4.开启事务: start transaction

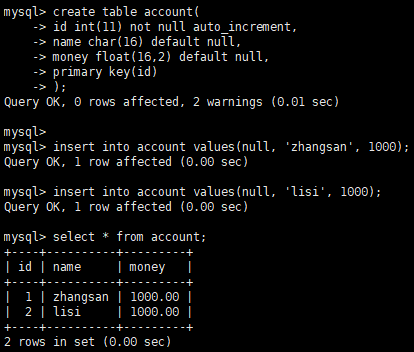
5.提交事务commit, 数据将会写到磁盘上的数据库。

6.回滚事务rollback, 数据回滚到最初的状态。

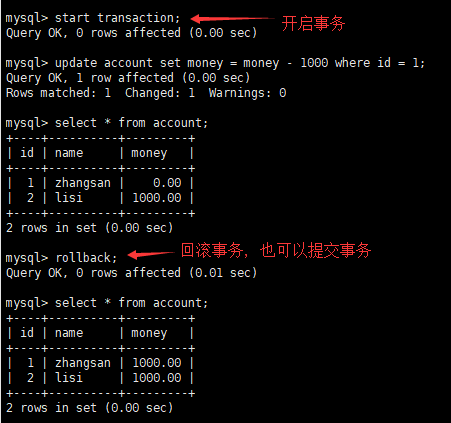
7.查询和关闭事务的自动提交



8.创建一张account表, 并插入2条数据。



9.演示事务



10.代码演示事务

|  |
| --- |
| package com.lywgames.myjdbc;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import com.lywgames.util.JDBCUtil;  public class MyJDBC {  public static void main(String[] args) {  Connection conn = null;  PreparedStatement ps = null;  ResultSet rs = null;    try {  conn = JDBCUtil.getConn();  // 1.关闭事务自动提交  conn.setAutoCommit(false);    // 2.扣除用户1的钱  ps = conn.prepareStatement("update account set money = money - ? where id = ?");  ps.setInt(1, 100);  ps.setInt(2, 1);  ps.executeUpdate();    // 3.抛一个运行时异常  new RuntimeException();    JDBCUtil.closeSt(ps);  // 4.给用户2加钱  ps = conn.prepareStatement("update account set money = money + ? where id = ?");  ps.setInt(1, 100);  ps.setInt(2, 2);  ps.executeUpdate();    // 5.提交事务  conn.commit();  } catch (SQLException e) {  try {  // 6.回滚事务  conn.rollback();  } catch (SQLException e1) {  e1.printStackTrace();  }  e.printStackTrace();  } finally {  JDBCUtil.release(conn, ps, rs);  }    }  } |