Java MySQL系列课程09

------ 事务处理

本文档包含以下内容：

1. 事务的概念
2. 事务的ACID特性
3. 事务的隔离级别
4. JDBC如何处理事务
5. MySQL的事务处理命令

#### 前言：

我们用Java JDBC操作数据库时，事务都是默认提交的，相当于一条SQL语句执行之前开启事务，SQL语句执行完成后，事务再进行提交，那么有的时候，事务不仅仅代表一条SQL语句，而是代表一组做增删改查操作的SQL语句，此时事务就得保持一定的特性，否则就会导致数据库数据不一致，或者在数据库做并发操作时，导致严重的后果，本章内容我们就来看看事务的基本概念，看看在校招中事务常常考察的两个问题：

**“事务的ACID特性是什么？”**

**“事务的隔离级别有哪些？”**

#### 一 事务的概念：

一个事务是由一条或者多条对数据库操作的SQL语句所组成的一个不可分割的单元，只有当事务中的所有操作都正常执行完了，整个事务才会被提交给数据库；如果有部分事务处理失败，那么事务就要回退到最初的状态，因此，事务要么全部执行成功，要么全部失败。

事务也可以设置保存点SavePoint，当事务执行发生错误的时候，可以选择回滚到保存点。

在JDBC编程中，事务默认是自动提交的，事务是通过commit提交或者rollback来回滚的。



**可以通过JDBC Connection连接对象的setAutoCommit方法来设置事务是否自动提交**。

**所以记住事务的几个基本概念**，如下：

1. 事务是一组SQL语句的执行，要么全部成功，要么全部失败，不能出现部分成功，部分失败的结果。保证事务执行要的原子操作。
2. 事务的所有SQL语句全部执行成功，才能提交（commit）事务，把结果写回磁盘上。
3. 事务执行过程中，有的SQL出现错误，那么事务必须要回滚（rollback）到最初的状态。

#### 二 事务的ACID特性：

**每一个事务必须满足下面的4个特性：**

**事务的原子性（Atomic）：**

事务是一个不可分割的整体，**事务必须具有原子特性**，及当数据修改时，要么全执行，要么全不执行，即不允许事务部分的完成。

**事务的一致性（Consistency）：**

一个事务执行之前和执行之后，数据库数据必须保持一致性状态。数据库的一致性状态必须**由用户来负责，由并发控制机制实现**。就拿网上购物来说，你只有让商品出库，又让商品进入顾客的购物车**才能构成一个完整的事务**！由于并发操作带来的**数据不一致性包括读脏数据（脏读），不可重复读和虚读（幻读）**。

**事务的隔离性（Isolation）：**

当两个或者多个事务并发执行时，为了保证数据的安全性，将一个事物内部的操作与其它事务的操作隔离起来，不被其它正在执行的事务所看到，例如对任何一对事务T1和T2，对T1而言，T2要么在T1开始之前已经结束，要么在T1完成之后再开始执行。**隔离性使得每个事务的更新在它被提交之前，对其它事务都是不可见的**。

让我们看看事务处理之父Jim Gray对事务隔离性的定义[1]：

Isolation: Concurrently executing transactions see the stored information as if they were running serially (**one after another**).

**事务的持久性（Durability）：**

事务完成以后，DBMS保证它对数据库中的数据的修改是永久性的，即使数据库因为故障出错，也应该能够恢复数据！

#### 三 事务的隔离级别：

**事务处理不经隔离，并发执行事务时通常会发生以下的问题**：

**脏读（Dirty Read）**：一个事务读取了另一个事务未提交的数据。例如当事务A和事务B并发执行时，当事务A更新后，事务B查询读取到A尚未提交的数据，此时事务A回滚，则事务B读到的数据就是无效的脏数据。**（事务B读取了事务A尚未提交的数据）**

**不可重复读（NonRepeatable Read）**：一个事务的操作导致另一个事务前后两次读取到不同的数据。例如当事务A和事务B并发执行时，当事务B查询读取数据后，事务A更新操作更改事务B查询到的数据，此时事务B再次去读该数据，发现前后两次读的数据不一样。**（事务B读取了事务A已提交的数据）**

**虚读（Phantom Read）/幻读**：一个事务的操作导致另一个事务前后两次查询的结果数据量不同。例如当事务A和事务B并发执行时，当事务B查询读取数据后，事务A新增或者删除了一条满足事务B查询条件的记录，此时事务B再去查询，发现查询到前一次不存在的记录，或者前一次查询的一些记录不见了。**（事务B读取了事务A新增加的数据或者读不到事务A删除的数据）**

由于多个线程会请求相同的数据，事务之间通常都会用锁互相隔离，由于数据库支持不同类型的锁，因此Java JDBC支持不同级别的事务处理，它们由Connection对象指定。在JDBC中，定义了以下5种事务隔离级别：

1. **TRANSACTION\_NONE**。 **表示不支持事务**

2. **TRANSACTION\_READ\_UNCOMMITTED**。**未提交读**。说明在提交前一个事务可以看到另一个事务的变化。这样读”脏”数据，不可重复读和虚读都是被允许的。

3. **TRANSACTION\_READ\_COMMITTED**。**已提交读**。说明读取未提交的数据是不允许的。这个级别仍然允许不可重复读和虚读产生。

4. **TRANSACTION\_REPEATABLE\_READ**。**可重复读**。说明事务保证能够再次读取相同的数据而不会失败，但虚读仍然会出现。

5. **TRANSACTION\_SERIALIZABLE**。**可序列化/串行化**。是最高的事务级别，它防止读脏数据，不可重复读和虚读。

|  |  |  |  |
| --- | --- | --- | --- |
| 隔离级别 | 脏读 | 不可重复读 | 幻读 |
| 未提交读 | 可以 | 可以 | 可以 |
| 已提交读 | 不可以 | 可以 | 可以 |
| 可重复读 | 不可以 | 不可以 | 可以 |
| 串行化 | 不可以 | 不可以 | 不可以 |

需注意：**事务隔离级别越高，为避免冲突所花费的性能也就越多**。可以通过Connection接口下面的函数来设置事务的隔离级别：



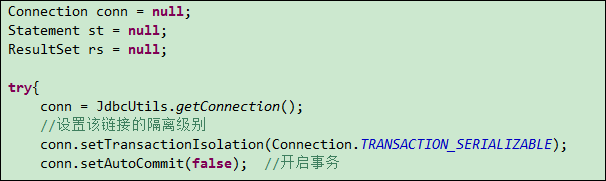
#### 四 JDBC如何处理事务：

在JDBC当中，可以通过调用**setAutoCommit(false)**方法来**禁止自动提交事务**，然后就可以把多个数据库操作的SQL表达式作为一个事务，在操作完成以后调用commit方法来实现事务提交，如果其中一个表达式失败，就会抛出异常而不会调用commit。在这种情况下，就可以在**异常代码处理中调用rollback**对已经发生的事务操作进行回滚。通过此种方法可以保持对数据库进行多次操作后，数据仍然是保持一致的。

如果是使用JDBC对数据库的事务设置隔离级别的话，也应该**是在调用Connection对象的setAutoCommit(false)方法之前，调用Connection对象的setTransactionIsolation(level)即可设置当前链接的隔离级别**，至于参数level，可以使用Connection对象的字段：



在JDBC中设置隔离级别的部分代码：



隔离级别的设置**只对当前链接有效**。对于使用MySQL命令窗口而言，一个窗口就相当于一个链接，也称作一个会话，当前窗口设置的隔离级别只对当前窗口中的事务有效；对于JDBC操作数据库来说，一个Connection对象相当于一个链接（或者称一个会话），而对于Connection对象设置的隔离级别只对该Connection对象有效，与其他链接Connection对象无关，也就是说**每个Connection对象都需要设置各自的隔离级别**。

#### 二 MySQL的事务处理命令：

打开MySQL的Command命令行窗口，测试以下命令：

1. **SELECT @@AUTOCOMMIT; 查看MySQL是否自动提交事务**

mysql> **select @@autocommit;**

+--------------+

| @@autocommit |

+--------------+

| 1 | **# 0表示手动提交事务 1表示自动提交事务**

+--------------+

1 row in set (0.01 sec)

1. **SET AUTOCOMMIT = 0;**

设置事务提交方式 0表示手动提交事务 1表示自动提交事务

1. **BEGIN;** 开启一个事务
2. **COMMIT;** 提交一个事务
3. **ROLLBACK;**  回滚一个事务到初始的位置
4. **SAVEPOINT point1;** 设置一个名字为point1的保存点
5. **ROLLBACK TO point1;** 事务回滚到保存点point1，而不是回滚到初始状态
6. **SET GLOBAL TX\_ISOLATION='REPEATABLE-READ';** 设置事务的隔离级别
7. **SELECT  @@ TX\_ISOLATION;** 查询事务的隔离级别

mysql> **select @@tx\_isolation;**

+-----------------+

| @@tx\_isolation |

+-----------------+

| REPEATABLE-READ | // **MySQL默认工作在“可重复读”的隔离级别**

+-----------------+

1 row in set (0.00 sec)