# 一、事务的基本概念

数据库中的事务是逻辑任务，一个事务由一个或一组SQL语句组成，事务机制可确保这一组SQL语句要么全部执行成功，完成整个任务，如果任何一个SQL语句执行失败，对数据库所做的操作可全部撤销。

事务的目的是为了保证数据的完整性，以银行转账业务为例：

A用户向B转账1000元，步骤如下：

1）A用户账户余额减少1000元；

2）B用户账户余额增加1000元；

3）记录A用户的交易日志。

4）记录B用户的交易日志。

一笔转帐交易将产生四次数据库操作，要么全部都成功，如果有一个失败，其它操作也应该全部被撤销。

// C++

加锁();

if (修改账户余额(A,-1000)==false) break;

if (修改账户余额(B,1000)==false) { 修改账户余额(A用户,1000); break; }

if (记录交易日志(A)==false)

{

修改账户余额(A用户,1000);

修改账户余额(B用户,-1000); break;

}

if (记录交易日志(B)==false)

{

修改账户余额(A用户,1000);

修改账户余额(B用户,-1000);

删除记录交易日志(A); break;

}

解锁();

// Oracle事务

开始事务;

if (修改账户余额(A,-1000)==false) break;

if (修改账户余额(B,1000)==false) { 回滚事务; break; }

sleep(100);

if (记录交易日志(A)==false) { 回滚事务; break; }

if (记录交易日志(B)==false) { 回滚事务; break; }

提交事务; // 解锁。

// A的余额是10000元，B的余额也是10000元。

// 其它进程查看A的余额仍是10000元，B的余额也是10000元。

// 其它进程无法修改A和B的余额（A和B的记录已被Oracle自动加锁）。

// 其它进程查看A的余额是9000元，B的余额是11000元，A和B的交易日志。

转账交易有四次数据库操作，即四条SQL语句，第一条SQL执行的时候是事务的开始，如果全部的SQL语句执行成功，提交事务，如果有任何一条SQL语句执行失败，回滚事务。

begin; -- 开始事务，默认从第一条操作了数据的SQL语句开始。

commit; -- 提交事务。

rollback; -- 回滚事务。

# 二、事务的特征

对一组SQL语句操作构成的事务，数据库系统必须保证这些操作的原子性、一致性、隔离性和持久性，即ACID原则。

## 1、原子性（Atomicity）

事务的原子性是指事务中包含的所有操作要么全做，要么不做，也就是说所有的操作在数据库中要么全部生效，要么全部不生效。

## 2、一致性（Consistency）

事务的一致性是指数据库在事务操作前和事务处理后，数据都是一致的。

## 3、隔离性（Isolation）

隔离性是指数据库允许多个并发的事务同时对数据进行读写或修改，即使出现了SQL语句叉操作数据的情况，也不会导致数据不一致。

## 4、持久性（Durability）

事务的持久性是指在事务处理结束后，它对数据的修改应该是永久的。通俗的理解就是事务结束后立即把数据写入硬盘中。

# 三、事务控制

**事务控制语句中最重要的两个指令，提交（commit）和回滚（rollback），可以满足99%的应用需求。**

本文接下来介绍事务控制的更多知识，大家了解一下就可以了，不必深入研究。

## 1、设置事物属性

set transaction可以用来设置事务的各种状态，比如只读、读/写、隔离级别，为事务分配名称或将事务分配回滚段等等。

set transaction [read only|read write]

[isolation level [serialize|read commited]]

[use rollback segment 'segment\_name']

[name 'transaction\_name'];

read only：将事务设置为只读事务。

read write：将事务设置为读/写事务。

isolation level：如果指定，它有两个选项：（1）serialize：如果事务尝试更新由另一个事务更新并未提交的资源，则事务将失败。（2）read commited：如果事务需要另一个事务持有的行锁，则事务将等待，直到行锁被释放。

use rollback segment：如果指定，它将事务分配给由'segment\_name'标识的回退段，该段是用单引号括起来的段名称。

name：为事务分配一个名称'transaction\_name'，该名称是用单引号括起来。

注意：

1）set transaction是事务处理的第一条语句，也就是说，必须在任何insert、update 、delete语句以及其他的事务处理之前。

2）在使用set transaction语句设置事务的属性时，很少指定回滚段，对于命名事务也非常简单，只有在分布式事务处理中才会体现出事务命名的用途。

## 2、数据异常

事务的隔离性定义了一个事务与其它事务的隔离程度，为了更好的理解隔离层，首先讨论一下并发事务对同一个数据库进行访问可能发生的情况，在并发事务中总体来说会发生如下3种情况：

**错读|脏读**

**非重复读取|不可重复读**

**假读|幻读**

**错读|脏读**：当一个事务修改数据时，另一事务读取了该数据，但是第一事务由于某种原因取消对数据修改，使数据返回了原状态，这是第二个事务读取的数据与数据库中数据不一致，这就叫错读。

**非重复读取**：是指一个事务读取数据库中的数据后，另一个事务则更新了数据，当第一个事务再次读取其中的数据时，就会发现数据已经发生了改变,这就是非重复读取。非重复读取所导致的结果就是一个事务前后两次读取的数据不相同。

**假读**：如果一个事务基于某个条件读取数据后，另一个事务则更新了同一个表中的数据，这时第一个事务再次读取数据时，根据搜索的条件返回了不同的行，这就是假读。

 事务中遇到的这些异常与事务的隔离性设置有关，事务的隔离性设置越多，异常就出现的越少，但并发效果就越低，事务的隔离性设置越少，异常出现的越多，并发效果越高。

## 3、选择隔离层

 针对上文中读取的数据时产生的不一致现象，在ANSI SQL标准92中定义了4个事务的隔离级别，如下图所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 隔离层 | 错读|脏读 | 非重复读取|不可重复读 | 假读|幻读 |
| READ UNCOMMITTED（非提交读） | 是 | 是 | 是 |
| READ COMMITTED（提交读） | 否 | 是 | 是 |
| Repeatable READ（可重复读） | 否 | 否 | 是 |
| Serializable（串行读） | 否 | 否 | 否 |

 Oracle支持上述四种隔离层中的两种：read committed 和serializable，除此之外oralce中还定义read only 和 read write隔离层。

 read committed：这是oracle默认的隔离层。

serializable：事务与事务之间完全隔开，事务以串行的方式执行，这并不是说一个事务必须结束才能启动另外一个事务，而是说这些事务的执行的结果于一次执行的事务的结果一致。

 read only和 read write 当使用read only时，事务中不能有任何修改数据库中数据的操作语句，这包括 insert、update、delete、create语句。read only是serializable的一个子集，区别是read only 只读，而serialzable可以执行DML操作。read write它是默认设置，该选项表示在事务中可以有访问语句和修改语句，但很少使用。

## 4、保存点

保存点（savepoint）是在数据库事务处理中实现“子事务”（subtransaction），也称为嵌套事务的方法，事务可以回滚到 savepoint 而不影响 savepoint 创建前的变化，不需要放弃整个事务。

rollback回滚的用法可以设置保留点 savepoint，执行多条SQL操作时，回滚到想要的那条SQL之前。

savepoint savepoint\_name; -- 声明一个savepoint。

rollback to savepoint\_name; -- 回滚到savepoint。