**事务**

**事务的详解**

1. 事务是恢复和控制并发情况发生的基本单位

-> 我们把一段高并发/出错的SQL语句放在一个事务中执行，当事务运行到事务结束的语句时，这一段就是一次控制并发情况发生的事务

1. 事务中执行的SQL语句，必须遵循事务的ACID特性

**遵循ACID特性**

-> 我们在一个事务中写一段SQL脚本，如果该SQL脚本遵行了ACID的特性，那么这个事务就是成功了

-> 事务成功：开启事务和提交事务之间所写的对数据库数据修改的SQL脚本就会成功的执行，也就是能够成功的把所操作的数据提交至数据库当中

**未遵循ACID特性**

-> 我们在一个事务中写一段SQL脚本，如果该SQL脚本没有遵行ACID的特性，那么这个事务就是失败了

-> 事务失败：开启事务和提交事务之间所写的对数据库数据修改的SQL脚本就会失效，也就是事务回滚了，数据会恢复到原始的状态

1. 事务的语法必须要搭配**try-catch**的使用，才能构成一个完整的事务
2. 要求保证数据完整性的，必须使用事务

**事务的ACID特性**

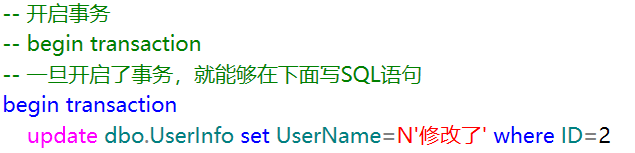
1. 原子性（atomicity）：一个事务是一个不可分割的工作单位，事务中包括的SQL操作要么都成功，要么都不成功
2. 一致性（consistency）：事务必须是使数据库从一个一致性状态变到另一个一致性状态，（比方说：A借了B 100块钱，那么A的资产就要减100，B的资产就要加100）,一致性与原子性是密切相关的
3. 隔离性（isolation）：一个事务的执行不能被其他事务干扰，即一个事务内部的操作及使用的数据对并发的其他事务是隔离的，并发执行的各个事务之间不能互相干扰，**通过锁实现**
4. 持久性（durability）：持续性也称永久性（permanence），指一个事务一旦成功提交，那么它对数据库中数据的改变就应该是永久性的，并且接下来的其他操作或故障不应该对其有任何影响

**使用事务时候的一些注意事项**

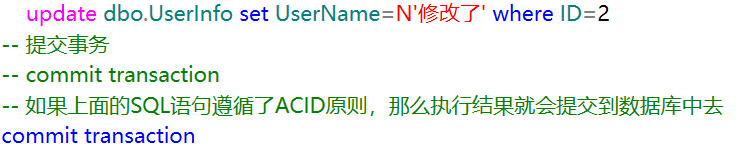
1. 嵌套事务的注意事项，嵌套事务内不能存在多次回滚：**<https://blog.csdn.net/xmzhaoym/article/details/5120372>**

**事务的语法**

1. 开启事务：当我们开启了事务，我们就能够在开启事务的下面写一些SQL语句了

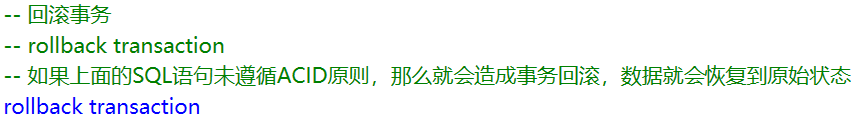


1. 提交事务：当我们提交了事务，如果上面的SQL语句成功遵循了ACID的原则，那么执行结果就会提交到数据库中去



1. 回滚事务：当开启事务和提交事务之间的SQL语句未遵循ACID原则，便会执行这条语句，使数据恢复到原始的状态

**配合try-catch的使用才能够使用回滚事务的功能**



1. **事务配合try-catch的使用**

