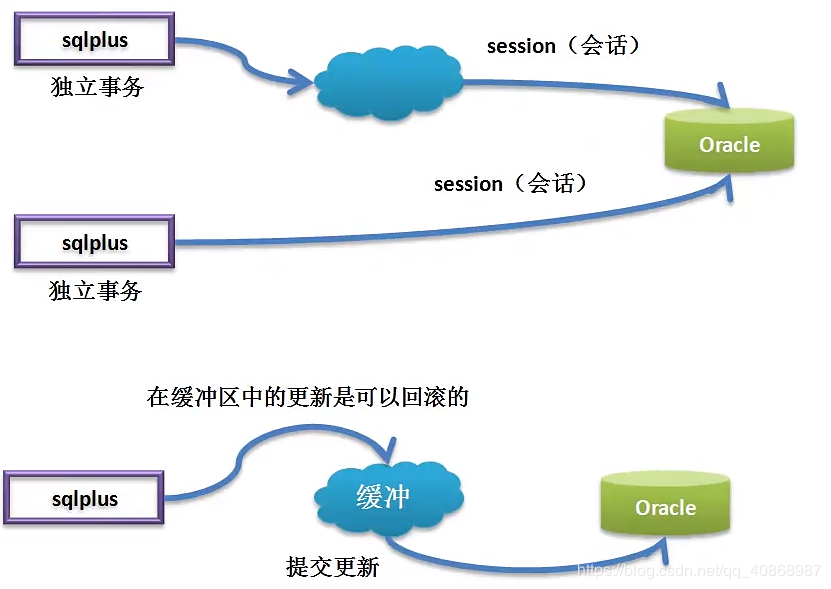
# 事务处理基本概念

事务是保证数据完整性的一种手段，事物具备ACID原则。



1）事务的原子性(Atomicity)：是指一个事务要么全部执行，要么不执行，也就是说一个事务不可能只执行了一半就停止了。比如你从取款机取钱，这个事务可以分成两个步骤：1划卡，2出钱。不可能划了卡，而钱却没出来。这两步必须同时完成，要么就不完成。

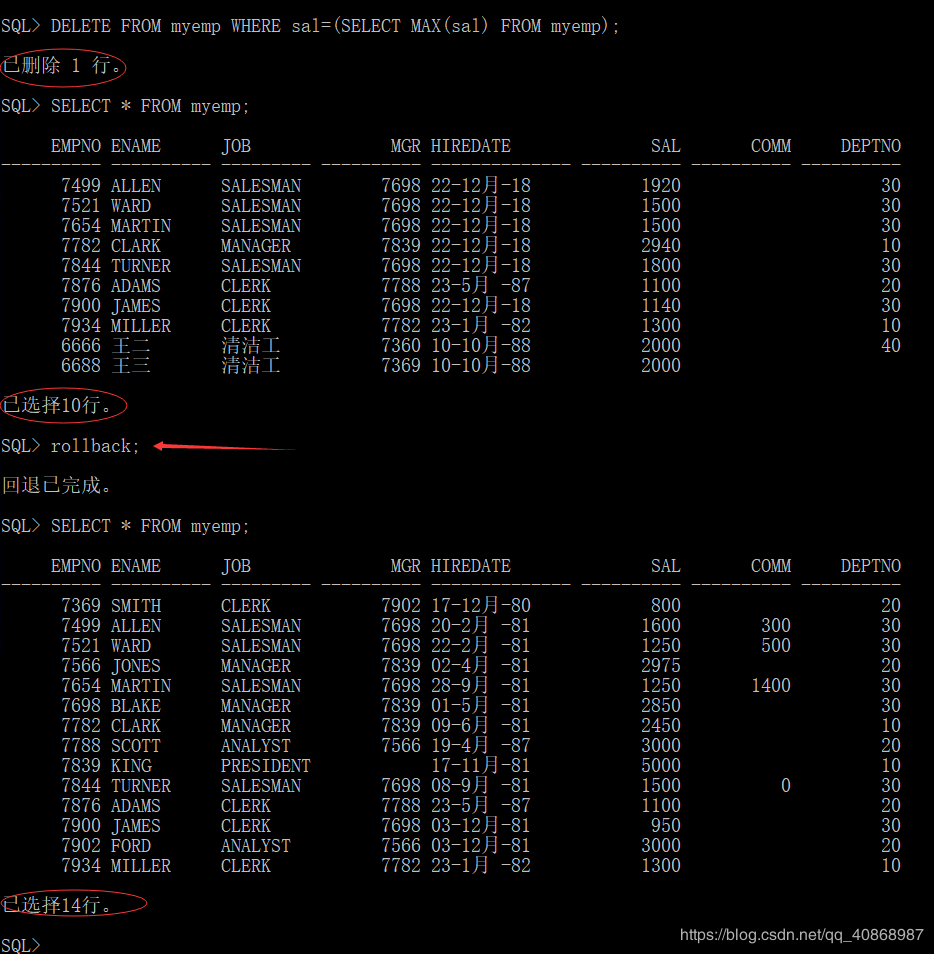
2）[事务](https://baike.baidu.com/item/%E4%BA%8B%E5%8A%A1)的[一致性](https://baike.baidu.com/item/%E4%B8%80%E8%87%B4%E6%80%A7)(Consistency)：例如，完整性约束了a+b=10，一个事务改变了a，那么b也应该随之改变。

3）独立性(Isolation）：事务的独立性也有称作隔离性，是指两个以上的事务不会出现交错执行的状态。因为这样可能会导致数据不一致。

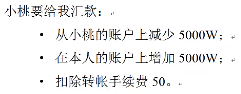
4）持久性(Durability）：[事务](https://baike.baidu.com/item/%E4%BA%8B%E5%8A%A1" \t "_blank)的[持久性](https://baike.baidu.com/item/%E6%8C%81%E4%B9%85%E6%80%A7)是指事务执行成功以后，该事务对数据库所作的更改便是持久的保存在数据库之中，不会无缘无故的[回滚](https://baike.baidu.com/item/%E5%9B%9E%E6%BB%9A)。

在Oracle中sqlplus是一个客户端，但是对于Oracle服务而言，每个sqlplus客户端都是独立的，都使用一个session。





真正使用了commit提交的时候才表示更新是可以正常完成的。所有的更新操作都可以被事物保护，

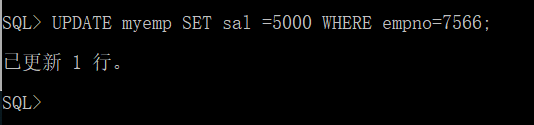


假如说现在第二步没有实现，小桃的账户应该恢复5000W,但是如果没有事务管理，那就表示，如果现在更新金额，就表示数据已将正常提交了，不管我的账户上是否增加有金额都减少。所以事物的好处是在一组操作全部更新完成，在进行提交处理，这样才可以保证数据的完整性。要成功就一起成功，失败就一起退回原点。

# 事物锁

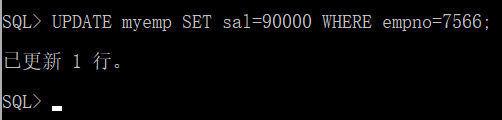
如果按照之前的讲解，每一个session都进行自己的事物处理，如果现在两个session同时操作同一条数据，该如何呢？以下通过实例说明存在的问题：

例：第一个session更新7566的雇员信息



此时的session并没有提交或回滚事物

例：第二个session更新7566雇员信息



此时发现更新操作并不能完成，因为两个session更新了同一条数据。此时就会出现锁的情况。即：在第一个session没有提交或回滚之前，第二个session要一直等待着更新完成。所以在事务处理过程之中，是存在行级锁定的概念，即：在提交或回滚之前，只能够有一个session操作数据，事物的隔离性。

虽然这种事物的处理很方便，但是这种锁定就很麻烦。平均每秒的在线人数有300人。25920000.假设这个站点的用户有3000W个用户，让你把所有用户的某一个字段更新为一个数据

\*\*\*直接发出UPDATE更新全部指令。目的是让3000W条数据一起完成更新：

假设现在每更新10条数据需要1秒时间，总的更新时间为：300000秒，833小时。

在这34天之内，所有的用户无法登录。

程序的评价标准有时间复杂度和空间复杂度。所以第一种做法是拿时间换空间，因为服务器有限，现实中很可怕。

\*\*\*按照这个空间换时间的思路，可以利用一个周期来完成。不直接更新所有的用户信息，而是在使用的时候才更新，虽然不是所有的用户都发生改变，但是大部分用户逐渐被修改。对于僵尸用户可以集中处理。

# oracle的锁

oracle的锁是自动的，不需要程序员的干预。

操作数据的时候，避免锁竞争。

# 注意事项

## 1. 避免锁竞争

多个会话对同一个表进行操作并不能提高效率。

## 2. 不要产生大事务

事务处理的数据量越大，对数据库的冲击就越大。

内存中 回滚表空间

1. 提交一次 事务开销

1000条 提交一次 刚刚好的大小

1千万 提交一次 受不了

insert 1百万。

update

delete

## 3. 不要产生长事务

事务越长，越容易产生锁竞争。

时间越长，持有锁的时间就越长。