[11.1 并发控制概述](#4856-1580897315233)

[11.2 封锁](#7090-1580895077779)

[11.3 活锁与死锁](#5663-1580896941609)

[11.3.1 活锁](#9317-1580897001823)

[11.3.2 死锁](#6073-1580897109990)

**11.1 并发控制概述**

多事务执行方法

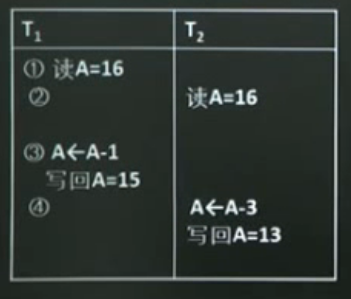
（1）事务串行执行

（2）交叉并发方式

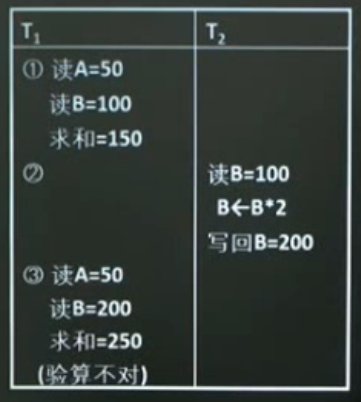
（3）同时并发方式

并发操作带来的数据不一致性

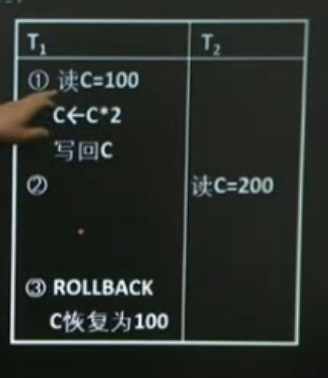
1.丢失修改



2.不可重复读

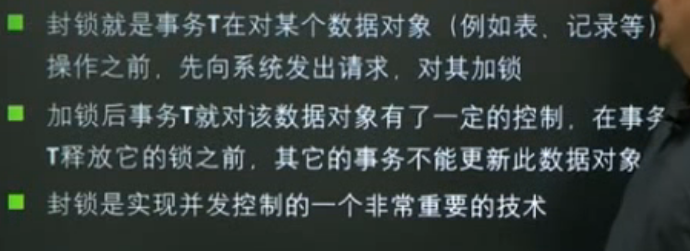


3.读“脏”数据



**11.2 封锁**

一、什么是封锁

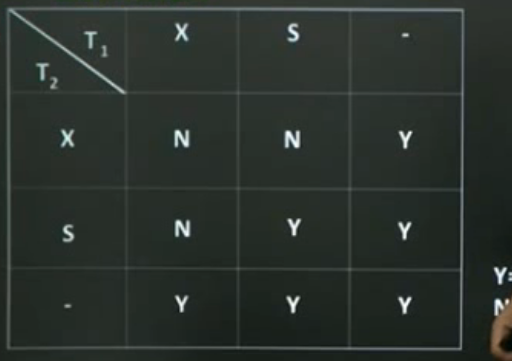


二、基本封锁类型

排它锁 写锁

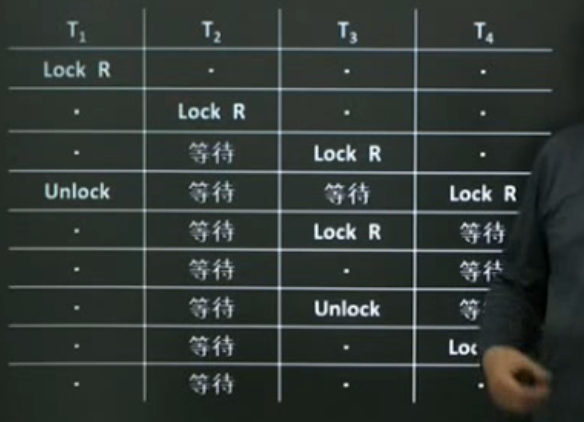
共享锁 读锁

三、锁的相容矩阵



**11.3 活锁与死锁**

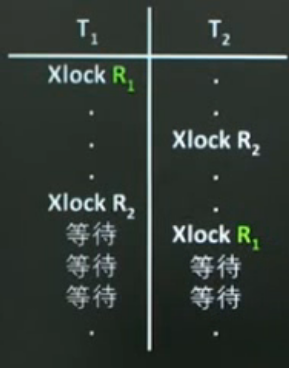
**11.3.1 活锁**



如何避免活锁：

采用先来先服务策略

**11.3.2 死锁**



1.死锁的预防

一次封锁法（难以一次确定需要封锁的对象）

顺序封锁法（维护成本高，难以实现）

2.死锁的诊断与解除

超时法（有可能误判）

等待图法

**11.4 并行调度的可串行性**

**11.4.1 可串行化调度**

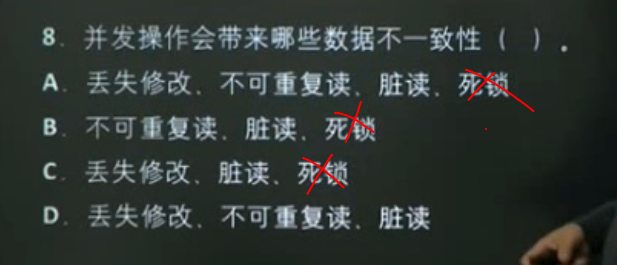
几个事务的并行执行是正确的，当且仅当其结果与按某一次序串行地执行它们的结果相同。

可串行性是并行事务正确性的唯一准则

**11.4.2 冲突可串行化调度**

冲突操作：不同的事务对同一数据的读写操作和写写操作

不同事务的冲突操作和同一事务的两个操作都是不能交换的



11.5 两段锁协议

封锁协议

一级封锁协议：事务T在修改数据R之前必须先对其加X锁，直到事务结束才释放

二级锁协议：一级plus事务T在读取数据R之间必须先对其加S锁，读完释放

三级封锁协议：一级plus事务T在读取数据R之前必须先对其加S锁，事务结束释放

两段锁协议：

1.在对任何数据进行读、写操作之前，事务首先要获得对该数据的封锁

2.在释放一个封锁后，事务不再获得任何其他封锁

两段锁含义：

事务分为两个阶段

第一阶段获得封锁：扩展阶段

第二阶段释放封锁：收缩阶段

并行执行的所有事务都遵守两段锁协议，则对这些事务的所有并行调度策略都是可串行化的

-->所以遵守两段锁协议的事务，其并行执行结果都是正确的

一次封锁法一定遵守两段锁协议

两段锁协议不一定是一次封锁法

遵守第三级封锁协议一定遵守两段锁协议

遵守两段锁协议不一定遵守三级封锁协议

**11.6 封锁的粒度**

**11.6.1 封锁粒度**

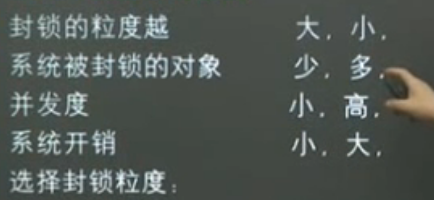
一、什么是封锁粒度

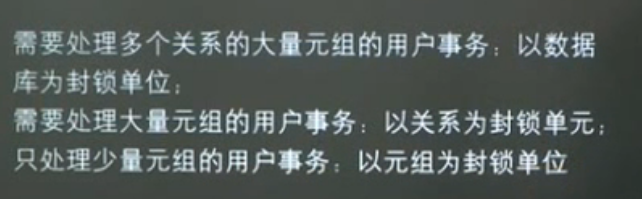
封锁的对象可以很大可以很小：对整个数据库加锁/对某个属性值加锁

封锁对象的大小称为封锁的粒度

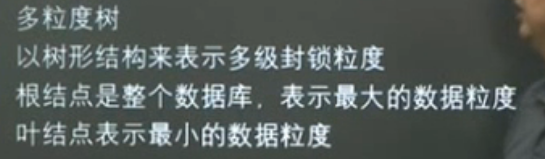
多粒度封锁：在一个系统中同时支持多种封锁粒度供不同的事务选择

二、选择封锁粒度的原则

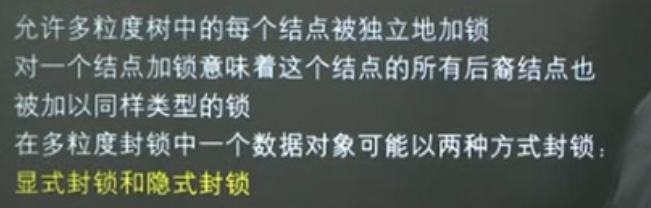




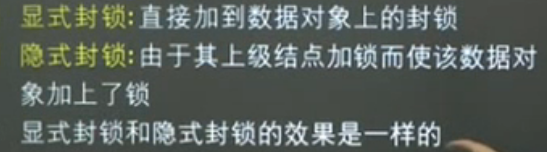
**11.6.2 多粒度封锁**



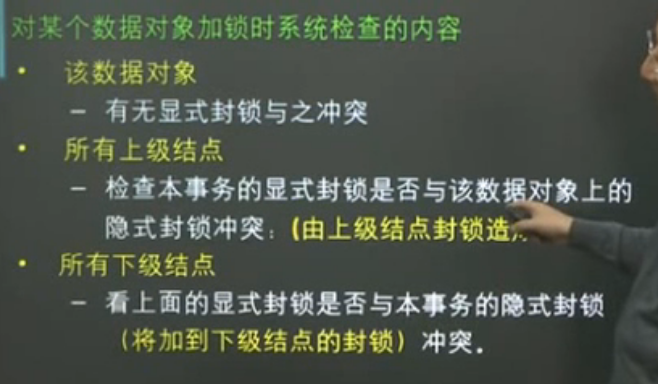
多粒度封锁协议：



显示封锁和隐式封锁

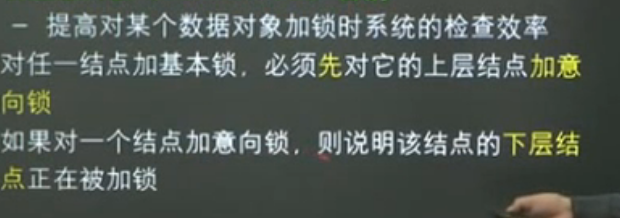


对某个数据对象加锁时系统检查的内容



**11.6.3 意向锁**

引进意向锁的目的



意向共享锁 IS

意向排它锁 IX

共享意向排它锁SIX

