11.5 两段锁协议

- 如何保证并发操作的调度是正确的
  - 为了保证并发调度的正确性，DBMS的并行控制机制必须提供一定的手段来保证调度是可串行化的。
  - 从理论上讲，在某一事务执行时禁止其它事务执行的调度策略一定是可串行化的调度，这也是最简单的调度策略，但实际上是不可行的：用户不能充分共享数据库资源。
  - 普遍采用的方法：两段锁(Two-Phase Locking，简称2PL)协议。
- 封锁协议
  - 运用封锁方法时，对数据对象加锁时需要约定一些规则：何时申请封锁、持锁时间、何时释放封锁等；
  - 约定不同的规则，就形成各种不同的封锁协议；

11.5 两段锁协议

“两段”锁的含义

- 事务对数据的加锁和解锁分为两个阶段
  - 第一阶段是获得封锁(只申请不释放)，也称为扩展阶段；
  - 第二阶段是释放封锁(只释放不申请)，也称为收缩阶段。

例：事务1的封锁序列：

Slock A ... Slock B ... Xlock C ... Unlock B ... Unlock A ... Unlock C；

事务2的封锁序列：

Slock A ... Unlock A ... Slock B ... Xlock C ... Unlock C ... Unlock B；

事务1遵守两段锁协议，而事务2不遵守两段协议。

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

(1)所有遵守两段锁协议的事务，并行执行结果一定是正确的；

(2)遵守两段锁协议是可串行化调度的充分条件，不是必要条件；

(3)可串行化的调度中，不一定所有事务都必须符合两段锁协议。

11.5 两段锁协议

T_1	T_2
Slock B	
读B=2	
Y=B	
Xlock A	
	Slock A
	等待
A=Y+1	等待
写回A=3	等待
Unlock B	等待
Unlock A	等待
	Slock A
	读A=3
	Y=A
	Xlock B
	B=Y+1
	写回B=4
	Unlock B
	Unlock A

(a) 遵守两段锁协议

T_1	T_2
Slock B	
读B=2	
Y=B	
Unlock B	
Xlock A	
	Slock A
	等待
A=Y+1	等待
写回A=3	等待
Unlock A	等待
	Slock A
	读A=3
	X=A
	Unlock A
	Xlock B
	B=X+1
	写回B=4
	Unlock B

(b) 不遵守两段锁协议

T_1	T_2
	Slock A
	读A=2
	X=A
	Unlock A
Slock B	
读B=2	Xlock B
Y=B	等待
Unlock B	Xlock B
	B=X+1
	写回B=3
	Unlock B
Xlock A	
A=Y+1	
写回A=3	
Unlock A	

(c) 不遵守两段锁协议

11.5 两段锁协议

- **两段锁协议与防止死锁的一次封锁法的比较：**
  - 一次封锁法要求每个事务必须一次将所有要使用的数据全部加锁，否则就不能继续执行，因此一次封锁法遵守两段锁协议；
  - 但是两段锁协议并不要求事务必须一次将所有要使用的数据全部加锁，因此遵守两段锁协议的事务可能发生死锁。

例：遵守两段锁协议的事务发生死锁

T_1	T_2
Slock B 读B=2	
	Slock A 读A=2
Xlock A 等待	
等待	Xlock B 等待