数据库系统概论
An Introduction to Database System

第十一章 并发控制

中国人民大学信息学院
陈红

第十一章 并发控制

11.5 两段锁协议

- 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

- 两段封锁协议 (Two-Phase Locking，简称 2PL) 是最常用的一种封锁协议，理论上证明使用两段封锁协议产生的是可串行化调度
- DBMS 普遍采用两段锁协议的方法实现并发调度的可串行性，从而保证调度的正确性

两段锁协议（续）

❖两段锁协议

指所有事务必须分两个阶段对数据项加锁和解锁

- 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
- 在释放一个封锁之后，事务不再申请和获得任何其他封锁

两段锁协议（续）

- “两段”锁的含义

  事务分为两个阶段

  - 第一阶段是获得封锁，也称为扩展阶段
    - 事务可以申请获得任何数据项上的任何类型的锁，但是不能释放任何锁
  - 第二阶段是释放封锁，也称为收缩阶段
    - 事务可以释放任何数据项上的任何类型的锁，但是不能再申请任何锁

两段锁协议（续）

例

事务 T*i* 遵守两段锁协议，其封锁序列是 ：

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

|←　　扩展阶段　　→ |　|←　　收缩阶段　　→ |

事务 T*j* 不遵守两段锁协议，其封锁序列是:

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

两段锁协议（续）

□□ T_1	□□ T_2
Slock(A)	
R(A)=260	
	Slock(C)
	R(C)=300
Xlock(A)	
W(A)=160	
	Xlock(C)
	W(C)=250
	Slock(A)
Slock(B)	□□
R(B)=1000	□□
Xlock(B)	□□
W(B)=1100	□□
Unlock(A)	□□
	R(A)=160
	Xlock(A)
Unlock(B)	
	W(A)=210
	Unlock(C)

遵守两段锁协议的可串行化调度

- 左图的调度是遵守两段锁协议的，因此一定是一个可串行化调度。
- 如何验证?

两段锁协议（续）

- 事务遵守两段锁协议是可串行化调度的充分条件，而不是必要条件。
- 若并发事务都遵守两段锁协议，则对这些事务的任何并发调度策略都是可串行化的
- 若并发事务的一个调度是可串行化的，不一定所有事务都符合两段锁协议

两段锁协议（续）

- 两段锁协议与防止死锁的一次封锁法
  - 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

两段锁协议（续）

[例] 遵守两段锁协议的事务发生死锁

T_1	T_2
Slock B	
R(B)=2	
	Slock A
	R(A)=2
Xlock A	
等待	Xlock A
等待	等待

遵守两段锁协议的事务可能发生死锁

第十一章 并发控制

封锁粒度

- 封锁对象的大小称为封锁粒度 (Granularity)
- 封锁的对象：逻辑单元，物理单元

  例：在关系数据库中，封锁对象：
  - 逻辑单元：属性值、属性值集合、元组、关系、索引项、整个索引、整个数据库等
  - 物理单元：页（数据页或索引页）、物理记录等

选择封锁粒度原则

- 封锁粒度与系统的并发度和并发控制的开销密切相关。
  - 封锁的粒度越大，数据库所能够封锁的数据单元就越少，并发度就越小，系统开销也越小；
  - 封锁的粒度越小，并发度较高，但系统开销也就越大

选择封锁粒度的原则（续）

例

- 若封锁粒度是数据页，事务 T1 需要修改元组 L1，则 T1 必须对包含 L1 的整个数据页 A 加锁。如果 T1 对 A 加锁后事务 T2 要修改 A 中元组 L2，则 T2 被迫等待，直到 T1 释放 A。
- 如果封锁粒度是元组，则 T1 和 T2 可以同时对 L1 和 L2 加锁，不需要互相等待，提高了系统的并行度。
- 又如，事务 T 需要读取整个表，若封锁粒度是元组，T 必须对表中的每一个元组加锁，开销极大

选择封锁粒度的原则（续）

- 多粒度封锁 (Multiple Granularity Locking)

  在一个系统中同时支持多种封锁粒度供不同的事务选择

- 选择封锁粒度

  同时考虑封锁开销和并发度两个因素，适当选择封锁粒度

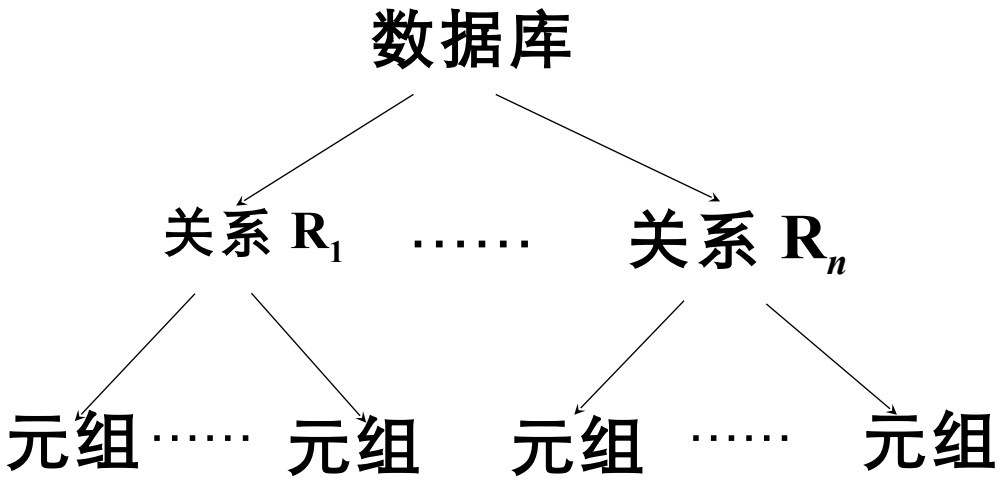
  - 需要处理多个关系的大量元组的用户事务：以数据库为封锁单位
  - 需要处理大量元组的用户事务：以关系为封锁单元
  - 只处理少量元组的用户事务：以元组为封锁单位

11.6.1 多粒度封锁

- 多粒度树
  - 以树形结构来表示多级封锁粒度
  - 根结点是整个数据库，表示最大的数据粒度
  - 叶结点表示最小的数据粒度

多粒度封锁（续）

例：三级粒度树。根结点为数据库，数据库的子结点为关系，关系的子结点为元组。



三级粒度树

多粒度封锁协议

- 允许多粒度树中的每个结点被独立地加锁
- 对一个结点加锁意味着这个结点的所有后裔结点也被加以同样类型的锁
- 在多粒度封锁中一个数据对象可能以两种方式封锁：显式封锁和隐式封锁

显式封锁和隐式封锁

- 显式封锁：直接加到数据对象上的封锁
- 隐式封锁：是该数据对象没有独立加锁，是由于其上级结点加锁而使该数据对象加上了锁
- 显式封锁和隐式封锁的效果是一样的

显式封锁和隐式封锁（续）

- 系统检查封锁冲突时
  - 要检查显式封锁
  - 还要检查隐式封锁
- 例如事务 T 要对关系 *R*1 加 X 锁
  - 系统必须搜索其上级结点数据库、关系 *R*1
  - 还要搜索 *R*1 的下级结点，即 *R*1 中的每一个元组
  - 如果其中某一个数据对象已经加了不相容锁，则 T 必须等待

显式封锁和隐式封锁（续）

- 对某个数据对象加锁，系统要检查
  - 该数据对象
    - 有无显式封锁与之冲突
  - 所有上级结点
    - 检查本事务的显式封锁是否与该数据对象上的隐式封锁冲突：（由上级结点已加的封锁造成的）
  - 所有下级结点
    - 看上面的显式封锁是否与本事务的隐式封锁（将加到下级结点的封锁）冲突

11.6.2 意向锁

- 引进意向锁（intention lock）目的
  - 提高对某个数据对象加锁时系统的检查效率

意向锁（续）

- 如果对一个结点加意向锁，则说明该结点的下层结点正在被加锁
- 对任一结点加基本锁，必须先对它的上层结点加意向锁
- 例如，对任一元组加锁时，必须先对它所在的数据库和关系加意向锁

常用意向锁

- 意向共享锁 (Intent Share Lock， 简称 IS 锁)
- 意向排它锁 (Intent Exclusive Lock， 简称 IX 锁)
- 共享意向排它锁 (Share Intent Exclusive Lock， 简称 SIX 锁)

意向锁（续）

IS 锁

- 如果对一个数据对象加 IS 锁，表示它的后裔结点拟（意向）加 S 锁。

  例如：事务 T1 要对 *R*1 中某个元组加 S 锁，则要首先对关系 *R*1 和数据库加 IS 锁

意向锁（续）

IX 锁

- 如果对一个数据对象加 IX 锁，表示它的后裔结点拟（意向）加 X 锁。

例如：事务 T1 要对 *R*1 中某个元组加 X 锁，则要首先对关 系 *R*1 和数据库加 IX 锁

意向锁（续）

SIX 锁

- 如果对一个数据对象加 SIX 锁，表示对它加 S 锁，再加 IX 锁，即 SIX = S + IX 。

例：对某个表加 SIX 锁，则表示该事务要读整个表（所以要对该表加 S 锁），同时会更新个别元组（所以要对该表加 IX 锁）。

意向锁（续）

意向锁的相容矩阵

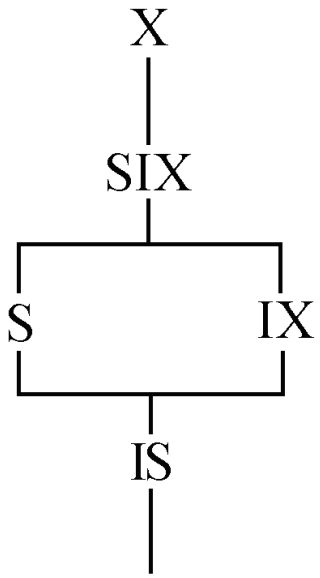
T_1 \ T_2	S	X	IS	IX	SIX	–
S	Y	N	Y	N	N	Y
X	N	N	N	N	N	Y
IS	Y	N	Y	Y	Y	Y
IX	N	N	Y	Y	N	Y
SIX	N	N	Y	N	N	Y
–	Y	Y	Y	Y	Y	Y

Y=Yes，表示相容的请求　　N=No，表示不相容的请求

(a) 数据锁的相容矩阵

意向锁（续）

- 锁的强度
  - 锁的强度是指它对其他锁的排斥程度
  - 一个事务在申请封锁时以强锁代替弱锁是安全的，反之则不然



(b) 锁的强度的偏序关系

意向锁（续）

- 具有意向锁的多粒度封锁方法
  - 申请封锁时应该按自上而下的次序进行
  - 释放封锁时则应该按自下而上的次序进行

例如：事务 T1 要对关系 *R*1 加 S 锁

- 要首先对数据库加 IS 锁
- 检查数据库和 *R*1 是否已加了不相容的锁 (X 或 IX)
- 不再需要搜索和检查 *R*1 中的元组是否加了不相容的锁 (X 锁)

意向锁（续）

- 具有意向锁的多粒度封锁方法
  - 提高了系统的并发度
  - 减少了加锁和解锁的开销
  - 在实际的数据库管理系统产品中得到广泛应用

第十一章 并发控制

11.7 小结

- 数据库的并发控制以事务为单位
- 数据库的并发控制通常使用封锁机制
  - 基本封锁
  - 多粒度封锁
- 活锁和死锁

小结（续）

- 并发事务调度的正确性
  - 可串行性
    - 并发操作的正确性则通常由两段锁协议来保证。
    - 两段锁协议是可串行化调度的充分条件，但不是必要条件
  - 冲突可串行性


DEM DEUTSCHEN VOLKE
2008/10/18