

第十一章 并发控制

- 11.1 并发控制概述
- 11.2 封锁
- 11.3 封锁协议
- 11.4 活锁和死锁
- 11.5 并发调度的可串行性
- 11.6 两段锁协议
- 11.7 封锁的粒度
- *11.8 其他并发控制机制
- 11.9 小结



11.5 并发调度的可串行性

- ❖ 数据库管理系统对并发事务不同的调度可能会产生不同的结果
- ❖ 串行调度是正确的
- ❖ 执行结果等价于串行调度的调度也是正确的，称为可串行化调度



11.5 并发调度的可串行性

11.5.1 可串行化调度

11.5.2 冲突可串行化调度



11.5.1 可串行化调度

❖ 可串行化(**Serializable**)调度

- 多个事务的并发执行是正确的，当且仅当其结果与按某一次序串行地执行这些事务时的结果相同

❖ 可串行性(**Serializability**)

- 是并发事务正确调度的准则
- 一个给定的并发调度，当且仅当它是可串行化的，才认为是正确调度



可串行化调度（续）

[例] 现在有两个事务，分别包含下列操作：

■ **事务T1**：读B； $A=B+1$ ；写回A

■ **事务T2**：读A； $B=A+1$ ；写回B

现给出对这两个事务不同的调度策略



串行调度,正确的调度

T_1	T_2
<u>Slock B</u>	
<u>$Y=R(B)=2$</u>	
<u>Unlock B</u>	
<u>Xlock A</u>	
<u>$A=Y+1=3$</u>	
<u>W(A)</u>	
<u>Unlock A</u>	
	<u>Slock A</u>
	<u>$X=R(A)=3$</u>
	<u>Unlock A</u>
	<u>Xlock B</u>
	<u>$B=X+1=4$</u>
	<u>W(B)</u>
	<u>Unlock B</u>

- 假设A、B的初值均为2。
- 按 $T_1 \rightarrow T_2$ 次序执行结果为**A=3, B=4**
- 串行调度策略,正确的调度



串行调度,正确的调度

T_1	T_2
	<u>Slock A</u>
	<u>$X=R(A)=2$</u>
	<u>Unlock A</u>
	<u>Xlock B</u>
	<u>$B=X+1=3$</u>
	<u>$W(B)$</u>
	<u>Unlock B</u>
<u>Slock B</u>	
<u>$Y=R(B)=3$</u>	
<u>Unlock B</u>	
<u>Xlock A</u>	
<u>$A=Y+1=4$</u>	
<u>$W(A)$</u>	
<u>Unlock A</u>	

- 假设A、B的初值均为2。
- $T_2 \rightarrow T_1$ 次序执行结果为
 $B=3, A=4$
- 串行调度策略,正确的调度



不可串行化调度，错误的调度

T_1	T_2
<u>Slock B</u>	
<u>$Y=R(B)=2$</u>	
	<u>Slock A</u>
	<u>$X=R(A)=2$</u>
<u>Unlock B</u>	
	<u>Unlock A</u>
<u>Xlock A</u>	
<u>$A=Y+1=3$</u>	
<u>$W(A)$</u>	
	<u>Xlock B</u>
	<u>$B=X+1=3$</u>
	<u>$W(B)$</u>
<u>Unlock A</u>	
	<u>Unlock B</u>

- 执行结果 **A=3, B=3**,
与(a)、(b)的结果都不同
- 是错误的调度



可串行化调度，正确的调度

T_1	T_2
<u>Slock B</u>	
<u>$Y=R(B)=2$</u>	
<u>Unlock B</u>	
<u>Xlock A</u>	
	<u>Slock A</u>
<u>$A=Y+1=3$</u>	<u>等待</u>
<u>$W(A)$</u>	<u>等待</u>
<u>Unlock A</u>	<u>等待</u>
	<u>$X=R(A)=3$</u>
	<u>Unlock A</u>
	<u>Xlock B</u>
	<u>$B=X+1=4$</u>
	<u>$W(B)$</u>
	<u>Unlock B</u>

- 执行结果 **A=3, B=4**,
与第一种串行调度的
执行结果相同
- 是正确的调度



四种调度

T ₁	T ₂
Slock B	
Y=R(B)=2	
Unlock B	
Xlock A	
A=Y+1=3	
W(A)	
Unlock A	
	Slock A
	X=R(A)=3
	Unlock A
	Xlock B
	B=X+1=4
	W(B)
	Unlock B

串行调度

T ₁	T ₂
	Slock A
	X=R(A)=2
	Unlock A
	Xlock B
	B=X+1=3
	W(B)
	Unlock B
Slock B	
Y=R(B)=3	
Unlock B	
Xlock A	
A=Y+1=4	
W(A)	
Unlock A	

串行调度

T ₁	T ₂
Slock B	
Y=R(B)=2	
	Slock A
	X=R(A)=2
Unlock B	
	Unlock A
Xlock A	
A=Y+1=3	
W(A)	
	Xlock B
	B=X+1=3
	W(B)
Unlock A	
	Unlock B

不可串行化的调度

T ₁	T ₂
Slock B	
Y=R(B)=2	
Unlock B	
Xlock A	
	Slock A
A=Y+1=3	等待
W(A)	等待
Unlock A	等待
	X=R(A)=3
	Unlock A
	Xlock B
	B=X+1=4
	W(B)
	Unlock B

可串行化的调度

11.5 并发调度的可串行性

11.5.1 可串行化调度

11.5.2 冲突可串行化调度



11.5.2 冲突可串行化调度

❖ 冲突可串行化

■ 一个比可串行化更严格的条件

❖ 冲突操作：是指不同的事务对同一数据的读写操作和写写操作：

$R_i(x)$ 与 $W_j(x)$ /*事务 T_i 读 x , T_j 写 x , 其中 $i \neq j$ */

$W_i(x)$ 与 $W_j(x)$ /*事务 T_i 写 x , T_j 写 x , 其中 $i \neq j$ */

涉及同一个数据库元素, 并且至少有一个是写操作



冲突

❖ 不冲突操作

- $r_i(X); r_j(Y)$ 读
- $r_i(X); w_j(Y)$, $X \neq Y$
- $w_i(X); r_j(Y)$, $X \neq Y$
- $w_i(X); w_j(Y)$, $X \neq Y$



冲突

❖ 不能交换（**Swap**）的动作：

■ 同一事务的两个操作

■ 不同事务的冲突操作

• $R_i(x)$ 与 $W_j(x)$

• $W_i(x)$ 与 $W_j(x)$



冲突可串行化

- ❖ 一个调度 S_c 在保证冲突操作的次序不变的情况下，通过交换两个事务不冲突操作的次序得到另一个调度 S_c' ，如果 S_c' 是串行的，称调度 S_c 是冲突可串行化的调度
- ❖ 若一个调度是冲突可串行化，则一定是可串行化的调度



冲突可串行化（续）

[例] 今有3个事务的一个调度

r3(B) r1(A) w3(B) r2(B) r2(A) w2(B) r1(B) w1(A)

判断该调度是否是冲突可串行化的调度。

Sc1 = r3(B) **r1(A)** **w3(B)** r2(B) r2(A) w2(B) r1(B) w1(A)



冲突可串行化（续）

[例] 今有3个事务的一个调度

r3(B) r1(A) w3(B) r2(B) r2(A) w2(B) r1(B) w1(A)

判断该调度是否是冲突可串行化的调度。

Sc1 = r3(B) r1(A) w3(B) r2(B) r2(A) w2(B) r1(B) w1(A)

r3(B) w3(B) r1(A) r2(B) r2(A) w2(B) r1(B) w1(A)



冲突可串行化（续）

[例] 今有3个事务的一个调度

r3(B) r1(A) w3(B) r2(B) r2(A) w2(B) r1(B) w1(A)

判断该调度是否是冲突可串行化的调度。

Sc1 = r3(B) r1(A) w3(B) r2(B) r2(A) w2(B) r1(B) w1(A)

r3(B) w3(B) r1(A) r2(B) r2(A) w2(B) r1(B) w1(A)



冲突可串行化（续）

[例] 今有3个事务的一个调度

r3(B) r1(A) w3(B) r2(B) r2(A) w2(B) r1(B) w1(A)

判断该调度是否是冲突可串行化的调度。

Sc1 = r3(B) r1(A) w3(B) r2(B) r2(A) w2(B) r1(B) w1(A)

r3(B) w3(B) r1(A) r2(B) r2(A) w2(B) r1(B) w1(A)

r3(B) w3(B) r2(B) r2(A) w2(B) r1(A) r1(B) w1(A)



冲突可串行化（续）

[例] 今有3个事务的一个调度

r3(B) r1(A) w3(B) r2(B) r2(A) w2(B) r1(B) w1(A)

判断该调度是否是冲突可串行化的调度。

Sc1 = r3(B) r1(A) w3(B) r2(B) r2(A) w2(B) r1(B) w1(A)

r3(B) w3(B) r1(A) r2(B) r2(A) w2(B) r1(B) w1(A)

r3(B) w3(B) r2(B) r2(A) w2(B) r1(A) r1(B) w1(A)

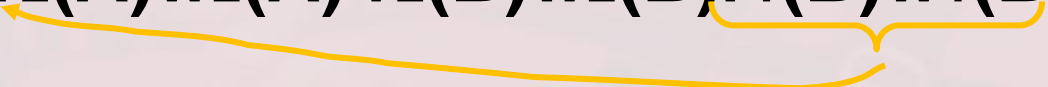
Sc2 = r3(B) w3(B) r2(B) r2(A) w2(B) r1(A) r1(B) w1(A)

所以 **Sc1**是冲突可串行化的调度。

冲突可串行化（续）

例：

$S_d = r_1(A)w_1(A)r_2(A)w_2(A) r_2(B)w_2(B)r_1(B)w_1(B)$



冲突可串行化（续）

例：

$S_d = r_1(A)w_1(A)r_2(A)w_2(A) r_2(B)w_2(B)r_1(B)w_1(B)$



冲突可串行化（续）

例：

$S_d = r_1(A)w_1(A)r_2(A)w_2(A) r_2(B)w_2(B)r_1(B)w_1(B)$



冲突可串行化（续）

例：

$S_d = r_1(A)w_1(A)r_2(A)w_2(A) r_2(B)w_2(B)r_1(B)w_1(B)$

- 不能通过无冲突交换将**S_d**变换为串行调度
- **S_d**不是冲突可串行化的调度



冲突可串行化调度

- ❖ 冲突可串行化调度是可串行化调度的充分条件，不是必要条件。还有不满足冲突可串行化条件的可串行化调度。

[例11.4]有3个事务

$$T_1=W_1(Y)W_1(X), \quad T_2=W_2(Y)W_2(X), \quad T_3=W_3(X)$$

- 调度 $L_1=\underline{W_1(Y)W_1(X)}\underline{W_2(Y)W_2(X)}\underline{W_3(X)}$ 是一个串行调度。
- 调度 $L_2=W_1(Y)W_2(Y)W_2(X)W_1(X)W_3(X)$ 不满足冲突可串行化。



冲突可串行化调度

- ❖ 冲突可串行化调度是可串行化调度的充分条件，不是必要条件。还有不满足冲突可串行化条件的可串行化调度。

[例11.4]有3个事务

$$T_1=W_1(Y)W_1(X), T_2=W_2(Y)W_2(X), T_3=W_3(X)$$

- 调度 $L_1=W_1(Y)W_1(X)W_2(Y)W_2(X)W_3(X)$ 是一个串行调度。
- 调度 $L_2=W_1(Y)W_2(Y)W_2(X)W_1(X)W_3(X)$ 不满足冲突可串行化。

但是调度 L_2 是可串行化的，因为 L_2 执行的结果与调度 L_1 相同， Y 的值都等于 T_2 的值， X 的值都等于 T_3 的值



小结

❖ 可串行化调度

❖ 冲突可串行化调度

❖ 可串行化调度与冲突可串行化调度之间的关系



思考题

❖ 判定一个调度是否是冲突可串行化的，我们课上学习了一种方法，就是对无冲突操作进行交换，看看能否将其转换为串行调度。当并发事务数目比较多时，这种方法的效率可能会存在问题。是否还有其他更高效的判定方法呢？



