Projeto de Banco de Dados

Controle de Concorrência

PROF. DR. THIAGO ELIAS

Controle de Concorrência

- Importância da propriedade ISOLAMENTO.
- Inicialmente, estudaremos os Protocolos Baseados em Bloqueios.
  - O princípio desses protocolos é que o acesso deve ser feito de forma mutuamente exclusiva.
- Dentre os diversos tipos de bloqueio, inicialmente destacam-se:
  - Compartilhado (S): Se Ti obteve um bloqueio compartilhado sobre o item de dado Q, então ela só pode ler o valor de Q.
  - Exclusivo (X): ler e escreve.

Controle de Concorrência

- Lock-S(Q) – Solicitação de bloqueio compartilhado ao item de dado Q.
- Lock-X(Q) – Solicitação de bloqueio exclusivo ao item de dado Q.
- Unlock(Q) – desbloqueia o item de dado Q.

	S	X
S	Verdadeiro	Falso
X	Falso	Falso

Matriz de compatibilidade de bloqueios

Protocolos Baseados em Bloqueio

- Consideramos T1 e T2:

```
T1: lock-X(B);
    read(B);
    B := B – 50;
    write(B);
    unlock(B);
    lock-X(A);
    read(A);
    A := A + 50;
    write(A);
    unlock(A)
```

```
T2: lock-S(A);
    read(A);
    unlock(A);
    lock-S(B);
    read(B);
    unlock(A);
    display(A + B)
```

Obs: Independentemente da ordem de execução, T2 tem que mostrar sempre o mesmo valor

- As transações desbloqueiam os respectivos itens de dado tão logo concluem a sua manipulação.
  - A princípio, isso seria vantajoso pois aumentaria a possibilidade de diferentes escalas de execução.

Continua.....

Protocolos Baseados em Bloqueio

Dentre as diversas escalas de execução, esta poderia ser formada

Qual o problema dela?

T2 apresenta um valor INCORRETO

T_1	T_2	Gerenciador de controle de concorrência
lock-X(B)		
		grant-X(B, T_1)
read(B)		
$B := B - 50$		
write(B)		
unlock(B)		
	lock-S(A)	
		grant-S(A, T_2)
	read(A)	
	unlock(A)	
	lock-S(B)	
		grant-S(B, T_2)
	read(B)	
	unlock(B)	
	display($A + B$)	
lock-X(A)		
		grant-X(A, T_2)
read(A)		
$A := A - 50$		
write(A)		
unlock(A)		

Protocolos Baseados em Bloqueio

- Solução:
  - Consideremos T3 e T4 que são idênticas a T1 e T2, respectivamente, porém os desbloqueios acontecem no final da transação.
  - Nesse caso, em todas as escalas que poderão vir a surgir, T4 sempre apresentará o valor correto.
  - Então o problema foi resolvido?

```
T3: lock-X(B);
    read(B);
    B := B − 50;
    write(B);
    lock-X(A);
    read(A);
    A := A + 50;
    write(A);
    unlock(B)
    unlock(A)
```

```
T4: lock-S(A);
    read(A);
    lock-S(B);
    read(B);
    display(A + B);
    unlock(A);
    unlock(B)
```

Continua.....

Protocolos Baseados em Bloqueio

- Dentre as diversas escala de execução, esta poderá surgir:

T_3	T_4
lock-X(B)	
read(B)	
$B := B - 50$	
write(B)	
	lock-S(A)
	read(A)
	lock-S(B)
lock-X(A)	

DEADLOCK (impasse)

Protocolos Baseados em Bloqueio

- Solução:
  - Desfazer a transação a fim de que seus itens de dados sejam liberados.
- O problema de deadlock é inerente aos protocolos de bloqueio.
- Porém, ele é preferível ao estado de inconsistência!!

Protocolos Baseados em Bloqueio

- Basta o SGBD obedecer a matriz de compatibilidade de bloqueios?
  - Problema da Inanição (Ti esperar indefinidamente por um bloqueio que nunca chegará)
  - Solução: Um bloqueio só será concedido se:
    - Não haja bloqueio conflitante
    - Não haja outra transação aguardando o desbloqueio do Item.

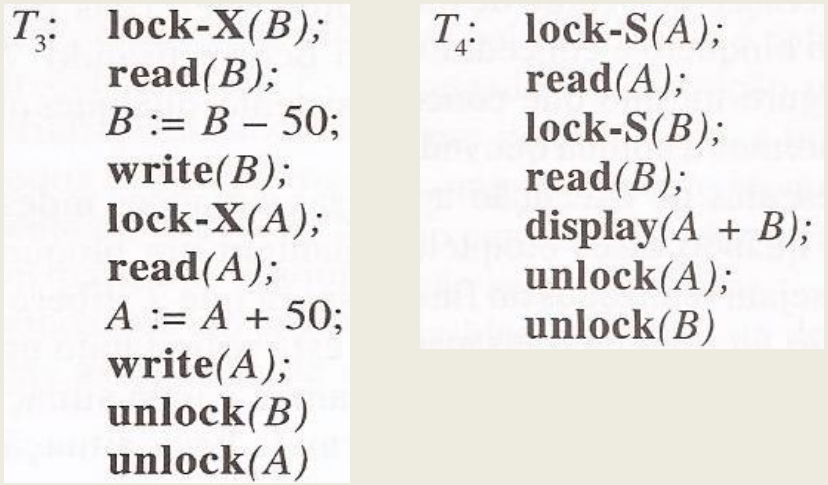
Protocolo de Bloqueio em Duas Fases

- As solicitações de bloqueio e desbloqueio são feitas em duas fases:
  - Fase de expansão: apenas solicita bloqueios.
  - Fase de encolhimento: apenas libera bloqueios.

- Quando se inicia, a transação está na fase de expansão. Tão logo libere o primeiro bloqueio, entra na fase de encolhimento.

Protocolo de Bloqueio em Duas Fases

- Exemplo:



  - As mesmas transações T3 e T4 apresentadas anteriormente já obedeciam o protocolo de duas fases.
  - Note que as instruções de desbloqueio não precisam aparecer no final da transação. Em T3, Unlock(B) poderia ser logo após Lock-X(A).

Protocolo de Bloqueio em Duas Fases

- Resumindo... O que importa para a garantia da serialização é o *ponto de bloqueio* de cada transação (*final da fase de expansão*).
- O protocolo de bloqueio em duas fases NÃO elimina a possibilidade de *deadlocks*.
  - Exemplo:

T_3	T_4
lock-X(B)	
read(B)	
$B := B - 50$	
write(B)	
	lock-S(A)
	read(A)
	lock-S(B)
lock-X(A)	

Protocolo de Bloqueio em Duas Fases

- Com o protocolo em duas fases, podem surgir escalas com rollback em cascata (falha de T5 após Read(A) de T7). Ex:

T_5	T_6	T_7
lock-X(A)		
read(A)		
lock-S(B)		
read(B)		
write(A)		
unlock(A)		
	lock-X(A)	
	read(A)	
	write(A)	
	unlock(A)	
		lock-S(A)
		read(A)

A solução para este problema seria o PROTOCOLO EM DUAS FASES SEVERO. Nele, todos os **bloqueios exclusivos** tomados por uma transação são mantidos até que a transação seja efetivada.
Tb há o RIGOROSO, onde **TODOS** os bloqueios são liberados apenas após a efetivação da transação.
Estes são os mais utilizados!!!!