

Controle de Concorrência

Banco de Dados: Teoria e Prática

André Santanchè

Instituto de Computação - UNICAMP

Outubro 2012

Controle de Concorrência

- Propósitos

- Garantir a propriedade de isolamento em transações concorrentes
- Preservar a consistência do banco pela preservação da consistência na execução das transações
- Resolver conflitos leitura-gravação e gravação-gravação

Bloqueio

- Um bloqueio (*lock*) é uma variável associada a um item de dados
 - Descreve a condição do item em relação às possíveis operações que podem ser aplicadas a ele
- Geralmente há um bloqueio para cada item no BD
- Tipos analisados:
 - bloqueio binário
 - bloqueios compartilhados/exclusivos

Bloqueio Binário

- Dois estados:
 - bloqueado (*locked*) - o item não pode ser acessado quando solicitado
 - desbloqueado (*unlocked*) - o item pode ser acessado quando solicitado
- Operações atômicas de bloqueio binário:
 - `lock_item(X)`
 - `unlock_item(X)`

Bloqueio Binário

Operação lock_item(X)

```
B: if (LOCK(X) = 0) then
    LOCK(X) ← 1
else {
    wait (until LOCK(X)=0 and
        <the lock manager wakes up
        the transaction>)
    goto B
}
```

(Elmasri, 2010)

Bloqueio Binário

Operação unlock_item(X)

LOCK(X) ← 0

```
if <any transactions are waiting> then  
    wake up one of the waiting the  
    transactions;
```

(Elmasri, 2010)

Bloqueio Binário

Controle de Concorrência

- Para cada transação
 - lock(X)
 - antes de ler(X)/gravar(X)
 - se X ainda não tiver lock
 - unlock(X)
 - depois de todas as operações ler(X)/gravar(X)
 - apenas se tiver o lock de X

Limitações?

Limitações?

- Duas ações que apenas leem registros precisam se bloquear mutuamente?

Limitações?

- Duas ações que apenas leem registros precisam se bloquear mutuamente?
 - não

Bloqueio Compartilhado/Exclusivo

- Bloqueio compartilhado (*shared lock*)
 - utilizado para leitura (*read lock*)
 - mais de uma transação pode empregá-lo
 - impede a requisição de um bloqueio exclusivo
- Bloqueio exclusivo (*exclusive lock*)
 - utilizado para gravação (*write lock*)
 - somente uma transação pode solicitá-lo

■ Matriz de compatibilidade:

	shared	exclusive
shared	S	N
exclusive	N	N

Gerenciador de Bloqueio

- Gerencia o bloqueio de itens
 - Mantém uma tabela de controle de bloqueio
 - Exemplo:
 - Controle(Transação, Item, Modo, Próximo_item)
- (Elmasri, 2007)

Bloqueio Compartilhado/Exclusivo

Operação rlock(X)

```
B: if LOCK(X) = "unlocked" then {  
    LOCK(X) ← "read-locked"  
    no_of_reads(X) ← 1  
}  
else if LOCK(X) = "read-locked" then  
    no_of_reads(X) ++  
else {  
    wait (until LOCK(X) = "unlocked" and  
        <the lock manager wakes up the  
        transaction>)  
    goto B  
}
```

(Elmasri, 2010)

Bloqueio Compartilhado/Exclusivo

Operação wlock(X)

```
B: if LOCK (X) = "unlocked" then
    LOCK (X) ← "write-locked"
else {
    wait (until LOCK(X) = "unlocked" and
        <the lock manager wakes up the
        transaction>)
    goto B
}
```

(Elmasri, 2010)

Bloqueio Compartilhado/Exclusivo

Operação unlock(X)

```
if LOCK(X) = "write-locked" then {  
    LOCK(X) ← "unlocked"  
    wakeup up one of the transactions,  
    if any  
}  
else if LOCK(X) = "read-locked" then {  
    no_of_reads(X) --  
    if no_of_reads(X) = 0 then {  
        LOCK(X) ← "unlocked";  
        wakeup up one of the transactions,  
        if any  
    }  
}
```

(Elmasri, 2010)

Bloqueio Compartilhado/Exclusivo

Controle de Concorrência

- Para cada transação
 - rlock(X)
 - antes de read(X)
 - se ainda não tiver rlock ou wlock
 - wlock(X)
 - antes de read(X) com intenção de write(X)
 - antes de write(X)
 - se ainda não tiver wlock
 - unlock(X)
 - depois de todas as operações ler(X)/gravar(X)
 - apenas se tiver o lock de X

Bloqueio Compartilhado/Exclusivo

Upgrade e Downgrade

- Lock Upgrade
 - $rlock(x) \rightarrow wlock(x)$
 - condição: não há outro $rlock$ em X
- Lock Downgrade
 - $wlock(x) \rightarrow rlock(x)$

Garantindo a Serialização

Transação 1: Transferência

T1

ler (X)

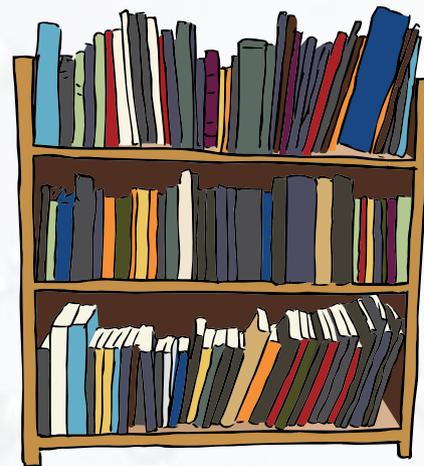
$X = X - N$

gravar (X)

ler (Y)

$Y = Y + N$

gravar (Y)



Prateleira X

transferência



Prateleira Y

Garantindo a Serialização

Transação 2: Aquisição

T2

ler (X)

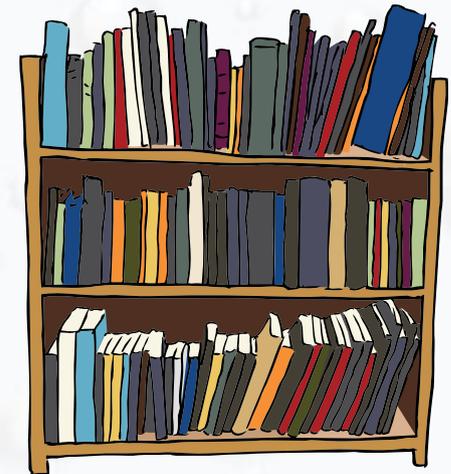
$X = X + M$

gravar (X)



aquisição

M livros



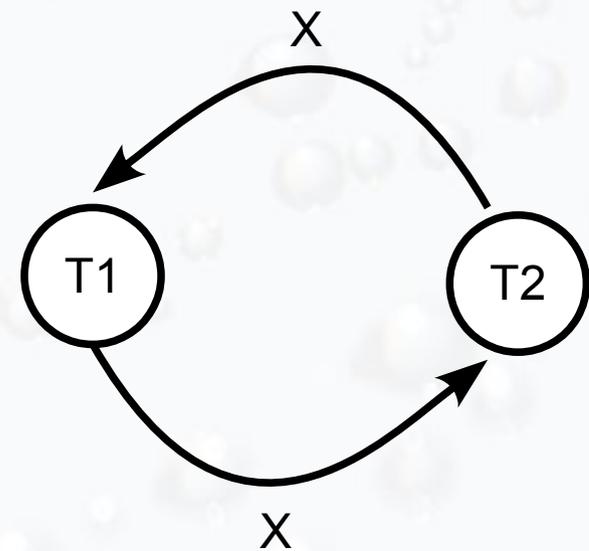
Prateleira X

Garantindo a Serialização Plano de Execução

T1	T2
ler (X) X = X - N gravar (X) ler (Y) Y = Y + N gravar (Y)	ler (X) X = X + M gravar (X)

Não Serializável

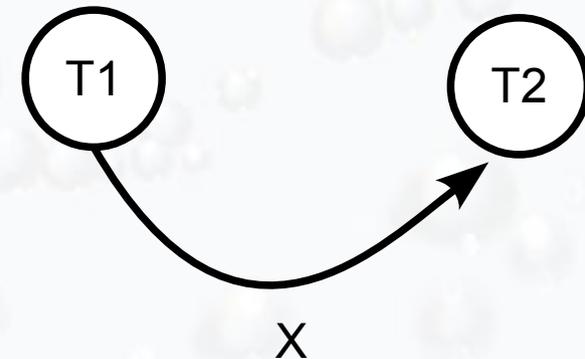
T1	T2
ler (X) $X = X - N$	
gravar (X) ler (Y)	ler (X) $X = X + M$
$Y = Y + N$ gravar (Y)	gravar (X)



Plano Não Serializável

Serializável

T1	T2
ler (X) $X = X - N$ gravar (X)	
	ler (X) $X = X + M$ gravar (X)
ler (Y) $Y = Y + N$ gravar (Y)	



Plano Serializável

Protocolo Two-phase (2PL) Locking

- Garante serialização
- Protocolo de bloqueio de duas fases
 - **Fase de crescimento:** Transação pode obter bloqueios, mas não pode liberar
 - **Fase de encolhimento:** Transação pode liberar bloqueios, mas não pode obter

2PL Diagrama

Transações com Locks

T1	T2
wlock (X) ler (X) $X = X - N$ gravar (X) unlock (X)	wlock (X) ler (X) $X = X + M$ gravar (X) unlock (X)
wlock (Y) ler (Y) $Y = Y + N$ gravar (Y) unlock (Y)	

T1

T2

wlock (X)

ler (X)

$X = X - N$

gravar (X)

wlock (Y)

unlock (X)

ler (Y)

$Y = Y + N$

gravar (Y)

unlock (Y)

wlock (X)

ler (X)

$X = X + M$

gravar (X)

unlock (X)

Plano 2PL

Garantindo a Serialização

Exemplo

- Cálculo de livros necessários de acordo com a média

T1
ler (X)
$Q = \text{Media} - X$
gravar (Q)



Garantindo a Serialização

Exemplo

- Com locks

T1
<code>rlock (X)</code>
<code>ler (X)</code>
<code>unlock (X)</code>
$Q = \text{Media} - X$
<code>wlock (Q)</code>
<code>gravar (Q)</code>
<code>unlock (Q)</code>

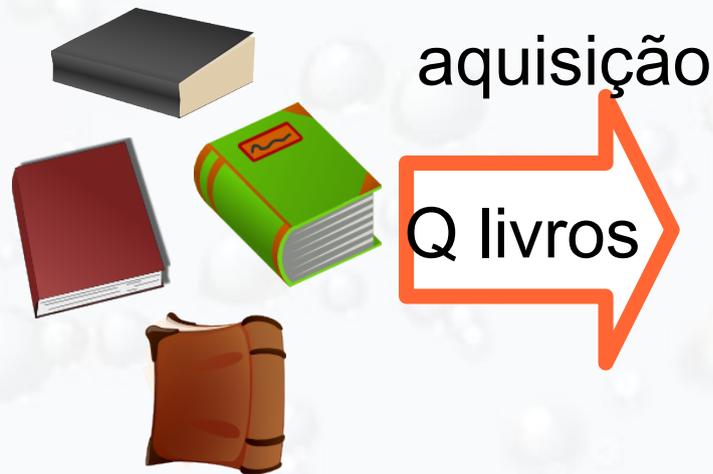


Garantindo a Serialização

Exemplo

- Aquisição de Q livros

T2
ler (Q)
ler (X)
$X = X + Q$
gravar (X)



Prateleira X

Garantindo a Serialização

Exemplo

- Com locks

T2

```
rlock (Q)  
ler (Q)  
unlock (Q)
```

```
wlock (X)  
ler (X)  
 $X = X + Q$   
gravar (X)  
unlock (X)
```



aquisição

Q livros

Prateleira X

Garantindo a Serialização

Exemplo

- Serializável?

T1	T2
	rlock (Q) ler (Q) unlock (Q)
rlock (X) ler (X) unlock (X)	
$Q = Media - X$	
wlock (Q) gravar (Q) unlock (Q)	
	wlock (X) ler (X) $X = X + Q$ gravar (X) unlock (X)

Garantindo a Serialização

Exemplo

- Não serializável
- Pode haver problemas no 2PL?

T1	T2
	rlock (Q) ler (Q) unlock (Q)
rlock (X) ler (X) unlock (X)	
$Q = \text{Media} - X$	
wlock (Q) gravar (Q) unlock (Q)	
	wlock (X) ler (X) $X = X + Q$ gravar (X) unlock (X)

Deadlock

- Impasse
- “Ciclo de transações esperando mutuamente pela liberação de locks.”

Tradução livre (Ramakrishnan, 2003)

Grafo de Espera

Tratando Deadlocks

- Prevenção de deadlock
- Detecção de deadlock

2PL

Conservador ou Estático

- Bloqueia todos os itens a ser lidos/gravados antes de iniciar a transação
- Livre de deadlock
- Exige pré-declaração (leituras/gravações) no início da transação

2PL Conservador ou Estático Diagrama

2PL

Estrito e Rigoroso

- 2PL Estrito
 - Não libera wlocks até o commit ou abort
 - Garante schedule estrito
 - T só lê e/ou grava valores que foram alterados por transações que já realizaram commit
- 2PL Rigoroso
 - Não libera rlocks/wlocks até o commit ou abort
 - Mais fácil de implementar que o Estrito

2PL Estrito e Rigoroso Diagrama

Prevenção de Deadlock

Rótulo de Tempo

- Transações com timestamps
- Considere T_i quer lock(X) e T_j tem lock(X)
- $\text{stamp}(T_i) < \text{stamp}(T_j)$ (maior prioridade)
- Políticas:
 - Wait-die (Esperar-morrer): T_i espera por T_j ; senão T_i aborta
 - Wound-wait (Ferir-esperar): T_j aborta; senão T_i espera

Detecção de Deadlock

- Atualiza e verifica grafo de espera
 - Aborta uma das transação em deadlock
 - Algoritmo de seleção da vítima - evitar transações executadas há muito tempo
- Timeout

Starvation

- Transação não pode prosseguir por um período indefinido (Elmasri, 2010)
- Soluções
 - Primeiro a chegar, primeiro a ser atendido
 - Prioridade aumenta com a espera

André Santanchè

<http://www.ic.unicamp.br/~santanche>

Referências

- Elmasri, Ramez; Navathe, Shamkant B. (2005) **Sistemas de Bancos de Dados**. Addison-Wesley, 4^a edição em português.
- Elmasri, Ramez; Navathe, Shamkant B. (2010) **Sistemas de Banco de Dados**. Pearson, 6^a edição em português.
- Ramakrishnan, Raghu; Gehrke, Johannes (2003) **Database Management Systems**. McGraw-Hill, 3rd edition.
- Ramakrishnan, Raghu; Gehrke, Johannes (2003b) **Database Management Systems**. McGraw-Hill, 3rd edition (companion slides).

Licença

- Estes slides são concedidos sob uma Licença Creative Commons. Sob as seguintes condições: Atribuição, Uso Não-Comercial e Compartilhamento pela mesma Licença.
- Mais detalhes sobre a referida licença Creative Commons veja no link:
<http://creativecommons.org/licenses/by-nc-sa/3.0/>
- Agradecimentos: fotografia da capa e fundo por Ben Collins -
<http://www.flickr.com/photos/graylight/>.
Ver licença específica em
<http://www.flickr.com/photos/graylight/261480919/>