Nome: Gabriel Henrique Vieira de Oliveira

Data: 19/11/2024
Matéria: Banco de Dados - **HO15: Controle de Concorrência**

Responder às seguintes questões:

1. **Considerando a técnica de controle de concorrência por bloqueio exclusivo (binário) com protocolo 2PL estrito e confirmação (*commit*) implícita (*commit* da transação ocorre logo após a última operação da transação no escalonamento), o escalonamento Sa possui *deadlock*? Entre quais transações? Qual o escalonamento que efetivamente será executado, considerando a técnica de resolução de *deadlock* que identifique o *deadlock* e mate a transação mais recente (aquela em que sua primeira operação se inicie depois da primeira operação das outras).**

2. **Considerando a técnica de controle de concorrência por bloqueio compartilhado (ternário) com protocolo 2PL estrito e confirmação (*commit*) implícita (*commit* da transação ocorre logo após a última operação da transação no escalonamento), o escalonamento Sa possui *deadlock*? Entre quais transações? Qual o escalonamento que efetivamente será executado, considerando a técnica de resolução de *deadlock* que identifique o *deadlock* e mate a transação mais recente (aquela em que sua primeira operação se inicie depois da primeira operação das outras)?**

3. **Considerando a técnica de controle de concorrência por ordenação de registros de *timestamp*, qual o escalonamento que efetivamente será executado?**

Considere que o escalonamento Sa apresentado abaixo foi constituído a partir das transações T1, T2 e T3 também apresentadas abaixo. Ressalta-se que, em um SGBDR diversas transações devem ser escalonadas para executarem simultaneamente, aumentando assim a concorrência e, consequentemente, diminuindo o tempo de processamento. No entanto, tal concorrência demanda a utilização de técnicas de controle de concorrência para garantir as propriedades de Atomicidade, Consistência, Isolamento e Durabilidade (ACID).

T1 = r(x), r(y), w(x), r(z)

T2 = r(z), r(x), r(y), w(z)

T3 = r(y), r(z), w(y), r(x)

Sa = r3(y), r2(z), r1(x), r2(x), r3(z), r2(y), w3(y), r1(y), w2(z), w1(x), r3(x), r1(z)

RESPOSTAS

Escalonamento Sa:

r3(y), r2(z), r1(x), r2(x), r3(z), r2(y), w3(y), r1(y), w2(z), w1(x), r3(x), r1(z)

1) Sim. Um deadlock ocorre quando há um ciclo de espera por recursos bloqueados, e nenhuma transação pode continuar. No escalonamento Sa, as transações entram em um ciclo de dependências:

1. **T1** bloqueia x e tenta acessar z, que está bloqueado por **T2**;
2. **T2** bloqueia z e tenta acessar y, que está bloqueado por **T3**;
3. **T3** bloqueia y e tenta acessar x, que está bloqueado por **T1**.

Esse ciclo entre T1↔T2↔T3 causa o deadlock.

Para resolver o deadlock, utiliza-se a técnica que identifica o ciclo e mata a transação mais recente (aquela que começou por último). Como:

- T1 inicia antes de T2, e T2 inicia antes de T3, a transação mais recente é T3;

Mata-se T3 e realocam-se T1 e T2, resultando no seguinte escalonamento:

Sfinal = r1(x), r1(y), w1(x), r2(z) ,r2(x), r2(y), w2(z), r1(z);

2) Sim. Mesmo com bloqueio compartilhado, o protocolo 2PL exige que os bloqueios sejam mantidos até o commit. Como T1, T2 e T3 precisam de bloqueios exclusivos para escrita (w(x),w(y),w(z)), o mesmo ciclo de dependências ocorre. O deadlock persiste entre T1, T2 e T3.

O critério para resolver o deadlock é o mesmo. Mata-se T3, e o escalonamento resultante é:

Sfinal = r1(x), r1(y), w1(x), r2(z), r2(x), r2(y), w2(z), r1(z);

3) Nesta técnica, cada transação recebe um timestamp no momento de sua criação. A execução das operações deve respeitar a ordem crescente dos timestamps. Qualquer operação fora de ordem resulta em rejeição e rollback.

- Supomos que os timestamps sejam:
  TS(T1) < TS(T2) < TS(T3);
- No escalonamento Sa, algumas operações de T3 violam a ordem (pois T3 tenta acessar recursos antes de T1 ou T2). Isso resulta em rollback de T3;

O escalonamento final inclui apenas T1 e T2, respeitando os timestamps:

Stimestamp = r1(x), r1(y), w1(x), r2(z), r2(x), r2(y), w2(z), r1(z);