CK0117 - Sistemas de Bancos de Dados - 2022

Javam C. Machado, Edvar Filho, Daniel Praciano, Paulo Amora

TRABALHO III - Transações e Controle de Concorrência

1 Aspectos Gerais

O trabalho consiste em desenvolver um gerenciador de transações de banco de dados com o suporte para o controle de concorrência baseado na técnica de bloqueio em duas fases restritivo. O programa terá duas classes principais a Tr_Manager e a Lock_Manager.

1. Tr_Manager

   Gerencia um grafo de transações que mantém o estado de cada uma das transações concorrentes e segue os eventos de mudança de estados. Essa classe irá manter um número sequencial que tem início com valor 0 e cresce em uma unidade a medida que uma transação é criada. Este número será armazenado em uma variável chamada Tr, de tal forma que $Tr.Id$ é o identificador da transação Tr e que $Tr.Ts$ denota o seu *timestamp*.

2. Lock_Manager

   Mantém duas estruturas de dados: (1) uma tabela de bloqueios sobre itens de dados, chamada Lock_Table e (2) uma lista de espera chamada Wait_Q (na Figura 1 é chamada de Wait For Data List), que mantém, para cada item de dado, uma lista FIFO de identificadores de transações que esperam pelo item.

O identificador da transação deve estar associado ao modo de espera. Portanto supondo It_{250} um item de dado, T_{20} uma transação e LS um bloqueio no modo compartilhado, se T_{20} espera por It_{250} para fazer uma leitura, a Wait_Q de It_{250} terá um elemento $[T_{20}, LS]$. A Lock_Table deverá ser implementada como um arquivo armazenado em disco, em que cada linha mantém o identificador do item bloqueado, o Id da transação que obteve o bloqueio e seu respectivo modo de bloqueio, S para bloqueio compartilhado e X para bloqueio exclusivo. A interface do Lock_Manager deve implementar pelo menos as funções:

LS(Tr, D) : Insere um bloqueio no modo compartilhado na Lock_Table sobre o item D para a transação Tr se puder, caso contrário cria/atualiza a Wait_Q de D com a transação Tr.

LX(Tr, D) : Insere um bloqueio no modo exclusivo na Lock_Table sobre o item D para a transação Tr se puder, caso contrário cria/atualiza a Wait_Q de D com a transação Tr.

U(Tr, D) : Apaga o bloqueio da transação Tr sobre o item D na Lock_Table.

O Lock_Manager deve implementar o controle de *deadlock* baseado nas estratégia de prevenção utilizando as técnicas *Wait_Die* e *Wound_Wait*. Seu programa vai usar o *timestamp*, para tanto o Tr_Manager vai precisar registrar o *timestamp* do início de cada transação. Em caso de *deadlock* iminente quando um bloqueio sobre um item de dado é requisitado ao Lock_Manager e a fim de evitar *deadlocks*, o Lock_Manager deve (1) colocar

a transação solicitante na Wait_Q do objeto se for o caso ou (2) sinalizar um *Rollback* da transação solicitante para o Tr_Manager, de acodro com cada técnica implementada. Veja o exemplo de estruturas de dados na Figura 1.

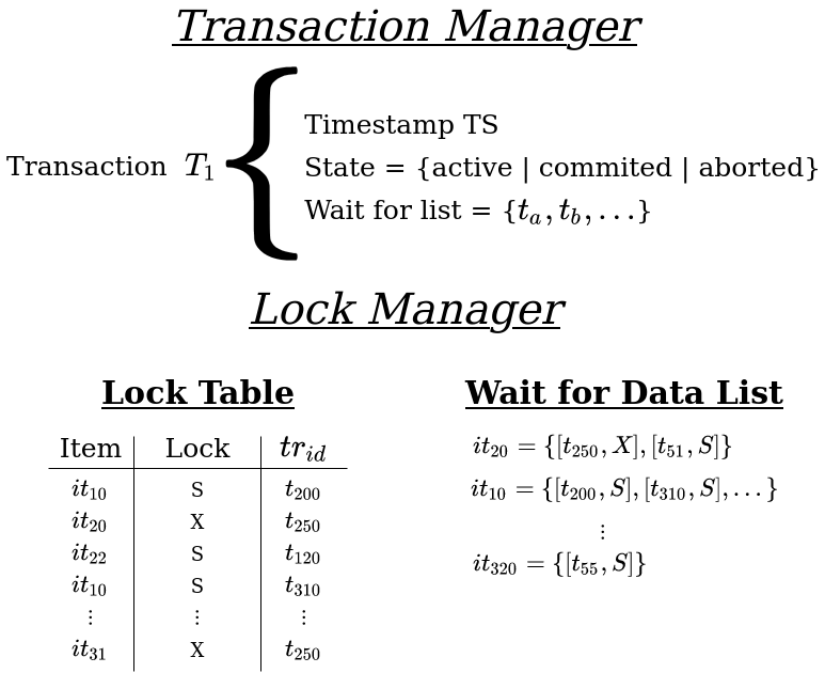

Figure 1: Estruturas de dados

2 Implementação

A interface de entrada do programa deverá permitir ao usuário entrar com uma história do tipo BT(1)r1(x)BT(2)w2(x)r2(y)r1(y)C(1)r2(z)C(2), por meio de um arquivo, onde:

- BT(X): inicia a transação X
- r1(x): transação 1 deseja ler o item x, portanto solicita bloqueio no modo compartilhado sobre o item x
- w1(x): transação 1 deseja escrever o item x, portanto solicita bloqueio no modo exclusivo sobre o item x
- C(X): Validação da transação X, quando todos os seus bloqueios devem ser liberados

O programa deve seguir a história sequencialmente e sinalizar o Tr_manager a cada comando mostrando, na interface de saída, a evolução do grafo de transações, a medida que a história vai avançando, e deve apresentar a lista de espera do item de dado que gerar *Rollback*.

A Lock_Table consistirá em um único arquivo de texto armazenado em disco, e a Wait_Q será mantida em memória contendo os itens de dados e as transações e bloqueios requisitados por estas transações para estes itens.

- A implementação deverá ser feita somente nas linguagens **C**, **C++**, **Java** ou **Python**

3 Entrega

Data da entrega: Quinta-feira - 16 de junho de 2022 até 23h59m com apresentação e arguição no LEC/DC no dia seguinte, 17 de junho, no horário da aula. O código do trabalho deve ser enviado no **classroom** até o final do horário da entrega. Envios posteriores serão penalizados. Quaisquer dúvidas podem ser enviadas aos monitores: Daniel Praciano (daniel.praciano@lsbd.ufc.br) ou Edvar Filho (edvar.filho@lsbd.ufc.br).