Sistemas de Banco de Dados

Fundamentos em Bancos de Dados Relacionais

Wladmir Cardoso Brandão
www.wladmirbrandao.com

PROCESSAMENTO DE TRANSAÇÃO

Transação

Conjunto de operações de acesso ao BD, constituindo uma unidade lógica

- Exemplo → saque de dinheiro em caixa eletrônico
  - Múltiplas operações de recuperação e atualização de dados
- Todas as operações são confirmadas ou abortadas como um bloco único
  - Somente Leitura → somente operações de recuperação de dados
  - Leitura-Escrita → contém operações de atualização de dados
  - Confirmada → operações concluídas e dados registrados permanentemente
  - Abortada → operações inócuas, sem qualquer efeito

Tipicamente executada em sistema multiusuário de forma intercalada, em ambiente multiprogramação, ou paralela em hardware com múltiplas CPUs

Transação

Processa itens de dados de forma concorrente

- Item → arquivo, bloco, registro, campo
- Concorrência → múltiplas transações concorrendo pela CPU
  - Maior Concorrência → mais transações executadas concomitantemente
  - Controle de Concorrência → garantia de independência de execução
- Granularidade → tamanho do item de dados
  - Baixa (Fina) → tamanho menor, maior concorrência, mais carga sobre o sistema de controle de concorrência
  - Alta (Grossa) → tamanho maior, menor concorrência, menos carga sobre o sistema de controle de concorrência

Transação

Executa dois tipos básicos de operação sobre itens de dados

- Leitura $r(x) \rightarrow$ copia item x do disco para variável de programa
  - Encontrar endereço de bloco de disco que contém x
  - Copiar bloco para *buffer* em memória primária
  - Copiar item x do *buffer* para variável de programa x
- Escrita $w(x) \rightarrow$ copia item x da variável de programa para disco
  - Encontrar endereço de bloco de disco que contém x
  - Copiar bloco para *buffer* em memória primária
  - Copiar item x da variável de programa x para *buffer*
  - Copiar *buffer* de memória para bloco de disco

Concorrência

Execução concorrente de múltiplas transações pode resultar em problemas

- Atualização Perdida → transações intercaladas escrevem o mesmo item, tal que a atualização de uma transação sobre o item é perdida por sobrescrição do mesmo item feita por outra transação

$$\begin{array}{lcccccc} T_1 \rightarrow & r(x) & & r(y) & & x = x + y & & w(x) \\ T_2 \rightarrow & & r(x) & & r(y) & & x = x - y & & w(x) \end{array}$$

Para as transações T_1 e T_2, considerando valores iniciais $x = 10$ e $y = 5$

- Execução sequencial → valor final de $x = 10$
- Execução concorrente → valor final de $x = 5$

A operação $w(x)$ de T_1 foi sobrescrita por $w(x)$ de T_2

Concorrência

Execução concorrente de múltiplas transações pode resultar em problemas

- Leitura Suja → uma transação atualiza um item e falha posteriormente, sendo que nesse meio tempo, outra transação lê o item atualizado (sujo)

$$
\begin{array}{lllllll}
T_1 \rightarrow & r(x) & r(y) & x = x + y & w(x) & & a \\
T_2 \rightarrow & & & & & r(x) \quad r(y) \quad x = x - y & w(x)
\end{array}
$$

Para as transações T_1 e T_2, considerando valores iniciais $x = 10$ e $y = 5$:

- Execução sequencial → valor final de $x = 5$
- Execução concorrente → valor final de $x = 10$

A operação $r(x)$ de T_2 fez uma leitura suja de $w(x)$ de T_1, que falhou

Concorrência

Execução concorrente de múltiplas transações pode resultar em problemas

- Leitura Não Repetitiva → a mesma transação lê valores diferentes para o mesmo item em momentos diferentes, uma vez que outras transações alteraram o valor do item nesse meio tempo

$$
\begin{array}{lccccccc}
T_1 \rightarrow & r(x) & & & & & & r(x) \\
T_2 \rightarrow & & r(x) & r(y) & x = x - y & w(x) & &
\end{array}
$$

A primeira e a última operação $r(x)$ de T_1 leem valores diferentes de x não escritos por T_1, o que pode ocasionar um problema na lógica de processamento se um teste condicional $(x = x)$ for executado, por exemplo

Concorrência

Execução concorrente de múltiplas transações pode resultar em problemas

- Resumo Incorreto → uma transação calcula uma função de agregação sobre itens que estão sendo atualizados por outras transações, provendo valores de resumo incorretos

Técnicas de controle de concorrência resolvem os problemas, garantindo execução concorrente e independente de transações

- Bloqueio → técnica baseada em bloqueios de leitura e escrita em itens
  - Restringe a concorrência
  - Transações bloqueiam e desbloqueiam itens quando necessário
  - Sujeito a problemas de travamento (*deadlock*) e espera indefinida (*starvation*)

Falha

Execução concorrente de transações está sujeita a diferentes tipos de falhas

- Sistema → hardware, software ou rede
- Operação → interrupção do usuário, lógica de programação
- Concorrência → técnica de controle de concorrência
- Condição de Exceção → programada na própria transação
- Disco → blocos de dados perdidos
- Catástrofe → *blackout*, incêndio, roubo, formatação acidental de disco

Sistemas de banco de dados estão preparados para lidar com falhas

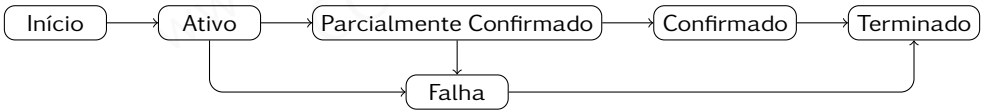
- Logging → registro histórico de transações
- Dumping → cópia de segurança do banco de dados

Processamento de Transações

Adicionalmente às operações de leitura e escrita sobre itens de dados, existem operações necessárias ao processamento de transações

- Begin Transaction (b) → marca o início da transação
- End Transaction (e) → marca o fim da transação
- Commit (c) → marca o ponto de confirmação de operações
- Abort (a) → ou *rollback*, marca o ponto de anulação de operações

Uma transação só é considerada terminada quando falhar, ou todas as suas operações tiverem sido executadas e registradas do arquivo de *log* do sistema



Processamento de Transações

Transações possuem propriedades que devem ser mantidas no processamento

- Atomicidade → unidade atômica, executada integralmente ou não executada de forma alguma
- Consistência → restrições especificadas no esquema devem continuar sendo respeitadas após processamento
- Isolamento → independência de execução, sem sofrer interferência de transações concorrentes
  - Nível 0 → sem leitura suja
  - Nível 1 → sem atualização perdida
  - Nível 2 → níveis 0 + 1
  - Nível 3 → nível 2, com leitura repetitiva
- Durabilidade → alterações confirmadas devem persistir

Escalonamento (Schedule) → intercalação de operações de transações distintas para execução simultânea

$$S_a \rightarrow b_1, r_1(x), b_2, r_2(x), r_1(y), w_2(x), w_1(y), c_2, e_2, a_1, e_1$$

- $S_a \rightarrow$ escalonamento a
- $o_i \rightarrow$ operação $o \in \{a, b, c, e, r, w\}$ realizada pela transação i
- $x, y \rightarrow$ itens de dados lidos ou escritos pelas transações
- Tipicamente b e e não são representadas, presumindo que b ocorre logo antes da primeira operação da transação, e e logo depois da última

$$S_a \rightarrow r_1(x), r_2(x), r_1(y), w_2(x), w_1(y), c_2, a_1$$

Escalonamento de Transações

Operações em um escalonamento podem estar em situação de conflito

$$S_a \rightarrow r_1(x), r_2(x), r_1(y), w_2(x), w_1(y), c_2, a_1$$

- Pertencem a transações diferentes
- Operam sobre o mesmo item
- Ao menos uma delas é w

Escalonamento de Transações

Completo → possui todas as operações de cada transação

- Operações são exatamente as mesmas das transações originais
- A ordem das operações nas transações originais é preservada
- Operações em conflito precedem ou sucedem umas as outras
- Exemplo → para as transações T_1 e T_2

$$T_1 \rightarrow r(x), r(y), w(y), a$$
$$T_2 \rightarrow r(x), w(x), c$$

o escalonamento S_a é completo

$$S_a \rightarrow r_1(x), r_2(x), r_1(y), w_2(x), w_1(y), c_2, a_1$$

Improváveis, novas transações são incorporadas a escalonamentos existentes

Escalonamento de Transações

Recuperável → transação confirmada não será desfeita, garantindo durabilidade

- Leitura Suja → pode demandar reversão de confirmação

$$S_a \rightarrow r_1(x), r_2(x), w_1(x), r_1(y), w_2(x), r_1(x), c_1, c_2$$

- Nenhuma transação T que leia um item escrito por outra transação T' pode confirmar antes de T'

$$S_a \rightarrow r_1(x), r_2(x), w_1(x), r_1(y), w_2(x), r_1(x), c_2, c_1$$

Escalonamento de Transações

Serial → sem intercalação, com todas as operações de uma transação precedendo todas as operações de outra

$$S_a \rightarrow r_1(x), w_1(x), r_1(y), r_1(x), c_1, r_2(x), w_2(x), c_2$$

- Isolamento → transação não sofre interferência de outra
- Concorrência → limitada, gerando ociosidade de CPU

Não Serial → operações de transações intercaladas

$$S_a \rightarrow r_1(x), r_2(x), w_1(x), r_1(y), w_2(x), r_1(x), c_2, c_1$$

- Isolamento → não é garantido
- Concorrência → melhor aproveitamento de CPU

Escalonamento de Transações

Serialização → processo de determinação de escalonamentos não seriais que sejam equivalentes a seriais, garantindo isolamento

- Multiplicidade → $n!$ escalonamento seriais possíveis para n transações, e um número muito maior de escalonamentos não seriais possíveis
  - Serializável → não serial equivalente a um serial
  - Não Serializável → não equivalente a qualquer serial
- Escalonamentos serializáveis são **corretos**
- Equivalência → operações são aplicadas a itens na mesma ordem
  - Conflito → ordem de operações em conflito nos escalonamentos é a mesma
  - Visão → operações r tem a mesma visão de itens nos escalonamentos

Escalonamento de Transações

Equivalência por Conflito → mesma ordem de operações em conflito

$$S_a \rightarrow r_1(x), w_1(x), r_1(y), w_1(y), r_2(x), w_2(x)$$

$$S_b \rightarrow r_2(x), w_2(x), r_1(x), w_1(x), r_1(y), w_1(y)$$

$$S_c \rightarrow r_1(x), r_2(x), w_1(x), r_1(y), w_2(x), w_1(y)$$

$$S_d \rightarrow r_1(x), w_1(x), r_2(x), w_2(x), r_1(y), w_1(y)$$

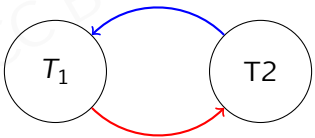
- S_d é serializável, enquanto S_c não é
  - S_c não equivalente a S_a → ordem $w_1(x)$ e $r_2(x)$ diferente
  - S_c não equivalente a S_b → ordem $w_2(x)$ e $r_1(x)$ diferente
  - S_d equivalente a S_a → mesma ordem para operações em conflito

Escalonamento de Transações

Grafo de Precedência → permite verificar se escalonamento é serializável sem precisar checar equivalência por conflito com $n!$ escalonamentos seriais

- Direcionado → ausência de ciclos indica escalonamento serializável
- Nós → transações do escalonamento
- Arestas → conflitos entre operações de transações
  - De transação sucedente para precedente, com operações em conflito

$$S_c \rightarrow r_1(x), r_2(x), w_1(x), r_1(y), w_2(x), w_1(y)$$



Processamento de Transações

SGBDs tipicamente permitem configuração de características de processamento de transações através do comando SET TRANSACTION

- ACESSO → operações permitidas em transações
  - RW → leitura e escrita (modo padrão)
  - RO → somente leitura
- ÁREA DE DIAGNÓSTICO → últimas *n* instruções SQL executadas
- ISOLAMENTO → nível de isolamento requerido no processamento
  - READ UNCOMMITED → ¬ atualização perdida
  - READ COMMITED → ¬ (atualização perdida ∨ leitura suja)
  - REPEATABLE READ → ¬ (atualização perdida ∨ leitura suja) ∧ leitura repetitiva
  - SERIALIZABLE → ¬ (atualização perdida ∨ leitura suja ∨ resumo incorreto) ∧ leitura repetitiva

Referências Bibliográficas

[1] Elmasri, Ramez; Navathe, Sham. *Fundamentals of Database Systems*. 7ed. Pearson, 2016.

[2] Silberschatz, Abraham; Korth, Henry F.; Sudarshan, S. *Database System Concepts*. 6ed. McGraw-Hill, 2011.

[3] Date, Christopher J. *An Introduction to Database Systems*. 8ed. Pearson, 2004.