

Sumário

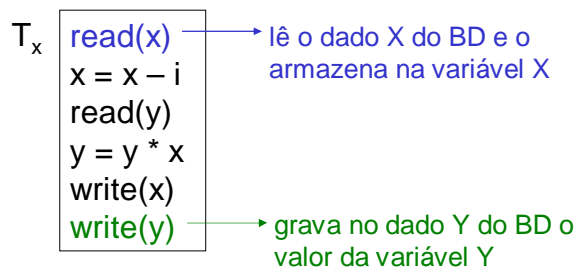
- 1 Introdução ao Processamento de Consultas
- 2 Otimização de Consultas
- 3 Plano de Execução de Consultas
- 4 Introdução a Transações**
- 5 Recuperação de Falhas
- 6 Controle de Concorrência
- 7 Fundamentos de BD Distribuído

Introdução a Transações

- SGBD
 - sistema de **processamento de operações de acesso** ao BD
- SGBDs são em geral **multi-usuários**
 - processam simultaneamente operações disparadas por vários usuários
 - deseja-se alta disponibilidade e baixo tempo de resposta
 - execução intercalada de conjuntos de operações
 - exemplo: enquanto uma operação *i* faz I/O, outra operação *j* é selecionada para execução
- Um conjunto de operações é chamado **transação**

Transação

- Unidade lógica de processamento em um SGBD
- Composta de uma ou mais operações
 - seus limites podem ser determinados em SQL
- De forma abstrata e simplificada, uma transação pode ser encarada como um conjunto de operações de leitura e escrita de dados



Propriedades de uma Transação

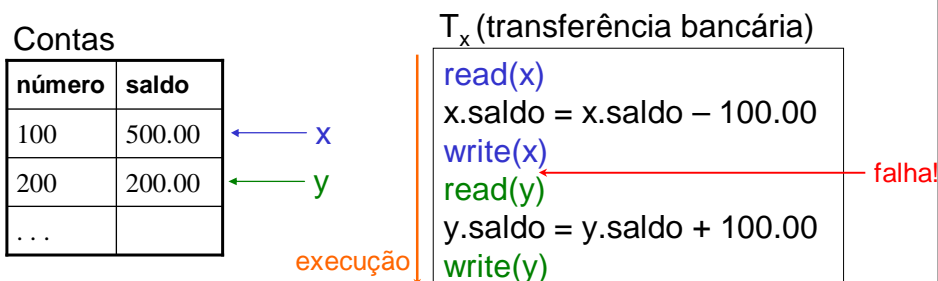
- Requisitos que sempre devem ser atendidos por uma transação
- Chamadas de **propriedades ACID**
 - Atomicidade
 - Consistência
 - Isolamento
 - Durabilidade ou Persistência

Atomicidade

- Princípio do “*Tudo ou Nada*”
 - ou todas as operações da transação são efetivadas com sucesso no BD ou nenhuma delas se efetiva
 - preservar a integridade do BD
- Responsabilidade do subsistema de recuperação contra falhas (*subsistema de recovery*) do SGBD
 - desfazer as ações de transações parcialmente executadas

Atomicidade

- Deve ser garantida, pois uma transação pode manter o BD em um estado inconsistente durante a sua execução



Consistência

- Uma transação sempre conduz o BD de um estado consistente para outro estado também consistente
- Responsabilidade conjunta do
 - DBA
 - definir todas as *RIs* para garantir estados e transições de estado válidos para os dados
 - exemplos: salário > 0; salário novo > salário antigo
 - subsistema de *recovery*
 - desfazer as ações da transação que violou a integridade

Isolamento

- No contexto de um conjunto de transações concorrentes, a execução de uma transação T_x deve funcionar como se T_x executasse de forma isolada
 - T_x não deve sofrer interferências de outras transações executando concorrentemente
- Responsabilidade do subsistema de controle de concorrência (*scheduler*) do SGBD
 - garantir escalonamentos *sem interferências*

Isolamento

T ₁	T ₂
read(A)	
A = A - 50	
write(A)	
	read(A)
	A = A+A*0.1
	write(A)
read(B)	
B = B + 50	
write(B)	
	read(B)
	B = B - A
	write(B)

escalonamento válido

T ₁	T ₂
read(A)	
A = A - 50	
	read(A)
	A = A+A*0.1
	write(A)
	read(B)
write(A)	
read(B)	
B = B + 50	
write(B)	
	B = B - A
	write(B)

T₁ interfere em T₂

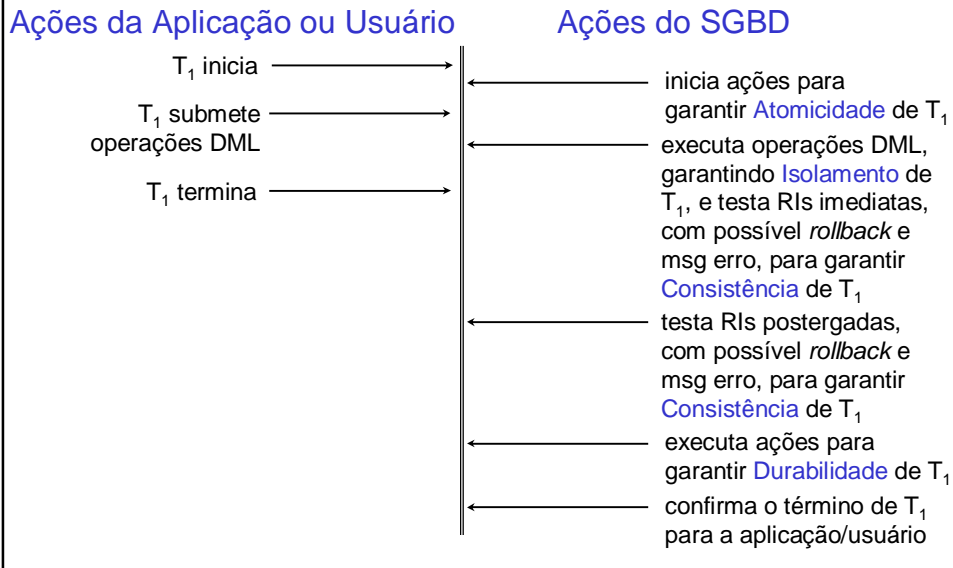
T₂ interfere em T₁

escalonamento inválido

Durabilidade ou Persistência

- Deve-se garantir que as **modificações realizadas por uma transação que concluiu com sucesso persistam no BD**
 - nenhuma falha posterior ocorrida no BD deve perder essas modificações
- Responsabilidade do **subsistema de recovery**
 - **refazer** transações que executaram com sucesso em caso de falha no BD

Gerência Básica de Transações



Transações em SQL

- Por *default*, todo comando individual é considerado uma transação
 - exemplo: `DELETE FROM Pacientes`
 - exclui todas ou não exclui nenhuma tupla de pacientes, deve manter o BD consistente, etc
- SQL Padrão (SQL-92)
 - `SET TRANSACTION`
 - inicia e configura características de uma transação
 - `COMMIT [WORK]`
 - encerra a transação (solicita efetivação das suas ações)
 - `ROLLBACK [WORK]`
 - solicita que as ações da transação sejam desfeitas

Transações em SQL

- Principais configurações (SET TRANSACTION)
 - modo de acesso
 - READ (somente leitura), WRITE (somente atualização) ou READ WRITE (ambos - *default*)
 - nível de isolamento
 - indicado pela cláusula ISOLATION LEVEL *nível*
 - *nível* para uma transação T_i pode assumir (várias configurações possíveis)
 - SERIALIZABLE (T_i executa com completo isolamento - *default*)
 - ...
 - READ COMMITTED (T_i só lê dados efetivados, mas outras transações podem escrever em dados lidos por T_i) – aceitável, por exemplo, em casos onde transações que lêem dados não irão escrever esses dados posteriormente
 - ...

Transações em SQL

- Exemplo (SQL embutida)

```
EXEC SQL SET TRANSACTION
        WRITE
        ISOLATION LEVEL SERIALIZABLE;
...
for(;;)
{...
EXEC SQL INSERT INTO Empregados
        VALUES (:ID, :nome, :salario)
...
EXEC SQL UPDATE Empregados
        SET salário = salário + 100.00
        WHERE ID = :cod_emp
if (SQLCA.SQLCODE <= 0) EXEC SQL ROLLBACK;
...
}
EXEC SQL COMMIT;
...
```

Variações do SQL Padrão

- Adotado por alguns SGBDs (script SQL)

```
BEGIN TRANSACTION T1

UPDATE Medicos
SET nroa = NULL
WHERE nroa = @nroAmb

IF @@ERROR <> 0 ROLLBACK TRANSACTION T1

DELETE FROM Ambulatorios
WHERE nroa = @nroAmb

IF @@ERROR <> 0 ROLLBACK TRANSACTION T1
ELSE COMMIT TRANSACTION T1
...
```