Aspectos de Implementação de Banco de Dados

Aula 1

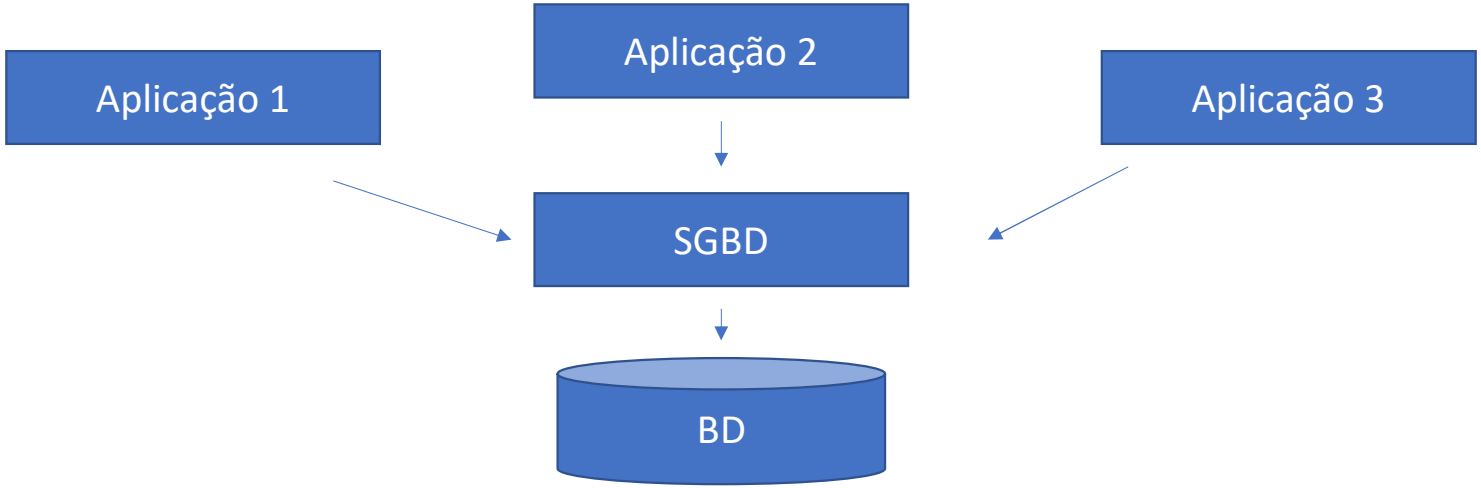
SGBD

- Sistema de Gerenciamento de Banco de Dados
  - um conjunto de dados inter-relacionados e programas para acessá-los
  - Tem como objetivo prover um ambiente conveniente, eficiente e seguro para armazenar e recuperar dados

- Quais as principais funções de um SGBD (vantagens de uso)?
  - independência de dados (separação dos dados das aplicações)
  - segurança dos dados
  - controle de acesso aos dados
  - **acesso aos dados por várias aplicações**

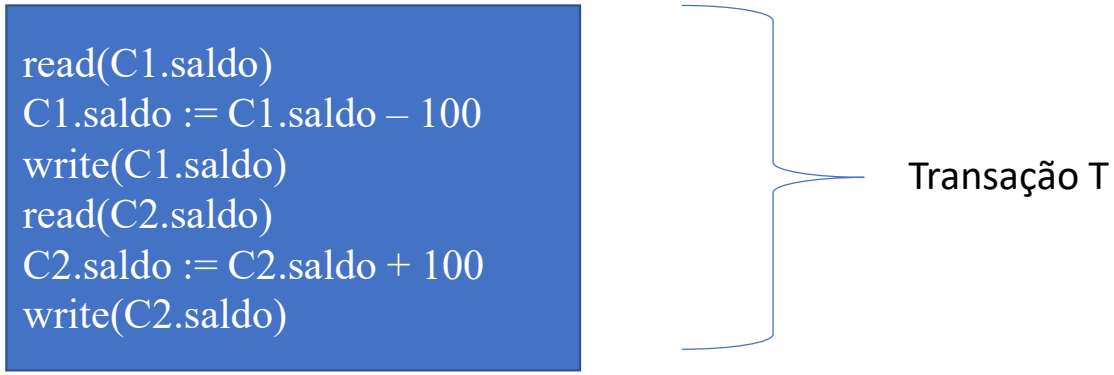
Ambiente multiusuário

- vários usuários (aplicações) com acesso aos mesmos dados ao mesmo tempo
- As aplicações executam de forma concorrente sobre o mesmo processador, entrelaçando suas execuções.
  - o processador executa a operação de uma aplicação, depois a de outra e assim por diante;
- execução concorrente possibilita maior eficiência no uso do processador



Transações

- Cada aplicação define operações a serem executadas sobre o BD. Certas "sequências" de operações "básicas" sobre os dados (leitura e escrita) devem ser executadas **indivisivelmente**, pois definem uma operação mais complexa da aplicação:
- Exemplo: operação de transferência de R$100,00 da conta C1 para a conta C2



Cada uma destas operações indivisíveis é chamada de **transação**: unidade de trabalho da aplicação.

Transação

T1:
read(a)
a:=a+100
write(a)
read(b)
b:=b-50
write(b)

Representação de uma única transação.

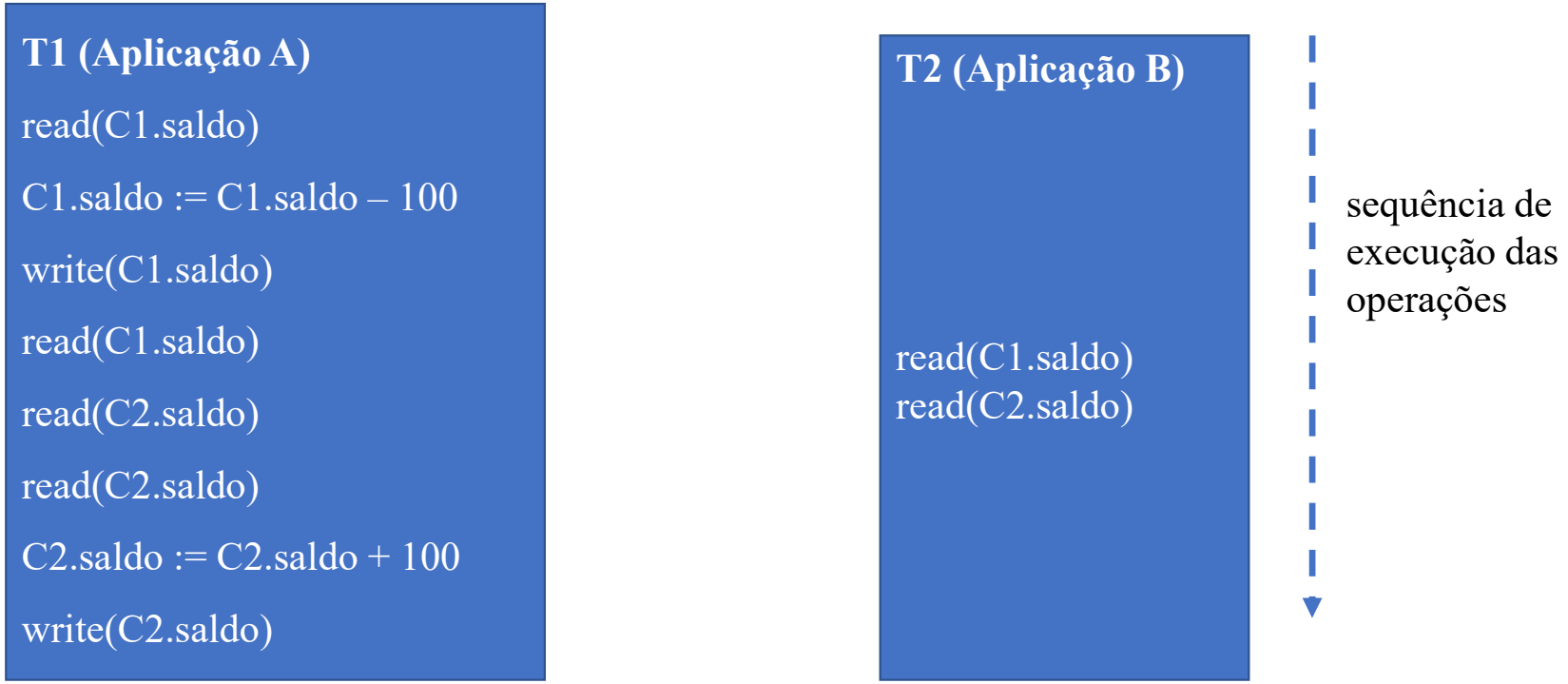
T1	T2	T3
		read(a)
		write(a)
read(a)		
write(a)		
	read(b)	
	write(b)	

Representação de transações concorrentes.

Problemas com transações concorrentes

- Aplicação A realiza a transferência de C1 para C2, e a aplicação B lê os saldos

de C1 e de C2 **depois** que os 100 reais foram retirados de C1, mas **antes** que eles sejam depositados em C2.



Problemas com transações concorrentes

- Sistemas de computador estão sujeitos a falhas (erros no sistema operacional, problemas no hardware, problemas de execução na aplicação).
- A interrupção de um programa pode levar a dados incorretos:

```
read(C1.saldo)
C1.saldo := C1.saldo – 100
write(C1.saldo)
FALHA
read(C2.saldo)
C2.saldo := C2.saldo + 100
write(C2.saldo)
```

Transações Concorrentes

Para garantir a consistência das informações, em caso de acessos concorrentes e de falhas, são necessários **protocolos** que permitam:

(1) a sincronização dos acessos concorrentes e

(2) A anulação ou complementação dos efeitos parciais das operações interrompidas pelas falhas.

Propriedades das transações

- Toda transação tem que assegurar a verificação das seguintes **propriedades** (**ACID**):

(A) Atomicidade: do ponto de vista da aplicação, ou a transação termina corretamente e seus efeitos se refletem nos dados manipulados, ou ela é interrompida e não tem nenhum efeito sobre os dados (ela é abandonada) ***princípio do tudo ou nada***;

(C) Consistência: uma transação ***preserva a consistência dos dados*** que ela manipula, ou seja ela deve executar, apenas, operações corretas, do ponto de vista da aplicação sobre o banco de dados. Isto é, a transação não pode transgredir qualquer restrição de integridade definida no SGBD;

(I) Isolamento: em ambientes multiusuário, a execução de uma transação não deve ser influenciada pela execução de outras transações.

(D) Durabilidade (ou persistência): ***os efeitos de uma transação validada são permanentes na base de dados.***

Uma transação que satisfaz essas quatro propriedades é denominada ACID.

Propriedades das Transações

- A propriedade de **Consistência** é assegurada pelas restrições de integridade (definidas segundo as aplicações)
- Isolamento entre transações: garantido pelos mecanismos de controle de concorrência
- Atomicidade e Durabilidade: garantidas pelos mecanismos de recuperação após falhas pelo controle de acesso (gerenciado pelo SGBD).

Transações

- **Operações básicas sobre o BD:**
  - READ (x) ou r(x): leitura de um dado do BD
  - WRITE (x) w(x): escrita de um dado no BD

- **Marcadores de uma transação:**
  - START : início de uma transação
  - COMMIT : fim de uma transação executada com sucesso
  - ABORT : fim de uma transação que não foi completada corretamente

Transações

START
read(C1.saldo)
C1.saldo := C1.saldo – VALOR
write(C1.saldo)
read(C2.saldo)
C2.saldo := C2.saldo + VALOR
write(C2.saldo)
COMMIT

Terminou com sucesso

Controle de Concorrência

- Se as transações fosses executadas em série, manteriam o BD consistente.
- Com a execução concorrente, as operações das transações se entrelaçam: se ambas ***atualizaram o mesmo dado*** pode haver inconsistência.

mais simples: proibir totalmente o entrelaçamento na execução de transações (execução SERIAL)

Consequência : má utilização dos recursos do sistema

- mais econômica : execução concorrente mas serializável de duas ou mais transações

Problemas que podem ocorrer devido ao entrelaçamento de operações:

- *lost update* (alteração perdida)
- *inconsistent retrieval* (recuperação inconsistente)
- *phantom problem* (problema dos dados-fantasma)

Lost Update

- Duas escritas sobre o mesmo valor inicial

TIME	T_1	T_2
t1	T_{1_begin}	
t2		T_{2_begin}
t3	$read(flightseat)$	
t4		$read(flightseat)$
t5	$checkIfFree(flightseat)$	
t6		$checkIfFree(flightseat)$
t7	$flightseat = "Mr.Smith")$	
t8	$write(flightseat)$	
t9		$flightseat = "Mrs.Mayr")$
t10		$write(flightseat)$
t11	T_{1_commit}	
t12		T_{2_commit}
INITIAL: $flightseat = "free"$, RESULT: $flightseat = "Mrs.Mayr"$		

Figura 1 - *Lost Update*: Mr. Smith sem assento [Stroka, 2009]

Inconsistent Retrieval

- O valor recuperado
- deixou de existir

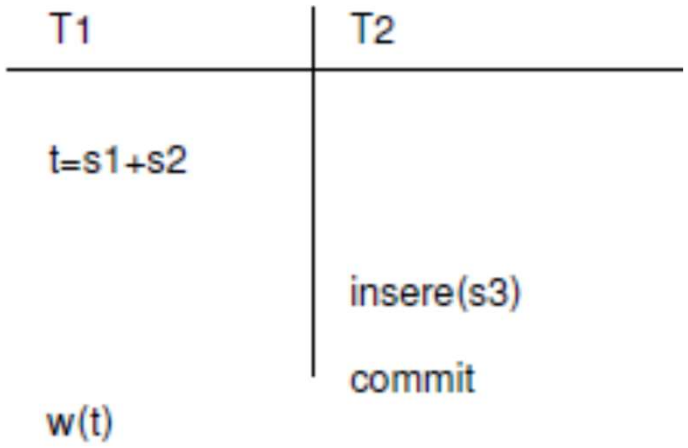
TIME	T_1	T_2
t1	T_{1_begin}	
t2	$read(money)$	
t3	$money = money + 400$	
t4	$write(money)$	
t5	...	
t6		T_{2_begin}
t7		$read(hotel)$
t8		$read(money)$
t9	$T_{1\_rollback}$	
t10		$money = money - 600$
t11		$hotel = hotel + 600$
t12		$write(money)$
t13		$write(hotel)$
t14		T_{2_commit}
INITIAL: $money = 300$, RESULT: $money = 100$		

Figura 2 - *Dirty Read*: Erro no saldo da conta bancária [Stroka, 2009]

Leitura Suja

Phantom Problem (Problema Fantasma)

- Durante uma operação executada sobre um conjunto de dados, um novo valor e inserido ou excluído e não computado na operação.



T1	T2
Select * from tabela1	
	insert into tabela1..
Commit	
	commit

Como identificar execuções que garantem a consistência?

- Execuções SERIAIS:

  Execuções corretas

  Executadas individualmente

  A execução de uma transação não interfere na execução da outra

MAU USO DOS RECURSOS

t1	t2
read(a) a=a-50 write(a) read(b) b=b+50 write(b) commit	read(a) temp=a*0,1 a=a-temp write(a) read(b) b=b+temp write(b) commit

Início:
a-1000
b-2000
Fim:
a-855
b-2145

Como identificar execuções que garantem a consistência?

- Execuções SERIALIZAVEL:
  - Execuções concorrentes
  - Equivalente a uma execução seria
  - Execuções corretas

t1	t2
read(a) a=a-50 write(a)	
	read(a) temp=a*0,1 a=a-temp write(a)
read(b) b=b+50 write(b)	
	read(b) b=b+temp write(b)

Início: a:1000
b:2000
Fim: a:855
b:2145

Exercício

- Identifique os conflitos entre as transações:

1) r1(a) w1(a) r2(b) w2(b) r2(a) w2(a) c2 r3(c) w3(c) c3 r1(c) c1
2) r2(a) w2(a) r3(b) w3(b) r2(c) w2(c) c2 c3 r1(d) w1(d) c1
3) r1(a) r1(b) w1(b) r3(a) r3(c) w3(c) r2(a) r2(d) w2(d) c1 c2 c3
4) r2(a) r2(b) w2(b) r3(b) w3(b) r3(b) r3) c3 c2 r1(c) r1(b) w1(c) w1(b) c1

1) r1(a) w1(a) r2(b) w2(b) r2(a) w2(a) c2 r3(c) w3(c) c3 r1(c) c1