

---

# Banco de Dados I

## 6 – Transações e Controle de Concorrência

---

Grinaldo Lopes de Oliveira (grinaldo@gmail.com)  
Curso Superior de Tecnologia em  
Análise e Desenvolvimento de Sistemas

---

# Agenda

- **Aprendendo**
  - **Transações**
  - **Controle de Concorrência**





Transação

---

## O que é uma Transação

- O termo transação refere-se a uma coleção de operações que forma uma única unidade lógica de trabalho.
    - Por exemplo, uma transferência de dinheiro de uma conta para outra é uma transação consistindo de duas atualizações, uma para cada conta.
-

---

## Exemplo de Transação

- Transferência bancária

BEGIN TRANSACTION

```
update conta1  
set saldo = saldo - 100
```

```
update conta2  
set saldo = saldo + 100
```

COMMIT TRAN

---

---

# Características de Transações

**T1: Transação T<sub>1</sub>**  
A=ler\_item (X);  
A = A - N;  
escrever\_item (X,A);  
B=ler\_item (Y);  
B = B + N;  
escrever\_item (Y,B);

**T2: Transação T<sub>2</sub>**  
C=ler\_item (X);  
C = C + M;  
escrever\_item (X,C);

Transações de vários  
usuários



- (1) executadas  
concorrentemente
- (2) podem acessar e atualizar o  
mesmo item de dados

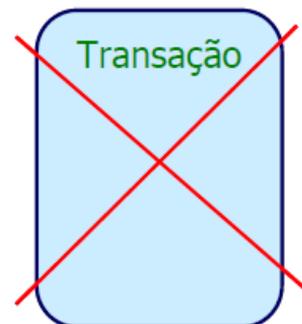
# Transações do SGBD

Na execução de uma **transação** o **SGBD** deve garantir:

(1) **Todas as operações** na transação foram completadas com **sucesso** e seu efeito será **gravado permanentemente** no BD



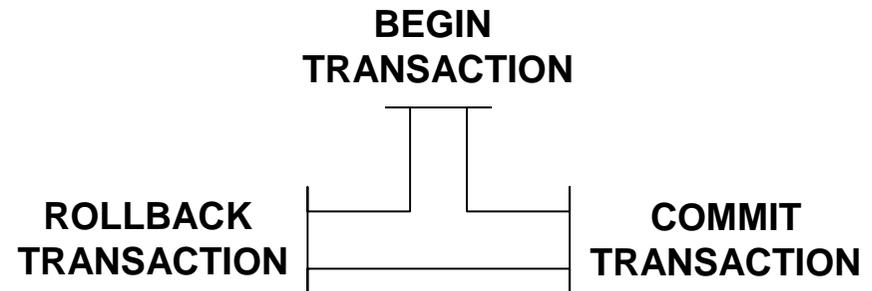
(2) Transação **NÃO** terá nenhum **efeito sobre o BD** ou outras transações (**TRANSAÇÃO FALHAR DURANTE EXECUÇÃO**)



---

# Finalização de uma Transação

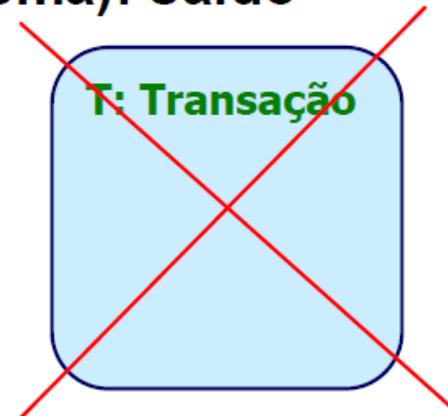
- Commit Transaction
  - ❑ Encerramento da transação com sucesso;
  - ❑ Torna as atualizações permanentes;
- Rollback Transaction
  - ❑ Término de transação com erro;
  - ❑ Retorna o banco de dados à posição anterior à Transação;



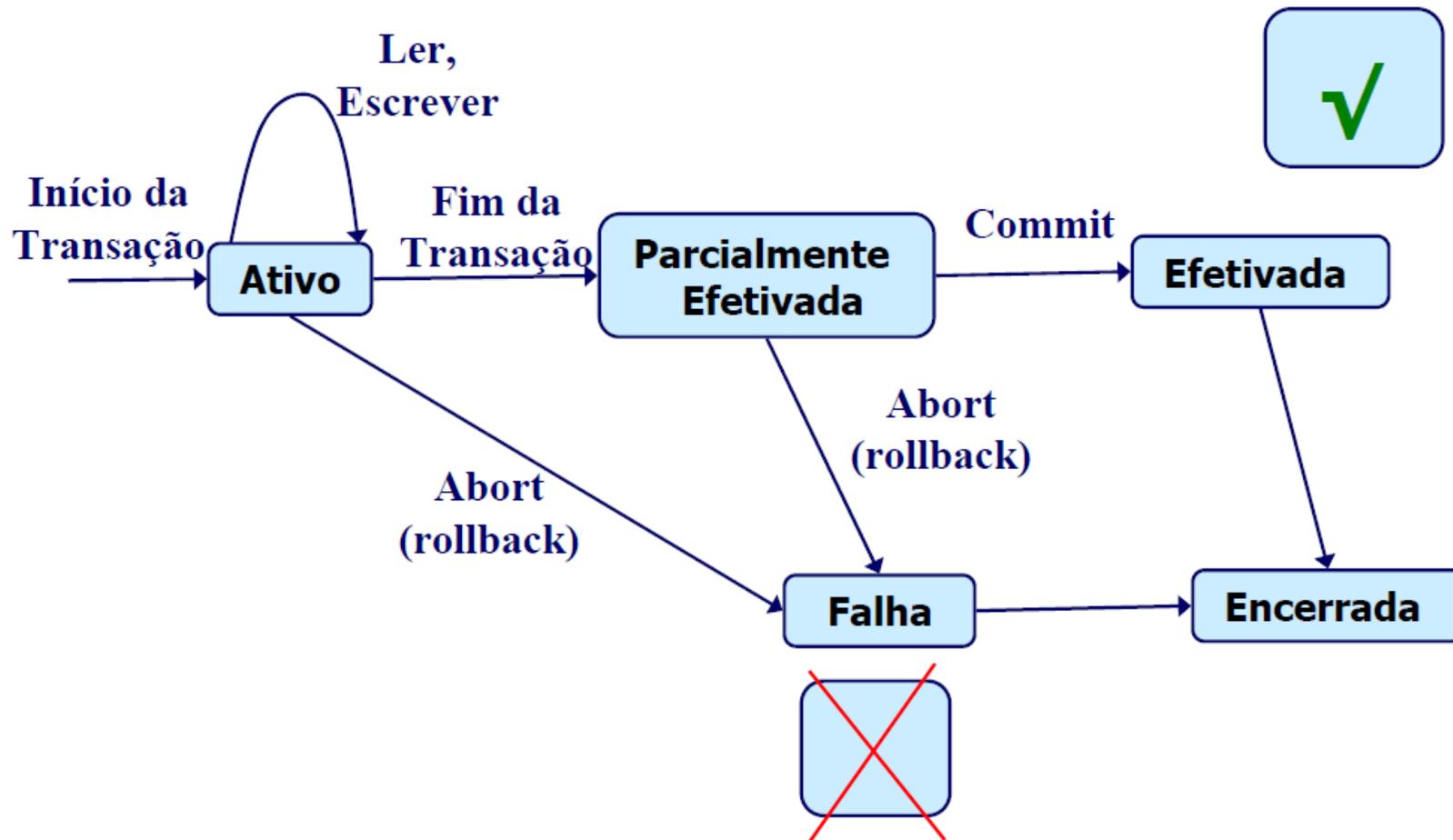
---

# O que faz uma transação falhar?

- ◆ **Computador falhar por hardware, software ou rede**
- ◆ **Erro durante execução de operação na transação: estouro de variáveis**
- ◆ **Condições de exceção detectadas pela transação (necessitam o cancelamento da mesma): saldo insuficiente em conta**
- ◆ **Falta de energia, ar-condicionado**



# Estados de uma Transação



---

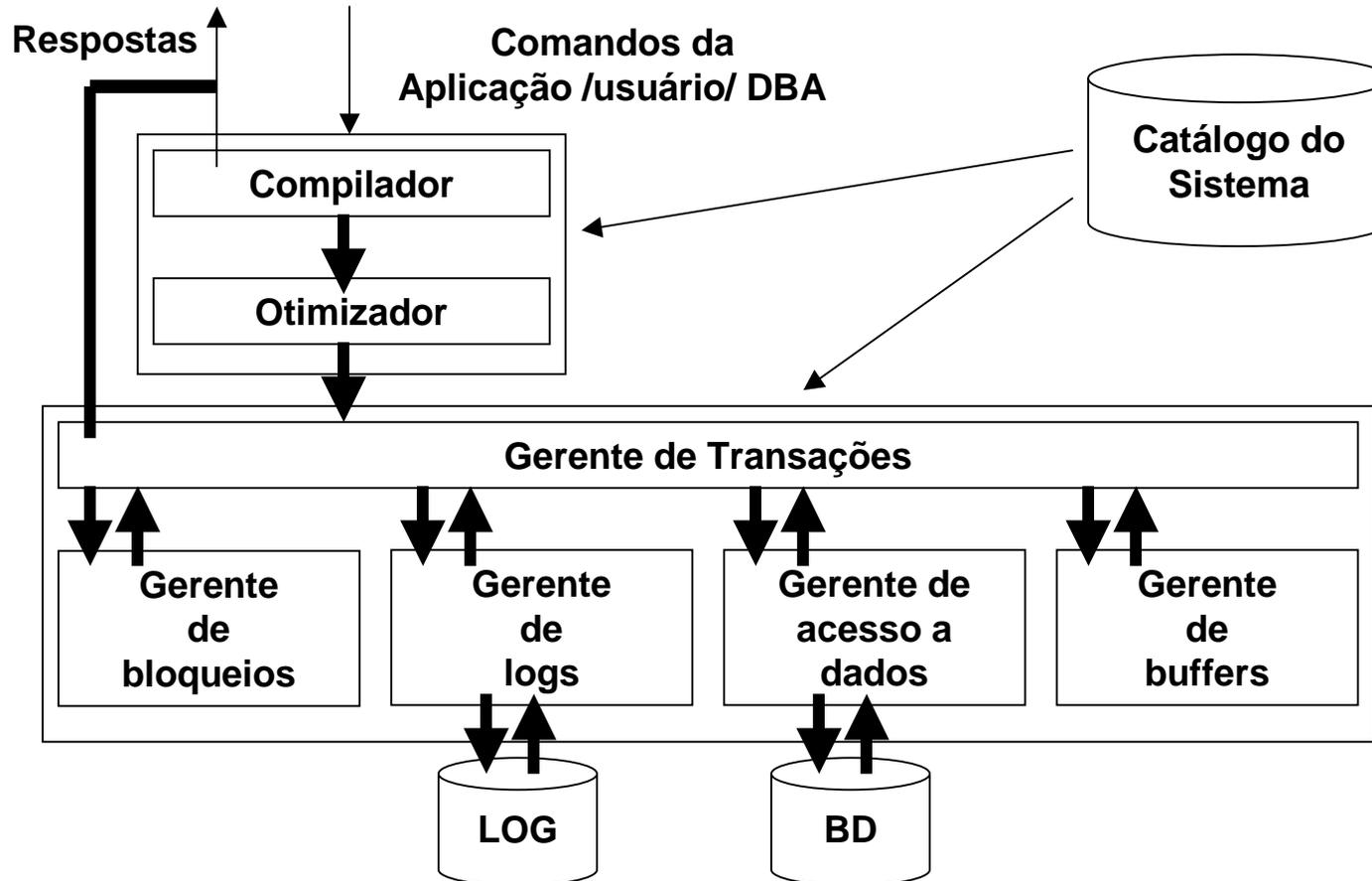
# Propriedades de uma Transação

## ■ ACID

- ❑ A: Atomicidade
    - Unidade lógica atômica (tudo ou nada)
  - ❑ C: Consistência (Preservação)
    - Ao final de uma transação o banco continua consistente
  - ❑ I: Isolamento
    - A execução de uma transação não deve sofrer interferência de outras transações concorrentes
  - ❑ D: Durabilidade (Persistência)
    - Após o ponto de confirmação, as alterações devem persistir no banco de dados
-

# Arquitetura de um SGBD

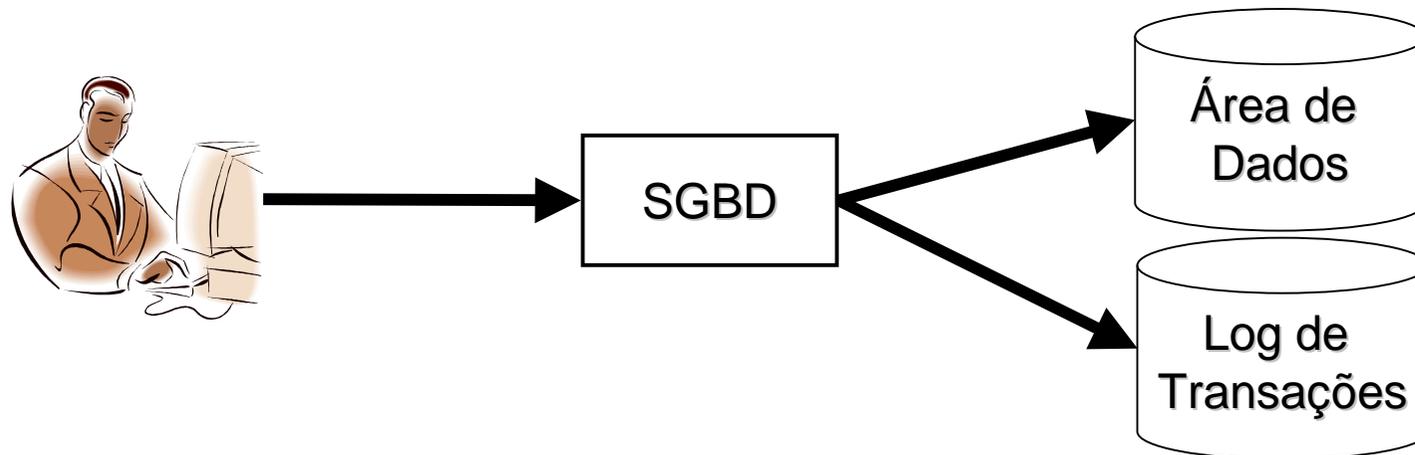
## Referência



# Log de Transações

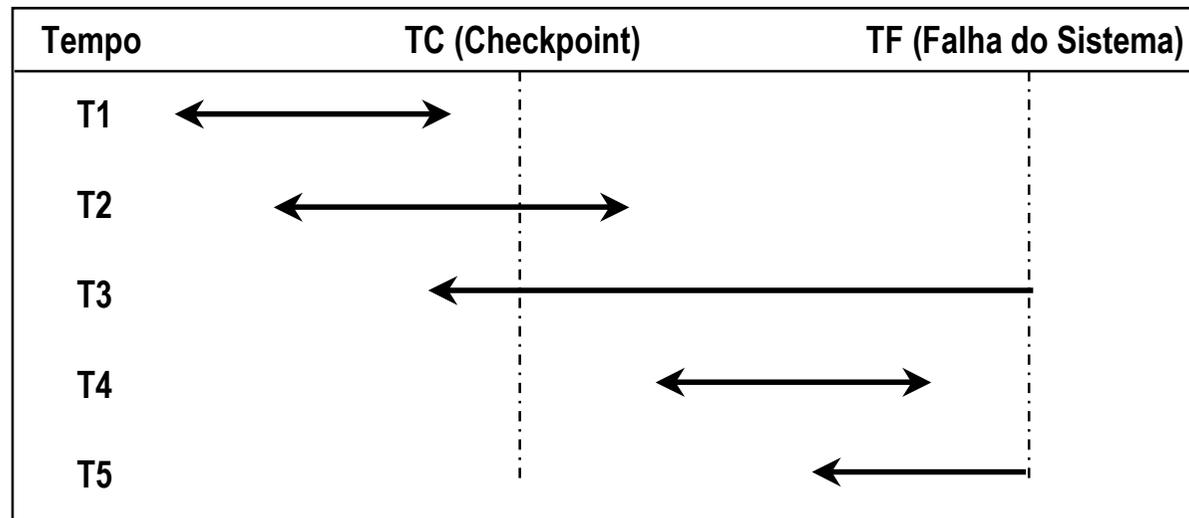
- Características:

- Registro de todas as operações (alterações) realizadas no banco de dados;
- Base para o processo de recuperação (Recovery)
- Atualização anterior à operação realizada sobre os dados;
- Atualização física (I/O), sem utilização de buffer's.



# Recuperação de Consistência (Falha de Sistema)

## ■ Processo de Recuperação



📄 T1

➔ Nada a fazer

📄 T2 e T4

➔ Ser refeitas (REDO)

📄 T3 e T5

➔ Ser desfeitas (UNDO)

• Checkpoint:

Atualização física dos  
buffer's do SGBD.

# Recuperação de Consistência (Falha de Sistema)

- Processo de Recovery
  - Iniciamos o algoritmo com duas listas (Undo e Redo). Undo com a lista das transações no momento do último checkpoint e Redo vazia;
  - Pesquisamos no log de transações a partir do registro de checkpoint;
  - Se um Begin Transaction for encontrado, adicionamos a transação à lista Undo;
  - Se um Commit for encontrado, move a transação da lista Undo para a Redo;

|                               |          |                          |
|-------------------------------|----------|--------------------------|
| <b>Refazer as transações</b>  | <b>→</b> | <b>Forward Recovery</b>  |
| <b>Desfazer as transações</b> | <b>→</b> | <b>Backward Recovery</b> |

---

# Transação Distribuída

## ■ Two-Phase Commit

- Protocolo para garantir o conceito de transação entre servidores distintos (Transação inter-servidores);
  - Opção de efetivar ou cancelar a transação cabe a um Coordenador:
    - FASE1: O coordenador instrui todos os participantes para estarem prontos para efetivar a transação. (Forçar a gravação dos registros de log envolvidos na transação). Os participantes devem responder “Ok” ou “Não Ok” caso tenham conseguido ou não se preparar;
    - FASE2: Se todas as respostas forem “OK”, o coordenador deve transmitir um Commit a todos os participantes, caso contrário emite Rollback (algum “Não Ok”).
-

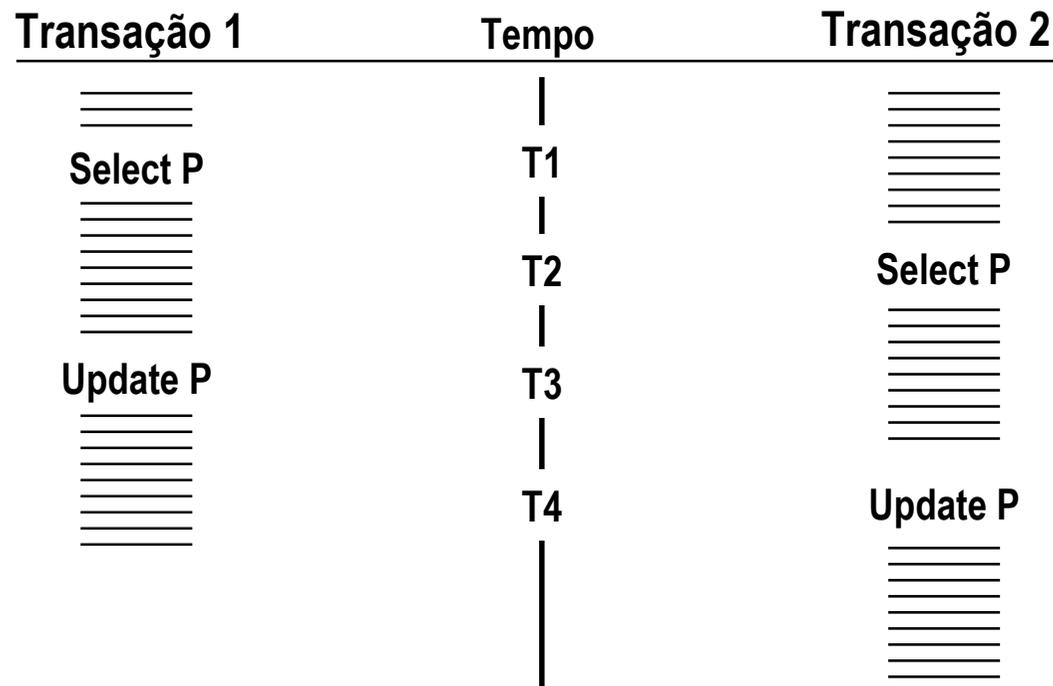


# Controle de Concorrência

---

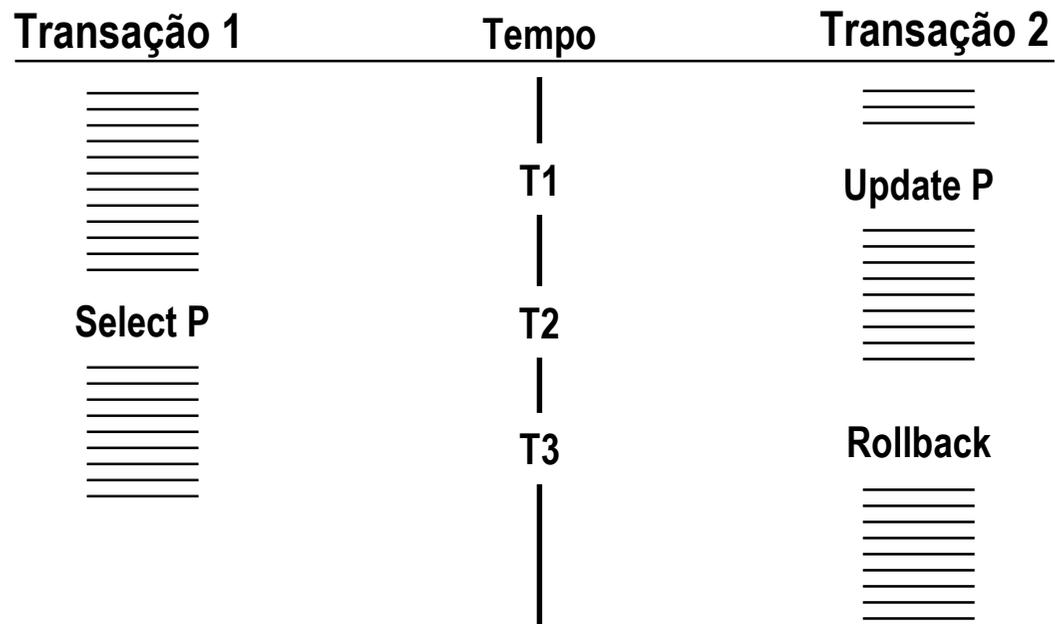
# Problemas de Concorrência

- Atualização Perdida:
  - A Transação 1 perde a atualização no tempo T4.



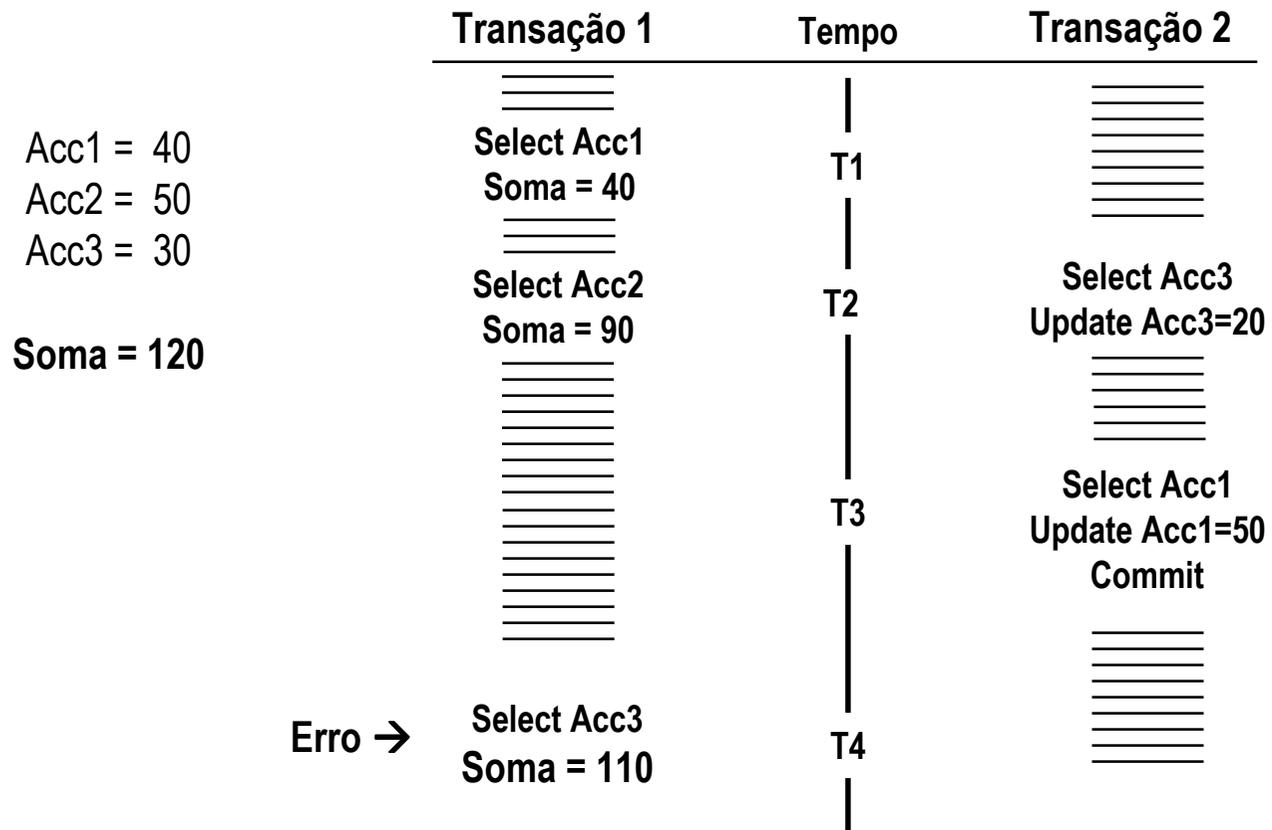
# Problemas de Concorrência

- Dependência de Transação não Concluída:
  - A Transação 1 tem acesso um dado que depende da conclusão da Transação 2.



# Problemas de Concorrência

## ■ Análise Inconsistente:



---

# Controle de Concorrência

## Primitiva

### **“Princípio da correção”**

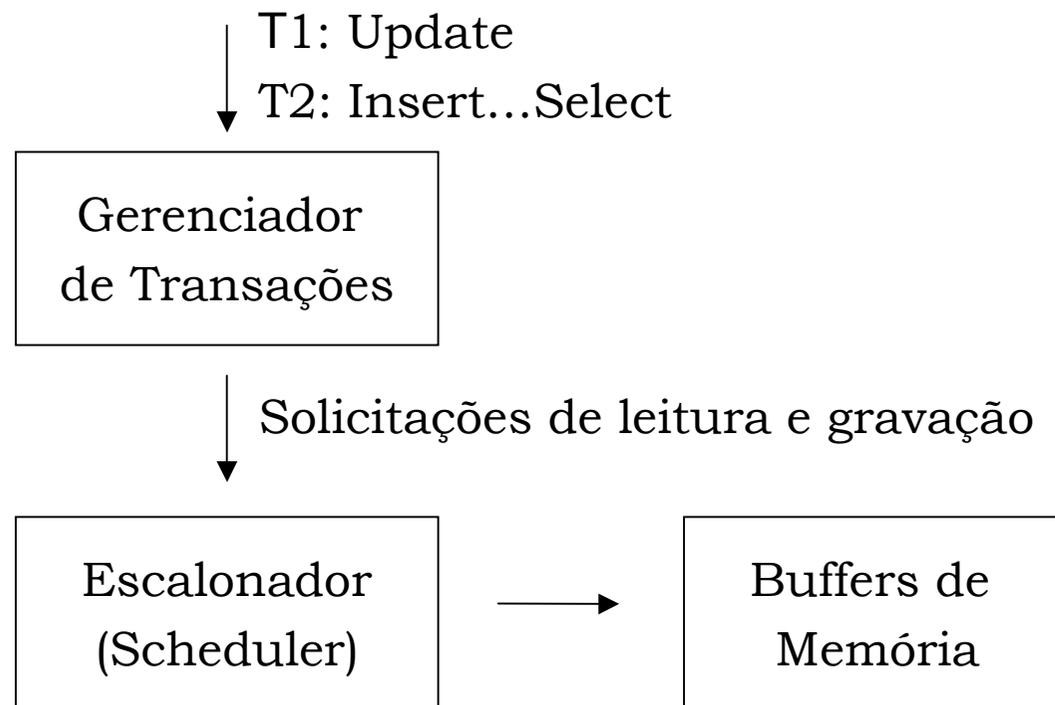
- Toda transação, se executada de forma isolada, transformará qualquer estado consistente em outro estado consistente.



# Controle de Concorrência

## Escalonador (Scheduler)

- Escalonador : Tem como função assegurar que as transações preservem a consistência quando executadas simultaneamente.



---

# Controle de Concorrência

## Escalonamento

- Serialização (Serializabilidade, Seriabilização)
    - Requisito abstrato que tem como objetivo assegurar que as transações executem de forma concorrente e ao final o estado do banco continue consistente.
  - Escalonamento Serializável:
    - Quando seu efeito sobre o estado do banco de dados é igual ao de algum escalonamento serial.
  - Escalonamento não-serializável:
    - Escalonamento que não assegura o estado consistente do banco de dados
-

---

# Controle de Concorrência

## Bloqueios

- Como os SGBDs impõem a serialização ?
  - A técnica mais comum adotada é a utilização de bloqueios sobre elementos do banco de dados, a fim de evitar um comportamento não serializável. (Pessimista)
  - Uma transação obtém bloqueios sobre os elementos do banco de dados para impedir que outras transações acessem esses elementos ao mesmo tempo e, portanto, haja risco da não serialização.
  - É utilizada uma tabela de bloqueios
  - Há outras técnicas:
    - Timbre de hora – Timestamp
    - Técnicas otimistas- Entendem que operações conflitantes são exceção
-

---

# Bloqueios

## ■ Definição:

- Mecanismo que permite a uma transação impedir que outras acessem ou atualizem registros de forma a evitar os problemas de concorrência;
  - XLOCK:
    - Bloqueio exclusivo sobre os registros;
    - Utilizado para atualizações;
  - SLOCK:
    - Bloqueio compartilhado sobre os registros;
    - Utilizado para consultas.
-

---

# Bloqueios

- Relação entre os Bloqueios:

|              | <b>XLOCK</b> | <b>SLOCK</b> |
|--------------|--------------|--------------|
| <b>XLOCK</b> | Não          | Não          |
| <b>SLOCK</b> | Não          | Sim          |

---

---

# Controle de Concorrência

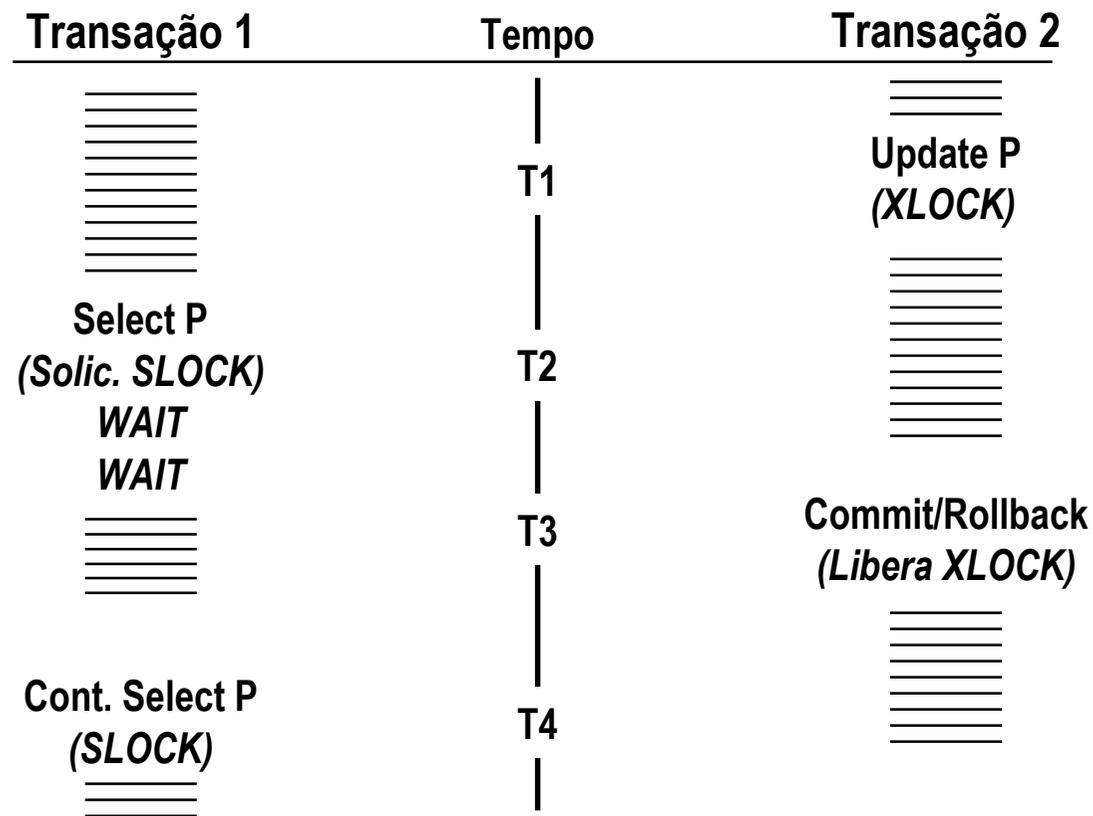
## Bloqueios

- Granularidade dos bloqueios
    - Linha (registros)
    - Página de disco
    - Tabela
    - Banco de Dados
  - O SGBD decide automaticamente o melhor nível de granularidade (Bloqueio de múltipla granularidade).
  - Alguns SGBDs permitem que o usuário altere a granularidade e o tipo do bloqueio.
-



## Resolução dos Problemas de Concorrência

- Dependência de Transação não Concluída:



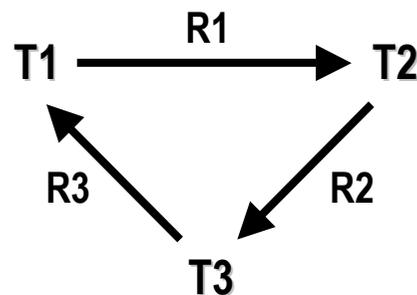


---

# Resolução dos Problemas de Concorrência

## ■ Deadlock

- ❑ Situação onde duas transações estão simultaneamente em estado de espera (Wait), cada uma aguardando pela liberação do bloqueio (Lock) da outra;
- ❑ O SGBD deve detectar e quebrar o deadlock (sacrificar uma transação);
- ❑ Gráfico de Espera (Wait-For Graph);



---

# Controle de Concorrência

## Problemas de Concorrência

- Problemas clássicos de concorrência
    - Atualização perdida (Lost Update)
    - Leitura suja (Dirty read)
    - Leitura não-repetitiva (Repeatable-read)
    - Fantasmas (Phantoms)
-

---

# Controle de Concorrência

## Problemas de Concorrência

- Atualização perdida (Lost Update)
    - Ocorre quando duas transações que acessam os mesmos itens do banco de dados têm suas operações entrelaçadas, de modo que torne incorreto o valor de algum item.
  - Leitura Suja (Dirty read)
    - Ocorre quando uma transação atualiza um item do banco de dados e, por algum motivo, outra transação ler esse item supostamente atualizado.
-

---

# Controle de Concorrência

## Problemas de Concorrência

- Leitura Não Repetitiva (Non-Repeatable read)
    - Ocorre quando uma transação lê itens em uma determinada condição e depois outra transação altera e efetiva um novo item que satisfaz a condição da transação anterior. Caso a primeira transação faça novamente uma leitura com a mesma condição, os registros podem aparecer de maneira diferente.
  - Fantasmas (Phantoms)
    - Ocorre quando uma transação lê itens em uma determinada condição e depois outra transação insere e efetiva um novo item que satisfaz a condição da transação anterior. Caso a primeira transação faça novamente uma leitura com a mesma condição, um registro “fantasma” irá aparecer
-

---

# Controle de Concorrência

## Níveis de Isolamento (Padrão SQL-92)

- Read uncommitted
    - Permite a leitura de dados não efetivados
    - Há ganho de performance, mas perda de segurança.
  - Read committed
    - Nível de isolamento Padrão
    - Leitura só pode ser feita em dados efetivados.
  - Repeatable Read
    - Registros lidos não serão alterados por outros processos, garantindo releituras idênticas.
  - Serializable
    - A mais restrita de todas.
    - Inserções ou deleções não podem ser feitas em conjuntos de registros lidos.
-

# Níveis de Isolamento (SQL-92)

| Nível de Isolamento     | Leitura Suja (Dirty Read) | Leitura não Repetível (Nonrepeatable Read) | Registro Fantasma (Phantom) |
|-------------------------|---------------------------|--|-----------------------------|
| <b>Read Uncommitted</b> | Ocorre                    | Ocorre                                     | Ocorre                      |
| <b>Read Committed</b>   | Não ocorre                | Ocorre                                     | Ocorre                      |
| <b>Repeatable Read</b>  | Não ocorre                | Não ocorre                                 | Ocorre                      |
| <b>Serializable</b>     | Não ocorre                | Não ocorre                                 | Não ocorre                  |

---

# Controle de Concorrência

## Níveis de Isolamento (Padrão SQL-92)

- Sintaxe SQL
  - A configuração do nível de isolamento pode ocorrer para todas as transações do banco de dados ou para alguma específica.
- SET TRANSACTION
  - ISOLATION LEVEL
    - [  
READ UNCOMMITTED ou  
READ COMMITTED \*(Padrão)  
REPEATABLE READ  
SERIALIZABLE  
]



---

## Debate em Sala de Aula

- Uma aplicação está com baixo tempo de resposta em virtude de muitas transações concorrentes e alto grau de bloqueios. O que você faria para minimizar este problema?



---

# Banco de Dados I

## 6 – Transações e Controle de Concorrência

---

Grinaldo Lopes de Oliveira (grinaldo@gmail.com)  
Curso Superior de Tecnologia em  
Análise e Desenvolvimento de Sistemas