

FGA0137

Sistemas de Banco de Dados 1

Prof. Maurício Serrano

Material original: Profa. Elaine Parros Machado de Sousa

Prof. Jose Fernando Rodrigues Junior

2021/2

Linguagem SQL

Transactions

Módulo 5

Transações

- **Transação**: Unidade lógica de trabalho
 - abrange um conjunto de operações de manipulação de dados que executam uma única tarefa

Conecta ao Banco de Dados

Começa transação

Operações de consulta/atualização

...

Finaliza transação

Começa transação

Operações de consulta/atualização

...

Finaliza transação

Desconecta

Transações

- São orientadas a dois tipos de problemas:
 - O que acontece se a energia acabar no meio de uma transação, ou se houver um problema com o disco?
 - O que acontece quando duas transações executam simultaneamente manipulando o mesmo dado?
- ⇒ O banco de dados pode ser levado a um estado inconsistente...

Falhas

	Tipo de falha	Abrangência	Como evitar/recuperar
Local	Local	Apenas a transação corrente	Registro de log/control de concorrência
Global	De sistema (soft crash)	Todas as transações em andamento	Registro de log
Global	De meio físico (hard crash)	Todas as transações em andamento	Backup

Transações – Propriedades ACID

- Atomacidade
- Consistência
- Isolamento
- Durabilidade

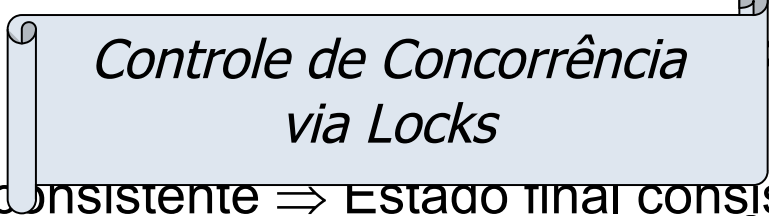
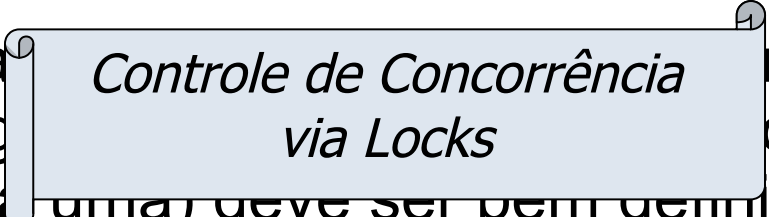
Transações – Propriedades ACID

- **Atomicidade:** todas as operações de uma transação devem ser efetivadas; ou, na ocorrência de uma falha, nada deve ser efetivado
 - “**tudo ou nada**” – não se admite parte de uma operação
- **Consistência:** transações preservam a consistência da base
 - Estado inicial consistente \Rightarrow Estado final consistente
- **Isolamento:** a maneira como várias transações em paralelo interagem (o que pode ser lido e o que pode ser escrito por cada uma) deve ser bem definido
- **Durabilidade:** uma vez consolidada (committed) a transação, suas alterações permanecem no banco até que outras transações aconteçam

Transações – Propriedades ACID

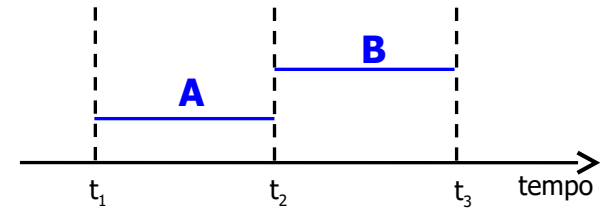
- **Atomacidade:** uma transação devem ser efetuada ou nada deve ser efetivado
Recuperação de falhas via log de uma falha,
 - “**tudo ou nada**” – não se admite parte de uma operação
- **Consistência:** transações preservam a consistência da base
 - Estado inicial consistente \Rightarrow Estado final consistente
- **Isolamento:** a maneira como várias transações em paralelo interagem (o que pode ser lido e o que pode ser escrito por cada uma) deve ser bem definido
- **Durabilidade:** (committed) a transação, sua recuperação via log no banco até que outras transações aconteçam

Transações – Propriedades ACID

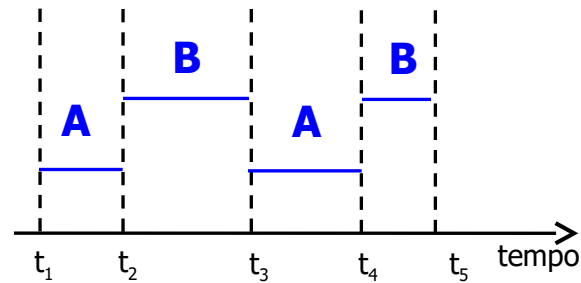
- **Atomacidade:** todas as operações de uma transação devem ser efetivadas; ou, na ocorrência de uma falha, nada deve ser efetivado
 - “**tudo ou nada**” – não se admite parte de uma operação
- **Consistência:** base
 - Estado inicial consistente \Rightarrow Estado final consistente
- **Isolamento:** transações em paralelo interagem de modo que o que pode ser escrito por cada uma, deve ser bem definido
- **Durabilidade:** uma vez consolidada (committed) a transação, suas alterações permanecem no banco até que outras transações aconteçam

Controle de Concorrência

- **Execução Serial (sequencial):**



- **Execução Intercalada:**



Controle de Concorrência

- **Execução Serial (sequencial):** diversas transações executadas em sequência
 - deixa a base de dados em estado correto e consistente
- **Execução Intercalada:** comandos de diversas transações são intercalados
 - pode levar a inconsistências

	Isolamento	Concorrência	Chance de inconsistências
Serial			
Intercalada			

Execução Serial X Intercalada

- **Execução serial**

- estado inicial correto e consistente \Rightarrow estado final correto e consistente

Execução Serial X Intercalada

- **Execução Intercalada**

- Toda execução serial é consistente
- Mas uma execução intercalada só é **consistente se for igual ao resultado de uma execução em sequência (em ordem conhecida)**
 - esta execução é dita **serializável**

Problemas de Execução Intercalada

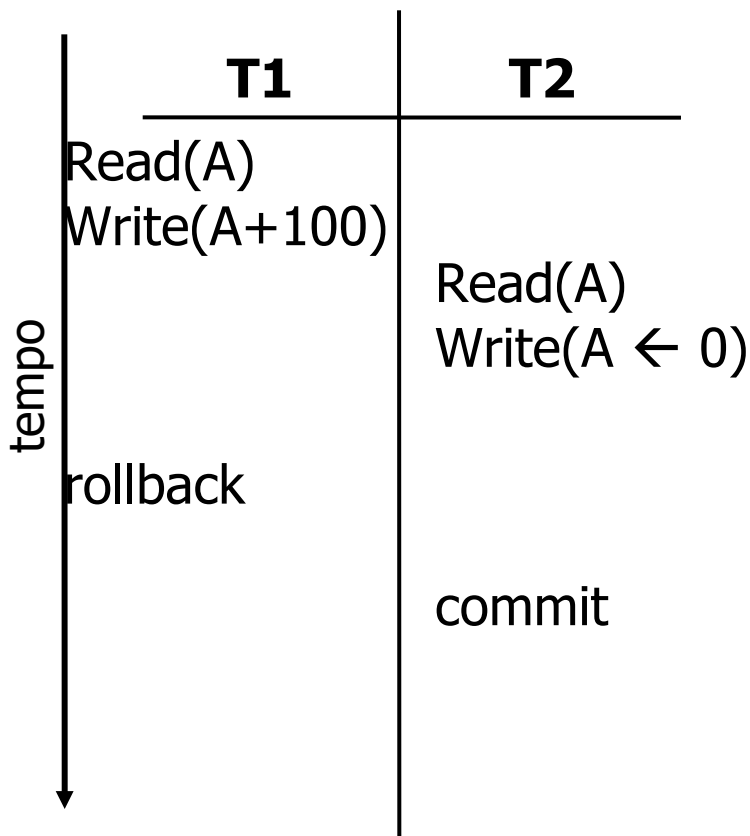
- Ocorrência de anomalias
 1. leitura inválida
 2. leitura não repetível
 3. leitura fantasma

Problemas de Execução Intercalada

- 1) **Leitura inválida (*Dirty Read*):**
 - transação T escreve um dado
 - transação T' lê o mesmo dado, mesmo antes do término de T;
- permite que outras transações possam ver os dados que ainda não foram consolidados (committed), isto é, mudanças que podem ser descartadas em seguida, por causa de uma instrução ROLLBACK por exemplo.

Problemas de Execução Intercalada

■ Ex: Leitura inválida (*Dirty Read*):

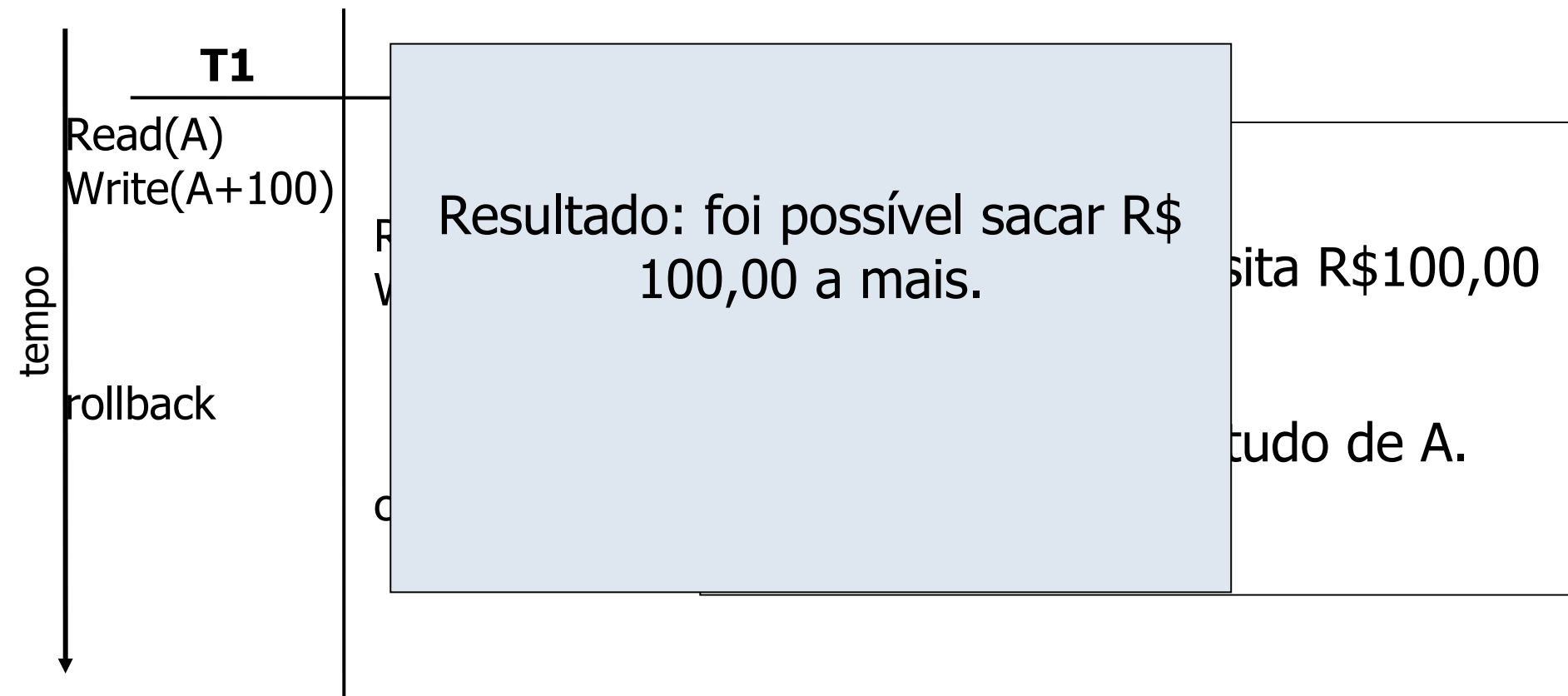


Exemplo 1:

- **Transação T_1** : deposita R\$100,00 na conta A.
- **Transação T_2** : saca tudo de A.
- T_1 é cancelada

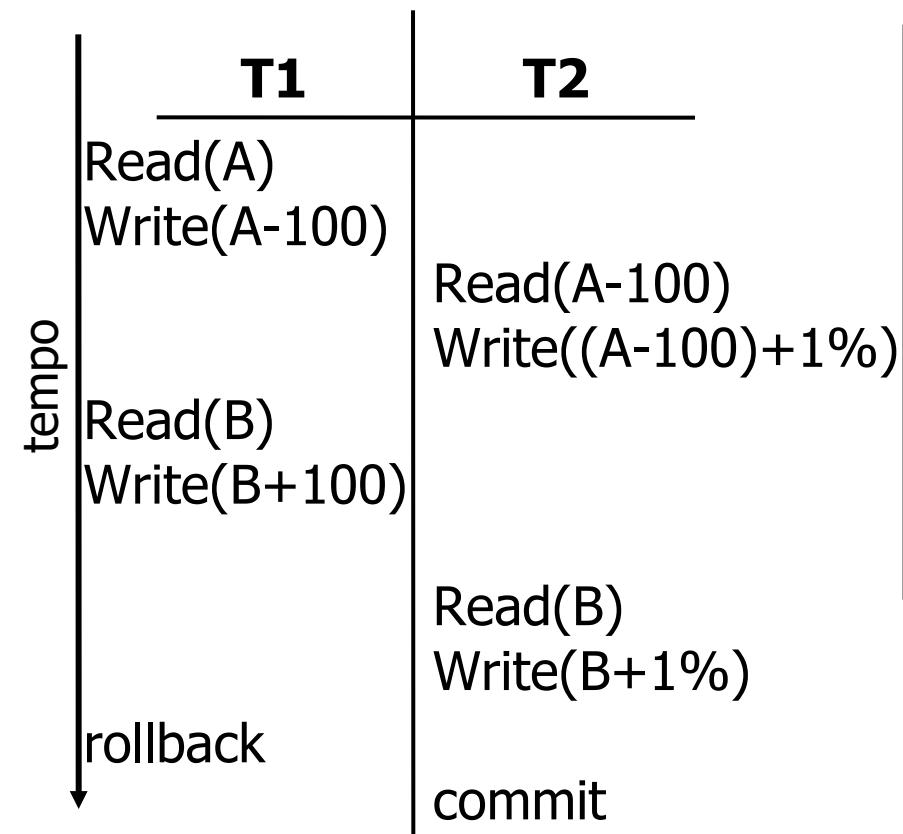
Problemas de Execução Intercalada

■ Ex: Leitura inválida (*Dirty Read*):



Problemas de Execução Intercalada

■ Ex: Leitura inválida (*Dirty Read*):

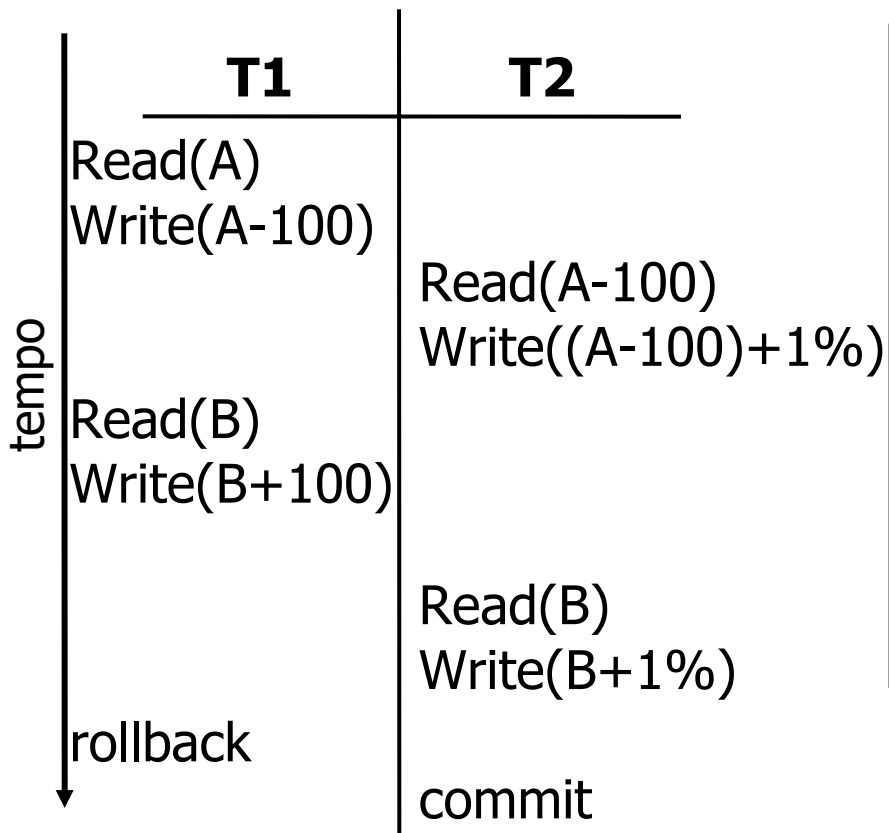


Exemplo 2:

- **Transação T_1** : transfere R\$100,00 da conta A para a conta B.
- **Transação T_2** : incrementa A e B em 1% (juros).

Problemas de Execução Intercalada

■ Ex: Leitura inválida (*Dirty Read*):



Exemplo 2:

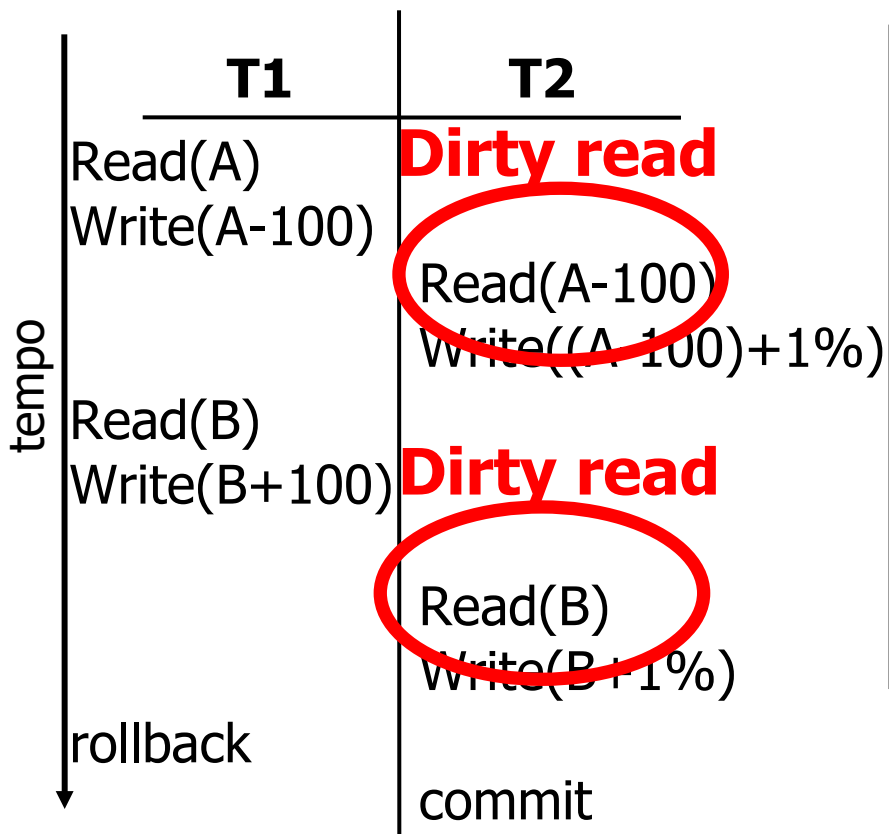
Supondo que inicialmente $A = 500$ e $B = 600$

- resultado esperado: T1 (rollback) seguido de T2 é $A = 505$ ($A+1\%$) e $B = 606$ ($B+1\%$)

- no entanto, com T1 e T2 em paralelo, o que se tem é: $A = 404$ ($A - 100 + 1\%$) e $B = 707$ ($B+100 + 1\%$)

Problemas de Execução Intercalada

■ Ex: Leitura inválida (*Dirty Read*):



Exemplo 2:

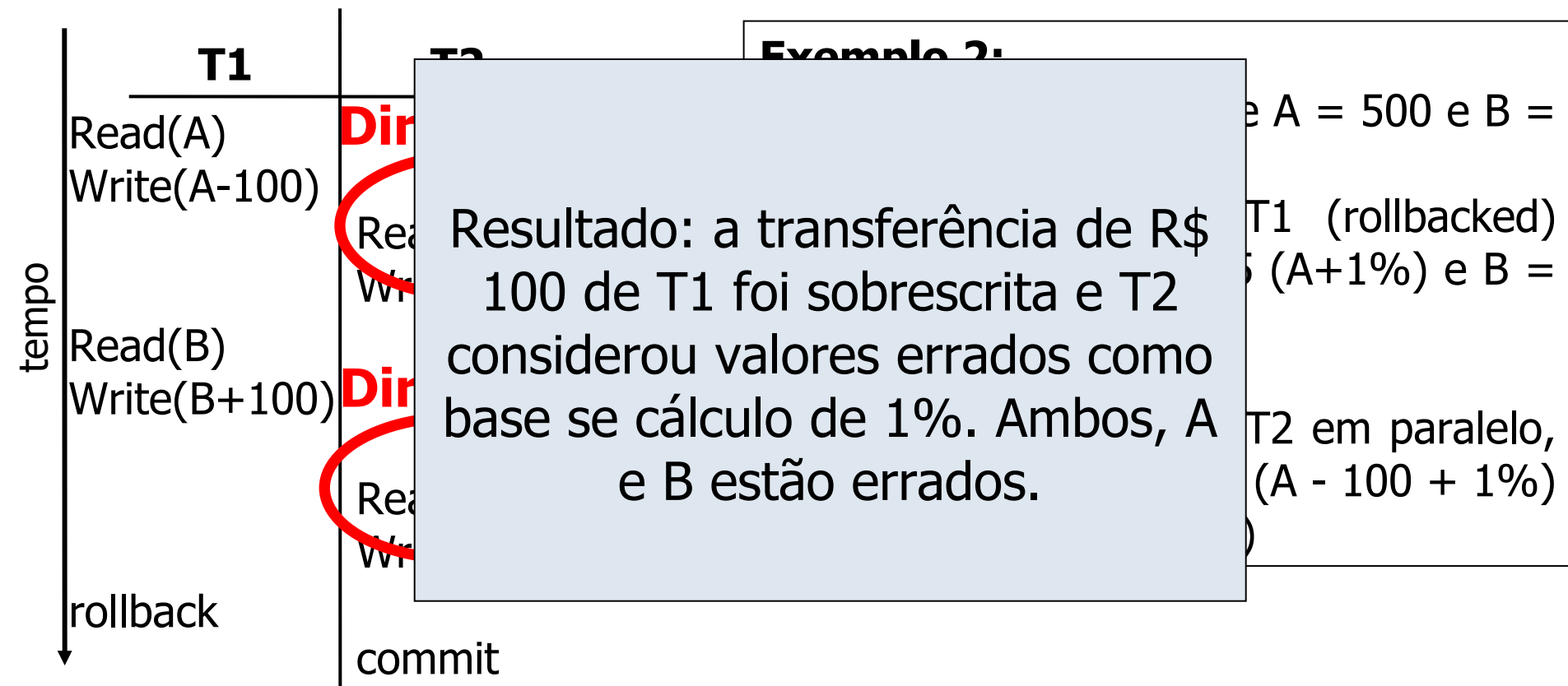
Supõe que inicialmente $A = 500$ e $B = 600$

- resultado esperado: T1 (rollback) seguido de T2 é $A = 505$ ($A+1\%$) e $B = 606$ ($B+1\%$)

- no entanto, com T1 e T2 em paralelo, o que se tem é: $A = 404$ ($A - 100 + 1\%$) e $B = 707$ ($B+100 + 1\%$)

Problemas de Execução Intercalada

■ Ex: Leitura inválida (*Dirty Read*):

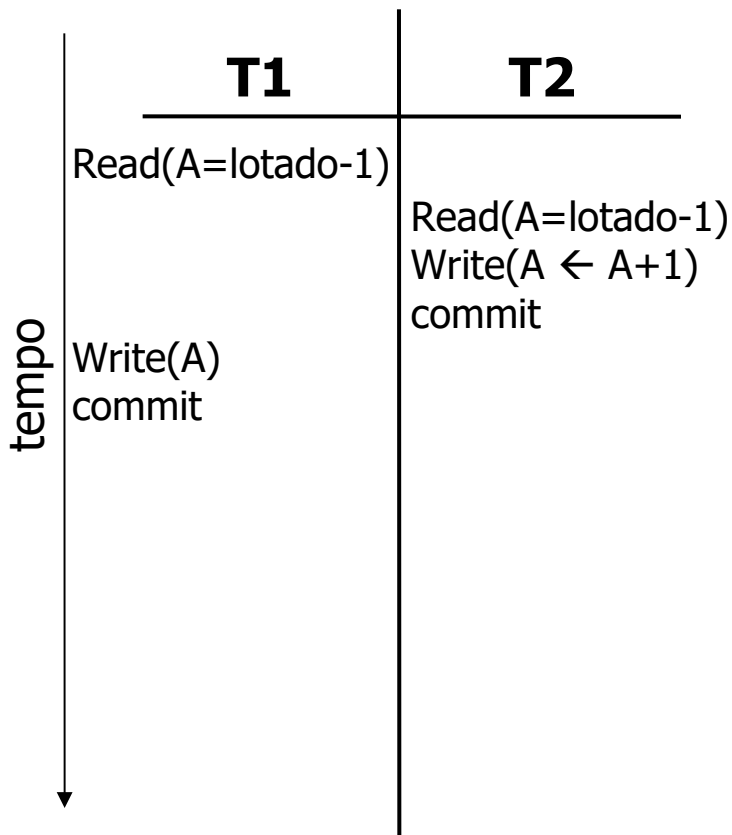


Problemas de Execução Intercalada

- **2) Leitura não repetível (*Nonrepeatable Read*):**
 - transação T lê um dado
 - esse dado é modificado por uma transação T' que começou depois de T
 - T' é efetivada
 - se T tentar reler o mesmo dado, obterá valores diferentes (*nonrepeatable read*)

Problemas de Execução Intercalada

■ Ex: Leitura não repetível (*Nonrepeatable Read*):

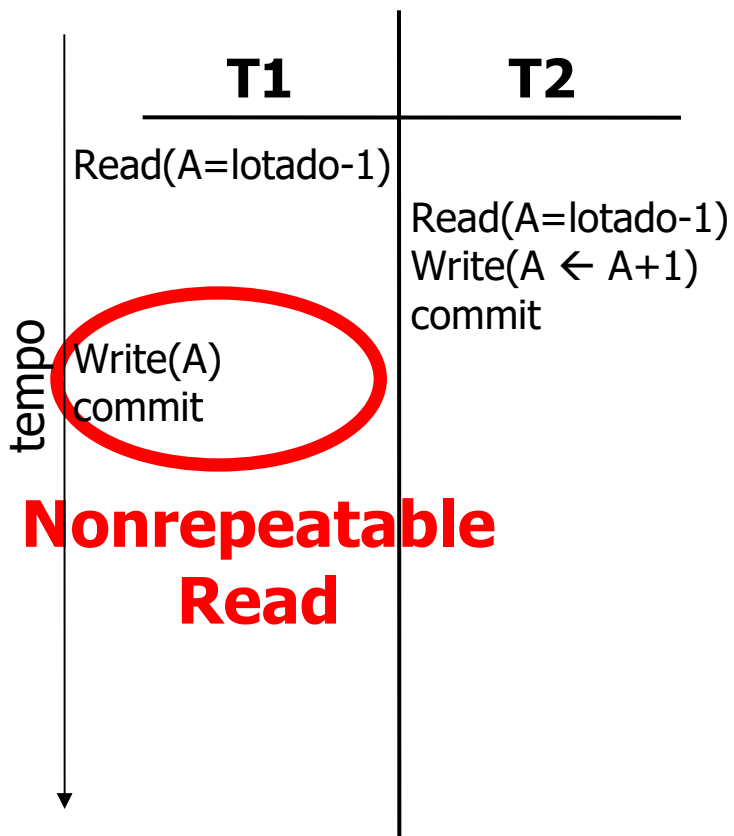


Exemplo:

- **Transação T_1** : lê reservas de um voo e verifica que há apenas um lugar disponível.
- **Transação T_2** : lê a mesma coisa
- **T_2** reserva o último lugar e é efetivada.
- **T_2** tenta reservar o lugar e ocorre um erro.

Problemas de Execução Intercalada

■ Ex: Leitura não repetível (*Nonrepeatable Read*):

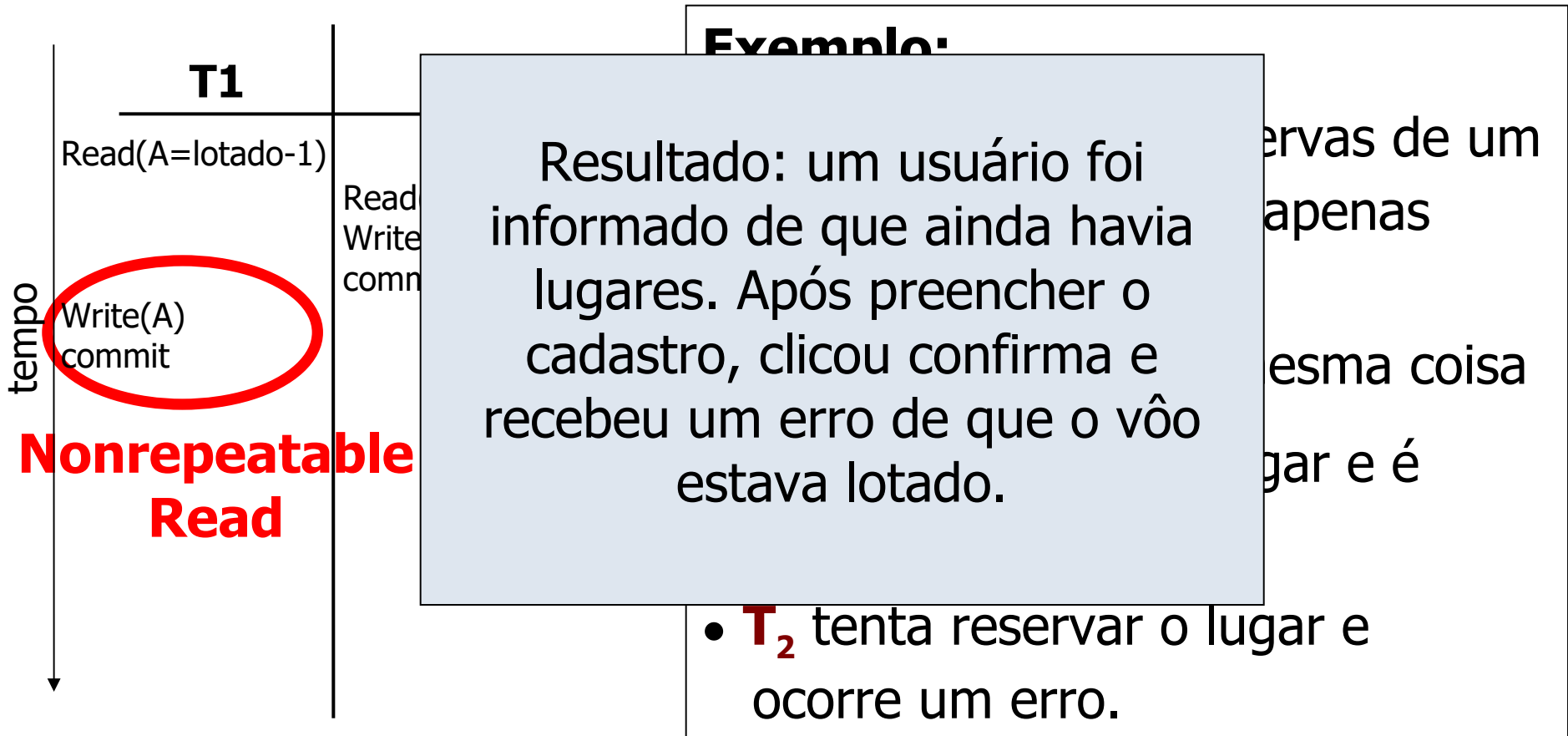


Exemplo:

- **Transação T_1** : lê reservas de um voo e verifica que há apenas um lugar disponível.
- **Transação T_2** : lê a mesma coisa
- T_2 reserva o último lugar e é efetivada.
- T_2 tenta reservar o lugar e ocorre um erro.

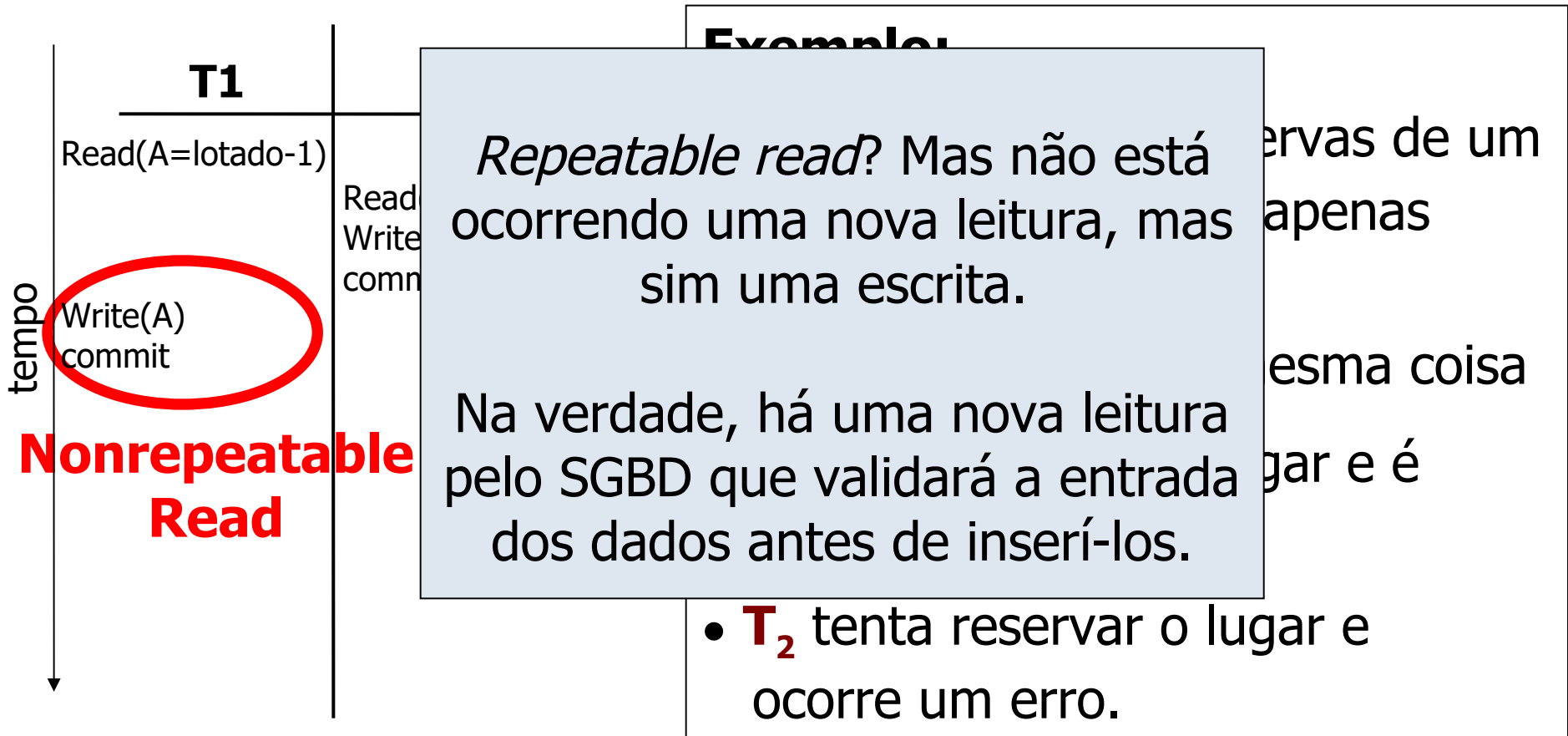
Problemas de Execução Intercalada

■ Ex: Leitura não repetível (*Nonrepeatable Read*):



Problemas de Execução Intercalada

■ Ex: Leitura não repetível (*Nonrepeatable Read*):

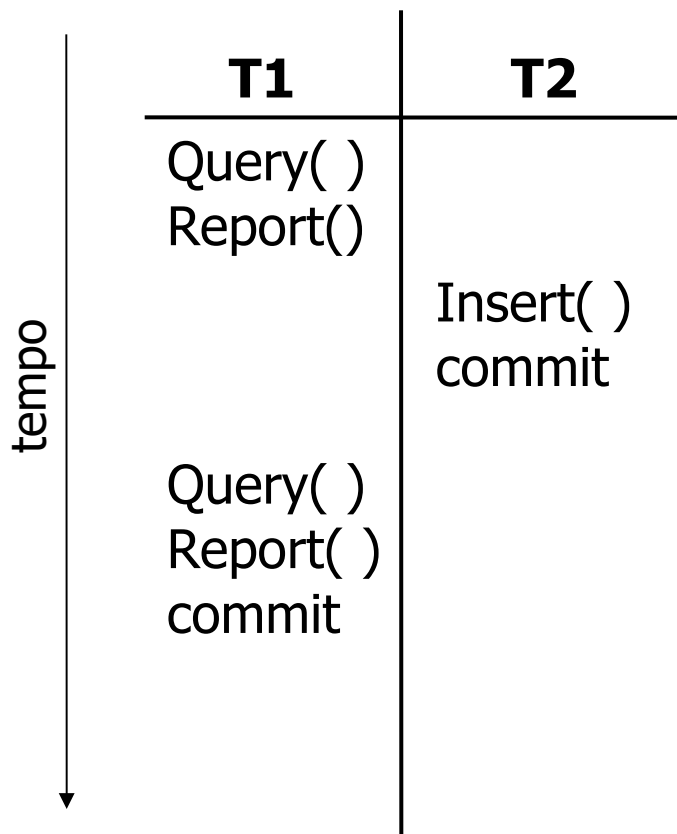


Problemas de Execução Intercalada

- **3) Leitura fantasma (*Phantom Read*):**
 - transação T lê um conjunto de tuplas que atendam a uma condição de consulta
 - transação T' **insere/remove/atualiza** uma tupla que atenderia a essa condição e é efetivada
 - se T refizer a mesma consulta, obterá um conjunto diferente de tuplas (*phantom read*)

Problemas de Execução Intercalada

■ Ex: Leitura fantasma (*Phantom Read*):



Exemplo:

- **Transação T_1** : faz uma consulta que retorna a média geral dos alunos que têm média ponderada acima de 5.0, e gera um relatório
 - **Transação T_2** : insere novos alunos com novas notas e é efetivada
 - **T_1** refaz a consulta para gerar relatório com nro de alunos por faixa de média
- ⇒ **relatórios inconsistentes.**

Problemas de Execução Intercalada

■ Ex

Repeatable read vs Phantom read

→ *Repeatable read*: lê valores diferentes de **um** mesmo dado que ainda está lá, mas foi alterado

→ *Phantom read*: lê **conjuntos** de dados diferentes, sendo que um dos conjuntos possui dados que não existem no(s) outro(s) conjunto(s) – fantasmas.

relatório com nro de alunos por faixa de média

⇒ **relatórios inconsistentes.**

Problemas de Execução Intercalada → Isolamento

- Ocorrência de anomalias

1. leitura inválida: leu um dado errado, o que causou uma operação inconsistente;
1. leitura não repetível: tentou ler um dado que foi alterado, impedindo que uma operação consistente fosse concluída;
1. leitura fantasma: teve alterado o conjunto de tuplas envolvidas em uma seleção – fazendo com que uma operação tenha resultados diferentes em momentos diferentes da transação

Problemas de Execução Intercalada → Isolamento

Um mesmo dado é **escrito e lido** em paralelo.

Solução: bloquear o dado que foi escrito, mas que ainda não foi consolidado, até que a transação de escrita termine.

Evita leitura inválida – bloqueia os dados depois de sua primeira escrita, uma transação só vê apenas os dados committed antes de seu início.

Um mesmo dado **lido e escrito** em paralelo.

Solução: bloquear o dado que tenha sido lido até que a transação de leitura termine.

Evita leitura inválida e leitura não repetível – bloqueia o dado depois de sua primeira operação de leitura ou escrita.

Não há concorrência por um mesmo dado específico – o que ocorre é que o **conjunto de dados envolvidos varia de maneira imprevisível.**

Solução: bloquear conjuntos de tuplas inteiros até que a transação termine.

Evita leitura inválida, leitura não repetível, e leitura fantasma – bloqueia todos os dados lidos por uma transação, outras transações são impedidas caso suas operações intercalem com os dados da transação sendo feita.

Transações – Propriedades ACID

- **Atomacidade:** todas as operações de uma transação devem ser efetivadas; ou, na ocorrência de uma falha, nada deve ser efetivado
 - “**tudo ou nada**” – não se admite parte de uma operação
- **Consistência:** transações preservam a consistência da base
 - Estado inicial consistente \Rightarrow Estado final consistente
- **Isolamento:** a maneira como várias transações em paralelo interagem (o que pode ser lido e o que pode ser escrito por cada uma) deve ser bem definido
- **Durabilidade:** uma vez consolidada (committed) a transação, suas alterações permanecem no banco até que outras transações aconteçam

Problemas de Execução Intercalada → Isolamento

- Ocorrência de anomalias
 1. leitura inválida
 2. leitura não repetível
 3. leitura fantasma
- Solução via isolamento em diferentes graus
 - Read uncommitted
 - Read committed
 - Repeatable read
 - Serializable

Interpretação

Nível de isolamento	Anomalias que PODEM ocorrer		
	1) Leitura inválida	2) Leitura não repetível	3) Leitura fantasma
Leitura mesmo do que NÃO FOI committed	Sim	Sim	Sim
Leitura apenas do que FOI committed	Não	Sim	Sim
Leitura apenas se a leitura repetida for garantida	Não	Não	Sim
Torna a execução equivalente à execução em série	Não	Não	Não

Transações – Propriedades ACID

- Tanto o PostgreSQL quanto o Oracle não implementam todas os quatro níveis de isolamento previstos pelo padrão SQL
- Em Oracle, apenas os níveis READ COMMITTED e SERIALIZABLE são aceitos; em PostgreSQL, todos os quatro níveis podem ser enunciados, no entanto, o READ UNCOMMITTED opera da mesma maneira que o READ COMMITTED e o REPEATABLE READ opera com as mesmas restrições do SERIALIZABLE.

Como implementar os níveis de isolamento?

➔ **Locks:** ou travas; uma transação T determina que se uma outra transação T' precisar daquele dado – para escrita ou leitura – ela deverá esperar

Isolamento \ Travas	Lock de escrita (write lock)	Lock de leitura (read lock)	Lock de leitura de conjunto (range lock)
Read uncommitted	IGNORA	IGNORA	IGNORA
Read committed	CONSIDERA	IGNORA	IGNORA
Repeatable read	CONSIDERA	CONSIDERA	IGNORA
Serializable	CONSIDERA	CONSIDERA	CONSIDERA

- *Lock de escrita:* se eu escrevo, ninguém lê até eu terminar
- *Lock de leitura (uma tupla):* se eu leio (ou escrevo) essa tupla, ninguém atualiza ela até eu terminar
- *Lock de leitura de conjunto de tuplas:* se eu leio esse conjunto de tuplas, ninguém atualiza o banco de dados de maneira que o conjunto determinado pelo predicado do conjunto seja afetado (incluindo os elementos e seus valores)

Transações – Propriedades ACID

- Considerando que os níveis de isolamento são conseguidos por meio do manuseio distinto dos locks, pode-se supor que o Oracle e o PostgreSQL tratam os locks de escrita como range locks de conjunto unitário, o que produz o mesmo efeito, a não ser pelo fato de que não é possível ter-se o nível de isolamento REPEATABLE READ assim como ele é previsto em teoria
- Consequentemente, não é possível demonstrar nem em Oracle, nem em PostgreSQL, o nível de isolamento REPEATABLE READ

Níveis de Isolamento em SQL99

Nível de isolamento	Anomalias que PODEM ocorrer		
	1) Leitura inválida	2) Leitura não repetível	3) Leitura fantasma
Read uncommitted	Sim	Sim	Sim
Read committed	Não	Sim	Sim
Repeatable read	Não	Não	Sim
Serializable	Não	Não	Não

Exercício

- Considere as transações T1 e T2, executadas sobre os itens de dados X e Y

T1

1.1)Read(X)
1.2)Read(Y)
1.3)Write(X)
1.4)commit

T2

2.1)Read(X)
2.2)Read(Y)
2.3)Write(X)
2.4)Write(y)
2.5)commit

- 1) Dê um exemplo de execução intercalada que resulte em uma anomalia de leitura inválida e explique o porquê.
- 1) Dê um exemplo de execução intercalada que resulte em uma anomalia de leitura não repetível e explique o porquê.

Exercício

- Considere as transações T1 e T2, executadas sobre os itens de dados X e Y

T1

1.1)Read(X)
1.2)Read(Y)
1.3)Write(X)
1.4)commit

T2

2.1)Read(X)
2.2)Read(Y)
2.3)Write(X)
2.4)Write(y)
2.5)commit

1) Dê um exemplo de execução intercalada que resulte em uma anomalia de leitura inválida e explique o porquê.

R.: 2.1,2.2,2.3,**1.1**,1.2,1.3,2.4,1.4,2.5

Aqui, T2 leu o valor de X e escreveu o valor de X (2.3), e antes de consolidar este valor, T1 leu o valor de X (**dirty read em 1.1**), e o reescreveu em 1.3. Em seguida T1 consolida os dados (1.4), seguido da consolidação dos dados por T2 (2.5). Qual é o valor de X, o que foi calculado por T2, ou o que foi calculado por T1?

Exercício

- Considere as transações T1 e T2, executadas sobre os itens de dados X e Y

T1

1.1)Read(X)
1.2)Read(Y)
1.3)Write(X)
1.4)commit

T2

2.1)Read(X)
2.2)Read(Y)
2.3)Write(X)
2.4)Write(y)
2.5)commit

1) Dê um exemplo de execução intercalada que resulte em uma anomalia de leitura inválida e explique o porquê.

R.: exemplo, em T2 alguém transfere 100 de X para Y, situação em que X deve ser $X-100$. antes que T2 termine no entanto, alguém lê o saldo de $X-100$ e deposita mais 100. quando T1 terminar, T1 irá consolidar $(X-100)+100$, ao passo que T2 irá consolidar $X-100$, anulando o que foi feito por T1.

Exercício

- Considere as transações T1 e T2, executadas sobre os itens de dados X e Y

T1

1.1)Read(X)
1.2)Read(Y)
1.3)Write(X)
1.4)commit

T2

2.1)Read(X)
2.2)Read(Y)
2.3)Write(X)
2.4)Write(y)
2.5)commit

2) Dê um exemplo de execução intercalada que resulte em uma anomalia de leitura não repetível e explique o porquê.

R.: 1.1, 2.1, 2.2, 2.3, 2.4, 2.5, 1.2, 1.3, 1.4

Aqui T1 lê o valor de X, o que em seguida é alterado por T2 (2.3 e 2.5). T1 então tenta atualizar o valor de X (1.3), o qual, por razão de uma restrição de domínio que fará uma nova leitura por exemplo, agora não aceita a atualização inicialmente pretendida por T1.

Sobre os níveis de isolamento

- Os níveis de isolamento, apresentam duas perspectivas: com relação à leitura e com relação à escrita.
 - Com relação à leitura, os mecanismos de isolamento resolvem os problemas sem restrições às atividades dos usuários
 - Com relação à escrita, os problemas não são resolvidos sem que se imponham restrições aos usuários

Isolamento	Leitura	Escrita sem concorrência	Escrita com concorrência
Read committed	Antes de cada operação , vê apenas suas alterações e as alterações que foram committed por outras transações - ignora o que está “em edição”	Escreve qualquer dado	Aguarda o término de outras transações para prosseguir
Serializable	Vê apenas suas alterações e as alterações que foram committed por outras transações antes do início da transação, ignorando tudo o que for alterado após seu início	Escreve qualquer dado	Não pode escrever em dados que foram committed por outras transações após seu início. ERRO, pois como a transação não vê o que foi alterado, não há como garantir que suas alterações concorrentes sejam consistentes

Isolamento	Sobre o uso
Read committed	<p>Seu uso é uma necessidade mínima de controle de concorrência, sendo que bancos de dados comerciais não aceitam níveis de isolamento abaixo de “read committed”.</p>
Serializable	<p>É recomendado em transações que farão intensa leitura de dados, o quê, ao longo de um período de tempo, poderia gerar anomalias pois dados lidos em um momento seriam diferentes de dados lidos posteriormente. Para transações com intensa atividade de escrita, o “read committed” é recomendado, pois como o serializable trabalha com um snapshot do banco criado no início da transação, escritas concorrentes não são gerenciáveis e, por consequência, não são permitidas. Deve-se considerar também, que transações Serializable são mais custosas e podem causar problemas de desempenho; não se deve usar serializable sem necessidade.</p>

Isolamento

Sobre o uso

Read committed

Seu uso é uma necessidade mínima de controle de concorrência, sendo

Além disso, há restrições de processamento com relação ao uso – o isolamento serializable requer mais recursos devido ao uso intenso de locks de escrita e leitura, portanto não deve ser usado desnecessariamente.

Serializable

gerenciáveis e, por consequência, não são permitidas. Deve-se considerar também, que transações Serializable são mais custosas e podem causar problemas de desempenho; não se deve usar serializable sem necessidade.

Transações em PostgreSQL/Oracle

Transação em PostgreSQL

- Comando **SET TRANSACTION**

START TRANSACTION

SET TRANSACTION ISOLATION LEVEL

**{ SERIALIZABLE | REPEATABLE READ | READ
COMMITTED | READ UNCOMMITTED }**

READ WRITE | READ ONLY

Níveis de Isolamento de SQL99 no PostgreSQL

Nível de isolamento	Anomalias que podem ocorrer		
	Leitura inválida	Leitura não repetível	Leitura fantasma
Read uncommitted	Não tem efeito → read committed.		
Read committed (padrão)	Não	Sim	Sim
Repeatable read	Não tem efeito → serializable.		
Serializable	Não	Não	Não

Níveis de Isolamento de SQL99 no PostgreSQL

A versão 9.1 do PostgreSQL passou a suportar níveis de isolamento "repeatable read" e "serializable", agora distintos em questões de funcionamento. Segundo a documentação do PostgreSQL 9.1, o nível "repeatable read" implementado é mais restritivo do que o que é previsto pelo padrão SQL, funcionando exatamente da mesma maneira que o nível serializable, apenas com diferenças de performance e restrições de atualização concorrente ainda mais restritas.

<http://www.postgresql.org/docs/9.1/static/transaction-iso.html#XACT-REPEATABLE-READ>

Transação em ORACLE

- Comando **SET TRANSACTION**

SET TRANSACTION

READ ONLY | READ WRITE | ISOLATION LEVEL
{ SERIALIZABLE | READ
COMMITTED } |
USE ROLLBACK SEGMENT rollback_segment
NAME 'nome_da_transacao';

Níveis de Isolamento de SQL99 no ORACLE

Nível de isolamento	Anomalias que podem ocorrer		
	Leitura inválida	Leitura não repetível	Leitura fantasma
Read uncommitted	Nunca permitido em Oracle.		
Read committed (padrão)	Não	Sim	Sim
Repeatable read	Não suportado especificamente (abrangido por serializable).		
Serializable	Não	Não	Não

Transações em PostgreSQL/Oracle

- Além dos modos de isolamento, pode-se usar também os modos de read write ou read only:
 - Read write: é o padrão
 - Read only: previne que uma dada transação execute os comandos INSERT, UPDATE, DELETE, e COPY FROM

Transações em PostgreSQL/Oracle

- Em PostgreSQL, as transações se iniciam com `START TRANSACTION`
- Em Oracle se **iniciam** com a cláusula `SET TRANSACTION`
- E **terminam**:
 - explicitamente com `commit` ou `rollback`
 - implicitamente quando um processo de usuário é finalizado
 - com sucesso – ex: disconnect (`commit`)
 - sem sucesso – ex: falha de sistema (`rollback`)
 - execução de comando DDL
- Possuem quatro possibilidades:
 - **Modo de leitura**: para transações apenas leitura
 - `READ-ONLY`, e `READ-WRITE`
 - **Modo de isolamento**: para transações com atualizações
 - `READ committed`, e `SERIALIZABLE`

Transação em ORACLE

- Comando **COMMIT**
 - termina a transação
 - torna permanente as ações da transação
 - libera os recursos bloqueados

Transações em ORACLE

- Modo de isolamento: para transações com atualizações
 - READ committed (padrão):
 - antes de uma operação, a transação aguarda até que quaisquer tuplas sendo atualizadas sejam liberadas e prossegue
 - a transação “vê” apenas dados consolidados (committed) antes do início de uma dada **operação**
 - SERIALIZABLE:
 - caso uma tupla seja alterada após o início da transação serializable, ocorre um erro se a transação serializable tentar alterar esta mesma tupla:

ORA-08177: Can't serialize access for this transaction.
 - ou seja, o Oracle informa que não é capaz de tornar a concorrência semelhante a um processamento em série
 - a transação “vê” apenas dados modificados pela própria transação e dados efetivados antes do início da transação

Transações em ORACLE

- Modo de isolamento: para transações com atualizações

- Idéia de Snapshot dos dados antes de uma operação – vários snapshots do banco. plas

- a transação vê apenas dados consolidados (committed) antes do início de uma dada **operação**

- Idéia de Snapshot dos dados antes de uma transação inteira – um único snapshot. ble,
sma

ORA-08177: Can't serialize access for this transaction.

- ou seja, o Oracle informa que não é capaz de tornar a concorrência semelhante a um processamento em série
 - a transação “vê” apenas dados modificados pela própria transação e dados efetivados antes do início da transação

Transações e Controle de Concorrência

■ Referências

- *Oracle Database Concepts*
- *OracleSQL Reference*
- Elmasri e Navathe. *Fundamentals of Database Systems*