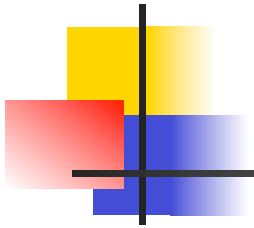


Banco de Dados II

Recuperação de Falhas

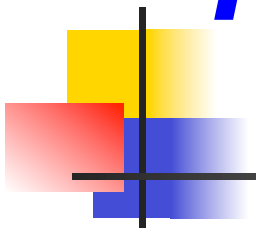
Recuperação de Falhas

Por quê é importante?

Quando uma transação é submetida ao SGBD, o sistema é responsável por garantir:

- **atomicidade**
- **durabilidade**

Recuperação de Falhas



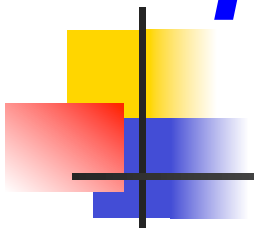
O SGBD deve possuir mecanismos para desfazer ou refazer operações de uma transação, quando o sistema se recupera de uma falha → restauração do BD a um estado consistente!!

Recuperação de Falhas

Tipos de Falhas

- **desastres naturais ou catástrofes**
  - **incêndio**
  - **enchentes**

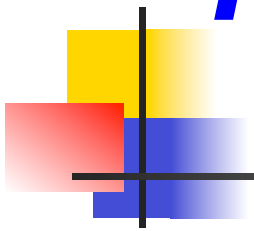
Recuperação de Falhas



Tipos de Falhas

- **queda do sistema**
- **mal funcionamento do hardware/software**

Recuperação de Falhas



Tipos de Falhas

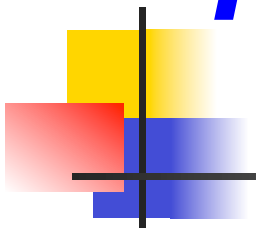
- **falhas no disco**
  - **quebra do cabeçote**
  - **falha durante transmissão dos dados**

Recuperação de Falhas

Tipos de Falhas

- **falhas na transação**
  - **Erro lógico** (entrada inadequada, dado não encontrado, overflow do sistema)
  - **Erro de sistema** (deadlock)

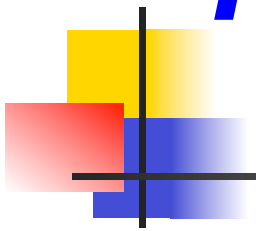
Recuperação de Falhas



Tipos de Armazenamento

- **Volátil**
- **Não – Volátil**
- **Estável**

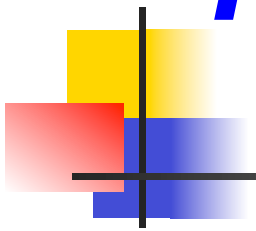
Recuperação de Falhas



Tipos de Armazenamento

- **Volátil:** Memória Principal / Cache
- **Não – Volátil:** Discos e fitas
- **Estável:** Sistemas RAID (Redundante Arrays of Inexpensive Disks)

Recuperação de Falhas

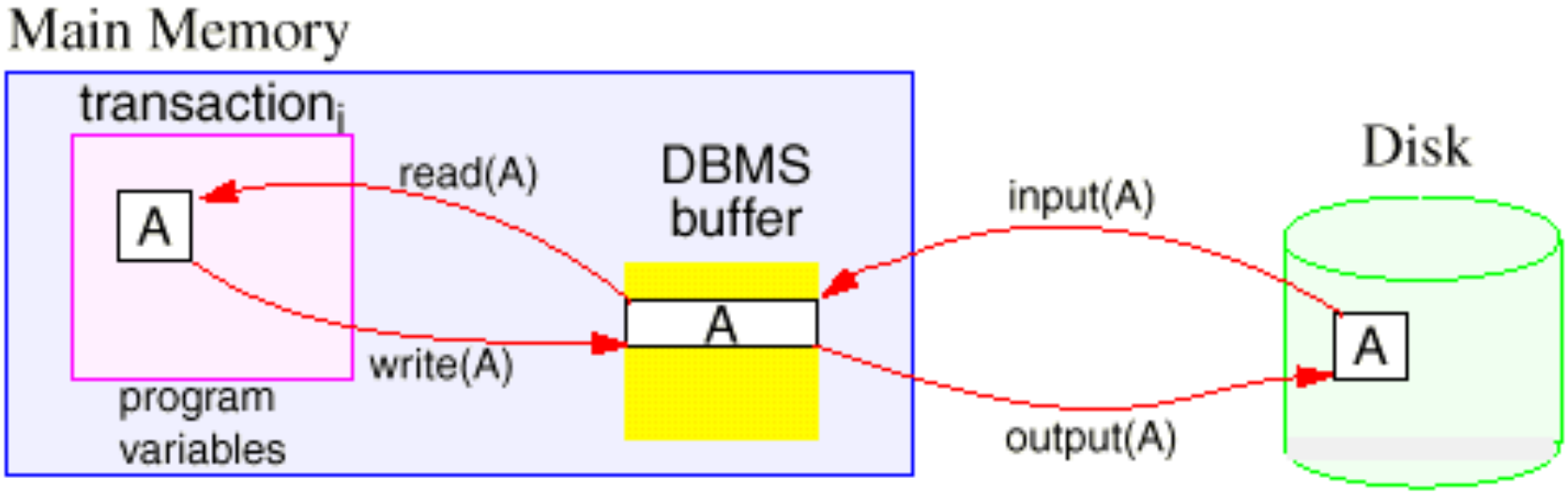


Implementação de Armazenamento Estável

- **Replicação Local dos Dados**
  - **Sistemas RAID (Discos Espelhados)**
- **Replicação Não-Local dos Dados**
  - **Discos Remotos**

Recuperação de Falhas

Transmissão de Dados entre Memória Principal e Memória Secundária



Recuperação de Falhas

Transmissão de Dados entre Memória Principal e Memória Secundária

Read (x): atribui à variável local x_i o valor do item de dados x

- Se o bloco B_x, no qual x reside não está no buffer, então é emitido um input(B_x);
- Designa à x_i o valor de x a partir do bloco do buffer.

Recuperação de Falhas

Transmissão de Dados entre Memória Principal e Memória Secundária

Write(x): atribui o valor da variável local x_i ao item de dados x do bloco do buffer

- Se o bloco B_x, no qual x reside não está no buffer, então é emitido um input(B_x);
- Designa o valor de x_i para x no bloco de buffer B_x.

Recuperação de Falhas

Transmissão de Dados entre Memória Principal e Memória Secundária

Output(B):

- Utilizado pelo SGBD para forçar saídas do buffer;
- Nem sempre é executado imediatamente após a operação write.
- Se o sistema cair depois de write(x) e antes de output(B_x), o novo valor de x é perdido, se não houver um mecanismo de recuperação.

Recuperação de Falhas

Arquivos LOG

Recuperação Baseada em LOG

LOG → seqüência de registros contendo informações sobre todas as atualizações efetuadas no Banco de Dados.

LOG → precisa residir em memória estável.

Recuperação de Falhas

Arquivos LOG

Tipos de Registros de LOG

$<T_i, \text{Start}>$ → início da transação T_i

$<T_i, X_j, V_{velho}, V_{novo}>$ → transação T_i executou operação write(X_j). O valor V_{novo} substituiu o valor V_{velho}.

Recuperação de Falhas

Arquivos LOG

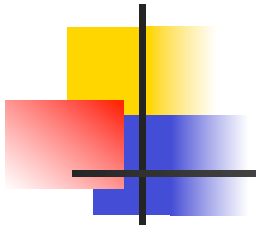
Tipos de Registros de LOG

<T_i, Commit> ➔ **transação T_i foi efetivada**

<T_i, Abort> ➔ **transação T_i foi abortada**

Recuperação de Falhas

Arquivos LOG



Registros de LOG

É necessário escrever os registros de LOG em memória estável antes do Banco de Dados ser fisicamente modificado!!

Recuperação de Falhas

Arquivos LOG

Operações de Recuperação

Redo(T_i): atribui a todos os itens de dados atualizados por T_i o valor V_{novo}

Undo(T_i): atribui a todos os itens de dados atualizados por T_i o valor V_{velho}

Recuperação de Falhas

Arquivos LOG

Exemplo

T0 (Transferência de Fundos)

read(A)

A = A – 50

write(A)

read(B)

B = B + 50

write(B)

Commit

T1 (Saque)

read(C)

C = C – 100

write(C)

Commit

Recuperação de Falhas

Arquivos LOG

Transação

T0 (Transferência de Fundos)

read(A)

A = A – 50

write(A)

read(B)

B = B + 50

write(B)

Commit

Arquivo Log

<T0 starts>

<T0, A, 1000, 950>

<T0, B, 2000, 2050>

<T0 commits>

Recuperação de Falhas

Arquivos LOG

Transação

T1 (Saque)

read(C)

C = C – 100

write(C)

Commit

Arquivo Log

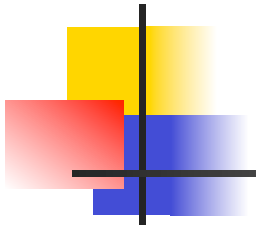
<T1 starts>

<T1, C, 700, 600>

<T1 commits>

Recuperação de Falhas

Arquivos LOG



Checkpoints

O Sistema executa periodicamente checkpoints que também são registrados no arquivo de LOG.

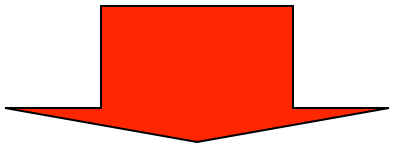
Checkpoints são importantes para evitar trabalho desnecessário durante a recuperação de falhas.

Recuperação de Falhas

Arquivos LOG

Checkpoints

Checkpoints evitam trabalho desnecessário durante a recuperação de falhas.



Transações efetivadas antes dos checkpoints não precisam ser refeitas.

Recuperação de Falhas

Arquivos LOG

Escritas Adiadas

- As escritas da transação são adiadas para o momento em que ela é efetivada
- Redo → aplicado na ordem cronológica dos commits.
- Undo → desnecessário.

Escritas Imediatas

- As escritas da transação são realizadas imediatamente
- Undo → aplicado na ordem cronológica inversa.
- Redo → aplicado na ordem cronológica.

Recuperação de Falhas

Arquivos LOG

Transações Concorrentes

- **2PL Severo ou Rigoroso**
- **Checkpoint com lista de transações ativas**

No arquivo de LOG registra-se:

<checkpoint L >

L = {lista das transações ativas no momento do checkpoint}

Recuperação de Falhas

Arquivos LOG

Transações Concorrentes

Construção das listas Undo e Redo

1. **Leia o arquivo LOG do final para o começo até encontrar o primeiro registro <checkpoint, L>**
2. **Para cada registro da forma <T_i,commits>, adicione T_i à lista Redo.**
3. **Para cada registro da forma <T_i, starts>, se $T_i \notin$ Redo, então adicione T_i à lista Undo;**
4. **Verifique a lista L. Se $T_i \in$ L e $T_i \notin$ Redo, adicione Ti à lista Undo.**

OBS: a lista de UNDO não é necessária para escritas adiadas

Recuperação de Falhas

Arquivos LOG

Transações Concorrentes

Procedimento de Recuperação (Escritas Adiadas)

1. **Localize o registro <Ti, Starts> da transação mais antiga, Ti, da lista Redo. Leia o arquivo LOG a partir deste registro até o final do arquivo e a cada registro <Tj, commits> encontrado, com Tj pertencendo à lista Redo, refaça todas as escritas feitas pela transação Tj na ordem cronológica.**

Recuperação de Falhas

Arquivos LOG

Transações Concorrentes

Procedimento de Recuperação (**Escritas Imediatas**)

1. Percorra o arquivo de LOG do fim para o início:
   - realizando UNDO(Ti) para todas as transações Ti existentes na lista-UNDO
   - marcando na lista-REDO as transações Ti cujos registros <Início Ti> estão sendo encontrados nessa varredura
2. caso todas as transações existentes na lista-UNDO tenham sido desfeitas e ficou alguma transação Ti não marcada na lista-REDO;
   - continua percorrendo o arquivo de LOG para trás até que todos os registros <Inicio Ti> das transações não marcadas na lista-REDO tenham sido encontradas
3. Percorre o arquivo de LOG para a frente, realizando REDO(Ti) para todas as transações existentes na lista-REDO.

OBS: se existir um registro de checkpoint, o REDO pode ser executado a partir dele, percorrendo o arquivo de LOG para frente.

Recuperação de Falhas

Bufferização dos Registros de LOG

Objetivo

Reduzir overhead (pois escrever os registros de log individualmente é bastante dispendioso)

Problema

Falhas podem causar a perda de registros de LOG, caso estes sejam bufferizados

Solução (write-ahead logging rule)

Mover antecipadamente para memória estável todos os registros de log referentes aos dados de um bloco que precisa ser transferido para o disco.