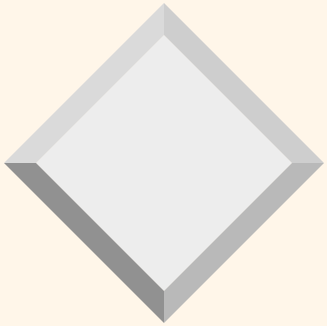


Recuperação de Falhas

Capítulo 18

Humpty sentou-se no muro de pedra
Humpty teve uma grande queda
Todos os cavaleiros e homens do reino
Não puderam juntá-lo de novo. - *velha rima de crianças*

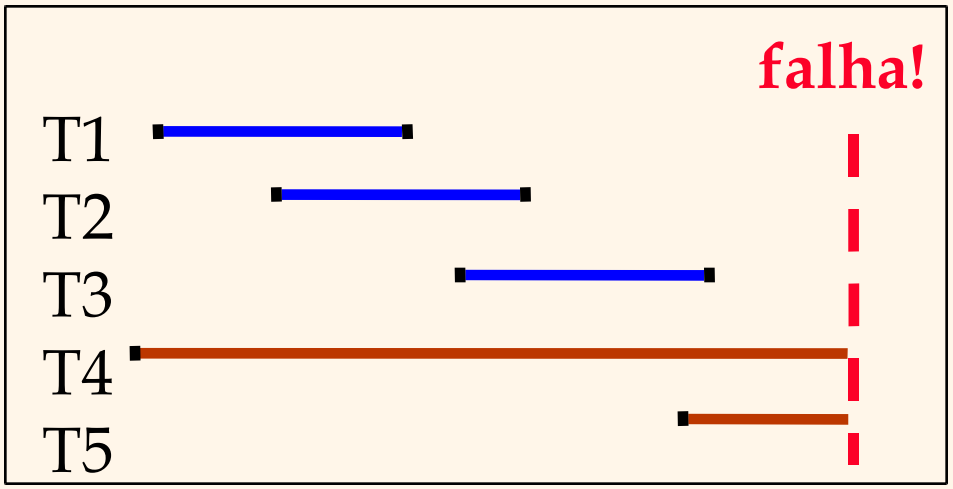
Revisão: As propriedades ACID

- **A** tomicidade: Todas as ações na transação acontecem, ou nenhuma acontece.
- **C** onsistência: Se toda a transação é consistente, e o BD inicia consistente, então o BD termina consistente.
- **I** solamento: A execução de uma transação é isolada de outras transações.
- **D** urabilidade: Se uma transação é finalizada, seu efeito persiste.

- O **Gerenciador de Recuperação** garante Atomicidade e Durabilidade.

Motivação

- Atomicidade:
  - Transações podem abortar.
- Durabilidade:
  - E se o SGBD pára de rodar? (Causas?)
- Comportamento desejado após o sistema reiniciar:
  - T1, T2 e T3 devem ser duráveis.
  - T4 e T5 deveriam ser desfeitas (efeitos não visualizados).



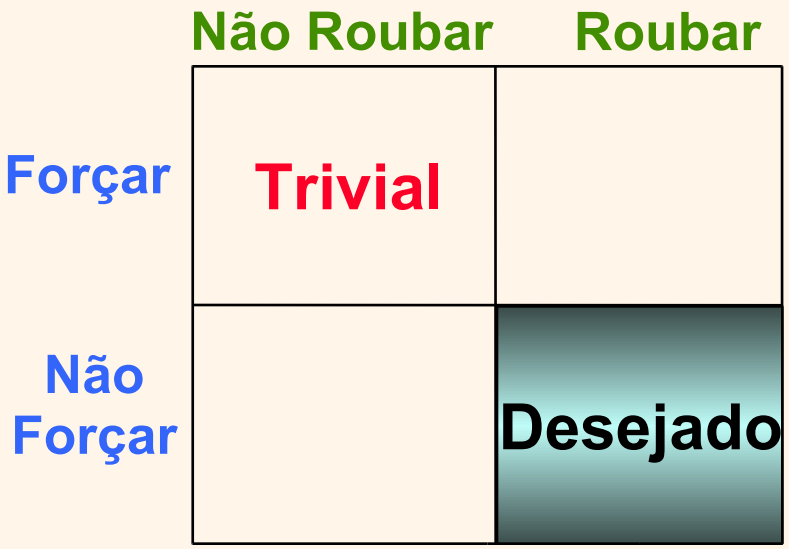
Premissas

- Controle da concorrência está em questão
  - Bloqueio em duas fases estrito (Strict 2PL), em particular.
- Atualizações ocorrem usando meio estável.
  - i.e. os dados são sobrescritos no (e removidos do) disco.

- Existe uma maneira simples para garantir a Atomicidade e Durabilidade?

Tratando a Área de Armazenamento

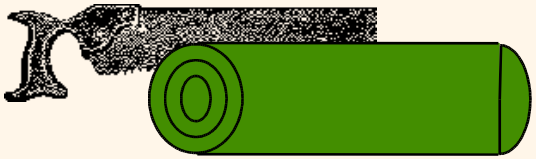
- Forçar todas escritas no disco?
  - Tempo de resposta insuficiente.
  - Porém, provê durabilidade.
- Roubar (Steal) frames da área de armazenamento na memória que não sofreram validação?
  - Se não, eficiência insatisfatória.
  - Se sim, como podemos garantir a atomicidade?



Mais sobre Forçar ou Roubar

- **ROUBAR** (Por que garantir Atomicidade é difícil?)
  - *Para roubar o frame F:* Página corrente em F (denominada P) é gravada no disco; alguma transação detém o bloqueio em P.
    - E se a transação com o bloqueio em P aborta?
    - Deve ser lembrado o valor anterior de P no momento do roubo (para permitir DESFAZER a escrita na página P).
- **NÃO FORÇAR** (Por que garantir Durabilidade é difícil?)
  - E se o sistema falha antes de uma página modificada ser gravada no disco?
  - Gravar o mínimo possível, em um local conveniente, no momento da validação *(commit)*, para permitir REFAZER as modificações.

Idéia Básica: Registrar



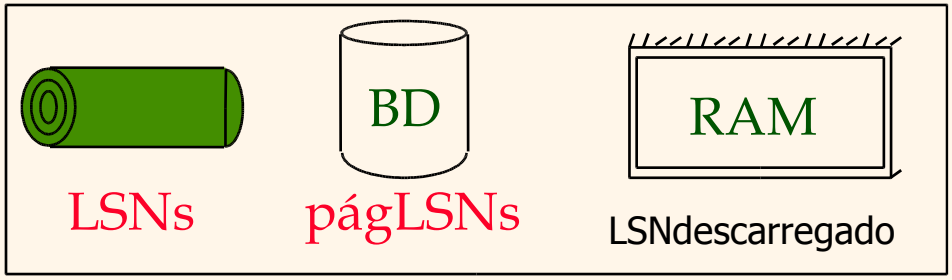
- Armazenar informações de REFAZER e DESFAZER, para cada atualização, em um *log*.
  - Escritas seqüenciais em um log (manter em um disco separado).
  - Mínimo de informações (diferenças) gravadas no log, tal que várias atualizações cabem em uma única página.
- Log: Lista ordenada de ações REFAZER/DESFAZER
  - Um registro do log contém:

    <ID_transação, ID_página, deslocamento, tamanho, dado_antigo, dado_novo>
  - E informações adicionais de controle (que veremos adiante).

Write-Ahead Logging (WAL)

- O protocolo Write-Ahead Logging (Escrita-Antecipada no Log):
  - ① Deve forçar a atualização no log *antes* que a página de dados correspondente seja gravada no disco.
  - ② Deve gravar no log todas as ações de uma transação *antes do commit*.
- #1 garante Atomicidade.
- #2 garante Durabilidade.

WAL e o Log



- Cada registro tem um único Número Sequencial de Log (LSN).
  - LSNs sempre aumentam.
- Cada *página de dados* contém um págLSN.
  - Guarda o LSN do *registro do log* mais recente para uma atualização nesta página.
- O sistema mantém controle do LSNdescarregado (flushedLSN).
  - O máx(LSN) gravado no disco até então.
- Cada Transação guarda (na tabela de transações) o últimoLSN
  - Último LSN executado da transação

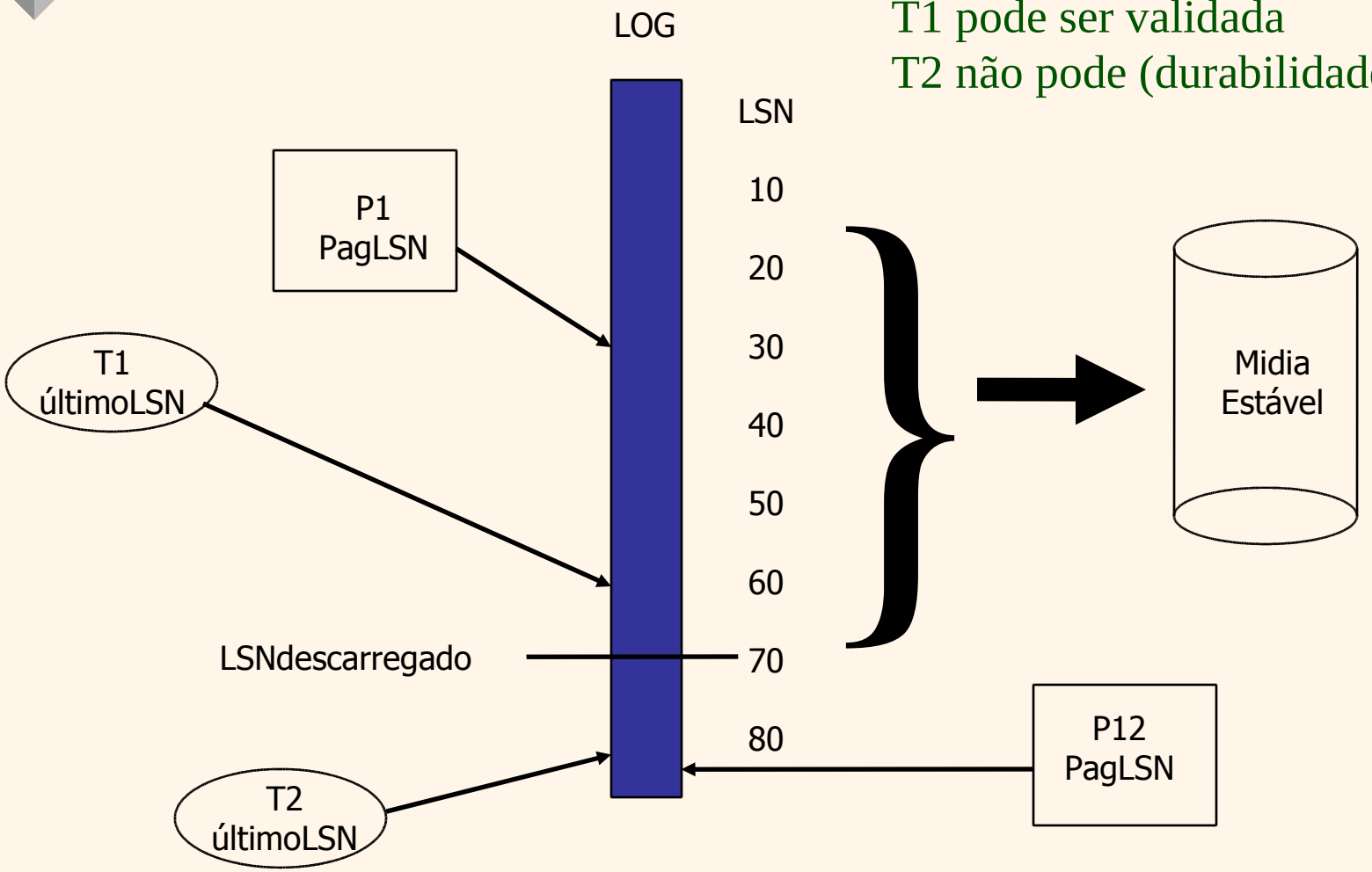
Wal e Log (Cont)

- WAL: *Antes* de uma página ser gravada:
  - págLSN ≤ LSNdescarregado
- *Antes* de uma transação ser validada *(commited)*
  - últimoLSN ≤ LSNdescarregado

Log e WAL (Cont)

P1 pode ser gravada em disco
P2 não pode (atomicidade)

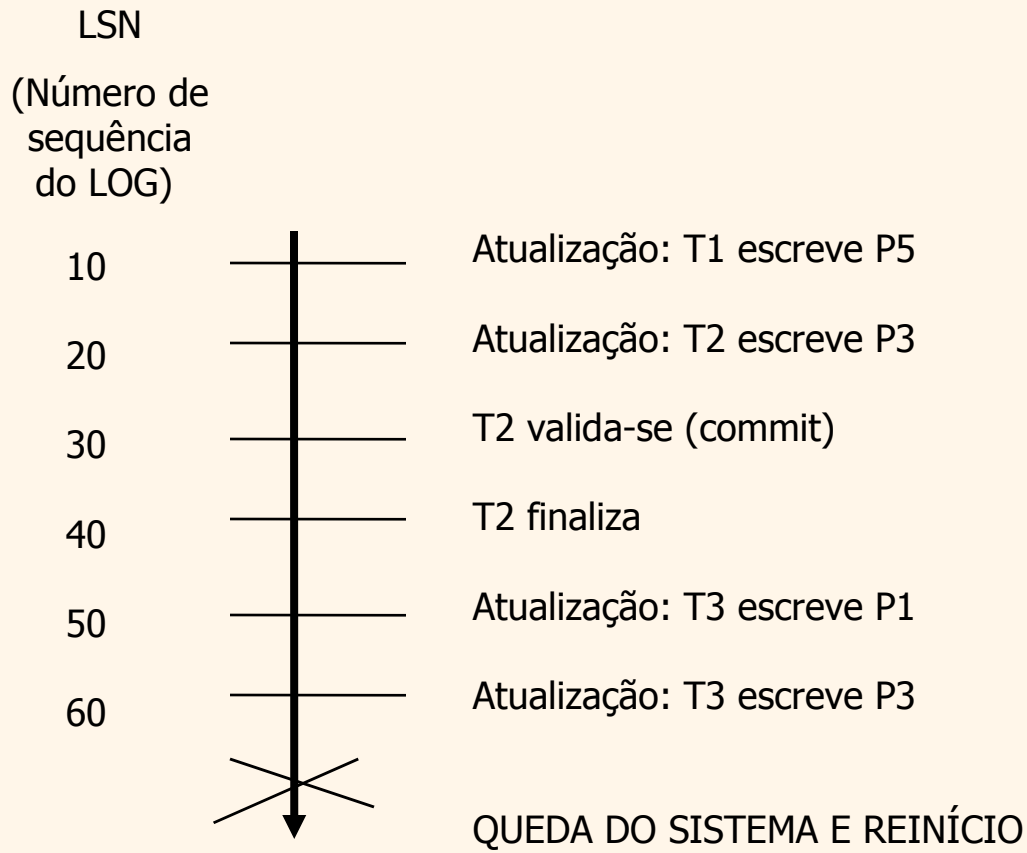
T1 pode ser validada
T2 não pode (durabilidade)



Introdução a ARIES

- ARIES é um algoritmo de recuperação
- Depois de uma queda do sistema
  - Analisa o estado do sistema
  - Refaz transações até a queda.
  - Desfaz as transações incompletas.

Exemplo de Atuação do ARIES



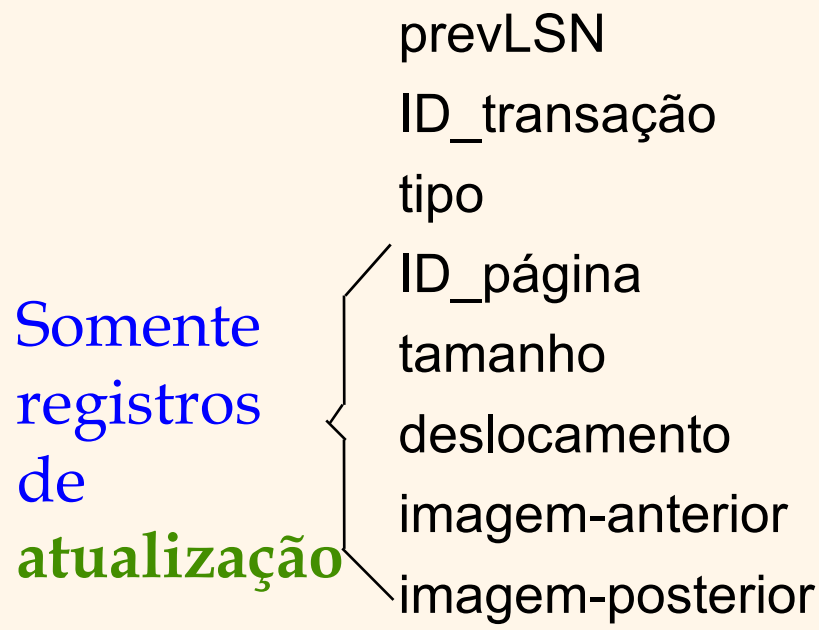
P1, P3 e P5: Páginas possivelmente sujas na queda do sistema

T1, T2 e T3 são refeitas na fase 'refazer" do algoritmo.

T1 e T3 são desfeitas na fase "desfazer" do algoritmo, na ordem Inversa de sua execução.

Registros do Log

Campos de um registro do Log:



Possíveis tipos de registros do log:

- **Atualização (Update)**
- **Validação (Commit)**
- **Desfazer (Abort)**
- **Fim** (significa o final de uma validação ou desfazer)
- Registros de Compensação (CLRs)
  - Para ações DESFAZER

Sobre Registro de Compensação

- CLR
  - Registros relativos a desfazer transações
  - Não podem ser desfeitos.
  - Não tem imagem anterior

Outras Estruturas Relacionadas ao Log

- Tabela de Transações:
  - Uma entrada por transação ativa.
  - Contém ID_transação, estado (executando/finalizada com sucesso (commit)/ desfeita), e últimoLSN.
- Tabela de Páginas 'Sujas':
  - Uma entrada por página suja da região de armazenamento.
  - Contém recLSN – o LSN do registro que tornou a página 'suja' ***primeiro***.

Execução Normal de uma Transação

- Seqüência de leituras e escritas, seguidas por uma validação ou desfazer.
  - Assumiremos que a escrita é atômica no disco.
    - Na prática, há detalhes adicionais para tratar escritas não atômicas.
- Sistema de bloqueio de duas fases estrito (Strict 2PL).
- Gerenciamento do buffer com ROUBAR, NÃO-FORÇAR com WAL (Escrita Antecipada no Log).

Verificação (Checkpointing)

- Periodicamente, o SGBD cria um ponto de verificação, para minimizar o tempo gasto para recuperar em caso de uma eventual falha do sistema. Grava no log:
  - registro begin_checkpoint: Indica quando a verificação iniciou.
  - registro end_checkpoint: Contém a *tabela de transações* e *tabela de páginas 'sujas'*. Este é um `ponto de verificação impreciso' (fuzzy):
    - Outras transações continuam executando; então estas tabelas estão precisas somente até o momento do registro de begin_checkpoint.
    - Não há nenhuma tentativa de forçar a escrita de páginas 'sujas' no disco; precisão do ponto de verificação limitada pela alteração mais antiga não gravada em uma página 'suja'. (Então é uma boa idéia gravar páginas 'sujas' no disco periodicamente!)
  - Armazenar o LSN de pontos de verificação em um local seguro (registro *mestre*).

Visão Geral: O Que é armazenado Onde

Registro de Log

prevLSN
ID_transação
tipo
págID
tamanho
deslocamento
imagem-anterior
imagem-posterior

Páginas de Dados

cada com um
págLSN
(LSN mais recente)

registro mestre

Tabela de transações

últimoLSN
Estado da transação

Tabela de Pág. 'Sujas'

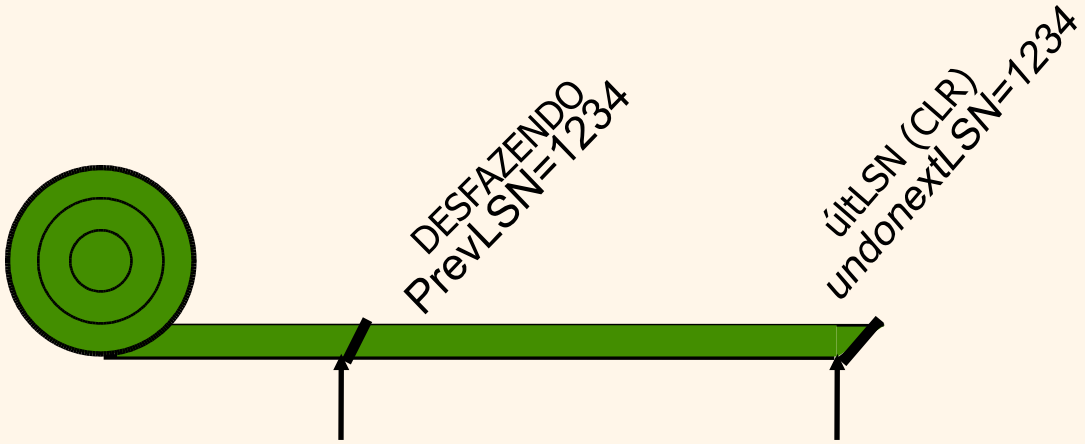
recLSN
(LSN mais antigo
que sujou a página)

flushedLSN

Desfazendo de Transação Simples (Abort)

- Por enquanto, considere um desfazer explícito de uma transação.
  - Nenhuma falha envolvida.
- Queremos repassar o log na ordem contrária, DESFAZENDO atualizações.
  - Pegar últimoLSN da transação da tabela de transações.
  - Pode seguir a sequência de registros do log de trás para frente através do campo prevLSN.
  - Antes de iniciar o DESFAZER, gravar um registro de *Abort*.
    - Para recuperar falhas durante o DESFAZER!

Abort, cont.

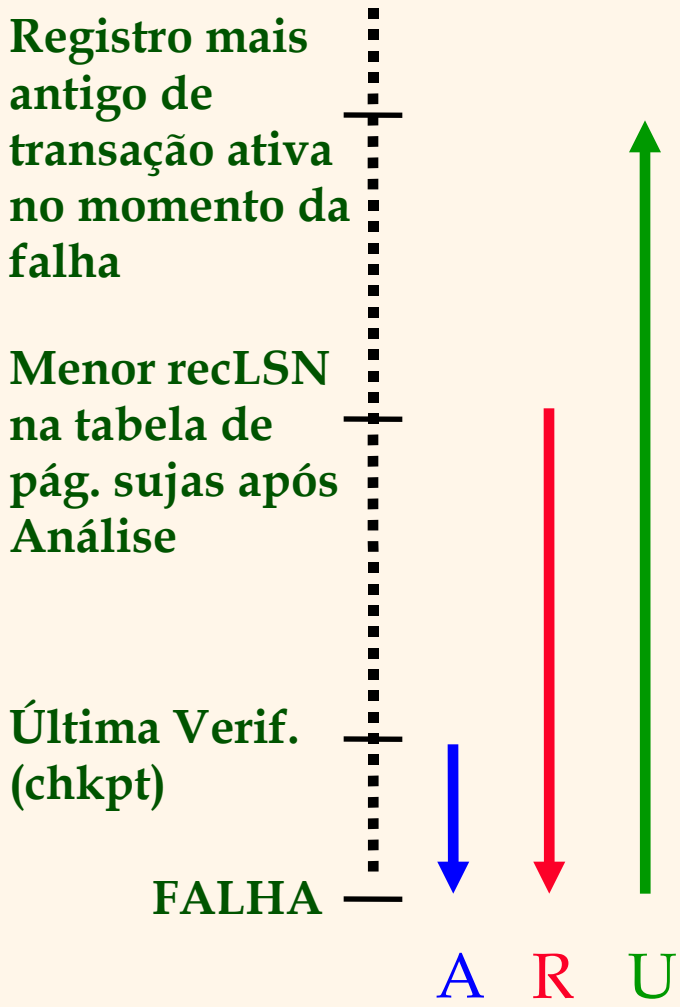


- **Para realizar um** DESFAZER**, deve ter um bloqueio no dado!**
  - Nenhum problema!
- **Antes de restabelecer o valor antigo de uma página, gravar um CLR:**
  - Continuar escrevendo no log enquanto DESFAZ!!
  - CLR tem um campo extra: desfazerPróxLSN (undonextLSN)
    - Aponta para o próximo LSN para desfazer (i.e. o prevLSN do registro que é desfeito no momento).
  - CLRs *nunca* são desfeitos (mas eles podem ser REFEITOS ao se repetir um histórico: garante Atomicidade!)
- **Ao final do** DESFAZER**, gravar um registro "fim" no log.**

Validação de Transação (commit)

- Gravar registro de commit no log.
- Todos registros do log até o últimoLSN da transação são gravados em mídia estável.
  - Garante que LSNdescarregado ≥ últimoLSN.
  - Note que as escritas do log em mídia estável são escritas síncronas e seqüenciais no disco.
  - Vários registros de log por página.
- Commit() retorna.
- Gravar registro de *fim* no log.

Recuperação a Falhas: Visão Geral



- Comece por um ponto de verificação (encontrado no registro mestre).
- Três fases. É necessário:
  - Encontrar quais transações tiveram *commit* desde o ponto de verificação, quais falharam (Análise).
  - REFAZER ***todas*** as ações.
    - (repetir histórico)
  - DESFAZER efeitos de transações que falharam.

Recuperação: A Fase de Análise

- Possui 3 tarefas:
  - Determina o ponto no log em que deve iniciar a fase DESFAZER;
  - Determina as páginas que estavam sujas;
  - Identifica transações ativas.
- Varrer o log a partir do ponto de verificação, para frente.
  - Registro de 'Fim': Remover transação da tabela de transações.
  - Outros registros: Adicionar transação na tabela de transações, atualizar últLSN=LSN, alterar o estado da transação no commit.
  - Registro de Update: Se P não pertencente à Tabela de Páginas 'Sujas',
    - Adicionar P na tabela de páginas sujas, atualizar o seu recLSN: recLSN=LSN.

Recuperação: A Fase de REFAZER

- *Repetimos o Histórico* para reconstruir o estado no momento da falha:
  - Reaplicar *todas* as atualizações (mesmo de transações abortadas!), refazer CLRs.
- Varrer o log a partir do registro que contém o menor recLSN da T.P.S. Para cada LSN de CLR ou registro de update, REFAZER a ação exceto quando:
  - Página afetada não está na Tabela de Páginas Sujas, ou
  - Página afetada está na T.P.S, mas tem recLSN > LSN, ou
  - PágLSN >= LSN do registro de log verificado.
- Para REFAZER uma ação:
  - Reaplicar as ações gravadas no log.
  - Ajustar págLSN para LSN. Nenhuma registro (log) adicional!

Recuperação: A Fase de DESFAZER

A Desfazer={ *l* | *l* um não sofreu *commit*}

Repetir:

- Escolha o maior LSN entre 'A Desfazer'.
- Se este LSN é um CLR e desfazerPróximoLSN==NULL
  - Escreva um registro de Fim para esta transação.
- Se este LSN é um CLR, e desfazerPróximoLSN != NULL
  - Adicione desfazerPróximoLSN em 'A Desfazer'
- Caso contrário, este LSN é uma atualização (update). Desfaça a atualização, escreva um CLR, adicione prevLSN em 'A Desfazer'.

Até que: 'A Desfazer' esteja vazio.

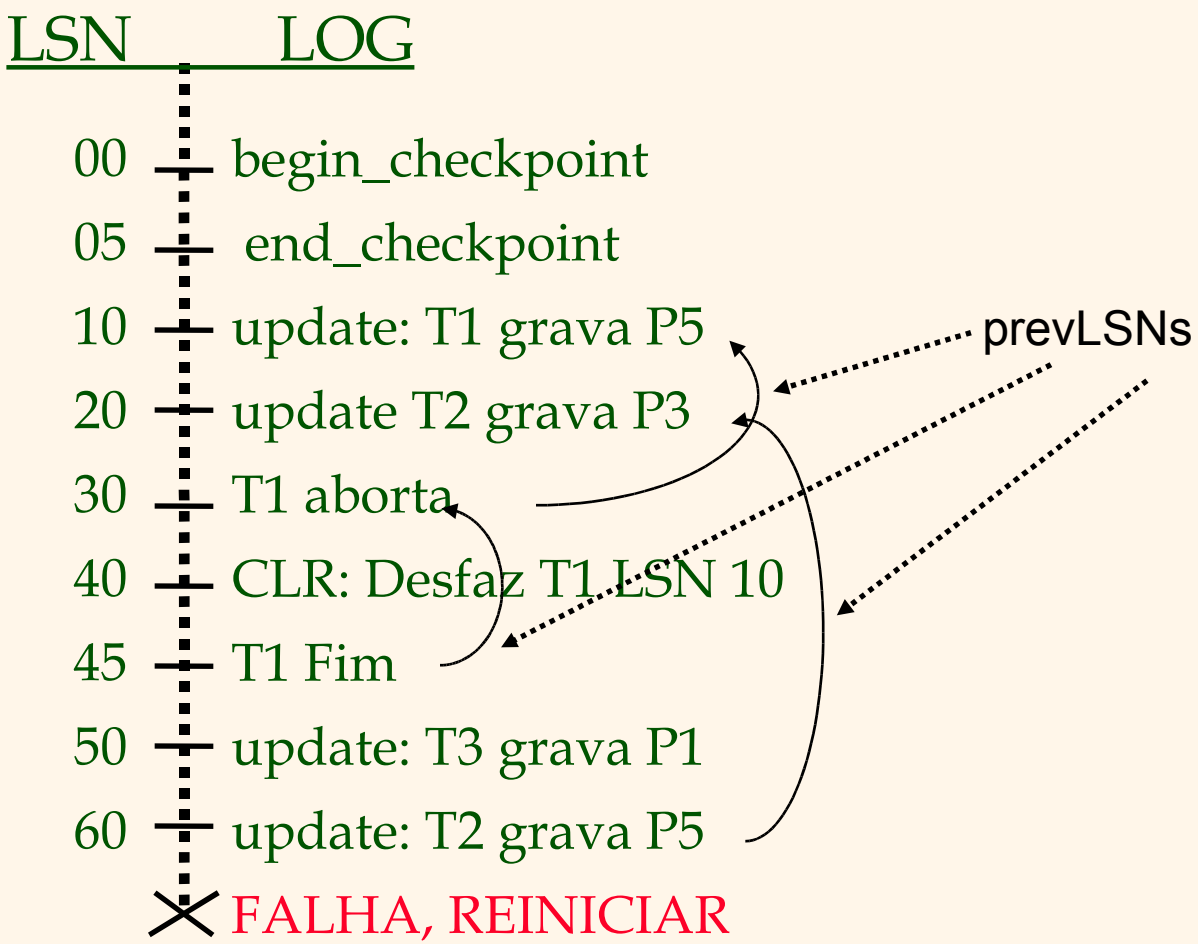
Exemplo de Recuperação de Falha

Tabela de transações
- últLSN
- estado

Tabela Páginas Sujas
- recLSN

LSN descarregado

'A Desfazer'



Exemplo: Falha Durante Reinicialização!

Tabela de transações
- últLSN
- estado

Tabela Páginas Sujas
- recLSN

flushedLSN

'A Desfazer'

LSN	LOG
00,05	begin_checkpoint, end_checkpoint
10	update: T1 grava P5
20	update T2 grava P3
30	T1 aborta
40,45	CLR: Desfaz T1 LSN 10, T1 Fim
50	update: T3 grava P1
60	update: T2 grava P5
	FALHA, REINICIAR
70	CLR: Desfaz T2 LSN 60
80,85	CLR: Desfaz T3 LSN 50, T3 fim
	FALHA, REINICIAR
90	CLR: Desfaz T2 LSN 20, T2 fim

desfazerpróxLSN

Tópicos Adicionais sobre Falhas

- O que ocorre se o sistema falha durante a Análise? E durante o REFAZER?
- Como limitar a quantidade de trabalho na fase de REFAZER?
  - Persistir páginas periodicamente.
  - Fazer *checkpoints* periodicamente.
- Como limitar a quantidade de trabalho na fase de DESFAZER?
  - Evitar transações de longa duração.

Resumo de Recuperação/Log

- Gerenciador de Recuperação garante Atomicidade e Durabilidade.
- Use WAL para permitir ROUBAR/NÃO FORÇAR sem comprometer exatidão.
- LSNs identificam registros do log; encadeados para trás, por transação (pelo prevLSN).
- págLSN permite comparação entre página de dados e registros do log.

Resumo, cont.

- Pontos de Verificação: Uma maneira rápida de limitar a varredura do log na recuperação.
- Recuperação em 3 fases:
  - Análise: A partir do ponto de verificação (para frente).
  - Refazer: A partir do recLSN mais antigo (para frente).
  - Desfazer: Para trás a partir do fim até o primeiro LSN da transação mais antiga ativa no momento da falha.
- Para Desfazer, gravar CLRs.
- Refazer "repete histórico": Simplifica a lógica!