Banco de Dados

Controle de Concorrência e Recuperação de Transação

Prof. João Eduardo Ferreira
Prof. Osvaldo Kotaro Takai

Última atualização: 20 de janeiro de 2006

Tópicos

- Modelo Transacional Clássico
  - Transações ACID
  - Teoria da Seriação
- Hipóteses Implícitas ao Modelo Clássico
  - Aplicações que as violam
    - CAD, Groupware, transações iterativas
    - Sistemas heterogêneos, Computação Móvel, etc.
- Alternativas Modernas, não - ACID
- Gerenciador de Recuperação de Transações

Evolução do Sistemas de BDs

- Sistemas Monousuários

Evolução do Sistemas de BDs

- Sistemas Monousuários
- Sistemas Centralisados Multiusuários
  - Necessidade de Controle de Concorrência
  - Tolerância a falhas

Evolução do Sistemas de BDs

- Sistemas Monousuários
- Sistemas Centralisados Multiusuários
  - Necessidade de Controle de Concorrência
  - Tolerância a falhas
- Sistemas Distribuídos Homogêneos

Evolução do Sistemas de BDs

- Sistemas Monousuários
- Sistemas Centralisados Multiusuários
  - Necessidade de Controle de Concorrência
  - Tolerância a falhas
- Sistemas Distribuídos Homogêneos
- Sistemas Distribuídos Heterogêneos Fixos
  - Modelo ACID não pode mais ser diretamente utilizado.

Evolução do Sistemas de BDs

- Sistemas Monousuários
- Sistemas Centralisados Multiusuários
  - Necessidade de Controle de Concorrência
  - Tolerância a falhas
- Sistemas Distribuídos Homogêneos
- Sistemas Distribuídos Heterogêneos Fixos
  - Modelo ACID não pode mais ser diretamente utilizado.
- Sistemas Distribuídos Heterogêneos Móveis

Controle de Concorrência: Objetivos

- Vários usuários
- Compartilhamento de recursos
  - no nosso caso: recurso = dados
  - dados = {dados propriamente, índices, dicionário de dados, etc}
- *Independência* no acesso
  - um usuário não precisa saber da existência de outros
- *Modelo comportamental* único para todos os acessos:
  - **Modelo Transacional**

Modelo Transacional Clássico

- Transação é um programa que acessa dados.
- Sempre termina.
- Término com sucesso: **confirmação** (COMMIT)
  - Todas as operações são confirmadas e mantidas
- Término com falha: **aborto** (ABORT)
  - Todas as operações são desfeitas
- Transações confirmadas não podem ser abortadas e vice-versa.

Propriedades ACID

- **A**tomicidade
- **C**onsistência
- **I**solamento
- **D**urabilidade

Atomicidade

- Uma transação é atômica se:
  - Ou todas suas operações são confirmadas no banco de dados;
  - Ou todas suas operações são desfeitas.
    - o sistema deve voltar ao mesmo estado em que estava antes do início da transação.

Consistência

- A execução de uma transação:

BD consistente

BD consistente

Isolamento

- A execução de uma transação não pode ser afetada por outras executando concorrentemente.
- Tudo deve se passar como se todos os recursos estivessem disponíveis.

Durabilidade

- Os efeitos de uma transação confirmada não podem ser desfeitos.

Modelos Não-Clássicos (Não-ACID)

- **Transações Aninhadas** (nested transactions)
  - Viola atomicidade
- **Transações Compensatórias**
  - Viola atomicidade e durabilidade
- **Transações Partidas** (split transactions)
  - Viola atomicidade
- **Transações Canguru**
  - Viola atomicidade e durabilidade

Escalonamentos

- As diversas operações de transações precisam ser executadas em alguma ordem
- Escalonamento = Ordem de execução
- Quando um escalonamento é correto?
- Como garantir que um escalonador gera apenas escalonamentos corretos?
- Teoria da Seriação

Sumário

- CC é fundamental para permitir o uso mais eficiente dos recursos computacionais.
- Na medida que se diversificam e modernizam as aplicações de BDs, aumenta-se a exigência sobre o CC.
- A cada evolução, tenta-se manter ao máximo as propriedades ACID.

Teoria da Seriação

- Trata das operações executadas
- Ignora o código da transação
- Operações consideradas
  - leitura: r[x] *(valor lido não importa)*
  - escrita: w[x] *(valor escrito não importa)*
  - confirmação: c
  - aborto: a
  - x: qualquer item de dado (dado, índice, dicionário, etc)

Tópicos

- Escalonamentos
- Teoria da Seriação Clássica
- Controle de Concorrência pelo Protocolo de Bloqueios Bifásicos.

O Problema dos Escalonamentos

- T1: depósito de v=$50

  A = Read[x];

  A = A + 50;

  Write[x, A];

  Commit;

- T2: juros de 10%

  B = Read[x];

  B = 1,1 * B;

  Write[x, B];

  Commit;

  - Escalonamentos **seriais** com $x_{inicial}$ = $100
    - (T1;T2) : x_{final} = $165
    - (T2;T1) : x_{final} = $160
  - Escalonamentos Concorrentes podem ter outro comportamento (Violação do **Isolamento**)

Escalonamento Concorrente Problemático

A = Read[x];	
	B = Read[x];
A = A + 50;	
	B = 1,1 * B;
Write[x, B];	
	Write[y, B];
Commit;	
	Commit;

Operação perdida →

- x_{final} = \$110 (Violação do **Isolamento**)
- Como evitar comportamentos indesejáveis?

Teoria Clássica da Seriação

- Trata das operações executadas
- Ignora o código da transação
- Operações consideradas
  - leitura: r[x] *(valor lido não importa)*
  - escrita: w[x] *(valor escrito não importa)*
  - confirmação: c
  - aborto: a
  - x: qualquer item de dado (dado, índice, dicionário, etc)
- Escalonamento anterior:
  - E = r1[x]; r2[x]; w1[x]; w2[x]; c1; c2;

Propriedades Desejáveis de Escalonamentos

- Escalonamentos recuperáveis.
- Escalonamentos livres de cascateamento de abortos.
- Escalonamentos Estritos.
- Escalonamentos corretos.

Conflitos

- Transações Ti ≠ Tj concorrentes;
- Duas operações sobre o mesmo dado x conflitam se uma delas é **operação de escrita**.
- Pares conflitantes

  ri[x] < wj[x] wj[x] < ri[x]

  rj[x] < wi[x] wi[x] < rj[x]

  wi[x] < wj[x] wj[x] < wi[x]
- Não conflitam:

  ri[x] < rj[x] rj[x]< ri[x]

Escalonamentos Equivalentes

- E1 é equivalente a E2:

  E1 e E2 ordenam conflitos da mesma forma

- Exemplo:

  E1 = r1[x]; w1[y]; c1; r2[y]; r2[z]; w2[x]; c2

  E2 = r1[x]; r2[z]; w1[y]; r2[y]; c1; w2[x]; c2

  E3 = r1[x]; r2[z]; r2[y]; w1[y]; c1; w2[x]; c2

- E1 é equivalente a E2 mas não a E3.

Escalonamentos Corretos: Escalonamentos Seriáveis

- Um escalonamento é **seriável** se for equivalente a algum escalonamento *serial.*
- Dado E, podem existir mais de um e escalonamentos seriais equivalentes.
- Escalonamentos não seriáveis devem ser proibidos pelo Escalonador.

Grafo da Seriação

- E é um escalonamento sobre T1, ... , Tn.
- Gs(E) é definido:
  - Nós: T1, ... , Tn.
  - Arestas dirigitas: Ti $\rightarrow$ Tj se existe x tal que:
    - pi[x] e qj[x] são operações conflitantes
    - pi[x] < qj[x] em E

Teorema da Seriação

Um escalonamento E é seriável se e somente se Gs(E) é acíclico.

Por exemplo:

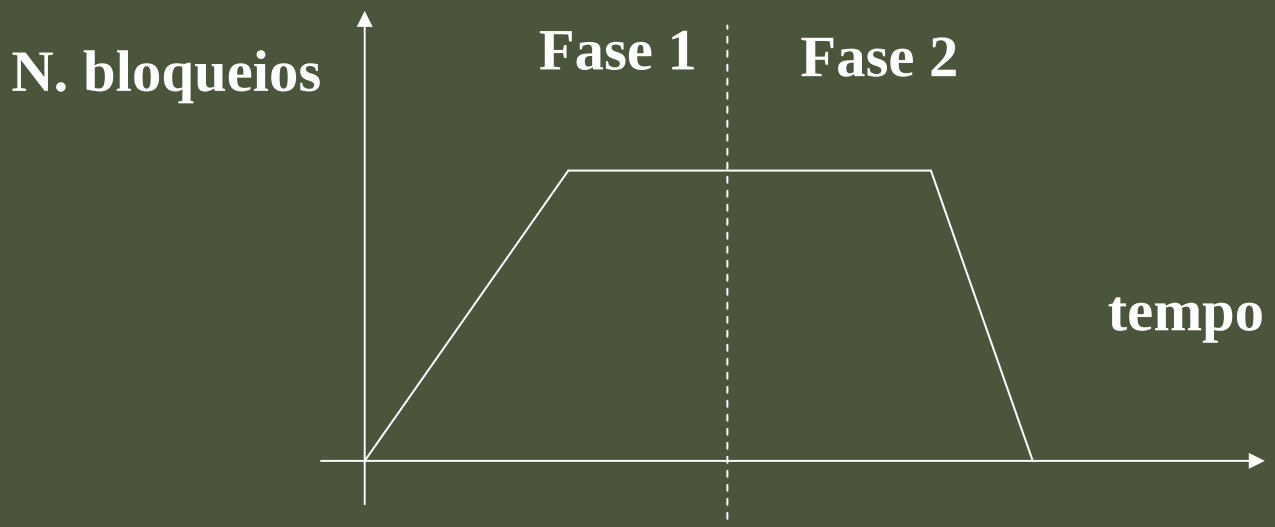
- E1 é serial, Gs(E1) é acíclico
- E2 é seriável, Gs(E2) é acíclico
- E3 é seriável, Gs(E3) é acíclico

O Protocolo de Bloqueios Bifásicos (BBF)

- Escalonamentos BBF são sempre seriáveis.
- pli[x] : **bloqueio** para executar pi[x]
- pui[x]: **desbloqueio** de pi[x]
- pli[x] < pi[x] < pui[x]
- pli[x] e qlj[x] conflitam se pi[x] e qj[x] conflitam.
- Ti requer pli[x] e obtém se não há Tj com qlj[x].
- Caso contrário, Ti é posta em espera.

Regra das Duas Fases

- Após liberar um bloqueio, uma transação não pode mais obter **nenhum outro bloqueio.**



BBF: Propriedades

- Pode gerar travamentos (*deadlocks*)
- Necessidade de monitoramento de travamentos.
- Uma vítima é escolhida, abortada e reiniciada.
- BBF é facilmente generalisável para transações distribuídas.
- Controle de travamento também é extensível no caso distribuído.
- BBF Estrito: bloqueios só liberados no fim.

Outros Protocolos

- Ordenação por Marcas de Tempo.
- Protocolos otimistas
- Protocolos que constroem o grafo da seriação

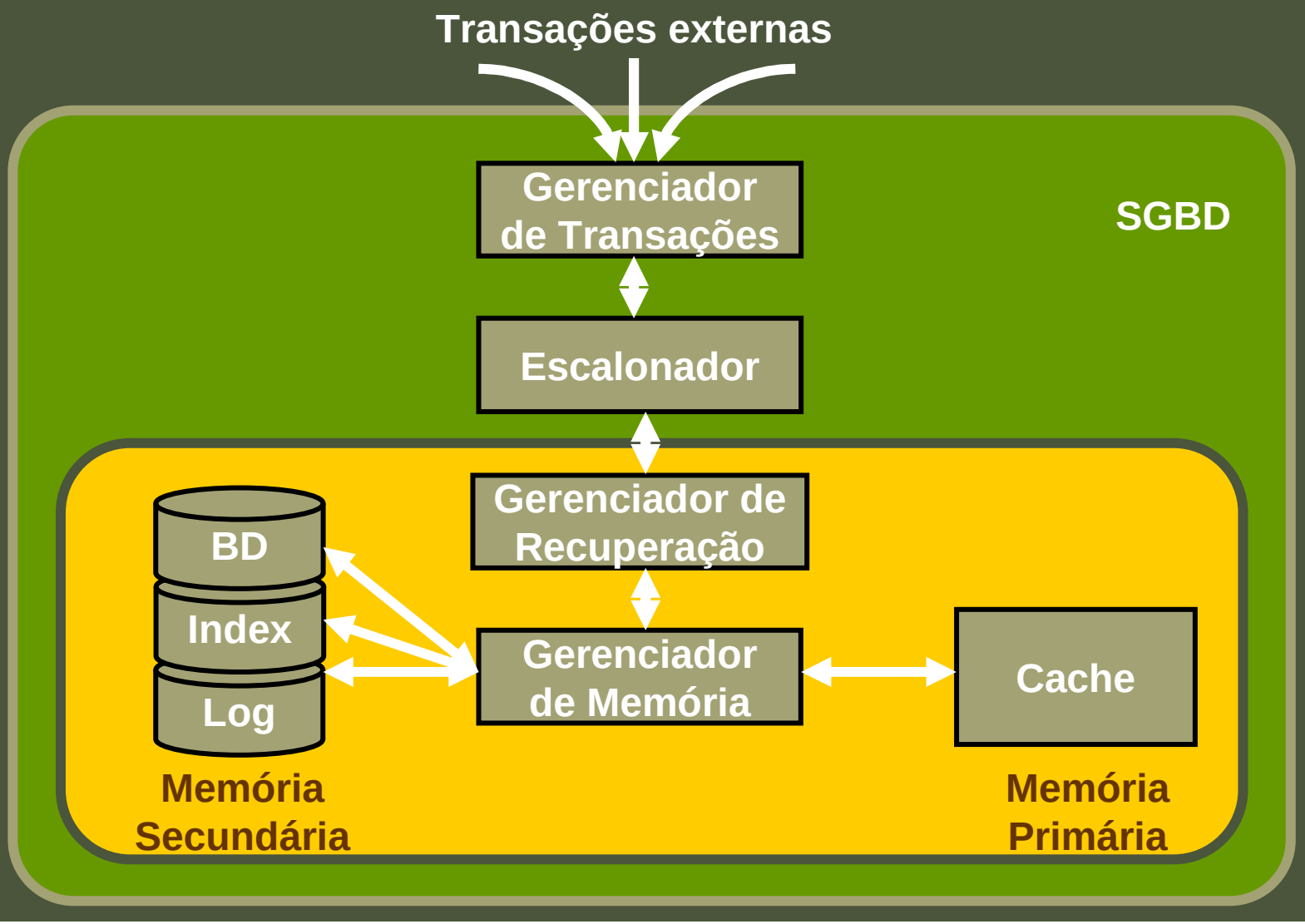
Sumário

- Teoria da Seriação abstrai o código das transações.
- Transações equivalentes ordenam conflitos da mesma forma.
- Escalonamentos seriáveis são equivalentes a algum escalonamento serial.
- BBF garante escalonamentos seriais.

Gerenciador de Memória e Recuperação

- Falhas
  - transação
  - sistema
  - meio de armazenamento
  - distribuídas
- Gerenciador de Dados

Gerenciador de Memória e Recuperação



Gerenciador de Memória (Cache)

- Busca(Item_de_dado)
- Descarrega(Posição)

- políticas de substituição de dados (LRU, FIFO, frequência)

Interface GR

- GR-leitura(Ti,x)
- GR-escrita(Ti,x,v)
- GR-confirma(Ti)
- GR-aborta(Ti)
- GR-reinicia

Estrutura do Log

- Ti - transação
- xj - localização em memória secundária
- va - valor anterior
- vp - valor posterior

Funções do registro de Log

- (Ti, inicia)
- (Ti, xj, va, vp) r/w
- (Ti, confirma)
- (Ti,aborta)

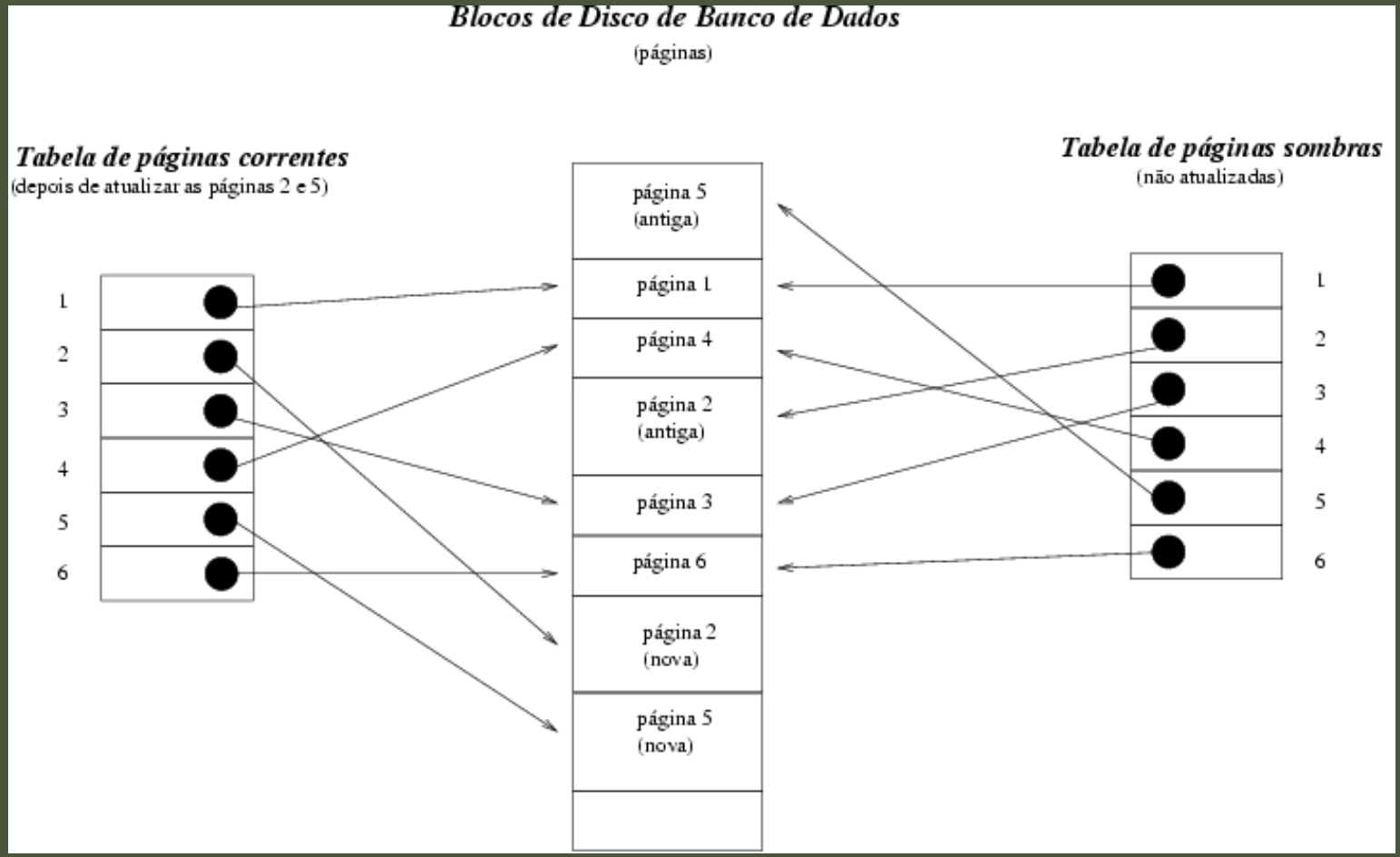
Exemplo

transação	arquivo de log
T0: início	(T0 inicia)
leia(A)	(T0, Aj, 30, 30)
A:=A-20	
escreva(A)	(T0, Aj, 30, 10)
leia (B)	(T0, Bj, 20, 20)
B:=B+30	
escreva(B)	(T0, Bj, 20, 50)
T0: fim	(T0, confirma)
(Ti, confirma)	
(Ti,aborta)	

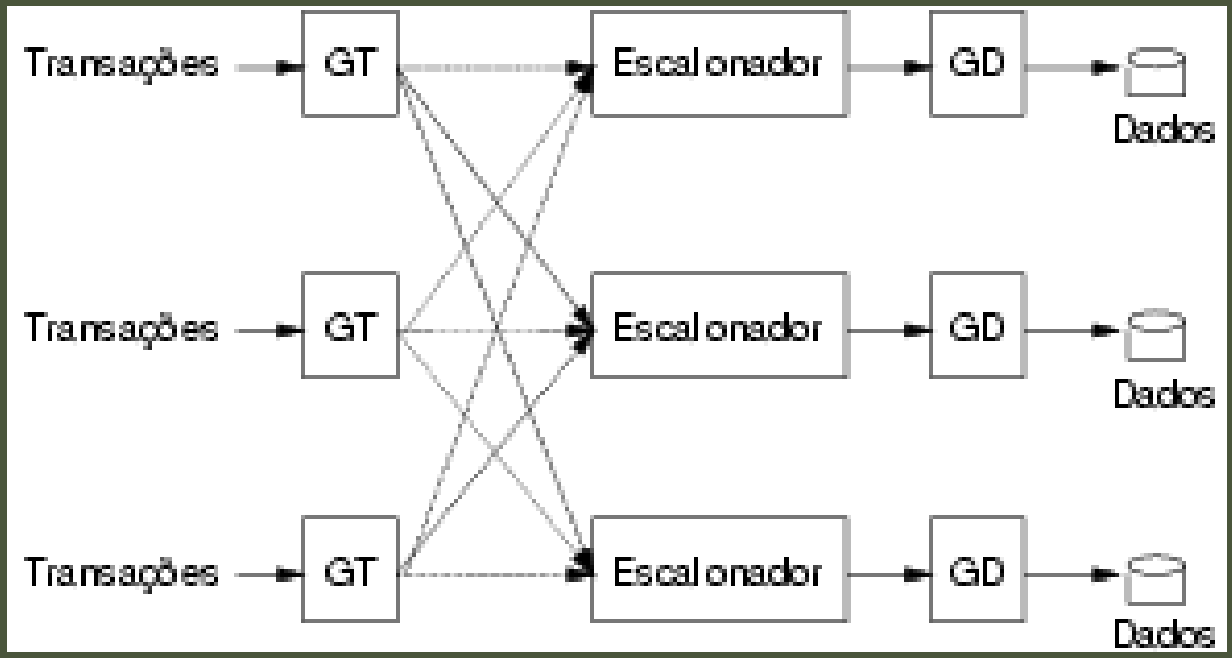
Algoritmos para GR

- atualizações imediatas
- atualizações postergadas
- combinações (no)UNDO, (no)REDO
- pontos de confirmação do registro do Log

GR - Páginas sombras



BD Distribuído



Protocolo 2PC

- Coordenador - Participantes
- fase 1 - votação
- fase 2 - decisão

regras - Protocolo 2PC

- todos os participantes devem atingir mesma decisão
- nenhum participante deve reverter sua decisão
- a confirmação global depende da confirmação local de cada participante
- cada participante em caso de falha, tem seu controle local

Passos - Protocolo 2PC

- nó coordenador solicita confirmação a todos os nós participantes(S/N)
- cada participante responde(**sim/não**)
- caso exista **não,** mensagem aborto a todos (exceto o(s) nó(s) participantes com **não**
- cada participante que votou **sim**, aguarda mensagem de confirmação do nó coordenador encerrando a sua participação

Sumário

- Arquitetura dos Gerenciadores de dados.
- Recuperação da Informação
- 2PC como garantia de integridade das transações distribuídas
- Base para as técnicas de distribuição de dados

Bibliografia

- Ferreira, J.E.; Finger, M., Controle de concorrência e distribuição de dados: a teoria clássica, suas limitações e extensões modernas, Coleção de textos especialmente preparada para a Escola de Computação, 12ª, São Paulo, 2000. Copia em: http://www.ime.usp.br/~jef/ec2000.ps

- Korth, H.; Silberschatz, A. Sistemas de Bancos de Dados. 3a. Edição, Makron Books, 1998.