Processamento de Transações

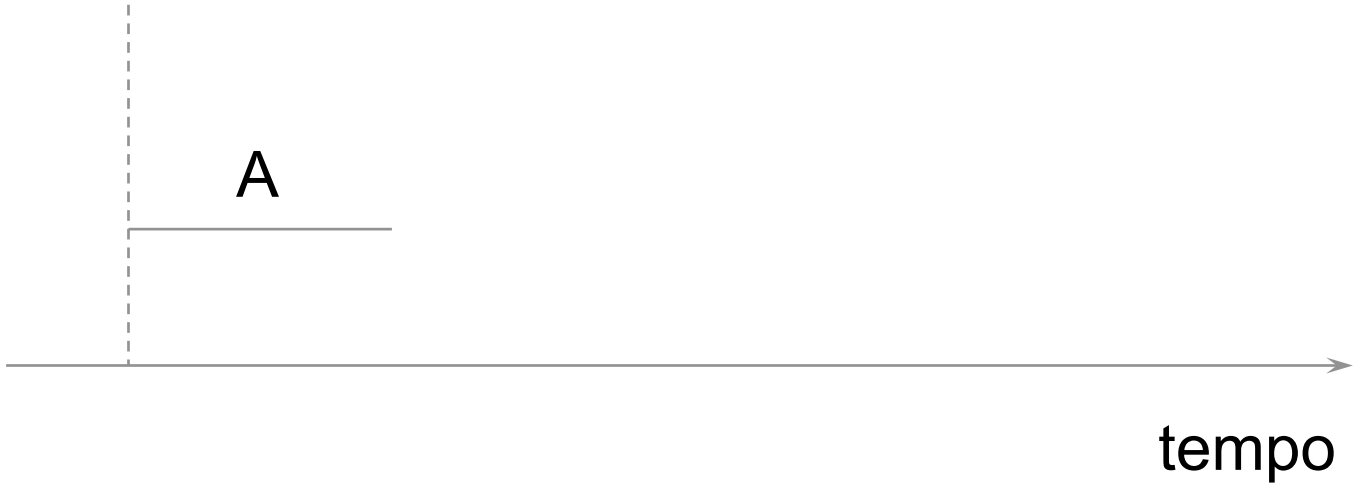
Banco de Dados

Profa. Dra. Cristina Dutra de Aguiar Ciferri

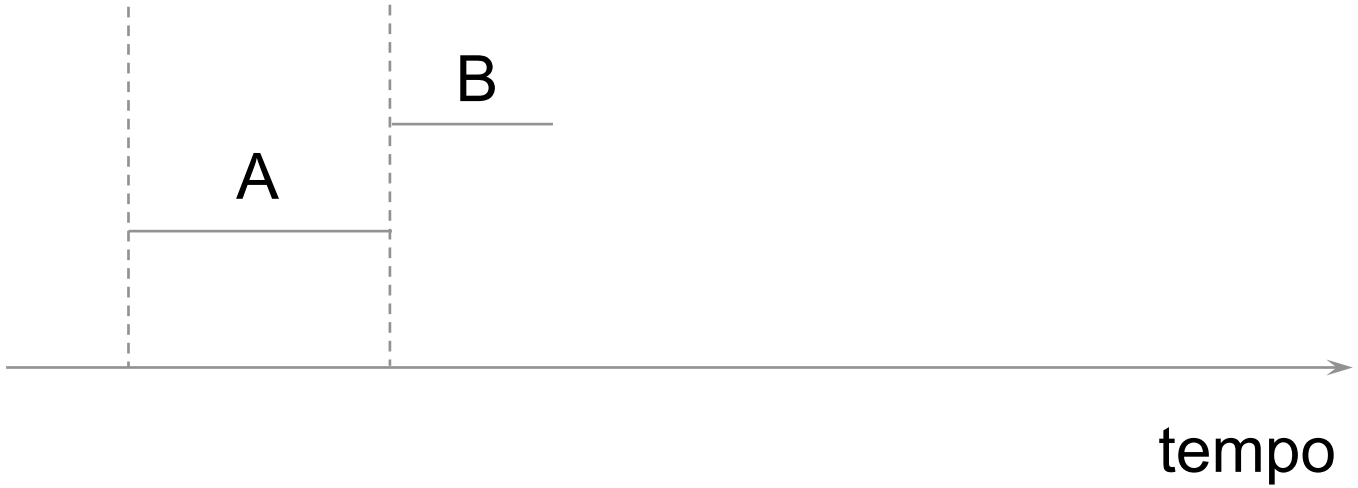
Introdução

- Ambiente multiusuário
  - vários usuários utilizam o mesmo sistema ao mesmo tempo
  - múltiplos programas (transações) compartilham a mesma CPU
- Forma de execução dos programas
  - intercalada (*interleaved*)
  - *alguns comandos de um programa são executados e o programa é suspenso; alguns comandos de outro programa são executados, o programa é suspenso, e assim por diante ...*

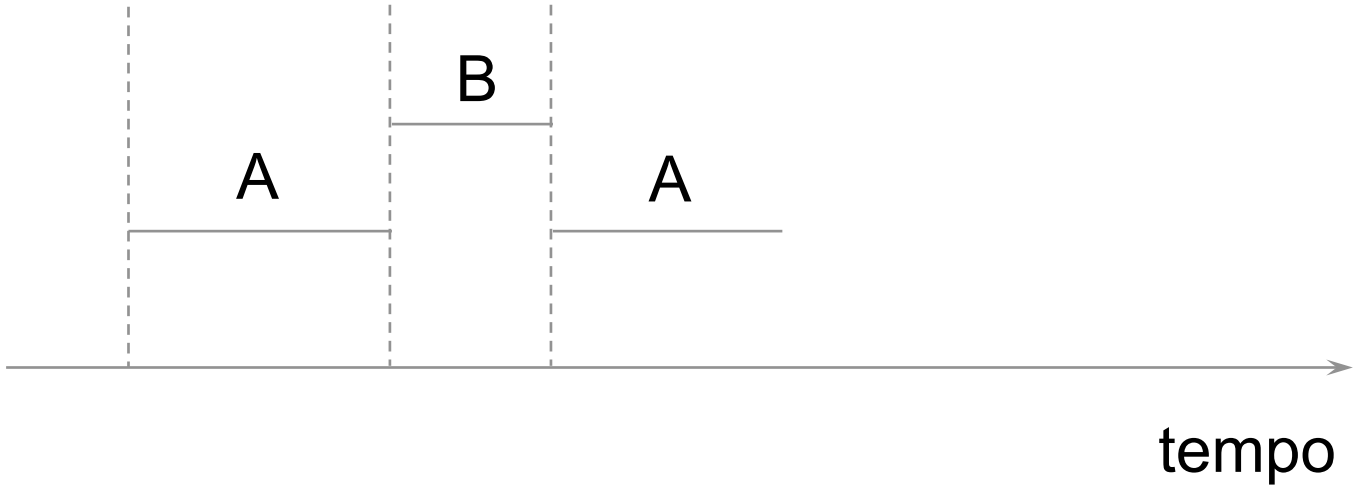
Execução Intercalada



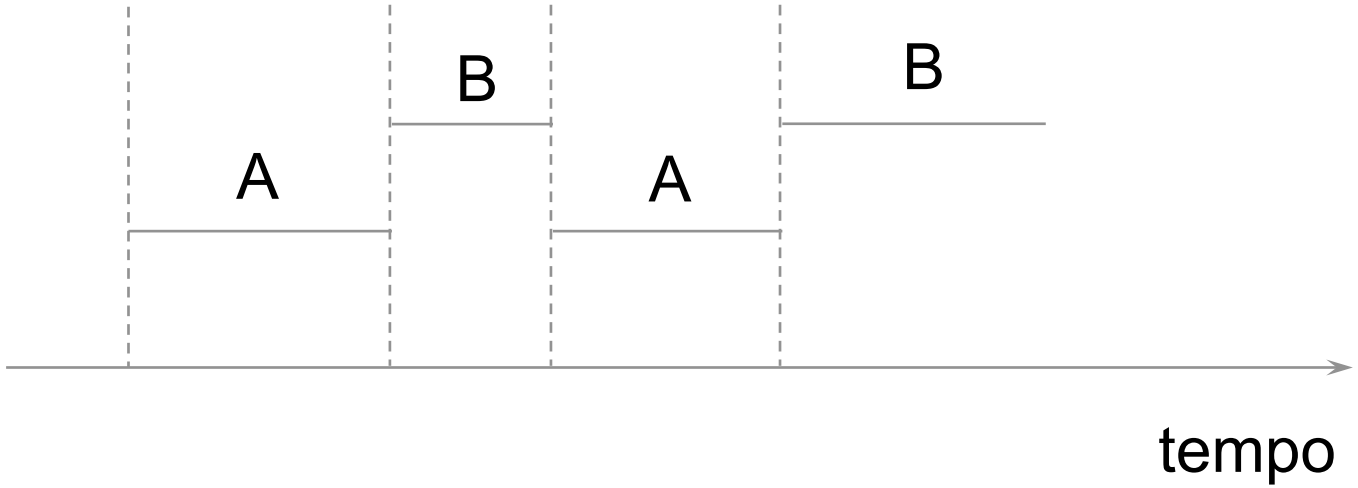
Execução Intercalada



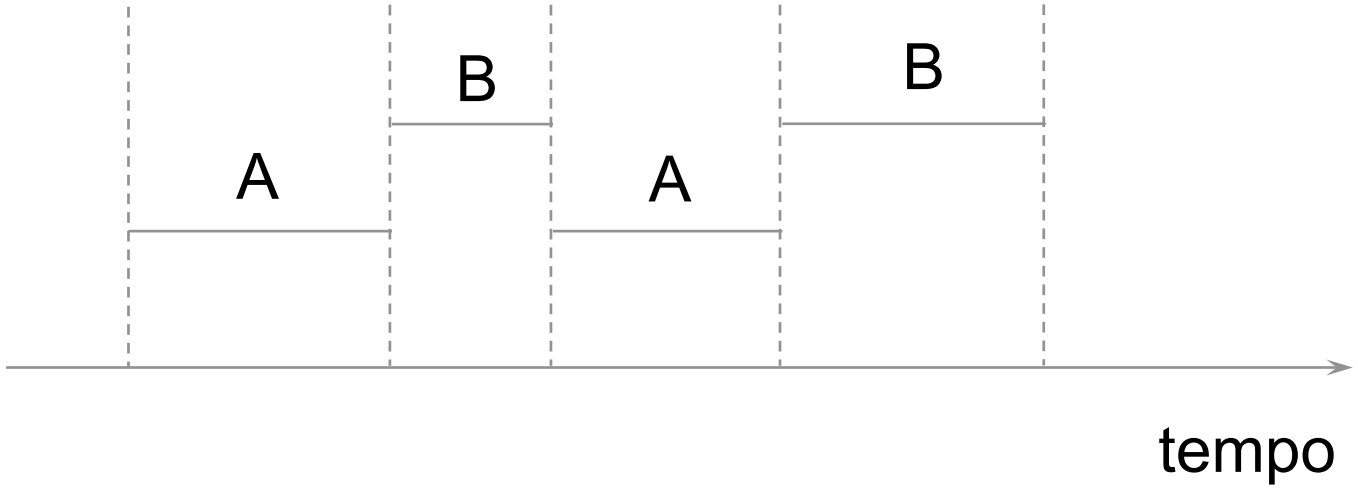
Execução Intercalada



Execução Intercalada



Execução Intercalada



- SBD multiusuário
  - recursos principais: dados armazenados no BD

Transação

- Unidade de execução de um programa que acessa ou altera o conteúdo do BD
- Seqüência de operações de escrita/leitura no BD
- Características
  - é delimitada por declarações da forma início da transação e fim da transação
  - todas as operações de escrita/leitura entre essas duas declarações são consideradas parte de uma mesma transação

Transação

- Características
  - um programa de aplicação pode conter mais do que uma transação, se as operações de escrita/leitura são limitadas por diferentes pares de declaração início da transação e fim da transação
- Observações
  - os dados do BD estão armazenados em memória secundária
  - as operações que não sejam leitura/escrita não apresentam efeito para o BD

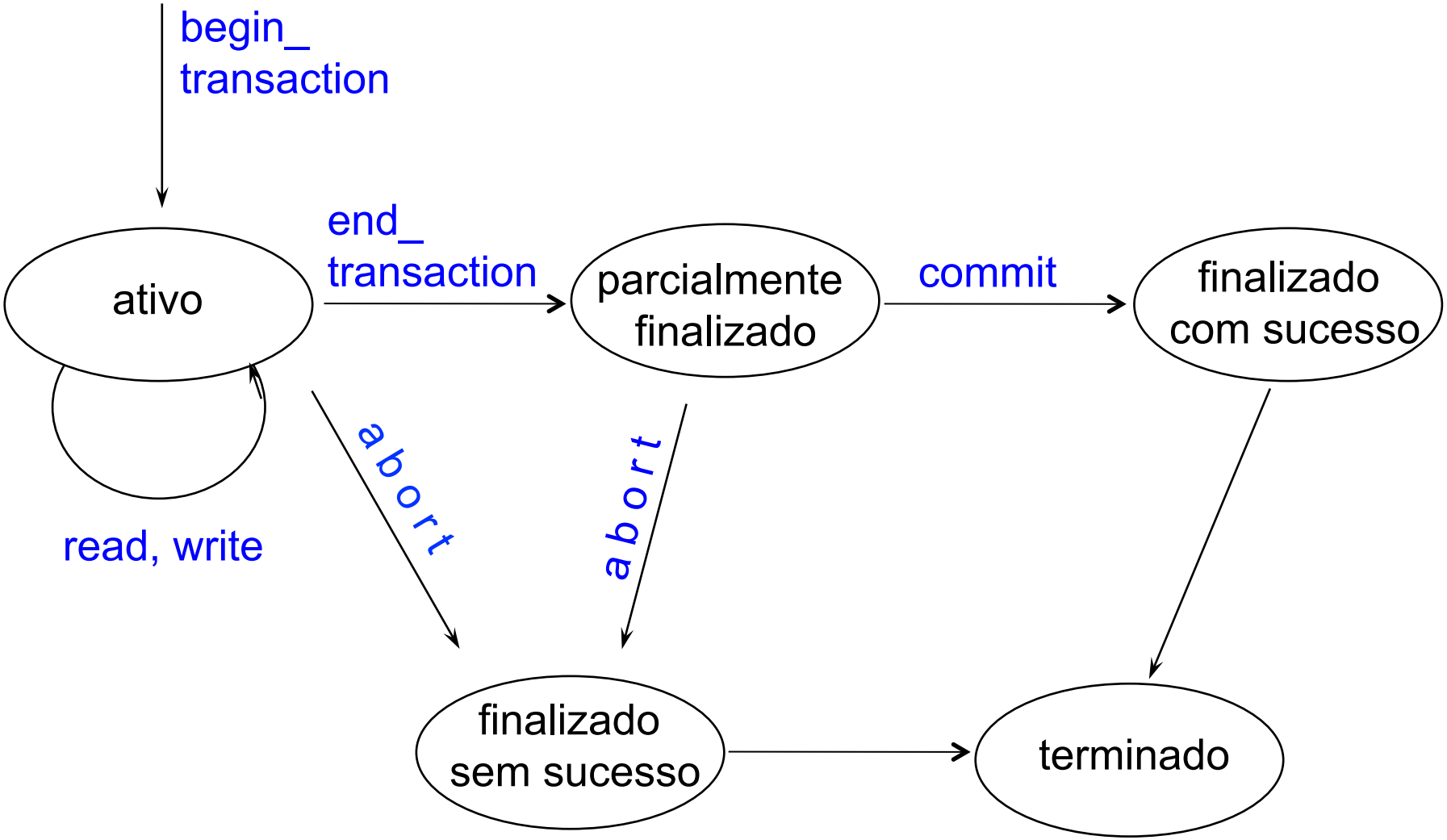
Operações das Transações

- begin_transaction
  - início da execução de uma transação
- read ou write
  - operações de leitura ou escrita nos dados do BD
- end_transaction
  - final da execução de uma transação
  - deve ser verificado se a transação executará commit ou abort

Operações das Transações

- commit_transaction
  - indica que a transação foi finalizada com sucesso
  - torna permanente as alterações realizadas no BD
- abort_transaction (rollback)
  - indica que a transação foi finalizada sem sucesso
  - descarta as alterações já realizadas no BD

Transição de Estado



Operação de Leitura

- Representada por r(x) ou read_item(x)
- Lê o item de dado x na variável de programa x
- Passos
  - encontrar o endereço do bloco que contém x
  - copiar o bloco para o *buffer* da memória principal (se necessário)
  - copiar o valor de x do *buffer* para a variável x

Operação de Escrita

- Representada por w(x) ou write_item(x)
- Escreve o valor da variável de programa x no item de dado x
- Passos
  - encontrar o endereço do bloco que contém x
  - copiar o bloco para o *buffer* da memória principal (se necessário)
  - copiar o valor da variável x para o *buffer*
  - escrever o novo valor do item de dado x no disco (atualização do BD)

Observações

- Transações submetidas pelos usuários
  - podem executar concorrentemente
  - podem acessar e alterar os mesmos itens de dados
- Execução não controlada
  - pode originar problemas
  - inconsistência do banco de dados

Exemplos

- Transação 1

  r(x)

  x := x - n

  w(x)

  r(y)

  y := y + n

  w(y)

- Transação 2

  r(x)

  x := x + m

  w(x)

- t_1: $r_1(x)\ w_1(x)\ r_1(y)\ w_1(y)\ c_1$

- t_2: $r_2(x)\ w_2(x)\ c_2$

Propriedades ACID das transações

Atomicidade

Consistência

Isolação

Durabilidade

Propriedades ACID

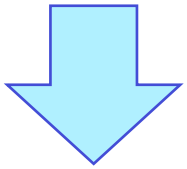
- Atomicidade
  - *uma transação é uma unidade atômica*
  - todas as operações das transações são finalizadas e refletidas no BD ou nenhuma delas é finalizada e refletida
- Consistência
  - *transformações preservam a consistência*
  - a execução correta de uma transação leva o BD de um estado consistente a outro estado consistente

Propriedades ACID

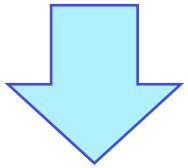
- Isolação
  - *transações são isoladas umas das outras*
  - cada transação assume que está sendo executada sozinha no sistema, e o SGBD garante que os resultados intermediários da transação são escondidos de outras transações executando concorrentemente
- Durabilidade
  - os valores dos dados alterados durante a execução de uma transação devem persistir após a finalização desta

Propriedades ACID

protocolos de controle de concorrência

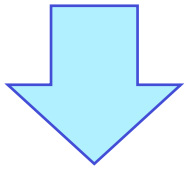


garantem a consistência dos dados através de acessos concorrentes

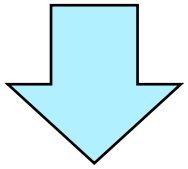


isolação

protocolos de recuperação de falhas



garantem a consistência dos dados após falhas do sistema



atomicidade e durabilidade

Por que é Necessário o Controle de Concorrência?

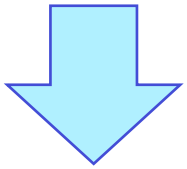
- Escalonamento serial das transações
  - correto, por que garante a consistência dos dados
  - diminui a concorrência
- Escalonamento intercalado das transações
  - pode gerar escalonamentos incorretos devido às operações conflitantes

Solução: Protocolo 2PL

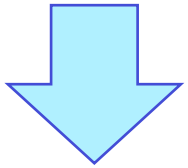
- Protocolo de controle de concorrência
  - oferece várias regras que devem ser seguidas pelas transações
- Baseado em
  - travas exclusivas
  - travas compartilhadas
- Garante escalonamentos serializáveis
  - escalonamento serializável
    - escalonamento não serial, equivalente a um escalonamento serial

Por que é Necessária a Recuperação de Falhas?

falhas catastróficas

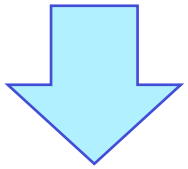


falhas que atingem uma grande porção do BD

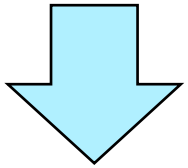


falhas de disco

falhas não catastróficas



falhas que geram inconsistências no BD



falhas da transação

falhas do sistema

Falhas da Transação

- Ocorrem quando uma transação não é finalizada com sucesso
- Exemplos
  - divisão por zero
  - leitura de um dado inexistente
  - transação é escolhida como vítima em algum protocolo de *deadlock*
  - valores de parâmetros incorretos
  - usuário interrompe uma transação através do CRTL-C

Falhas do Sistema

- Usualmente relacionadas à destruição dos dados da memória principal
- Exemplos
  - *bugs* no *software* do sistema gerenciador de banco de dados
  - *bugs* no *software* do sistema operacional
  - falhas de *hardware* na CPU

Falhas de Disco

- Relacionadas à destruição de dados da memória secundária
- Também chamadas de problemas físicos
- Exemplos
  - quebra física do disco
  - as funções de leitura/escrita em disco não funcionam corretamente

Soluções

- Em nível de programação
  - tratamento apropriado de exceções
- Em nível de SGBD
  - oferecimento de mecanismos de recuperação de falhas baseados em arquivos de *log*
- Em nível macro
  - realização de *backups* periódicos
    - banco de dados
    - arquivo de *log*