Disciplina	Sistema de Banco de Dados 1 Turma A
Professor	Vandor Rissoli
Alunos	Pedro Rodrigues Pereira - 17/0062686 Matheus Salles Blanco - 16/0138400 Alan Ronison de Lima e Silva - 16/0109256 Ezequiel De Oliveira dos reis - 16/0119316
Atividade	Recuperação a Falhas no BD (estudo)

A) EXISTEM FALHAS NOS BANCOS DE DADOS?

Sim, existem falhas nos bancos de dados, e podem ser classificados em dois tipos de falhas. As falhas sem dano físico ao BD, e as falhas com dano físico ao BD.

- Falhas sem dano físico ao BD:
  - Falha do servidor:

    Algum tipo de erro no servidor, que tire o banco de dados do funcionamento normal.

  - Erro de transação ou do sistema:

    Erro na transação que obrigue a transação a ser cancelada.

  - Erros locais ou condições de exceção detectadas pela transação

    Um exemplo é uma divisão por zero, ou algum outro problema que faça uma transação ser desfeita.

  - Imposição do controle de concorrência:

    Problema comum do controle de concorrência e o deadlock por exemplo.

- Falhas com dano físico ao BD:
  - Falha de disco:

    Falha de disco e um problema que afeta o banco a nível de hardware, defeito no HD por exemplo.

  - Problemas físicos e catástrofes:

    Um bom exemplo de falha por problema físico ou catástrofe seria enchentes, que danificam o banco de dados, fogo e outros desastres naturais caso o banco de dados não esteja devidamente protegido

B) EXISTE COMO RECUPERAR AS FALHAS NO BANCO DE DADOS?

Sim, existem maneiras de se recuperar falhas em banco de dados. Pois o sistema mantém um *log* dos dados existentes. Este *log* é mantido em disco físico, de modo que não é afetado pela maioria dos tipos de falhas, exceto por falha de disco ou catastrófica (falta de energia, fogo, sabotagem). O *log* é periodicamente copiado para um sistema de armazenamento (fita) para que exista uma cópia dos dados.

No momento da restauração, o banco será restaurado para o estado de consistência mais recente.

Estratégias para recuperação de um banco de dados, por categoria

- Não houve dano físico (falha não catastrófica):
    - Onde deve-se reverter as mudanças que causaram as inconsistências, para posteriormente re-implementá-las de maneira que atenda aos requisitos esperados. Se divide entre técnica de recuperação baseada em atualizações adiadas e imediatas.
- Dando em grande porção do Bando de Dados (falha catastrófica):
    - Restauração a partir de cópia (*backup*) realizado em mídia física (fitas, pen-drives). Posteriormente realiza-se a re-implementação para que o Banco de Dados alcance um estado mais atual.

- **Técnica de Recuperação no caso de Falhas Não-Catastróficas: Baseadas em atualização adiada:**
    - Adiar qualquer atualização real (gravação em disco) no banco de dados até que a transação complete sua execução com sucesso e alcance o ponto de efetivação
    - Durante a execução de uma transação as atualizações devem ser registradas no *log* e nos *buffers*
    - Após alcançar a efetivação e forçar a realização da gravação do *log* no disco, as atualizações são então finalmente registradas no banco de dados.
    - **Protocolo:**
        - Uma transação não pode alterar o banco em disco até que ela alcance seu ponto de efetivação e esse ponto não é alcançado até que todas as suas operações sejam registradas no *log* e até que seja forçada a gravação do *log* no disco
    - **Algoritmo NO-UNDO/REDO**:
        - A operação UNDO não será necessária nesta operação, visto que o banco de dados nunca é atualizado em disco até que a transação seja efetivada
        - A operação REDO deverá ser realizada caso o sistema falhe após a transação. Entrentanto, apenas será possível se as alterações não terem sido gravadas em disco.

- **Técnica de Recuperação no caso de Falhas Não-Catastróficas: Baseadas em atualização imediata:**
    - Antes que uma transação alcance o ponto de efetivação, será possível que o banco de dados seja atualizado pelas operações. Mas esta atualização deve ser registrada no banco de dados, para que a recuperação correta seja possível e, no caso de uma transação falhar depois dos registros das novas mudanças, os efeitos de suas operações no banco deverão ser desfeitos (UNDO).

C) O QUE SIGNIFICAM AS SIGLAS RELACIONADAS ABAIXO?

- **LOG em Banco de Dados**: Se caracteriza por ser o arquivo responsável por armazenar todas as operações de um banco de dados. Ou seja, sempre que é efetuado algum comando no BD, é criado um registro no log. Este log é mantido em disco. Desta maneira não pode ser afetado por nenhum tipo de falha durante o processo. Com exceção de falha de disco ou falha catastrófica. Este arquivo de log deve ser executado pelo SGBD (Sistema gerenciador de banco de dados) quando uma transação que estiver sendo executada for cancelada por algum motivo. Quando necessário realizar uma recuperação, os efeitos em questão devem ser desfeitos, para isto ser possível é necessário varrer os arquivo de log e identificar as operações já realizadas pela transação para que uma ação seja tomada, para garantir a consistência mais recente.

Exemplos:
[start, T]:
[write,T,X,valor_antigo,valor_novo]:
[write,T,X,valor_novo]:
[read,T,X]:
[commit,T]:
[abort,T]:
[checkpoint]:

- UNDO: O algoritmo UNDO se caracteriza por uma atualização imediata de falhas não catastróficas. Ou seja, permite a atualização física do banco de dados por operações de transação antes que atinja seu ponto de confirmação. De modo mais específico é utilizado para desfazer uma ação, para isso uma entrada no log para gravação deve incluir o valor antigo (imagem anterior) do item gravado.

Procedimento:

Examinar as entradas de log do tipo [write_item, T, X, valor_antigo, novo_valor] e ajustar o valor do item X no banco de dados para valor_antigo (imagem anterior).
É importante mencionar que O processo de desfazer as operações write_item de uma ou mais transações do log deve acontecer na ordem inversa da ordem em que as operações foram gravadas no log.

Exemplo no Log:

Log	Banco de Dados
<T_0 start>	
<T_0, A, 1000, 950>	
<T_0, B, 2000, 2050>	
	A = 950
FALHA	
<T_0 commit>	
FALHA	
	B = 2050

Quando o sistema retorna após a falha, ele contra no log <T0 start>, mas não <T0 commit>, portanto será processado um **UNDO** para desfazer a ação.

- REDO: Significa refazer uma ação mal sucedida, examinando as entradas no log do formato [write_item, T, X, novo_valor] e ajustando o valor do item "X" no banco para o "novo_valor".

Condições para a realização do REDO:

Em uma transação Tn, ela somente deve ser refeita caso o log demonstra os registros <Tn start> e/ou <Tn commit> no log.

Exemplo no Log:

Log	Log	Log
<T_0 start>	<T_0 start>	<T_0 start>
<T_0, A, 950>	<T_0, A, 950>	<T_0, A, 950>
<T_0, B, 2050>	<T_0, B, 2050>	<T_0, B, 2050>
	<T_0 commit>	<T_0 commit>
	<T_1 start>	<T_1 start>
	<T_1, C, 600>	<T_1, C, 600>
		<T_1 commit>
(a)	(b)	(c)

A) Não é necessário fazer nada afinal não possui nenhum commit e start.
B) É necessário REDO na T0, já que o log possui <T0 start> e <T0 commit> registrado;
C) É necessário REDO na T0 e na T1, já que o log possui <T0 start>, <T0 commit> e <T1 start> e <T1 commit>.

- CHECKPOINT em Banco de Dados:

Um checkpoint é um ponto de verificação, a cada período de tempo, o sistema irá forçar a gravação dos buffers do SGBD com intuito de criação de um ponto de verificação no Log.

A principal vantagem na utilização dos checkpoints é que as transações que possuem <T commit> no log antes de um checkpoint não necessitam de ser refeitas.

Processos de um checkpoint:

1 - Parar temporariamente a execução das transações.
2 - Forçar a gravação dos buffers na memória.
3 - Gravar um registro <checkpoint > no log.
4 - Continuar a execução das transações.