UNIOESTE – Universidade do Oeste do Paraná
Campus Foz do Iguaçu - PR

Banco de Dados

Roberto Gil Brasil

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

Apresentação

Este trabalho tem como finalidade apoiar as aulas na disciplina de Banco de Dados e deve ser complementado com leitura nas bibliografias de referência para área.

Procurou-se conciliar a introdução de assuntos teóricos com aspectos práticos da área. A ênfase em temas ligados a Modelagem de Dados, deve-se a importância, quanto ao uso deste tipo tecnologia e do inter-relacionamento destas práticas com as melhores práticas de desenvolvimento de software na atualidade.

A prática de boas técnicas de modelagem de dados contribui não somente com o aprimoramento de conhecimentos que se adquire na área de Ciência da Computação, mas também, com o conhecimento que se adquire indiretamente com as outras áreas de conhecimento humano, uma vez que para definir um bom modelo de dados é necessário também conhecer e estudar a área de atuação relacionado ao problema.

Roberto Gil Brasil
robertogbrasil@uol.com.br

Índice

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

1. Programa da Disciplina de BANCO DE DADOS

1. Semestre: Modelo lógico
Modelagem de Dados
2. Semestre: Estudo de Modelo físico

Programa da disciplina para Ciência da Computação:

1.11. e 2. Bimestre

Introdução
Modelagem de Dados
- Conceitos básicos e avançados
- Metas de projeto Modelagem de Dados
- NORMALIZAÇÃO : Modelagem 1. FN a 3. FN

Álgebra Relacional
SQL
APIs de SGBD

1.23. Bimestre

Conceitos Avançados Modelo E-R
Banco de dados OO
Banco de dados Relacionais Objeto
Tópicos Especiais - Segurança, triggers, store-procedures
Estrutura de Arquivos e de Armazenamento
Indexação e Hashing

1.34. Bimestre

Processamento de Consultas
Transações
Controle de Concorrência
Sistemas de Recuperação
Banco de Dados Distribuídos
- Fragmentação Horizontal e Vertical
- Two Phase Commit

Apêndices

- Banco de Dados Paralelos
- Processamento de Transações Avançadas
- Novas Aplicações
- Modelo de Rede
- Modelo Hierárquico

1.4 Aula Inicial

- Apresentação
- Objetivo da Matéria:
  Proporcionar ao aluno uma visão geral de Banco de Dados, o que é, qual objetivo, o por que,

- Ênfase para o modelo lógico:
  - Explicar a diferenciação entre modelo físico e modelo lógico, dando um exemplo empregado tem 1 cargo, indicando que um dos objetivos principais é mostrar como chegar a um bom modelo.

- O que será visto durante o ano (Curriculum da Máteria: listar assuntos acima e breve explicação de cada item)
- Trabalhos práticos
- Forma de avaliação: Prova 8,0 trabalhos 2,0...
  - Trabalhos: Serão apresentados em sala de aula e cobrados em prova, assim como lista de exercícios.
- Bibliografia usada
- Assumir que todos tem bons conhecimentos de Engenharia de Software (Análise, etc.), isto será fundamental.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

2. Introdução

Todo sistema de computador, manipula e processa grande quantidade de informações, onde a maior parte destas informações precisam estar presentes a qualquer momento e permanecer não importando se o computador será ou não desligados, ou seja, elas precisam "estar persistentes" em algum meio que "sobreviva" ao desligamento do computador, ou simplesmente a desativação do programa que está rodando.

Onde isto pode ser mantido: Em arquivos gravados em memória secundária (HD, etc)

Definião de BD:

Coleção de dados e informações de interesse que são mantidas para acessos diversos (consultas).

Definição de SGBD:

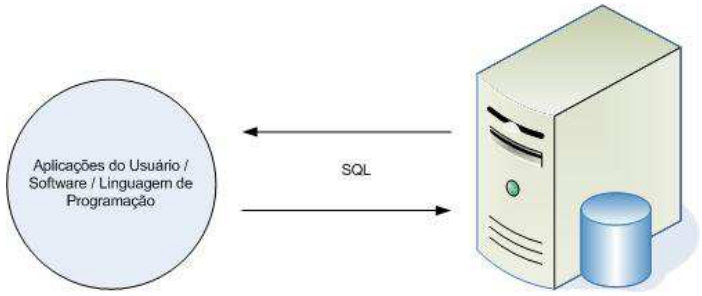
É uma coleção de dados inter-relacionados e um conjunto de programas para acessar esses dados [Sistemas Banco de Dados, 5. Edição, Silberchatz]

- [*Ver livro 5. edição, página 1, 1. parágrafo]*

***SGBD**:* Conjunto de Software e componentes voltados a gerenciar a base de dados com objetivo de garantir a esta base de dados:

- Segurança,
- Integridade,
- Performance
- Padrão único de acesso ao Banco de Dados independente de tecnologia
- Tratamento de Transação
- Tratamento de Acesso concorrente aos dados

Relacionamento da Aplicação com SGBD é uma relação de serviço:



Onde, o SQL é o pedido do que se quer do SGBD

Ex: SELECT nome FROM tabAlunos
WHERE matricula = 1000

O SQL linguagem não procedural, para "solicitar um serviço ao SGBD" é encaminhada via APIs de preferência padronizadas, específicos do SGBD no caso de linguagens de programação, tais como: JDBC, ODBC, ADO, etc.

SGBD:

a) Organização dos dados (M.E.R., tabelas)
b) Segurança e Integridade dos dados
c) Disponibilizar Informação para consultas futuras, de forma padronizada. Assim muitas tecnologias diferentes, podem ter acesso a mesma informação.

2.1 Aplicações do SGBD

Áreas de uso comum

- Banco
- Linhas aéreas
- Universidades
- Transações com cartão de crédito
- Telecomunicação
- Finanças
- Vendas
- Indústria
- Recursos Humanos
- Controles de processos de negócio em geral
- Etc.

Áreas de uso avançados / tecnologias

- BI – Business Intelligence
- Big Data
- IA – Inteligência Artificial (base de dados para aprendizado, etc)
- IoT – Internet das coisas
- Data Science

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

2.2 Exercício e Exemplo prático de modelagem de dados

Exercício: *Armazenar dados de uma aluno matriculado em um curso e em determinadas disciplinas: nome do aluno, endereço, telefone, e-mail, nome do curso, nome da disciplina, nome do professor (aplicar na forma desnormalizada até chegar a 3. FN)*

Cenário: Matricula do Aluno

Nome do Aluno: _____ CPF: _____
Telefone do Aluno: _____
e-mail do Aluno: _____
Data Matrícula no Curso: _____
Nome do Curso: _____

Disciplinas a Cursar:

Nome Disciplina	Ano Letivo	Turma	Nome Professor	e-mail Professor
_____	_____	_____	_____	_____
_____	_____	_____	_____	_____
_____	_____	_____	_____	_____
_____	_____	_____	_____	_____

Para este exercício/exemplo, vamos realizar a modelagem da seguinte forma:

a) Uma tabela para todos os dados (forma Desnormalizada)
b) Modelar na 1. forma normal (1.FN)
c) Modelar na 2. forma normal (2.FN)
d) Modelar na 3. forma normal (3.FN)

Para cada alternativa discutir: problemas e alternativas. Ilustrando os motivos de boa modelagem:

a) Evitar redundância dados;
b) Evidenciar outros requisitos e conceitos existentes num problema

Discutir, como na **análise** de um simples cenário, encontra-se detalhes relevantes e fundamentais que justificam um projeto todo.

2.3 Outras definições iniciais

O exercício acima, entre outras coisas, evidenciou outros conceitos básicos a entender:

a) Entidade
b) Atributos
c) Relacionamentos

Entidade: Representa um objeto, um conceito da área a ser análisada, sendo real ou abstrato, contendo características comuns, representadas estas por atributos (um dado específico de uma entidade).

Representa um tipo de informação com conceito próprio e importante para o problema analisado. *Na análise, observe "coisas"/ "substantivos" como candidatas a serem entidades.*

Atributos: Características / propriedades de uma entidade.
Relacionamentos: forma como as entidades se relacionam para compor a informação completa.

Um bom sistema de computador se baseia numa boa definição do banco de dados que ele manipulará (modelo E-R)

O modelo E-R determina o quanto flexível e durável pode ser uma aplicação

Exemplo: cliente com produtos de preferência / cliente com classificação cliente (bom, regular, etc...)

Implicação processo de análise/projeto do sistema (performance, flexibilidade, etc)

Evitar redundância de armazenamento de dados (tabelas repetidas c/nomes diferentes manipulados por sistemas diferentes ou com atributos repetidos e mau distribuídos)

O processo de modelagem de dados é um processo de decomposição das informações, num nível que se possa identificar todo tipo de informação importante e relevante, que isoladamente tem significado próprio.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

2.4 Exercício prático

Realizar a modelagem do seguinte problema: Uma fatura.

Número da Fatura: _____ Vencimento: _____/____/_____

Nome do Cliente: _____CPF:_____
Fone(s): _____ E-mail(s) do Cliente: _____
Endereço Residencial:
CEP: _____ Rua/Av: _____ Nro _____
Complemento: _____ Bairro: _____
Cidade: _____ UF: _____
Endereço Comercial:
CEP, Rua/Av, nro, complemento, bairro, cidade, UF

Data Emissão	Nro Fatura	Vencimento	Valor Fatura	Situação	Nro Nota Fiscal
___/___/____	_____	__/__/____	R$ _____	_____	_____
___/___/____	_____	__/__/____	R$ _____	_____	_____
___/___/____	_____	__/__/____	R$ _____	_____	_____
___/___/____	_____	__/__/____	R$ _____	_____	_____

TOTAL DA FATURA R$ _____

- Faça a modelagem do exercício acima, e simule a situação de que no BD serão cadastrados: 3 clientes, cada cliente terá 2 faturas pelo menos e cada fatura terá 3 locais de compra.

2.5 Banco de Dados e Conceituações

- Definição de banco de dados (BD)
  Um conjunto de dados relacionados, contendo informações do(s) sistema(s) de uma organização qualquer.

  Estes dados relacionados se organizam em forma de: Entidades (tabelas, arquivos)
  Que contém seus atributos
  Relacionamentos

  Definição básica: Entidade ou Conjunto de Entidades
  Atributo (nome, tipo, tamanho)
  Relacionamento
  Exemplo

  Controle de Viagens (C.Comp) – Ver anexo.

Aluno + curso (EE)
Aluno: nome, endereço, telefone, qual curso, qual disciplinas

Empregado tem 1 cargo, 1 cargo vários empregados

SGBD – Sistema Gerenciador de Banco de Dados
Conjunto de dados associados a um conjunto de programas para acesso / manipulação do banco de dados.
O principal objetivo é proporcionar um ambiente conveniente e eficiente para recuperação e armazenamento das informações do banco de dados

***SGBD**:* Conjunto de Software e componentes voltados a gerenciar a base de dados com objetivo de garantir a esta base de dados:

- Segurança,
- Integridade,
- Performance
- Padrão único de acesso ao Banco de Dados independente de tecnologia
- Tratamento de Transação
- Tratamento de Acesso concorrente aos dados

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

2.6 Banco de Dados – Objetivos

- Armazenar dados de forma segura e íntegra;
- Organizar a forma de armazenamento dos dados
  - Modelagem: Entidades, relacionamentos, redundância
  - Nível Físico: Tabelas, organização arquivo físico

Modelo Lógico: Organização dos dados, Entidades / Relacionamentos

Modelo Físico: Forma de armazenamento dos dados, organização do arquivo físico, tipo de implementação de índices.

a) Estrutura básica: Técnicas de Engenharia de Software, analise de sistemas, etc, chegam a uma quantidade de informações básicas da qual subsidiam a criação de um modelo E-R.
b) O sistema é "estruturado" de tal forma, a melhor manter/manipular os dados
A linguagem de programação em si, (você escreve...) emite comandos que manipulam o BD, estes comandos apenas entregam/passam a responsabilidade para o SGBD recuperar ou gravar informações.
Em sistemas, onde não existem a figura de BD, geralmente as informações são gravadas em sistemas de arquivos proprietários e em estruturas facilmente manipuladas apenas pela linguagem de programação, e portanto, neste caso, toda a responsabilidade de recuperação / integridade / etc, passam a ser responsabilidade da linguagem e do programador, o que não é o caso mais indicado.
Exemplo: Cobol (micro)
Clipper c/ Dbase

Exercício: Realizar um exercício de modelagem de dados, caso de estudo simples.

2.7 Objetivos de um Sistema Gerenciador de Banco de Dados

- Um sistema pode gravar/recuperar em arquivos - sistema de processamento de arquivos
- Neste caso, o software escrito é quem se responsabiliza por todos os procedimentos de gravação e recuperação das informações. Isso gera uma série de inconvenientes, que vão desde tornar o projeto mais oneroso (deve-se prever componentes, algoritmos de tratamento de arquivo, etc., no projeto de software) a possibilidade dos dados serem acessíveis apenas para a aplicação específica. Abaixo, trata-se de outros problemas relacionados.

- Há desvantagens no sistema de processamento de arquivo, como:
  - Inconsistência e redundância de dados
    Arquivos criados para um aplicação, desenvolvida numa tecnologia específica necessitam acessar / gravar informações em arquivos criados para outra aplicação desenvolvida em outra tecnologia (linguagem de programação). Como as formas de tratamento podem não ser as mesmas fica inevitável a redundância, replicando-se o arquivo com os dados necessários.
    Inconsistência: qd há dois lugares diferentes para armazenar os mesmos dados, pode acontecer dos mesmos não coincidir.
    Exemplo: Cadastro de empregados para sistema de folha de pagamento e outro para controlar freqüência. Ambos desenvolvido em linguagens e tecnologias diferentes. Quem garante que o nome do empregado está escrito da mesma forma em ambos os arquivos ?

  - Dificuldade de acesso aos dados
    Falta linguagem manipulação tipo SQL – tudo depende do programa / programador
    Extração de dados depende do desenvolvimento de programa apenas;
    Por estar fortemente acoplado a tecnologia de arquivos usada pela linguagem adotada no projeto
    Exemplo: Arquivo COBOL e PASCAL ???
    Impossibilidade de se utilizar softwares como gerador relatórios, pois não há padrão para linguagem de acesso, formas de representar arquivo, etc.
  - Isolamento de dados – *tipos de atributos, formatos diferentes de arquivo*
    O dado / atributo tem uma definição física atrelada ao programa / tecnologia utilizada. Se alterar, por exemplo, um atributo de float para int todos os programas devem ser alterados / compilados.
    Formatos diferentes de arquivo dependetente da tecnologia usada na linguagem.
  - Problemas de integridade - *validação de atributos, relacionamentos, etc.*
  - Problemas de atomicidade – *transação – possibilidade e retornar a transação ao estado anterior em caso de problemas (hardware, integridades, etc) que impeça a conclusão da atualização com sucesso.*
  - Anomalias no acesso concorrente – *lock, unlock,*
  - Problemas de segurança

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

2.8 SGBD X Processamento de Arquivos

O **SGBD** fornece um conjunto de serviços que visa além da armazenagem/recuperação dos dados garantir:

- Forma padrão de acesso, possibilitando assim qualquer software, de qualquer tecnologia acessar/manipular os mesmos dados – facilitar acesso as informações;
- Garantir integridade, acesso concorrente
- Tratar transação
- Segurança de acesso

O **processamento de arquivos**: visa leitura/gravação de dados, normalmente associado a determinada tecnologia (linguagem de programação)

2.9 Exercícios de fixação

a) Baseado nos exercícios anteriores, escreva 3 programas em linguagens diferentes (PASCAL, C++, Java) onde a técnica utilizada para inserir, alterar, excluir e consultar é processamento de arquivos e outros mesmos programas, utilizando-se de SGBD.

b) Realizar modelo de dados (M.E.R) para um Cadastro Técnico de Equipamentos/Produto

Dados: Codigo Produto, nome produto, tipo Produto, Fabricante (nome, endereço, email(s), fone(s), site do fabricante referente ao produto, Características Técnicas do produto (cod, característica, valor, unidade medida, obs da característica), produtos existentes na empresa (nro série, data compra, situação (em operação, estoque, desativado,etc).

2.10 Visão dos Dados

- Benefício BD é proporcionar ao usuário uma visão Abstrada dos Dados – ocultar detalhes sobre forma de armazenamento e manutenção de dados
- Abstração de Dados
  - **Nível Físico** - *como é armazenado, estruturas físicas utilizadas*
  - **Nível Lógico** – *quais dados, entidades e relacionamentos*
  - **Nível de Visão** – *Abstração do nível lógico (M.E.R) utilizado para dar acesso aos dados sob outra perspectiva, tais como:*
    - Segurança (uma visão de uma entidade, sem determinados atributos)
    - Acesso simplificado aos dados (como o M.E.R é decomposto por várias entidades, (3. Forma normal), é possível criar visões na 1.FN ou 2.FN, simplificando o acesso a informação, em determinados usos (softwares para usuário/gestores, geradores de relatórios e consultas ou visão simplificada para outros sistemas que acessem a mesma base de dados).

Exemplos de abstração de dados:
. Nível lógico: um exemplo
. Nível físico: como o banco gravará as informações. Exemplo: tudo num arquivo físico, tipos de dados para os atributos.

. Nível de visão – abstração da camada lógica, utilizado tb para mecanismos de segurança.

(*voltar um ER 3.FN em 1.FN*)

2.11 Definição de Instância e Esquema

- Instância: Conjunto de informações contidas em determinado banco
- Esquema: Projeto geral do banco
- Exemplo: esquema cliente (nome, endereço)
  Instância (Roberto, Av. Andradina 130)
  Exemplo 2: Aluno e curso.

- Independência de Dados: Capacidade de modificar a definição de esquemas em determinado nível sem afetar o esquema do nível superior
- Independência de dados física - modificar o esquema físico sem ter que alterar o programa de aplicação)
- Independência de dados lógica - modificar o esquema lógico sem alterar o programa de aplicação --- depende !!! (criar novos campos, implica em alterar programas que irão manipular estes novos campos)
  Exemplo de # p/ sistemas de arquivos (COBOL micro) e banco de dados

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

2.12 Modelo de Dados

Conjunto de ferramentas conceituais usadas para a descrição de dados, relacionamentos entre dados, semântica de dados e regras de consistência.

2.12.1 Modelo lógico

Usado para descrição de dados no nível lógico e de visões

Modelos incluídos nesta categoria:

Modelo Entidade-Relacionamento
Modelo Orientado a Objeto

2.12.2 Modelo Entidade Relacionamento

Tem por base a percepção do mundo real como um conjunto de objetos básicos, chamados entidades, e do relacionamento entre eles.

Uma entidade é uma coisa, um objeto do mundo real.
Exemplo: Pessoas, conta corrente, empregado, fornecedor, cliente, carro.

Além de relacionamentos, o modelo E-R representa certas regras, as quais o conteúdo do banco de dados precisa respeitar – cardinalidades.

Exemplo: Empregado possui 0 ou n dependentes.

Cardinalidades: quantidade de entidades às quais outra entidade se relaciona.

Podem ser:
1 para 1 (não muito usado)
1 para muitos
muitos para muitos
0 ou 1 para muitos
1 para 0 ou muitos

(muitos: 1 ou mais)

Diagrama E-R é uma forma de representação gráfica do conhecimento que se tem sobre uma realidade qualquer.

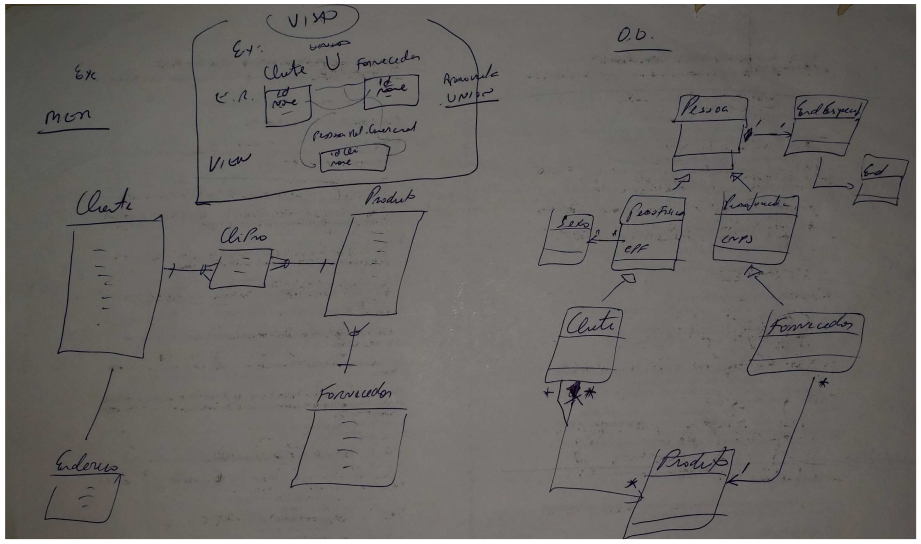
2.12.3 Modelo Orientado a Objetos

Tem por base um conjunto de objetos, onde um objeto possui valores armazenados em variáveis instâncias, e possuem algoritmos que operam este objeto – métodos.
Objetos com mesmo tipo de valores e métodos são agrupados em classes.

--- Neste caso o conceito tem tudo haver com OO --- não entraremos aqui neste mérito.

Ao contrário do modelo E-R, cada objeto possui uma única identidade, independente dos valores neles contidos. Desta forma dois objetos com os mesmos valores, são objetos distintos.

2.12.4 Exemplo M.E.R, Modelo OO e Visão



UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

2.13 Tipo de Banco de Dados

Neste caso estão:

Modelo Relacional [fig 1.3 – 3. Edição]
Orientado a Objetos
Modelo Objeto-Relacional

Modelo de Rede [fig 1.4 – pg. 10 – 3. Edição]
Modelo Hierárquico

Banco de Dados No-SQL

Considerações:

- O mais usado atualmente é o modelo relacional e objeto-relacional.
- O de rede e hierárquico estão em desuso.
- BD NoSQL utilizado em situações específicas onde o uso deste tipo de tecnologia envolvendo fatores como escalabilidade e tipo de informação pode ser mais vantajoso que o uso de um SGBD relacional ou objeto-relacional. Atualmente, utilizado em modelos de software com Arquiteturas de Microserviços e Cloud-computing. (*A realizar pesquisa sobre o tema – A discussão sobre este modelo envolve discussão relacionada a Engenharia de Software / Arquitetura de Desenvolvimento de Software)*

2.13.1 Modelo Relacional

Usa um conjunto de tabelas para representar tanto os dados como a relação entre eles. (Veja fig. 1.3 do livro ou exemplo das viagens) .... Modelo E-R

Modelo de Rede e Hierárquico usam ponteiros e links nas relações entre tabelas, enquanto o relacional se relaciona os registros por valores próprios a eles (citar exemplo ou fig. 1.4 pg. 10 3.Edição)

2.14 Linguagens de Banco de Dados

Dois tipos:

- Linguagens de Definição de Dados

  DDL – *data-definition language*
  Esta linguagem permite implementar o esquema de dados (tabelas e atributos)

  O resultado da compilação dos parãmetros DDLS é um arquivo especial chamado dicionário de dados. Dicionário de dados é um arquivo de metadados – dados a respeito de dados.

  O resultado final de uma DDL é um conjunto de instruções para especificar os detalhes de implementação dos esquemas de banco de dados.

- Linguagem de Manipulação de Dados
  Comandos de manipulação do banco: recuperação, inserção, remoção e modificação.
  Exemplo: SQL

DML – linguagem de manipulação de dados.
Dois tipos DML:
DML procedural – usuário especifica quais dados e como obtê-los
DML não procedural – usuário especifica quais dados sem especificar como obtê-los.

Exemplo: SQL e aqui exige esquema para processamento consultas (***leia mais no capítulo específico do livro: 3 Edição cap. 12, 5. Edição: Cap 13 e 14***) para otimização, pois é mais fácil programar consulta ineficiente.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

2.15 O administrador de Banco de Dados (DBA)

Razão para uso de SGBD é o controle centralizado dos dados e programas de acesso a esses dados.

2.15.1 Funções importantes de um DBA:

- Implementação do esquema no SGBD
  Cria e "executa" a DDL
  O processo de definição do modelo E-R é tarefa do Analista de Sistema / Desenvolvedor.

- Esquema e modificações na organização física
- Autorizações de acesso no SGBD
- Especificações de regras de integridade

- Validar modelos E-R, observando inclusive casos de redundância no armazenamento de dados (tabelas a serem criadas x tabelas já existentes)

- Pode criar versões de banco de dados específicos: Exemplo: Ambiente Desenvolvimento, Ambiente de Produção, etc.

2.15.2 Responsabilidades do Analista de Sistemas

- Definir modelo E-R projeto lógico;
- Indicar no projeto físico, o tipo de dados para cada atributo e seu tamanho (string, numérico, etc.)

2.16 Gerenciamento de transação

- Conceito de transação e atomicidade

2.17 Gerenciamento de Memória

- buffer

2.18 Visão Geral da Estrutura do Sistema

Ver e apresentar figura no livro, fim capítulo 1.

Um sistema de banco de dados é dividio em módulos específicos.
Os componentes funcionais podem ser divididos em:

Componentes de processamento de consultas:

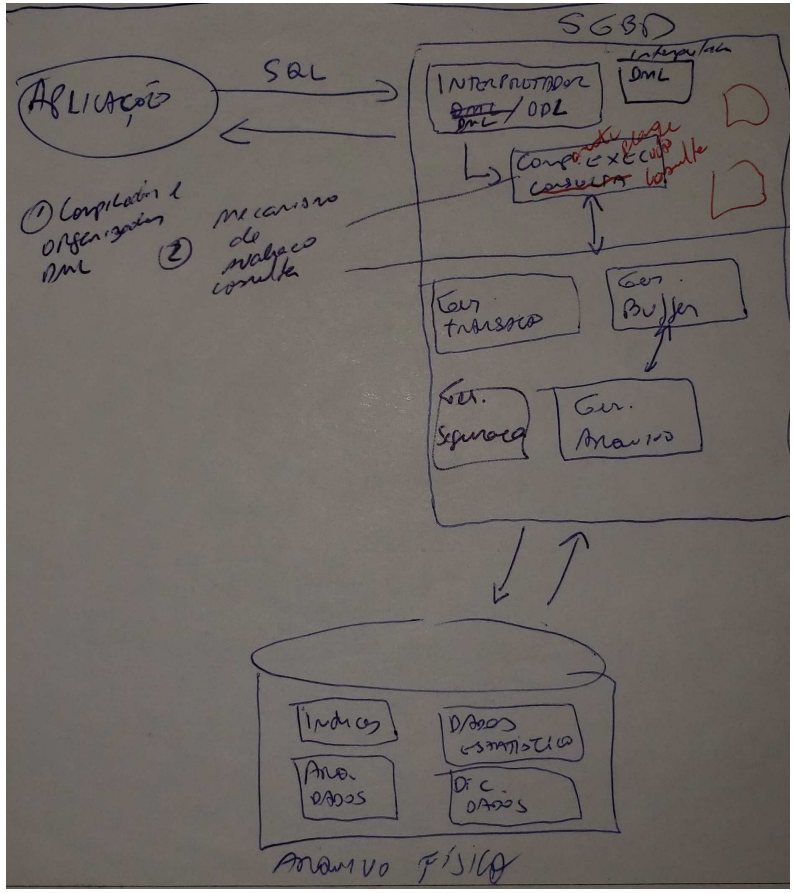
Compilador DML
Pré-compilador para comandos DML
Comandos DML inseriros em programas de aplicação – em linguagem hospedeira.
Interpretador DDL
Componentes para tratamento de consultas
Executam instruções de baixo nível geradas pelo compilador DML

Componentes para administração de armazenamento de dados:

Gerenciamento de autorizações e integridade
Gerenciamento de transações
Administração de arquivos
Gerencia a alocação de espaço em disco e estruturas de dados para representar informações
Administração de buffer
Intermediação dados em disco para memória principal – memória cache.

- Outras estruturas de dados como parte da implementação física:

Arquivo de dados – arquivo que armazena o próprio banco de dados;
Dicionário de Dados
Índices
Estatísticas de dados
Armazenam informações estatísticas quanto aos dados contidos que são usadas pelo processador de consultas para seleção de meios eficientes para execução de uma consulta.



2.19 Exercícios

a) Faça um modelo E-R de uma aplicação:

Um sistema deve gerenciar um cadastro de pessoas para emitir etiquetas, segundo o gosto pessoal por uma linha de produtos específico. Este cadastro deve possuir além do nome da pessoa, o endereço e a preferência por qual linha de produtos.

Modele:
Caso 1: Situação em que o sistema deva gerenciar apenas um tipo de linha de produto por pessoa
Caso 2: A situação em que cada pessoa tem de 1 a 5 preferências por linha de produto
Caso 3: A situação em que cada pessa tem de 1 a n preferências por linha de produto

b) Exercícios para este capítulo, veja:

[3. Edição] Questões 1.1, 1.2, 1.3, 1.4, 1.5, 1.7.

[5. Edição] Questões 1.1 a 1.11

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

3. MODELAGEM DE DADOS

É uma técnica de representação da realidade que mostra o conjunto de dados de um determinado domínio do problema, através de suas entidades e dos relacionamentos existentes entre as mesmas.

Para pensar:

Por que precisamos fazer ?

Representar e organizar as informações (dados) que serão acessados/mantidos para um determinado domínio de problema (requisitos do software e da área do negócio).

Lembrar que:

- Uma organização (empresa) cresce, e portanto a quantidade de dados e formas de controle crescem com o tempo;

Características para elaboração de modelo de dados são:

- Os dados dentro de uma empresa são mais estáveis que os processos
  - Dado é um fator estático e que fica preservado ao longo do tempo, enquanto que os processos – programas – podem se alterar por:
    - Crescimento da empresa;
    - Mudança na política e formas de trabalho;
    - Mudança nas leis.
  - O dado só se altera em casos excepcionais, quando:
    - Muda o negócio da empresa;
    - Muda produtos, serviços e mercados;
    - Objetivos corporativos.
- O mesmo fato físico deve ser expresso com o mesmo valor para todos os usuários dentro da empresa
  - *Redundância, duplicação de tabelas, etc.*
  - Redundância, entre outros fatores, determina fluxo de dados entre as áreas da empresa
- Usuários diferentes podem tratar de formas diferentes os mesmos dados – *saídas de relatórios, etc.*

3.1 MODELO ENTIDADE-RELACIONAMENTO

Conceitos básicos:

[3. Edição] Cap 2. Modelo entidade – relacionamento (pg. 21 a 39 3. Edição /)
Definições básicas
[5. Edição] Leitura cap. 6 – página 133.

Entidade:

Representa um objeto do mundo real, uma coisa real ou abstrata, um grupo de pessoas, um fato ou um elemento do negócio cujas caraacterísticas são interesssantes para a empresa e o sistema a ser desenvolvido deverá processá-lo de alguma maneira. É um objeto distinguivel dos outros através de atributos próprios que o descreve.

Exemplos: Empregado, conta corrente, aluno , cliente, nota fiscal.

Alertar para uma outra visão que é a do livro:

Entidade é um objeto, ou seja cada pessoa de uma empresa.
Um conjunto de entidades é um conjunto que abrange entidades de mesmo tipo que compartilham as mesmas propriedades: atributos.

3.2 Atributo

São os dados que qualificam a entidade, ou seja, as informações que representam um objeto.

Podem ser:

Simples: Dominio de valores bem definidos, uma informação básica e indivisível. Exemplo: numero do documento, sobrenome, nome.

Compostos: ocorre quando um atributo é formado por mais de um sub-atributo:
Exemplo:
Endereco ----> rua, bairro, cidade, cep, complemento...
Numero Documento ----> Ano + número
Nome da pessoa: *primeiro nome, nome do meio, sobrenome (último nome)*

Monovalorados: possui um único valor;

Multivalorados: um atributo pode representar um conjunto de valores: (array)
Exemplo:

Num primeiro momento de análise, definição lógica não modelada, como:
Um empregado possui um atributo *nome_dependente* que possui valores (nomes) dos dependentes do empregado.
Este tipo de atributo depende do banco de dados usado (implementação física)
Exemplo ADABAS: campos periodicos e múltiplos

No processo de modelagem estes tipos de atributos tendem a desaparecer – exceto se o banco prevê este tipo de situação e exista uma decisão de projeto bem justificada, pois seu uso indiscriminado, pode deixar um projeto na 2.Forma normal e gerar dificuldades em consultas.

Nulos: Quando um objeto não possui valor para determinado atributo

Derivado: Atributos cujo o valor, podem ser resultantes de um processo.
Exemplo:
Total_de_vendas_do_produto
Idade (decorrencia data atual – data nascimento).

Neste caso, para banco atributos derivados, podem ser usados para entidades (tabelas) de estatísticas, resumos, etc. para aumentar performance do sistema.

Domínio: São características e possíveis valores que um atributo pode assumir. Faz parte do domínio de um atributo:

- Tipo: número, caracter, data
- Tamanho
- Faixa de Valores
- Conteúdo nulo ou não

Exemplo:

Entidade: Fatura

Atributo	Domínio
Número_Fatura	número inteiro (de 1 a 999999) (N 6 ou int)
Nome_Fornecedor	conjunto de caracteres, (String, 50 caracteres)
Data Vencimento	Data maior que data_emissão_fatura

Metas de modelagem:

- Tornar atributos das entidades: simples, monovalorados e se possível não derivados.

3.3 Chaves

Conceitualmente entidades e relacionamentos são distintos, entretanto na perspectiva do banco de dados, a diferença entre ambos deve ser estabelecida em termos de seus atributos.

Chave principal é **um ou vários** atributos que identifica unicamente um objeto específico (*na visão do livro entidade no seu conjunto de entidades)* num conjunto de entidades.
Exemplo:

Entidade Empregado: ***codigo_emp***, nome, endereço.....

Superchave é um conjunto de vários atributos que compõe uma chave.
Exemplo:

Entidade Dependente_Empregado: ***código_emp, código_dep,*** nome, parentesco....

Porque não o atributo nome como chave ?
O atributo nome nem sempre garante valor único, podem existir homônimos.

Chave candidata são os atributos que possivelmente podem ser a chave.

Nem sempre é obvio, durante a análise determinar qual / quais os atributos chave para uma entidade. Muitas vezes elas não existem no modelo e é necessário criar então (*os famosos: código)*

Chave primária: Chave principal – Seus atributos não podem ter valor nulo

Chave secundária: A nível final de análise / projeto, para efeito de otimização de acessos podem ser criadas outras chaves para a mesma entidade (com os atributos já definidos), entretanto, não tem a mesma finalidade da chave primária.
São outras chaves de acesso que permitem para efeito de performance recuperar/selecionar dados de entidades (registros / objetos) conforme o valor destas chaves.

Características de Atributos que são/compõe a Chave Primária

- Não pode ter valor núlo;
- Não deve ser alterado com o tempo;
- Não pode ter o mesmo valor para objetos diferentes;
- Não pode estar sujeiro a controle de segurança;
- Não recomendável que seu valor ou definição esteja sujeito a fatores externos a organização. Exemplo: número placa do veículo, onde quem define a metodolia é o governo e pode, por exemplo estar sujeito a mudança.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

3.4 Relacionamentos

Relacionamento é uma **associação** entre uma ou várias entidades.

A modelagem de dados é resultado de decomposição da informação, onde cada entidade representa um conceito. Este processo, gera diversas entidades, onde cada uma, conceitualmente representa um tipo de informação relevante para o problema (dentro dos princípios de boa modelagem). Isto significa, análise melhor detalhada e conseqüentemente o problema melhor resolvido – além de outras questões importantes que são discutidas nos tópicos sobre técnicas de modelagem.

Entretanto, toda informação decomposta, deve ser reconstituída. Portanto, é neste momento que devemos analisar os **relacionamentos** entre as entidades que fundamentam os requisitos do problema.

Os relacionamentos podem ser:

- Relacionamento Simples
- Relacionamento com Atributos
- Auto-Relacionamento

Cada relacionamento, tem uma função específica, que é melhor definido indicando um nome para este: **papel.**

Cada relacionamento, possui **cardinalidade**: o número de instâncias (objetos) que cada entidade se relaciona com a outra. Nos próximos tópicos analisaremos este item.

3.4.1 Relacionamento Simples

Empregado ------ possui -------- dependente

Cliente --------- possui ---------- conta corrente

Produto * ---------- 1 fabricante

Maior parte dos relacionamentos são binários, ou seja, entre 2 entidades. Mas é comum, eventualmente surgerem os ternários (3 entidades) e mais raramente + que 3.

Exemplo: Empregado --------- possui --------- Historico de Cargo -------- tem ---------- cargo

Historico de cargo é o resultado de : Empregado + cargo + data promoção.

A entidade Histórico Cargo não existe, se não houver empregado e cargo.

Cliente -------- possui ------- empréstimo, cada conjunto cliente-emprestimo está relacionada a uma agência.

3.4.2 Papel

Papel é o nome que um relacionamento pode assumir entre entidades (ou mesma entidade). Ou seja, é a função que uma entidade desempenha em um relacionamento. É útil para estabelecer um significado de relacionamento quando necessário.

3.4.3 Auto-Relacionamento

Quanto o relacionamento é entre objetos da mesma entidade.

Exemplo:

Empregado --------*gerente* -------- Empregado
Pessoa --------- *conjuge* ------- *Pessoa*

3.4.4 Relacionamentos com Atributos

Um relacionamento pode possuir atributos. Um atributo específico de um relacionamento é um atributo que não faz parte exclusivamente de uma das entidades que formam o relacionamento, mas sim de todas as entidades. Exemplo:

Cliente ---------- deposita ---------- Conta Corrente

(deposita possui: Data de Depósito)

Pode-se assumir que toda vez que um relacionamento possuir atributos ele pode ser mapeado como uma entidade de relacionamento (ou entidade associativa).

Adotar o princípio de que um conjunto de relacionamentos descreve uma ação que ocorre entre entidades.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

3.5 Cardinalidade

Cardinalidade expressa o número de entidades às quais outra entidade pode estar associada num relacionamento.

Pode ser:

- Um para um
- Um para muitos
- Muitos para um
- Muitos para Muitos

Muitos: nenhum ou várias entidades ou um ou várias entidades

Dica:
Um diagrama E-R já normalizado, praticamente é formado por relacionamentos: ***0..1 para muitos***.

O mapeamento destas cardinalidades depende do problema analisado.

Exemplo:

Empregado possui n dependentes, cada dependente possui um empregado.

Observações diversas:

Um para um: Dificilmente existe, quando existir, atentar para ver se as 2 entidades não são uma só.

1 p/ * --- o atributo chave normalmente está do lado *.
Exemplo:
* Produto ----- possuem -------- 1 Fornecedor
Produto: cod. Produto, descrição, ***código_fornecedor***
Fornecedor: ***código_fornecedor,*** *nome, endereço****.***

3.6 Dependência de existência

A existência de um objeto depende da existência de outro objeto.
Ou seja:

Cliente ------ possui ------ Conta_Corrente

A cada objeto (ocorrência) de conta_corrente depedente da existência de um objeto Cliente. Se um cliente não existir ou deixar de existir, a conta_corrente também não existe.

3.7 Entidades Fracas

São entidades que não tem atributos suficientes para formar uma chave primária, neste caso, esta entidade fraca possui atributos de chave primária de uma **entidade forte.**

Exemplo:

Cliente ----- possui ------- pagamento_empréstimo ------ possui ------ empréstimo.

Cliente: ***código_cliente***, nome, endereço
Empréstimo: ***número_empréstimo,*** *data_empréstimo, valor*

Pagamento_empréstimo: ***código_cliente, número_empréstimo****, data pagamento*

*Outro exemplo: empregado (PK: matricula) 1 ---------- * dependente (PK: matricula do empregado, nro dependente)*

Chave estrangeira: Conjunto de um ou mais atributos de uma entidade que são chave primária de outra entidade.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

3.8 Exercícios de modelagem

1. Modele um cadastro de livros e seus autores. Onde cada livro pode ser composto por 1 ou mais autores e um autor pode escrever vários livros. As informações sobre cada livro são: código ISBN, título, sinopse e Editora.

2A. Modele os dados de um cadastro de equipamentos, onde os dados são: código do equipamento e nome do equipamento. Observar que no nome do equipamento o usuário tem o seguinte padrão: "nome do equipamento + modelo", por exemplo: Ventilador XP300, Computador Notebook XP1111.
O modelo deve propiciar ao sistema a possibilidade de emitir uma consulta, onde se liste todos os equipamentos que sejam "computador", "notebook" , "ventilador", etc.

2B. Modele os dados de um cadastro de equipamentos, onde se deseja cadastrar apenas: código do equipamento, nome do equipamento e tipo do equipamento. O tipo refere-se a que o equipamento pode ser: computador, ventilador, etc. No final o modelo deve propiciar ao sistema a possibilidade de emitir uma consulta de equipamentos por tipo de equipamento.

2C. Com base nos exercícios 2A e 2B acrescente a necessidade de relacionar ao equipamento, todas as suas características técnicas. Por exemplo:
Equipamento: XXXXXX
Características
Potência 1000W
Tensão Elétrica: 110/220v
Resolução: 720p, 1080p

3. Modele o cadastro de músicas para CD. É necessário armazenar dados sobre: título do CD, distribuidora, nome das músicas e interprete de cada música. Uma música pode estar presente em outros títulos de CD, bem como um interprete pode cantar diversas músicas.

3.2 Considere vários tipos de mídia: DVD, Blu-ray, HD, distribuição por internet.

4. Crie um cadastro para armazenar clientes, constando: nome e endereço de cada cliente, onde o endereço é composto de: rua, bairro, cidade e estado.

5. Crie um cadastro de pessoas que registre o histórico de endereços da pessoa. O cadastro de pessoas é composto por nome e data de nascimento e o endereço é composto pelos atributos rua, bairro, cidade e estado. É necessário saber a partir de que data ele esteve ou está em tal endereço.

6. Criar um cadastro de equipamentos, contendo nome e modelo do equipamento e uma lista de que materiais compõe este equipamento. Lembrar que um material em especial pode estar presente em equipamentos diferentes, assim como um equipamento possui N materiais.

--- Quantos ventiladores tem ?
--- Quais equipamentos com material X existem ?

7. Criar um cadastro de equipamentos, contendo nome e modelo do equipamento e identificação individual de cada equipamento em especial, onde é necessário registrar número de série e data de compra para o equipamento. Exemplo:

Equipamento: VENTILADOR 110V, Modelo KV 14
Existe: número de série 390293-232-3232 comprado em 01/09/2000
Número de série 239392-2321-1232 comprado em 14/09/2001
... etc.

8. Num sistema, é necessário manter um cadastro detalhado dos equipamentos disponíveis, com suas características técnicas associadas. O cadastro é dado como exemplo abaixo:

Equipamento: NOTEBOOK DELL XPS 300
Características Técnicas:
Memória RAM: 12 Gb
Capacidade HD: 1 Tb
Tipo HD: HD Hibrido
Distribuidor:
RGB Distribuidora de Hardware S/A. / CNPJ / ENDEREÇO / FONES / EMAILS

Faça o modelo de dados, considerando o exemplo acima.

8B. Considerando o exercício 1, considere que cada equipamento na empresa tem o registro de um número de Bem Patrimonial, onde se registra número de série do equipamento (se houver), data da compra e situação do mesmo (Ativo / Inativo / etc).

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

4. MODELAGEM DE DADOS – Parte II

4.1 Metas de Projeto

O conjunto de Entidades e relacionamentos que formam o modelo E-R (ou diagrama E-R) – *com respeito a modelagem de dados* – não são uma coisa precisa, e o muitas vezes há várias formas diferentes de se definir --- dependendo de fatores como a visão do analista de sistemas sobre o problema, etc.

4.2 O uso de Conjunto de Entidades ou Atributos

Exemplo:

Entidade Empregado c/ os atributos: *nome e telefone*

O telefone não poderia ser uma entidade ?

Ficando:
Entidade: Empregado c/ nome
Entidade: Telefone c/ número e localização
Entidade de Relacionamento: empregado_telefone

::: Discutir diferenças e analisar qual seria a melhor solução. Tudo depende do negócio em si.

- Modelo 2 é mais flexível, pois possibilita que um empregado possua + que 1 telefone, etc...

4.3 Diagrama Entidade-Relacionamento

É o diagrama gráfico do modelo de dados, ou seja, do modelo E-R.

Definir simbologia:

- Padrão sugerido pelo autor do livro (3. Edição pg 34/35, 5. Edição: pg. 11 fig. 1.3)
- Padrão UML
- Padrão atribuída a James Martin, conhecida como “pé de galinha”

4.4 Exercícios

a) Faça o diagrama E-R do seguinte problema:

Uma empresa presta serviços a seus clientes. Quando um cliente necessita de um determinado serviço ele preenche um formulário com os seguintes dados:

Nome do cliente, endereço, telefone, descrição básica do serviço e a lista de produtos e quantidades de cada produto a ser usado naquele serviço.

Modele este problema.

O diagrama deve responder:

1. *É possível eu listar todos os clientes e a quantidade de serviços a realizar;*
2. *É possível cada cliente ter vários pedidos de serviços ?*
3. *É possível saber em quais serviços determinados produtos são usados ?*

b) Tendo como base o exercício (a), acrescente:

É necessário saber quando a empresa começou a executar o serviço (data) e terminou sua execução (data final), bem como registrar os diversos passos entre o início e fim da execução de um serviço. Esses passos são informações do tipo data e descrição do que foi realizado.

O diagrama deve responder:

- *É possível saber o "status" do serviço em uma data específica ?*

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

2 Bimestre

5. NORMALIZAÇÃO DE DADOS

Pré-requisitos para o tema:
Ter noção exata de Entidade, relacionamento, CHAVE.
Outros conceitos a serem considerados:
Decomposição, Dependência Funcional e Dependência Transitiva.

Caso um conjunto de dados seja fornecido para ser representado no Banco de dados, como decidiremos sobre a forma lógica mais adequada de estrutura de dados ? É um problema de projeto de banco de dados. O projeto pode ser uma tarefa extremamente complexa, mas a teoria de Normalização é uma ferramenta útil no projeto. **É um mecanismo para que se obtenha as entidades participantes do projeto.**

As regras de Normalização visam evitar que elementos de dados que não pertencem às entidades sejam mantidas ou adicionadas às mesmas.

Modelo de dados normalizado não possui redundâncias (duplicação de dados) acidental. Cada atributo está relacionado com sua própria entidade e não se mistura com atributos relativos à entidades diferentes.

O processo de normalização de dados é um processo de **decomposição,** onde uma série de informações (dados) são decompostas em esquemas (entidades e seus atributos) que se relacionam. Toda informação normalizada deve ser possível de ser recuperada, ou seja, não pode haver perda de junção.

A noção fundamental neste processo é a de **DEPENDÊNCIA FUNCIONAL**.

5.1 Dependência Funcional

Dados os atributos "A" e "B" de uma entidade, diz-se que "B" é funcionalmente dependente de "A" se e somente se, a cada valor de "A" está associado um único valor de "B". Ou seja, se conhecermos o valor do atributo "A" então podemos encontrar o valor de "B" associado a ele.

Exemplo:

Entidade: EMPREGADO

MATRÍCULA ---> NOME_EMPREGADO, SALÁRIO, CPF

Os atributos nome_empregado, CPF, salário dependem funcionalmente do atributo chave MATRICULA. Ou seja, o atributo MATRICULA implica/determina a existência desses atributos.

#F	Nome	Cidade	Cond.
10	Mufato	Foz	10
20	Real	Curitiba	20
30	Parati	Curitiba	5
40	Beal	Cascavel	10
50	Vitória	Ponta Grossa	20

Cada fornecedor tem um número que indica unicamente este fornecedor, um nome, não necessáriamente único; um valor para condição e uma localidade.

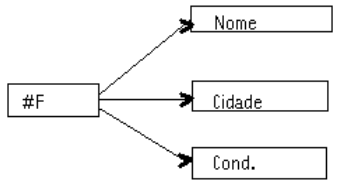


Diagrama de Dependência Funcional

R.X --> R.Y
R.X determina funcionalmente R.Y

5.2 Dependência Funcional Composta

É a mesma situação para dependência funcional vista no item 3.1, considerando-se que o atributo CHAVE é composto. Por exemplo:

Entidade: EMPREGADO

EMPRESA, MATRICULA --> NOME_EMPREGADO, ENDEREÇO_EMPREGADO, SALÁRIO

Os atributos nome_empregado, etc., dependem funcionalmente do conjunto de atributos da chave primária: empresa **e** matricula.

5.3 Dependência Transitiva

Dado os atributos "A" , "B" e "C" de uma entidade, sendo "A" a chave primária, diz-se que "B" e "C" são dependentes transitivos se e somente se, forem funcionalmente dependente de "A" além de existir uma dependência funcional entre eles.

Exemplo:

Entidade: DEPARTAMENTO

CODIGO_DEPARTAMENTO --> NOME_DEPARTAMENTO, SIGLA_DEPARTAMENTO, MATRICULA_GERENTE, NOME_GERENTE

Neste caso o atributo NOME_GERENTE depende do atributo MATRICULA_GERENTE.
O atributo MATRICULA_GERENTE depende da chave primária CODIGO_DEPARTAMENTO.

5.4 PROCESSO DE NORMALIZAÇÃO (1. FN a 3. FN)

O processo de normalização pressupõe que o analista conheça o problema que está sendo analisado e consiga identificar características de todos os dados que o futuro sistema irá gerenciar.

De uma forma resumida o processo até a 3.FN consiste em:

1. FN – Tratar entidades com atributos multivalorados;
2. FN - Tratar entidades onde há dependência funcional entre atributos;
3. FN - Tratar entidades onde há dependência transitiva entre atributos.

5.4.1 PRIMEIRA FORMA NORMAL (1FN)

Uma entidade está na 1FN se ela não contém grupo de atributos repetitivos (multivalorados)

Exemplo de PEDIDO DE VENDA:

Um pedido de vendas possui os dados:
Número do pedido, data pedido, código do cliente, nome do cliente, endereço do cliente, a lista de: (codigo de produto, nome do produto, quantidade pedida, preço do produto, total do produto) e total geral do pedido.

Na estrutura não normalizada teriamos uma tabela ou entidade chamada pedido com os atributos acima:

Cod Pedido	Data Pedido	Cod Cliente	Nome Cliente	End. Cliente	Cod. Produto	Nome Prod.	Qtde	Preco	Total Produto	Total Pedido
1	11/04/2000	2	Ana	Av K, 1	32	CD	3	5	15	35
1	11/04/2000	2	Ana	Av K, 1	45	DVD	1	8	8	35
1	11/04/2000	2	Ana	Av K, 1	18	Modem	4	3	12	35
2	20/04/2000	1	Fernando	Rua Z, 3	18	Modem	1	3	3	9
2	20/04/2000	1	Fernando	Rua Z, 3	32	CD	1	6	6	9
3	15/03/2000	2	Ana	Av K, 1	100	Pentium	2	50	100	106
3	15/03/2000	2	Ana	Av K, 1	101	Papel	2	1	2	106

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

3	15/03/2000	2	Ana	Av K, 1	560	Camisa	2	2	4	106
4	18/04/2000	4	Roberto	Av X, 32	18	Modem	1	3	3	3

Supõem-se que o atributo derivado total do pedido não será gravado no banco uma vez que pode ser gerado via processamento (programa).

Analisando as operações básicas sobre a tabela descobrimos as anomalias desta representação não normalizada:

Inserção não se pode entrar com um novo cliente até que o cliente efetue um pedido de venda. A mesma observação vale para produto.

Remoção se removermos um pedido, destruiremos informações sobre produto e cliente.

Atualização Se for alteração a descrição de um produto ou nome de cliente, deve-se pesquisar seqüencialmente toda a tabela para efetuar a alteração.

Há problemas de armazenamento de informações redundantes, como dados de cliente e produto, entre outros problemas.

A solução para estes problemas é dividir esta relação em duas outras, que denominaremos PEDIDO e PRODUTO-PEDIDO.

Ficando na 1FN assim:

Entidade **PEDIDO**: NUMERO_PEDIDO --> DATA_PEDIDO, CODIGO_CLIENTE, NOME_CLIENTE,
ENDERECO_CLIENTE, TOTAL_PEDIDO.

Entidade **PRODUTO_PEDIDO**: NUMERO_PEDIDO, COD_PRODUTO --> NOME_PRODUTO,
QTDE_PEDIDA, PRECO_PRODUTO

Alguns dos problemas encontrados após solução do modelo na 1 FN:

Inserção não se pode incluir um novo cliente ou produto sem ter um pedido de venda associado

Remoção se removermos um pedido qualquer, perde-se toda a informação de um cliente.

Atualização Se o nome do cliente mudar a necessidade de se acessar sequencialmente todos os objetos da entidade PEDIDO para alterar o nome.

5.4.2 SEGUNDA FORMA NORMAL (2FN)

Uma relação está na 2FN se e somente se ela estiver na 1FN e todos os atributos não chave forem funcionalmente dependentes da chave primária.

Para a 2FN:

Remover os atributos não funcionalmente dependentes de toda uma chave primária.

Exemplo:

A entidade PEDIDO não apresenta este problema, portanto já está na 2FN.

A entidade PRODUTO_PEDIDO já apresenta problemas e precisa remover os atributos.

Entidade **PRODUTO_PEDIDO**: NUMERO_PEDIDO, COD_PRODUTO --> NOME_PRODUTO,
QTDE_PEDIDA, PRECO_PRODUTO

Fica:

Entidade **PRODUTO_PEDIDO**: NUMERO_PEDIDO, COD_PRODUTO --> QTDE_PEDIDA

Entidade **PRODUTO**: COD_PRODUTO --> NOME_PRODUTO, PRECO_PRODUTO

A solução completa ficará na 2FN assim:

Entidade **PEDIDO**: NUMERO_PEDIDO --> DATA_PEDIDO, CODIGO_CLIENTE, NOME_CLIENTE,
ENDERECO_CLIENTE, TOTAL_PEDIDO.

Entidade **PRODUTO_PEDIDO**: NUMERO_PEDIDO, COD_PRODUTO --> QTDE_PEDIDA, PRECO

Entidade **PRODUTO**: COD_PRODUTO --> NOME_PRODUTO

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

5.4.3 TERCEIRA FORMA NORMAL (3FN)

Uma entidade está na 3FN se ela está na 2FN e não possui dependências transitivas.

Uma entidade que está na 2FN pode ter um atributo que não é uma chave mas que por si identifica outros atributos. Refere-se a isto como uma dependência transitiva (veja item 3.3).

Analisando o exemplo do pedido de venda, constata-se que a entidade PEDIDO possui dependência transitiva.

Portanto a entidade PEDIDO que estava assim:

Entidade **PEDIDO**: NUMERO_PEDIDO --> DATA_PEDIDO, CODIGO_CLIENTE, NOME_CLIENTE,
ENDERECO_CLIENTE, TOTAL_PEDIDO.

fica normalizada na 3FN assim:

Entidade **PEDIDO**: NUMERO_PEDIDO --> DATA_PEDIDO, CODIGO_CLIENTE, TOTAL_PEDIDO.

Entidade **CLIENTE:** CODIGO_CLIENTE --> NOME_CLIENTE,
ENDERECO_CLIENTE

Note que o atributo transitivo era o codigo_cliente que não fazia parte da chave primária da entidade PEDIDO na 2FN, e este mesmo atributo determina a existência de nome e endereço cliente, que não são diretamente dependentes do número do pedido.

Todo o exemplo estará na 3FN da seguinte forma:

Entidade **PEDIDO**: NUMERO_PEDIDO --> DATA_PEDIDO, CODIGO_CLIENTE, TOTAL_PEDIDO.

Entidade **CLIENTE:** CODIGO_CLIENTE --> NOME_CLIENTE, ENDERECO_CLIENTE

Entidade **PRODUTO_PEDIDO**: NUMERO_PEDIDO, COD_PRODUTO --> QTDE_PEDIDA, PRECO_NEGOCIADO_PRODUTO

Entidade **PRODUTO**: COD_PRODUTO --> NOME_PRODUTO, PRECO_PRODUTO

Neste modelo já normalizado podemos:

Inserção	Inserir a qualquer momento um novo cliente ou produto.
Eliminação	Se eliminar uma linha de um pedido qualquer não estaremos perdendo informações
Atualização	Se muda qualquer nome de cliente ou descrição de produto, é suficiente acessar o objeto desejado e modificá-lo.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

5.4.4 Outros Ajustes após 3. FN

O processo até a 3.FN não vai garantir 100% de todas as necessidades, mas durante o processo muitos outros detalhes surgem e devem ser considerados, tais como:

- Atributos Derivados – Manter ou Não ?
- Novos Atributos
- Atributos compostos
- Novas entidades
- Atributos que devem ser tratados como uma nova entidade

Exemplo: Numa entidade PRODUTO, se tiver o atributo: nome da Marca, a melhor opção é ter uma entidade MARCA.

- Novas entidades

Por exemplo:

O atributo PRECO da entidade **PRODUTO_PEDIDO** refere-se ao preço do produto, no momento do pedido específico. Mas, se houver a necessidade de saber previamente qual é o preço de lista do produto no momento ?

Então, ajustes no modelo devem ser realizados. Neste caso, é realizado modificação na entidade PRODUTO, conforme exemplo abaixo:

Entidade **PRODUTO**: COD_PRODUTO --> NOME_PRODUTO, **PRECO_PRODUTO**

Outro caso a ser discutido: Atributo Derivado.

A princípio este tipo de atributo é uma informação redundante, pois seu valor pode ser obtido via regra de processamento.

Por exemplo, o atributo TOTAL_PEDIDO na entidade **PEDIDO.**

Neste caso, devemos considerar as seguintes hipóteses:

a) Manter no esquema, e gerenciá-lo;
b) Retirar do esquema e considerar a obtenção deste valor durante processamento (algoritmo de software apenas).

Por exemplo, mantendo o atributo derivado TOTAL_PEDIDO:

a) O software, durante a execução de transação, deve garantir consistência desta informação (ou seja, a soma dos itens **PRODUTO_PEDIDO** deve ser igual ao gravado aqui);

b) É possível que haja diferenças neste caso: Ou por erro de software, tratamento de transação ou exclusão/alteração de um dos itens da entidade PRODUTO_PEDIDO sem os devidos reflexos na entidade PEDIDO;
c) Tem como vantagem ganho de performance em consultas, bem como facilidades na elaboração de consultas diversas;

Sem o atributo derivado, garante que não haja erro de inconsistência, mas poderá gerar algumas dificuldades de consulta, especialmente se desejar disponibilizar as tabelas para um gerador de relatórios/consultas voltado a usuários finais.

Mas, a decisão de eliminar ou não atributos derivados do esquema não é simples, e para cada caso, a uma vantagem ou desvantagem. Não é considerado errado manter atributos derivados no esquema. Mas quando há, deve-se observar que seu valor deve ser gerenciado, garantido pelo software. Outro exemplo: Idade da pessoa, se você já tem a data de nascimento. Se você tem este atributo no banco, deve haver algum processo de atualização diário do banco que sempre calcule e atualize este valor !!!

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

5.5 Outras Metas em Projeto e Modelagem de Dados para banco de dados Relacional

[3. Edição pg 219 a 222 livro,

- Decomposição sem perda na junção

Quando se decompõe informações em entidades e relacionamentos é crucial que a decomposição não resulte em perda de informação.
Ou seja, o projeto do banco de dados de uma aplicação – modelo E-R, deve permitir a "recomposição" - junção dos dados na sua forma original (exemplo: modelo E-R de um pedido de venda deve ser recomposto para poder emitir novamente o mesmo pedido de venda).

- Preservação da dependência

Quando ocorre uma atualização no banco de dados, o sistema deve checar se ela criará uma relação ilegal – isto é, uma relação que não satisfaça todas as dependências funcionais (entre entidades e/ou atributos).

- Evitar redundância de informações

O modelo E-R final não deve permitir o armazenamento de informações redundantes (duplicadas - um atributo ou mais presentes em várias entidades), a menos que isto seja assumido concientemente como uma decisão de projeto.

5.6 Outras dependências

- Pode haver dependências funcionais (de existência) entre entidades. Estas restrições afetam o projeto e o sistema a ser desenvolvido, pois elas tem que ser verificadas.

Exemplo:

Empregado 1 ------*possui* -------- * Dependentes

Não há sentido em haver dependente cadastrado sem o empregado correspondente, mas o contrário não é verdade.

5.7 Outras considerações sobre M.E.R

- O modelo E-R não representa toda a realidade, principalmente considerando-se a parte dinâmica (comportamental) da aplicação. Tais como regras de integridade e restrição. Por isso, o diagrama E-R é um dos vários artefatos de especificação do sistema a ser desenvolvido.

- O modelo E-R possui extensões como generalização e especialização que são ferramentas úteis quando se está analisando ainda num nível mais conceitual do problema (não modelo lógico). Exemplo:

  Modelo conceitual -------- > Modelo lógico --------> Modelo Físico.

- O processo de normalização 3FN é um dos processos úteis de modelagem de dados, existem outras maneiras;
- Nenhum processo de modelagem de dados garante um bom projeto de banco de dados se o analista de sistemas não compreender e analisar adequadamente o negócio (problema) a ser resolvido.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

6. Álgebra Relacional

[3. Edição - pg 69 a 84, 5. Edição: Cap. 2]

a) Linguagem de consultas procedural;
b) Consiste Conjunto de operações tendo como entrada uma ou mais relações e produzindo, como resultado outra relação;

c) Operações fundamentais álgebra relacional:

- Select
- Project
- Union
- Diferença entre conjuntos
- Produto Cartesiano
- Rename

d) Outras operações

- Interseção de Conjuntos;
- Junção Natural;
- Operação de Divisão;
- Operação de Designação.

e) Definição de tupla: é cada linha de uma relação gerada por uma operação em álgebra relacional

[Seguir livro pg 69 3. Edição]

6.1 Operação Select (Seleção)

pg. 69

Seleciona tuplas que satisfazem um dado predicado (condição).

Podem ser utilizados conectivos lógicos, com E($\wedge$) e OU($\vee$)
Operações de comparação: =, $\neq$, <, >, <=, >=

Ex:

$\sigma_{\text{valor} > 1000 \text{ ^ ag\_nome} = \text{"Ponte da Amizade"}}$(Empréstimo)

6.2 Operação Project (Projeção)

pg. 70

Operação que copia uma relação qualquer, selecionando as colunas (atributos) desejados. As colunas que são mantidas possuem seus nomes citados no predicado de projeção π. Opera sobre uma relação, e retorna outra relação com o mesmo número de tuplas, porém com aridade diferente (aridade = quantidade de colunas).

6.3 Operação Relacional de Comparação

Project e Select

--- Quando chegar a finalizar 3.2.13 passar exercício para Project, Select e Operação Relacional de Comparação

6.4 Operação Union

Pg. 72
União de duas relações. Valores duplicados são eliminados.
Duas condições:

a) As relações r e s devem possuir o mesmo número de atributos;
b) Os atributos envolvidos na relação (*r* , *s*) devem ser os mesmos.

6.5 Operação Diferença entre conjuntos

A operação diferença entre conjuntos, representada pelo símbolo: -
permite-nos encontra as tuplas que estão em uma relação, mas não em outra. A expressão *r – s* resulta na relação que contém as tuplas que estão em ***r,*** mas não em ***s.***
[livro 3.Edição SILBERSCHATZ, página 72]

---- Exercícios envolvendo operações acima

6.6 Operação Produto Cartesiano

[pg 73 3. Edição]

Permite combinar informações de duas relações quaisquer. Representamos o produto das relações r1 e r2 por r1 x r2.
Ver fig. 3.12 a 3.14 (livro 3. Edição)

----- Exercícios

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

6.7 Operação Rename

Pg. 76

Permite dar um nome para uma relação gerada por um resultado em álgebra relacional.

Exemplo: executar o resultado da fig. 3.16 (simplificar para:

500, 700, 900, 400

fig. 3.15 ::: 500, 700, 400
fig. 3.16 ::: 900

produto carteziano

conta.saldo	d.saldo
500	500
500	700
500	900
500	400
700	500
700	700
700	900
700	400
900	500
900	700
900	900
900	400
400	500
400	700
400	900
400	400

seleção (conta.saldo < d.saldo)

500
700
400

Operação projeção saldo (conta) - resultado acima

500, 700, 900, 400 - 500, 700, 400

resultado = 900

---- Executar com os alunos ajudando o próximo pg. 77 clientes... (ou forma de exercício)

fig. 3.17 é o resultado

Exercícios para rename

6.8 Operações Adicionais

Pg. 79

6.8.1 Interseção

--- Gera uma relação cujo o conjunto de duas relações diferentes possuam o mesmo valor para uma coluna dada. (fig. 3.18)

6.8.2 Junção Natural

(pg, 79)

operação binária que permite combinar seleções e um produto cartesiano dentro de uma operação

Notar que a junção é realizada pela comparação dos atributos comuns entre as relações.

resolução fig. 3.19 pg. 79

Produto cartesiano

------ fig. 3.12 (pg 74) resposta
------ relaciona em comum o número do empréstimo
------ resultado fig. 3.19

ou o modelo simplificado na fig. 3.19

resolução pg. 81 fig. 3.20

Junção teta: combinar uma seleção e um produto cartesiano em uma única operação;

Exemplo:

$$\pi_{\text{cliente.cll\_nome, cliente.cl\_cidade}} \left(\left(\sigma_{\text{emprestimo.cl\_nome=cliente.cl\_nome ^ emprestimo.ag\_nome='Ponte da Amizade'}} (\text{EmprestimoxCliente}) \right) \right)$$

ou

$$\pi_{\text{cliente.cll\_nome, cliente.cl\_cidade}} (\text{Emprestimo}[x]_{\theta}\text{Cliente})$$

$$\theta = \text{emprestimo.cl\_nome=cliente.cl\_nome ^ emprestimo.ag\_nome='Ponte da Amizade'}$$

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

6.8.3 Operação de Divisão

(pg 81)
r / s --- atributos de s tem que estar em r;

r / s equivale ao seguinte conjunto de expressões fundamentais:

r / s = PROJ atrib r – atrib s (ESQUEMA r) – ((PROJ atrib r – atrib s (ESQUEMA r) X Esquema S) – PROJT r –s, s (ESQUEMA r) pg. 83

Exemplo:

PROJ nome_cliente, nome_agencia (DEPOSITANTE | X | CONTA) dividido
PROJ nome_agencia (SELECT cidade_agencia = “Brooklyn” (AGENCIA))

---- cli, age
RGB Brooklyn
AGN Brooklyn
SAV centro
FDX Unioeste

resultado : CLI
RGB
AGN

Exercício: Simular esta operação utilizando-se dos comandos fundamentais de álgebra relacional.

6.8.4 Operação de Designação

(pg. 83)

6.8.5 Junção Externa (Outer Join)

pg. 92

- Extensão da operação de junção para tratar de informações omitidas.
- Exemplos fig. 3.26 a 3.30
  - JUNÇÀO externa à direita
  - Junção externa à esquerda.
  - Junção externa total

6.8.6 Funções Agregadas

(pg. 94)

--- avg, min, max, sum, count-distinct,
---- veremos em SQL.

6.9 Visões

(pg. 99)

- Qualquer relação que não faça parte do modelo lógico, mas é visível para o usuário como uma relação virtual.

Create view *nome_da_visão AS expressão*

- Indicado criar para consultas.
- Atualizar valores para visões criam problemas para o modelo lógico em banco de dados, pois conforme a visão, pode gerar valores nulos para atributos presente no modelo lógico, mas não ao de visão, por isso, modificações a nível de visão não são permitidas, exceto em casos limitados (pg 101,102).
- É possível criar visões de outras visões

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

6.10 Exercícios com Álgebra Relacional

Exercícios livro: 3.5 e 3.10

Modelo de dados proposto para exercício dos alunos:

Empregado: codigo empregado, nome , cod. Profissão, salário

Profissão: codigo profissão, nome profissão

Dependente: codigo empregado, codigo dependente, nome dep, codigo profissão

Empregado:

1	Roberto	001	1000,00
2	Alex	001	800,00
3	Maria	002	300,00
4	Claudia	005	900,00
5	José	001	2000,00
6	Manoel	004	1500,00

7. Profissão

001 Analista
002 Professor
003 Administrador
004 Engenheiro
005 Bioquímico
006 Farmaceutico
007 Médico

8. Dependente

1	1	Fernando	4	
1	2	Gabriela		5
2	1	Marta	1	
5	1	Ricardo		3

Exercícios para Project, Select, Union, etc., etc.

Exercícios diversos para o aluno descobrir que expressão usar....

Exercícios específicos:

Project e Select

a) Project na tabela empregado para indicar quais são os empregados existentes;
b) Project na tabela empregado que possuem a profissão 001, com os atributos nome do empregado e código da profissão;
c) Project na tabela Dependente para indicar quais são os dependentes código = 1 e que tem a profissão 4;

Operação Union e Diferença entre conjuntos

a) Union do código do empregado da tabela empregado e o código do empregado da tabela dependente;
b) Encontrar quais são os códigos de empregados que não possuem dependentes;

Operação produto cartesiano

a) Produto cartesiano de empregado com tabela de profissões;
b) Produto cartesiano de empregado com os dependentes;
c) Produto cartesiano dos empregados com a sua profissão;
d) Produto cartesiano dos empregados com os seus dependentes;

Operação rename

a) Utilize a operação rename para indicar o empregado que possui o maior salário;
b) Utilize a operação rename para indicar o empregado que possui o menor salário;

Operação Interseção

a) Listar as profissões que possuam empregado;
b) Listar os empregados que possuam dependentes

Operação de Junção Natural

a) Liste os dependentes e o nome do empregado de cada dependente;
b) Liste o empregado e o nome da profissão de cada empregado;
c) Liste os dependentes e o nome do empregado de cada dependente, cujo o empregado possua a profissão 005

---

1. Quais os nomes dos empregados existentes
2. Quais os nomes dos empregados com profissão 1
3. Quais os nomes dos dependentes código 1 com profissão 4
4. Quais profissões que tem tanto para empregado, quanto para dependente
   - a) Só mostrar código da profissão
   - b) O nome da profissão

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

5. quais profissões que não existem para os empregados
   a) Só o código da profissão
   b) Mostrar o nome da profissão
6. Quais os nomes dos empregados que não possuem dependentes

Dado o modelo:

Produto 1 -------- * compras
1 -------- * Vendas

produto (codPro, nomePro)
compras (nroCompra, codPro, qtde, valor)
vendas (nroVenda, codPro, qtde, valor)

Responda em álgebra relacional:

a) Quais são os codPro que tanto há em compras, quanto em vendas ?
b) Quais são os produtos (nomePro) que não tem vendas registradas ?
c) Quais são os produtos (nomePro) que há compras, mas não tem vendas registradas ?
d) Quais são os produtos (nomePro) que não tem nem vendas e nem compras ?
e) Qual o maior valor de venda ?
f) Qual o menor valor de compra ?

7. LINGUAGEM SQL

(Structured Query Language – Linguagem de consulta estruturada)

- Versão original desenvolvida pela IBM
- Inicialmente seu nome era SEQUEL e surgiu no início dos anos 70.
- A linguagem SQL tem diversos tipos de comandos para:
  - DDL
  - DML
  - Definição de Visões
  - Autorização
  - Integridade
  - Controle de transações

DDL: CREATE TABLE, DROP TABLE, ALTER TABLE

DML: SELECT, INSERT, UPDATE, DELETE

Visões: CREATE VIEW

Transações: COMMIT, ROOLBACK

1. Comandos para DDL

   - O SQL DDL permite não só a especificação de um conjunto de relações, como também informações a respeito destas relações, incluindo:
     - O esquema da relação;
     - Domínio de Valores;
     - Regras de Integridade;
     - Conjunto de índices;
     - Informações sobre segurança e autoridade sobre cada relação;
     - Estrutura de armazenamento físico de cada relação no disco.

1.1 Domínios em SQL

- CHAR (n)
- VARCHAR(n) - tamanho variável (character varying)
- Int
- Smallint
- Numeric (p,d) - p número de dígitos, mais o sinal e d número dígitos decimais;
- Real, double precision;
- Float (n) - precisão definida pelo usuário em n dígitos
- Date
- Time.

1.2 Create Domain (pg. 139)

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

1.3 Create Table (pg. 139 a 142)

2. Comandos DML
   - INSERT pg. 130
   - UPDATE pg. 132
   - DELETE pg. 129
   - SELECT pg. 111 a pg. 128

3. Definição de Visão

   - CREATE VIEW pg. 128

Exercícios livro: 4.1, 4.2, 4.3, 4.4, 4.14 e 4.15.

Semestre 2 – 3. A 4. Bimestre

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

8. Extensões Modelo E-R

Cap. 2 - Extensões modelo E-R
(pag. 39 a 59)

- Uso de técnicas de modelagem de dados relacionados a OO.

Generalização e Especialização
Herança
Agregação * (não será tratado)

Nota: (Para banco de dados relacionais e qualquer outro não Orientado a Objetos)
Recursos de especialização, generalização, herança e agregação são modelagens de dados para nível conceitual. É necessário transformar estes modelos em tabelas, ou seja, ir para o nível lógico em forma de tabelas, onde não se usa estas extensões (banco de dados relacionais).

--- desenhar modelo de herança conta ← conta_poupança e ← conta_corrente

Especialização

- Um conjunto de entidades (entidade) pode conter subrupos de objetos que possuem algumas diferenças em relação a outros objetos do mesmo conjunto. Por exemplo: Um objeto pode possuir um atributo que não é compartilhado com os demais objetos do conjunto.

Exemplo:
Entidade: Conta (*número_conta, saldo)*

Especializações:
Conta_poupança (*taxa_juros)*
Conta_corrente (*limite_cheque_especial)*

- Padrão de representação
  Modelo livro pg. 41 figura 2.15 e modelo UML.

Generalização

- Generalização é o inverso da especialização
- Generalização exprime o relacionamento existente entre entidades de nível superior e um ou mais conjuntos de entidades de nível inferior, ou seja, agrupa em uma entidade "pai" características comuns (atributos) a dois ou mais entidades comuns. Estas entidades comuns devem possuir finalidades semelhantes, ou seja, não criar generalização simplesmente para compartilhar atributos comuns como: *descrição, código, nome.*

- Entidade nível superior = superclasses
- Entidade nível inferior = subclasses.
- Abordagem top-down (ver classes nível superior para inferior) e abordagem bottom-up (ver as classes nível inferior e chegar as de nível superior).

- O uso de generalização procede para o reconhecimento de um número de entidades que compartilham características comuns (possuem os mesmos atributos e compartilham dos mesmos relacionamentos)

Generalização e Especialização (opcional)

- Só use especialização para representar distinções entre entidades nível inferior (que possuem atributos diferenciados). Se para o exemplo: *conta_poupança e conta_corrente* não houvesse atributos específicos em cada entidade, não haveria necessidade de se criar a especialização.
- Conceituação semelhante para OO, embora há diferenças de foco sobre este assunto.

Herança

Propriedade criada pela especialização e generalização é herança. As entidades de nível inferior herdam os atributos das entidades de nível superior.

Exemplo: As entidades *conta_corrente* e *conta_poupança* herdam todos os atributos e relacionamentos da entidade de nível superior *conta*.

Decisões de projeto

Transformação modelo conceitual generalização - especialização para o modelo relacional específico para o banco (M.E.R):

Pode haver várias decisões de projeto:
a) Cada especialização é uma entidade no banco, redundando todos os atributos da entidade de nível superior. Neste caso a entidade nível superior desaparece.
b) Cada entidade nível superior e inferior são consideradas entidades. Neste caso há um relacionamento 1 p/ 1 entre entidade nível superior e inferior.
c) Todas as entidades que participam da generalização - especialização é transformada em uma única entidade, diferenciada pelo atributo tipo.

Quando se optar e modelar a entidade de nível superior em separado, ela deve possuir algum atributo, por exemplo, *tipo_*, para diferenciar qual a especialização que o conjunto de dados pode participar.

Restrições

Durante o projeto de banco, pode haver várias restrições, como:

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

a) Uma entidade de nível superior pode participar de nenhuma, uma ou várias entidades de nível inferior. A condição nenhuma se aplica quando não houve atributo específico para criar uma entidade especializada.
b) Alguns objetos de uma entidade especializada podem dinamicamente mudar para outra entidade especializada. Exemplo:
   Entidade nível superior: *Empregado*
   Entidade nível inferior: *Gerente* , *EmpregadoComum*

Agregação

- Neste caso, para modelagem de dados agregação é a abstração por meio da qual os relacionamentos são tratados como entidades de nível superior.
- Exemplo: pg. 45 e 46 livro figura 2.16 e figura 2.17.

Exercícios

Livro: 2.14, 2.15 , 2.16, 2.17 e 2.18

Sugestão de Pesquisa: BD XML, BD Multimedia, Data Services, Virtualização de BD

9. Banco de Dados Baseados em Objeto

[Cap. 8 - Banco de Dados OO pag. 249 a 271 – 3. Edição]

- Implementam conceitos de OO, como:
  - Tipo complexo de dados;
  - Herança

- Implementam o paradigma OO: Classe, Herança, encapsulamento, etc.
- Enfoque do processo de definição do modelo diferente do MER (orientado a dados).

- A tecnologia de banco de dados já pode ser utilizada fora do processo convencional que é processamento de dados. Novas características são introduzidas que um banco moderno tem que atender:

a) CAD - projeto auxiliado por computador;
b) CASE - engenharia de software auxiliada por computador
c) Multimidia;
d) OIS - Sistemas de informação de escritório - textos, planilhas, ferramentas de criação e recuperação de documentos, agenda de compromissos, etc.;
e) Hipertexto - texto enriquecido com links que apontam para outros documentos.

Todos estes novos tipos de aplicações, assim como novos sistemas de uso geral já demandam a utilização de tipos complexos de dados e não somente os básicos (int, char,...).

- Enfoque neste tema para Linguagens de Programação Persistente.

Modelo de Dados OO

A Estrutura Objeto

- Um objeto é uma coisa do mundo real, abstrata ou não
- Paradigma OO está baseado no *encapsulamento* de dados e em código relacionado a um objeto;
- Todas as interações entre um objeto e o resto do sistema são mensagens;
- Um objeto tem associado a ele:
  - Atributos;
  - Mensagens que um objeto responde - cada mensagem pode Ter 0, um ou N parâmetros;
  - Métodos - que possuem código que implementam a mensagem. Um método retorna nenhum ou um valor como resposta a mensagem. Implementam operações específicas que pertencem ao "comportamento" de um objeto.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

- Os métodos de um objeto podem ser somente de leitura - que não afeta o valor de seus atributos - e de atualização, que modificam valores de seus atributos.

Exemplo:

Pessoa ← Empregado (salário, cargo)
<- Cliente (limite_crédito)
atributo: endereço
classe: endereço.

Classe

Agrupam conjunto de objetos similares. Ou seja, possuem a estrutura (atributos, mensagens e métodos) comuns a mesmos tipos de objetos.
Objetos similares indicam:
- Usam as mesmas estruturas de atributos e mensagens.

Exemplo: classe Pessoa, classe Empregado, classe Cliente, classe Endereço.

Herança
Refere-se a hierarquia e é similar a generalização - especialização, onde a classe mais geral possui atributos e métodos comuns para as especializações.
Exemplo:

Pessoa ← empregado
← Cliente

■ outro exemplo, livro pg. 254 fig. 8.1

Um objeto empregado contém todos os atributos e métodos da classe Pessoa, além dos seus específicos, mas não os específicos de cliente.

Herança dá a noção de **reusabilidade,** que é um dos grandes objetos de OO e **polimorfismo** em programação OO.

Polimorfismo:
O polimorfismo permite que diferentes classes possam definir métodos e mensagens do mesmo nome. O comportamento do método, por exemplo, depende da classe de objeto a qual ela está instanciada ou das mensagens enviadas para o objeto. Isto é importante porque permite que se escreva rotinas e métodos que operem com qualquer classe que implemente os métodos necessários, portanto, o polimorfismo aumenta a reusabilidade do código, tornando-o genérico.

Herança Múltipla

Quando uma classe especializada (subclasse) herda de duas classes mais gerais (exemplo: livro pg. 257, fig. 8.4/8.5)
Herança múltipla, pode Ter problemas quando há ambiguidade de atributos ou métodos entre as duas classes e a subclasse, ou seja, se as duas classes (superclasses) possuírem 1 atributo ou método em comum. Neste caso pode haver algumas alternativas:

- Tratar a situação com erro;
- Definir explicitamente na subclasse de quem ele herdará o atributo em comum;

Muitas linguagens não tratam herança múltipla. Exemplo: Java.

Identidade de Objeto

Tipos de identidade:

- Valor: Usado em sistemas relacionais, o valor da chave primária identifica uma tupla.
- Nome: Usado em sistemas de arquivo, o nome do arquivo é o identificador.
- Embutido: Usado em OO. Para cada objeto criado é atribuído automaticamente um identificador. O usuário não fornece nenhum valor.

Em OO os identificadores de objetos são únicos e independem dos valores atribuídos aos atributos do objeto. Considerando-se isto, em sistemas reais, há a necessidade de se criar mecanismos para identificar objetos iguais (do ponto de vista **não** computacional - semelhante a noção de chave primária para o mundo relacional).

Objetos Compostos

São objetos que contém outros objetos como parte de sua especificação.

Exemplo: fig. 8.6, pg. 260

- Noção de ***é parte de***. Em herança uma especialização ***é um***. Em composição um objeto faz parte de outro objeto. Conceitos semelhantes a agregação em OO.
- Composição permite que os dados sejam visualizados por diferentes granularidades entre usuários. No exemplo, um usuário pode manipular e se preocupar apenas com o objeto roda e outro com toda a bicicleta.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

Persistência de Objetos

As linguagens OO possuem mecanismos para definir e criar ojbetos. Todo objeto criado é a princípio ***transiente***, ou seja, fica na memória. O programa termina, o objeto desaparece.
Persistência é poder manter o objeto além da sua "vida" em memória - gravar seus valores em um arquivo, por exemplo e poder recuperar seu estado quando necessário.
Em OO há conjunto de classes de objetos que tratam da persistência, principalmente quando persistência de um objeto não é realizada em banco de dados OO.
Informações sobre o que é e para que serve "camada de persistência" veja site:

- www.omg.org , www.roguewave.com, www.javasun.com.

Ver também: JPA, Hibernate, EJB 3.x, JDO

Abordagem para tornar objetos persistentes:

a) *Persistência por classe*: Declarar explicitamente a classe que é persistente. Abordagem não flexível, pois pode ter a situação de existir objetos transientes e persistentes para a mesma classe.
b) *Persistência por criação*: Dependendo de como o objeto foi criado ele é persistente ou não.
c) *Persistência por marcação*: Todos os objetos são criados como transientes e os que devem ser persistentes são marcados explicitamente.
d) *Persistência por referência*: Um ou mais objetos são explicitamente declarados persistentes. Os outros objetos que são referenciados por estes diretamente ou não, são considerados persistentes.

Identidade de Objetos e Ponteiros

Quando o objeto persistente é criado ele recebe um identificador único no banco OO. Da mesma forma, na memória, para a linguagem OO ele recebe um identificador de memória para este objeto (ponteiro ou referência).
Em muitos casos o identificador do banco não é o mesmo para o da linguagem. O do banco é persistente e o da linguagem transiente e muda toda vez que o objeto for instanciado.
Ponteiro persistente é um tipo de ponteiro que permanece válido após o fim da execução de um programa.

Armazenamento e Acesso a Objetos Persistentes

Os dados, ou seja, o valor de cada atributo de um objeto é armazenado individualmente para cada objeto.
Quanto aos métodos, há as seguintes possibilidades:

a) Armazenar os métodos junto como esquema do banco;
b) Armazenar externamente em arquivos fora do banco para não integrar compiladores de linguagem OO ao banco.

Mapeamento OO em tabelas banco de dados Relacional

Se a persistência de objetos será em banco de dados relacional, após análise e projeto OO há a necessidade de se criar tabelas em banco de dados relacional para as classes OO.

Há várias alternativas, entre elas:

- Toda classe é um tabela;
- Tratar hierarquias de generalização - especialização de modo particular:
  a) Toda estrutura é uma tabela única;
  b) Toda especialização é uma tabela e "absorve" a definição da classe pai;
  c) Cada classe é uma tabela;

- Este tema é um pouco mais complexo e deve ser tratado em aulas específicas de OO.

Exercícios

Livro 3. Edição: 8.2, 8.3, 8.4, 8.5, 8.7, 8.8, 8.9.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

10.Banco de Dados Relacionais – Objeto

[Cap. 9 - Banco de Dados Objeto - Relacionais pág. 273 a 288 3. Edição]

- Estendem o modelo de dados relacional, incluindo OO e acrescentando possibilidade de criação de novas estruturas de dados (não os tipos comuns: int, string)

Exemplo:

Endereco (rua, bairro, cidade)

Cadastro
Nome
Endereco : endereco

- Sistemas relacionais-objeto visam tornar a modelagem de dados e a consulta mais fáceis pelo uso de dados complexos, incluindo dados multimídia.

Comparações:

- *Sistemas Relacionais*: tipos de dados simples, linguagens de consulta poderosas, alta proteção
- *Linguagem de programação simples baseada em BDOOs*: tipos de dados complexos, integração com linguagem de programação.
- *Sistemas relacionais-objeto*: tipos de dados complexos, linguagens de consulta poderosas, alta proteção.

Alta proteção: Isolamento do BD quanto a erros de programação
Linguagens de consulta poderosas: Linguagens declarativas, onde o SGBD define estratégias de consulta com alta performance em função quantidade de registros (objetos)

- Ver livro pg. 273 a 286 e testar em laboratório com PostGres comandos extendidos SQL para entender o assunto.

Exercícios:
9.1, 9.4, 9.5, 9.7

11.TÓPICOS ESPECIAIS

[Cap. 19 - Tópicos Avançados pág. 635 a 668 – 3. Edição]

Segurança e Integridade

- Dados armazenados no banco de dados precisam ser protegidos de acessos não autorizados, destruição ou alteração insidiosa e introdução acidental de inconsistência.

- Perda acidental de consistência de dados pode ser conseqüência de:
  - Quedas durante o processamento de transações;
  - Anomalias causadas por acesso concorrente;
  - Anomalias causadas pela distribuição de dados em diversos computadores;
  - Erros lógicos que violam regras impostas para que as transações preservem as restrições de consistência do banco de dados.

- Formas de acesso insidioso:

  - Leitura não autorizada de dados;
  - Modificação não autorizada de dados;
  - Destruição não autorizada de dados.

- Segurança do banco de dados refere-se à proteção contra acesso insidioso e integridade se refere à precaução contra perda acidental de consistência.

Autorização

a) Para acesso a dados:
- Autorização read
- Autorização insert
- Autorização update
- Autorização delete

b) Para modificação esquemas em banco

- Autorização index
- Autorização resource – permite a criação de novas relações (tabelas)
- Autorização alteration – adição / remoção de atributos
- Autorização drop

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

- Autorizações também podem ser concedidas a nível de visão.

Especificação de segurança em SQL

Grant <lista de privilégio> **on** <nome da relação ou visão> **to** <lista usuários>
[**with grant option**]

Lista de privilégio: delete, insert, select e update
Privilégio Select corresponde a read;

Update – pode ser a nível de atributo. Exemplo:

Grant update (total) on empréstimo to usuario1, usuario2

[with grant option] – permite que um usuário que ganhou privilégio sobre uma relação dê privilégios a outros
Ex: **grant select on** agencia to usuario1 with **grant** option

- Revogação de autorização

Revoke <lista de privilégio> on <nome da relação ou visão> from <lista usuário> [restrict | cascade]

Restrict | cascade – revogação em cascata. Quando um usuario2 teve autorização de usuario1. E usuario1 perdeu privilégio.

- Alteração Senha para usuário (Oracle)

alter user <usuario> identified by <novapassword>

Padronização

1.1 ODBC – Open Database Connectivity

1.2 X/Open XA

1.3 JDBC – Java DataBase Conection

Outros Pontos

Benchmarks

Medidas de desempenho para comparação entre ferramentas, como Banco de dados.

Gargalos
[pg. 652 – 3. Edição]

Pontos específicos que causam problemas de desempenho.
Transações e consultas
Transação exige vários serviços do BD, como:

a) entrada de novo processo no servidor (computador / SO)
b) leitura de disco
c) ciclos de CPU e bloqueio para controle de concorrência

Cuidados específicos: OLTP (processamento on-line de transação) e OLAP (processamento analítico on-line – consultas).

Ajustes de desempenho

- **Parâmetros de hardware** : ajuste memória, cpu, dispositivo de armazenamento;
- **Ajuste de esquemas:**
  - Particionamento de tabelas:
    - Exemplo:
      - Conta (nome_agencia, numero_conta, saldo)
      - Para:
        - Conta_agencia (numero_conta, nome_agência)
        - Conta_saldo (número_conta, saldo)
  - Resolve desempenho quando a grande quantidade de acessos é para saldo, por exemplo - porque? Livro pg. 655
  - Outra situação: Criar relações desnormalizadas, para evitar junções entre tabelas- custo disto: gerenciamento e controle da redundância gerado no sistema.
  - *Desempenho e qualidade de projeto de normalização de dados nem sempre tem soluções compatíveis.*

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

- **Ajuste de índices:**
  - Acelera para consultas, quando ela é o gargalo
  - Prejudica as atualizações, quando ela é o gargalo.
  - Escolha do tipo de índice (hash, b-tree) também influencia.
- **Ajuste de transações (***tema melhor compreendido após estudo sobre transações***)**
  - cuidar para não gerar uma transação com grande quantidade de atualizações, se ela pode ser quebrada em várias transações menores;
  - *conforme o tipo de transação ou consulta é interessante estipular horários específicos para permitir processamento (batch)*
- **Store-procedures**

Banco de dados Ativos

- Executa ações em resposta a eventos.
- **Trigger**
- Cuidados para projetar regras ativas: finalização (disparo de outros eventos, a partir de um evento processado e que pode disparar outros), prioridade, vínculo evento-execução (condição para execução de uma regra) e tratamento de erros.

Exercícios

a) 19.1, 19.2, 19.3, 19.4, 19.5, 19.12, 19.14, 19.15, 19.18. [3. Edição]

b) Criar visões de tabelas já existentes (pegue um MER já implementado no BD e crie uma visão de dados)

c) Implemente controle de segurança num esquema de BD:

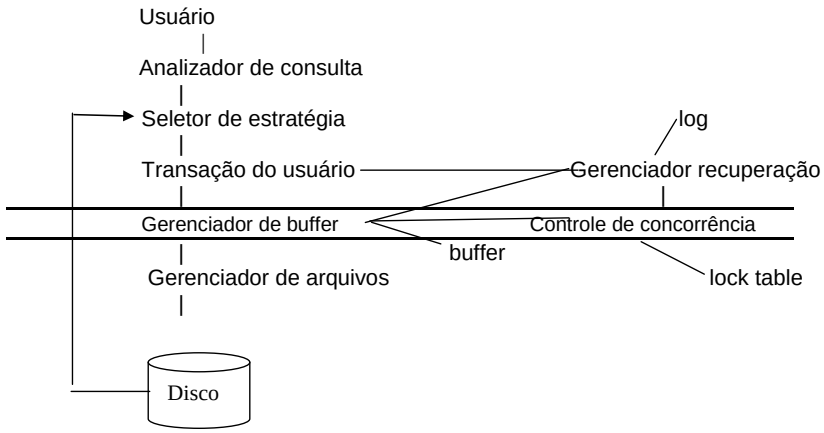
1. grupo de usuários com acesso somente para leitura das tabelas;
2. grupo de usuários que podem atualizar produtos, mas não podem atualizar lançamentos sobre estes produtos (venda, devolução);
3. grupo de usuários com acesso total;
4. permissão para um usuário poder alterar o esquema de banco do sistema quando necessário
5. criar visão para listar quantidade e nome do produto vendido. Esta visão propiciará um comando SQL – SELECT sem junção entre tabelas.
6. Grupo de usuários que tem acesso a tabela produtos, mas nenhum acesso a outras tabelas do sistema.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

12.ESTRUTURA DE ARQUIVOS E DE ARMAZENAMENTO

[Cap. 10 - Estrutura de Arquivos e de Armazenamento pág. 291 a 336 3.Edição]

Estrutura Geral de um SGBD :



Visão BD:

Lógica: Esquema (MER), acesso ao BD
Físico: Como armazenar os dados, estrutura de dados, estrutura de índices.

Meios Físicos de Armazenamento :

- cache
- memória principal
- disco magnético (HD)
- CD, DVD, blu-ray
- Memória flash, SD Card, etc.

  - Discos magnéticos são os principais meios de armazenamento de dados – possui grande capacidade de armazenamento, custo baixo, embora o acesso a dados seja bem mais lento que memória RAM, cache.

Disco rígido: fig. 10.2 – divisão lógica em trilhas e setores.

Medidas de Desempenho de Discos

- tempo de acesso – tempo gasto desde um pedido de leitura / escrita até o início da transferência dos dados. (tempo que o braço – cabeça de leitura/gravação – se move para chegar na trilha e setor correto de acesso)
- tempo de procura – tempo de reposicionamento do braço para chegar em uma trilha qualquer.
- Tempo médio de procura – média estatística dos tempos de procura
- Tempo de latência rotacional – tempo gasto aguardando o acesso ao setor, uma vez que o braço já esteja posicionado na trilha.

Tempo de acesso é a soma dos tempos de procura e tempo de latência.

Otimização de Acesso de Blocos de Disco

- Bloco : seqüência contígua de setores de uma única trilha
- Os dados são transferidos do disco para a memória em bloco

Algumas técnicas para otimização acesso ao bloco:

- - Agendamento : transferir os blocos na ordem na qual eles passarão sob a cabeça de leitura e escrita – algoritmo do elevador;
- Organização do arquivo: Evitar a fragmentação no armazenamento do arquivo.

RAID

- *Redundante arrays of independent (inexpensive – barato) disks*

Técnica de conexão de grande quantidade de discos em paralelo, formando um conjunto único.

Melhoria de confiabilidade por meio de redundância

- Se tem um disco lógico, porém dois discos físicos. Cada disco físico é copia do outro – técnica de *espelhamento ou sombreamento.*

Melhoria no Desempenho por meio de paralelismo

- Pode-se ter acesso simultâneo a vários setores, um setor acessado por disco participante do array.
- Desta forma, os dados de um arquivo pode ser distribuído paralelamente pelos múltiplos discos.
- Grosseiramente, consegue-se tempos menores e maior velocidade, proporcional a quantidade de discos físicos disponíveis
- A distribuição do arquivo pelos discos pode ser: por bit, byte de um setor, setores de um bloco ou blocos.

Níveis de RAID

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

- Espelhamento fornece alta confiabilidade.
- Paralelismo fornece alta taxa de transferência de dados, mas não melhora confiabilidade.

Há várias configurações e arranjos de RAID que pode ser feita e estas técnicas são classificadas em níveis que vão do nível 0 ao 6.

■ estudar no livro páginas 302 a 306 (3. Edição).

Gerenciamento do Buffer
[pg. 308, 3. Edição]

O processo de transferência de dados do arquivo para memória – I/O - (vice-versa) é por bloco.

- O SGBD tem como objetivo minimizar o número de transferência de blocos entre o disco e a memória.
- O sistema *gerenciador de buffer* é responsável pelo processo de alocação / transferência dos blocos de dados do arquivo em memória para o disco e do disco para memória. O objetivo maior é fazer com que sempre que se necessite de um dado ele já esteja presente em memória. (Grosseiramente equivale ao gerenciador de memória virtual do SO)

No algoritmo que envolve a decisão de manter um bloco em memória ou retorná-lo para disco (I/O), leva-se em conta vários requisitos, entre eles:

- Número de acessos – blocos mais acessados;
- Blocos Envolvidos em transação ainda não terminada;
- Término de transação (commit): Forçar bloco para I/O por efetivação de transação (commit) – normalmente os blocos associados a logfile (se o gerenciamento de transação for por log).

Organização de Arquivo
[pg. 311 Edição]

- Um arquivo é organizado logicamente como uma seqüência de **registros.**

- ***Tamanho fixo***
  É a abordagem mais simples de se implementar. Um arquivo armazena uma tabela que contém uma estrutura específica de dados. Cada registro, possui o tamanho em bytes para armazenar esta estrutura, nas diversas instâncias (linhas) que pode possuir esta tabela.
  Exemplo:
  Arquivo conta: nome_agencia (char(22)), numero_conta (char(10)), saldo (real).

- ***Tamanho Variável***
  Para banco de dados pode propiciar:
  a) Armazenar em um arquivo, vários tipos de registros diferentes (várias tabelas diferentes e conseqüentemente com tamanhos diferentes na sua estrutura de dados);
  b) Tipos de registros que permitem tamanhos variáveis para um ou mais atributos;

É a abordagem mais utilizada nos banco de dados modernos, embora possua implementação mais complexa.
Nesta situação, um arquivo para o SO, pode armazenar várias tabelas (entidades) diferentes no BD.

Vantagens:
- *Melhor otimização de armazenamento e leitura de dados que é ocupado pelo arquivo;*

Organização de Registros em Arquivo

a) Arquivo heap: Quando o registro alocado em qualquer lugar, onde haja espaço no arquivo. Nesta abordagem, normalmente há um arquivo por tabela;
b) Arquivo Sequencial : Os registros são armazenados em ordem sequencial, baseado no valor da chave primária - ver fig. 10.15[3.Edição] ou fig. 11.10 e 11.11 [5.Edição];
   Desvantagem: reorganizações futuras

c) Arquivo Hashing: Uma função hasing calcula que bloco o registro será armazenado

d) Organização de Arquivos com Agrupamento de Múltiplas tabelas ou Arquivo Clustering: Diferentes registros (de diferentes tabelas) podem estar no mesmo bloco. Portanto, registro de diferentes relações (tabelas) podem estar no mesmo arquivo. Exemplo: fig. 10.19, [3. Edição, pg 322] ou fig 11.14 e 11.15 [5. Edição, pg 314].

Ilustração: DISCO com N blocos representando o arquivo físico do BD

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

+ 1 bloco e seus vários registros (fixo / variável)
(Organização bloco: seqüencial, hash, lista ligada, arvore, etc.);

Decisões do DBA: num bloco várias tabelas diferentes ou não.

Armazenamento de Dicionário de Dados

Dicionário de dados = conjunto de informações referentes ao esquema (modelo E-R) que está sendo armazenado e gerenciado pelo banco.

O dicionário de Dados armazena informações do tipo:
a) Nome das tabelas (entidades, relações);
b) Nome dos atributos para cada tabela;
c) Domínio e comprimento dos atributos;
d) Nome das visões definas no BD e sua respectiva definição;
e) Requisitos de integridade.

Também:
a) Nome dos usuários autorizados;
b) Informações sobre autorização de acesso para cada usuário.

Pode manter dados estatísticos:

a) Número de tuplas de cada entidade;
b) Método de armazenamento.

Informações sobre índices:

a) Nome do índice;
b) Nome da relação sendo indexada;
c) Atributos que compõe o índice;
d) Tipo de índice formado.

Mapeamento de Objetos em Arquivos

- É similar ao do armazenamento de tuplas para um banco relacional.
- Considerando-se apenas que um objeto pode ter um tamanho muito grande (dados multimídia) e por isso requer tratamento especial nestes casos quanto a gerenciamento de buffer, bloco, etc., para manter performance.

Objetos Grandes

- Campos BLOBs (grandes objetos binários) ou campos compridos para sistema relacional.
- Problemas: campo BLOB muitas vezes pode ultrapassar o tamanho do buffer no BD, e isto torna o gerenciamento buffer mais complexo.

Normalmente, objetos grandes como: texto, gráfico, áudio e vídeo são gravados em arquivos separados do arquivo BD de dados. Neste caso os métodos de atualização desses dados grandes são: checkout / checkin.

Exercícios: 10.1 a 10.5, 10.8 a 10.16

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

13.INDEXAÇÃO E HASHING

[Cap. 11 - Indexação e Hashing pág. 339 a 380 – 3. Edição]

- Índice é uma estrutura cujo o objetivo é fazer com que se localize um registro diretamente. Um registro contendo um objeto de uma tabela com uma chave de acesso cujo o valor é X.

Exemplo: Tabela: Produtos (código, nome, quantidade)
Índice: código.

Estrutura simplificada entrada de um índice

- Chave de acesso: valores de **um ou conjunto** de atributos que formam um índice específico.

- Tipos básicos de índices:

  Índices ordenados
  Índices hash.

- Nenhuma técnica de manter e criar índices é melhor que outra. Mas dependendo da aplicação uma técnica em particular pode ser melhor que a outra. Avaliar as técnicas nos seguintes fatores:
  - Tipo de acesso (por um determinado valor ou faixa de valores)
  - Tempo de acesso
  - Tempo de inserção
  - Tempo de exclusão
  - Sobrecarga de espaço – espaço em disco adicional para armazenar estrutura de índices.

- Uma tabela pode ter diversos índices;

- Cada índice necessita de uma estrutura para seu armazenamento. Qualquer manutenção nos valores de uma tabela, cujo os atributos afetados formam parte do índice, implica em manutenção na estrutura do índice também.

Índices Ordenados

- Um índice armazena os valores das chaves de procura de forma ordenada e associa a cada chave de procura os registros que contêm aquela chave de procura.
- Se o arquivo que contém os registros está ordenado seqüencialmente, o índice cuja chave de procura especifica a ordem seqüencial do arquivo é o índice primário.

Índice Primário

- Arquivos indexados seqüencialmente: arquivos ordenados seqüencialmente por uma chave de procura e que possui um índice primário para o mesmo.
- Exemplo: fig. 11.1 pg. 341

Índices Densos e Esparsos

Denso: Cada chave de acesso possui uma entrada no arquivo índice.
Esparso: Apenas alguns valores de chave de acesso possuem entradas no arquivo índice.

Exemplos: fig. 11.2 e 11.3 pg. 342

Índices secundários

- Estrutura de arquivo de indíce semelhante para índice primário denso. Cada chave de acesso possui entradas no arquivo índice.
- Neste caso o arquivo índice está em ordem seqüencial, mas não o arquivo de dados.

Pergunta: Porque um arquivo índice secundário não pode ser esparso ?

Problemas com arquivo seqüencial:

- Desempenho se degrada conforme o arquivo cresce, a menos que haja reorganizações freqüentes no arquivo, o que não é conveniente, pois isto implica em não deixar disponível o banco durante as reorganizações.

Arquivo de índices em Árvore B+

- Mantém eficiência de acesso.
- Maior sobrecarga ao desempenho para inclusão e exclusão (remoção) dados e requer maior espaço em disco. Entretanto, o desempenho é aceitável e evita reorganizações do arquivo.

Exemplo: fig. 11.8 e 11.9 [3. edição pg. 348]

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

Árvore B semelhante a árvore B+, a diferença é que na árvore B os nós não folha fazem referências também para os blocos com os registros de dados correspondentes. Desta forma, estes valores de chave de procura já referenciadas não aparece nos nós folhas mais [exemplo: 11.18, pg 357 3. Edição].

Hash

- A partir de cálculo sobre o valor da chave de acesso se obtém diretamente o endereço de bloco de disco onde o registro está gravado.
- Cuidados:
  - Algoritmos para função hash, não direcionar muitos registros para o mesmo bloco (bucket) e poucos para outros blocos.
  - Quando um bucket está cheio, ocorre overflow e um novo bucket é criado, formando uma lista ligada entre estes blocos.

Hash estático

Função de cálculo hash, considerando-se que o arquivo tem um tamanho definido. Se o banco crescer, há a necessidade de se reeorganizar a estrutura hash.

Hash dinâmico

Permite modificar a função hash dinamicamente para acomodar o crescimento ou diminuição do tamanho do arquivo.

Hash:

- Grande desempenho para acesso a dados, de uma chave específica - (select x from tabA where a = 2);
- Problemas quando a consulta é por faixa de valores (select x from tabA where a > 1 and a < 100);

exemplo: fig 11.20, 11.21 pg 360/361, 11.22 pg 363

Definição de índice em SQL

Para criar índice:
create [unique] index <nome_índice> on <nome_relação> <lista_atributos>

Para eliminar um índice:
drop index <nome_índice>

Exercícios livro:

11.1 a 11.6 / 11.14

Exercício em sala:

Exercícios modelagem / fazer questões X valendo 1 ponto nota bimestre.
Interpretar e sugerir alterações para o problema X

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

4. Bimestre

14.PROCESSAMENTO DE CONSULTAS

[Cap. 12 - Processamento de Consultas pág. 381 a 438 – 3. Edição]

O processamento de consultas são atividades para se extrair dados de um BD da forma mais eficiente possível, considerando-se a existência de índices para o arquivo, tamanho de bloco, tamanho e organização do arquivo, número de tuplas, tipo e tamanho do registro de dados.

Uma das atividades do processamento de consulta é:

- Traduzir consultas expressas em linguagem de alto nível em expressões que podem ser implementadas no nível físico do BD. Por exemplo, traduzir uma expressão SQL.

Passos envolvidos n processamento de consulta:

a) Análise sintática e tradução;
b) Otimização;
c) Avaliação.

Análise sintática e tradução: Consiste em validar a expressão SQL, conferir a sintaxe, verificar nome das relações, atributos, etc. e traduzir para expressões de algebra relacional fundamental.

Otimização: Visa escolher o melhor caminho e de melhor desempenho na execução de uma expressão SQL.
SQL é linguagem declarativa e o usuário não específica como será a recuperação de dados. Em bancos de dados mais antigos (modelo em rede , hierárquicos e outras variações) ou em banco de dados onde não há o conceito que envolve a linguagem SQL, se utiliza de linguagens procedurais embutidas na linguagem de programação hospedeira. Nesta situação, é responsabilidade do usuário de banco de dados (programador, analista) escrever o algoritmo que garanta a melhor alternativa na recuperação de dados. Exemplo: Natural/Adabas.

Exemplo de situações:

SELECT SALDO FROM CONTA WHERE SALDO < 2500

Pode ser traduzido em:

a) $\sigma_{saldo < 2500}$ (Π_{saldo}(conta)) :: aqui se faz uma projeção e filtra-se após o resultado da projeção;
b) Π_{saldo} ($\sigma_{saldo < 2500}$(conta)) :: aqui durante a execução da projeção se executa a operação de seleção.

Além de alternativas de algoritmos, como:
- leitura seqüencial em toda a tabela;
- Se houver índice, usar o índice para localizar tuplas.

Outro exemplo:

SELECT nome, descrCargo FROM cargo, empregado WHERE empregado.codCargo = cargo.CodCargo

Alternativas:
- Produto cartesiano;
- Junção natural

Alternativas de algoritmos:
- Executa leitura a partir de empregado e do empregado se relaciona com cargo;
- Executa leitura a partir de cargo e do cargo se relaciona com o empregado.

Avaliação: Definir o algoritmo a ser usado para cada operação específica.
Diferentes planos de avaliação, possuem custos diferentes.
SQL não especifica o melhor plano. É responsabilidade do BD determinar qual.

- A decisão sobre usar um caminho de leitura ou outro é baseado em informações estatísticas que o banco mantém;

A otimização de consulta é o processo de selecionar o plano de avaliação de consulta mais eficiente.

Informações para estimativa de custo

Basicamente um BD mantém as seguintes informações estatísticas:

a) Número de tuplas de uma relação (tabela);
b) Número de blocos que contém tuplas de uma relação;
c) Tamanjho em bytes de uma tupla (registro de uma tabela);
d) Fator de bloco – número de tuplas (registros) que cabem em um bloco
e) Etc.

- Manter dados estatísticos causa overhead. Portanto a atualização é periódica, em situações de organização, manutenção, backup, etc., de um banco.

Medidas de Custo

- Acessos a disco : Número de blocos a serem recuperados é o fator mais importante.
- Tempo de CPU para executar consulta
- Custo de comunicação para banco de dados distribuídos

Informações gerais

- Caminhos de leitura para uma tabela leva em consideração tamanho e organização do BLOCO;
- Nem sempre o acesso por índice é o escolhido. As vezes, a leitura seqüencial do arquivo é mais rápido (para SELECT com faixa de valores). Isto devido a quantidade de blocos em que o arquivo (da relação) e do índice estão ocupando. Exemplo: Se a tabela é pequena e ocupa um bloco, a leitura por índice implicará em acessar mais um bloco. Neste caso será mais rápido recuperar o bloco e fazer busca seqüencial, uma vez que o bloco estará em memória.

Cuidados em expressões SQL

As informações abaixo referem-se a maneira como um usuário de banco pode garantir melhor performance no comando SQL e depende da avaliação que o banco a ser usado faz em relação a expressão SQL. Exemplos dados, baseiam-se em como o otimizador de consultas do Oracle trata:

a) WHERE a AND b AND c

Ordem de execução das preposições é de c, b e a , portanto as expressões mais custosas (que impliquem em menor performance) devem ser as primeiras.
Exemplo:

```
SELECT codigo, nome FROM tabelaA WHERE
    (tabelaA.situacao = SELECT situacao FROM tabSituacao) AND
        codigo > 10
```

b) WHERE a OR b

Neste caso a ordem de execução é das preposições a , b.

c) Utilização de índices

Índices são usados se o interpretador de consultas na avaliação da expressão SQL detectar algum indício da necessidade do índice.
Prioridades:

Avaliar primeiro as clausulas WHERE;
Depois ORDER BY
E GROUP BY.

Exemplo:
Tabela TESTE, tem os atributos a, b , c, d. O atributo a é chave primária, b + c compõe um índice secundário, c outro índice, c + a outro índice.

SELECT * FROM TESTE WHERE a > 10 ORDER BY b

O índice a ser usado é o de a.

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

SELECT * FROM TESTE WHERE d = “x” ORDER BY C , A
Neste caso utiliza-se o índice c+a

SELECT * FROM TESTE WHERE d < “a” ORDER BY A , C

Neste caso não é utilizado nenhum tipo de índice.

d) Determinação do melhor caminho

SELECT nomeA, nomeB FROM tabelaA, tabelaB WHERE ...

Se o otimizador não encontrar o melhor caminho para executar o comando SQL (a partir da tabelaA ou da tabelaB) ele opta em ler a partir da última tabela especificada no comando SQL, ou seja, tabelaB. Portanto é interessante, quando se vai executar o comando SQL e se sabe qual o melhor caminho já colocar nesta ordem.

e) Repetição de comandos SQL

Se você vai dar comandos SQL repetidos: SELECT * FROM TABELA, escreva sempre da mesma forma, sem diferenças, pois o banco pode comparar a última expressão executada e utilizar o “seu cache” para retornar a resposta.

SELECT * FROM TABELA é diferente de select * from tabela.

Nota: diferente aqui não significa no resultado do comando.

Conclusão deste tópico

Embora o exemplo dado é para o Oracle, o importante é que o usuário do banco de dados saiba como o “seu” banco se comporta neste assunto para elaborar a expressão SQL que garanta que o banco execute da maneira mais eficiente.

Exercícios

Livro 3. Edição: 12.1, 12.2, 12.3, 12.4, 12.6, 12.7.

15.TRANSAÇÕES

[Cap 13 pág. 441 – 3. Edição]

Transação é um conjunto de operações no banco de dados (inserts, updates, deletes em tabelas) que formam uma única unidade lógica de trabalho. Esta unidade lógica de trabalho é vista do ponto de vista do problema (negócio) a ser analisado e desenvolvido e não uma questão puramente técnica.
Exemplo: Uma transação que envolve transferência de valores de uma conta corrente para outra conta corrente é uma transação única.

É tarefa do banco de dados prover mecanismos que garantam a execução apropriada das transações a despeito de falhas – ou seja, ou a transação é executada por completo ou nenhuma parte dela é executada.

--- Se a transação envolve INSERT em duas tabelas e UPDATE em outras três tabelas, qualquer destes comandos que falhem ou não venham a ser executados a transação toda é abortada.

--- Falhas que podem ocorrer:

- Falhas de hardware;
- Redes de comunicação;
- Falhas de software;
- deadlock
- Etc.

--- Problemas a serem vistos:

- Recuperação ao estado inicial quando uma transação falha;
- Acesso concorrente;
- Etc.

Conceito de Transação

É geralmente resultado da execução de um programa de usuário escrito em linguagem de alto nível (SQL, COBOL, NATURAL, C, etc.) e é delimitada por declarações (ou chamadas de função) que marcam o inicio(begin transaction) e o fim (end transaction) de uma transação.

Para assegurar a integridade dos dados e de toda a transaçãoo sistema de banco de dados deve manter as seguintes propriedades (ACID):

- **Atomicidade:** Ou todas as operações da transação são realizadas ou nenhuma;

A questão de consistência em que a propriendade de atomicidade deve garantir não se refere apenas a consistência a nível de modelo de dados, mas também a de negócio (veja exemplo transferência conta corrente). Pois muitas vezes, numa transação a consistência de dados está OK mas não a de negócio. Exemplo: Se uma transação envolve o débito em uma conta e o crédito em outra, se ela falha após o débito, como fica o crédito. Por isso, neste caso a propriedade de atomicidade deve garantir o cancelamento da parte bem sucedida que foi o débito.

- **Consistência:** A execução de uma transação isolada (sem execução concorrente de outra transação) preserva a consistência do banco de dados (dados gravados válidos para o negócio após execução da transação, além das garantias de integridade referencial do MER). A consistência de uma transação (como negócio) em particular é responsabilidade do desenvolvedor da aplicação.

- **Isolamento:** Embora possa várias transações estarem acontecendo ao mesmo tempo, e freqüentemente na mesma tabela e registro (objeto), o sistema garante que, cada transação não toma conhecimento de outras transações concorrentes no sistema.
  Mesmo asseguradas as propriedades de atomicidade e consistência, quando há diversas transações concorrentes, suas operações pode ser intercaladas de modo inconveniente. Por exemplo:

T1:	read(A)	T2: ....
	A = a + 50	read (a)
	Write (a)	a = a + 20
	Read (b)	write (a)
	B = b – 50	...
	Write (b)	

  Neste exemplo o read(a) está ocorrendo antes da atualização de A na transação T1, quando o write(a) de T2 é executado o valor de **a** não tem incorporado os 50 da transação T1 (*lembre: T1 está sendo executado concorrentemente com T2, possívelmente transações disparadas de máquinas e usuários diferentes e portanto os valores em memória para A são outros...).*

Uma solução para este problema é adotar mecanismos de serialização de transação, entretanto não é a melhor solução, pois possibilitar transações simultâneas é mais performático. Mais a frente será discutido o assunto.

- **Durabilidade**: Transação terminada com sucesso, as mudanças no banco persistem até mesmo se houver falhas no sistema.

Garantir durabilidade:

- Atualizações realizadas pela transação estarem gravadas em disco (ou seja, não mais em buffer de memória);
- Informações gravadas no disco serem suficientes para o banco de dados reconstruir atualizações, quando o BD for reinicializado após falha – exemplo: *logfile.*

Estado da Transação

- Ativa : Estado inicial da transação – transação em execução;
- Em efetivação parcial: Após a execução da última declaração;
- Em falha : Após a descoberta de que a execução normal já não pode se realizar;
- Abortada : Transação desfeita e banco de dados restabelecido ao estado anterior do nício da execução da transação (rollback);
- Em efetivação: Após a conclusão com sucesso (commit)

Implementação de Atomicidade e Durabilidade

- Implementação mais simples: cópias *shadow* – não indicada pois é ineficiente.
  Baseia-se em copia completa do BD antes do início de uma transação.

- Gravação em *logfile* (discutido cap. 15)
  Alterações no banco são registradas em arquivos de logfile de forma que é possível recuperar estado anterior a uma transação em caso de abortagem ou recuperar toda uma transação em caso de queda no banco, etc.

Execuções concorrentes

- Várias transações sendo executadas concorrentemente;
- Duas razões para permitir concorrência de transação (*a despeito das complicações técnicas que isso exige)*:
  - Uma transação consiste em diversos passos. Algumas atividades envolvem I/O, outras atividades de CPU. Atividades de CPU podem ocorrer paralelamente a atividades de I/O em disco;
  - Pode haver várias transações diferentes em execução simultânea no sistema, umas curtas e outras longas.

Portanto executar seqüencialmente transações podem tornar um banco ineficiente.

Exemplo:
Duas transações T1 e T2. T2 só é executada após término de T1.

- Quando várias transações são processadas concorrentemente a consistência do banco pode ser destruída. O sistema de banco de dados deve controlar a interação entre as transações concorrentes para impedi-las de destruir sua consistência.

- Instruções significativas para BD são **read** e **write**, o que existe entre elas são processamentos definidos no programa, com valores em memória no buffer local (o programa, não BD)

Exemplos escalas seqüenciais:

UNIOESTE – Disciplina de Banco de Dados - 3. Ciência Computação
Prof. Roberto Gil Brasil

Figura 13.3 pg. 450 para T1 e T2 modo seqüencial e os valores são: 855 A, 2145 B. Ok.
Figura 13.4 pg. 451 para T1 e T2 A e B são 850 e 2150. Ok.

Exemplos escalas concorrentes:

Figura 13.5 pg.452 – esta escala equivale a fig. 13.3 (Concorrência Serializável)
Figura 13.6 pg. 452 – esta escala tem problemas para valor B. (concorrência não serializável).

Exercícios

Faça uma escala de execução seqüêncial, outra concorrente e outra concorrente causando problemas para o seguinte problema:

a) Uma transação deve somar 10 itens ao valor saldo do produto A (read (produto A)) e retirar 5 itens do valor saldo do produto B.
b) Em paralelo outra transação ocorre retirando 2 itens do valor de saldo do produto A e incluindo 3 itens do valor de saldo do produto B.

Ta

Read (a)
Saldo = saldo + 10
Write (a)
Read (b)
Saldo = saldo – 5
Write (b)

Tb

Read (a)
Saldo = saldo – 2
Write (a)
Read (b)
Saldo = saldo + 3
Write (b)

Recuperação

- Após transações concorrentes o BD precisa ficar em estado consistente.
- Em transações concorrentes, se uma transação Ti falhar, é necessário desfazer seus efeitos garantindo a propriedade de atomicidade. É necessário assegurar que qualquer transação Tj que seja dependente de Ti (Tj leu dados escritos por Ti) também seja abortada. (Exemplificação usando fig. 13.5 pg. 452)
- Nem sempre escalas de recuperação são recuperáveis
  Exemplo: Ti começa, Tj lê dado previamente gravado em Ti e efetiva-se antes de Ti.

Esta é uma situação que o BD não deve permitir

- Deve-se evitar escalas com retorno em cascata. É quando uma transação T10 é desfeita e em conseqüência, leva a desfazer T11 e T12. T11 depende de T10, T12 depende de T11. Figura 13.14 pg. 459
  Esta é uma situação que o BD não deve permitir

Implementação do Isolamento

- Através de mecanismos de bloqueios (LOCK e UNLOCK) e outros mecanismos (ver. Capítulo sobre controle de Concorrência)

Definição de transação em SQL

COMMIT WORK executa efetivação da transação
ROLLBACK WORK aborta a transação

WORK é opcional

Implementação de transação para JAVA:
Em JAVA, com JDBC – autocommit()

Exercícios

13.1, 13.3, 13.4, 13.5, 13.7.

16.Controle de Concorrência

[cap. 14 pg. 471 3. Edição]

Protocolos com Base em Bloqueios (Lock)

- Meio de garantir serialização é obrigar o acesso aos itens de dados seja feito de maneira mutuamente exclusiva.

Bloqueios:

1. **Compartilhado** : LOCK-S
   Uma transação T1 requer este tipo de bloqueio ela pode ler um dado Q, mas não gravá-lo.
2. **Exclusivo**: LOCK-X
   A transação pode tanto ler quanto gravar (escrever) um dado Q.

3. **unlock** – desbloqueia

- Veja matriz fig. 14.1 – pág. 472

Exemplos:

- Fig. 14.2, pg. 474

Exemplo com DeadLock:

- Fig. 14.3 pag. 475

- Deadlocks são problemas inerentes ao bloqueio, necessário se desejarmos evitar estados inconsistentes. Os deadlocks podem ser preferíveis a estados inconsistentes, já que há mecanismos que detectam e tratam os deadlocks por meio de rollback da transação, enquanto estados inconsistentes podem originar problemas reais, não tratados pelo sistema de banco de dados.
  Sobre deadlocks estudar tópito 14.6 do livro pg. 492.

Recuperação após deadlock:

- Após detectado existência de deadlock, o sistema precisa recuperar-se. Três ações são tomadas:
  1. Selecionar uma vítima
     Considerar fatores de custo: como tempo de processamento da transação já gasto, quantidade de itens de dados da transação, etc.
  2. Rollback
  3. Inanição (starvation) – cuidar para o algoritmo não selecionar sempre a mesma transação.

Exercícios:

- Repita o exercício anterior (cap.13) incluindo estas instruções de bloqueio e faça um arranjo que provoque deadlock.

Protocolo de bloqueio em duas fases

a) Fase de expansão: uma transação pode obter bloqueios, mas não pode liberar nenhum.
b) Fase de encolhimento: uma transação pode liberar bloqueios, mas não consegue obter nenhum bloqueio novo.

A partir do momento em que uma transação libera (unlock) um item, ela não pode obter nenhum outro bloqueio sobre outro item.

Exemplos: T3 e T4 (pg. 473 e 474) estão em duas fases.
T1 e T2 não estão.

- Este protocolo não garante ausência de deadlock.

Rollback em cascata, veja exemplo fig.14.4 pg. 477

Como evitar:

- Bloqueio em duas fases severo (strict two-phase locking) – Exige que todos os bloqueios em modo exclusivo sejam mantidos até o final da transação.
- Bloqueio em duas fases rigoroso – Exige que todos os bloqueios (exclusivo ou compartilhado) sejam mantidos até o final da transação.

Os sistemas de banco de dados, em sua maioria utiliza um dos dois protocolos acima.

Exemplo de protocolo em 2 fases:

Transação protocolo bloqueio em duas fases rigoroso:

Transação:
Lock-x(A)
Read (a)
A = a * 2
Write (a)
Lock-x(b)
Read (b)
B = b / 2
Write (b)

Unlock(a)
Unlock(b)
Commit

Transação em protocolo bloqueio em duas fases normal (não rigoroso e nem severo):

Transação:
Lock-x(A)
Read (a)
A = a * 2
Write (a)
Lock-x(b)
Read (b)

Unlock (a)

B = b / 2
Write (b)

Unlock(b)

Commit

- analise as duas transações acima e imagine como seria a execução das mesmas de forma concorrente. Qual das variações deste protocolo teria maior propensão a deadlock ou rollback em cascata ? Explique.

Protocolos com Base em Timestamp (Registro de Tempo)

- Determina ordem de serialização de uma transação usando o método de marcação de tempo.

Marcadores de tempo:

- Valor do relógio do sistema
- Um contador lógico interno incrementado pelo sistema de banco de dados (SGBD)

- W-timestamp (Q) – tempo que qualquer transação executa um write (Q)
- R-timestamp (Q) – tempo que qualquer transação executa um read (Q).

Toda operação de read ou write atualiza o marcador de tempo.

O protocolo de ordenação de marcador de tempo (timestamp) assegura que qualquer operação de read ou write conflitante seja executada na ordem do marcador de tempo.

Operações de Inserção e Remoção

Para efeito de concorrência **insert** e **delete** são tratados como **write**.

Erros lógicos:

- read em um item depois de ter sido removido (delete);
- read em um item antes de ter sido inserido (insert);
- delete em item inexistente (uma transação tentar deletar um item Q já deletado em outra transação).

SQL

SELECT ... FOR UPDATE - executa a operação de leitura realizando um lock-S nas tuplas recuperadas

- Este tipo de instrução não é recomendável, a menos que a lógica da aplicação exija. Exemplo:

```
SELECT...
    ... Algoritmo complexo
UPDATE .... set CAMPO = valor definido
```

UPDATE - O comando update quando é executado no banco já realiza a operação de READ, lock-X e write.

- É a situação mais indicada.
- Exemplo:
- UPDATE ... set SALDO = SALDO + valor

Exercícios:

14.1, 14.2, 14.3, 14.4, 14.6, *14.14*, 14.15, 14.24, 14.25, 14.26.

17.SISTEMA DE RECUPERAÇÃO

[Cap. 15, pág. 511 3. Edição]

- Processo de recuperação do banco de dados ao estado inicial após uma falha;
- Falha:
  - **Falha de transação**: erro lógico (dado não encontrado, limite recurso excedido, violação de integridade, etc.) ou erro de sistema (deadlock).
  - **Queda do sistema**: (hardware , sofware envolvidas em toda estrutura computacional que roda o BD e/ou Aplicativo);
  - **Falha de disco**

Estrutura de Armazenamento

- Armazenamento volátil: Exemplo: RAM
- Armazenamento não volátil: Exemplo: Disco magnético – mas há probabilidade de perda por falha em hardware do disco.
- Armazenamento estável: Possui estratégias de garantia de recuperação da informação, por redundância de informação, backup, etc., Exemplo: RAID.

Acesso de Dados

- A operação de **write** no banco, atualiza um item de dado. Este item de dado está num bloco recuperado do disco (input) e está em buffer (na memória principal). Após a operação de write o bloco que contém o item de dado e que está no buffer é atualizado, **mas não há garantias de que o bloco será gravado imediatamente em disco**, isto é feito em outro momento, quando o gerenciador de buffer determina (veja cap. 10).
- A propriedade de durabilidade só é garantida se o bloco estiver gravado em meio estável de armazenamento.
- Pode ocorrer falhas no processo de transferência do buffer para disco (output).

Recuperação baseada em LOG

- O esquema de log é o mais utilizado.
- O logfile pode ser aproveitado para recuperar todas as transações realizadas entre o último backup e a falha ocorrida, nos casos em que a falha implica em recorrer a backup.
- Registro do log contém os seguintes campos:
  - Identificador da transação
  - Identificador de item de dado
  - Valor antigo
  - Valor novo

- Registros de log:
  - <Ti start> - inicio da transação
  - <Ti, X , V1, V2> - registro da transação: identificador da transação, item de dado, valor antigo, valor novo
  - <Ti commit>
  - <Ti abort>

- Operações que se realizam em log:
  - Undo – inutilizar / desfazer uma transação;
  - Redo – recuperar / refazer uma transação

  - Undo / Redo são idempotente – isto é, executá-la várias vezes é o equivalente a executar uma vez só.

Modificações adiadas do banco de dados

- A atualização na área de dados do banco só se realiza após o final da transação.
- Não há necessidade de operação **undo** para recuperação no banco, apenas **redo**.

- Exemplo: fig. 15.4 pág. 520

Modificação Imediata em Banco de Dados

- A atualização na área de dados do banco é realizada após cada registro no log da operação que está sendo efetuada;
- Executa **undo** / **redo** no processo de recuperacao do banco.

- Exemplo: fig. 15.7 pág. 523

CheckPoints

- Pontos de Controle
- No processo de recuperação o banco consulta o logfile a partir do último checkpoint (processo de recuperação/leitura logfile: fim para o inicio do arquivo logfile, portanto ele começa lendo o último item do logfile indo até o primeiro checkpoint encontrado)

Se o item é gravado na área de dados, porque ainda assim é necessário checkpoint para evitar a execução de redos no logfile ?

Recuperação por reinício

- processo de UNDO deve acontecer antes do processo de REDO na recuperação.
- Motivo: um item de dado OK para redo, pode voltar com estado errado se o UNDO for feito antes

Paginação Shadow

- Não adequada, por causa das execuções de transação concorrente;
- consiste em copias de bloco de disco onde contém o item de dado. Se atualiza na copia shadow. Se a transação estiver OK, o bloco shadow passa a ser a área de dados OK e anterior deve ser eliminada;
- Esta técnica pode causar grande fragmentação de disco, conflitando com conceitos de performance, atualização, etc. já vistos no capítulo 10.

Considerações finais

- O sistema de recuperação de BD tem relação / dependência direta com o sistema de controle de concorrência.
  - Porque: conforme o tipo de controle de concorrência, pode propiciar situações de roolbaks em cascatas e situações em que não é possível dar roolback em cascata pois uma transação menor foi finalizada, antes da transação maior que causou o rollback, provocando neste caso inconsistência no banco (propriedade de Consistência). Lembrar que sempre que o commit é efetivado, o BD não pode perder mais a informação (propriedade de durabilidade).
  - explanar +, + justificativas, ...
  - Por isso, protocolo de bloqueio em duas fases severo ou rigoroso é o + indicado. (porque ?)

Exercícios

Livro 3. Edição: 15.1, 15.2, 15.3, 15.4, 15.5, 15.6.

Exercício prático:

Dada as transações:

T0

Read (A)
A = A + 300
Write (a)
Read (b)
B = b – 300
Write (b)

T1

Read (a)
A = a + 10
Write (a)
Read (b)
B = b – 10
Write (b)

T2

Read (c)
C = c * 2
Write (c)

Valores iniciais: A = 1000, B = 600, C = 200.
Determine um logfile pelo esquema de modificação imediata e adiada para o mesmo. Faça um logfile onde a operação T0 conclua normalmente e T1 não. E descreva o que o processo de undo / redo irá realizar em cada caso (modificação imediata e modificação adiada).

a) Faça escala concorrente para as transações T0,T1 e T2 com protocolo de bloqueio em duas fases;
b) Faça o log adiado e imediato das transações
c) Faça log onde T1 falha após write (b)
d) Para os logs, quem executa REDO e UNDO

2) Porque não há necessidade de realizar a operação undo em log com modificação adiada ?

18.Banco de Dados Distribuídos

[Cap. 16 / 18 pág. 589]

Sistemas Centralizados

Neste tipo de arquitetura o sistema de banco de dados são executados sobre um único sistema computacional que não interagem com outros sistemas.
Neste caso se aplica processamento centralizado, com tecnologia de mainframes, onde há diversos usuários ligados através de terminais.
Outro caso são computadores pessoais monousuários. Neste caso, se aplica banco de dados mais simples que nem implementam controle de concorrência e são sistemas voltados a atender um único usuário.

Cliente / Servidor

Baseado em redes de computadores, com microcomputadores front-end (micro cliente) e back-end (servidores). Os banco de dados para esta tecnologia, atendem a vários pedidos de diferentes clientes. No microcomputador cliente está a aplicação que solicita serviços do servidor de banco de dados (recuperação, transação) e no lado servidor (back-end) está o sistema gerenciador de banco de dados que responde a estas solicitações.

Banco de Dados Distribuídos

- A principal diferença entre sistemas de bancos de dados centralizados e distribuídos é que, no primeiro, os dados residem em um computador específico, enquanto nos distribuídos os dados residem em vários computadores. Entretanto, o SGBD distribuído deve fornecer **transparência**, ou seja, não importa a maneira e o jeito como é distribuído os dados, o sistema deve, na visão de usuário/aplicação comportar-se como se fosse um único conjunto (visão SGBD centralizado).

Site: local de armazenamento dos dados (computador c/ SGBD)

- Cada **nó** (servidor), possui um sistema de banco de dados e é capaz de processar transações.
- Cada nó pode participar de transações globais (que acessem dados de vários nós).

Armazenamento Distribuído dos Dados

- **Replicação**
  - Garante disponibilidade: Se um dos bancos falharem, existe uma réplica.
  - Aumento de paralelismo : possibilita em processos que implique somente em leitura, de processar uma consulta, mais rapidamente, “quebrando os pedidos” pelas réplicas.
  - Aumento overhead – custo para manter a réplica e a segurança de integridade entre as réplicas.

- **Fragmentação**
  - Uma tabela (relação) é particionada em sites diferentes.

- **Replicação e Fragmentação**
  - Combinação dos dois recursos.

- Tabelas pertencentes ao esquema do banco de dados serem disponibilizadas em sites diferentes.

Fragmentação

- **Fragmentação horizontal**

  Quando as tuplas de uma tabela são armazenada em sites diferentes segundo algum critério de seleção.
  Exemplo:
  - No computador servidor de Curitiba são armazenadas as contas correntes de clientes pertencentes a esta região.
  - No computador servidor de Foz do Iguaçu são armazenadas as contas correntes de clientes pertencentes a esta região.

  Veja fig. 18.2 pg. 592

- **Fragmentação vertical**

  Neste tipo de fragmentação há uma decomposição de uma tabela. Ou seja, o conjunto de atributos que pertencem a uma tabela são divididos em outros subconjuntos. Cada subconjunto é implementado em uma tabela. Assim cada tabela fica armazenada em um site diferente. A condição prévia é que deve garantir a junção natural destes subconjuntos para formar a tabela única novamente.

  Exemplo: fig 18.3 e 18.4 pg. 593

- **Fragmentação mista**

  Combinação de framentação horizontal e vertical.

Transparência

- Há a necessidade de se trabalhar com o conceito de **visão** para garantir transparência a nível de usuário em relação aos tipos de fragmentações que podem ocorrer no banco de dados.

Protocolo de Efetivação de Transação para BD Distribuído

Para garantir a atomicidade, nos sites envolvidos na execução de uma transação é necessário estabelecer um protocolo que garanta que todos os sites concluam com êxito uma transação ou não.

- Existe um computador que é o coordenador de uma transação, cuja as finalidades são:
  - Coordenar a execução de várias transações locais e globais iniciadas em seu nó;
  - Adotar os procedimentos de:
    - Iniciar a execução da transação;
    - Quebrar uma transação em subtransações, distribuindo-as nos nós apropriados;
    - Coordenar a conclusão da transação, efetivando ou abortando a transação em todos os nós envolvidos.

Two Phase Commit Protocol
Protocolo de Efetivação em duas fases

Quando a transação (T) termina, isto é, quando todos os sites em que T é executada informam ao coordenador (C1) que T está completa, - C1 inicia o protocolo.

Site:
Fase 1:
- C1 acrescenta o registro <prepare T> no log e força a manutenção no log em memória estável;
- Envia <prepare T> para todos os sites participantes da transação;
- Cada site verifica se é possível completar a transação.
  - Se não for possível, ele adicionará um registro <**no** T> em seu log e envia mensagem de <**abort** T> para C1.
  - Se for possível, adiciona o registro <ready T> em seu log e envia uma mensagem <ready T> para C1.

Fase 2:
- Quando C1 recebe a resposta do prepare T dos sites é determinado se a transação T pode ser efetivada ou não. No caso de todos responderem <ready T> é registrado no log de C1 <commit T> e enviado a todos os sites participantes, caso contrário, é registrado <abort T> em seu log e enviado esta mensagem para os sites participantes.

--- Estudar item: Manuseio de Falhas pag. 607
Veja também: Protocolo de Efetivação em três fases [pg. 610 3. Edição]

- O protocolo de efetivação em duas fases é o mais utilizado.

Desvantagens da distribuição

- Complexidade para assegurar a coordenação entre nós;
- Complexidade de desenvolvimento do software
- Complexidade na administração do banco;
- Maior potencial para erros.

Exercícios

18.1 , 18.2, 18.3, 18.4, 18.5, 18.11, 18.12, 18.16.

Cap. 17, 20, 21, Ap. A e Ap. B

Trabalho:

Exercício 16.3 pg 564

1. O que são banco de dados baseados em sistemas paralelos e explique os diversos tipos de arquiteturas de banco de dados paralelo. Cap. 16/17

Cap. 18:

1. O que são banco de dados distribuídos e no que ele afeta no desenvolvimento de um sistema (em relação ao banco de dados) ?
2. O que é fragmentação horizontal e vertical. Cite exemplos.
3. Desenvolva uma transação distribuída com o protocolo em 2 fases ? Esta transação deve envolver pelo menos a atualização de 3 tabelas em 2 sites diferentes.

Cap. 20: 20.1, 20.2, 20.4

Cap. 21:

1. O que são sistemas de suporte à decisão.
2. O que é data mining ?
3. O que é data warehousing ?

Apêndice A, B:

1. Descreva um banco de dados baseado no modelo de rede.
2. Descreva um banco de dados baseado no modelo hierárquico.
3. Cite diferenças entre banco de dados modelo de rede e hierárquico em relação ao modelo relacional.

19.Exercícios de Modelagem de Dados I

1. Um sistema deve gerenciar um cadastro de pessoas para emitir etiquetas, segundo o gosto pessoal por uma linha de produtos específico. Este cadastro deve possuir além do nome da pessoa, o endereço e a preferência por qual linha de produtos.

   O sistema deverá ser capaz de emitir relatórios de:
   Pessoas por linha de produtos
   Etiquetas para endereço.

2. Modelo um sistema para locadora de veículos:

   Cadastrar o cliente e o carro alugado.
   Considerações sobre o sistema:
   a) Um cliente pode alugar vários carros;
   b) Cada formulário de aluguel para cada carro consta os dados como data do aluguel do veículo, nome do cliente, endereço do cliente, documento identidade, CPF, passaporte, carro pretendido, modelo e marca do carro, ano modelo do carro, placa do carro, quilometragem atual, valor da diária;
   c) Na devolução do carro é informada a quilometragem final e data de devolução;
   d) Emitir histórico de aluguéis de carro por cliente. O sistema deverá ser capaz de indicar o status de cada aluguel: Carro devolvido e não devolvido.
   e) Emitir relatório gerencial, informando num mês específico quantidade de carros alugados.

   Considere que a locadora de veículos é nova e não possui filiais.
   O cliente deverá devolver o carro no local em que alugou o carro.

3. De acordo com o exercício 2, responda --→ em caso negativo, modele os dados para que o sistema consiga fazer:

   a) Indicar para o administrador da locadora a quantidade de carros existentes, disponível para aluguel e alugados ?
   b) Manter uma tabela básica de preço de diária de aluguel para cada modelo de carro ? Considere que a sempre negociação de preço com o cliente.
   c) O sistema consegue indicar se houve algum tipo de sinistro e qual no carro durante o aluguel para um cliente ?
   d) O sistema consegue responder ao administrador a possível previsão de devolução do veículo ?

   Considere a seguinte manutenção no sistema:

   A locadora de veículos expandiu suas atividades, de forma a possuir várias filiais e representações em outras cidades. Portanto, o cliente pode alugar um carro em uma locadora e devolver em outra de sua preferência.

4. Modele uma nota fiscal de venda ao consumidor.

   Considerar que a nota fiscal deverá informar nome do cliente, endereço para entrega, produtos vendidos, quantidade vendida e preço.

Outros requisitos:

a) No momento da venda, o sistema deverá ser capaz de informar o preço de venda de um determinado produto. O vendedor pode negociar este preço com o cliente;
b) O endereço para entrega do produto nem sempre é o mesmo do cliente;
c) Nem sempre é necessário informar o endereço de entrega;

5. Modelo um talão de água:

Um sistema deve gerenciar a conta de água das famílias envolvidas, portanto ele deverá gerenciar os dados de cada responsável pela conta de água (nome, endereço, etc), o consumo de água, valor cobrado, etc.
O sistema deverá ser capaz de:

a) emitir a fatura mensal da conta de água;
b) emitir relatórios por cliente informando o consumo mensal de água num período de um ano;
c) Um cliente pode possuir várias contas de água, sendo cada conta de água num endereço específico;

6. Sugestão de exercício durante o processo de formas normais:

FICHA DE MANUTENÇÃO DE PRODUTOS REALIZADO – Nro: OS _____

Cliente: _____
Data Solicitação: _____

Produto: _____ Marca Produto: _____

Descrição do problema:

Manutenções realizadas

DATA	COD SERV	NOME SERVIÇO	DESCRIÇÃO SOLUÇÃO	VALOR SERVIÇO
____	___	_____	_____	_____
____	___	_____	_____	_____
____	___	_____	_____	_____
____	___	_____	_____	_____

Apresente a solução deste problema evoluindo da 1FN até chegar a 3FN descrevendo problemas e soluções encontradas em cada fase.

6B – Acrescentar nesta OS:

Endereço do Cliente
Endereço de Entrega do Produto
Data Início e Fim dos Trabalhos
Nome do Responsável pela OS
Tipo de Problema

7. Modelo um sistema para locadora de veículos:

Cadastrar o cliente e o carro alugado.
Considerações sobre o sistema:

f) Um cliente pode alugar vários carros;
g) Cada formulário de aluguel para cada carro consta os dados como data do aluguel do veículo, nome do cliente, endereço do cliente, documento identidade, CPF, passaporte, carro pretendido, modelo e marca do carro, ano modelo do carro, placa do carro, quilometragem atual, valor da diária;
h) Na devolução do carro é informada a quilometragem final e data de devolução;

Emitir em SQL as seguintes consultas sobre o problema:

a) Os carros que um cliente já alugou ou está alugando:
Emitir histórico de aluguéis de carro por cliente. O sistema deverá ser capaz de indicar o status de cada aluguel: Carro devolvido e não devolvido.
b) Quantidade de carros alugados num intervalo de data específico.
c) Listar os carros disponíveis para aluguel, não considerando os já alugados no momento;
d) Totalizar o valor arrecadado com aluguel por cliente num determinado período (intervalo de datas);

8. Campeonato de Futebol

- Todos os times e jogadores devem ser registrados na confederação;
- Registrar para cada campeonato, jogos, placar, quem fez o gol;
- Todo time deve ser cadastrado com nome do diretor, técnico e jogadores;
- O time só pode inscrever jogadores registrados na confederação. Cada jogador, possui número de registro, nome, endereço, documento de identidade, e-mail, telefone

O sistema deve prever: quais os jogadores do time X que participaram do campeonato Y ?
Quem foi o artilheiro do campeonato ?
Jogador X joga/jogou em que time no campeonato Y ?

9. Cadastro técnico do Produto

Produto: _____ Tipo Produto: _____

Características Técnicas do Produto:
Característica, Valor, Unidade Medida

Fabricante: _____ fone:_____
Distribuidor: _____ fone: _____
Tempo Garantia: _____ em _____ (dias, meses, anos)

Obs: fabricante, distribuidor possuem endereço
É mantido cadstro clientes especiais, onde é enviado novidades, conforme novos produtos apareçam, conforme preferência do cliente.

10. Criar um cadastro de clientes com nome, endereço e o tipo de produto que o cliente mais compra. Este sistema deve ser capaz de emitir relatórios de: cliente por tipo de produto. Exemplo:

Tipo: XXX
Cliente 1
Cliente 5
Cliente 3
Tipo: YYY
Cliente 4
Cliente 6
Cliente 2

11. Baseado no exercício 1, complemente este cadastro registrando todas as compras que o cliente faz, registrando o tipo de produto, data e valor pago. O sistema deverá permitir que se realize consultas do tipo:

Cliente: xxxxxxxx

Tipo de produto	data	valor
Produto 1	31/08/2000	100,00
Produto 2	20/09/2000	90,00

Totais de compras por cliente no período de xx/xx/xxxx a yy/yy/yyyy

Cliente	Total comprado
Cliente X	190,00
Cliente Y	210,00

Totais de compras por produto no período de xx/xx/xxxx a yy/yy/yyyy

Produto	Valor total
Produto 1	180,00
Produto 2	230,00

12. Criar um banco de dados para implementar um sistema de maladireta. Este sistema deverá registrar dados como nome e endereço de pessoas físicas, empresas, clubes e escolas. O modelo de dados deve possibilitar que o sistema emita relatórios e etiquetas por diversos critérios, como:

- Todos os cadastrados
- Só empresas
- Só para pessoas físicas;
- Só para clubes
- Só para escolas
- Para escolas e clubes.

13. MER para Controlar Torneios de um campeonato

- Relação do time com jogadores, dirigentes, comissão técnica e seus dados pessoais;
- Árbitros
- Campeonato (periodo) e modalidade esportiva envolvida
- Times inscritos no campeonato
- Grupos
- Tabela de Jogos
- Resultados dos jogos
- Quem foi os árbitros
- Dados estatísticos de cada jogo

14. É necessário criar um sistema que gerencie o controle de estoque de uma empresa. Inicialmente este sistema manterá um cadastro simplificado de produtos e permitirá a execução básica de três operações: venda, compra e devolução de venda.
Na operação de venda o usuário informará a quantidade vendida, preço de venda, produto, número da nota e data de venda.
Na operação de devolução de vendas informará a quantidade devolvida, preço de venda de devolução, número nota e data de devolução.
Na operação de compra é informado a data, quantidade, produto, número nota do fornecedor e nome do fornecedor.
Por este sistema o usuário precisará extrair os seguintes dados:

a) Listagem de produtos com nome, preço de venda e quantidade em estoque;
b) Total vendido do produto em um período qualquer;
c) Total comprado do produto em um período qualquer;
d) Total devolvido do produto em um período qualquer;
e) Um histórico de operações realizadas com um produto, classificado por data em um período qualquer.

Conforme problema acima, faça:

- Diagrama E-R na 3. FN;
- Crie o modelo no banco de dados usando SQL;

- Dê os comandos SQL necessários para recuperar as informações pedidas (item a,b,c,d,e);
- Represente os mesmos comandos SQL em álgebra relacional.

15. Modele nas 3. Formas normais, a ficha médica

20.Ficha médica

Nro Paciente:_____

Nome Paciente: _____

Data Nascimento: _____/___/_____ Sexo: _____
Estado Civil: _____ Doc. Identidade: _____
Endereço: _____
Fones:_____
Emails:_____

Lista Consultas realizadas
Nro Consulta / Data / Nome Médico / CRM Médico / Diagnóstico
Nro Consulta: sempre começa em 1 para cada paciente.

Lista Exames realizados

Nro Exame / Data / Nro Consulta Ref. Ao pedido Exame / Nome Tipo Exame
Nro Exame: sempre começa em 1 para cada paciente.

Exclusivo Professor

Resposta 1.7
DEFINIÇÃO DO SISTEMA
a) Levantamento de informações (análise de requisitos // Análise de Uso)
b) Análise do sistema (análise funcional, considerando os objetivos, resultados, etc. //)
PROJETO DE BD
c) Determinar quais são as entidades (arquivos) que compõe o sistema de informações (SI)
d) Determinar quais atributos e seus tipos e consistências específicas
e) Relacionamentos entre as entidades
f) Determinar quais as regras de integridade, chaves de acesso, etc, para manipular o BD.
OUTROS:
g) Verificar após projeto BD se no banco de dados já utilizado já não existem tabelas que são manipuladas por outros sistemas e poderão a vir a ser usados pelo sistema análisado ---- Evitar duplicação / redundância de informações

Resposta 1.2 - Desvantagens Banco de Dados
a) Há banco de dados para todos os tamanhos de aplicação e empresa. A Adoção de um bom banco de dados (o que atende com qualidade a maior parte dos requisitos técnicos necessários) envolve em custo. Portanto, o sistema baseado em processamento de arquivo é mais vantajoso a nível financeiro.
b) Espaço de armazenamento em disco.
c) O processamento a nível de arquivo, por ser específico para a aplicação a ser desenvolvida, tem maior eficiência que o sistema de banco de dados;
d) Não é todo tipo de banco de dados que armazena eficientemente tipos específicos de atributos, como os que envolve uma figura, por exemplo.

Nota: A adoção de um banco de dados envolve mais benefícios que custos. Portanto, na relação custo/benefício, dificilmente um bom banco de dados perderá por um sistema "proprietário" baseado em processamento de arquivo.