INSTITUTO FEDERAL SUL-RIO-GRANDENSE
UNIVERSIDADE ABERTA DO BRASIL
Programa de Fomento ao Uso das
TECNOLOGIAS DE COMUNICAÇÃO E INFORMAÇÃO NOS CURSOS DE GRADUAÇÃO - TICS

Projeto de Banco de Dados Relacional

Rosaura Espírito Santo da Silva

Ministério da Educação
GOVERNO FEDERAL BRASIL PAÍS RICO É PAÍS SEM POBREZA
CAPES
INSTITUTO FEDERAL SUL-RIO-GRANDENSE
UAB UNIVERSIDADE ABERTA DO BRASIL SEED/MEC
TICs

Projeto de Banco de Dados Relacional

SILVA, Rosaura Espírito Santo da

2012/1

Produzido pela Equipe de Produção de Material Didático da
Universidade Aberta do Brasil do Instituto Federal Sul-rio-grandense

PRESIDÊNCIA DA REPÚBLICA

Dilma Rousseff
PRESIDENTE DA REPÚBLICA FEDERATIVA DO BRASIL

MINISTÉRIO DA EDUCAÇÃO

Fernando Haddad
MINISTRO DO ESTADO DA EDUCAÇÃO

Luiz Cláudio Costa
SECRETÁRIO DE EDUCAÇÃO SUPERIOR - SESU

Eliezer Moreira Pacheco
SECRETÁRIO DA EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

Luís Fernando Massonetto
SECRETÁRIO DA EDUCAÇÃO A DISTÂNCIA – SEED

Jorge Almeida Guimarães
PRESIDENTE DA COORDENAÇÃO DE APERFEIÇOAMENTO DE PESSOAL DE NÍVEL SUPERIOR - CAPES

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA SUL-RIO-GRANDENSE [IFSUL]

Antônio Carlos Barum Brod
REITOR

Daniel Espírito Santo Garcia
PRÓ-REITOR DE ADMINISTRAÇÃO E DE PLANEJAMENTO

Janete Otte
PRÓ-REITORA DE DESENVOLVIMENTO INSTITUCIONAL

Odeli Zanchet
PRÓ-REITOR DE ENSINO

Lúcio Almeida Hecktheuer
PRÓ-REITOR DE PESQUISA, INOVAÇÃO E PÓS-GRADUAÇÃO

Renato Louzada Meireles
PRÓ-REITOR DE EXTENSÃO

IF SUL-RIO-GRANDENSE
CAMPUS PELOTAS

José Carlos Pereira Nogueira
DIRETOR-GERAL DO CAMPUS PELOTAS

Clóris Maria Freire Dorow
DIRETORA DE ENSINO

João Róger de Souza Sastre
DIRETOR DE ADMINISTRAÇÃO E PLANEJAMENTO

Rafael Blank Leitzke
DIRETOR DE PESQUISA E EXTENSÃO

Roger Luiz Albernaz de Araújo
CHEFE DO DEPARTAMENTO DE ENSINO SUPERIOR

IF SUL-RIO-GRANDENSE
DEPARTAMENTO DE EDUCAÇÃO A DISTÂNCIA

Luis Otoni Meireles Ribeiro
CHEFE DO DEPARTAMENTO DE EDUCAÇÃO A DISTÂNCIA

Beatriz Helena Zanotta Nunes
COORDENADORA DA UNIVERSIDADE ABERTA DO BRASIL – UAB/IFSUL

Marla Cristina da Silva Sopeña
COORDENADORA ADJUNTA DA UNIVERSIDADE ABERTA DO BRASIL – UAB/IFSUL

Cinara Ourique do Nascimento
COORDENADORA DA ESCOLA TÉCNICA ABERTA DO BRASIL – E-TEC/IFSUL

Ricardo Lemos Sainz
COORDENADOR ADJUNTO DA ESCOLA TÉCNICA ABERTA DO BRASIL – E-TEC/IFSUL

IF SUL-RIO-GRANDENSE
UNIVERSIDADE ABERTA DO BRASIL

Beatriz Helena Zanotta Nunes
COORDENADORA DA UNIVERSIDADE ABERTA DO BRASIL – UAB/IFSUL

Marla Cristina da Silva Sopeña
COORDENADORA ADJUNTA DA UNIVERSIDADE ABERTA DO BRASIL – UAB/IFSUL

Mauro Hallal dos Anjos
GESTOR DE PRODUÇÃO DE MATERIAL DIDÁTICO

PROGRAMA DE FOMENTO AO USO DAS TECNOLOGIAS DE COMUNICAÇÃO E INFORMAÇÃO NOS CURSOS DE GRADUAÇÃO –TICs

Raquel Paiva Godinho
GESTORA DO EDITAL DE TECNOLOGIAS DE INFORMAÇÃO E COMUNICAÇÃO – TICS/IFSUL

Ana M. Lucena Cardoso
DESIGNER INSTRUCIONAL DO EDITAL TICS

Lúcia Helena Gadret Rizzolo
REVISORA DO EDITAL TICS

EQUIPE DE PRODUÇÃO DE MATERIAL DIDÁTICO – UAB/IFSUL

Lisiane Corrêa Gomes Silveira
GESTORA DA EQUIPE DE DESIGN

Denise Zarnottz Knabach
Felipe Rommel
Helena Guimarães de Faria
Lucas Quaresma Lopes
EQUIPE DE DESIGN

Catiúcia Klug Schneider
GESTORA DE PRODUÇÃO DE VÍDEO

Gladimir Pinto da Silva
PRODUTOR DE ÁUDIO E VÍDEO

Marcus Freitas Neves
EDITOR DE VÍDEO

João Eliézer Ribeiro Schaun
GESTOR DO AMBIENTE VIRTUAL DE APRENDIZAGEM

Giovani Portelinha Maia
GESTOR DE MANUTENÇÃO E SISTEMA DA INFORMAÇÃO

Carlo Camani Schneider
Efrain Becker Bartz
Jeferson de Oliveira Oliveira
Mishell Ferreira Weber
EQUIPE DE PROGRAMAÇÃO PARA WEB

SUMÁRIO

GUIA DIDÁTICO GD

APRESENTAÇÃO

Prezado(a) aluno(a),

Olá, sejam bem-vindos à disciplina de Banco de Dados Relacional. Ela tem por objetivo apresentar os conceitos principais utilizados em BD, bem como possibilitar que ao final do estudo os alunos sejam capazes de construir uma base de dados.

Com o grande volume de dados existente hoje no mundo é necessário que esta informação possa ser armazena, administrada e acessada por um grande número de pessoas. Cada vez mais pessoas têm acesso à informação, há alguns anos não se imaginava que teríamos uma ferramenta que tornasse possível a comunicação intercontinental de forma rápida e barata. Com o surgimento da Internet, as distâncias diminuíram, o conhecimento se disseminou e a informação ficou acessível a um número muito maior de pessoas.

Quem de nós já não fez uso desse recurso? Seja para auxiliar em um trabalho escolar, ou esclarecer uma dúvida ou aprender algum conteúdo, isso sem falar nas redes sociais, compras pela internet, e-mails, blogs, etc.

Pois bem, a Internet hoje se utiliza da tecnologia de banco de dados para conseguir gerenciar todo o volume de dados a que nós usuários da rede acessamos.

Quando efetuamos uma simples busca na rede estamos acessando alguma base de dados, que mantém os dados organizados. Dessa forma, quando entramos com alguma palavra de busca em algum site, essa palavra ou conjunto de palavras pode ser encontrada dentro de um documento ou página que está disponível na rede.

Nas empresas, esse avanço também possibilitou que fossem quebradas barreiras físicas, hoje uma empresa não tem que ter um endereço físico (como é o caso de algumas empresas virtuais), é possível trabalhar com inúmeras filiais como se fosse uma só. A tecnologia de BD possibilitou que uma única base de dados contenha todos os dados da empresa, facilitando o acesso e evitando a redundância.

Esses são alguns dos motivos que fazem com que os profissionais de banco de dados sejam reconhecidos no mercado de trabalho. Para manter o negócio da empresa competitivo, temos que manter a informação acessível, de forma rápida e segura e possibilitar que o acesso aconteça de lugares diversos.

Bom trabalho a todos.

Objetivos

Objetivo Geral

- Ao final desta disciplina o aluno será capaz de elaborar projeto conceitual, lógico e físico de um banco de dados, implementar um projeto de banco de dados usando Linguagem de Descrição de Dados, implementar consultas a banco de dados usando Linguagem de Manipulação de Dados e implementar restrições de integridade de um banco de dados usando a linguagem de descrição de dados, funções e gatilhos.

Objetivos Específicos

- Identificar os objetivos, as vantagens e as desvantagens do uso de Sistemas Gerenciadores de Banco de Dados (SGBD);
- Reconhecer os componentes funcionais de um SGBD;
- Identificar a Hierarquia de Abstrações de Dados;
- Identificar os tipos de usuários de um SGBD;
- Identificar as funções do administrador de um Banco de Dados;
- Identificar Restrições de Integridade;
- Compreender o Paradigma da Transação como Garantia de Correção do BD;
- Compreender a garantia de Atomicidade e de Durabilididade;
- Compreender o controle de Concorrência e a Serializabilidade;
- Criar Diagrama Entidade-Relacionamento (DER);
- Produzir um modelo lógico de um sistema pela efetivação de relacionamentos, identificando cardinalidade as normalizações necessárias;
- Criar a estrutura de tabelas e relacionamentos em um Sistema Gerenciador de Banco de Dados;
- Garantir a Integridade de domínio e referencial necessária;
- Compreender a Linguagem de Consulta Padrão para SGBD´s Relacionais;
- Criar os grupos de usuários necessários, fornecendo direitos de acesso conforme sua função dentro de um sistema de informação;
- Criar usuários vinculando-os a um determinado grupo;
- Automatizar procedimentos em nível de SGBD que garantam a segurança e a confiabilidade dos dados;
- Efetuar testes de funcionamento do SGDB alocado;
- Desenvolver Consultas SQL para SGBD´s Relacionais de forma otimizada;
- Realizar procedimentos de cópia de segurança e restauração de cópias, garantindo a confiabilidade da estrutura dos dados de um banco.

Referências

DATE, C. J.; *Introdução a Sistemas de Banco de Dados*, Editora Campus, 8. Ed. 2004.

ELMASRI, Rames; NAVATHE, Shamkant B. *Sistema de Banco de Dados*. Rio de Janeiro, 4ed.: LTC, 2000.

GONZAGA, Jorge Luiz, *Dominando o PostgreSQL*.Rio de Janeiro: Editora Ciência Moderna Ltda, 2007..

HEUSER, Carlos Alberto. *Projeto de Banco de Dados*. 4ª ed., Porto Alegre: Sagra Luzzatto, 2001.

KORTH, Henry F., SILBERSCHATZ, Abraham. *Sistema de bancos de dados*. 3. ed. São Paulo : Makron, 1999.

NEVES, Denise Lemes Fernandes. *PostgreSQL*: Conceitos e Aplicações – São Paulo: Érica, 2002

NETO, Álvaro Pereira. *PostgreSQL*: Técnicas Avançadas – Versões OpenSource 7.X. 1ª ed., Érica, 2003.

NIEDERAUER, Juliano, *Guia de Consulta Rápida PostgresSQL*. Novatec Editora Ltda.

OLIVEIRA, Celso Henrique Poderoso de. *SQL*: Curso Prático. São Paulo: Novatec, 2002.

SILBERSCHATZ, A.; KORTH, H.; SUDARSHAN, S. *Sistema de Banco de Dados*. 5ª ed. Rio de Janeiro: Campus, 2006.

PostgreSQL Prático - http://pt.wikibooks.org/wiki/PostgreSQL_Prático - Acesso em 05/2011

http://pt.wikibooks.org/wiki/PostgreSQL_Prático/DCL/Administração_de_usuários,_grupos_e_privilégios – Acesso em 10/2011

A

Introdução

Unidade A
Projeto de Banco de Dados Relacional

UNIDADE A

INTRODUÇÃO

Banco de dados (BD) pode ser entendido como qualquer sistema que reúna e mantenha organizada uma série de informações relacionadas a um determinado assunto em uma determinada ordem.

Uma lista telefônica é um exemplo, nela percebemos que todos os dados referentes a uma pessoa estão em uma mesma **linha**, a isso chamamos **registros**.

Um BD é uma coleção de dados com algum significado inerente, é projetado, construído e "alimentado" com dados correlatos e servem a um propósito específico.

Dado x Informação x Conhecimento

Dado

Registro de um fato, cadeias numéricas ou alfanuméricas que não possuem significado associado. **Dados:** conjunto de valores.

- Ex.: Matriz de valores contendo as compras efetuadas por um cliente

Informação

Dado que foi **processado** de forma a se tornar relevante para uma determinada pessoa ou organização. **Informação:** dados associados ao seu significado.

- Ex.: Matriz com valores e o significado de cada coluna.
  Coluna 1 = nome do cliente
  Coluna 2 = valor da compra

Conhecimento

Ato ou efeito de abstrair ideia ou noção de alguma coisa. **Conhecimento:** há um entendimento sobre o significado dos dados e é possível extrair conhecimento a partir dos mesmos.

- Ex: Valor total das compras de um cliente em um determinado período de tempo.

Nos dias atuais, o sucesso de uma organização depende:

- Da capacidade de adquirir dados de forma correta com rapidez;
- Da capacidade de gerenciar os dados de forma efetiva;
- De utilizar os dados para agregar valor ao seu negócio.

Definições

Base de Dados (BD)

É uma coleção de dados relacionados e armazenados em algum dispositivo. Genericamente pode ser qualquer conjunto de dados como, por exemplo: uma agenda com os endereços de pessoas conhecidas, uma lista de elementos, um livro, apontamentos.

O objetivo de criar e manter uma BD são poder obter e utilizar os dados lá guardados: procurar a morada

de uma determinada pessoa, saber o que foi dito nas aulas sobre um tema.

Base de Dados

É uma coleção de dados relacionados e armazenados em algum dispositivo.

- Representação é livre
- Arquivos texto
- Às vezes provê informação

Banco de Dados

É uma coleção de dados relacionados, os quais são fatos que podem ser gravados e que possuem um significado.

- Obrigatoriamente provém informação;
- Dados são representados segundo um padrão;
- Pressupõe um sistema de gerenciamento.

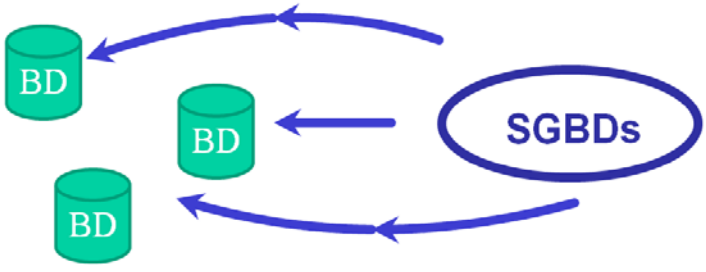
Representação gráfica de uma Base de Dados.

Gerência de Dados

- Suporte à extração do conhecimento a partir de bancos de dados;
- Suporte à manipulação dos dados, garantindo consistência e integridade dos dados;
- SGBDs: Sistemas de Gerência de Bancos de Dados.

SGBDs

Coleção de programas que permitem ao usuário definir, construir e manipular Bases de Dados para as mais diversas finalidades.



Autoinformação

Estas informações são armazenadas no catálogo do SGBD, o qual contém informações como a estrutura de cada arquivo, o tipo e o formato de armazenamento de cada tipo de dado, restrições, etc.

Usuários

Pessoas envolvidas, desde o projeto, uso, até a manutenção. Dividem-se em três categorias: causais; novatos e sofisticados.

Usuários causais

Acessam o banco de dados casualmente, mas que podem necessitar de diferentes informações a cada acesso e utilizam sofisticadas linguagens de consulta para especificar suas necessidades.

Usuários novatos

Utilizam porções pré-definidas do banco de dados, usando consultas pré-estabelecidas que já foram exaustivamente testadas (programas).

Usuários sofisticados

São usuários que estão familiarizados com o SGBD e realizam consultas complexas.

Administrador de Dados (AD)

Desenvolve e administra centralizadamente estratégias, procedimentos, práticas e planos capazes de disponibilizar os dados corporativos necessários, quando necessários, com integridade, privacidade, documentação e compartilhamento.

Participa dos levantamentos de dados e as regras de negócio da empresa. Elabora e/ou acompanha a confecção de modelos. Participa da compatibilização do planejamento de sistemas com os modelos lógicos.

Administrador de Banco de Dados (DBA)

Em um ambiente de banco de dados, o recurso primário é o banco de dados por si só e o recurso secundário o SGBD e os softwares relacionados. A administração destes recursos cabe ao Administrador de Banco de Dados, o qual é responsável pela autorização de acesso ao banco de dados e pela coordenação e monitoração de seu uso, bem como da criação das estruturas, restrições e integridades, definidas no projeto.

Linguagem de Consulta de Estruturada (SQL - Structured Query Language)

Para que possamos criar a estrutura de um banco de dados, controlar e manipular seu conteúdo, é necessário que exista uma ou mais linguagens que trabalhem com estas situações. Existem três definições para estas linguagens: **DDL** (Data Definition Language - Linguagem de Definição de Dados), **DML** (DataManipulation Language - Linguagem de Manipulação de Dados) e **DCL**(Data Control Language – Linguagem de Controle de Dados).

Linguagem de Definição de Dados (DDL)

É a linguagem que permite a defin̨ção e manipulação de toda a estrutura de um banco de dados (campos, tipos, arquivos, etc.). Essas definições de dados devem ser armazenadas em algum lugar no banco de dados. Dessa forma, são mantidas no DD (Data Dictionary - Dicionário de Dados).

- **Exemplos: CREATE** (criar), **DROP** (deletar), **ALTER** (alterar)

Linguagem de Manipulação de Dados (DML)

É a linguagem que permite aos usuários do banco de dados manipularem os dados. Com esta linguagem é possível inserir, alterar e excluir os dados nas estruturas criadas.

- **Exemplos: SELECT** (seleção de dados), **INSERT** (inserção de dados), **UPDATE** (alteração de dados), **DELETE** (exclusão de dados).

Linguagem de Controle de Dados (DCL)

É a linguagem/comandos que permitem ao administrador de banco de dados controlar o acesso aos dados deste banco.

- **Exemplos:**
  GRANT: Permite dar permissões a um ou mais usuários e determinar as regras para tarefas determinadas.
  REVOKE: Revoga permissões dadas por um GRANT.

Dicionário de Dados (DD)

Arquivo que contém metadados; isto é, dados acerca de dados. Este arquivo é consultado antes de dados reais serem lidos ou modificados no sistema de banco de dados.

Entidade: Cliente				
Atributo	Classe	Domínio	Tamanho	Descrição
Codigo_cliente	Determinante	Numérico		
Nome	Simples	Texto	50	
Telefone	Multivalorado	Texto	50	Valores sem as máscaras de entrada
Cidade	Simples	Texto	50	
data_nascimento	Simples	Data		Formato dd/mm/aaaa

Arquitetura

Forma como os dados estão armazenados em um banco de dados. Um banco de dados deve apresentar os dados de forma que o usuário possa interpretá-los e modificá-los.

Podemos destacar três níveis principais segundo a visão e a função que o usuário irá realizar sobre o banco de dados:

- Nível Interno
- Nível conceitual
- Nível externo

Nível Interno

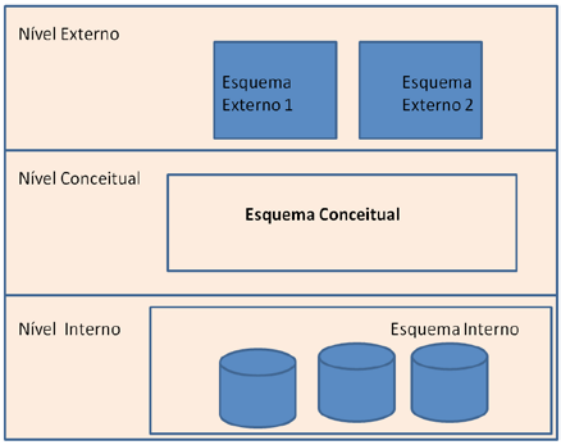
É o nível mais perto do armazenamento físico dos dados. Permite escrevê-los tal e como estão armazenados no computador. Neste nível se desenham os arquivos que contêm a informação, a localização dos mesmos e sua organização, ou seja, criam-se os arquivos de configuração.

Nível conceitual

Concentra-se no mais alto nível de abstração e não considera o banco de dados em si, mas a forma como as estruturas serão criadas para armazenar os dados.

Nível externo

É o mais próximo ao usuário. Neste nível se descrevem os dados ou parte dos dados que mais interessam aos usuários.



Tipos de arquiteturas

Plataformas centralizadas

Na arquitetura centralizada, existe um computador com grande capacidade de processamento, o qual é o hospedeiro do SGBD e emuladores para os vários aplicativos. Esta arquitetura tem como principal vantagem a de permitir que muitos usuários manipulem grande volume de dados. Sua principal desvantagem está no seu alto custo, pois exige ambiente especial para mainframes e soluções centralizadas.

Sistemas de Computador Pessoal - PC

Os computadores pessoais trabalham em sistema stand-alone, ou seja, fazem seus processamentos sozinhos. No começo esse processamento era bastante limitado, porém, com a evolução do hardware, os PCs têm grande capacidade de processamento. Eles utilizam o padrão Xbase e quando se trata de SGBDs, funcionam como hospedeiros e terminais. Dessa maneira, possuem um único aplicativo a ser executado na máquina. A principal vantagem dessa arquitetura é a simplicidade.

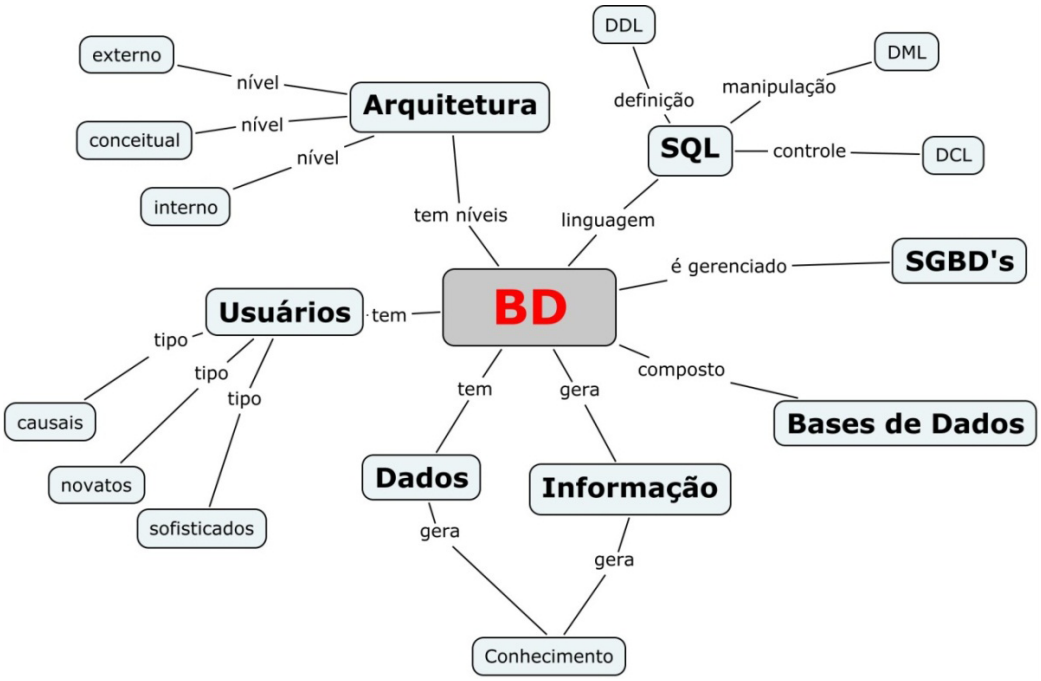
Banco de Dados Cliente-Servidor

Na arquitetura Cliente-Servidor, o cliente executa as tarefas do aplicativo, ou seja, fornece a interface do usuário (tela, e processamento de entrada e saída). O servidor executa as consultas no DB e retorna os resultados ao cliente. A principal vantagem dessa arquitetura é a divisão do processamento entre dois sistemas, o que reduz o tráfego de dados na rede.

Banco de Dados Distribuídos (N camadas)

Nesta arquitetura, a informação está distribuída em diversos servidores. Caso a informação solicitada seja mantida por outro servidor ou servidores, o sistema encarrega-se de obter a informação necessária, de maneira transparente para o aplicativo, que passa a atuar consultando a rede, independente de conhecer seus servidores.

Síntese



Mapa conceitual da disciplina de Banco de Dados Relacional

B

Modelo de dados

Unidade B
Projeto de Banco de Dados Relacional

UNIDADE B

MODELO DE DADOS

Modelo de dados é um tipo de abstração de dados usado para prover a representação conceitual. O modelo de dados utiliza os conceitos lógicos, como objetos, suas propriedades e seus interrelacionamentos, que podem ser mais fáceis para os usuários entenderem os conceitos de armazenamento computacionais. Consequentemente, o modelo de dados esconde os detalhes de armazenamento e da implementação, desinteressantes para a maioria dos usuários de banco de dados (ELMASRI, 2000).

Classificação

Modelo Conceitual

- Representação dos conceitos e características observados no ambiente;
- Ignorar particularidades de implementação.

Modelo Lógico

Regras de Derivação

- Normalização das estruturas de dados
- Derivação de estruturas de agregação e generalização-especialização
- Derivação de relacionamentos

Regras de Restrição:

- Restrição de domínio
- Restrição de Integridade
- Restrição de Implementação

Modelo Físico

- Inclui a análise das características e recursos necessários para armazenamento e manipulação das estruturas de dados (estrutura de armazenamento, endereçamento, acesso e alocação física).

Características

Conjunto de conceitos que podem ser usados para descrever a estrutura de um banco de dados.

- Tipos de dados, relacionamentos e restrições.
- Operações dinâmicas.

Categorias

- **Alto nível (modelos de dados conceituais):** ou modelo de dados conceitual, que fornece uma visão mais próxima do modo como os usuários visualizam os dados realmente.
- **Baixo nível (modelos de dados físicos):** ou modelo de dados físico, que fornece uma visão mais detalhada do modo como os dados estão realmente armazenados no computador.

Modelo Conceitual

Os modelos de dados conceituais utilizam conceitos como:

- Entidades;
- Atributos;
- Relacionamentos.

Uma **entidade** representa um objeto do mundo real ou um conceito, como um funcionário ou um projeto, que são descritos no banco de dados.

Entidade pode ser entendida como uma "coisa" ou algo da realidade modelada onde se deseja manter informações no banco de dados (BD). A entidade é representada por um retângulo, que contém o nome da entidade.

Representação de Entidade

A entidade **ALUNO** representa todos os estudantes sobre as quais se deseja manter informações no BD. Quando é necessário especificar um objeto particular (para o exemplo, determinado estudante) usa-se o termo **ocorrência de entidade**.

Atributos são propriedades (características) que identificam as entidades. Uma entidade é representada por um conjunto de atributos. Os atributos podem ser **simples, composto, multivalorado ou determinante**.

Um **atributo** corresponde a alguma propriedade de interesse que ajuda a descrever uma entidade, como o nome do funcionário ou seu salário.

Representação de Atributo na entidade Funcionário

Relacionamento é um conjunto de associações entre entidades. O relacionamento é representado por um losango. Esse losango é ligado por linhas aos retângulos que representam as entidades participantes do relacionamento

O **relacionamento** entre duas ou mais entidades mostra uma associação entre estas, por exemplo, um relacionamento *trabalha-em* de um funcionário com um projeto.

Representação do Relacionamento Trabalha entre as entidades Funcionário e Projeto.

Modelo de Dados

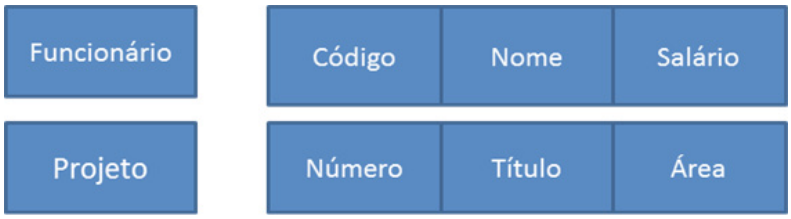
Conceitos

- **Esquema:** descrição do banco de dados
- **Instância:** uma determinada ocorrência
- **Estado do Banco de Dados:** conjunto de instâncias

Esquema

O esquema de um sistema de banco de dados é uma estrutura descrita em uma linguagem formal suportada pelo Sistema de Gerenciamento de Banco de Dados(SGBD). Em um banco de dados relacional, o esquema define as tabelas, os campos em cada tabela e os relacionamentos entre os campos e tabelas.

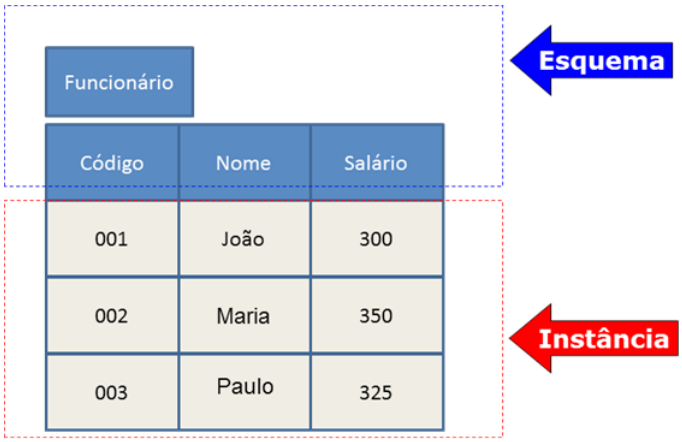
Esquemas geralmente são armazenados em um dicionário de dados. Apesar de um esquema ser definido em linguagem texto de banco de dados, o termo é frequentemente utilizado para referir-se a uma representação da estrutura do banco de dados.



Representação da estrutura das entidades Funcionário e Projeto.

Instância

- Informação propriamente dita.
- Coleção de informações do BD em um determinado instante = instância do BD.
- Alterada com inserções e remoções de informação.
- Corresponde ao conceito de valor armazenado na variável nas linguagens de programação.



Independência de Dados

A "independência de dados" pode ser definida como a capacidade de se alterar um esquema em um nível em um banco de dados sem ter que alterar um nível superior.

Existem dois tipos de independência de dados:

- **Independência de dados lógica:** é a capacidade de alterar o esquema conceitual sem ter que alterar o esquema externo ou as aplicações do usuário.
- **Independência de dados física:** é a capacidade de alterar o esquema interno sem ter que alterar o esquema conceitual, o esquema externo ou as aplicações do usuário.

Histórico dos Bancos de Dados

No processamento de dados tradicional, as aplicações (**sistemas**) eram basicamente dirigidas a um setor específico da empresa e, consequentemente, sua construção baseava-se nos dados utilizados em um determinado setor ou departamento.

Assim, o sistema de informação da empresa, era constituído por diversos sistemas, possuindo processos e arquivos próprios e independentes que não poderiam ser acessados de forma integrada.

Problemas :

- **Integração:** A comunicação de dados entre tais empresas, quando necessária era difícil.
- **Redundância não controlada:** Ocorrência de duplicação de dados e tarefas, gerando inconsistência.
- **Custo:** Aumento do custo de manutenção, devido ao fato de uma mesma tarefa estar sendo realizada em dois ou mais processos distintos.

Identificados esses problemas, a evolução natural impôs aos sistemas uma eliminação da duplicação e integração entre os sistemas (**dados**). Surge então um novo problema.

Devido à integração, dois ou mais processos passam a compartilhar o mesmo dado. É necessário **controlar a concorrência** no acesso aos dados.

Outra dificuldade encontrada foi a diversidade de programas, onde os dados poderiam ter diferentes estruturas físicas com tipos e tamanhos diferentes. Seria necessário tratar **independência entre dados e processos.**

Para solucionar os problemas e facilitar o acesso aos dados, surge então o Banco de Dados

Objetivos de um BD

Permitir a independência entre dados e programas

- **Independência lógica:** permite que a visão global dos dados se modifique sem que as aplicações existentes tenham que ser alteradas.
- **Independência física:** permite que a representação física das estruturas de dados se modifique de acordo com os requisitos de novas aplicações sem que as existentes tenham que ser alteradas.

Permitir o controle de redundância de dados

Controle centralizado dos dados das diversas aplicações que utilizam o banco de dados, podendo estabelecer procedimentos de controle e verificação e também podendo criar padrões.

Garantir a integridade dos dados

Duas ou mais aplicações podem vir a compartilhar um mesmo dado concorrentemente, portanto, deve existir um mecanismo que garanta a integridade dos dados. A integridade também deve ser mantida através de logs de atualização, possibilitando desfazer alterações corretas feitas por outras aplicações.

Garantir a privacidade

Garantir que o acesso aos dados possa ser controlada pelo administrador do banco de dados, garantindo a segurança dos dados contra acessos e modificações indevidas.

Permitir a facilidade de criação de novas aplicações

sendo o banco de dados criado a partir de um modelo conceitual da empresa, ele deve constituir a base de dados necessária a todas as aplicações da empresa, dessa forma, podemos dizer que facilita a criação de novas aplicações.

Segurança de dados

Por ser constituída de informações da empresa, a segurança dessas contra perdas ou destruição deve ser um ponto importante. O banco de dados deve permitir cópias dos dados que possam ser restaurados parcial ou totalmente.

Controle automático de relacionamento entre registros

Efetuar o controle e manutenção do relacionamento entre registros. Em um banco de dados este controle deve ser automático, a partir da definição do esquema global da empresa.

Otimização da utilização de espaço de armazenamento

Devido ao grande volume de dados envolvidos em um banco de dados que, por sua vez, utiliza espaço de armazenamento, que normalmente é limitado, podem ser utilizadas técnicas de compressão de dados e reaproveitamento automático dos espaços gerados por eliminações.

Aplicações Típicas

- Sistemas de Informação
- Aplicação web de comércio eletrônico
- ERP
- CRM
- Correio Eletrônico
- Softwares em geral

Síntese

Banco de dados é um conjunto de registros (dados) dispostos em estrutura de tabelas que possibilita a organização dos mesmos e produção de informação. Um banco de dados normalmente agrupa registros de um mesmo fim.

Um banco de dados é normalmente mantido e acessado por meio de um software conhecido como Sistema Gerenciador de Banco de Dados (SGBD). Um SGBD adota um modelo de dados. O termo SGBD não é sinônimo de Banco de Dados, é apenas um sistema que auxilia na Administração das Bases de Dados.

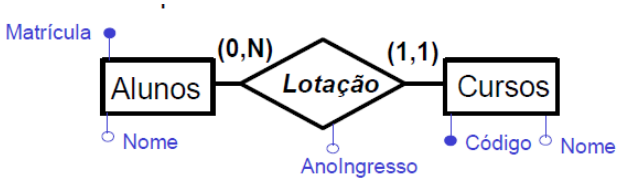
Modelo de Dados é uma descrição dos tipos de informações que estão armazenadas em um banco de dados. Em uma indústria, o modelo de dados poderia informar que o banco de dados armazena informações sobre produtos e que, para cada produto, são armazenados seu código, preço e descrição. Observe que o modelo de dados não informa quais os produtos que estão armazenados no banco de dados, mas apenas que o banco de dados contém informações sobre produtos.

Linguagem de Modelagem é utilizada para construir um modelo de dados, podem apresentar-se de forma textual ou em linguagem gráfica. Um mesmo modelo de dados pode ser representado de várias formas, essas representações são conhecidas como “esquema de banco de dados”. Um modelo de dados

deve respeitar o nível de conhecimento dos usuários que deve atender, para um usuário final. O modelo não deve conter detalhes técnicos de implementação que serão necessários para um usuário mais avançado do sistema.

Modelo Conceitual registra que dados podem aparecer no banco de dados, mas não registra como esses dados estão armazenados em nível de SGBD. A técnica mais difundida de modelagem conceitual é a *abordagem entidade-relacionamento* (ER). Nessa técnica, um modelo conceitual é usualmente representado através de um diagrama, chamado *diagrama entidade-relacionamento-DER*.

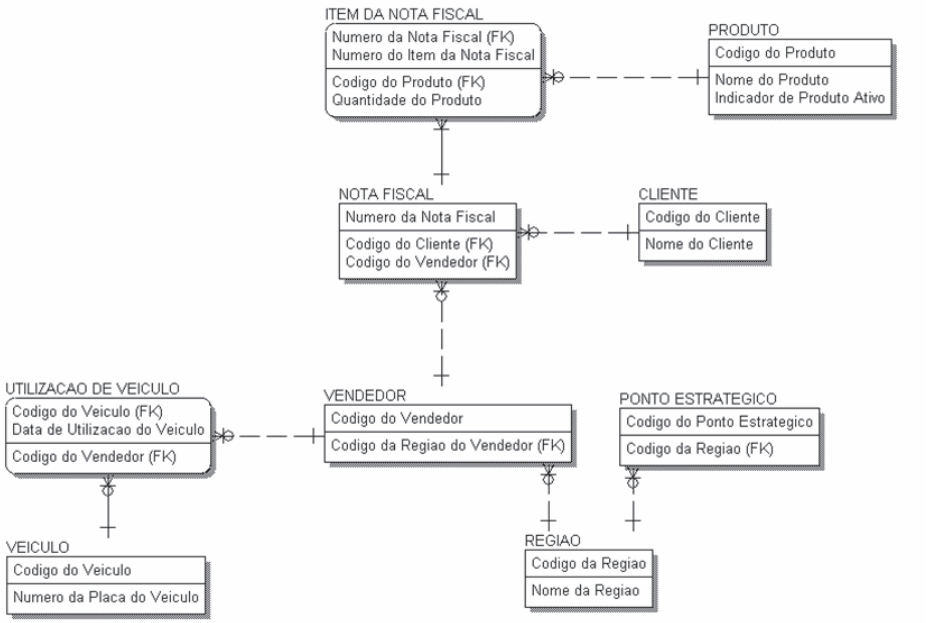
Exemplo de um DER:



Modelo lógico é uma representação lógica das informações da área de negócios, não é um banco de dados, é independente do modelo físico. Este é o conceito chave da modelagem de dados lógica. Ele deve ser independente da tecnologia implementada. Os componentes desse modelo devem estar intimamente ligados aos negócios, não a tecnologia.

Um modelo lógico de dados para uso meramente operacional/transacional deve:

- Ser completamente normalizado.
- Representar fielmente o NEGÓCIO e NÃO necessariamente a base de dados desejada, a qual será construída posteriormente por ocasião do Projeto Físico.
- Conter descrição sucinta das entidades, atributos e relacionamentos.
- Conter os nomes de entidades e atributos, extensos e abreviados, atribuídos de acordo com algum padrão adotado na organização e formados por termos previamente convencionados em um glossário.
- Contemplar, para cada um dos atributos, o tipo de dado, tamanho e opcionalidade.



Exemplo de Modelo Lógico

Modelo físico: no modelo físico fazemos a modelagem física do modelo de banco de dados. Levam-se em conta as limitações impostas pelo SGBD escolhido e deve ser criado sempre com base nos exemplos de modelagem de dados produzidos no item anterior, modelo lógico.

Atividades

Com base no material anterior responda às questões abaixo.

1. Qual das alternativas abaixo não pode ser considerada como um banco de dados?

a. Lista telefônica
b. Coleção de CD's
c. Partida de Futebol
d. Relação de aluno

2. Qual alternativa não é uma linguagem do SQL?

a. DML
b. DDL
c. UML
d. DCL

3. A classificação do Modelo de dados é:

a. Conceitual, Lógico e Dinâmico.
b. Conceitual, Lógico e Físico.
c. Conceitual, Interno e Externo.
d. Conceitual, Físico e Abstrato.

4. Uma das alternativas apresenta os níveis da Arquitetura de um BD.

a. Interno, Central, Externo.
b. Externo, Conceitual, Interno.
c. Interno, Médio, Externo.
d. Externo, Físico, Interno.

5. São objetivos de um BD:

a. Acesso, distribuição, replicação.
b. Copiar, Incluir, Deletar.
c. Controle de redundância, integridade, privacidade, segurança.
d. Entrada de dados, manipulação de dados, pesquisa.

6. São conceitos utilizados de um modelo de dados conceitual:

a. Entidades, atributos, relacionamentos.
b. Esquema, Instância, Modelo.
c. Arquitetura, Dados, Esquema.
d. Relacionamento, Entidade, Esquema.

7. Um esquema de BD é armazenado onde?

a. no Dicionário de Dados
b. no Repositório de Dados
c. na Base de Dados
d. na Coleção de Dados

8. O que você entende por "independência de Dados"?

a. Pode ser definida como a capacidade de se alterar um esquema em um nível em um banco de dados alterando o nível superior.
b. Pode ser definida como a capacidade de não se alterar um esquema em um nível em um banco de dados sem ter que alterar um nível superior.
c. Pode ser definida como a capacidade de se alterar um dado em um nível em um banco de dados sem ter que alterar o dado em um nível superior.
d. Pode ser definida como a capacidade de se alterar um esquema em um nível em um banco de dados sem ter que alterar um nível superior.

9. São aplicações de Banco de Dados:

a. softwares, correio eletrônico e sistemas de informação.
b. softwares, hardware e periféricos.
c. dicionário de Dados, tabelas e registros.
d. esquema, entidades e instância.

10. Em Banco de Dados um sinônimo para Entidade é?

a. Instituição
b. Empresa
c. Tabela
d. Tupla

C

Modelo conceitual

Unidade C
Projeto de Banco de Dados Relacional

UNIDADE C

MODELO CONCEITUAL

Definições

Entidade

abstração de um fato do mundo real para o qual se deseja manter seus dados no BD

- Notação: retângulo nomeado

Representa um conjunto de ocorrências do fato

Representação gráfica | Interpretação

Heuser, 2001

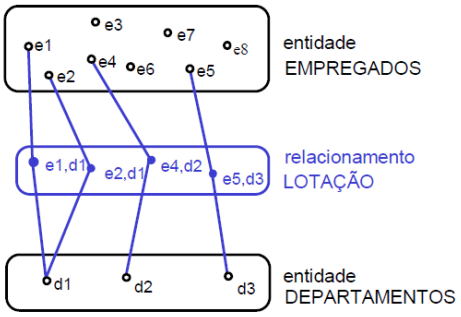
Relacionamento

Abstração de uma associação entre (ocorrências de) entidades.

- Notação: losango nomeado
- Representa um conjunto de ocorrências de relacionamentos

Representação gráfica

Heuser, 2001

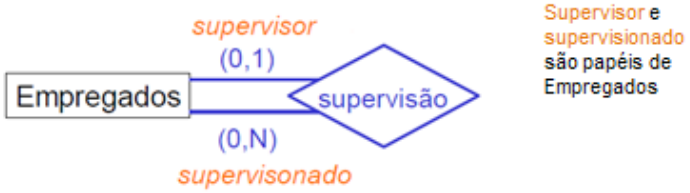


Representação gráfica dos relacionamentos (Heuser, 2001)

Autorrelacionamento

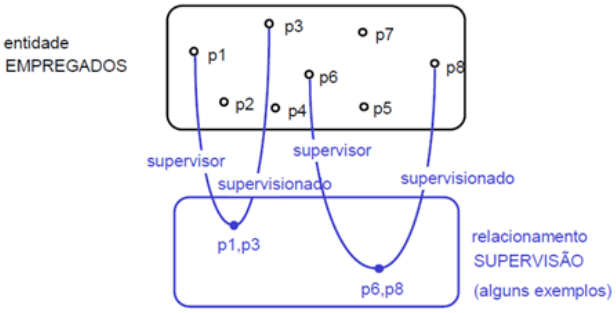
Representa uma associação entre ocorrências de uma mesma entidade.

- Exige a identificação de papéis.



Leitura: *Um empregado pode ser* ***supervisionado por no máximo*** *1 empregado. Um empregado pode* ***supervisionar no máximo*** *N empregados.*

Heuser, 2001

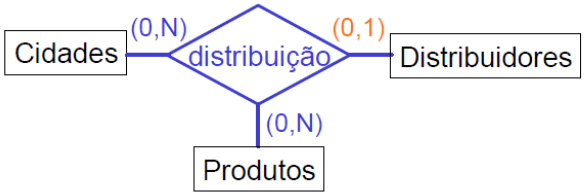


Exemplo de Autorrelacionamento (Heuser, 2001)

Relacionamento "N"-ário

Abstração de uma associação entre "N" (ocorrências de) entidades.

- Relacionamento ternário.

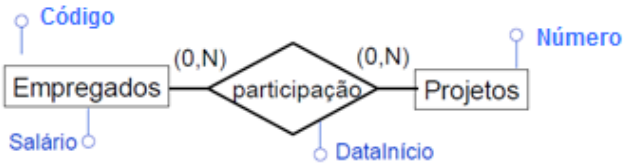


Heuser, 2001

Leitura: Um produto em uma cidade pode ser entregue por no máximo 1 distribuidor.

Atributo

Dado ou informação que é associado a cada ocorrência de uma entidade ou de um relacionamento.



Heuser, 2001

Exemplos de atributos: código, salário, data inicio, número.

Classificação de Atributo

1. obrigatórios OU opcionais
2. monovalorados OU multivalorados
3. simples OU compostos

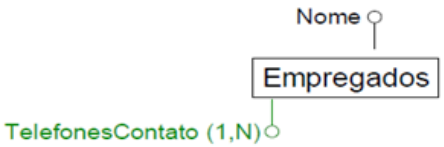
Cardinalidade mínima:

- Atributo obrigatório (cardinalidade mínima "1") cada entidade possui no mínimo um valor associado.
- Atributo opcional (cardinalidade mínima "0")

Heuser, 2001

Cardinalidade máxima:

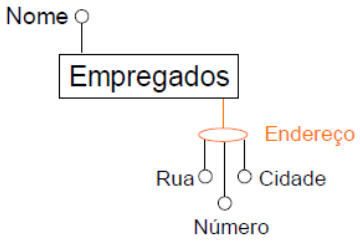
- Atributo monovalorado (cardinalidade máxima "1") cada entidade possui no máximo um valor associado.
- Atributo multivalorado (cardinalidade máxima "n")



Heuser, 2001

Atributo Simples e Composto

- Atributo simples - recebe um valor único associado a ele.
- Atributo composto - seu conteúdo é formado por vários itens menores como no exemplo Endereço.



Heuser, 2001

Normalização

Normalização é uma forma que os bancos de dados relacionais utilizam para evitar redundâncias e possibilitar um maior desempenho nas pesquisas.

É um processo de organização eficiente dos dados dentro de um banco de dados.

Objetivos

- Eliminar dados redundantes (por exemplo, armazenando os mesmos dados em mais de uma tabela).

- Garantir que as dependências entre os dados façam sentido (armazenando apenas dados logicamente relacionados em uma tabela).

Existem cinco estágios de normalização, 1º, o 2º, o 3º, o 4º e o 5º. Para que um banco de dados se encontre em cada um desses estágios ou formas (denominadas formas normais), cada uma de suas tabelas deve atender a alguns pré-requisitos. Os pré-requisitos são cumulativos, isto é, para alcançar a 3ª forma normal (3NF), um banco de dados precisa atender aos pré-requisitos das 1ª e 2ª formas normais, acrescidos dos requisitos exclusivos da 3NF.

Primeira Forma Normal (FN1)

- Parte da definição formal do Modelo Relacional.
- Todos os atributos da relação devem ser atômicos e monovalorados.
- Cada ocorrência da chave primária deve corresponder a uma e somente uma informação de cada atributo, ou seja, a entidade não deve conter grupos repetitivos (multivalorados).
- Gerar uma nova relação contendo o grupo de repetição e a chave primária da relação original.
- Decompor em tantas entidades quanto for o número de conjuntos de atributos repetitivos.

- **Exemplo:** Tabela Cliente não está na FN1

Cliente

ClienteID	Nome	Telefone
234	Maria Olive	9981-8765
567	Murilo Perez	8123-9087 6125-9087
765	Rosa Maria Vaz	9276-0832

Existem dois números de telefone para um mesmo cliente.

- Transição da tabela Cliente para a FN1

Cliente

ClienteID	Nome	Telefone
234	Maria Olive	9981-8765
567	Murilo Perez	8123-9087 6125-9087
765	Rosa Maria Vaz	9276-0832

Cliente

ClienteID	Nome
234	Maria Olive
567	Murilo Perez
765	Rosa Maria Vaz

Telefone

ClienteID	Telefone
234	9981-8765
567	6125-9087
567	8123-9087
765	9276-0832

Devemos criar a tabela Telefone para armazenar os valores duplicados.

Segunda Forma Normal (FN2)

Uma relação está na FN2 quando duas condições são satisfeitas:

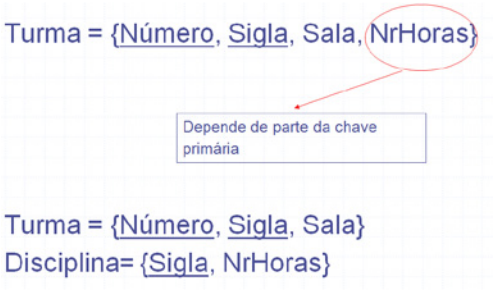
1. A relação está na 1FN.
2. Todo atributo da tabela seja dependente funcional da chave completa e não de parte da chave. Ou seja, todos os atributos não-chave dependem funcionalmente de toda a chave primária.

Dependência funcional Total

Na ocorrência de uma chave primária concatenada, dizemos que um atributo ou um conjunto de atributos depende de forma completa ou total desta chave primária concatenada, se e somente se, a cada valor da chave (e não parte dela) está associado um valor para cada atributo.

Se a remoção de qualquer atributo A de X implicar que há dependência, não mais será assegurada.

A resolução da aplicação da segunda forma normal é realizada através da exclusão dos atributos que não dependem totalmente da chave primária, da tabela original, e constituindo-se com estes uma nova tabela, que terá como chave primária o atributo participante da chave primária da tabela origem.



Segunda Forma Normal (FN2) (HEUSER, 2001)

Criação de uma segunda tabela “Disciplina” para armazenar os dados de cada disciplina da turma.

Terceira Forma Normal (FN3)

- Deve estar na 2FN
- Não existem atributos não-chave que sejam dependentes de outros atributos não-chave.

Dependência transitiva

Quando um atributo ou conjunto de atributos A depende de outro atributo B que não pertence à chave primária, mas é dependente funcional desta, dizemos que A é dependente transitivo de B.

- Tabela Torneio de FUTSAL está na FN2 mas não está na FN3

Torneio de FUTSAL

TORNEIO	ANO	VENCEDOR	FUNDAÇÃO
Parana	2002	Azul FC	1989
Rio de Janeiro	2004	Falção EC	2000
Rio Grande Sul	2004	Azul FC	1989
São Paulo	2008	ABC FS	2001

Existem valores repetidos na data de fundação, o campo data de fundação está diretamente ligada ao vencedor.

- Transição da tabela Torneio de FUTSAL da FN2 para a FN3

Vencedores de FUTSAL

TORNEIO	ANO	VENCEDOR
Parana	2002	Azul FC
Rio de Janeiro	2004	Falção EC
Rio Grande Sul	2004	Azul FC
São Paulo	2008	ABC FS

Torneio de FUTSAL

TORNEIO	ANO	VENCEDOR	FUNDAÇÃO
Parana	2002	Azul FC	1989
Rio de Janeiro	2004	Falção EC	2000
Rio Grande Sul	2004	Azul FC	1989
São Paulo	2008	ABC FS	2001

Clube - Fundação

CLUBE	FUNDAÇÃO
ABC FS	2001
Azul FC	1989
Falção EC	2000

Devemos criar a tabela Clube-Fundação para evitar os valores duplicados.

Abordagem Entidade Relacionamento

- Modelo formal
- Preciso
- Não ambíguo

Importante!

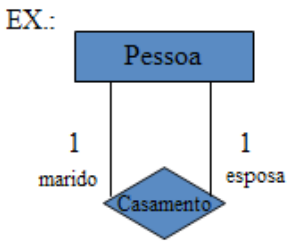
- Diferentes leitores de um mesmo ER devem sempre entendê-lo exatamente.
- Pode ser usado como entrada para uma ferramenta CASE, na geração de um Banco de Dados Relacional.
- Todos os que manipulam modelos ER devem estar capacitados para a sua compreensão.

A abordagem Entidade Relacionamento tem poder de expressão limitada.

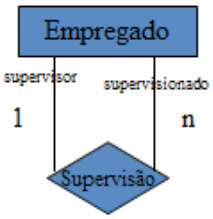
Em um modelo ER são mostradas apenas algumas propriedades de um banco de dados, podendo ser necessário utilizar outras linguagens para anotar propriedades importantes.

- **Exemplo:** Linguagem natural.

Representação do Modelo Entidade Relacionamento



A cardinalidade do modelo garante a integridade



Relacionamento recursivo não garante a integridade

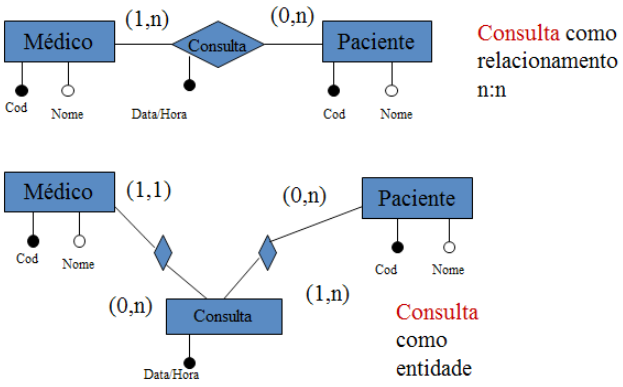
Heuser, 2001

- **Restrição de Integridade:** Regra estabelecida pela realidade e que deve ser obedecida pelo BD.

Diferentes modelos devem ser equivalentes

Equivalentes:

- Quando expressam a mesma realidade.
- Quando geram o mesmo modelo de BD relacional.
- BD que abstraindo as diferenças de nomes de construções (tabelas, atritutos, ...) tenham a mesma estrutura



Exemplo da Abordagem Entidade Relacionamento (HEUSER, 2001)

Determinando Construções

O objeto a ser modelado não pode ser analisado isoladamente.

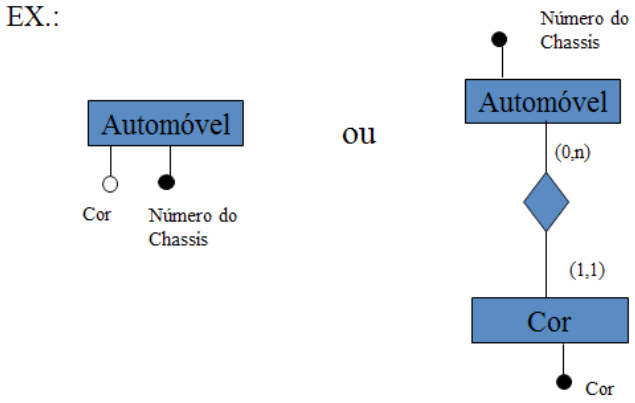
É necessário conhecer o contexto, o modelo dentro do qual o objeto está inserido.

Critérios que podem ser usados na escolha da construção de modelos

Atributo versus entidade relacionada

Caso o objeto cuja modelagem está em discussão esteja vinculado a outros objetos, caso o objeto tenha propriedades (atributos, relacionamentos, entidades genéricas ou especializadas), deve ser modelado como entidade.

Quando existem transações no sistema que alterem o conjunto de valores do objeto:

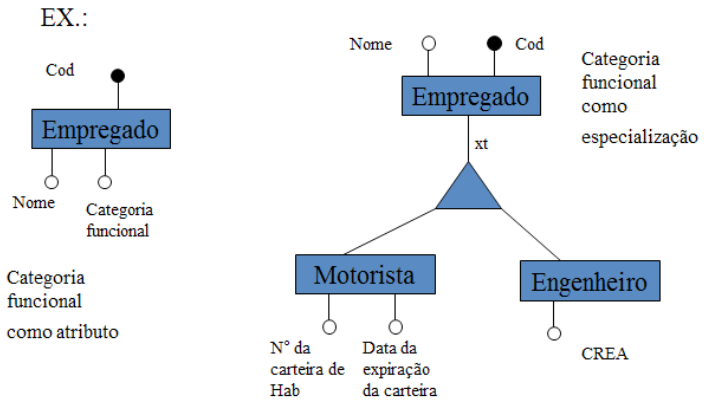


Heuser, 2001

O atributo Cor se tranforma em uma entidade relacionada à entidade Automóvel.

Atributo versus especialização

Uma especialização deve ser usada quando se sabe que as classes especializadas de entidades possuem propriedades (atributos, relacionamentos, generalizações, especializações) particulares.



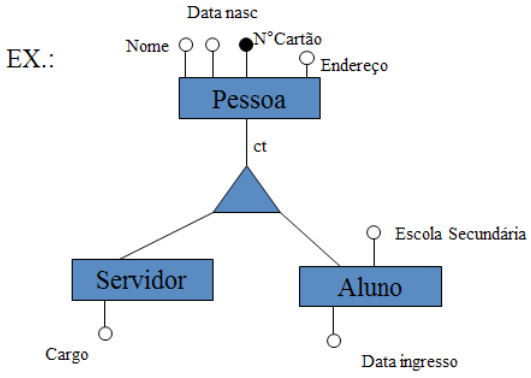
Heuser, 2001

O atributo Categoria Funcional se tranforma em uma especialização da entidade Empregado.

Entidade relacionada versus especialização

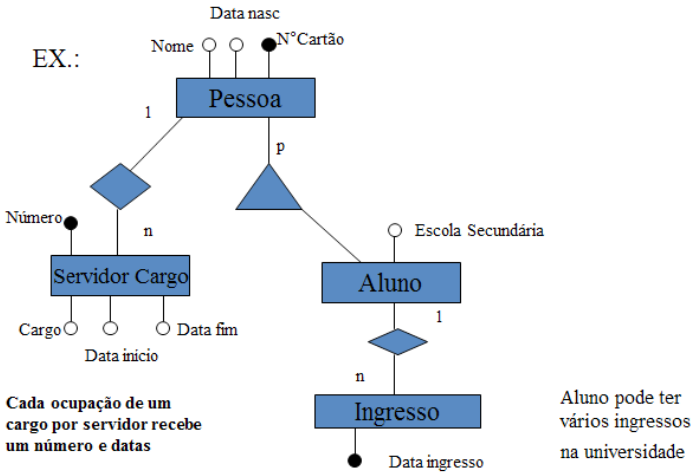
Para que uma entidade possa ser considerada especialização de outra é necessário que ela herde o identificador (chave primária) da entidade genérica.

Para cada ocorrência do objeto genérico pode existir no máximo 1 ocorrência na especialização.



Heuser, 2001

Entidade relacionada versus especialização



Heuser, 2001

Atributos opcionais e multivalorados

Atributo opcional

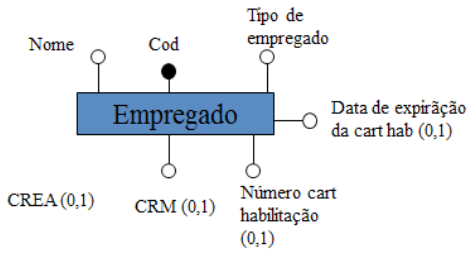
Indicam subconjuntos de entidades que são modelados mais corretamente através de especializações.

Toda vez que aparecer um atributo opcional, é recomendável verificar se a modelagem através de entidades especializadas não é mais conveniente.

Atenção

No processo de modelagem é aconselhável tentar restringir-se ao uso de atributos obrigatórios e monovalorados.

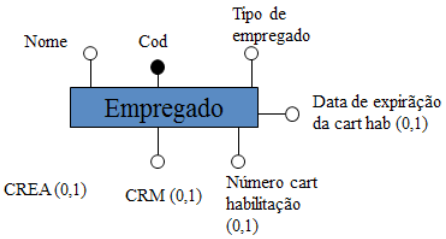
EX.:



Heuser, 2001

No exemplo, o uso de atributos opcionais esconde as diferentes categorias de empregados e suas entidades.

EX.:



Heuser, 2001

Neste modelo, fica mais claro quais são os atributos de cada um dos subconjuntos particulares de empregado.

Atributos opcionais e multivalorados

Atributo multivalorado

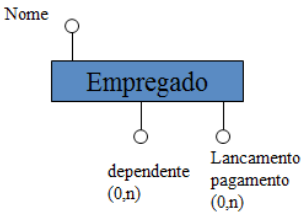
Não existe, em um SGBD relacional, construção de arrays.

Atributos multivalorados podem induzir a um erro de modelagem, que é o de ocultar entidades e relacionamentos em atributos multivalorados.

Atenção

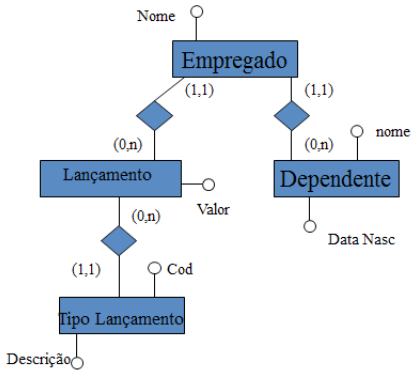
No processo de modelagem é aconselhável tentar restringir-se ao uso de atributos obrigatórios e monovalorados.

EX.:



Usando atributos multivalorados (HEUSER, 2001)

EX.:



Substituindo atributos multivalorados por entidades relacionadas. (HEUSER, 2001)

Verificação do modelo Entidade Relacionamento

Uma vez construído o modelo ER deve ser validado e verificado. A verificação é o controle de qualidade que procura garantir que o modelo usado para a construção do banco de dados gerará um bom produto.

Requisitos

- Modelo deve ser correto.
- Modelo deve ser completo.
- Modelo deve ser livre de redundância.
- Modelo deve refletir o aspecto temporal.
- Entidade isolada e entidade sem atributos.

Modelo deve ser correto.

O modelo está correto quando não contém erros de modelagem, quando os conceitos de modelagem são corretamente empregados.

Tipos de erros:

- **Sintáticos:** Ocorrem quando o modelo não respeita as regras de construção de um modelo ER.
  - Exemplos:
    - Associar atributos a atributos.
    - Associar relacionamentos a atributos.
    - Associar relacionamentos a outros relacionamentos.
    - Especializar relacionamentos ou atributos.
- **Semânticos:** Ocorrem quando o modelo, apesar de obedecer às regras de construção de modelo ER (correto sintaticamente), reflete a realidade de forma inconsistente.
  - Exemplos:
    - Estabelecer associações incorretas.
    - Usar uma entidade do modelo como atributo de outra entidade.
    - Usar número incorreto de entidades em um relacionamento.
  - Obs.: Regras de normalização de base verificam a correção de modelos ER.
- **Modelo deve ser completo:** O modelo deve fixar todas as propriedades desejáveis do BD. Isso somente pode ser verificado por alguém que conhece profundamente o sistema a ser implementado.
  - Exemplos:
    - Verificar se todos os dados que devem ser obtidos do BD estão presentes.
    - Verificar se todas as transações de modificação do BD podem ser executadas sobre o modelo.
- **Modelo deve ser livre de redundância:** O modelo deve ser mínimo, não deve conter conceitos redundantes.
  - Exemplos:
    - Relacionamentos redundantes: relacionamentos que são resultados da combinação de outros relacionamentos entre as mesmas entidades.
    - Atributos redundantes: são atributos deriváveis a partir da execução de procedimentos de busca de dados e/ou cálculos sobre o BD.
- **Modelo deve ser livre de redundância:** Construções redundantes devem ser omitidas do modelo ER.
  - Exemplos:
    - Redundância não controlada: é indesejável.

    - Redundância controlada: deve ser necessariamente evitada.
  - Obs.: Em alguns casos, redundância controlada pode servir para aumentar a performance de operações de busca, mas não devem aparecer no modelo conceitual do BD.

- **Modelo deve refletir aspecto temporal:** Certas aplicações exigem que o BD guarde o histórico de alterações de informação. O modelo de BD que armazena somente valores atuais de uma informação é diferente do modelo do BD que armazena o histórico da informação. Portanto, é necessário considerar o aspecto temporal na modelagem de dados.

- Obs.: Não existem regras gerais de como proceder neste caso.

  - **Relacionamentos que se modificam ao longo do tempo:** Assim como atributos podem ser modificados, relacionamentos também podem ser modificados ao longo do tempo. Quando é considerada a história de suas alterações, relacionamentos 1:1 ou 1:n são transformados em n:n.
  - Exemplo: Endereço de Cliente, Salário de um empregado.

  - Obs.: O relacionamento transforma-se em entidade e deve ter um atributo identificador, normalmente "data".

  - **Consultas a dados referentes ao passado:** Muitas vezes, para evitar o crescimento do BD, informações referentes ao passado são eliminadas. Mas tais informações podem se fazer necessárias no futuro, por motivos legais, auditorias ou tomada de decisão. Caso informações antigas fiquem no BD, podem ser necessários atributos para indicar o status da informação, se atual ou antiga.

  - **Consultas a dados referentes ao passado:**
    - **Planejar o arquivamento de informações antigas**
      - Para as informações que serão retiradas do BD e armazenadas em arquivos convencionais, é necessário fazer um planejamento de como essas informações serão acessadas no futuro.
      - Uma solução é reincluir as informações no BD, quando elas forem necessárias. Isso permite que informações passadas sejam acessadas da mesma forma que as atuais.

  - Obs.: É necessário observar que informações em um BD estão normalmente relacionadas a outras.

    - **Planejar informações estatísticas:** Em alguns casos, informações antigas são necessárias apenas para tomada de decisões. Nesse caso, muitas vezes desejam-se apenas dados resultantes de cálculos ou estatísticas sobre as informações, como totais, contagens, médias. Assim, pode ser conveniente manter no banco de dados essas informações compiladas e eliminar as informações usadas na compilação.

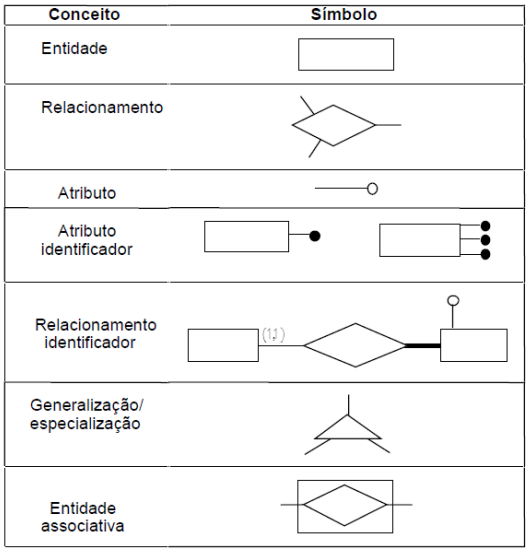
Entidade isolada e entidade sem atributos

Uma entidade isolada não apresenta nenhuma ligação com outra entidade. Em princípio entidades isoladas não estão incorretas. Uma entidade que modela a organização na qual o sistema implementado no BD está embutido.

Outra situação que não está incorreta, mas que deve ser investigada é a de uma entidade sem atributos.

Obs.: Não existem regras gerais de como proceder neste caso.

- Modelo Conceitual – Notação:



Símbolos usados na construção de esquemas ER's (HEUSER, 2001)

Síntese

Elementos necessários para a criação de um modelo conceitual

Modelagem de Dados

A modelagem de dados deve ser iniciada a partir do conhecimento do mundo real, é a etapa de análise e coleta de dados sobre o problema a ser resolvido. O modelo conceitual, que é o primeiro passo a ser desenvolvido, pois independe de tecnologia e serve como base para um projeto de desenvolvimento de software, vamos conhecer os elementos necessários para construção do modelo conceitual:

Entidade: Tipo de dado a ser tratado ex: (Pessoa, Cliente, Departamento, Empresa):

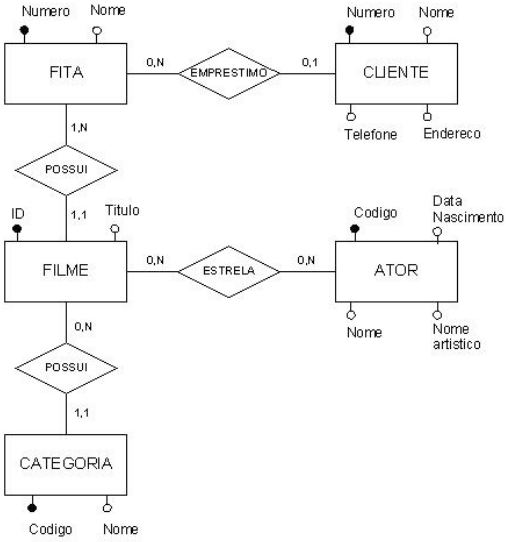
Relacionamento: Indica relação entre entidades ex: (trabalha, estuda, atua)

Cardinalidade mínima e máxima: Indica quantos tipos de dados podem aparecer em um relacionamento

Exemplo de relacionamento entre 2 entidades

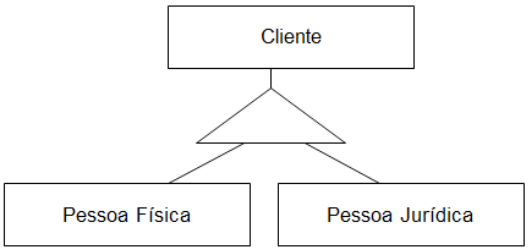
Atributo: corresponde a alguma propriedade de interesse que ajuda a descrever uma entidade, ex: (nome do funcionário, salário).

- Modelo conceitual de uma locadora de filmes



Generalização/Especialização

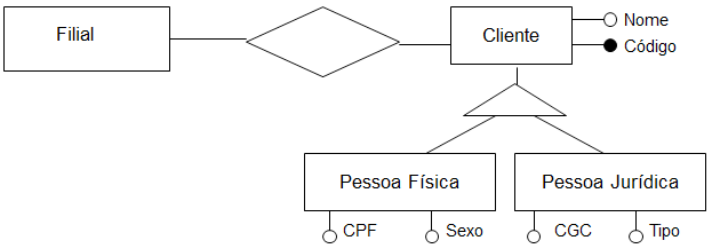
- Propriedades podem ser atribuídas a entidades através do conceito de generalização/especialização.
- Podem-se atribuir propriedades particulares a um subconjunto das ocorrências (especializadas) de uma entidade genérica.



Heuser, 2001

Herança de Propriedades

Cada ocorrência da entidade especializada possui, também, além de suas próprias propriedades (atributos, relacionamentos e generalizações/especializações), as propriedades da ocorrência da entidade genérica correspondente.



Heuser, 2001

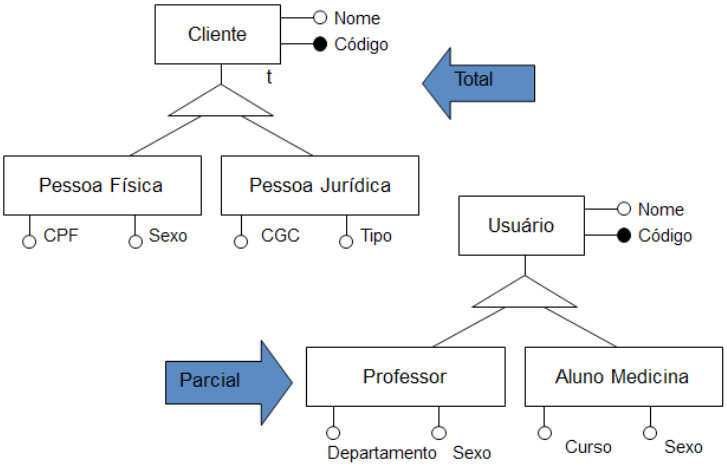
- A entidade Pessoa Física possui, seus atributos particulares, CIC e Sexo, como também todas as propriedades da

ocorrência CLIENTE correspondente, ou seja, os atributos Nome e Código.

Classificação

Total (t)

- Para cada ocorrência da entidade genérica existe sempre uma ocorrência em uma das entidades especializadas.
- A toda ocorrência da entidade Cliente corresponde uma ocorrência em uma das duas especializações.



Heuser, 2001

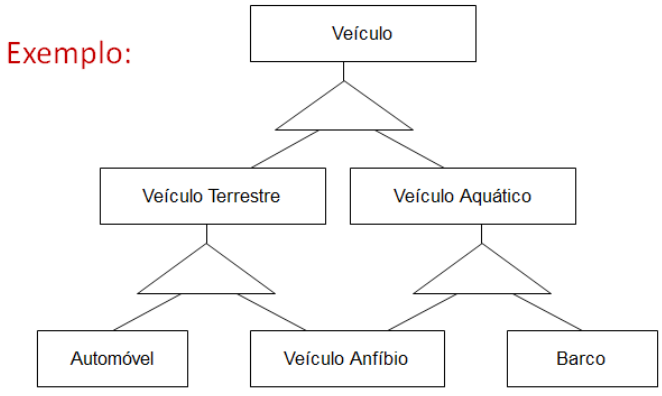
Propriedades

Uma entidade pode ser especializada em qualquer número de entidades, inclusive uma única.

- Se apenas os motoristas possuíssem propriedades particulares, haveria apenas uma entidade especializada (motoristas).

Não há limite no número de níveis hierárquicos da generalização/especialização.

- Uma entidade especializada em uma generalização/especialização pode ser entidade genérica em uma outra generalização/especialização.



Generalização/Especialização (HEUSER, 2001)

Tipos

Especialização Exclusiva

Significa que uma ocorrência de entidade genérica aparece, para cada hierarquia generalização/ especialização no máximo uma vez.

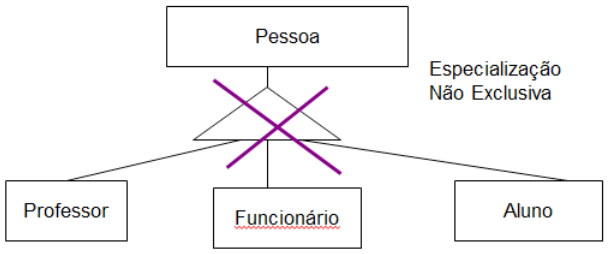
- Cada veículo pode ser um automóvel ou veículo anfíbio ou barco e somente um destes.

Especialização Generalização Não Exclusiva

Significa que uma ocorrência de entidade genérica aparece para cada hierarquia generalização/ especialização várias vezes.

- Não podem herdar o identificador da entidade genérica.

Exemplo:



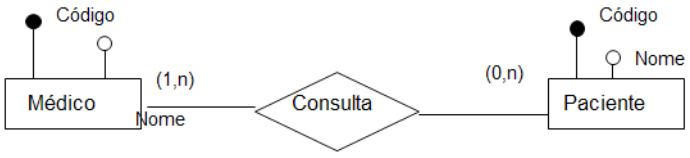
Heuser, 2001

Entidade Associativa

Um relacionamento é uma associação entre entidades.

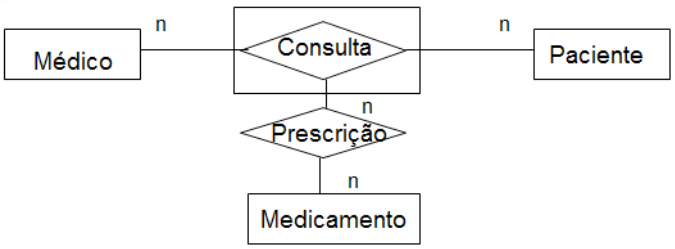
Em ER não foi prevista a possibilidade de associar uma entidade com um relacionamento ou então associar dois relacionamentos entre si.

- **Exemplo. Dada a situação abaixo:**
  - Registrar a informação de que em cada consulta um ou mais medicamentos podem ser prescritos ao paciente.



Heuser, 2001

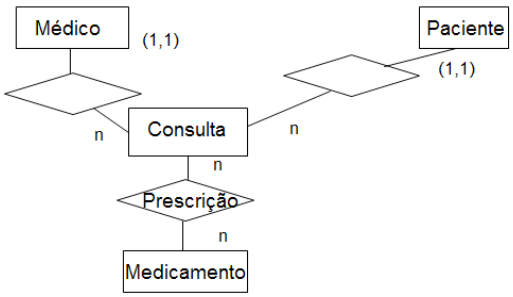
Exemplo:



Heuser, 2001

- A entidade Medicamento deve ser ligada à consulta, pois se for ligado ao médico, perde-se a informação sobre o paciente e vice-versa.
- Deseja-se relacionar o medicamento à consulta.
- Uma entidade associativa é a redefinição de um relacionamento, que passa a ser tratado como se fosse uma entidade.
- O retângulo desenhado ao redor do relacionamento **CONSULTA** indica que este relacionamento passa a ser visto como uma entidade associativa.
- Sendo **CONSULTA** uma entidade, é possível associá-la através de relacionamentos a outras entidades.

Alternativa:



Heuser, 2001

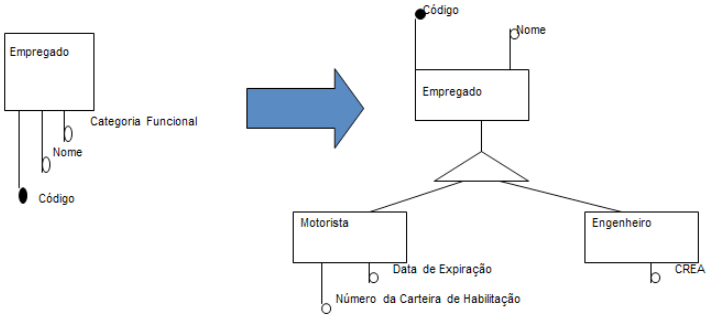
Atributo x generalização / especialização

Uma especialização deve ser usada quando se sabe que as classes especializadas de entidades possuem propriedades (atributos, relacionamentos, generalizações, especializações) particulares.

Faz sentido especializar a entidade empregado de acordo com a categoria funcional, no caso de classes particulares possuírem atributos ou relacionamento próprios.

Ainda no exemplo acima, o sexo do empregado é melhor modelar como atributo de empregado, caso não existam propriedades particulares de homens e mulheres a modelar na entidade considerada.

Exemplo:

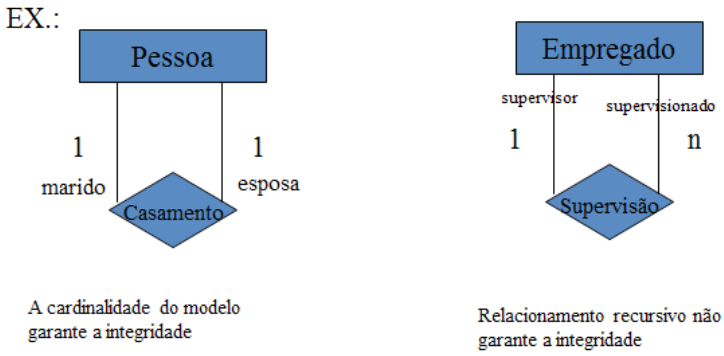


Heuser, 2001

Abordagem Entidade Relacionamento

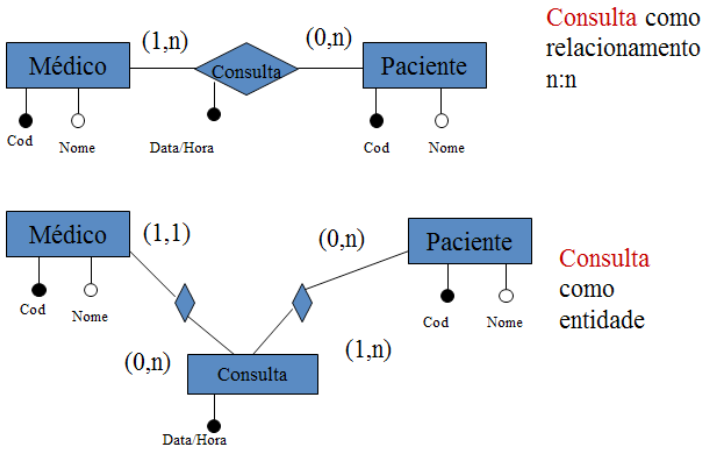
Poder de expressão limitada. Em um modelo ER são mostradas apenas algumas propriedades de um banco de dados, podendo ser necessário utilizar outras linguagens para anotar propriedades importantes.

- EX.: Linguagem natural



Heuser, 2001

- Restrição de Integridade: Regra estabelecida pela realidade e que deve ser obedecida pelo BD.
- Diferentes modelos devem ser equivalentes.
- Equivalentes:
  - Quando expressam a mesma realidade.
  - Quando geram o mesmo modelo de BD relacional.
  - BD que abstraindo as diferenças de nomes de construções (tabelas, atritutos, ...) tenham a mesma estrutura.



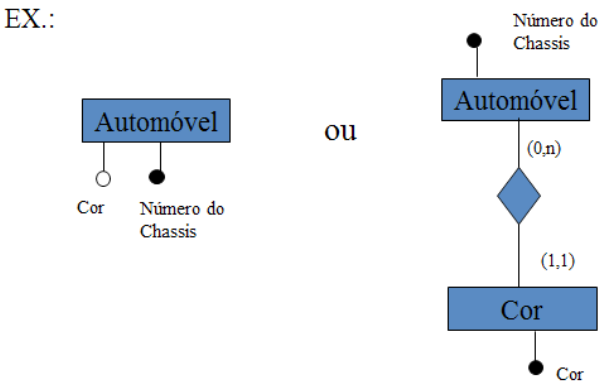
Heuser, 2001

Determinando Construções

- O objeto a ser modelado não pode ser analisado isoladamente.
- É necessário conhecer o contexto, o modelo dentro do qual o objeto esta inserido.

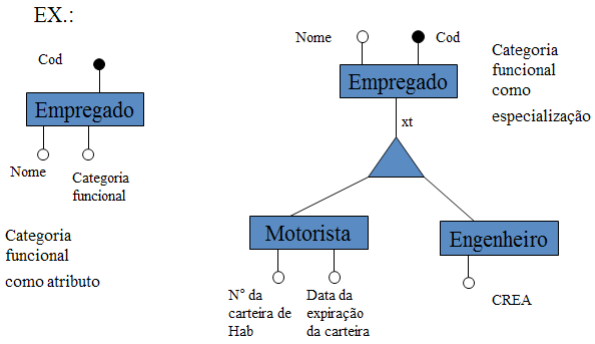
Critérios que podem ser usados na escolha da construção de modelos:

- **Atributo versus entidade relacionada:**
  - Caso o objeto cuja modelagem está em discussão esteja vinculado a outros objetos, caso o objeto tenha propriedades (atributos, relacionamentos, entidades genéricas ou especializadas), o objeto deve ser modelado como entidade.
  - Quando existem transações no sistema que alterem o conjunto de valores do objeto.



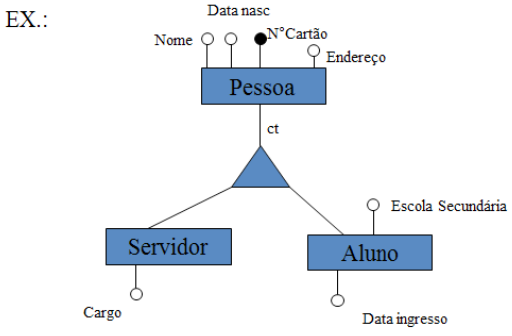
Heuser, 2001

- **Atributo versus especialização**
  - Uma especialização deve ser usada quando se sabe que as classes especializadas de entidades possuem propriedades (atributos, relacionamentos, generalizações, especializações) particulares.

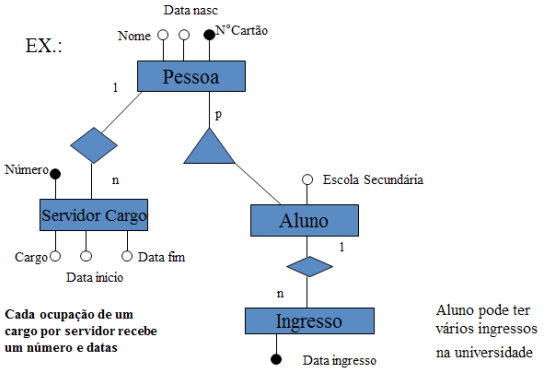


Heuser, 2001

- **Entidade relacionada versus especialização**
  - Para que uma entidade possa ser considerada especialização de outra é necessário que ela herde o identificador (chave primária) da entidade genérica.
  - Para cada ocorrência do objeto genérico pode existir no máximo 1 ocorrência na especialização.



Heuser, 2001

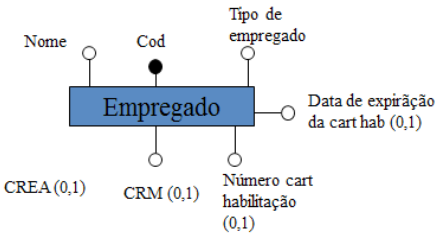


Heuser, 2001

- **Atributos opcionais e multivalorados**
  - Atributo opcional: Indicam subconjuntos de entidades que são modelados mais corretamente através de especializações. Toda vez que aparecer um atributo opcional, é recomendável verificar se a modelagem através de entidades especializadas não é mais conveniente.

No processo de modelagem é aconselhável tentar restringir-se ao uso de atributos obrigatórios e monovalorados.

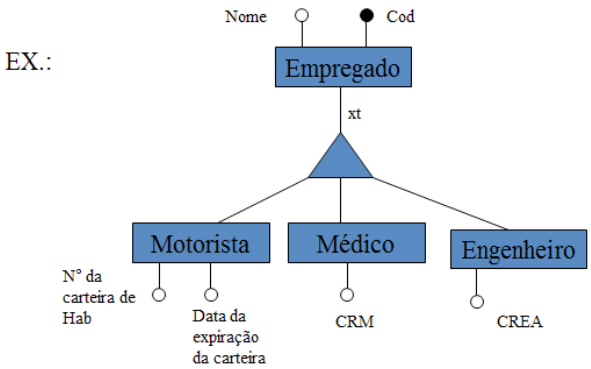
EX.:



Atributo Opcional, Heuser, 2001

No exemplo, o uso de atributos opcionais esconde as diferentes categorias de empregados e suas entidades.

Neste modelo, fica mais claro quais são os atributos de cada um dos subconjuntos particulares de empregado.

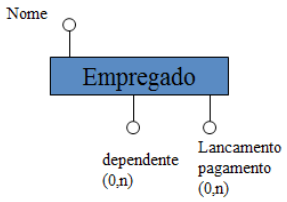


Heuser, 2001

- Atributo multivalorado: Não existe, em um SGBD relacional não existe contrução de arrays. Atributos multivalorados podem induzir a um erro de modelagem, que é o de ocultar entidades e relacionamentos em atributos multivalorados.

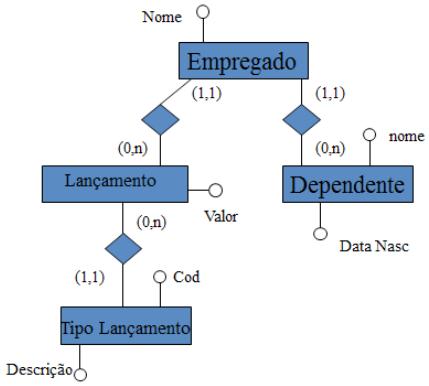
No processo de modelagem é aconselhável tentar restringir-se ao uso de atributos obrigatórios e monovalorados.

EX.:



Usando atributos multivalorados, Heuser, 2001

EX.:



Substituindo atributos multivalorados por entidades relacionadas, Heuser 2001

Abordagem Entidade Relacionamento

Verificação do modelo

Uma vez construído o modelo ER deve ser validado e verificado. A verificação é o controle de qualidade que procura garantir que o modelo usado para a construção do banco de dados gerará um bom produto.

- **Requisitos**
  - Modelo deve ser correto.
  - Modelo deve ser completo.
  - Modelo deve ser livre de redundância.
  - Modelo deve refletir o aspecto temporal.
  - Entidade isolada e entidade sem atributos.

Modelo deve ser correto

O modelo está correto quando não contém erros de modelagem, quando os conceitos de modelagem são corretamente empregados.

- sintáticos
- semânticos

Erros Sintáticos

Ocorrem quando o modelo não respeita as regras de construção de um modelo ER.

Exemplos:

- Associar atributos a atributos.
- Associar relacionamentos a atributos.
- Associar relacionamentos a outros relacionamentos.
- Especializar relacionamentos ou atributos.

Erros Semânticos

Ocorrem quando o modelo, apesar de obedecer às regras de construção de modelo ER (correto sintaticamente), reflete a realidade de forma inconsistente.

Observação

Regras de normalização de base verificam a correção de modelos ER.

Exemplos:

- Estabelecer associações incorretas.
- Usar uma entidade do modelo como atributo de outra entidade.
- Usar número incorreto de entidades em um relacionamento.

Modelo deve ser completo

O modelo deve fixar todas as propriedades desejáveis do BD. Isso somente pode ser verificado por alguém que conhece profundamente o sistema a ser implementado.

Exemplos:

- Verificar se todos os dados que devem ser obtidos do BD estão presentes.
- Verificar se todas as transações de modificação do BD podem ser executadas sobre o modelo.

Modelo deve ser livre de redundância

O modelo deve ser mínimo, não deve conter conceitos redundantes.

Exemplos:

- Relacionamentos redundantes - relacionamentos que são resultados da combinação de outros relacionamentos entre as mesmas entidades.
- Atributos redundantes - são atributos deriváveis a partir da execução de procedimentos de busca de dados e/ou cálculos sobre o BD.

Modelo deve ser livre de redundância

Construções redundantes devem ser omitidas do modelo ER.

- Redundância não controlada - é indesejável.
- Redundância controlada - deve ser necessariamente evitada.

Observação

Em alguns casos, redundância controlada pode servir para aumentar a performance de operações de busca, mas não devem aparecer no modelo conceitual do BD.

Modelo deve refletir aspecto temporal

- Certas aplicações exigem que o BD guarde o histórico de alterações de informação.
- O modelo de BD que armazena somente valores atuais de uma informação é diferente do modelo do BD que armazena o histórico da informação. Portanto, é necessário considerar o aspecto temporal na modelagem de dados.

Observação

Não existem regras gerais de como proceder neste caso.

- Relacionamentos que se modificam ao longo do tempo.
- Assim como atributos podem ser modificados, relacionamentos também podem ser modificados ao longo do tempo.
- Quando é considerada a história de suas alterações, relacionamentos 1:1 ou 1:n são transformados em n:n.
- EX.: Endereço de Cliente, Salário de um empregado

Observação

O relacionamento transforma-se em entidade e deve ter um atributo identificador, normalmente "data".

- Consultas a dados referentes ao passado.
- Muitas vezes para evitar o crescimento do BD, informações referentes ao passado são eliminadas.
- Mas tais informações podem se fazer necessárias no futuro, por motivos legais, auditorias ou tomada de decisão.
- Caso informações antigas fiquem no BD, podem ser necessários atributos para indicar o status da informação, se atual ou antiga.
- Consultas a dados referentes ao passado.
- Planejar o arquivamento de informações antigas.
- Para as informações que serão retiradas do BD e armazenadas em arquivos convencionais, é necessário fazer um planejamento de como essas informações serão acessadas no futuro.
- Uma solução é reincluir as informações no BD, quando elas forem necessárias. Isso permite que informações passadas sejam acessadas da mesma forma que as atuais.

Observação

É necessário observar que informações em um BD estão normalmente relacionadas a outras.

Em alguns casos, informações antigas são necessárias apenas para tomada de decisões. Neste caso, muitas vezes deseja-se apenas dados resultantes de cálculos ou estatísticas sobre as informações, como totais, contagens, médias.

Assim, pode ser conveniente manter no banco de dados estas informações compiladas e eliminar as informações usadas na compilação.

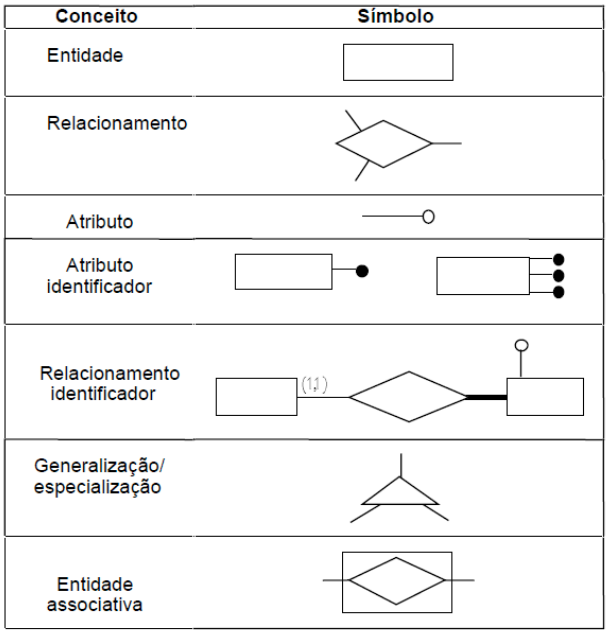
Entidades isoladas e entidades sem atributo

Uma entidade isolada não apresenta nenhuma ligação com outra entidade. Em princípio, entidades isoladas não estão incorretas. Uma entidade que modela a organização na qual o sistema implementado no BD está embutido.

Outra situação que não está incorreta, mas que deve ser investigada é a de uma entidade sem atributos.

Não existem regras gerais de como proceder neste caso.

Modelo Conceitual – Notação



Símbolos usados na construção de esquemas ER's. (Heuser, 2001)

Síntese

Modelo lógico

O modelo conceitual difere do modelo lógico por descrever um nível de abstração dependente do sistema responsável pelo gerenciamento de banco de dados (SGBD). O modelo lógico no nosso caso será desenvolvido sob a ótica de um sistema gerenciador de banco de dados relacional (SGBDr), de maneira que os dados envolvidos no negócio serão interpretados como entidades.

No modelo lógico, as entidades possivelmente apresentem classes de dados que precisam agora ser desdobradas em seus atributos primitivos. Além das classes de dados, que devem ser representadas de forma atômica, é necessário identificar atributos que ainda não foram descritos nas entidades.

Não basta apenas efetuar a expansão da classe de dados, as entidades, de acordo com o negócio, é importante nessa etapa envolver pessoas que conheçam o negócio da empresa. Com essas informações, será possível refinar as entidades com relação ao tipo que representam, por exemplo, se as informações irão referir-se a pessoas físicas ou jurídicas. O atributo tipo não foi identificado no modelo conceitual, de maneira que a descrição daquelas entidades precisa ser reformulada para que possam ser contempladas suas novas características.

Quando as entidades são descritas, podem surgir grupos repetidos de atributos que, na verdade, constituem entidades autônomas que devem ser convenientemente descobertas e descritas para compor o modelo lógico.

Na sequência, será necessário definir os relacionamentos muitos-para-muitos, que costumam esconder dados importantes para o negócio, razão pela qual esses dados devem ser descritos por meio de um elemento de interseção como uma entidade associativa. Um relacionamento muitos-para-muitos pode ser transformado em entidade associativa, neste caso, o modelo lógico deve ser manipulado, para que todos os relacionamentos muitos-para-muitos sejam convertidos em relacionamentos um-para-muitos.

Dica

Quando identificarmos um relacionamento muitos-para-muitos ele deverá se transformar em uma entidade.

Quando no modelo lógico, as entidades apresentem associações consigo mesmas, devemos identificar os autorrelacionamentos, também conhecidos como relacionamentos involutos, reflexivos ou recursivos.

Após o ajuste das entidades e relacionamentos, devemos voltar a atenção para os atributos, que são, em última análise, os elementos de maior importância na construção do modelo lógico. Quando esquecemos ou definimos erroneamente um atributo é possível que informações relevantes para o negócio não fiquem disponíveis no modelo. O descuido na definição dos atributos pode trazer muitos transtornos ou prejuízos ao projeto do banco de dados que se pretende construir a partir do modelo lógico.

Devemos ter bastante cuidado no momento da modelagem de uma entidade, em um primeiro momento devemos supor que os seus atributos sejam obrigatórios, ou seja, devem estar presentes em toda a ocorrência da entidade. Como isso nem sempre acontece, é preciso identificar os atributos considerados opcionais. Um atributo pode ser opcional apenas por decisão do negócio, neste caso, devemos investigar se o atributo é um dado relevante ou não para o negócio, o tributo pode ser considerado opcional se não for imprescindível. Devemos analisar a opcionalidade dos atributos, este critério pode indicar a existência de entidades especializadas, principalmente nos casos em que o valor de um atributo depende do valor contido em outro.

Quanto aos atributos multivalorados, devemos observar a necessidade de mantermos esses dados em uma entidade específica para garantirmos a integridade referencial do Projeto de Banco de Dados.

D

Modelo entidade-relacionamento

Unidade D
Projeto de Banco de Dados Relacional

UNIDADE D

MODELO ENTIDADE-RELACIONAMENTO

A partir de modelo desenvolvido por Peter Chen em 1976, muitas extensões e notações foram definidas ao longo do tempo. Fornecem ao usuário um alto nível de abstração e, por conseguinte, facilitam a construção de um esquema de BD. A estrutura lógica do Banco de Dados pode ser expressa graficamente pelo diagrama E-R, possibilitando simplicidade e expressividade na descrição do Banco de Dados.

Um banco de dados representado por um modelo E-R, pode ser representado por uma coleção de tabelas. O mapeamento entre os modelos E-R e Relacional é relativamente simples, existem várias ferramentas destinadas a mapear o Modelo E-R para Relacional. O Modelo Entidade Relacionamento é também chamado de esquema E-R ou diagrama E-R.

Elementos que compõem o modelo:

- Entidade: elemento do conjunto de entidades identificado por características individuais definidas por meio do conceito de atributos.
  - Ex: “coisas”, objetos, pessoas
- Atributos: Propriedades que descrevem a entidade ou o relacionamento entre entidades.

Relacionamento: conjunto de associações entre conjunto de entidades; podem ser caracterizados por atributos.

Entidade

O conceito fundamental da abordagem Entidade-Relacionamento (ER) é o conceito de entidade (HEUSER, 2001).

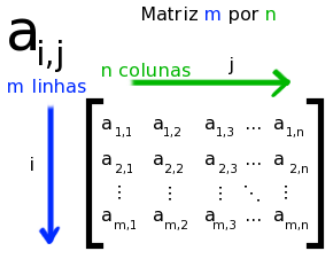
entidade
=
conjunto de objetos1 da realidade modelada
sobre os quais deseja-se manter informações
no banco de dados

Entidades são tabelas que armazenam informações em um banco de dados.

Cliente

ClienteID	Nome	Telefone
234	Maria Olive	9981-8765
567	Murilo Perez	8123-9087 6125-9087
765	Rosa Maria Vaz	9276-0832

Tabelas são estruturas matriciais compostas por linhas e colunas.



Organização de uma matriz

A tabela é um conjunto de dados dispostos em número finito de colunas e número ilimitado de linhas.

Cliente

ClienteID	Nome	Telefone
234	Maria Olive	9981-8765
567	Murilo Perez	8123-9087 6125-9087
765	Rosa Maria Vaz	9276-0832

As colunas são tipicamente consideradas os campos da tabela e caracterizam os tipos de dados que deverão constar na tabela (numéricos, alfa-numéricos, datas, coordenadas, etc).

Cliente

ClienteID	Nome	Telefone
234	Maria Olive	9981-8765
567	Murilo Perez	8123-9087 6125-9087
765	Rosa Maria Vaz	9276-0832

O número de linhas pode ser interpretado como o número de combinações de valores dos campos da tabela, e pode conter linhas idênticas, dependendo do objetivo.

Cliente

ClienteID	Nome	Telefone
234	Maria Olive	9981-8765
567	Murilo Perez	8123-9087 6125-9087
765	Rosa Maria Vaz	9276-0832

Importante: Uma linha também pode ser chamada Registro, Instância ou Tupla em uma tabela.

Acesso a tabelas

As tabelas podem ser acessadas por quaisquer critérios envolvendo os campos de uma ou mais linhas. Programadores escrevem consultas sem considerar a existência de caminhos de acesso.

- Caminho de acesso:
  - estrutura auxiliar (índice, cadeia de ponteiros,...);
  - a recuperação de registros por determinados critérios;
  - evita a leitura exaustiva de todos os registros de um arquivo.

Chaves

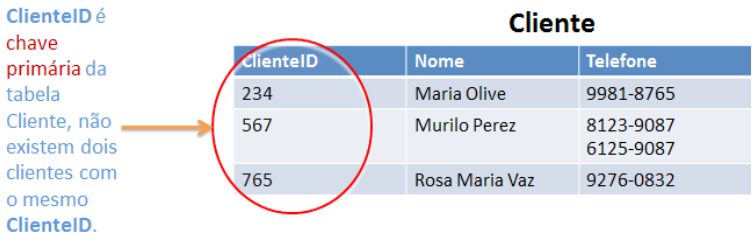
Conceito usado para especificar restrições de integridade básicas de um SGBD relacional.

Três tipos:

- chave primária
- chave alternativa
- chave estrangeira

Chave primária

A forma de referenciar inequivocamente uma única linha é através da utilização de uma chave primária.



Importante

As chaves primárias devem ser identificadas no modelo, normalmente é utilizado um símbolo gráfico para efetuar a identificação.

Chaves primárias (em inglês **Primary Keys** ou **PK**), sob o ponto de vista de um Banco de Dados Relacional, referem-se aos conjuntos de um ou mais campos, cujos valores, considerando a combinação de valores de todos os campos da tupla, **nunca se repetem** e que podem ser usadas como um índice para os demais campos da tabela do banco de dados. Em chaves primárias, **não** pode haver **valores nulos** nem **repetição de** tuplas.

Características de Chave Primária:

- Não pode haver duas ocorrências de uma mesma entidade com o mesmo conteúdo na Chave Primária.
- A chave primária não pode ser composta por atributo opcional, ou seja, atributo que aceite nulo.
- Os atributos identificadores devem ser o conjunto mínimo que pode identificar cada instância de uma entidade.
- Não devem ser usadas chaves externas. (Atributos sobre os quais você não tem controle Ex: CPF).
- Cada atributo identificador da chave deve possui um tamanho reduzido.
- Não deve conter informação volátil.

Chave Alternativa

Mais de uma coluna ou combinações de colunas podem servir para distinguir uma linha das demais.

- Uma das colunas (ou combinação de colunas) é escolhida como chave primária.
- As demais colunas ou combinações são denominadas chaves alternativas.

Chave Alternativa é a chave **candidata** que não é a chave primária.

Fornec_ID	Nome	Cidade	CNPJ
123	ABC Peças	São Paulo	35.124.562/0001-07
124	Ind. Peças Dinamo	Recife	35.042.105/0001-04
125	Unidas Peças SA	Porto Alegre	35.520.555/0001-05
126	Com. Brasil Ltda	São Paulo	35.802.053/0001-09

Importante

A chave alternativa pode possibilitar a identificação de uma linha em uma tabela, desde que ela seja única.

Chave Estrangeira

Uma coluna ou uma combinação de colunas, cujos valores aparecem necessariamente na chave primária de uma tabela, mecanismo que permite a implementação de relacionamentos em um banco de dados relacional.

Importante

A chave estrangeira é quando um atributo de uma entidade (tabela) é a chave primária de outra entidade.

Vejamos a tabela abaixo, temos a primeira Tabela_Funcionário e a Segunda Tabela_Dependente, obsevem que na segunda temos uma chave estrangeira, vinda de outra tabela.

Funcionário

Func_ID	Nome	CPF
1056	João da Silva	36589612001
1058	Mária Albuquerque	78923645601
1060	Sueli Alves	12578526914
1062	Willian Porto	78245612863

Dependente

Func_ID	Depen_ID	Tipo	CPF
1056	123	Esposo	36589612001
1058	566	Filho	78923645601
1060	523	Filho	12578526914
1062	189	Esposo	78245612863

Func_ID é chave primária em Funcionário e chave Estrangeira em Dependente

- Quando da inclusão de uma linha na tabela que contém a chave estrangeira o valor da chave estrangeira deve aparecer na coluna da chave primária referenciada.
- Quando da alteração do valor da chave estrangeira o novo valor de uma chave estrangeira deve aparecer na coluna da chave primária referenciada.

Chave Primária Composta

Chave primária composta é aquela criada em dois campos e, dessa forma, passa a utilizar a junção dos dados dos dois campos indicados para formar um valor único e assim aplicar o bloqueio de duplicidade.

Fornecedor

Fornec_ID	Nome	Cidade
123	ABC Peças	São Paulo
124	Ind. Peças Dinamo	Recife
125	Unidas Peças SA	Porto Alegre
126	Com. Brasil Ltda	São Paulo

Peça

Peças_ID	Nome	Medida
10	Porca	8
20	Parafuso	10
30	Prego	15
40	Rebite	12

Pedido

Peças_ID	Fornec_ID	Qtde
10	123	80
20	123	100
30	124	155
40	124	120
20	125	250
30	125	100
30	126	90
40	126	136

Na tabela Pedido os campos Peças_ID e Fornec_ID são chaves primárias.

Desta forma mantemos a integridade dos dados pois todos dos fornecedores podem fornecer a mesma peça.

Relacionamento

- Relacionamento é uma associação entre uma ou várias entidades.
- Conjunto de Relacionamentos é um conjunto de relacionamentos de mesmo tipo.
- Expressam uma rica semântica entre os conjuntos de entidades por meio dos conceitos como:
  - Cardinalidade
  - Restrição de participação (total ou parcial)
  - Grau de Relacionamento

Esses conceitos impõem restrições aos dados que alimentarão o banco de dados.

- Notação DER: losango

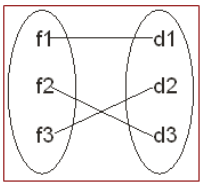
- O grau de relacionamento: é o número de entidades participantes.
  - Binário, ternário, etc.
- Dependendo do conjunto de entidades associadas entre si, é necessário adicionar atributos em um relacionamento.
  - Ex: Horas em Trabalha_Em entre Funcionário e Projeto

Cardinalidade

A cardinalidade expressa o número de entidades às quais outra entidade pode estar associada em um relacionamento.

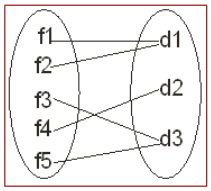
- **Tipos de Relacionamento:**
  - Um para um (1 para 1)
  - Um para muitos (1 para N)
  - Muitos para muitos (N para N)
- **Um para um (1 para 1):**
  - Uma entidade em A está associada no máximo a uma entidade em B, e uma entidade em B está associada

no máximo a uma entidade em A.



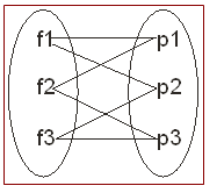
Leitura: Um funcionário pode gerenciar no máximo 1 departamento. Um departamento é gerenciado por no máximo 1 funcionário.

- **Um para muitos (1 para N):**
  - Uma entidade em A está associada a várias entidades em B. Uma entidade em B, entretanto, deve estar associada no máximo a uma entidade em A.



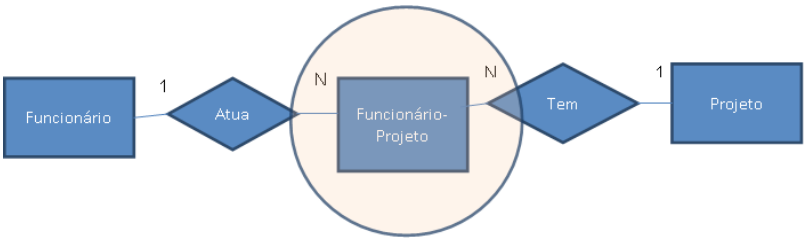
Leitura: Um funcionário está lotado no máximo em 1 departamento. Um departamento tem até N funcionários.

- **Muitos para muitos (N para N) - pode ser substituído por qualquer outra letra, como M, P, Q):**
  - Uma entidade em A está associada a qualquer número de entidades em B e uma entidade em B está associada a um número qualquer de entidades em A

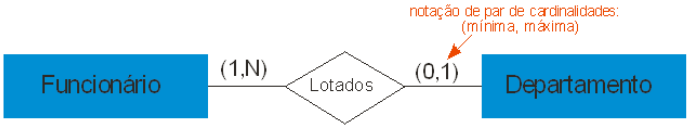


Leitura: Um funcionário participa de N projetos. Um projeto pode ter a participação de N funcionários.

Quando tabelas têm entre si relação n..n, é necessário criar uma nova tabela com as chaves primárias das tabelas envolvidas, ficando assim uma chave composta, ou seja, formada por diversos campos-chave de outras tabelas. A relação então se reduz para uma relação 1..n, sendo que o lado n ficará com a nova tabela criada.



- **Cardinalidade Máxima e Mínima:**
  - Indica se a participação das ocorrências de entidades no relacionamento é obrigatória ou opcional.



Leitura: *Um funcionário* ***pode estar*** *lotado no máximo em 1 departamento. Um departamento* ***obrigatoriamente*** *tem até N empregados.*

Grau de Relacionamento

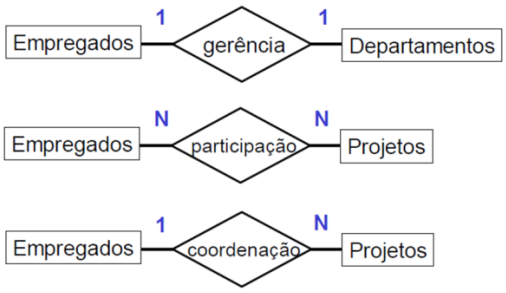
Indica quantos conjuntos de entidades estão envolvidos em determinado relacionamento. Os relacionamentos podem ter associado vários conjuntos de entidades, caracterizando:

- relacionamentos binários (grau de relacionamento 2);
- relacionamentos ternários (grau 3);
- relacionamentos quaternários (grau 4), entre outros.

É importante observar que um relacionamento com grau N > 2 só é justificável se não puder ser decomposto em relacionamentos com graus menores e ainda manter a semântica desejada.

Relacionamentos Binários

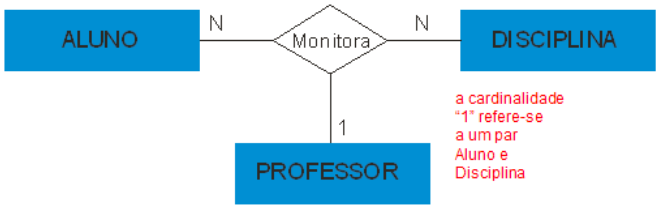
Um relacionamento binário é aquele cujas ocorrências envolvem duas Entidades.



Exemplos

Relacionamentos Ternários

Os relacionamentos entre múltiplas entidades expressam um fato em que todas as entidades ocorrem simultaneamente, ou seja, todas as ocorrências do relacionamento possuem, sempre, ligações com todas as entidades envolvidas no relacionamento.



OBS: O "1" na linha que liga o retângulo representativo da entidade PROFESSOR ao losango representativo do relacionamento expressa que cada par de ocorrências (Aluno, Disciplina) está associado no máximo a um Professor. Em outros termos, não há concorrência pela distribuição de um Aluno em uma Disciplina.

Autorrelacionamento

Representa uma associação entre ocorrências de uma mesma entidade, requer a identificação de papéis.

Papel do relacionamento: utilizado quando não é clara a participação de um determinado conjunto de entidades em um relacionamento.

A representação do papel é extremamente importante, quando se trata de um relacionamento unário ou autorrelacionamento.

- Exemplo:

Leitura: Um Funcionário pode ser supervisionado por no máximo 1 Funcionário. Um Funcionário pode supervisionar no máximo N Funcionários.

- **Modelo - Notação**

Conectividade	Peter Chen	James Martin
1:1		
1:N	1 N	
N:N	N N	
Existência		
Obrigatório	1 N	
Opcional	1 N	

Notações utilizadas para representar relacionamentos entre entidades.

Modelo Conceitual – Atributo

- Propriedades usadas para descrever uma entidade
- Domínio de Atributo
  - Conjunto de valores possíveis
  - **Exemplo – Entidade Funcionário**
    - **Identificador:** inteiro não-negativo
    - **Nome:** cadeia de caracteres
    - **Idade:** inteiro não-negativo
    - **Sexo:** Masculino ou Feminino

Atributo Simples

- Assume um único valor atômico para cada entidade.

Os atributos simples são aqueles que guardam apenas um dado como, por exemplo, nome, sobrenome, etc.

Atributo Composto

- Formado por um ou mais subatributos.

Os atributos composto são aqueles que possuem mais de um dado, por exemplo, endereço (possui rua, número, bairro, etc.).

Atributo Multivalorado

- Pode possuir diversos valores para uma única entidade.

Os atributos Multivalorados reúnem um ou mais dados do mesmo tipo, por exemplo, telefone, uma pessoa pode ter um telefone residencial, celular e comercial, etc.

Atributo Derivado

- Atributo cujo valor pode ser derivado a partir de outro atributo (base).

Modelo Conceitual - Atributo

Podem ser:

- Atributo Identificador (Chave) - Permite identificar univocamente cada entidade em um conjunto entidade.
- Chave Candidata - Atributo ou conjunto de atributos que podem identificar uma entidade.
- Chave Primária - Chave candidata escolhida para o esquema do conjunto entidade.

Modelo lógico

Unidade E
Projeto de Banco de Dados Relacional

UNIDADE E

MODELO LÓGICO

Apresenta o Banco de Dados no nível do SGBD, depende do SGBD que será usado.

Um modelo lógico de dados para uso meramente operacional/transacional deve:

- Ser completamente normalizado;
- Representar fielmente o NEGÓCIO, e NÃO necessariamente a base de dados desejada, a qual será construída posteriormente por ocasião do Projeto Físico;
- Conter descrição sucinta das entidades, atributos e relacionamentos.

Um modelo lógico de dados para uso meramente operacional/transacional deve:

- Conter os nomes de entidades e atributos, extensos e abreviados, atribuídos de acordo com algum padrão adotado na organização e formados por termos previamente convencionados em um glossário;
- Contemplar, para cada um dos atributos, o tipo de dado, tamanho e opcionalidade.

Um Modelo Lógico de Dados para uso meramente operacional/transacional não deve conter:

- Replicações de atributos
- Atributos derivados
- Atributos repetitivos

- **Replicações de atributos:** fisicamente pode ser interessante alguma redundância com o objetivo de melhorar a performance de determinado(s) processo(s). No modelo lógico isso não pode ser feito; um atributo só é representado na Entidade a que pertence.
- **Atributos derivados:** pelos mesmos motivos apontados anteriormente, a implementação das tabelas pode requerer o armazenamento de uma informação derivada de outra(s) (valor do saldo, por exemplo). Tal tipo de informação não se constitui em um atributo do modelo lógico.
- **Atributos repetitivos:** o uso de atributos repetidos, como Telefone-1 e Telefone-2, não é admitido. Se existe a possibilidade de uma pessoa possuir mais de um telefone, então Telefone deve ser representado como uma entidade, mantendo relacionamento Nx1 com a entidade Pessoa.

Os SGBD's relacionais são os mais difundidos, neste tipo de SGBD os dados são organizados em tabelas.

O modelo lógico do BD relacional deve definir quais as tabelas e o nome das colunas que compõem estas tabelas.

Modelo Lógico – Tabelas

Aluno		
matricula	Nome_alu	Endereco_alu
20111	José Maria	Rua 3 bloco A apto 201A
20112	Martha Regina	Pça 20 de setembro, 101
20113	Silvia Machado	Alameda das Flores, 4
20114	Regina da Costa	Rua Conceição, 56 casa 2

Turma		
turma	sala	turno
11	8	Manhã
22	13	Vespertino
33	21	Noite

Apresenta a estrutura das tabelas, define quais as tabelas e o nome das colunas que compõem estas tabelas.

Exemplo de modelo lógico:

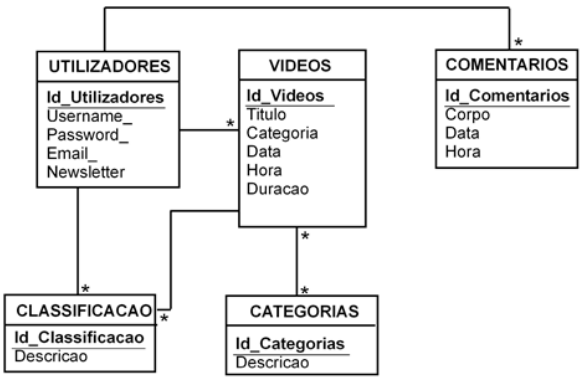
Aluno (matricula, nome_alu, endereco_alu)
Turma (turma, sala, turno)

Modelo Lógico x Modelo Físico

Detalhes internos de armazenamento, não são descritos no modelo lógico, pois estas informações fazem parte do modelo físico, que nada mais é que a tradução do modelo lógico para o SGBD definido para a criação do Banco de Dados.

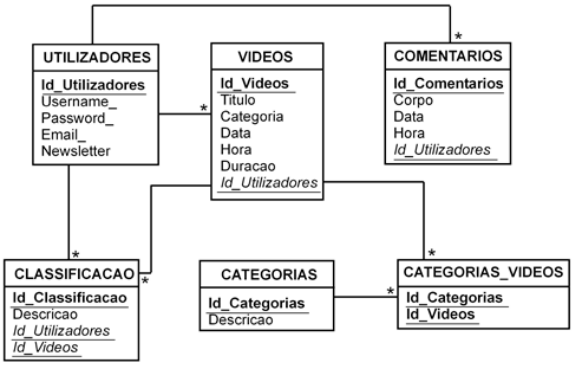
Modelo Lógico

O modelo lógico leva em conta algumas limitações e implementa recursos como adequação de padrão e nomenclatura. Define as chaves primárias e estrangeiras. Deve ser criado levando em conta os exemplos de modelagem de dados criados no modelo conceitual.



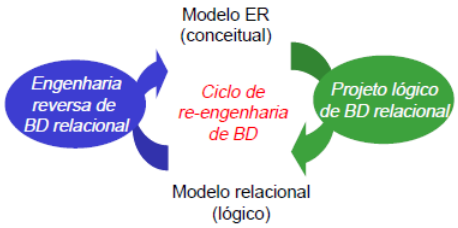
Modelo Físico

No modelo físico fazemos a modelagem física do modelo de banco de dados. Deve-se levar em conta as limitações impostas pelo SGBD escolhido e ser criado sempre com base nos exemplos de modelagem de dados produzidos no modelo lógico.

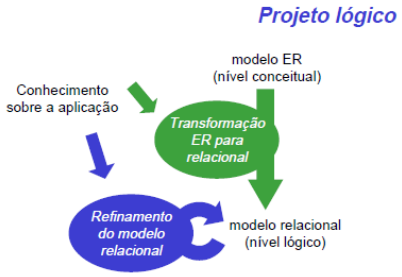


Transformações entre modelos

A partir do Modelo Conceitual do BD teremos subsídios para a construção do Modelo Lógico.



Heuser, 2001

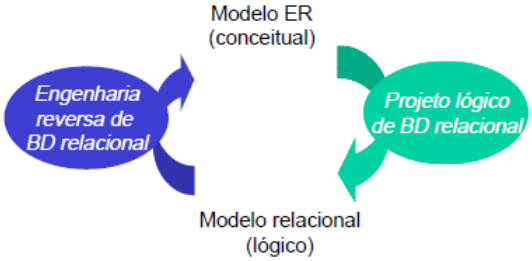


Heuser, 2001

Engenharia reversa de modelos relacionais

Engenharia Reversa:

- Parte de modelo de implementação.
- Obtém modelo de especificação (modelo conceitual).



Heuser, 2001

Passos:

- Identificação da construção ER correspondente a cada tabela.
- Definição de relacionamentos 1:n e 1:1.
- Definição de atributos.
- Definição de identificadores de entidades e relacionamentos.

Esquema relacional para engenharia reversa

Disciplina (CodDisc,NomeDisc)
Curso (CodCr,NomeCr)
Curric (CodCr,CodDisc,Obr/Opc)
CodCr referencia Curso
CodDisc referencia Disciplina
Sala (CodPr,CodSl,Capacidade)
CodPr referencia Prédio
Prédio (CodPr,Endereço)
Turma (Anosem,CodDisc,SiglaTur,Capacidade,CodPr,CodSl)
CodDisc referencia Disciplina
(CodPr,CodSl) referencia Sala
Laboratório (CodPr,CodSl, Equipam)
(CodPr,CodSl) referencia Sala

Heuser, 2001

Identificação da construção ER correspondente a cada

Uma tabela pode corresponder a:

- Uma entidade
- Um relacionamento n:n
- Uma entidade especializada
- Um ator determinante
- Uma composição de sua chave primária

Tipos de tabelas para identificação de construção ER

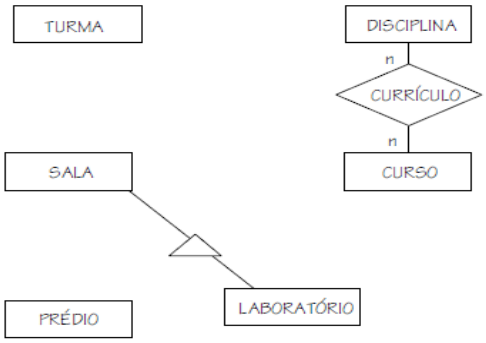
Composição da chave primária	Construção ER correspondente
Múltiplas chaves estrangeiras	Relacionamento n:n
Toda chave primária é uma chave estrangeira	Especialização
Demais casos	Entidade

Heuser, 2001

Construções identificadas

Disciplina (CodDisc,NomeDisc) **entidade**
Curso (CodCr,NomeCr) **entidade**
Curric (CodCr,CodDisc,Obr/Opc) **relacionamento n:n**
CodCr referencia Curso
CodDisc referencia Disciplina
Sala (CodPr,CodSl,Capacidade) **entidade**
CodPr referencia Prédio
Prédio (CodPr,Endereço) **entidade**
Turma (Anosem,CodDisc,SiglaTur,Capacidade,CodPr,CodSl) **entidade**
CodDisc referencia Disciplina
(CodPr,CodSl) referencia Sala
Laboratório (CodPr,CodSl, Equipam) especialização
(CodPr,CodSl) referencia Sala

Heuser, 2001



Heuser, 2001

Identificação de relacionamentos 1:n ou 1:1

Chave estrangeira que não se enquadra nas regras acima

Representa:

- relacionamento 1:n

ou

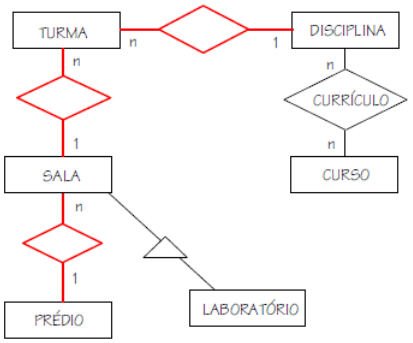
- Relacionamento 1:1

Esquema não informa se é 1:1 ou 1:n

Disciplina (CodDisc,NomeDisc)
Curso (CodCr,NomeCr)
Curric (CodCr,CodDisc,Obr/Opc)
CodCr referencia Curso
CodDisc referencia Disciplina
Sala (CodPr,CodSl,Capacidade)
CodPr referencia Prédio
Prédio (CodPr,Endereço)
Turma (Anosem,CodDisc,SiglaTur,Capacidade,CodPr,CodSl)
CodDisc referencia Disciplina
(CodPr,CodSl) referencia Sala
Laboratório (CodPr,CodSl, Equipam)
(CodPr,CodSl) referencia Sala

Chaves estrangeiras por tratar indicadas em vermelho

Heuser, 2001



Heuser, 2001

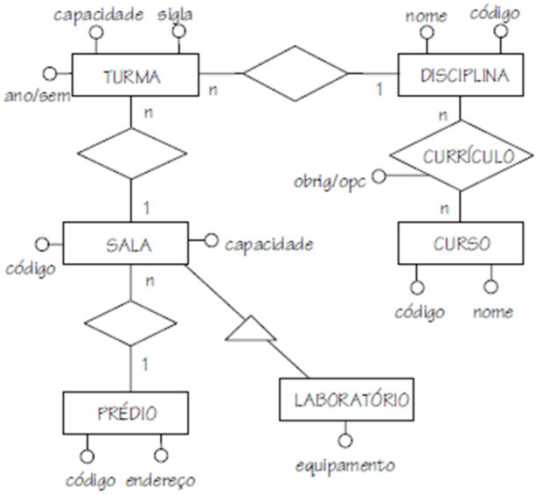
Definição de atributos

Cada coluna não chave estrangeira é

- Um atributo na entidade/relacionamento correspondente à tabela

As colunas chave estrangeira não correspondem a atributos

- Correspondem a relacionamentos
- Já foram tratadas nas etapas anteriores



Heuser, 2001

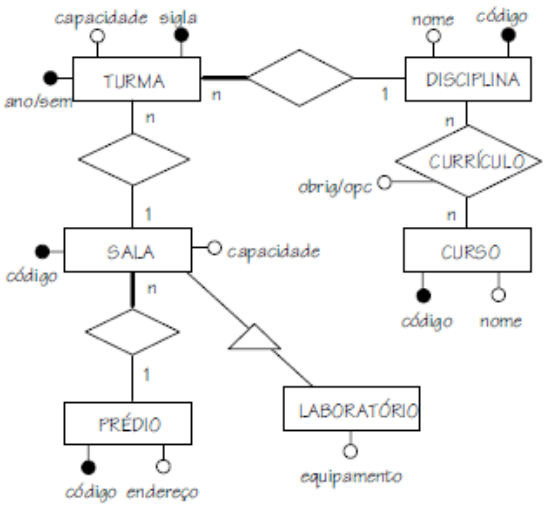
Definição de identificadores de entidades

Coluna da chave primária que não chave estrangeira

- Corresponde a um atributo identificador da entidade ou relacionamento.

Coluna da chave primária que é chave estrangeira

- Corresponde a um relacionamento identificador da entidade.



Heuser, 2001

Administração do Modelo de Dados

Ferramentas Case

- Dr. Case (www.squadra.com.br)
- ER/Studio (www.embarcadero.com)
- DB-MAIN (www.db-main.be)
- DBDesigner (www.fabforce.net/dbdesigner4)
- ER-WIN (www.ca.com/us/products/product.aspx?id=260

Dicas

Geralmente, o processo de modelagem é iterativo, onde:

- Inicialmente identificamos e representamos os conjuntos de entidades (fortes e fracas);
- Identificamos e representamos os conjuntos de relacionamentos;
- Procuramos os atributos das entidades e dos relacionamentos e o domínio destes atributos;
- Definindo as superchaves, chaves candidatas e chaves parciais;
- Definimos os tipos de atributos (simples/compostos, mono/multi valorados, armazenado/derivado);
- Definimos as cardinalidades dos conjuntos de relacionamentos;
- Definimos a participação (total ou parcial) de cada papel de relacionamento;
- Refinamos o modelo ER, a fim de verificar se este atende às necessidades das diferentes visões.

Síntese

Modelo Lógico de Dados

Modelos de Dados

A modelagem de dados é o mecanismo utilizado para representar um conjunto de requerimentos de informações de negócio. É uma etapa fundamental para a implementação de um sistema de informação.

Através da modelagem de dados é possível identificar as operações que deverão ser executadas pelo sistema, bem como as informações e os dados que serão manipulados por ele.

Atualmente existem três tipos de modelos de dados: o conceitual, o lógico e o físico.

- **Modelo Conceitual** – Representa os conceitos e características do ambiente ignorando detalhes de implementação.
- **Modelo Lógico** - Apresenta o Banco de Dados no nível do SGBD, depende do SGBD que será usado.
- **Modelo Físico** – Preocupa-se com os recursos necessários para armazenamento e manipulação dos dados (estrutura de armazenamento, endereçamento, acesso e alocação física).

O modelo lógico de dados representa de forma lógica as informações referentes ao negócio e está desvinculado do SGBD. Tendo em vista que as tecnologias mudam muito rapidamente, este modelo deve ser independente da tecnologia, possibilitando assim que sua representação permaneça fiel às regras de negócio.

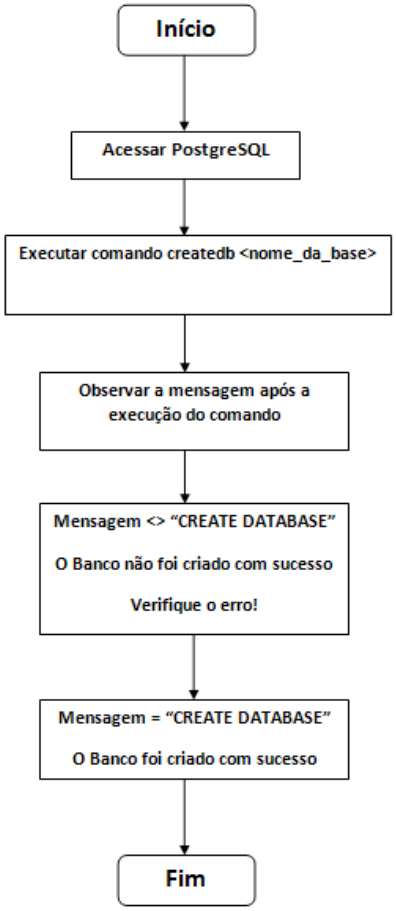
O modelo relacional foi definido por E.F. Codd em 1970. O modelo relacional está alicerçado em dois aspectos básicos: a álgebra que possibilita a manipulação dos dados nas tabelas e um mecanismo de definição de relações.

Cada entidade no Modelo de negócio dá origem a uma relação. Neste caso, o nome da relação é o nome lógico da entidade. As propriedades da entidade tornam-se atributos da relação e o identificador da entidade passa a ser chave primária da relação.

O Modelo Lógico de Dados deve primar pela integridade da base de dados e deve assegurar uma constante concordânciados dados com o mundo real que a base de dados deve representar. Um dos aspectos mais importantes relacionados com a integridade dos dados é a integridade semântica.

A integridade semântica deve assegurar que os dados armazenados estão coerentes em relação ao seu significado. Para isso, o conjunto das restrições de integridade deverá ser respeitado pelos dados da base.

Um modelo lógico de dados deve representar todas as informações necessárias para a criação de um banco de dados, onde as atividades operacionais das empresas possam ser controlada se armazenadas. Este modelo pode ser representado de várias formas diferentes, existem várias ferramentas no mercado que nos possibilitam fazer isso. O Modelo de Entidade Relacionamento (MER) é uma das técnicas mais usadas para representar um modelo lógico de dados.



Fluxograma para criação de uma Base de Dados

Referências

Documentação PostgreSQL 8.2.0. Disponível em: <http://pgdocptbr.sourceforge.net/pg82/tutorial-createdb.html> - acesso em 08/2011

Projeto de banco de dados

Unidade F

Projeto de Banco de Dados Relacional

UNIDADE F

PROJETO DE BANCO DE DADOS

Projeto de Banco de Dados tem a finalidade de criar Bases de Dados que atendam às necessidades das organizações quanto ao armazenamento, acesso à informação e segurança dos dados.

Um Projeto de Banco de Dados é composto pelo Projeto Conceitual, Lógico e Físico.

Características

Modelagem de dados: objetiva transmitir e apresentar uma representação única, não redundante e resumida, dos dados de uma aplicação.

No mercado, o Modelo Entidade-Relacionamento (MER) é o mais utilizado para representar as aplicações em um nível conceitual.

Modelo Entidade-Relacionamento (ER)

- Foi definido por Peter Chen em 1976, e teve como base a teoria relacional criada por E. F. Codd (1970).
- Ao longo dos anos, vários estudiosos evoluíram e expandiram este "metamodelo".

Vantagens

- Independente de detalhes de implementação em um SGBD.
- Pode ser mapeado para qualquer modelo lógico de BD
- Facilita a manutenção do modelo lógico e a migração para outro modelo lógico.
- Também chamado de esquema ER ou diagrama ER.

Modelagem Conceitual

É constituída de Entidades e Relacionamentos que permitem visualizar a essência do sistema, não sendo representados procedimentos ou fluxo de dados existentes.

Para realizar a modelagem conceitual, esqueça os procedimentos do sistema. Deve-se apenas relatar as informações pertencentes ao sistema.

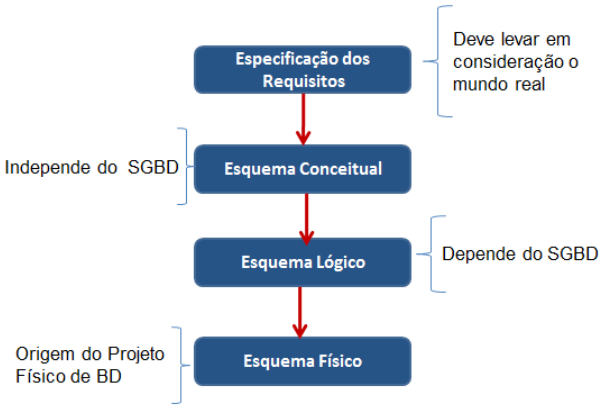
Objetos Conceituais

Quando Peter Chen formulou a proposta do MER, baseou-a não na visão de um sistema de aplicação como princípio e sim na compreensão da realidade em que se situava o problema.

Chen dedicou-se a destacar a importância de reconhecer os objetos que compõem este negócio, independentemente de preocupar-se com formas de tratamento das informações, procedimentos, programas, etc. Ele classificou os objetos conceituais em 2 grupos: **Entidades e Relacionamentos.**

Fases do Projeto de Bases de Dados

- Conceitual: envolve as definições e ligações dos dados pertencentes ao sistema estudado. Abstração de mais alto nível.
- Lógico: representação do projeto conceitual em um modelo lógico (ex: orientado a objeto, relacional, objeto-relacional, etc.)
- Físico: implementação do projeto lógico em um Sistema Gerenciador de Banco de Dados (SGBD), a fim de representar o modelo lógico de uma forma computacional.



Fases do Projeto de Base de Dados

Especificação dos Requisitos

Entrada : Mundo Real

- deve considerar a realidade a organização;
- a informação a ser tratada com todas suas propriedades, requisitos e restrições;
- Informação inexata.

Processo

- coleta e análise;
- entrevistas, documentos, formulários, observações.

Resultado : Especificação de Requisitos

- descrição em linguagem natural;
- descrição em formatos mais estruturados (gráficos);
- utilização de questionários, formulários e documentos em geral.

Modelagem Conceitual

Entrada : Especificação dos Requisitos

- Apresentação
- informal, dúbio, incompleto, redundante, contraditório, incoerente, ...
- longo, de difícil manipulação.

Processo

- Modelagem, de acordo com um modelo conceitual
- Entidade Relacionamento

Resultado : Esquema Conceitual

- descrição sucinta (diagramas e texto);
- clara, concisa, sem ambiguidades, sem contradições;
- utilização de um padrão.

Projeto Lógico

Entrada : Esquema Conceitual

Processo

- mapeamento;
- regras padronizadas em função do modelo conceitual usado e a família do SGBD utilizado.

Resultado : Esquema Lógico

- descrição das estruturas de representação na base de dados;
- depende da família de SGBD.

Projeto Físico

Entrada: Esquema Lógico

Processo

- Escolha das estruturas de armazenamento e métodos de acesso.
- Leva em consideração o SGBD utilizado.
- Realimenta o esquema lógico.

Resultado

- Descrição do esquema na Linguagem de Definição do SGBD.

O Projeto Físico é a descrição de um banco de dados no nível de abstração visto pelo usuário do SGBD, este modelo depende do SGBD que está sendo usado.

Nesta etapa, são detalhados os componentes da estrutura física do banco, como tabelas, campos, tipos de valores, índices, etc.

Com o projeto físico concluído o banco de dados propriamente dito, poderá ser criado com a utilização de um SGBD.

Modelo físico: descreve os detalhes de armazenamento (interno) dos dados e das formas de acesso a esses dados.

- São derivados a partir do respectivo modelo lógico.
    - Possui: detalhes de armazenamento: organização de arquivos e acesso aos dados: utilização de índices.

Ferramentas CASE

São ferramentas baseadas em computadores que auxiliam atividades de engenharia de software, desde análise de requisitos e modelagem até programação e testes.

- CASE - Computer-Aided Software Engineering

Administração do Modelo de Dados

Ferramentas Case para Banco de Dados:

- Dr. Case (www.squadra.com.br)
- ER/Studio (www.embarcadero.com)
- DB-MAIN (www.db-main.be)
- DBDesigner (www.fabforce.net/dbdesigner4)
- ER-WIN (www.ca.com/us/products/product.aspx?id=260

Dicas

O processo de modelagem é iterativo, onde é necessário:

- Identificar e representar os conjuntos de entidades (fortes e fracas);
- Identificar e representar os conjuntos de relacionamentos;
- Definir os atributos das entidades e seus relacionamentos e os domínios.

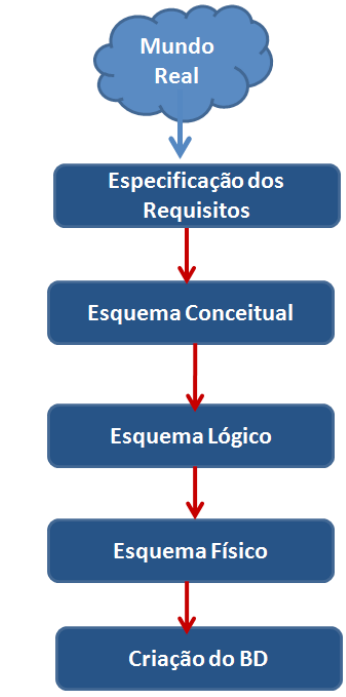
O processo de modelagem é iterativo, onde é necessário:

- Definir as chaves primárias, chaves candidatas e chaves parciais;
- Definir os tipos de atributos (simples ou compostos, mono ou multivalorados, armazenado ou derivado);
- Definir as cardinalidades dos conjuntos de relacionamentos.

O processo de modelagem é iterativo, onde é necessário:

- Definir a participação (total ou parcial) de cada papel de relacionamento;
- Refinar o modelo ER a fim de verificar se este atende às necessidades do mundo real.

Síntese



Etapas de um projeto de banco de dados

TICs

G

Linguagem de consulta estruturada SQL

Unidade G
Projeto de Banco de Dados Relacional

UNIDADE G

LINGUAGEM DE CONSULTA ESTRUTURADA – SQL

Linguagem de Definição de Dados (DDL).

Desenvolvida pelo departamento de pesquisa da IBM na década de 1970 (System R) - Sequel.

Linguagem padrão de Bancos de Dados Relacionais apresenta vários padrões evolutivos, que foram evoluindo com o tempo: SQL86, SQL89(SQL1), SQL92 (SQL2), SQL99(SQL3).

A última versão definida pela ANSI/ISO traz características novas como: store procedures, triggers, suporte à programação OO, XML, entre outras (SQL2003).

Diferentes fornecedores de SGBDS apresentam versões de SQL com algumas particularidades próprias.

Características

- Estilo declarativo, não procedimental.
- Permissão de otimizações.
- Utilização por várias classes de usuários.
- Sintaxe simples e bem definida.
- Presença em todos os SGBDs Relacionais.
- Incorporação comumente a uma outra linguagem.
- Não constituição de uma linguagem de programação Delphi, C ou Java.
- Portabilidade entre sistemas operacionais.

Linguagem

- Disponibiliza uma série de comandos DDL para definição de dados.
- Disponibiliza um conjunto de comandos DML para manipulação de dados.
- Proporciona um conjunto de instruções para definição de visões (tabelas virtuais).
- Permite controle de autorização de acesso.
- Permite controle de transações e concorrência.
- Disponibiliza instruções para especificação de restrições de integridade.

SQL- DDL

Principais comandos da DDL:

- **CREATE** - Cria uma definição
- **ALTER** - Alterar alguma definição
- **DROP** - Exclui uma definição

Outros comandos da DDL:

- **TABLE** - Define uma tabela, seus campos e as restrições de campo.
- **INDEX** - Define um índice associado a um campo de uma tabela.
- **DOMAIN** - Define um tipo de dado.
- **GRANT** - Define usuários e autorizações para cada um.
- **EXCEPTION** - Define uma mensagem de erro.

- **TRIGGER** - Define um conjunto de instruções que são automaticamente executadas antes ou depois de um comando **INSERT**, **UPDADE** ou **DELETE**.

- **PROCEDURE** - Define um conjunto de instruções. Podem receber ou retornar valores. Podem ser executadas através de uma solicitação do usuário ou por um **TRIGGER**.

CREATE DATABASE

- **CREATE DATABASE** - Cria um novo banco de dados ou esquema

```
CREATE {DATABASE | SCHEMA}  <nome do banco>
```

CREATE TABLE

- **CREATE TABLE** - Cria uma nova tabela com seus campos e define as restrições de campo

```
CREATE TABLE <nome da tabela>(
        <coluna><tipo-do-dado> [not null] [not null with default],
        <coluna><tipo-do-dado> [not null] [not null with default], ...
        primary key (<coluna-pk>)
        foreign key (<coluna-fk>) references <tabela-pai> (<coluna-pk-pai>)
                )
```

Onde:

- **<tabela>** : Define o nome da tabela.
- **<coluna>** : Define o nome da coluna. A definição das colunas de uma tabela é feita relacionando-as uma após a outra, separadas por vírgula.
- **<tipo-do-dado>** : Define o tipo e o tamanho da coluna definida.
- **<not null>** : Define uma coluna de preenchimento obrigatório.
- **<not null with default>** : Define uma coluna de preenchimento obrigatório e que, no momento da inclusão de uma linha na tabela, tenha um valor pré-definido.
- **primary key (<coluna-pk>)** : Define a coluna que será a chave primária da tabela. No caso de mais de uma coluna ser chave primária, separá-las por vírgula.
- **foreign key (<coluna-fk>) references <tabela-pai> (<coluna-pk-pai>)** : Define a <coluna-fk>, de <tabela>, como chave estrangeira. A <tabela-pai>. Define a tabela-pai relacionada a fk e <colunapk-pai>. Define a coluna pk, de <tabela-pai>.

Exemplo:

```
create table curso
(
cod_curso INTEGER not null,
nome_curso VARCHAR(40) not null,
primary key (cod_curso)
);
create table turma
(
cod_turma INTEGER not null,
nome_turma VARCHAR(40) not null,
cod_curso INTEGER not null,
primary key (cod_turma),
foreign key (cod_curso) references curso (cod_curso)
);
```

- **CONSTRAINTS:** Constraints são regras agregadas a colunas ou tabelas. Assim, pode-se definir como obrigatório o preenchimento de uma coluna que tenha um valor-padrão quando uma linha for incluída na tabela ou quando aceitar apenas alguns valores pré-definidos. No caso de regras aplicadas a tabelas, tem-se a definição de chaves primárias e estrangeiras.

Exemplos:

```
create table aluno
(
matricula INTEGER not null,
nome_aluno VARCHAR(40) not null,
sexo CHAR(1) check (upper(sexo) = 'M' or upper(sexo) = 'F'),
cpf NUMERIC(11) unique,
cod_curso INTEGER not null,
primary key (matricula),
foreign key (cod_curso) references curso (cod_curso)
);
```

- **ALTER TABLE** - Altera definições na estrutura de uma tabela do BD, acrescentando, alterando e removendo nomes.
- Formatos das colunas e regras de integridade referencial.

```
ALTER TABLE <nome da tabela> (
[ drop <coluna> ] |
[ add <coluna><tipo-do-dado> [not null] [not null with default]] |
[ rename <coluna><novo-nome-coluna>] |
[ rename table <novo-nome-tabela>] |
[ add primary key ( <coluna-pk> ) ] |
[ drop primary key ( <coluna-pk> ) ] |
[ add foreign key (<coluna-fk>) references <tabela-pai> (<coluna-pk-pai>)
[ drop foreign key (<coluna-fk>) references <tabela-pai> ]
                    )
```

Onde:

- **<tabela>** : Define o nome da tabela.
- **drop <coluna>** : Remove a coluna da estrutura da tabela.
- **add <coluna><tipo-do-dado> [not null] [not null with default]** : Acrescenta uma nova coluna. No caso da existência de linhas nesta tabela, o valor da nova coluna será de acordo com as definições das cláusulas not null e not null with default.
- **rename <coluna><novo-nome-coluna>** : Troca o nome da coluna.
- **rename table <tabela>** : Troca o nome da tabela.

Onde:

- **add primary key (<coluna-pk>)** : Define uma chave primária à uma nova coluna acrescentada na tabela.
- **add foreign key (<coluna-fk>) references <tabela-pai> (<coluna-pk-pai>)** : Acrescenta uma nova chave estrangeira.
- **drop foreign key (<coluna-fk>) references <tabela-pai>** : Remove a definição de chave estrangeira.
- **drop primary key (<coluna-pk>)** : Remove a definição de chave primária da coluna.

Exemplo:

```
create table aluno
(
matricula INTEGER not null,
```

```
nome_aluno VARCHAR(40) not null,
cod_curso INTEGER not null,
primary key (matricula)
);
alter table aluno
add foreign key (cod_curso) references curso (cod_curso);
);
```

- **CREATE INDEX** - Cria uma estrutura de índice de acesso para uma determinada coluna de uma tabela. Um índice permite um acesso mais rápido aos dados e pode ser criado a partir de uma ou mais colunas da tabela. Toda chave primária possui um índice definido.

```
CREATE [UNIQUE] INDEX <nome do indice> on <tabela>
( <coluna> [ASC] [DESC], <coluna> [ASC] [DESC]
 )
```

Onde:

- **<índice>** : Define o nome do índice. A cláusula unique define que não existirão duas linhas com o mesmo conteúdo de <coluna>. A definição do índice de uma chave primária contém unique.
- **<tabela>** : Define o nome da tabela que contém a coluna que terá o índice
- **<coluna>** : Define a coluna da tabela. As opções ASC/DESC representam, respectivamente, uma ordenação ascendente e descendente.

- DROP - Remove um objeto do banco de dados.

```
DROP <OBJETO><nome do objeto>;
```

Exemplo:

- REMOVE TABELA

```
drop table curso;
```

- REMOVE VIEW

```
drop view v_curso;
```

- REMOVE ÍNDICE

```
drop index curso_pk;
```

Linguagem interativa de manipulação de dados (DML).

Algebra Relacional

Linguagem de consulta que consiste em um conjunto de operações, tendo como entrada uma ou duas relações e produz como resultado uma nova relação.

Composta por um conjunto básico de operações que permitem ao usuário especificar novas relações em um banco de dados.

É uma linguagem de consulta formal, porém procedural, que possibilita ao usuário realizar uma sequência de operações na base de dados para calcular o resultado desejado, implementa e otimiza consultas em SGBD's.

Os conceitos da Algebra Relacional são utilizados na linguagem de consulta padrão SQL.

Características

- Maneira teórica de se manipular um BD relacional;
- Linguagem de consulta procedural:
  - usuários especificam os dados necessários e como obtê-los;
- Consiste de um conjunto de operações:
  - entrada: uma ou duas relações;
  - saída: uma nova relação resultado

Expressão da Álgebra Relacional

Sequência de operações:

- Operadores de Comparação: =, <, <=, >, >=, ≠
- Operadores lógicos: ^ (and), v (or), ¬ (not)

Teoria dos Conjuntos

Utiliza operações usuais da Teoria dos Conjuntos, na Álgebra Relacional cada relação é considerada um conjunto de tuplas. Levam em consideração apenas a estrutura da relação e não a semântica, nas operações binárias sobre conjuntos a maioria exige Compatibilidade de Domínio nas relações.

- **Relação** - Uma relação é um conjunto de tuplas, todos os elementos de um conjunto são distintos. Nenhuma tupla pode ter a mesma combinação de valores para todos os seus atributos.

Operações

- Seleção (σ): Indicada por σ (a letra grega sigma), é uma operação que para um conjunto inicial fornecido como argumento, produz um subconjunto estruturalmente idêntico, mas apenas com os elementos do conjunto original que atendem a uma determinada condição. A seleção pode ser entendida como uma operação que filtra as linhas de uma tabela, e é também uma operação unária, já que opera sobre um único conjunto de dados.
- Seleção - Produz um subconjunto das linhas da tabela, sendo dada uma condição a ser verificada.
- Ex.: Selecionar os produtos com preço < 300,00

Tabela de Produtos

Cod_Prod	Valor	Descrição	Fornecedor
001	100,00	Cadeira	5001
002	300,00	Mesa	5001
003	150,00	Cadeira	5033

Resultado

Cod_Prod	Valor	Descrição	Fornecedor
001	100,00	Cadeira	5001
003	150,00	Cadeira	5033

- Projeção (π): Geralmente indicada na literatura por π (a letra grega pi) produz um conjunto onde há um elemento para cada elemento do conjunto de entrada, sendo que a estrutura dos membros do conjunto resultante é definida nos argumentos da operação. Pode ser entendida como uma operação que filtra as colunas de uma tabela. Por operar sobre apenas um conjunto de entrada, a projeção é classificada como uma operação unária.
- Projeção - Produz um subconjunto das colunas da tabela, sendo dados os atributos desejados no resultado.
- Ex.: Projetar a tabela Produtos para Produto e Valor

Tabela de Produtos

Cod_Prod	Valor	Descrição	Fornecedor
001	100,00	Cadeira	5001
002	300,00	Mesa	5001
003	150,00	Cadeira	5033

Resultado

Cod_Prod	Valor
001	100,00
002	300,00
003	150,00

- Junção: É uma operação que produz uma combinação entre as linhas de uma tabela com as linhas correspondentes de outra tabela, sendo em princípio correspondente a uma seleção pelos atributos de relacionamento sobre um produto cartesiano dessas tabelas.
- Junção - Produz uma tabela composta de duas outras que se relacionam.
- Ex.: Juntar a tabela Produtos e Fornecedor através de Fornecedor e Cod_Fornec

Tabela de Produtos

Cod_Prod	Valor	Descrição	Fornecedor
001	100,00	Cadeira	5001
002	300,00	Mesa	5001
003	150,00	Cadeira	5033

Tabela de Fornecedor

Cod_Fornec	CGC	Razao_Social
5001	45621336546	Moveis 10
5002	78515235533	Jair Móveis SA
5033	12365466315	Planejado LTDA

Resultado

Cod_Prod	Valor	Descrição	Fornecedor	CGC	Razao_Social
001	100,00	Cadeira	5001	45621336546	Moveis 10
002	300,00	Mesa	5001	45621336546	Moveis 10
003	150,00	Cadeira	5033	12365466315	Planejado LTDA

SQL – DML

Linguagem de manipulação de dados (ou DML, de Data Manipulation Language) é uma linguagem para a recuperação, inclusão, remoção e modificação de informações em um banco de dados.

Linguagem de manipulação de dados (ou DML, de Data Manipulation Language. Pode ser procedural, que

especifica como os dados devem ser obtidos do banco; pode também ser declarativa (não procedural), em que os usuários não necessitam especificar o caminho de acesso, isto é, como os dados serão obtidos.

- Comandos:
  - Insert
  - Select
  - Update
  - Delete

- **INSERT** - usado para inserir um registro (formalmente uma tupla) a uma tabela existente.

INSERT INTO tabela (coluna1, [coluna2, ...]) VALUES (valor1, [valor2, ...])

- O número de colunas e valores devem ser o mesmo. Se uma coluna não for especificada, o valor padrão é usado se estiver definido previamente.
- Os valores especificados (ou incluídos) pela declaração INSERT devem satisfazer todas as restrições aplicáveis.
- Se ocorrer um erro de sintaxe ou se algumas das restrições forem violadas, a nova linha não é adicionada à tabela e um erro é retornado.

- Exemplos

```
INSERT INTO Pessoa (id, nome, sexo) VALUES
  (1, 'João da Silva', 'M');

INSERT INTO agenda (nome, numero) VALUES
  ('Ana Maria', '333-31562');
```

- **SELECT** - declaração SQL que retorna uma lista com os resultados de registros de uma ou mais tabelas.

SELECT* FROM<nome_tabela>* Retorna todos os campos da tabela

Select<campos da tabela>
From<nome da tabela>
Where<condição>

DML – SELECT

Este comando recupera zero ou mais linhas de uma ou mais tabelas-base, tabelas temporárias ou visões em um banco de dados.

- SELECT é o comando de Linguagem de Manipulação de Dados(DML) mais utilizado.
- Consultas SELECT especificam um conjunto de resultados, mas não especificam como calculá-los. Esta função é executada pelo sistema de banco de dados, mais especificamente pelo otimizador de consulta.

- Exemplos

```
SELECT * FROM pessoa;
```

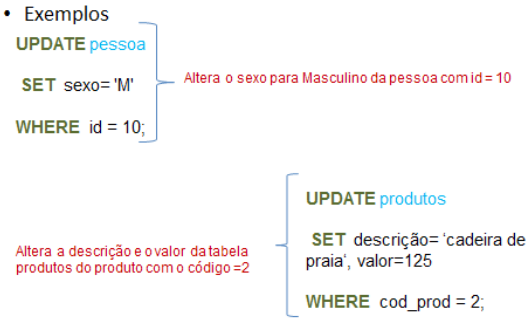
Lista todos os registros da tabela pessoa.

```
SELECT id, nome
FROM pessoa
WHERE sexo = 'M' ;
```

Lista os campos id e nome das pessoas do sexo masculino.

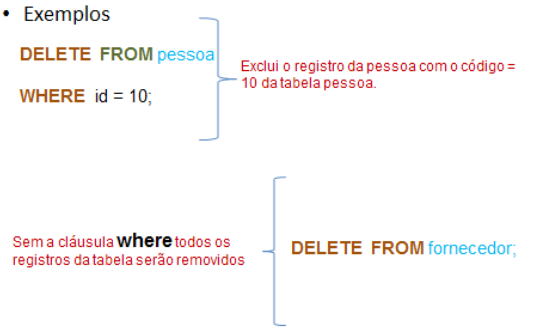
- **UPDATE** - altera os dados de um ou mais registros em uma tabela. Todas as linhas podem ser atualizadas, ou um subconjunto pode ser escolhido através de uma condição.

UPDATE <nome da tabela>
SET <coluna1 = valor1, coluna2 = valor2>
WHERE <condição>



- DELETE - permite remover linhas existentes de uma tabela.

DELETE FROM <nome da tabela>
WHERE <condição>



DML – Distinct

A cláusula DISTINCT elimina os dados duplicados em um Select.

Exemplo:

SELECT DISTINCT * FROM CLIENTE;

DML - Funções Agregadas

- **COUNT(*)** : Obtém o número total de registros, incluindo nulos
  SELECT COUNT(*) FROM <nome da tabela>
- **COUNT(<campo>)** : Obtém o número de registros, excluindo os nulos
  SELECT COUNT(coluna1) FROM <nome da tabela>
- **AVG(<campo>)** : Obtém o valor médio (média dos valores) do campo especificado
  SELECT AVG(coluna1) FROM <nome da tabela>
- **MIN(<campo>)** : Obtém o valor mínimo (menor valor) do campo especificado
  SELECT MIN(coluna1) FROM <nome da tabela>
- **MAX(<campo>)**: Obtém o valor máximo (maior valor) do campo especificado.
  SELECT MAX (coluna1) FROM <nome da tabela>
- **SUM(<campo>)** : Obtém a soma dos valores para o campo especificado.
  SELECT SUM(coluna1) FROM <nome da tabela>

Visões/views

Tabela lógica de um banco de Dados, não contém dados.

As Visões são utilizadas quando não é desejável que todos os usuários tenham acesso a todo o esquema conceitual.

- Visões precisam ser definidas

Uma Visão é uma relação virtual que não faz parte do esquema conceitual, mas que é visível a um grupo de usuários, uma visão é uma tabela virtual que é definida a partir de outras tabelas, contendo sempre os dados atualizados.

A visão é definida por uma DDL e é computada cada vez que são realizadas consultas aos dados daquela visão.

O catálogo do SGBD é o repositório que armazena as definições das visões, uma visão possui nome, uma lista de atributos e uma query que computa a visão

- View (Visão) : logicamente representam subconjuntos de uma ou mais tabelas e se comportam como tabelas quanto a inserção, alteração, exclusão e consulta de dados.

É possível inserir, alterar, excluir ou consultar dados em uma view, respeitando sempre as regras de integridade e de segurança que estiverem definidas sobre as Tabelas envolvidas na constituição da view.

Uma visão é uma relação virtual, não faz parte do conjunto de relações reais de um Banco de Dados, mas que é visível para os usuários como tal.

Forma Geral

CREATE VIEW <Nome da View> AS <Expressão de Consulta>

onde <Expressão de Consulta>em geral corresponde a uma

consulta, em geral SELECT... FROM .... WHERE.

Chamamos de tabelas fonte àquelas que são consultadas para a criação da visão.

Sintaxe

Criando uma View

- Sintaxe:

```
create view <nome da view>
                as<expressão de consulta>
```

Uma vez que uma view é definida, o nome da view pode ser usado para se referir à relação virtual gerada por ela.

Uma view não é o mesmo que criar uma nova relação avaliando a expressão de consulta. Em vez disso, uma definição de view causa o salvamento de uma expressão. A expressão é substituída nas consultas usando a view.

Consultando uma View

- Sintaxe:

```
Select * from <nome da view>
```

Excluindo uma View

- Sintaxe:

```
drop view <nome da view>
```

Quando uma view deixar de ser útil, assim como uma tabela, ela pode ser excluída.

- Visões - Exemplo:

```
Create view V_Vendas
        (Cod_Venda, Vlr_Total,
        Cod_Item, Cod_produto, Dcr_Produto,
        Vlr_Produto, Qtde, Vlr_Total_Item)
 As select v.cod_venda, v.vlr_total,
        iv.cod_item, p.cod_produto,        p.dcr_produto,
        iv.vlr_produto, iv.qtde,
        iv.vlr_produto * iv.qtde
 from vnd_vendas as v, vnd_itens_vendas as iv,
        cad_produtos as p
 where v.cod_venda = iv.cod_venda
        and iv.cod_produto = p.cod_produto
```

Consulta

Uma vez criada, basta utilizar uma consulta simples (select) que emprega a view.

```
select * from V_Vendas
```

Exclusão

Uma visão pode ser removida da Base de Dados.

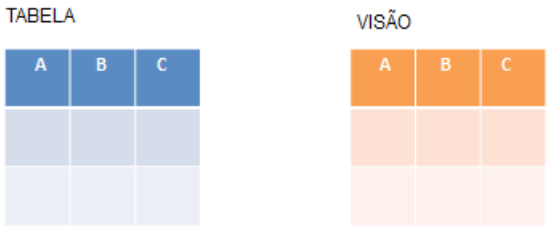
```
drop view V_Vendas
```

Finalidades

Permitem adaptar a aparência da base de dados considerando diferentes usuários, restringem o acesso aos dados e simplificam o acesso aos dados.

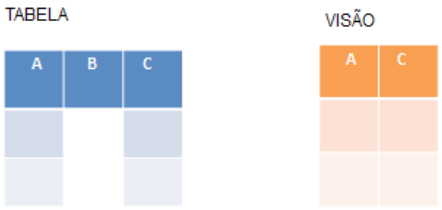
Visão Idêntica

Cria uma visão idêntica à tabela de origem.



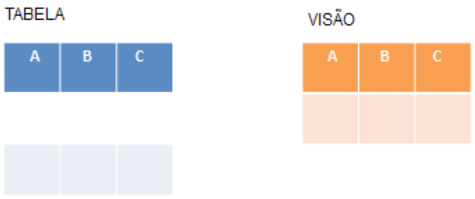
Visão por Seleção de Colunas

Cria uma visão selecionando colunas da tabela de origem.



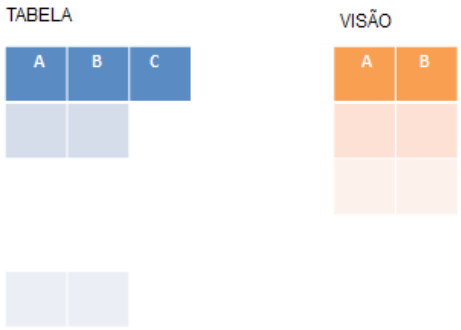
Visão por Seleção de Linhas

Cria uma visão selecionando linhas da tabela de origem.



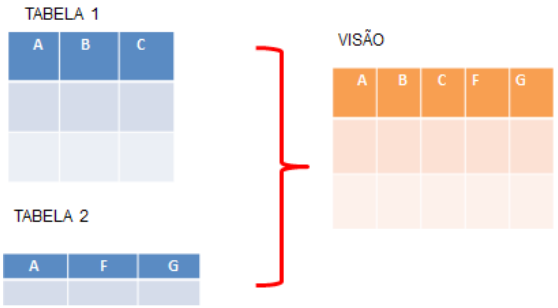
Visão por Seleção de Linhas e Colunas

Cria uma visão com a seleção de linhas e colunas da tabela de origem.



Visão por junção de Tabelas

Cria uma visão através da contatenação de linhas e colunas de várias tabelas.



Propriedades

- As atualizações nas tabelas refletem nas views.
- Uma view pode ser atualizável.
- Se a tabela básica é destruída todas views são destruídas.

Autorização/Segurança

Em ambientes multiusuários, é importante proteger o banco de dados de alterações indevidas nos dados ou nas estruturas das tabelas, as quais podem comprometer a integridade do banco de dados.

Segurança de banco de dados relaciona-se com o controle de uso dos objetos do banco de dados e as ações que esses usuários podem realizar sobre os objetos.

- Quando vários usuários têm acesso à base de dados, em geral eles têm privilégios diferentes quanto à manipulação dos dados.
- O SGBD deve garantir que usuários autorizados realizam operações corretas na base de dados.
  - identificação de usuários e de dados;
  - autenticação de usuários;
  - manutenção da matriz de autorização.
- A Linguagem de Controle de Dados - DCL (Data Control Language) controla os aspectos de autorização de dados e licenças de usuários para controlar quem tem acesso para ver ou manipular dados dentro do banco de dados.

DCL – Comandos

- GRANT- autoriza o usuário executar ou setar operações.
- REVOKE- remove ou restringe a capacidade de um usuário de executar operações.

outros comandos DCL:

- CREATE USER
- ALTER PASSWORD
- CREATE SYNONYM

Integridade

A integridade de um banco de dados é atingida com base em:

- Regras de Integridade estabelecidas na modelagem de dados;
- Boa implementação das regras de negócio da aplicação.

As regras de integridade são restrições sobre os dados que permitem controlar a forma como os valores são inseridos e mantidos no banco de dados. Toda vez que um dado é manipulado pelo SGBD, as restrições de integridade são avaliadas para verificar a consistência do dado afetado.

As restrições de integridade protegem contra danos acidentais no banco de dados, garantindo que as mudanças feitas no banco de dados por usuários autorizados não resultem em uma perda da consistência dos dados.

As restrições de integridade servem de apoio para a qualidade da informação, uma vez que elas controlam como os dados são inseridos e mantidos. Isso agrega maior confiabilidade aos dados, com consequente aumento da qualidade da informação.

Essa qualidade é útil em diversos cenários, tais como:

- Relatórios;
- Carga de dados;
- Integração com outros sistemas.

Alguns exemplos de restrições de integridade:

- Chave Primária;
- Chave Estrangeira;
- Not Null;
- Check(P), onde P é um Predicado.

Tipos

- **Integridade de Colunas:** Refere-se aos valores que uma determinada coluna pode assumir.
- Exemplo:
  - Restrições de Check (Constraint);
  - Nulidade de Coluna (ex.: create table pessoa (... idade int NOT NULL));
- **Integridade de Linhas**: Refere-se às restrições que devem ser obedecidas entre linhas.
- Exemplo:
  - Chave Primária.
- **Integridade Referencial :** Refere-se a valores que devem ser validados conforme valores em outra tabela, isto é, valores que se referem a outros valores.
- Exemplo:
  - Chave Estrangeira.

Controle de Tansações

Algumas operações devem ser realizadas como se fossem uma só.

- Exemplo: transferência de valores entre contas bancárias (debitar de uma conta e creditar em outra).

As Transações devem ser realizadas na totalidade quando obtiverem sucesso ou não ser realizadas no caso em que ocorra alguma falha.

Uma TRANSAÇÃO é uma unidade lógica de trabalho que NÃO deve ser subdividida.

Propriedades ACID

- Atomicidade – a transação deve ser executada como uma unidade.
- Consistência – a transação deve deixar o banco de dados em um estado consistente.
- Isolamento – as transações devem se comportar como se tivessem acesso exclusivo ao banco de dados.
- Durabilidade – as modificações feitas por transações devem ser duráveis no banco (devem persistir).

Instruções

Transação é controlada por três instruções

- BEGIN { WORK | TRANSACTION }; declara o início de uma transação.
- COMMIT [WORK | TRANSACTION]; indica que a transação teve sucesso.
- ROLLBACK[WORK | TRANSACTION]; indica que a transação será abortada, cancelando todas as alterações feitas, devendo voltar ao estado anterior à transação.
- Enquanto uma transação não é comitada (COMMIT), o banco de dados não faz as alterações das instruções da transação.
- Uma transação bloqueia os objetos que ela usa, não permitindo que outros usuários possam acessá-los, transações devem ser do menor tamanho possível.
- **Não mantenha transações abertas desnecessariamente no Banco de Dados.**

Gerenciadores de banco de dados SGBD's

Unidade H
Projeto de Banco de Dados Relacional

UNIDADE H

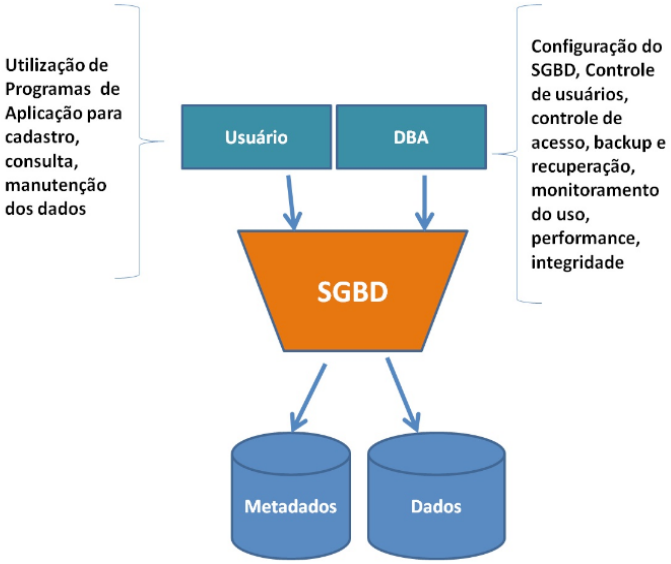
GERENCIADORES DE BANCO DE DADOS – SGBD´S

Conceito

Um SGBD é uma coleção de dados inter-relacionados mais um conjunto de programas para acessar e manipular esses dados (Silberschatz ,2006).

É um Sistema cujo objetivo principal é gerenciar o acesso e a correta manutenção dos dados armazenados em um banco de dados.

SGBD é um conjunto de programas responsáveis pelo gerenciamento do banco de dados.



Funções

Um SGBD auxilia no projeto, construção e povoamento de um Banco de Dados. O SBBD deve garantir:

- Métodos de acesso
- Integridade Semântica
- Segurança
- Concorrência
- Independência

Métodos de acesso

Possibilitam a criação e a manipulação de um Banco de Dados.

- DDL (Data Definition Language): especificação do esquema do BD (dados e seus tipos de dados, índices, ...)
- DML (Data Manipulation Language) manipulação de dados (I, A, E, C): processamento eficaz de consultas considera relacionamentos, predicados de seleção, volume de dados, índices, ...

Integridade Semântica

Garantia de dados sempre corretos com relação ao domínio de aplicação. Exemplos:

- estados válidos para os dados (I sexo; A salário);
- relacionamentos válidos entre os dados

Segurança

Evitar violação de consistência dos dados e garantir a segurança de acesso (usuários e aplicações). Exemplos:

- matrizes de autorização
- visões

segurança contra falhas (recovery)

Exemplos:

- **Monitoração de transações**
  - Transação, conjunto de operações a serem realizadas no BD
- **Categorias de falhas**
  - transação, sistema e meio de armazenamento
- **Manutenção de histórico de atualizações (logs) e backups do BD**

Concorrência

Evita conflitos de acesso simultâneo a dados por transações (scheduler).

Principais técnicas:

- Bloqueio (lock)

Independência

Garante a transparência da organização dos dados.

níveis de independência

- **Independência física**
  - transparência de organização (esquema) física dos dados
  - Exemplos: organização dos arquivos, indexação, distribuição, agrupamento.
- **Independência lógica**
  - transparência do esquema lógico do BD
  - Exemplo: visões (vários esquemas externos)

Objetivo

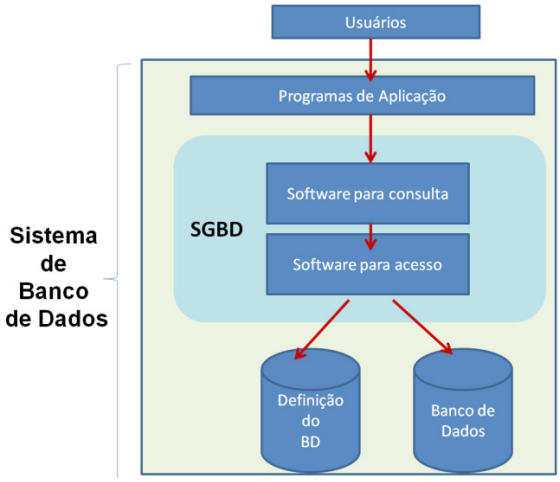
Fornecer um ambiente que seja tanto conveniente como eficiente para recuperação e armazenamento de informações.

Gerência dos Dados

- Definir estruturas de armazenamento.
- Fornecer mecanismos para a manipulação de informações.

O SGBD precisa garantir a segurança, apesar de falhas de sistema ou tentativas de acesso não autorizado.

Arquitetura



- Interface de alto nível de abstração que provê:
  - Consultas;
  - Manipulação de dados;
  - Definição de dados;
  - Geradores de relatórios.
- Tradutor/otimizador de consultas.
- Visões do usuário de BD.
- Controle de concorrência (sincronização de acessos simultâneos ao BD).
- Controle de integridade (validação de restrições de integridade).
- Controle de segurança (autorização de acesso aos dados).
- Controle de recuperação (torna o SGBD tolerante a falhas).
- Eficiente sistema de arquivos com técnicas indexação eficientes que permitem armazenar e manipular os dados do BD.

Ações

Um SBBD possibilita:

- Controle de redundância;
- Compartilhamento de dados;
- Independência de dados;
- Segurança;
- Backup e recuperação a falhas;
- Força a restrições de integridade;
- Aumento da produtividade e disponibilidade;
- Flexibilidade, padronização.

Importância

"Embora as interfaces de usuário ocultem detalhes de acesso a um banco de dados, e a maioria das pessoas nem mesmo tenha consciência de estar lidando com um banco de dados, acessar banco de dados é uma parte essencial da vida de quase todo mundo hoje"(Silberschatz ,2006).

Regras de Codd

Em 1985, Edgard F. Codd estabeleceu Doze Regras para determinar o quanto um banco de dados é Relacional ou não.

1. **Regra das informações em tabelas:**
   As informações a serem apresentadas no banco de dados devem ser apresentadas como relações (tabelas formadas por linhas e colunas) e o vínculo de dados entre as tabelas deve ser estabelecido por meio de valores de campos comuns.
   Isso se aplica tanto aos dados quanto aos metadados (descrições dos objetos do banco de dados).

2. **Regra de acesso garantido:**
   Para que o usuário possa acessar às informações contidas no banco de dados, o método de referência deve ser o nome da tabela, o valor da chave primária e o nome do campo.
   A ordem de apresentação dos dados não tem importância no contexto.

3. **Regra de tratamento sistemático de valores nulos:**
   Os valores nulos (diferente do zero, da string vazia, da string de caracteres em brancos e outros valores não nulos) existem para representar dados não existentes de forma sistemática e independente do tipo de dado.

4. **Regra do catálogo relacional ativo:**
   Toda a estrutura do banco de dados (tabelas, campos, índices, etc.) deve estar disponível em tabelas (catálogo). Essas tabelas são manipuladas pelo próprio sistema, quando o usuário efetua alterações na estrutura do banco de dados.

5. **Regra da atualização de alto nível:**
   O usuário deve ter capacidade de manipular as informações do banco de dados em grupos de registros, ou seja, ser capaz de inserir, alterar e excluir vários registros ao mesmo tempo.

6. **Regra da sublinguagem de dados abrangente:**
   Um sistema relacional pode suportar várias linguagens e formas de uso, porém deve possuir ao menos uma linguagem com sintaxe bem definida e expressa por cadeia de caracteres e com habilidade de apoiar a definição de dados, a definição de visões, a manipulação de dados, as restrições de integridade, a autorização e a fronteira de transações.

7. **Regra da independência física:**
   Quando for necessária alguma modificação na forma como os dados são armazenados fisicamente, nenhuma alteração deve ser necessária nas aplicações que fazem uso do banco de dados.
   Devem também permanecer inalterados os mecanismos de consulta e manipulação de dados utilizados pelos usuários finais.

8. **Regra da independência lógica:**
   Qualquer alteração efetuada na estrutura do banco de dados, como inclusão e exclusão de campos de uma tabela ou alteração no relacionamento entre tabelas não deve afetar o aplicativo que o usa. O aplicativo deve manipular visões das tabelas. Visões são uma espécie de tabela virtual, que agrupam dados de uma ou mais tabelas físicas e apresentam ao usuário os dados.

9. **Regra da atualização de visões:**
   Toda visão que for teoricamente atualizável será também atualizável pelo sistema.

10. **Regra da independência de integridade:**
    As várias formas de integridade do banco de dados (integridade de entidade, referencial, restrição e obrigatoriedade de valores, etc.) precisam ser estabelecidas, dentro do catálogo do sistema ou dicionário de dados, e ser totalmente independente da lógica dos aplicativos.

11. **Regra da independência de distribuição:**

Sistemas de banco de dados podem estar distribuídos em diversas plataformas, interligados em rede e podem inclusive estar fisicamente distantes entre si. Essa capacidade de distribuição não pode afetar a funcionalidade do sistema e dos aplicativos que fazem uso do banco de dados.

12. Regra não subversiva:

O sistema deve ser capaz de impedir que qualquer usuário ou programador de passar por cima de todos os mecanismos de segurança, regras de integridade do banco de dados e restrições, utilizando algum recurso ou linguagem de baixo nível que eventualmente possam ser oferecidas pelo próprio sistema.

Exemplos de SGBD

- IBM Informix;
- PostgreSQL;
- Firebird;
- HSQLDB;
- IBM DB2;
- MySQL;
- Oracle;
- SQL-Server;
- TinySQL;
- JADE;
- ZODB;
- Microsoft Visual Foxpro.

Atividades

Com base no material anterior, responda às questões a seguir.

1. Qual das alternativas abaixo é o conceito de um SGBD?

a. Sistema que permite criação de visões de dados.

b. Sistema cujo objetivo principal é gerenciar o acesso e a correta manutenção dos dados armazenados em um banco de dados.

c. Sistema cujo objetivo principal é o acesso e a consulta dos dados armazenados em um banco de dados.

d. Sistema que possibilita consulta, exclusão e inclusão dos dados em uma base de dados.

2. São vantagens de um SGBD:

a. DDL, DML, DCL

b. Linguagem de manipulação, softwares aplicativos

c. Aplicação, Controle, Remoção

d. Controle de redundância, controle de integridade, mecanismos de backup

3. São características de um SGBD:

a. Replicação, acesso, controle de concorrência, recuperação de rapada e falhas

b. Consultar, Incluir, Deletar

c. Segurança, integridade, controle de concorrência, recuperação de rapada e falhas

d. Entrada de dados, manipulação de dados, recuperação

4. São exemplos de sistemas gerenciadores de banco de dados:

a. PostresSql, Tibia, FIFA, Mysql, IBM DB2

b. Oracle, Myspace, Mysql, SQL Server

c. PostresSql, Oracle, SQL-Server, Mysql, IBM DB2

d. Mysql, LinkSQL, Notesql

5. Criador das doze regras de um banco de dados relacional:

a. Peter Chen

b. Edgard F. Codd

c. Carlos Alberto Heuser

d. Silberschatz

6. O que você entende por “Dicionário de Dados”?

a. Lista de elementos que fazem parte de um banco de dados.

b. Informações sobre as definições de elementos de dados e respectivas características – descreve os dados, quem os acessa, etc.

c. Relação das entidades de uma base de dados.

d. Listagem dos atributos pertencentes a uma entidade.

7. São aplicações de Banco de Dados:

a. Softwares, correio eletrônico, sistemas de informação

b. Softwares, hardware, periféricos

c. Dicionário de Dados, tabelas, registros

d. Esquema, entidades, instância

Transações em banco de dados

Unidade I

Projeto de Banco de Dados Relacional

UNIDADE

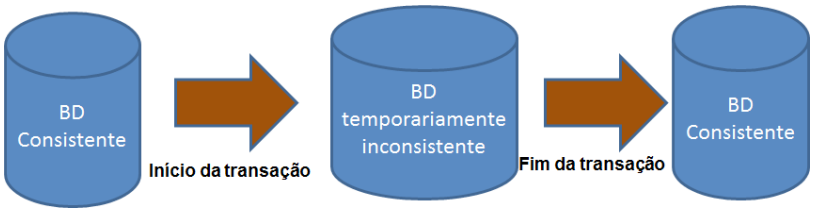
TRANSAÇÕES EM BANCO DE DADOS

Definições

Unidade lógica de processamento em um SGBD, composta de uma ou mais operações. De forma abstrata e simplificada, uma transação pode ser encarada como um conjunto de operações de leitura e escrita de dados.

Uma transação é uma unidade da execução de programa que acessa e possivelmente atualiza vários itens de dados. Uma transação precisa ver um banco de dados consistente.

Durante a execução da transação, o banco de dados pode ser temporariamente inconsistente. Quando a transação é completada com sucesso (é confirmada), o banco de dados precisa ser consistente.



Problemas

Durante a execução da transação alguns problemas podem acontecer, é necessário tratar estas situações com o objetivo de manter o BD consistente.

Dois problemas principais para resolver:

- Falhas de vários tipos, como falhas de hardware e quedas de sistema;
- Execução concorrente de múltiplas transações.

Propriedades

Existem requisitos que sempre devem ser atendidos por uma transação.

Chamadas de propriedades ACID:

- Atomicidade
- Consistência
- Isolamento
- Durabilidade ou Persistência

Atomicidade

- **Princípio do "Tudo ou Nada"**
  - ou todas as operações da transação são efetivadas com sucesso no BD ou nenhuma delas se efetiva.
- **Deve preservar a integridade do BD.**
- **Responsabilidade do subsistema de recuperação contra falhas (subsistema derecovery) do SGBD.**
  - desfazer as ações de transações parcialmente executadas.

Consistência

- Uma transação sempre deve manter o BD consistente.
- Desfazer as ações da transação que violaram a integridade da base de dados

Isolamento

- **No contexto de um conjunto de transações concorrentes, a execução de uma transaçãoT1 deve funcionar como se T1 executasse de forma isolada.**
  - T1 não deve sofrer interferências de outras transações executadas em forma de corrência.
- **Responsabilidade do subsistema de controle de concorrência (scheduler) do SGBD garantir escalonamentos sem interferências.**

Durabilidade

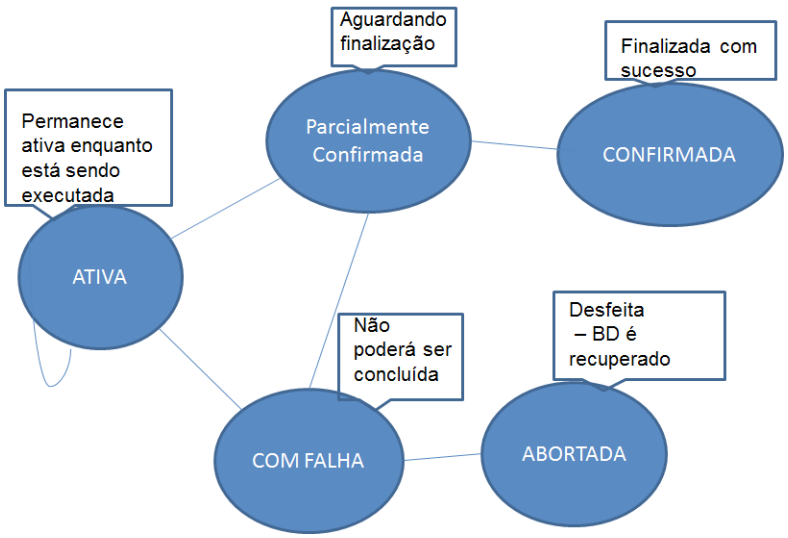
- **Deve-se garantir que as modificações realizadas por uma transação que concluiu com sucesso persistam no BD.**
  - nenhuma falha posterior ocorrida no BD deve perder essas modificações.
- **Responsabilidade do subsistema de recovery.**
  - refazer transações que executaram com sucesso em caso de falha no BD

Estados

O estado de uma transação é monitorado constantemente pelo SGBD. Tal estado representa a situação na qual se encontra a transação em um momento específico.

Estados possíveis de uma transação:

- **Ativa** – O estado inicial; a transação permanece nesse estado enquanto está executando.
- **Parcialmente confirmada** – Depois que a instrução final foi executada.
- **Com Falha** – Depois da descoberta de que a execução normal não pode mais prosseguir.
- **Abortada** – Depois que a transação foi revertida e o banco de dados foi restaurado ao seu estado anterior ao início da transação.
  - **Duas opções após ter sido abortada:**
  - Reiniciar a transação; pode ser feito apenas se não houver qualquer erro lógico interno.
  - - Excluir a transação – erro lógico interno.
- **Confirmada (Commited)** – Após o término bem sucedido.



Execuções simultâneas (Concorrentes)

Várias transações podem ser executadas simultaneamente no sistema.

As vantagens são:

- Melhor utilização do processador e do disco, levando a um melhor throughput de transação: uma transação pode estar usando a CPU, enquanto outra está lendo ou escrevendo no disco.
- Tempo de médio de resposta reduzido para transações: as transações curtas não precisam esperar atrás das longas.
- Esquemas de controle de concorrência – mecanismos para obter isolamento, ou seja, para controlar a interação entre as transações concorrentes, a fim de evitar que elas destruam a consistência do banco de dados.
- Throughput: número de transações executadas em determinada quantidade de tempo.

Escalonamento

Sequências de instruções que especificam a ordem cronológica em que as instruções das transações concorrentes são executadas.

- Um escalonamento para um conjunto de transações precisa consistir em todas as instruções dessas transações.
- Precisam preservar a ordem em que as instruções aparecem em cada transação individual.
- Uma transação executada com sucesso terá uma instrução *commit* como a última instrução.
- Uma transação não finalizada com sucesso terá a instrução *abort* como a última instrução.

Serialização

Cada transação preserva a consistência do banco de dados.

Portanto, a execução serial de um conjunto de transações preserva a consistência do banco de dados.

Implementação

Um banco de dados precisa fornecer um mecanismo que garanta que todos os escalonamentos possíveis sejam seriais e recuperáveis.

Transações explícitas

Transações explícitas seguem o conceito formal de transações, onde devemos indicar o seu início pelo comando *BEGIN TRANSACTION* e o seu término, através dos comandos *COMMIT* ou *ROLLBACK*.

Transações automáticas

Modo padrão de muitos SBGD's. Cada instrução Transact-SQL individual é confirmada na conclusão. Você não precisa especificar nenhuma instrução para controlar transações.

Se uma instrução for concluída com sucesso, será confirmada, se encontrar qualquer erro, será revertida.

Transações implícitas

Transações implícitas são as que ocorrem automaticamente quando enviamos os comandos *INSERT, UPDATE* e *DELETE* para o BD. Essas transações são chamadas de implícitas, pois não precisamos indicar o início através de um comando *BEGIN TRANSACTIO*N ou um térmico através do comando *COMMIT* ou *ROLLBACK*.

Comandos: Insert, Update, Delete, Create, Drop, Alter, entre outros.

Finalizando Transações

- *COMMIT* – A efetivação da transação garante que a transação seja realizada com sucesso.
- *ROLLBACK* – Falhas ou cancelamento desfazem todas as modificações feitas na transação, retornando os dados ao estado anterior, ao início da transação.
- CONCLUÍDA: É o estado final de toda e qualquer transação. Nesse estado, a transação deixa de ser executada e as informações armazenadas temporariamente são descartadas.

Classificação das Falhas

- Falha de transação: erro lógico (overflow) ou erro de sistema (deadlock).
- Queda do sistema: erros de hardware e bugs de software (não perde dados).
- Falha de disco (recuperação por backup).
- Algoritmos para recuperação.
- Ações executadas durante a operação normal do sistema;
- Ações executadas pós-falha.
- Silberschatz (1999).

Síntese

Transação

Vários processos existentes no mundo real devem ser mapeados e implementados em sistemas de informação, tais sistemas devem refletir as ações do mundo real em um sistema off-line ou on-line. Atualmente podemos efetuar compra e venda pela internet, ter acesso a vários tipos de prestação de serviço através de um telefone ou dispositivo móvel. Todas essas vantagens do mundo moderno são possíveis pelo uso de um Banco de Dados. Cada ação executada em um banco de dados tem o nome de transação, muitos são os conceitos encontrados, mas a maioria deles nos remete a um mesmo entendimento.

Para DATE, "Transação é uma unidade lógica de trabalho, envolvendo diversas operações de bancos dados."

> *"...uma transação é uma sequência de operações num sistema de gerência de banco de dados que são tratadas como um bloco único e indivisível (atômico) durante uma recuperação de falhas e também prover isolamento entre acessos concorrentes na mesma massa de dados" (Wikipedia).*

Podemos considerar que uma transação é qualquer operação de escrita em uma tabela em um banco de dados. Quando executamos uma instrução UPDATE estamos executando uma transação, da mesma forma quando executamos um DELETE ou um INSERT. Esse trio de comandos efetivamente altera o conteúdo dos dados dentro de uma tabela.

O controle de transações nos possibilita definir regras de negócio para o funcionamento das Bases de Dados. Na verdade, devemos controlar múltiplas transações, torná-las consistentes e dependentes entre si, mas devemos fazer isso de forma segura tendo como objetivo primeiro a preservação dos dados e a integridade Base de Dados.

Propriedades ACID

As propriedades ACID são atributos que toda transação precisa ter para que não existam problemas durante a execução. ACID é uma sigla que significa Atomicidade, Consistência, Isolamento e Durabilidade. A seguir, apresentaremos o conceito de cada um deles:

- **Atomicidade:** Uma transação é uma unidade indivisível de alteração da base de dados: ou ela é executada por completo, ou então nada é executado.
- **Consistência:** as transações devem preservar a consistência do banco de dados, ou seja, transforma um estado consistente do banco de dados em outro estado consistente, sem necessariamente preservar o estado de consistência em todos os pontos intermediários.
- **Isolamento:** Outras transações não devem "ver" alterações parcialmente realizadas por uma transação, até seu encerramento com sucesso.
- **Durabilidade:** uma vez comprometida uma transação, suas atualizações sobrevivem no banco de dados, mesmo que haja uma queda subsequente do sistema.

Tipos de Transações

- **Transações Implícitas** - São transações que ocorrem automaticamente quando enviamos os comandos INSERT, UPDATE e DELETE para o BD. Essas transações são chamadas de implícitas, pois não precisamos indicar o início através de um comando BEGIN TRANSACTION ou um térmico através do comando COMMIT ou ROLLBACK.
- **Transações Explícitas –** Utilizam-se do conceito formal de transações. Nas transações explícitas devemos informar o início o fim da transação, devemos utilizar o comando BEGIN TRANSACTION para indicar o seu início e os comandos COMMIT ou ROLLBACK para indicar o fim da transação.
  - COMMIT – quando a transação foi finalizada com sucesso.
  - ROLLBACK - quando a transação foi finalizada com falha ou abortada.
- **Transações Automáticas** - é o modo padrão de gerenciamento de transações da maioria dos SGBD's. Uma instrução Transact-SQL é confirmada ou revertida quando concluída. Se uma instrução for concluída com sucesso, será confirmada; se ocorrer uma falha, será revertida. Esse mecanismo atua em modo de confirmação automática, sempre que esse modo padrão não for substituído por transações explícitas ou implícitas.

Administrando Falhas

Reconstrução (recovery)

Em algum momento no tempo, todo sistema computacional apresentará uma falha. O SGBD deve incorporar mecanismos de proteção e recuperação em caso de falhas.

Tipos de falhas

- **Falha de transação**
  - Uma transação que está executando alterações sobre a base de dados termina de forma anormal.
  - **Causas:**
  - Erro no programa que executa a transação, cancelamento por Deadlock, interrupção pelo usuário;
  - Demais programas permanecem em execução.
- **Falha de sistema**
  - O SGBD encerra sua execução enquanto há transações de alteração em execução.
  - **Causas:**
  - Erro interno no SGBD, problema de execução no software que suporta o SGBD, falha no hardware, problema na alimentação, etc.
- **Falha no meio de armazenamento**
  - Parte ou toda a base de dados está inacessível ou incorreta (falha no meio de armazenamento, falha na controladora, programas fazem alterações incorretas,...)

Recuperação de erro de transação

Deve ser executada pelo SGBD quando uma transação que estava sendo executada é cancelada (explicita ou implicitamente):

- Programa que executava a transação foi descontinuado (divide-byzero, etc.).
- **Recuperação:**
  - Os efeitos da transação em execução devem ser desfeitos ("undo").
  - Somente esses efeitos são atingidos pela reconstrução.
- **Consequências.**
  - O arquivo de "log" deve ser acessável de forma randômica.
  - Não pode haver outros usuários que "viram" os dados alterados pela transação (isolamento).

Recuperação de falhas no meio de armazenamento

A base de dados está danificada. Deve ser usada uma cópia de segurança. A partir dela, todas as operações sobre a área afetada devem ser refeitas. A cópia de segurança pode ser de diferentes tipos:

- **Total**
  - Toda a base de dados é copiada.
  - Não podem existir transações em execução (cópia "off-line").
- **Parcial**
  - Parte da base de dados é copiada.
  - Outras partes podem continuar em uso.
- **Incrementa**l
  - Somente as partes da base de dados que foram alteradas, desde a última cópia total, são copiadas.
- **"On-line"**
  - A base de dados é copiada enquanto transações são executadas.
  - Mecanismo de estampas de tempo é usado para sincronizar as cópias com o arquivo de "log".

Comandos Básicos SQL - PostgreSQL

COMANDO	FUNÇÃO
ALTER GROUP	Inclui ou exclui usuários em um grupo.
ALTER TABLE	Altera a definição da tabela.
ALTER USER	Altera a conta de um usuário do banco de dados.
BEGIN	Inicia um bloco de transação.
COMMIT	Efetiva a transação corrente.
CREATE CONSTRAINT TRIGGER	Define um novo gatilho de restrição.
CREATE DATABASE	Cria um banco de dados novo.
CREATE FUNCTION	Define uma nova função.
CREATE GROUP	Define um novo grupo de usuários.
CREATE INDEX	Define um índice novo.

CREATE RULE	Define uma nova regra.
CREATE SEQUENCE	Define um novo gerador de sequência.
CREATE TABLE	Define uma nova tabela.
CREATE TABLE AS	Cria uma nova tabela a partir do resultado de uma consulta.
CREATE TRIGGER	Define um novo gatilho.
CREATE TYPE	Define um novo tipo de dado.
CREATE USER	Define uma nova conta de usuário do banco de dados
CREATE VIEW	Define uma nova visão.
DELETE	Exclui linhas de uma tabela.
DROP DATABASE	Remove um banco de dados.
DROP FUNCTION	Remove uma função definida pelo usuário.
DROP GROUP	Remove um grupo de usuários.
DROP INDEX	Remove um índice.
DROP OPERATOR	Remove um operador definido pelo usuário.
DROP RULE	Remove uma regra.
DROP SEQUENCE	Remove uma sequência.
DROP TABLE	Remove uma tabela.
DROP TRIGGER	Remove um gatilho.
DROP TYPE	Remove um tipo de dado definido pelo usuário.
DROP USER	Remove uma conta de usuário do banco de dados.
DROP VIEW	Remove uma visão.
END	Efetiva a transação corrente.
GRANT	Define privilégios de acesso.
INSERT	Cria novas linhas na tabela.
LOCK	Bloqueia explicitamente uma tabela.
REINDEX	Reconstrói índices corrompidos.
REVOKE	Revoga privilégios de acesso.
ROLLBACK	Aborta a transação corrente.
SELECT	Retorna linhas de uma tabela ou de uma visão.
SELECT INTO	Cria uma nova tabela a partir do resultado de uma consulta.
SET	Muda um parâmetro de tempo de execução.
SET CONSTRAINTS	Especifica o modo de restrição da transação corrente.

SET SESSION AUTHORIZATION	Define o identificador do usuário da sessão e o identificador do usuário corrente, da sessão corrente.
SHOW	Mostra o valor de um parâmetro de tempo de execução.
UPDATE	Atualiza linhas de uma tabela.

Referências

DATE, C. J.; **Introdução a Sistemas de Banco de Dados**, Editora Campus, 8. Ed. 2004.

ELMASRI, Rames; NAVATHE, Shamkant B. **Sistema de Banco de Dados**. Rio de Janeiro, 4ed.: LTC, 2000.

GONZAGA, Jorge Luiz, **Dominando o PostgreSQL**. Rio de Janeiro: Editora Ciência Moderna Ltda, 2007.

HEUSER, Carlos Alberto. **Projeto de Banco de Dados**. 4ª ed., Porto Alegre: Sagra Luzzatto, 2001.

KORTH, Henry F., SILBERSCHATZ, Abraham. **Sistema de bancos de dados**. 3. ed. São Paulo : Makron, 1999.

NEVES, Denise Lemes Fernandes. **PostgreSQL**: Conceitos e Aplicações – São Paulo: Érica, 2002

NETO, Álvaro Pereira. **PostgreSQL:** Técnicas Avançadas – Versões OpenSource 7.X. 1ª ed., Érica, 2003.

NIEDERAUER, Juliano, **Guia de Consulta Rápida PostgresSQL**. Novatec Editora Ltda.

OLIVEIRA, Celso Henrique Poderoso de. **SQL:** Curso Prático. São Paulo: Novatec, 2002.

SILBERSCHATZ, A.; KORTH, H.; SUDARSHAN, S. **Sistema de Banco de Dados.** 5ª ed. Rio de Janeiro: Campus, 2006.

PostgreSQL Prático - http://pt.wikibooks.org/wiki/PostgreSQL_Prático - Acesso em 05/2011

http://pt.wikibooks.org/wiki/PostgreSQL_Prático/DCL/Administração_de_usuários,_grupos_e_privilégios – Acesso em 10/2011