Banco de Dados

Amilcar Fernandes Costa de Abreu

Florianópolis
2010

A162b de Abreu, Amilcar Fernandes Costa
Banco de Dados / Amilcar Fernandes Costa de Abreu / Florianópolis : Publicações do IF-SC, 2010.
101 p. : il. ; 00 cm.

Inclui Bibliografia

ISBN:

1. Informática. 2. Banco de dados. 3. Banco de dados – sistemas gerenciadores. 4. Banco de dados – projetos. 5. Linguagens de consulta. I. Título.

CDD: 001.64

Sistema de Bibliotecas Integradas do IF-SC
Biblioteca Dr. Hercílio Luz – Campus Florianópolis
Catalogado por: Augiza Karla Boso – CRB14/1092
Elaine Santos da Silva – CRB 14/1182

Organização de conteúdo:
Andrino Fernandes
Elaine Luz Barth
André de Oliveira Leite

Comissão Editorial:
Hamilcar Boing
Andrino Fernandes
Elaine Luz Barth

Produção e design instrucional:
André de Oliveira Leite
Elaine Luz Barth

Diagramação:
Hudson Ricardo Borges

Capa:
Lucio Baggio

Revisão ortográfica:
Marcos Pessoa

Sumário

Apresentação

Mensagem de Boas Vindas!

Caro Aluno (a),

Seja muito bem-vindo a esta unidade curricular. Nela você estudará os principais conceitos e aprenderá a utilizar os recursos que um banco de dados pode proporcionar.

Aliás, você sabe o que é um banco de dados?

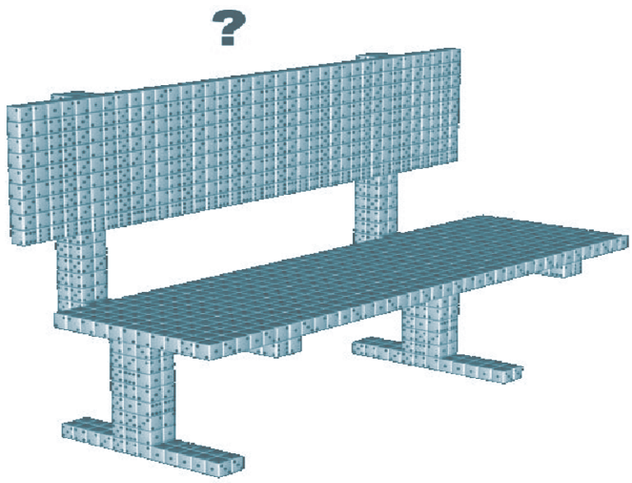


Figura 1:Banco de dados.

Se esta imagem lhe veio à cabeça assim que leu a pergunta, eu diria que foi uma boa tentativa. Entretanto, os bancos de dados são um pouco mais do que isso.

O principal objetivo de um banco de dados é garantir que seja armazenado, em suas estruturas, um grande volume de dados, de forma que esses dados possam ser acessados por vários usuários simultaneamente e que estejam disponíveis sempre que necessário. Eles são fundamentais para a existência de diversos sistemas com os quais nos deparamos todos os dias. Seja no sistema de vendas de um supermercado, na automação de um posto de gasolina, no trabalho, na escola, nos momentos de lazer, etc.. Sempre existe um sistema de informações que se utiliza de um banco de dados informatizado para manter seus dados "sãos e salvos".

Bom, agora que você já tem uma visão inicial da importância dos bancos de dados, que tal aprendermos um pouco mais sobre essa tecnologia?

CAPÍTULO

1

CONCEITOS BÁSICOS

Objetivo

Este capítulo inicial focaliza aspectos introdutórios da área de Banco de Dados e apresenta conceitos básicos sobre Dados, Informação e Sistemas de Informação.

Adotaremos como uma definição formal, que um banco de dados é um conjunto de dados organizados em um repositório de forma que os sistemas de informação possam utilizá-lo.

Certamente esta definição não é a única. Existe na literatura especializada uma série de outras definições. Você também encontrará a expressão base de dados (do original em inglês: database) como um sinônimo de bancos de dados.

DADOS, INFORMAÇÃO E SISTEMAS DE INFORMAÇÃO

Mas afinal, o que são dados?

Dado é o registro de um fato utilizando um conjunto de símbolos que o representam. Você sabia que o nome de uma pessoa é um dado? Que o preço de um produto é um dado? Qual a razão?

Simplesmente, porque em um nome são utilizadas as letras, símbolos do nosso alfabeto, dispostas e organizadas de forma que formem uma palavra que possa ser compreendida. Nesse mesmo sentido, o preço de um produto é representado por um conjunto de números dispostos de forma predefinida.

Vamos ver outro exemplo de dado? Para representar uma simples *data*, também utilizamos o conjunto de números (símbolos), apesar de organizá-los de forma um pouco diferente, é sem dúvida um tipo de dado.

Você conseguiria ilustrar outros exemplos de dados?

Anote: _____

_____

Segundo Elmasri, "os dados são fatos que podem ser gravados e que possuem um significado implícito". O que o autor quer dizer é que podemos memorizar dados e identificá-los conforme a situação. Nesse sentido, ao observarmos a representação de um dado, na maioria dos casos, *entendemos o que ele significa*.

Veja o seguinte exemplo: Imagine os dados abaixo escritos em uma folha de papel. O que está implícito nestas palavras? Fácil, não é mesmo?

- José da Silva
- R$ 2,35
- 21/10/2010

PARA REFLETIR

Em geral, os dados são manipulados por aplicações corporativas, chamados Sistemas de Informações.

Por exemplo, uma empresa pode ter um banco de dados contendo nome, telefone, CPF e endereço de seus clientes. Nesse mesmo banco de dados, podem ser armazenados os dados referentes às compras realizadas por este cliente, criando um sistema de informações sobre a relação comercial com seus clientes.

O que são Sistemas de Informações?

Entende-se por Sistema de Informação o conjunto de hardware, software e dados utilizados para prover informações a uma organização ou grupo de indivíduos.

Você utiliza algum sistema de informação em seu dia a dia?

Anote: _____

_____

Dessa forte ligação entre dados e sistemas de informação, Silberschatz, define bancos de dados como sendo "uma coleção de dados inter-relacionados, representando informações sobre um domínio específico".
O domínio específico ao qual ele se refere é uma porção do mundo real sobre o qual queremos guardar os dados.

É IMPORTANTE SABER QUE:

Na literatura da área de informática, será bastante comum a associação de dados e informações, pois são conceitos muito próximos. Um diagrama clássico da área demonstra que uma informação é o processamento aplicado a um dado ou a um conjunto deles. Provavelmente, a Figura 2 a seguir, já foi vista alguma vez pela maioria dos alunos dos cursos de informática.

Figura 2: Relação entre Dados e Informação.

Mas, o que é então "informação"?

Uma informação consiste na atribuição ou ampliação de um significado a um determinado dado.

Observe os exemplos:

Uma data é um dado, entretanto, saber que aquela data representa o dia do aniversário de um amigo é uma informação. Ou seja, o dado (a data) passou a ser informação ao ser atribuído a ele um significado maior do que a representação de um dia no ano.

Um conjunto de número pode significar muitas coisas. Enquanto conjunto de números, por exemplo, *23222321* é apenas um dado, mas ao se atribuir a ele um *significado*, ele passa a ser uma informação. Por exemplo, esse dado pode representar o número do telefone de um amigo.

Podemos concluir então que:

O dado tem que ser entendido por quem o lê (receptor) para se tornar uma informação para ele.

• Observe as seguintes características mencionadas nos itens abaixo, que podem IMPEDIR ou PERMITIR que um dado possa ser considerado uma informação:

a) Caso um texto esteja em uma língua que o receptor não entende é possível que esse dado nunca se torne uma informação para ele.

Imagine uma situação onde existem duas garrafas transparentes cheias de um líquido incolor. Em cada uma delas existe um rótulo onde respectivamente, encontramos as seguintes palavras: giften e vann. Para auxiliar os rótulos contam com a inscrição: Bottled in Norway. Caso você não entenda a língua inglesa, não descobriria que as garrafas são de origem Norueguesa. Mesmo que entendesse Inglês, ainda teria que conhecer o significado (em Norueguês) de cada uma das palavras escritas nas garrafas para decidir qual das duas escolheria para matar sua sede.

GLOSSÁRIO

Segundo um dicionário Norueguês-Portugues, giften significa água e vann significa veneno.

b) Além de ser compreensível, o dado tem que ter valor para o receptor.

Utilizando novamente o exemplo numérico *23222321*, por mais que ele representasse o número de telefone de um excelente mecânico, ele não teria muito valor para uma pessoa que não possui um veículo. Então, diz-se que o dado tem que ter um valor ou ser do interesse do receptor para poder ser considerado uma informação.

c) Outra característica importante é a tempestividade: em alguns casos o dado pode ter valor se ele for entregue ao receptor em um determinado instante no tempo. Passado esse momento, ele já não tem mais valor para o receptor e se torna apenas um dado.

Vamos pensar que a seqüência de números: 02-10-22-23-45-60, é a combinação de números sorteada no último jogo da loteria. Para alguém que não fez apostas para o último sorteio da loteria, eles são apenas um dado. Por outro lado, mesmo para alguém que nunca arriscou a sorte na loteria, se essa combinação representasse os números que, garantidamente, serão sorteados no próximo jogo, ela se torna uma informação.

CONCLUINDO

Apesar de existir uma fronteira subjetiva entre dado e informação, cabe enfatizar um ponto importante: NÃO É POSSÍVEL processar informações em um computador.

Por esse motivo, as informações são representadas por dados, que são armazenados e processados pelo computador para que se possam obter informações.

Em outras palavras, um computador processa os dados que representam as informações. Cabe então ao ser humano a organização e apresentação dos dados para que ele signifique uma informação para o receptor.

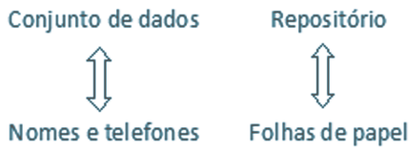
1.1 Banco de Dados

Partindo da definição de banco de dados utilizada no início deste capítulo, vamos utilizar uma situação cotidiana, fora do ambiente computacional, para nos auxiliar a compreender todos os recursos providos pelas tecnologias de bancos de dados.

Qual é a definição de bancos de dados que estamos utilizando?

Anote: _____
_____

Um elemento conhecido de todos e que pode ser considerado como um banco de dados é o caso de uma agenda de endereços e telefones. Seu objetivo principal é guardar, de forma organizada, nomes de pessoas, endereços e números de telefone de nosso interesse. Nesse sentido, nossa agenda de telefones corresponde perfeitamente a essa definição.



Um esquema típico (organização) de uma agenda de endereços e telefones pode ser visto na Figura 3, abaixo. Nós o utilizaremos como exemplo para fazer algumas analogias com outros conceitos sobre bancos de dados.

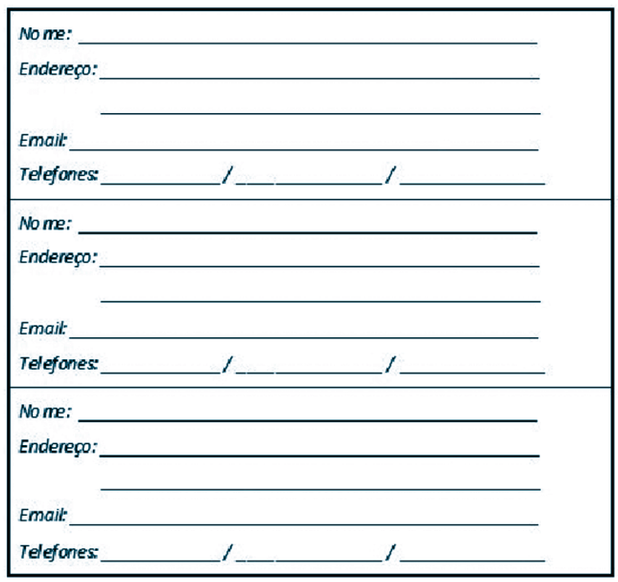


Figura 3: Organização da Agenda de endereços e telefones.

1.2 Elementos de um Banco de Dados

A Figura 3 retrata uma das páginas de nossa agenda, onde você pode ver quais são os dados a serem armazenados e como eles serão organizados. A partir dela, verificamos a existência de 2 conceitos importantes: atributo e registro. Veja:

• Cada dado específico de uma pessoa é chamado de atributo. Assim, podemos identificar na Figura 3, quatro atributos: Nome, Endereço, Email e Telefones.

• Cada conjunto de dados de uma pessoa, que será anotado na agenda, é chamado de registro. Assim, em cada página de nossa agenda, podemos armazenar até três registros.

Esses conceitos serão utilizados com freqüência a partir de agora. Grave-os bem.

No ambiente computacional os dados também são armazenados na forma de registros. Antes da existência dos bancos de dados atuais, os dados eram armazenados em arquivos de texto, também chamados de arquivos planos (flat file) – Veja mais informações no Quadro 1, na página 18. Atualmente, podemos imaginá-lo como uma tabela, onde cada atributo corresponde a uma coluna e que cada registro ocupa uma linha.

Com essa visão de um banco de dados, uma forma de representar nossa agenda de telefones e endereços seria conforme mostra a tabela abaixo:
Em um banco de dados, podemos realizar as seguintes operações básicas: inclusão, atualização, exclusão, recuperação. Vejamos:

Nome	Endereço	Email	Telefones
José da Silva	7 de Setembro,12 Joinville/SC	jsilva@email.com	32323232,88334433
Maria da Graça	Itu,456 Blumenau/SC	mg456@email.com	98987770
João Pereira	Margaridas,33 Irati/PR	jpereira@email.com	

Inclusão: ocorre quando queremos cadastrar um novo registro em nosso banco de dados;

Atualização: ocorre quando queremos alterar o valor de um atributo de registro que já existe em nosso banco de dados;

Exclusão: ocorre quando queremos eliminar um ou mais registros existentes em nosso banco de dados;

Recuperação: ocorre quando consultamos o banco de dados para obter um dado que está armazenado em nosso banco de dados;

1.3 O caso prático

Os bancos de dados realizam as operações acima de forma bastante eficiente. Porém, eles ainda não têm a capacidade de analisar o mundo real e identificar quais dados devem ser armazenados para atender a um sistema de informações. Cabe a nós, profissionais da informática, o projeto de um banco de dados capaz de armazenar os dados necessários ao sistema de informações que pretendemos desenvolver.

Para tal, a seguir descreveremos o caso da J&J Vídeo, uma vídeo locadora, que identificou a necessidade de automatizar seu processo de trabalho em virtude do crescimento de seu empreendimento.

A J&J Vídeo foi fundada há cinco anos e atendia a uma pequena clientela da região. Inicialmente, os irmãos Juvêncio e Juvenal, anotavam em um caderno, o nome da pessoa, o filme que fora alugado, a data da devolução do mesmo e o valor pago.

Os anos se passaram e o "caderninho" atendia bem. Porém, há mais ou menos um ano, uma grande indústria instalou-se na região e com ela surgiram vários novos moradores. Com o aumento da clientela, os irmãos sentiram uma grande dificuldade em gerenciar a locação de filmes utilizando um controle manual. Solicitaram então a uma empresa de informática da capital que desenvolvesse um sistema informatizado para controlar o funcionamento de sua empresa.

Os requisitos do sistema são simples:
- quando um filme novo é adquirido ele é cadastrado no sistema;
- um cliente, para alugar um filme, deve estar cadastrado no sistema;

E o processo continua o mesmo:
- o cliente se identifica;
- o cliente escolhe um filme;
- é registrada a locação do referido filme;
- Na devolução do filme, o registro de locação é atualizado, indicando que o filme foi devolvido e está disponível para nova locação;

Nesta unidade, utilizaremos o caso da J&J Vídeo para projetar um banco de dados que atenda as necessidades da empresa.

QUADRO 1

Uma organização bastante comum era a gravação de um registro em cada linha do arquivo e a separação dos atributos era feita por meio de um caractere separador pré-definido.

```
José da Silva;7 de Setembro, 12 - Joinville/SC;jsilva@email.com.br;32323232,88334433
Maria das Dores;Itu,456 - Blumenau/SC;mdores@email.com.br;98987770
João Pereira;Margaridas, 33 - Irati/PR;jp@email.com.br;
```

Arquivo com ";" como separador.

Em outra situação, o espaço destinado aos dados era definido pela largura padrão de cada coluna, independente do tamanho dado armazenado. Perceba que essa forma de armazenamento é muito parecida com as tabelas que costumamos construir.

```
José da Silva   7 de Setembro,12  Joinville/SC   jsilva@email.com   32323232,88334433
Maria da Graça  Itu,456           Blumenau/SC   mg456@email.com    98987770
João Pereira    Margaridas,33     Irati/PR      jpereira@email.com
```

Arquivo com largura fixa para cada atributo

RESUMO

O objetivo da utilização da tecnologia de bancos de dados é o armazenamento seguro dos dados e a sua rápida disponibilização quando eles se fazem necessários.

Entendemos que um banco de dados pode ser definido como um conjunto de dados devidamente relacionados. Uma disposição desordenada dos dados não pode ser referenciada como um banco de dados. Por dados podemos compreender como fatos conhecidos que podem ser armazenados e que possuem um significado implícito.

Esses dados pertencem a uma porção do mundo real que desejamos guardar informações. Para tal, um banco de dados deve ser uma representação, um modelo computacional, de como os dados estão organizados no mundo real.

A criação de um banco de dados abrange a organização interna de um arquivo especial para armazenar os dados que serão inseridos.

A manipulação dos dados trata das operações que poderão ser realizadas com esse banco de dados: consulta, atualização e exclusão de dados existentes e inclusão de novos dados.

ATIVIDADES DE APRENDIZAGEM

1 – Elabore uma lista dos atributos que você acha que seriam relevantes na construção de um banco de dados para a J&J Vídeo;

Anote:

2 – Discuta com mais 3 colegas e identifique os atributos divergentes;

CAPÍTULO

2

SISTEMAS GERENCIADORES DE BANCOS DE DADOS

Objetivo

Este capítulo aborda os Sistemas Gerenciadores de Bancos de Dados. Trata-se de um sistema que facilita o desenvolvimento de aplicações que utilizam um banco de dados.

A manipulação dos dados se torna um tanto trabalhosa, principalmente quando o volume de dados aumenta significativamente. Veja o QUADRO 2, a seguir. Da mesma forma, o compartilhamento dos dados por um grande número de usuários requer um controle de concorrência eficiente, além da necessidade de limitar o acesso aos dados.

Para facilitar a manipulação de dados por grandes sistemas e garantir a integridade dos dados foram desenvolvidas aplicações específicas chamadas Sistemas Gerenciadores de Bancos de Dados (SGBD).

"O principal objetivo de um SGBD é proporcionar um ambiente tão conveniente quanto eficiente para a recuperação e armazenamento das informações do banco de dados." (Silberschatz, 2004)

PARA REFLETIR

Embora a idéia de desenvolver um sistema que sirva de apoio para outro software possa parecer estranha, não esqueça que qualquer programa de computador, utiliza de recursos providos por outro software, o sistema operacional, para funcionar de forma eficiente.

QUADRO 2

Conforme vimos no QUADRO 1, antes dos bancos de dados atuais, os dados eram armazenados em arquivos e o processamento desses arquivos começou a se tornar um problema para os profissionais da área. Entenda o porquê:

- No processamento tradicional de arquivos, a estrutura dos dados estava incorporada ao programa de acesso. Desta forma, qualquer alteração na estrutura de arquivos implicava na alteração no código fonte de todos os programas. Além disso, o programador tinha que criar todas as rotinas para a manipulação e acesso aos dados: inclusões de novos dados, exclusões e atualizações dos dados existentes.
- Observou-se que a tarefa de acesso aos dados era uma constante em todos os sistemas de informação. A partir disso, algumas empresas começaram a desenvolver algumas ferramentas específicas para prover a manipulação e acesso aos dados. Inicialmente elas ofereciam, para o desenvolvimento de sistemas, uma linguagem de programação com funções específicas para a manipulação dos dados em arquivos e um padrão para a organização dos dados nesses arquivos. Não existia um padrão de mercado e a troca de um fornecedor significava o desenvolvimento de um novo sistema para substituir o anterior.

As características de um SGBD

Um SGBD é o elo entre os sistemas de informação de uma organização e os bancos de dados. Ele fornece um conjunto padrão de operações para a criação e a manipulação dos bancos de dados, além de facilitar o desenvolvimento de novas aplicações, garantindo que os dados armazenados sob sua responsabilidade estarão seguros. Para tal, ele possui 5 características que o diferem dos processadores de arquivos e o tornam extremamente vantajosos.

São elas: abstração de dados, controle de redundância de dados, controle de concorrência, controle de acesso e tolerância a falhas.

Vamos ver como funciona cada uma delas?

a) Abstração de dados

Um SGBD cria inicialmente o que chamamos de *abstração de dados*. Em outras palavras, ele fornece uma visão conceitual dos dados que não correspondem exatamente à organização física dos dados.

Essa visão conceitual está mais próxima de como o usuário imagina o dado ou a organização desses dados, facilitando a sua manipulação.

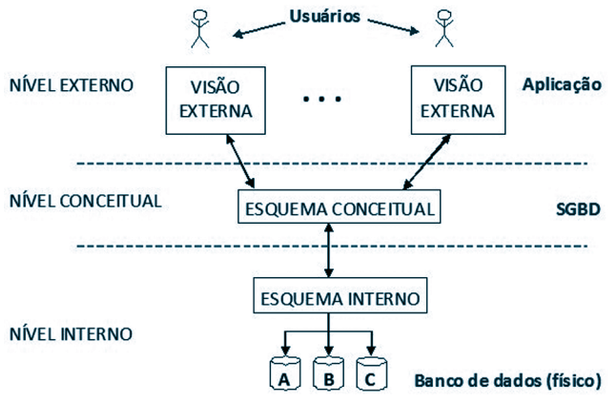


Figura 4: Abstração dos dados.

Vamos discutir um pouco sobre a figura acima.

Um SGBD opera em uma arquitetura de software de níveis, onde cada um tem uma função específica, entende como as coisas funcionam em seu ní-

vel, porém abstraem como as coisas estão organizadas nos níveis mais abaixo.

Assim, os usuários só precisam se preocupar como os dados conforme a aplicação (sistema de informação) os apresenta. Os projetistas da aplicação, por sua vez, trabalham com uma visão conceitual, pois , quando eles desenvolvem as aplicações para acessar os dados, eles vêem os dados por meio de um SGBD.

Como o SGBD entende como a aplicação vê os dados e como eles estão organizados fisicamente, é fácil para ele obter os dados solicitados pela aplicação. Entenda cada nível separadamente:

• *Nível Interno (Físico)*: Nível mais baixo de abstração. Descreve como os dados estão realmente armazenados, englobando estruturas complexas de baixo nível;

• *Nível Conceitual*: descreve quais dados estão armazenados e seus relacionamentos. Neste nível, o banco de dados é descrito através de estruturas relativamente simples, que podem corresponder a estruturas complexas no nível físico.

• *Nível de visões do usuário*: descreve partes do banco de dados, de acordo com as necessidades de cada usuário, individualmente.

Comentário:

A capacidade de gerar visões diferentes dos dados de forma transparente significa que, a partir de um mesmo SGBD pode-se obter dados que pertencem a sistemas distintos. Observando o diagrama da Figura 4, um SGBD poderia fornecer dados formados pelos bancos A e B para um usuário ou sistema e para outro os dados dos bancos B e C, de tal forma que os usuários e os desenvolvedores dos sistemas não saberiam a partir de qual banco os dados foram fornecidos.

b) Controle de Redundância de dados

A duplicação dos dados (também denominada de redundância de dados) foi um dos primeiros problemas do processamento tradicional de arquivos. Nessa técnica, cada sistema deveria manter seu próprio conjunto de arquivos e dados. Desta forma, os dados utilizados por um usuário existiam nos arquivos de outro usuário. Além disso, a atualiza-

ção de um dado deveria ser feita em todas as cópias do referido dado para manter sua integridade.

Essa situação se torna um problema se considerarmos o processamento excessivo para atualizar todas as cópias de um determinado dado e o espaço físico ocupado por suas inúmeras cópias.

O controle de redundância de dados provido por um SGBD é baseado no compartilhamento dos dados. Nesse caso, existe uma única ocorrência de um determinado dado e esse dado é acessado pelos vários usuários ou sistemas que o necessitam.

Além disso, um SGBD impede que um mesmo dado seja cadastrado em duplicidade. Por exemplo, um médico não pode ter dois pacientes agendados para o mesmo horário.

c) Controle de Concorrência

Em alguns casos, sistemas ou usuários distintos precisavam de um mesmo dado ao mesmo tempo. Para o caso da leitura dos dados isso não era um problema muito difícil de resolver. A situação crítica a ser resolvida acontecia quando ambos os sistemas precisavam atualizar o dado. Um exemplo prático é descrito no quadro abaixo.

QUADRO 3

O sistema de almoxarifado de uma organização registra a quantidade em estoque de um determinado produto A. Cada vez que algum funcionário retira uma porção desse produto, a quantidade restante deve ser atualizada.

Operador 1		Operador 2	
Ação	**Resultado**	**Ação**	**Resultado**
Consulta Estoque Produto A	Saldo = 15	Consulta Estoque Produto A	Saldo = 15
Retira 3 unidades	Saldo = 12	Retira 5 unidades	Saldo = 10
Atualiza Saldo Produto A	Saldo = ~~X~~2	Atualiza Saldo produto A	Saldo = ~~X~~0

Situação real: Saldo = 8 unidades

O saldo do produto estará errado em qualquer um dos casos. Essa situação origina um problema conhecido como **inconsistência de dados**.

Numa situação como essa, o programador deveria programar mecanismos para tentar evitar que a atualização de um dado ocorresse enquanto outro usuário ou sistema utiliza aquele mesmo dado. Um SGBD implementa esse mecanismo, que é chamado de controle de concorrência.

d) Controle de Acesso aos Dados

Evitar que pessoas não autorizadas se apoderassem dos dados existentes nos arquivos e começassem a manipulá-los foi outro desafio enfrentado pelos programadores. Qualquer um que tivesse acesso aos programas e aos arquivos poderia atualizá-los sem que fossem autorizados a fazê-lo.

Podemos citar como exemplo, o caso de um programador que acessou o arquivo do sistema de Recursos Humanos e atualizou seu próprio salário.

Cada sistema deveria implementar mecanismos de controle de acesso aos dados. Para facilitar o trabalho dos desenvolvedores, um SGBD mantém diversas estruturas nas quais os usuários devem ser cadastrados e são atribuídos a eles as possibilidades e as restrições de cada usuário.

e. Tolerância a falhas

Inicialmente, os programadores não tinham ainda a menor idéia de como proceder nos casos de falha do sistema, embora eles exercessem controle sobre os possíveis problemas que poderiam ocorrer com os dados.

Para ilustrar melhor o problema, vamos utilizar uma situação descrita a seguir:

Imagine um sistema de automatização de vendas de um supermercado. Podemos dizer que ele tem algumas operações bastante simples, conforme descrição no diagrama abaixo:

1 – Inicia a venda;
2 – Registra o produto na nota fiscal;
3 – Atualiza estoque do produto com uma unidade a menos;
4 – Calcula o valor total da compra;
5 – Volta ao passo 2 até que o último produto seja registrado;
6 – Imprime a nota fiscal com todos os itens vendidos;

Em uma situação normal de operação pode-se dizer que o sistema, executando o algoritmo acima, atenderia às necessidades da organização. Porém, caso ocorresse uma queda no fornecimento de energia elétrica durante o processo, seria complicado determinar o ponto de partida do sistema após o restabelecimento da energia elétrica.

Um SGBD mantém estruturas e controles sobre todas as operações que ocorrem em um banco de dados, e em caso de falha, restaura os dados conforme estavam antes da ocorrência da falha.

Qual a definição de tolerância de falhas?

Definimos como tolerância a falhas a capacidade de um sistema de se recuperar, ou seja, retomar o processamento normal sem gerar inconsistência nos dados, após uma falha.

RESUMO

A utilização das oferece inúmeras vantagens comparadas com o processamento tradicional de arquivos. Dentre elas podemos citar:
• Isolar os usuários dos detalhes mais internos do banco de dados (abstração de dados) criando a independência entre dados e aplicação;
• Proporcionar redução de redundância e de inconsistência de informações.
• Permitir compartilhamento de dados;

Para facilitar a utilização dos recursos fornecidos pelas tecnologias de bancos de dados, surgiram sistemas dedicados ao seu gerenciamento. São denominados de Sistemas Gerenciadores de Bancos de Dados (SGBD). Eles são essenciais para a interligação entre aplicação e os bancos de dados.

A segurança de um banco de dados deve ser vista sob vários aspectos, sendo que todos eles são atendidos num SGBD moderno. Em muitas situações, um banco de dados é acessado por várias pessoas ao mesmo tempo (compartilhamento de dados). Além de fornecer uma identificação para cada usuário que acessa o banco de dados (restrição a acesso não autorizado), ele permite conceder permissões diferenciadas para cada usuário (segurança lógica dos dados).

No que se refere à segurança física dos dados, existem algumas funcionalidades providas pelos SGBDs para evitar a perda de dados. Em uma delas, ele cria rotinas automatizadas de cópia de segurança (backup) e mecanismos de restauração do banco de dados a partir de uma cópia de segurança (restore). Além disso, um SGBD garante que, mesmo em caso de falhas, os dados estarão no mesmo estado que estavam antes da falha (tolerância a falhas).

Atividade de aprendizagem

1 – Pesquise na internet 3 fornecedores de SGBDS sendo que pelo menos um deles deve ser Software Livre;

2 – Participe do fórum de discussões sobre os fornecedores de SGBSs encontrados e avalie dos produtos que você encontrou em relação aos produtos que seus colegas encontraram.

CAPÍTULO

3

MODELOS DE DADOS

Objetivo

O capitulo 3 apresenta o modelo conceitual de Banco de Dados, especificamente Modelos lógicos com base em objetos na perspectiva Modelo Entidade--Relacionamento

Sabendo que um banco de dados é a pedra fundamental de um sistema de informações, nosso objetivo é utilizar os recursos computacionais para armazenar os dados e processá-los, oferecendo informações úteis aos usuários desse sistema.

Nesse sentido, o primeiro passo para a criação de um banco de dados é entender a porção do mundo real que este sistema deseja monitorar. É preciso identificar todos os elementos que compõem o domínio da aplicação a ser mantido.

Não esqueça que o domínio da nossa aplicação é composto pelos dados da J&J Vídeo. Por esse motivo, boa parte dos exemplos a seguir será baseada nele.

3.1 Modelo conceitual de dados

Em geral, um sistema de informações é especificado a partir de um conjunto de regras e funcionalidades que ele deve implementar, são os chamados requisitos. Para a criação do banco de dados, que irá suportar esse sistema, é preciso identificar quais dados serão necessários para atender aos requisitos do sistema. Identificados os dados, é preciso ainda identificar os relacionamentos existentes entre eles, o significado semântico dos dados, entre outras coisas.

Em virtude da independência entre dados e aplicação, para a criação do banco de dados são utilizadas algumas ferramentas conceituais para a descrição dos dados e dos relacionamentos entre eles.

Vários modelos conceituais desenvolvidos são classificados em três grupos diferentes: *modelos lógicos com base em objetos, modelos lógicos com base em registros e modelos físicos.* (Silberschatz, 2004)

Vamos agora, aprender um pouco mais sobre os modelos conceituais:

Modelos Lógicos com base em objetos

- Modelo entidade-relacionamento;
- Modelo orientado a objeto;
- Modelo semântico de dados;
- Modelo funcional de dados;

Nesta unidade estudaremos apenas o modelo entidade-relacionamento pois, ele é bastante utilizado na prática para o projeto em alto nível de um banco de dados. Seu principio básico é a representação do mini-mundo, a ser informatizado, por meio de objetos básicos, chamados entidades, e dos relacionamentos entre eles. (Figura 5)

Uma entidade é um elemento existente no mini-mundo real a ser modelado. Por exemplo, uma Pessoa é uma entidade, um Livro, etc. As entidades são descritas no banco de dados por meio de seus atributos: nome, título, autor, etc. Além disso, o modelo entidade-relacionamento apresenta algumas regras que o conteúdo do banco de dados deve respeitar. Por exemplo, um livro deve ter pelo menos um autor.

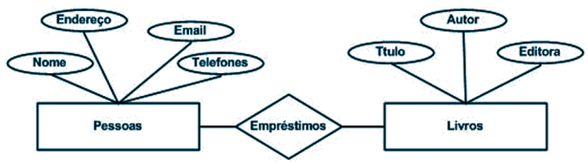


Figura 5: Modelo Entidade-Relacionamento.

Modelos Lógicos com base em registros

Os modelos lógicos baseados em registros são utilizados para descrever um banco de dados por meio de registros, onde cada registro define um número fixo de atributos (ou campos) de tamanho fixo. Existem três modelos baseados em registros. São eles: Modelo relacional, Modelo de rede e Modelo hierárquico.

• Modelo relacional (Figura 6): usa um conjunto de tabelas para representar tanto os dados como os relacionamentos entre eles. Cada tabela possui múltiplas colunas (atributos) e cada uma possui um nome único.

• Modelo em rede (Figura 7) e o Modelo hierárquico (Figura 8), representam os dados por meio de registros (similar a estrutura de dados record comuns em linguagens de programação) e as relações entre esses registros são representadas por links entre eles.

Para ilustrar os modelos baseados em registros, vamos utilizar a tabela abaixo onde são colocados alguns registros de um banco de dados de uma escola que apresenta o nome dos alunos, sua matrícula, seu email, a disciplina que ele cursou e a nota obtida.

Nome	Matricula	email	Disciplina	Nota
José da Silva	2010020188	jsilva@email.com	AVA	E
João Pereira	2010020193	jpereira@email.com	PED	E
Maria da Graça	2010020232	Mg456@email.com	AVA	S
José da Silva	2010020188	jsilva@email.com	PED	E
Maria da Graça	2010020232	Mg456@email.com	PED	S
João Pereira	2010020193	jpereira@email.com	INN	S
Maria da Graça	2010020232	Mg456@email.com	IIN	S

No modelo relacional os dados são dispostos em tabelas distintas, de acordo com os dados que elas contêm. Assim, ele apresenta mais de uma tabela para evitar a repetição dos dados.

José da Silva	2010020188	jsilva@email.com
João Pereira	2010020193	jpereira@email.com
Maria da Graça	2010020232	mg456@email.com

2010020188	AVA	E
2010020232	PED	E
2010020193	AVA	S
2010020188	PED	E
2010020232	PED	S
2010020193	INN	S
2010020232	IIN	S

Figura 6: Modelo Relacional.

E os outros dois, quais a diferença entre eles? No modelo em rede os registros são organizados em gráficos arbitrários e no modelo hierárquico, os registros estão organizados em árvores.

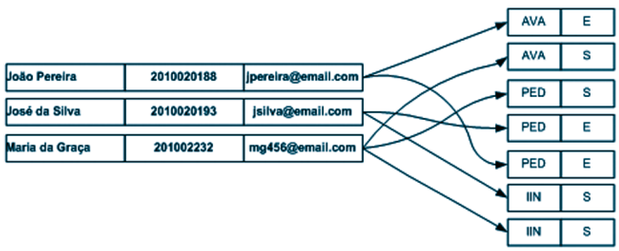


Figura 7: Modelo em rede.

PARA REFLETIR

O modelo relacional tem sido o mais utilizado para a modelagem de bancos de dados nos últimos anos. Os modelos em rede e hierárquico foram bastante utilizados no passado.

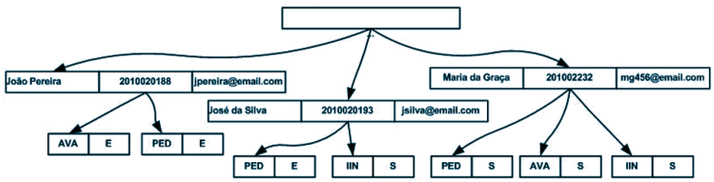


Figura 8: Modelo hierárquico.

Importante: Não podemos esquecer que, devido a algumas limitações da máquina (computador) não podemos criar um modelo conceitual dentro do ambiente computacional. Apesar de existirem aplicativos que nos auxiliam na elaboração desses modelos, eles são apenas representações gráficas de uma abstração dos dados.

RESUMO

Modelos Lógicos de Dados	Conjunto de ferramentas conceituais para a descrição dos dados, dos relacionamentos entre os mesmos e das restrições de consistência e integridade. Dividem-se em: baseados em objetos, baseados em registros.
Modelos lógicos baseados em objetos	Fornecem uma descrição dos dados no nível conceitual e de visões de usuários. *Exemplos:* entidade-relacionamento, orientado a objetos. NOTA: No modelo orientado a objetos, código executável é parte integrante do modelo de dados.
Modelos lógicos baseados em registros	O banco de dados é estruturado em registros de formatos fixos, de diversos tipos. Cada tipo de registro tem sua coleção de atributos. Há linguagens para expressar consultas e atualizações no banco de dados. *Exemplos:* relacional, rede, hierárquico. NOTA: No modelo relacional, dados e relacionamentos entre dados são representados por tabelas, cada uma com suas colunas específicas.

ATIVIDADE DE APRENDIZAGEM

1 – Procure identificar alguns dados possíveis de serem encontrados no domínio do sistema da J&J Vídeo e elabore um diagrama de cada tipo de modelo visto: relacional, em rede e hierárquico.

CAPÍTULO

4

MODELO ENTIDADE RELACIONAMENTO

Objetivo

Este capítulo focaliza o conceito e as características do Modelo Entidade-Relacionamento (MER). Esse modelo descreve os dados na forma de um diagrama que apresenta as Entidades, seus atributos e os Relacionamentos entre essas entidades.

Conforme introduzido no capítulo anterior, o Modelo Entidade-Relacionamento define uma representação dos dados do mundo real por meio de elementos básicos.

Observando esses elementos separadamente, não se tem uma visão completa de todas as características existentes nos dados. Para resolver isso, é comum colocá-los em um único diagrama, chamado diagrama Entidade-Relacionamento (DER).

Existem vários formas de representação gráfica dos elementos do modelo E-R em um DER. Será apresentada uma notação que pode não ser a mesma encontrada em outras literaturas sobre o assunto. Porém, também não existe um consenso entre a melhor forma de representá-los.

Como precisaremos de um mini-mundo para modelar, utilizaremos o problema da J&J Vídeo como caso de estudo, para facilitar o entendimento e ilustrar os conceitos que serão vistos a seguir.

Com certeza, você já entrou em uma vídeolocadora para alugar um filme. Agora eu lhe pergunto: você já percebeu que existe um sistema para controlar o aluguel dos filmes?

4.1 O Conceito de Entidade

O ponto de partida para organização dos dados de um sistema de informações é a identificação das entidades envolvidas.

Uma *entidade* é uma representação de um elemento existente no mundo real do qual se deseja guardar os dados. Esse elemento pode ser uma pessoa, um objeto, ou um fato que existe (ou existiu) no domínio observado.

Se imaginarmos um sistema para controle do movimento de uma vídeolocadora, podemos identificar rapidamente algumas entidades: o cliente da vídeolocadora, o filme a ser locado, o atendente da locadora. Em outros sistemas, uma entidade pode ser um funcionário, o departamento onde ele está lotado, os materiais produzidos ou comercializados pela empresa, seus fornecedores, etc.

Entretanto, quando se fala em modelo conceitual, identificada uma entidade do nosso sistema, obviamente devemos nos lembrar que estamos trabalhando com um conjunto delas, ou seja, não teremos apenas um cliente, ou um único filme cadastrado em nosso sistema, mas sim um conjunto deles.

Deve-se então, modelar o *domínio da aplicação* prevendo que as entidades que têm as mesmas características farão parte do mesmo conjunto de entidades. Assim teremos o conjunto das entidades cliente, o conjunto das entidades filme, e assim sucessivamente.

No *Modelo Entidade-Relacionamento* (MER), utilizaremos um retângulo para representar uma entidade. Para identificá-la, utilizaremos sempre um substantivo no plural e com a primeira letra em maiúsculo. Com isso, ficará claro que estamos representando o conjunto de todas as entidades daquela categoria.

Figura 9: Conjuntos de Entidades.

4.2 Atributos

Cada entidade possui um conjunto de dados que a caracteriza. Por exemplo, um cliente tem nome, endereço, telefone, etc.; um filme tem título, diretor, atores, etc.. Conforme visto anteriormente, esses dados são chamados de atributos.

Assim, para representar uma entidade em nosso MER, além nome para a identificação do conjunto de entidades de mesmas características, utilizamos os atributos que descrevem os dados da entidade.

Figura 10: Entidade e seus atributos.

Alguns atributos têm características diferentes de outros, em virtude disso, eles são classificados de acordo com o tipo de dados que eles armazenarão. Eles podem ser: elementares (atômicos, ou seja, simples e indivisíveis), compostos (por vários outros dados) e multivalorado (formado por vários valores diferentes).

a) Atributos Elementares

Um atributo elementar carrega um único dado sobre a entidade que estamos modelando. Em geral, os atributos elementares são chamados apenas de atributos. Para representar um atributo em um MER, adotaremos uma notação simples, composta por uma elipse contendo a identificação no singular para cada atributo da entidade. Conforme mostra a Figura 10, apresentada anteriormente.

b) Atributos Compostos

Um atributo deve ser considerado como sendo um valor único para uma determinada característica da entidade em questão, não podendo ser decomposto em partes menores. Por exemplo, se consideramos que o Nome é um atributo, não podemos decompô-lo em primeiro nome e sobrenome.

Entretanto, existem situações onde a decomposição é necessária. Diante disso, consideramos que este atributo em específico é um atributo composto, pois seu valor é composto por diversas partes e cada uma delas se tornará um atributo.

Se observarmos o atributo Endereço, apresentado no exemplo da agenda de telefones do Capítulo 1, veremos que ele é formado por Rua, Bairro, Estado, Cidade e CEP. Temos assim, um atributo composto por outros atributos. A representação de um atributo composto pode ser vista na Figura 11, abaixo.

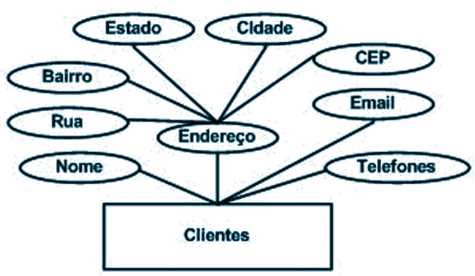


Figura 11: Atributo Composto.

c) Atributos Multivalorados

PARA REFLETIR

O conceito de atributo multivalorado remete a um conceito contrário a esse, o de mono-valorado. Assim, vale esclarecer que, um atributo elementar, é um atributo mono-valorado, pois pode assumir um único valor para cada entidade daquele conjunto.

Mas como a maioria dos atributos é elementar, dispensaremos essa nomenclatura para não gerar confusões durante o processo de modelagem, deixando o detalhamento para as exceções que são os atributos compostos e os multivalorados.

Até o momento, identificamos em nossas entidades dois tipos de atributos: os elementares (simplesmente chamados de atributos) e os atributos compostos. Um atributo elementar tem um valor único e que não pode ser decomposto. Um atributo composto é formado por atributos elementares, onde cada atributo elementar pode ter apenas um valor único.

Entretanto, ainda existem situações que um determinado atributo pode assumir mais de um valor, chamado de atributo multivalorado. Se observamos, no mesmo exemplo do Capítulo 1, o atributo Telefones, veremos que ele é um atributo multivalorado, pois ele pode conter mais de valor. Um atributo multivalorado permite que, cada entidade desse conjunto, contenha um número qualquer de valores associados a esse atributo.

A representação de um atributo multivalorado em um diagrama entidade-relacionamento (DER) é uma elipse com linhas duplas, conforme mostra a Figura 12, abaixo:

Figura 12: Atributo Multivalorado.

Por meio desses três tipos de atributos, podemos representar todos os dados que serão armazenados sobre uma entidade. Entretanto, é preciso fazer uma distinção de dois casos especiais.

Num deles, o atributo merece um destaque, em virtude de sua importância, e em outra situação o atributo é relevante, pois pode ser obtido a partir de outros atributos. Vejamos então o conceito de atributo determinante e de atributo derivado.

d) Atributos determinantes

Um atributo muito importante, que deve ser identificado a partir do modelo conceitual, é um atributo que assuma um valor único e que identifique cada uma das entidades que compõem o conjunto de entidades. Isto é, dado um conjunto de entidades, não existem duas entidades no conjunto com o mesmo valor para aquele atributo. A esse atributo damos o nome de atributo determinante.

Por exemplo, em nosso caso de estudo, podemos ter várias cópias de um mesmo filme. Ou seja, a entidade Filmes conterá várias entidades onde o atributo Título tem o mesmo valor.

Para diferenciar um exemplar de outro do mesmo filme, daremos um "número de filme" para cada cópia de filme disponível na vídeolocadora. Assim, cada exemplar de um filme tem um único número e a cada valor de "número de filme" corresponde a um único exemplar de um filme.

Com isso, o atributo "número de filme" é o atributo determinante da entidade Filmes, identificando unicamente cada exemplar.
A representação de um atributo determinante é feita de forma simples, simplesmente sublinhando o nome do atributo, conforme pode ser visto na Figura 13, abaixo.

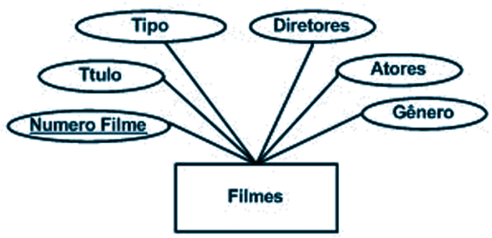


Figura 13: Atributos Determinantes.

e) Atributos Derivados

Um atributo derivado, como o próprio nome indica, tem seu valor derivado de um ou mais atributos ou entidades a ele relacionados.

De forma simples, podemos tomar como exemplo uma Nota Fiscal de venda que possui o produto vendido, seu preço unitário, a quantidade vendida e o preço total da nota. O atributo Preço Total é considerado um atributo derivado, pois seu valor pode ser determinado pela multiplicação de Valor Unitário e Quantidade.

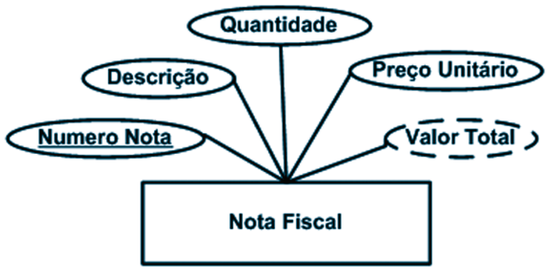


Figura: 14. Atributos derivados.

4.3 Relacionamentos

Até o momento identificamos os dados do nosso sistema de informações localizando os elementos (entidades) que o compõem e as características (atributos) que os identificam.

Entretanto, para representar o mundo real em um ambiente computacional, é necessário modelar as interações entre esses elementos. Em geral, essas interações descrevem a ocorrência de um fato, que pode ou não se repetir, de forma integral ou parcialmente.

Se voltarmos ao nosso caso de estudo, (a vídeolocadora) perceberemos que sempre existirá uma interação entre um cliente e um filme, no momento em que um cliente decide alugar um filme. Assim, torna-se necessário modelar os dados que caracterizam essa interação.

Porém, não podemos definir esses novos atributos como pertencentes à entidade Filmes ou à entidade Clientes, por se tratar de uma característica temporária.

Vejamos um exemplo: o cliente "José da Silva" alugou o filme "Tropa de Elite" na data de hoje, mas amanhã poderá alugar outro; nesse mesmo sentido, o filme "Tropa de Elite" foi alugado pelo cliente "José da Silva" hoje, mas poderá ser alugado por outro amanhã. Ou seja, as características que descrevem essa interação entre o cliente "José da Silva" e o filme "Tropa de Elite" são temporárias, não podendo pertencer a uma ou outra entidade.

Nota-se então, a necessidade de algo que represente uma associação no mundo real entre o elemento-pessoa "José da Silva" e o elemento--filme de título "Tropa de Elite". Esse novo elemento do nosso MER é chamado de conjunto de relacionamentos, pois ele descreve de que forma uma entidade se relaciona com a outra.

A Figura 14, abaixo, retrata a forma de representação de um relacionamento conforme descrito acima:

Figura 14: Relacionamento entre entidades.

Observando a Figura 14, podemos perceber que, da mesma forma que foi feito para as entidades, foi colocado um nome para o conjunto de relacionamentos explicitando a que o conjunto se refere. Para o nosso caso de estudo, utilizamos o nome Locações, pois se trata do conjunto das locações existentes entre os clientes e os filmes de nossa vídeolocadora.

Conceitualmente, um conjunto de relacionamentos que relaciona elementos de dois conjuntos de entidades é um conjunto de duplas dessas entidades. Assim dizemos que uma das duplas de Locações representa as entidades "José da Silva" e "Tropa de Elite", para a qual utilizaremos a notação textual (José da Silva, Tropa de Elite).

Características dos Relacionamentos

A utilização de relacionamentos em um DER tem como objetivo representar características importantes da interação entre as entidades que participam desse relacionamento. Vamos estudar duas características dos relacionamento a Multiplicidade de Relacionamentos ou Cardinalidade e os Atributos de Relacionamentos. Além disso, veremos um tipo especial de relacionamento:o Auto-relacionamento.

a) Multiplicidade de Relacionamentos ou Cardinalidade

A multiplicidade indica o número de duplas em que um determinado elemento de um dos conjuntos de entidades pode ocorrer em um relacionamento. Em alguns casos, essa característica deve ser respeitada para garantir a integridade dos dados.

Complicou???

Vamos ver exemplos práticos que fica mais fácil entender.

Na J&J Vídeo é comum que um cliente possa alugar mais de um filme, entretanto, aquele filme só pode estar alugado para um único cliente. Assim, são feitas anotações no DER para indicar essas restrições. A Figura 15, a seguir apresenta esse detalhamento.

Figura 15: Multiplicidade em relacionamentos

No caso, o 1 anotado na aresta que liga Locações e Clientes, indica que cada entidade de Filmes pode estar relacionada por meio de Locações com no máximo uma entidade de Clientes. Por outro lado, o N anotado na aresta que liga Locações e Filmes, indica que cada entidade de Clientes pode estar relacionada por meio de Locações com um número qualquer de entidades de Filmes.

Importante: Vale lembrar, que estamos nos referindo a uma única locação, obviamente um determinado filme pode ser alugado por outro cliente, porém será uma outra associação pertencente ao conjunto Locações.

A forma de ler essa multiplicidade depende do ponto de vista, ou seja, pode ser tanto de uma entidade quanto de outra. Vejamos, do ponto de vista da entidade Clientes, essa multiplicidade é de 1:N (diz-se de 1 para N). Do ponto de vista da entidade Filmes, diz-se de N:1 (N para1). Assim, "1 cliente aluga vários filmes" ou "Vários filmes são locados por 1 cliente".

Desta forma, a ocorrência de um filme poder estar alugado por dois clientes ao mesmo tempo demonstra um erro do nosso sistema. Por isso, podemos dizer que a integridade dos dados não foi preservada.

Uma multiplicidade com maior restrição de integridade é a multiplicidade 1:1. Neste caso, apenas uma única entidade pode aparecer em cada lado da relação.

Vejamos o exemplo da figura abaixo:

Figura 16: multiplicidade 1:1 em relacionamentos.

Esse diagrama descreve que uma entidade de Funcionários se relaciona por meio de Coordenações com uma entidade de Departamentos. Da mesma forma que uma entidade de Departamentos se relaciona por meio de Coordenações com uma entidade de Funcionários.

No mundo real, isso representa que um departamento tem um único funcionário que exerce a função de coordenador. Da mesma forma que um funcionário só pode coordenar um departamento de cada vez.

Já que existe uma restrição mais rígida, existe uma multiplicidade que permite uma associação mais livre entre entidades, é a multiplicidade N:N.

Vejamos o exemplo da Figura 17, a seguir.

Figura 17: Multiplicidade N:N em relacionamentos.

No mundo real, um ator pode participar em mais de um filme e um filme tem a participação de vários atores. Observa-se então que cada entidade de Atores se relaciona por meio de Participações com várias entidades de Filmes. Da mesma forma que, uma entidade de Filmes se relaciona por meio de Participações com várias entidades de Atores.

b) Atributos de Relacionamentos

Da mesma forma que as entidades, os relacionamentos também podem ter atributos que identifiquem relações específicas entre as duplas das entidades que relacionam. Eles são representados por elipses contendo seus nomes, da mesma forma que foi feito para um atributo de entidade, conforme mostra a Figura 18.

Conforme citado anteriormente, um determinado filme está locado para um cliente em um dia e no outro está locado para outro. Para poder diferenciar em que momento os relacionamentos ocorreram, é preciso utilizar alguns atributos no relacionamento Locações.

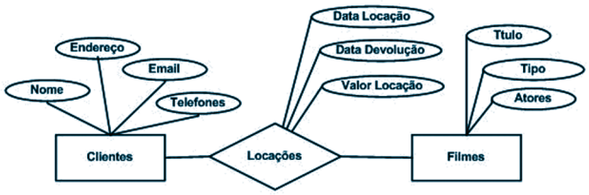


Figura 18: Atributos de Relacionamentos

c) Auto-relacionamentos

Existem ainda algumas situações específicas onde um elemento de uma entidade se relaciona com outro elemento da mesma entidade. A essa situação chamamos de auto-relacionamento. Um exemplo clássico é o relacionamento Chefia entre a entidade dois elementos da entidade Funcionários.

A representação desse tipo de relacionamento é mostrada na Figura 19, abaixo.

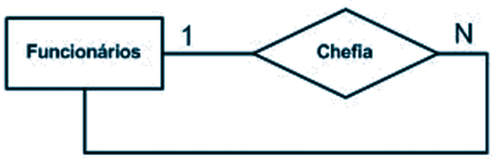


Figura 19: Auto-relacionamento

Nesse caso, a leitura da Figura 19 deve ser feita de seguinte forma: "1 Funcionário chefia 1 ou mais Funcionários; Um Funcionário é chefiado por apenas 1 Funcionário".

4.4 – Representação de modelos E-R

Conforme apresentado anteriormente, não existe um acordo sobre qual a melhor forma de representar graficamente um elemento do modelo E-R. Neste capítulo apresentamos uma forma de notação, que pode diferir um pouco de outras literaturas sobre o assunto. Porém, os conceitos são os mesmos.

Talvez uma das diferenças que mereça consideração é que em alguns relacionamentos muito específicos, é definida uma limitação para a quantidade de entidades de um conjunto que pode participar de um relacionamento com uma entidade de outro conjunto. Essa situação é chamada de Restrição Estrutural(min,max).

Vejamos um exemplo: Em um campeonato de corrida de carros, a cada prova podem participar no máximo 20 veículos. Assim, para que uma prova seja realizada, ela deve ter no mínimo 1 e no máximo 20 pilotos inscritos, cada um com seu carro. O campeonato é composto por 15 provas, das quais um piloto pode participar de no máximo 12 provas. Ou seja, um piloto está limitado ao número máximo de 12 corridas.

Essa situação está representada pelo diagrama E-R da Figura 20, a seguir.

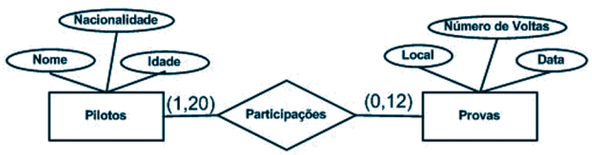
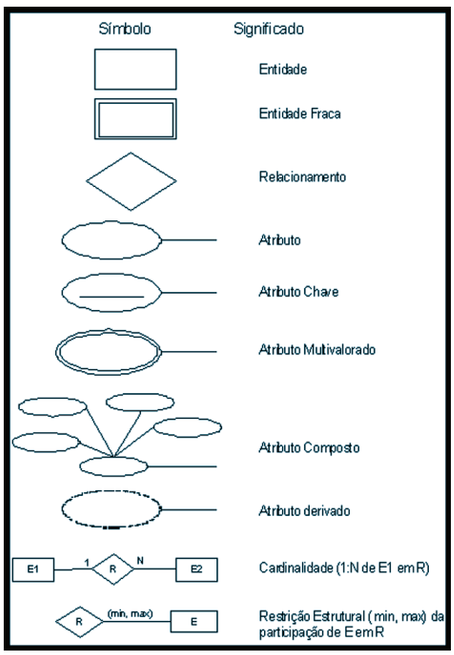


Figura 20: Restrição Estritural(min, max).

Resumo



Atividades de aprendizagem

1 - Partindo do seu conhecimento sobre a locação de um filme em uma locadora, elabore um DER o mais detalhado possível sobre o domínio da J&J Vídeo;

CAPÍTULO

5

O MODELO RELACIONAL

Objetivo

No capítulo 5 são apresentos conceitos utilizados no modelo relacional e a ensinar também, como partir de um modelo E-R e derivar para um modelo Relacional, capaz de ser criado em um ambiente computacional.

Os primeiros sistemas de bancos de dados foram criados a partir dos conceitos elaborados pelos modelos em rede e hierárquico, visto que esses modelos se aproximavam mais do nível físico, ou seja, da forma como os dados eram armazenados.

Mesmo propondo uma representação distante da organização física dos dados, "o modelo relacional estabeleceu-se como o primeiro modelo de dados para aplicações comerciais" (Silberschatz, 2004), em virtude da facilidade de implementação de seus conceitos e do desempenho eficiente na recuperação dos dados.

Obviamente isso fez surgir um certo número de fornecedores de SGBDs que se especializaram em implementar bancos de dados que utilizam os conceitos derivados desse modelo. Esses bancos são chamados de Bancos de Dados Relacionais.

5.1 Relações, Atributos e Tuplas

Elmasri (Elmasri, 2006) apresenta de forma simplificada que "o modelo relacional representa o banco de dados como uma coleção de relações", ou simplesmente tabelas. Podemos então visualizar cada entidade de um DER como sendo uma tabela de nosso banco de dados relacional.

Mas como representar os relacionamentos?

O modelo relacional não possui um elemento para representar um relacionamento, porém, se nos lembrarmos da definição de relacionamento, veremos que ele representa uma dupla de entidades do conjunto que ele relaciona. Assim, cada relacionamento será uma relação do modelo relacional, identificada por um nome, para facilitar a manipulação dos dados.

Em geral, uma Tabela de um banco de dados relacional:

- Contém os dados de um conjunto de entidades do mesmo tipo ou de um conjunto de relacionamentos;

- Apresenta em cada linha os dados de uma única entidade;

• Caso a tabela armazena dados de um conjunto de relacionamentos, ela contém as duplas das entidades que relaciona e, conforme o caso, alguns atributos do relacionamento;

• Os atributos identificam cada uma das colunas da relação e cada linha, chamada de tupla, contém os dados de uma entidade ou de um relacionamento do mundo real.

Assim, para representar a entidade Clientes, uma tabela receberia esse nome para facilitar a compreensão de seu conteúdo e sua manipulação. Bem como, cada coluna receberia o nome do atributo que ela representa. No caso, podemos utilizar Nome, Endereço, Email e Telefones como nome das colunas da tabela Clientes.

Nome	Endereço	Email	Telefones
João da Silva	7 de Setembro, 12 - Ipiranga - Joinville/SC	jsilva@email.com	32323232
Maria da Graça	Itu, 456 - Centro – Blumenau/SC	mg456@email.com	98987770
José Pereira	Margaridas, 33 - Fazendas – Irati/PR	jpereira@email.com	

MUITA ATENÇÂO! IMPORTANTE!

Conceito de Domínio: é um conceito importante existente no modelo relacional. Um domínio pode ser entendido como um conjunto de valores atômicos (indivisíveis).

Exemplos de Domínio: Tomemos um exemplo definido pela matemática: o conjunto dos números inteiros. Todos os valores existentes neste domínio são números inteiros que não podem ser decompostos em outros que não sejam números inteiros.

Outro exemplo de domínio seria os meses do ano. Esse conceito define uma regra fundamental dos bancos de dados relacionais que determina que o valor em uma coluna deve sempre pertencer a um domínio. Isso auxilia a garantir a integridade dos dados.

5.2 Atributo Chave de uma Relação

Como as relações representam um conjunto de entidades, o modelo relacional deve identificar uma forma de escolher ou identificar uma entidade específica do conjunto.

Na seção 3.2 vimos o conceito de atributo determinante. A idéia é similar, porém no modelo relacional, isso pode ser definido por um ou mais atributos e leva o nome de Atributo chave da relação.

No modelo E-R da J&J Vídeo, colocamos um atributo chamado numero de filme para identificar cada filme disponível na videolocadora. Fizemos isso porque poderia haver mais de uma cópia do mesmo filme e o atributo Título não poderia identificar cada cópia. No modelo relacional não seria diferente e neste caso, o atributo numero de filme é o atributo chave da tabela Filmes.
Lembre-se, o atributo chave (ou chave) é uma forma de identificar unicamente uma entidade (ou um registro).

5.3 – Restrições do Modelo Relacional

Se observarmos atentamente o mundo real, no que se refere aos dados e informações com os quais interagimos, veremos que existem certas regras das quais não podemos fugir. Por exemplo, imagine a reação de uma pessoa que lhe pergunta as horas e você responde: "Branco de bolinhas azuis". Afinal de contas, ela espera uma resposta em horas e minutos utilizando valores que pertencem a esse domínio.

Ao elaborarmos um modelo relacional do nosso mini-mundo, é preciso explicitar essas regras de alguma forma, ou seja, precisamos evidenciar as restrições nos dados para cada atributo. Para tal, existem no modelo relacional três tipos de restrições: Restrições de domínio, Restrição de chave, Restrição de valor nulo.

a) Restrições de Domínio

Essa restrição é simples de entender, o exemplo das horas, acima, descreve bem essa restrição. Um determinado dado deve pertencer a um domínio para ser válido. Assim, ao descrevermos o modelo relacional, devemos explicitar o tipo de dado esperado para aquela coluna da nossa tabela. Exemplos

ATRIBUTO	DOMÍNIO
Nome Cliente	Alfanumérico
Data de Nascimento	Data
Número de filhos	Numérico – inteiro
Ano de Lançamento	Numérico – entre 1900 e 2010
Valor da Locação	Numérico com duas casas decimais

Essa restrição é importantíssima, pois o modelo deve informar o que se espera para um atributo. Por exemplo, caso o domínio do atributo seja uma data válida, algo como: “ontem”, “semana passada” ou mesmo “31 de fevereiro”, apesar de estar relacionado a datas, deve ser evitado.

b) Restrições de Chave

Em algumas situações específicas, os dados são coerentes com o tipo esperado, porém seu significado rompe, de alguma forma, com a semântica do mundo real modelado.

Por exemplo, por mais que a placa de um automóvel seja a combinação de 3 letras e 4 números, não podemos permitir que existam dois veículos com a mesma placa. Esta restrição é chamada de restrição de chave primária. Veja mais detalhes logo a seguir.

De forma similar, se a sigla dos estados brasileiros é formada por duas letras, não se pode assumir que “TT” é um valor válido para a sigla do estado de Tocantins (TO), para evitar esse problema, cria-se uma tabela que contém a relação de valores possíveis (definindo um novo domínio) e indica-se por meio de uma restrição de chave estrangeira que nesse atributo só serão aceitos valores que constem na tabela referenciada.

Restrições de Chave Primária

Conforme visto anteriormente, um atributo chave identifica uma única entidade do conjunto de entidades. Esse atributo é chamado de chave primária da relação. Uma restrição de chave primária determina que, ao se definir um atributo como a chave primária de uma relação, os valores dos atributos que correspondem a essa chave não podem ser repetidos para duas entidades distintas.

É o caso das placas dos automóveis. Se definirmos o atributo Placa como sendo a chave primária, não poderá existir nos dados, duas entidades com o mesmo valor para esse atributo.

Número de Filmes, conforme visto anteriormente, também é um exemplo de uma chave primária para a relação Filmes.

Você consegue imaginar uma relação e identificar qual seria sua chave primária?
Anote: _____
_____

Restrições de Chave Estrangeira

A restrição de chave estrangeira é, de certa forma, similar a restrição de domínio. Ela também determina que os valores associados a um determinado atributo devam pertencer a um domínio. Entretanto, esse domínio corresponde ao atributo chave de outra entidade.

Vamos a um exemplo prático:

Anteriormente colocamos a situação onde foi sugerida a utilização incorreta da sigla TT para o estado do Tocantins (sigla TO). Digamos que esse atributo pertence a entidade Clientes. Como evitar que um valor assim seja considerado válido?

Para garantir que os dados desse atributo sejam valores válidos, devemos ter uma entidade Unidades da Federação composta pelos atributos Sigla e Nome. Onde o atributo Sigla é a chave primária da relação. Nessa tabela, são cadastrados os valores válidos para Sigla e para Nome das UFs.

Ao definirmos a entidade Clientes, determinamos que o atributo Estado tem uma restrição de chave estrangeira com a entidade Unidades da Federação. Assim, na entidade Clientes só serão permitidos valores para o atributo Estado que existam na chave primária (atributo Sigla) da entidade Unidades da Federação.

c) Restrições de valores Nulos (Null)

Essa restrição impede que sejam atribuídos valores nulos para um atributo de uma relação. Por exemplo, para uma pessoa poder abrir uma conta em um banco é preciso ter um número de inscrição no CPF. Assim, ao modelarmos um sistema desse tipo devemos definir que o atributo Numero CPF possui uma restrição de valores nulo, ou seja, ele deve ser informado.

5.4 Representação de um Modelo Relacional

DICA

Objetos de um banco de dados são todos os elementos estruturais que o compões: tabelas, atributos e restrições. Além de outros objetos mais complexos tais como os procedimentos armazenados (stored procedures) e triggers (ainda não criaram um termo melhor em português).

Agora que conhecemos os componentes e as regras para criação de um modelo relacional, vamos conhecer os elementos gráficos que são usados para representá-los.

Vamos utilizar aqui uma notação chamada Engenharia da Informação (IE - Information Engeneering), conhecida na comunidade dos programadores como notação pé-de-galinha.

a) Tabela

No modelo relacional cada tabela é representada por um retângulo identificado com um nome. Não pode haver duas relações com o mesmo nome em um mesmo banco de dados. Esse retângulo é dividido em duas partes. Na parte superior será registrado o atributo determinante (ou mais de um se for o caso) e na parte de baixo, os demais atributos. Veja a Figura 21, a seguir.
Em geral alguns bancos de dados possuem algumas regras para a formação dos nomes dos objetos do banco de dados. Sempre que for projetar um banco de dados, observe as regras do SGBD a ser utilizado.

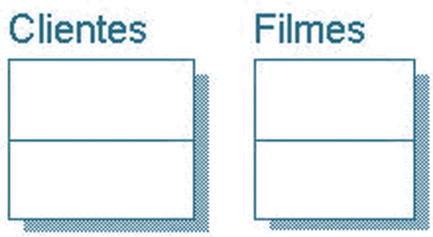


Figura 21: Tabelas no modelo relacional.

b) Atributo ou coluna

Cada atributo tem registrado o seu nome e as suas restrições. Da mesma forma que as tabelas, os atributos também possuem regras para a formação de nomes. Não há limites de tamanho na maioria dos SGBDs atuais. Caracteres especiais não são aceitos em nomes de campo, tais como: “,”, “:”, “;”, “/”, aspas, parênteses, “%”, “$”, “#”, etc. Não se deve usar caracteres acentuados por questões de compatibilidade entre SGBD.

IMPORTANTE: O fato de não se utilizar caracteres acentuados nos nomes dos atributos, não significa que o dado armazenado naquela coluna não possa ter caracteres acentuados. Por exemplo, o atributo Funcionario, não deve ser acentuado, porém, caso o dado a ser armazenado seja "Mário Sérgio Carreirão" não haverá problema.

O caractere de espaço não é aceito na maioria deles. Assim, sugerimos a utilização de substantivos no singular, sem espaço entre eles e sem utilizar preposições. Glup!!!

Ok.... Vamos a um exemplo: temos na entidade Filmes o atributo "numero de filme", quando vamos transformá-lo em uma coluna de uma tabela, suprimimos a preposição "de", juntamos as duas palavras e colocamos as primeiras letras de cada uma em maiúsculo, assim: "NumeroFilme".

E caso o nome fique muito grande, você também abreviá-lo, porém sem utilizar o ponto (também é proibido). Exemplo: Para a data de nascimento de um cliente, podemos utilizar:
"DtNascimentoCli" ou "DataNascCli".

Fácil? Então tente alguns:

Nome do Atributo	Nome da Coluna
Ano de lançamento	
Data de devolução	
Código do tipo de filme	
Nome do chefe de departamento	
Sigla da Unidade da Federação	

A figura abaixo mostra como ficam os atributos dentro das tabelas no modelo relacional.

Figura 22: Atributos de uma tabela.

Você ainda lembra quais são as restrições que um atributo pode ter?

c) Restrições

A primeira restrição a ser anotada é a restrição de domínio. Essa restrição indica que tipo de dado se espera para aquele atributo. Veja o Quadro 4 sobre os domínios e os bancos de dados relacionais.

A segunda restrição que vimos foi a restrição de chave primária. Essa restrição se aplica apenas ao atributo determinante.

Depois temos a restrição de chave estrangeira. Essa restrição deve ser aplicada aos atributos cujos valores estejam registrados em outra relação.

E por último a restrição contra valores nulos.

Agora, basta colocá-las ao lado dos atributos, conforme mostra a figura, a seguir.

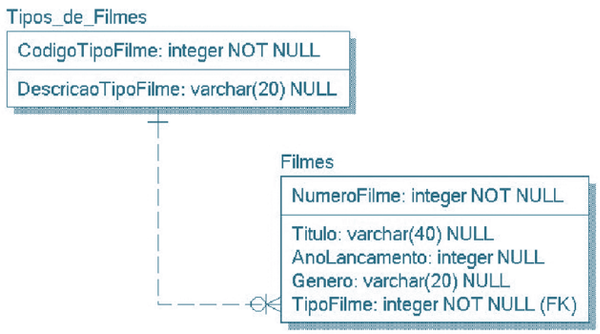


Figura 23: Restrições em atributos.

PARA REFLETIR

Você viu que a tabela Tipos_de_Filmes tem algo diferente do que vimos até agora??? Percebeu?

Isso mesmo, o caractere "_" Pode ser utilizado para juntar nomes de tabelas e atributos. Nesse caso, extrapolamos e colocamos também a preposição "de". Mas isso foi apenas para ficar mais claro a explicação. Na prática, você deve usar nomes simples, no singular, e evitar nomes muito longos.

Vamos olhar atentamente para a Figura 23. Colocamos uma nova informação neste modelo. A tabela Filmes deve ter a informação do tipo do filme. Entretanto, para evitar que o usuário digitasse algum dado errado ou fora de um certo conjunto de tipos, criamos a tabela Tipos_de_Filmes.

Essa tabela conterá todas as descrições possíveis de tipos de filmes e um código identificador para cada uma delas. Assim, para garantir que só sejam informados valores válidos para o campo TipoFilme da tabela

Filmes, armazenaremos nele um código, que corresponde a um valor cadastrado na tabela Tipos_de_Filmes.
Para definir essa restrição, criamos uma restrição de chave estrangeira no atributo TipoFilme da tabela.

QUADRO 4

Domínios e tipos de dados mais comuns:
1) Numéricos:
- Smallint - Armazena valores numéricos, em dois bytes binários, compreendidos entre o intervalo -32768 a +32767.
- Integer - Armazena valores numéricos, em quatro bytes binários, compreendidos entre o intervalo -2147483648 a +2147483647
- Decimal(n,m) - Armazena valores numéricos com no máximo n dígitos. Nesta opção deve ser definida a quantidade de casas decimais (m) existentes no campo. A quantidade de inteiros é definida por n – m ;

2) Alfanuméricos:
- Varchar(n) - campo alfanumérico de até n caracteres, onde n deve ser menor ou igual a:
    - 8000 para MS-SQL Server;
    - 10485760 para Postgres
- Char(n) - campo alfanumérico de n caracteres, onde n deve ser menor ou igual a:
    - 8000 para MS-SQL Server;
    - 10485760 para Postgres

3) Datas e horas:
- Date - campo que irá armazenar datas.
- Time - campo que para o armazenamento de valores de tempo (horas, minutos, segundos e milésimos de segundos).
- Observação: Alguns tipos de dados podem existir em um SGBD e não existir para outro. Exemplo: MS-SQL Server os tipos de dados DATE e TIME não existem. Entretanto existe o tipo DATETIME que atende aos 2 tipos.

E o tal “pé-de-galinha”?

Conforme indicado, a notação que foi apresentada é chamada de notação de pé-de-galinha por causa das linhas utilizadas para definir as restrições de chave estrangeira. Além de indicar a dependência entre as tabelas elas definem a número de registros possíveis de serem as-

sociados entre elas. Assim, sabemos características mais detalhadas dessa restrição. Vejamos outro exemplo.

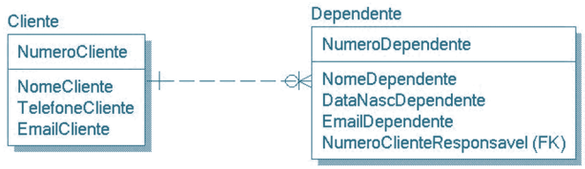


Figura 24: Detalhes de restrições.

A Figura 24 mostra as tabelas Cliente, que contém os dados dos clientes, e Dependente, que contém os dados básicos dos dependentes de cada cliente cadastrado. Existe então uma restrição de chave estrangeira para o atributo NumeroClienteResponsavel. A linha que indica a dependência nos informa que cada dependente pode ter apenas um cliente como seu responsável. Por isso, ao lado da tabela Cliente, aparece um pequeno traço vertical.

Por outro lado, um cliente pode ser responsável por zero, um ou vários dependentes. Isso está representado pelo pequeno círculo (zero), pelo traço vertical (um) e pelos 2 traços em diagonal (vários), todos dispostos ao lado tabela Dependente.

Agora observe bem a forma da linha, o traço vertical e os 2 traços em diagonal. Não parece um desenho de um pé-de-galinha? Ninguém pode dizer que o pessoal da informática não tem senso de humor.

5.5 Transformação de um modelo E-R para um Modelo Relacional

Conforme apresentado no início desse capítulo, é preciso entender os conceitos do modelo relacional para saber como derivar um modelo E-R para um modelo Relacional. Alguns dos passos desse processo já foram ilustrados anteriormente, porém vamos revisá-los para não deixar dúvidas.

Podemos dividir esse processo de conversão em 4 passos:

i. Representação das entidades por uma estrutura tabular;
ii. Representação dos relacionamentos por uma estrutura tabular;
iii. Combinação de tabelas e eliminação de tabelas redundantes;
iv. Representação de atributos multivalorados por tabelas;

VAMOS VER NA PRÁTICA COMO ISSO OCORRE PARA FIXAR ESSES PASSOS?

Para tal, vamos retomar o nosso caso de estudo: o sistema da J&J Vídeo.

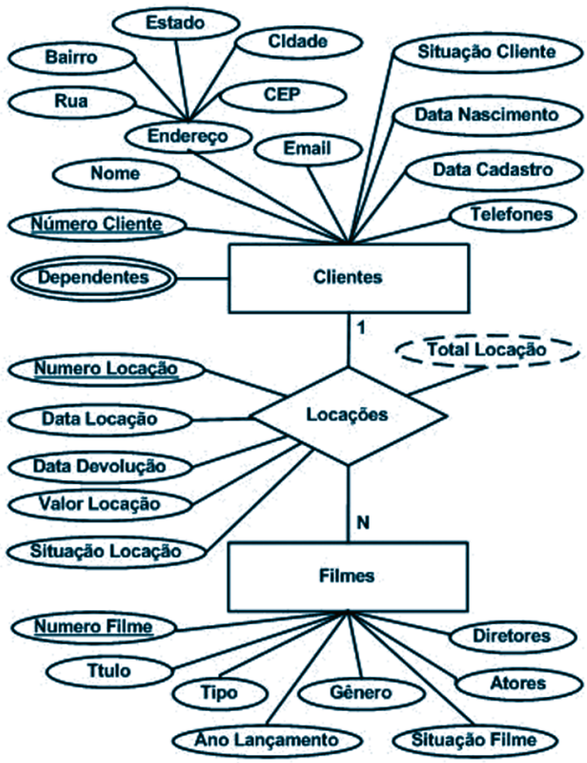


Figura 25: O Diagrama E-R da vídeolocadora.

Você ainda lembra o que significa cada elemento desse diagrama?

Resumo

Elmasri (Elmasri, 2006) apresenta de forma simplificada que "o modelo relacional representa o banco de dados como uma coleção de relações", ou simplesmente tabelas.

Elementos do modelo relacional

Tabela	Representa as entidades e os relacionamentos do modelo E-R
Campo, coluna ou atributo	Representa um atributo de uma entidade
Restrições	Representa as regras para os dados que serão inseridos nas tabelas. Pode ser: - de domínio; - de chave primária; - de chave estrangeira; - de valor nulo;

Tipos de Restrição	
- de domínio;	Garante que o dado inserido na coluna tem o tipo de dados esperado ;
- de chave primária;	Garante que o atributo determinante de uma entidade não possa ter seu valor repetido em outro registro da mesma tabela;
- de chave estrangeira;	Garante que os dados inseridos em uma coluna da tabela tenham valores definidos por m conjunto (outra tabela);
- de valor nulo;	Garante que o valor dos dados daquele campo nunca seja nulo.

Para projetar um modelo relacional devemos partir do modelo E-R. Sendo que o processo de conversão pode ser dividido em 4 passos:

i. Representação das entidades por uma estrutura tabular;
ii. Representação dos relacionamentos por uma estrutura tabular;
iii. Combinação de tabelas e eliminação de tabelas redundantes;
iv. Representação de atributos multivalorados por tabelas;

Atividades de aprendizagem

1 - Elaborar o modelo relacional a partir do modelo detalhado da J&J Vídeo, disponibilizado no ambiente virtual.

CAPÍTULO

6

LINGUAGENS DE CONSULTA

Objetivo

O capitulo 6 introduz aspectos conceituais e práticos de Linguagens de Consulta nas perspectivas de definição, consulta, manipulação, controle de transação, controle de acesso e segurança dos dados.

Com a disseminação do uso dos bancos de dados, era necessário fornecer uma linguagem mais acessível aos usuários. Surgiu então, no início dos anos 70, a linguagem SEQUEL que no português significa de "Linguagem de Consulta Estruturada em Inglês" (Structured English Query Language). Posteriormente reduzida para "Linguagem de Consulta Estruturada" (Structured Query Language) e atualmente conhecida apenas por SQL, especialmente desenvolvida para ser utilizada com bancos de dados relacionais.

Apesar de ser definida inicialmente como uma linguagem de consulta, veremos nesse capítulo muitas das características originais do SQL foram inspiradas na álgebra relacional.

GLOSSÁRIO

Que a linguagem SQL, em inglês, é comumente pronunciada "síquel" ao invés de "és-kiú-él", letra a letra. Isso se deve ao fato de inicialmente ela ser SEQUEL. No entanto, em português, a pronúncia mais corrente é a letra a letra: "ésse-quê-éle".

6.1 – A SQL

A SQL se diferencia de outras linguagens de consulta a banco de dados, pois uma consulta SQL especifica qual o resultado esperado e não o caminho para chegar a ele. Por exemplo, em SQL eu solicito ao banco de dados para "mostrar todas as locações que ocorreram na data de ontem".

Em outras linguagens eu teria que solicitar ao banco de dados para "ler as datas das locações e mostrar aquelas cujas datas sejam iguais a data de ontem". Parece simples quando existe apenas uma tabela para pesquisar. Entretanto, se pensarmos em algo mais elaborado, mesmo em SQL será mais simples de construir a consulta do que elaborar um algoritmo para obter tal informação.

Você duvida? Então vamos lá!

Baseado em nosso caso de estudo: a vídeolocadora, eu gostaria de saber quais foram os filmes que o ator o Tom Cruise participou e foram considerados como os mais assistidos no último ano. Tente elaborar um algoritmo para recuperar essa informação. Utilize o modelo relacional criado a partir do modelo E-R da Figura 20.

Anote aqui a solução elaborada:

SAIBA MAIS

Embora a SQL tenha sido originalmente criada pela IBM, rapidamente surgiram vários "dialetos" desenvolvidos por outros produtores. Essa expansão levou à necessidade de ser criado e adaptado um padrão para a linguagem.

Esta tarefa foi realizada pela American National Standards Institute (ANSI) em 1986 e ISO em 1987. A SQL foi revista em 1992(a esta versão foi dado o nome de SQL-92), foi revista em 1999 (SQL3) e novamente em 2003 (SQL:2003).

A SQL, embora padronizado pela ANSI e ISO, possui muitas variações e extensões produzidas pelos diferentes fabricantes de SGBDs. Em geral, a linguagem pode ser migrada de plataforma para plataforma sem grandes alterações estruturais.

Atualmente a maioria dos SGBDS, permite ainda a utilização de algum código procedural embutido em rotinas do banco de dados, podendo interagir com o mesmo. Por exemplo, o Oracle e outros incluem Java na base de dados, enquanto o PostgreSQL permite que funções sejam escritas em Perl, Tcl, ou C, entre outras linguagens.

6.2 – Os comandos em SQL

Apesar de ter sido criada como uma linguagem de consulta, a SQL oferece um conjunto de comandos responsáveis pela definição das tabelas, atualização dos dados, definição de restrições, controle de acesso e otimização das consultas em um SGBD.

Esses comandos podem ser subdivididos em cinco grupos:
- Comandos DDL (Data Definition Language) - Conjunto de comandos responsáveis pela criação, alteração e deleção da estrutura das tabelas e índices de um sistema.
- Comandos DQL (Data Query Language) – Conjunto de comandos para a elaboração de consultas aos dados de um banco de dados.
- Comandos DML (Data Manipulation Language) - Conjunto de comandos responsáveis pela manipulação dos dados armazenados em um banco de dados: inclusão, atualização e exclusão.
- Comandos DCL (Data Control Language) - Conjunto de comandos responsáveis criação dos usuários e definição das permissões de cada um no banco de dados.
- Comandos DTL (Data Transaction Language) – Conjunto de comandos para a inicialização e finalização de transações em um banco de dados.

Apesar de serem específicos, para cada uma das operações que realizam, todos esses comandos são considerados "comandos SQL".

Vamos ver então, os mais importantes? São eles:
DDL/ Linguagem de definição de dados,
DQL/ Linguagem de consulta de dados,
DML/ Linguagem de manipulação de dados,
DCL/ Linguagem de controle de dados
DTL/ Linguagem de transação de dados

a) DDL – Linguagem de definição de dados

Conforme citado inicialmente, esse conjunto de comandos se destina a criação das estruturas de um banco de dados. Todo o processo de modelar o mundo real em um diagrama E-R e depois convertê-lo em um modelo relacional, tem por objetivo a construção das estruturas para o armazenamento dos dados.

Assim, para criar um banco de dados devemos tê-lo modelado antes para que ele seja capaz de atender as necessidades do sistema que estamos produzindo.

O primeiro passo na criação de um banco de dados é a criação do banco de dados propriamente dito. Todo SGBD, como o próprio nome diz, é um sistema, onde é preciso um usuário autorizado a utilizar os recursos que ele provê. Um SGBD pode ter vários usuários. Veja quadro N, na página XX.

Para poder criar um banco de dados dentro de um ambiente de um SGBD, precisamos ter esse direito definido no sistema. Esse direito é conhecido como perfil. E a cada usuário é atribuído um perfil. No caso, precisamos de um perfil com direito de criação de banco de dados (database creator).

Criando um banco de dados

Um vez que tenhamos o perfil necessário e estejamos conectados ao servidor, pode mos então criar nosso banco de dados. Em cada banco de dados que criamos recebemos o perfil de dono do "banco de dados" (database owner).

Em geral, apenas o dono do banco de dados, ou alguém autorizado por ele, pode criar as estruturas internas do banco. Para criar um banco de dados com a linguagem SQL, utilizamos o comando `Create Database`.

A sintaxe varia um pouco de um SGBD para outro, em linhas gerais, precisamos explicitar o nome do banco de dados. Assim resumidamente nosso comando ficaria assim:

```
CREATE DATABASE <nome-do-banco>
```

Alguns SGBDs necessitam de parâmetros extra para criar um banco de dados. Porém, para nossa sorte, caso esses parâmetros não forem informados eles possuem valores padronizados (default) para esses parâmetros e o comando funciona perfeitamente.

Para executar um comando SQL temos que ter um programa que permita sua execução. Existem vários programas utilizados para isso. Em geral, os fornecedores de SGBDs fornecem também um programa para isso. Por exemplos temos:

SGBD	Nome do programa
Oracle	SQLplus
Microsoft SQL Server	SQL Query Analyzer
Postgresql	PGAdmin
MySQL	MySQLAdmin

Criando tabelas

A criação de tabelas consiste na definição da estrutura necessária para implementar no banco de dados os objetos definidos no modelo relacional, ou seja, as tabelas (relações), as colunas (atributos ou campos) e as restrições de chave primária e estrangeiras existentes.

Para cada coluna, além do nome, é preciso indicar o tipo de dados esperado (restrição de domínio) e as restrições de valor nulo. O comando básico para essa tarefa é o `CREATE TABLE`.

Sua sintaxe é um tanto extensa, vamos nos ater aos principais elementos de uma tabela.

```
CREATE TABLE <nome-tabela>
(<nome-coluna>  <tipo-do-dado>  [NOT NULL]
                                | [NOT NULL WITH DEFAULT]
CONSTRAINT  <Nome Constraint>
        PRIMARY KEY (nome-coluna-chave)
CONSTRAINT  <nome_constraint>
        FOREIGN KEY (nome-coluna-chave-estrangeira) REFERENCES
                (nome-tabela-pai) ON DELETE [RESTRICT]
                                            [CASCADE]
                                            [SET NULL]
```

onde:

1. nome-tabela - Representa o nome da tabela que será criada;

2. nome-coluna - Representa o nome da coluna que será criada. A definição das colunas de uma tabela é feita relacionando-as uma após a outra, separando-as por uma vírgula;

3. tipo-do-dado - Cláusula que define o tipo e tamanho dos campos definidos para a tabela. Os tipos de dados mais comuns foram apresentados no QUADRO 4.

4. NOT NULL - Exige o preenchimento do campo, ou seja, no momento da inclusão é obrigatório que possua um conteúdo.

5. NOT NULL WITH DEFAULT - Preenche o campo com valores pré-definidos, de acordo com o tipo do campo, caso não seja especificado o seu conteúdo no momento da inclusão do registro. Os valores pré-definidos são:

- Campos numéricos - Valor zero.
- Campos alfanuméricos - Caracter branco.
- Campo formato Date - Data corrente.
- Campo formato Time - Horário no momento da operação.

6. CONSTRAINT <nome_constraint> Identifica, por meio de um nome, a restrição (constraint) que será associada a tabela.

7. PRIMARY KEY (nome-coluna-chave) - Definir para o banco de dados a coluna que será a chave primária da tabela. Caso ela tenha mais de um coluna como chave, elas deverão ser relacionadas entre os parênteses.

8. FOREIGN KEY (nome-coluna-chave-estrangeira) REFERENCES (nome--tabela-pai) - Definir para o banco de dados as colunas que são chaves estrangeiras, ou seja, os campos que são chaves primárias de outras tabelas. Na opção REFERENCES deve ser especificado a tabela na qual a coluna é a chave primária.

9. ON DELETE - Esta opção especifica os procedimentos que devem ser feitos pelo SGBD quando houver uma exclusão de um registro na tabela pai quando existe um registro correspondente nas tabelas filhas. As opções disponíveis são:
- RESTRICT - Opção default. Esta opção não permite a exclusão na tabela pai de um registro cuja chave primária exista em alguma tabela filha.
- CASCADE - Esta opção realiza a exclusão em todas as tabelas filhas que possua o valor da chave que será excluída na tabela pai.
- SET NULL - Esta opção atribui o valor NULO nas colunas das tabelas filhas que contenha o valor da chave que será excluída na tabela pai.

Como exemplo, tomemos a criação das tabelas Estado e Cliente.

```
CREATE TABLE Estado
(
      SiglaUF char(2) NOT NULL,
      NomeEstado varchar(20) not null,

      CONSTRAINT Estado_PK
            PRIMARY KEY (SiglaUF)
)

CREATE TABLE Cliente
(
      NumeroCliente Integer NOT NULL,
      NomeCliente varchar(70) NOT NULL,
      DataNascimento DATE ,
      RuaEndereco  varchar(80) NOT NULL,
      NumeroEndereco integer,
      Bairro varchar(30),
      SiglaUF char(2) ,
      Cidade varchar(30),
      CONSTRAINT Cliente_PK
            PRIMARY KEY (NumeroCliente) ,
      CONSTRAINT FK_ClienteSiglaUF
            FOREIGN KEY (SiglaUF) REFERENCES
                  Estado ON DELETE RESTRICT
)
```

É preciso criar a tabela Estado contendo as siglas dos Estados brasileiros (UF - Unidades da Federação), para depois criar a tabela Cliente, pois ela tem um restrição de chave estrangeira para a tabela Estado.

Para a criação das tabelas é preciso especificar o tipo de dado que será guardado naquela coluna da tabela. Para tal, existem alguns tipos de dados comuns a maioria dos bancos de dados compatíveis com o padrão SQL. O QUADRO 4, apresenta exemplos dos principais tipos de dados.

Entretanto é recomendável que você consulte o manual do fornecedor do banco de dados que vai utilizar para saber quais são os tipos compatíveis, quais os novos tipos de dados e quais estão obsoletos.

Alterando tabelas existentes

Muitas vezes, depois de criarmos uma tabela, mesmo que o projeto do banco de dados tenha sido cuidadosamente elaborado, é preciso alterar algum item da estrutura da tabela. Para isso, existe o comando `ALTER TABLE`. Ou seja, ele permite alterar a estrutura de uma tabela acrescentando, retirando e alterando nomes, mudando o formato das colunas e a integridade referencial definidas.

Sua sintaxe está descrita a seguir.

```
ALTER TABLE <nome-tabela>
        DROP <nome-coluna>
        ADD  <nome-coluna> <tipo-do-dado> [NOT NULL]
                                [NOT NULL WITH DEFAULT]
        RENAME <nome-coluna> <novo-nome-coluna>
        RENAME TABLE <novo-nome-tabela>
        MODIFY <nome-coluna> <tipo-do-dado> [NULL]
                                [NOT NULL]
                                [NOT NULL WITH DEFAULT]
        ADD CONSTRAINT <nome_constraint> PRIMARY KEY <nome-coluna>
        DROP PRIMARY KEY <nome-coluna>
        ADD FOREIGN KEY (nome-coluna-chave-estrangeira) REFERENCES
                (nome-tabela-pai) ON DELETE [RESTRICT]
                                            [CASCADE]
                                            [SET NULL]
        DROP FOREIGN KEY (nome-coluna-chave-estrangeira) REFERENCES
                (nome-tabela-pai)
```

onde:

1. nome-tabela - Representa o nome da tabela que será alterada.

2. nome-coluna - Representa o nome da coluna que será criada.

3. tipo-do-dado - Cláusula que define o tipo e tamanho dos campos definidos para a tabela.

4. DROP <nome-coluna> - Realiza a retirada da coluna especificada na estrutura da tabela.

5. ADD <nome-coluna> <tipo-do-dado> - Realiza a inclusão da coluna especificada na estrutura da tabela. Na coluna correspondente a este campo nos registros já existentes será preenchido o valor NULL (Nulo). As definições NOT NULL e NOT NULL WITH DEFAULT são semelhantes à do comando CREATE TABLE.

6. RENAME <nome-coluna> <novo-nome-coluna> - Realiza a troca do nome da coluna especificada.

7. RENAME TABLE <novo-nome-tabela> - Realiza a troca do nome da tabela especificada.

8. MODIFY <nome-coluna> <tipo-do-dado> - Permite a alteração na característica da coluna especificada.
Opções:
Além das existentes na opção ADD (NOT NULL e NOT NULL WITH DEFAULT), temos a opção NULL que altera a característica do campo passando a permitir o preenchimento com o valor Nulo.

9. ADD PRIMARY KEY <nome-coluna> - Esta opção é utilizada quando é acrescido um novo campo como chave primária da tabela.

10. DROP PRIMARY KEY <nome-coluna> - Esta opção é utilizada quando é retirado um campo como chave primária da tabela.

11. ADD FOREIGN KEY <nome-coluna> - Esta opção é utilizada quando é acrescido um novo campo sendo ele uma chave estrangeira.

12. DROP FOREIGN KEY <nome-coluna> - Esta opção é utilizada quando é retirado uma chave estrangeira da estrutura da tabela.

Vejamos alguns exemplos:

```
ALTER TABLE Cliente
ADD COLUMN EmailCliente varchar(50) ;


ALTER TABLE  Cliente
DROP COLUMN  Bairro;
```

Excluindo uma tabela

Em situações muito específicas, será necessário eliminar uma tabela inteira. Ou seja, remover a estrutura da tabela do banco de dados. Para isso utilizaremos o comando. `DROP TABLE`. Após a execução deste comando estarão deletados todos dados, a estrutura e os índices que estevam a ela associados.

ATENÇÃO: só utilize este comando se você tem certeza de que não quer preservar os dados que estão na tabela a ser excluída.

Sintaxe:
DROP TABLE <nome-tabela>

onde:

1. nome-tabela - Representa o nome da tabela que será eliminada.

Exemplo:

```
DROP TABLE Cliente
```

b) DQL – Linguagem de consulta de dados

Pode-se dizer que essa é a parte SQL da SQL!

A partir de agora veremos o grande trunfo da linguagem SQL: a facilidade em se obter os dados, a partir da especificação daquilo que se espera como resultado da consulta. Inicialmente parecerá um pouco confuso, principalmente porque temos uma série de variações na linguagem, mas a partir do momento que se tenha entendido como um SGBD relacional trabalha, ficará fácil elaborar qualquer consulta aos dados.

O comando básico para a consulta dos dados em um banco relacional é o comando SELECT. Seu objetivo é selecionar um conjunto de registros em uma ou mais tabelas que atenda a uma determinada condição definida pelo comando.

Sua sintaxe simplificada é composta de três partes distintas:
`SELECT - FROM - WHERE.`

`SELECT` – quais dados são esperados
`FROM` – em quais as tabelas se encontram os dados ou as condições para trazê-los;
`WHERE` – quais são as condições que um dado deve atender para que ele possa ser entregue ao solicitante.

Um exemplo simples:

```
SELECT Nome
FROM  Cliente
WHERE Bairro = "Centro"
```

A sintaxe completa:

```
SELECT ALL FROM <nome-tabela>  [, <nome-tabela>]
           DISTINCT
       WHERE <condição>
       GROUP BY <nome-coluna>
       HAVING <condição>
       ORDER BY <nome-campo>  ASC/DESC
```

onde:

1. nome-tabela - Representa o nome da(s) tabela(s) que contem as colunas que serão selecionadas ou que serão utilizadas para a execução da consulta.

2. condição - Representa a condição para a seleção dos registros. Este seleção poderá resultar em um ou vários registros.

3. nome-coluna - Representa a(s) coluna(s) cujos resultados são grupados para atender à consulta.

4. ALL - Mostra todos os campos de todas as tabelas indicadas. Na maioria dos SGBDS utiliza-se o * para indicar todos os atributos;

5. DISTINCT - Opção que mostra os valores obtidos na seleção eliminando as duplicidades.

6. WHERE - Especifica o critério de seleção dos registros nas tabelas especificadas.

7. GROUP BY - Especifica o(s) campo(s) que serão grupados para atender a consulta.

8. HAVING - Especifica uma condição para seleção de um grupo de dados. Esta opção só é utilizada combinada com a opção GROUP BY.

9. ORDER BY - Esta opção quando utilizada apresenta o resultado da consulta ordenado de forma crescente ou decrescente pelos campos definidos.

Algumas funções utilizadas no comando `Select`.

1. COUNT(*) | (<nome-campo>)

Objetivo: Retorna a quantidade de registros existentes no campo especificado. Quando a opção * é utilizada o resultado é a quantidade de registros existentes. Quando é referenciado o nome de um campo retorna a quantidade de valores existentes na coluna.

2. SUM (ALL | <nome-campo>)
DISTINCT

Objetivo: Retorna a soma dos valores existentes no campo especificado.

3. AVG (ALL | <nome-campo>)

Objetivo: Retorna a média dos valores existentes no campo especificado.

4. MAX (ALL | <nome-campo>)

Objetivo: Retorna o maior valor existente no campo especificado.

5. MIN (ALL | <nome-campo>)

Objetivo: Retorna o menor valor existente no campo especificado.

Índices agilizam uma consulta

DICA

Não há limite para a criação de índices, entretanto um número muito grande de índices pode impactar no desempenho do banco de dados, pois cada vez que um novo registro é inserido no banco de dados, todos os índices criados para aquela tabela devem ser atualizados. Vale lembrar que, no ambiente computacional, o esforço pode ser medido por meio do número de leituras que o SGBD realiza no repositório onde o banco de dados está armazenado e o número de operações que ele deve realizar para localizar ou gravar um registro.

Um índice é uma estrutura extremamente útil para agilizar a busca por uma determinada informação. Nós já estamos familiarizados com a utilização de índices na vida prática: temos um índice no início do livro. Seu objetivo é indicar em que página do livro se encontra o início de um determinado capítulo.

Em uma biblioteca também temos um índice, ou um sistema de "consulta ao acervo" que funciona como um índice para os livros que estão naquela biblioteca. Porém imagine se ele não existisse. E agora reflita: qual seria o esforço necessário para localizar o livro com o título "Sistema de Bancos de Dados"?

Nossa única opção seria percorrer todas as prateleiras e estantes da biblioteca, lendo a capa de cada um dos livros até encontrá-lo. Por mais que os livros estivessem ordenados por título, e eu escolhesse livros aleatoriamente, para saber se estou próximo ou distante dele, o tempo gasto seria grande. Por esse motivo, dizemos que o tempo de busca é indeterminado. Não sabemos ao certo se será rápido ou demorado procurar por um livro, independentemente do seu título.

O índice de uma biblioteca fornece uma estrutura de dados que mantém a localização física dos livros e alguns dados por meio dos quais um livro pode ser procurado. Ou seja, a estrutura de indexação provê uma representação simplificada dos dados facilitando o processo de localização deles. Claro que essa estrutura contém apenas informações por meio das quais se pretende localizar um livro.

Sabe-se que é muito comum a procura de um livro por meio de seu título. Então, deve-se criar um índice por título de livro para agilizar esse tipo de consulta. Outra consulta bastante comum é pelo autor do livro. Da mesma forma, é preciso criar um índice baseado nos nomes dos autores. E assim sucessivamente.

Na tecnologia de bancos de dados o funcionamento de um índice é similar. Imagine que existem milhares de dados armazenados em uma tabela de nosso banco de dados. Para localizar uma informação seria necessário pesquisar no banco de dados inteiro, caso não houvesse um índice.

Com a criação dele, é possível reduzir significativamente o esforço na localização de um dado. Em geral, aconselha-se que sejam criados índices para os atributos por meio dos quais as consultas serão criadas no banco de dados. Ou seja, se o sistema que estou desenvolvendo irá realizar consultas através do Nome do Cliente, é preciso que seja criado um índice utilizando esse atributo. Caso a busca seja por título do filme, cria-se outro índice para o atributo correspondente.

c) DML Linguagem de manipulação de dados

Os comandos da linguagem SQL para a manipulação de dados realizam as operações de inclusão, exclusão e alteração dos dados das tabelas.

Inclusão

O comando para a inclusão de um novo registro (ou vários) em uma tabela do banco de dados é o comando `INSERT`.

Sintaxe:

INSERT INTO <nome-tabela> [(<nome-coluna>, [<nome-coluna>])]
VALUES (<relação dos valores a serem incluídos>)

onde:

1. nome-tabela - Representa o nome da tabela onde será incluído um novo registro.

2. nome-coluna - Representa o nome da(s) coluna(s) que receberão dados no momento da operação de inclusão.

Exemplo:

```
INSERT INTO Estado (SiglaUF, NomeEstado)
 VALUES ("SC", "Santa Catarina")
```

Exclusão

O comando para a exclusão de um ou mais registros de uma tabela do banco de dados é o comando `DELETE`.

Sintaxe:

```
DELETE FROM  <nome-tabela>
      WHERE <condição>
```

onde:

1. nome-tabela - Representa o nome da tabela de onde os registros serão deletados.

2. condição - Representa a condição para a deleção dos registros. Este seleção poderá resultar em um ou vários registros. Neste caso a operação irá ocorrer em todos os registros selecionados.

Exemplo:

```
DELETE FROM  Cliente WHERE Cidade = "Florianópolis"
```

Atualização

O comando para a atualização dos dados de um ou um grupo de registros de uma tabela do banco de dados é o comando `UPDATE`.

Sintaxe:

```
UPDATE <nome-tabela>
      SET <nome-coluna> = <novo conteúdo para o campo>
          [<nome-coluna> = <novo conteúdo para o campo>]
      WHERE <condição>
```

onde:

1. nome-tabela - Representa o nome da tabela cujo conteúdo será alterado.

2. nome-coluna - Representa o nome da(s) coluna(s) terão seus conteúdos alterados com o novo valor especificado.

3. condição - Representa a condição para a seleção dos registros que serão atualizados. Este seleção poderá resultar em um ou vários registros. Neste caso a alteração irá ocorrer em todos os registros selecionados.

Exemplo:

```
UPDATE Cliente
SET DataNascimento = '19/09/1990'
WHERE NumeroCliente = 5234
```

d) DCL Linguagem de controle de dados

Os dados armazenados no banco de dados precisam ser protegidos de acessos não autorizados, destruição ou alteração insidiosa e introdução acidental de inconsistência. As restrições (contraints) garantem um certo grau de consistência em uma banco de dados. A integridade pode ser resolvida por mecanismos de recuperação em caso de falhas de sistema. Resta-nos agora verificar os mecanismos de segurança de acesso aos dados.

Segurança e violação de integridade:

O mau uso do banco de dados pode ser classificado como intencional (insidioso) ou acidental. A perda acidental de consistência de dados pode ser conseqüência de:

- Quedas durante o processamento de transações;
- Anomalias causadas por acesso concorrente ao banco de dados;
- Anomalias causadas pela distribuição de dados pelos diversos computadores;
- Erros lógicos que violam as regras impostas para que as transações preservem as restrições de consistência do banco de dados;

É mais fácil a proteção contra perda acidental da consistência dos dados do que se proteger contra o mau uso intencional ao banco de dados. Dentre as formas de acesso insidioso estão as seguintes:

- Leitura não autorizada de dados (roubo de informação);
- Modificação não autorizada de dados;
- Destruição não autorizada de dados;

A proteção absoluta do banco de dados contra abusos é impossível, mas o custo para o criminoso pode tornar-se tão alto que impedirá a maioria, se não todas elas, das tentativas de acesso ao banco de dados sem a devida autorização.

Autorização

Um usuário pode ter várias formas de autorização sobre partes do banco de dados. Dentre elas estão as seguintes:

• Autorização para leitura (`select`) – permite a leitura, mas não a modificação, dos dados;

• Autorização para inclusão (`insert`) – permite a inserção de novos dados, mas não a modificação dos dados existentes;

• Autorização para atualização (`update`) – permite a modificação, mas não a remoção , de dados;

• Autorização para remoção (`delete`) – permite a remoção dos dados.

Um usuário pode receber todos, nenhum ou uma combinação desses tipos de autorização. Além dessas formas de autorização de acesso a dados, pode ser concedida autorização a um usuário para modificar esquemas do banco de dados:

• Autorização `index` – permite a criação e remoção de índices;

• Autorização `resource` – permite a criação de novas relações;

• Autorização `alteration` – permite a adição ou remoção de atributos em uma relação;

• Autorização `drop` – permite a remoção de relações;

As autorizações drop e delete diferem na medida em eu a autorização de remoção permite somente a remoção de tuplas. Se um usuário remover todos os dados de uma relação, embora vazia ela ainda existirá. Se uma relação é retirada, nem ela, nem seus dados existirão.

A capacidade de criar novas relações é controlada pela autorização resource. Um usuário com autorização resource que cria uma nova relação recebe automaticamente todos os privilégios sobre ela.

A última forma de autorização é aquela dada ao administrador do banco de dados (DBA – database administrator). O DBA pode autorizar novos usuá-

rios, reestruturar o banco de dados e assim por diante. Essa forma de autorização é análoga àquela fornecida a um superusuário ou operador de um sistema operacional.

Especificação de segurança em SQL

A linguagem SQL compreende comandos para conceder e revogar privilégios. O padrão SQL inclui os privilégios: `insert`, `select`, `update`, `delete`. A declaração Grant é usada para conferir a autorização. A forma básica dessa declaração é a seguinte:

GRANT <lista de privilégio> ON <nome da relação ou visão> TO <lista de usuários>

Exemplo:

```
GRANT SELECT ON Locacao TO Operador1;
GRANT SELECT,INSERT,UPDATE ON Filme, Cliente to Operador1,Operador2;
```

O privilégio `update` pode ser conferido a todos os atributos da relação como a apenas um deles. Se houver autorização `update` em uma declaração Grant, a lista dos tributos sobre os quais a autorização update é concedida aparece, opcionalmente entre parênteses, imediatamente após a palavra-chave `update`.

Se a lista for omitida, o privilégio `update` será concedido obre todos os atributos da relação.

```
GRANT UPDATE (ValorLocacao) ON Locacao TO Operador1;
```

Para revogar uma autorização, usamos a declaração `revoke`. Ela tem forma semelhante à declaração grant:

REVOKE <lista de privilégios> ON <nome da relação> FROM <lista de usuários>

Então para revogar privilégios concedidos previamente, escrevemos:

```
REVOKE SELECT ON Filme FROM Operador1;
REVOKE UPDATE (ValorLocacao) ON Locacao from Operador1;
```

e) DCL Linguagem de controle de dados

Normalmente, considera-se um conjunto de várias operações no banco de dados como sendo uma única unidade lógica do ponto de vista do usuário.

Por exemplo, a locação de um filme em nosso sistema de vídeolocadora envolve várias operações no SGBD: inclusão dos dados da locação, atualização da situação do filme como: "Locado", débito dos créditos do cliente no saldo de sua conta, etc. Por conseguinte, todas as operações devem ser realizadas com sucesso.

Entretanto, cada operação realizada no banco de dados é considerada como uma transação. Porém, é necessário fazer o controle de execução de uma unidade lógica. Isso significa que as operações que compõem uma unidade lógica só poderão ser efetivadas se todas as operações dessa unidade lógica forem executadas com sucesso.

Caso contrário, ou seja, caso uma das operações de uma unidade lógica falhe, todas as operações anteriores devem ser desfeitas para que os dados continuem íntegros. Vamos a um exemplo clássico para ilustrar o que a falta de controle faria.

Imagine uma operação bancária de transferência de um determinado valor da conta A para a conta B. Temos que o saldo da conta A é de R$ 200,00 e o saldo da conta B é de R$ 120,00. A operação de transferência deverá retirar R$ 70,00 (valor a ser transferido) da conta A e creditar R$ 70,00 na conta B.

Diante disso, podemos afirmar que as operações a serem realizadas são:

```
UPDATE ContaCorrente
SET Saldo = Saldo – 70
Where Conta = A;

UPDATE ContaCorrente
SET Saldo = Saldo = Saldo + 70
Where Conta = B
```

Se não houvesse um controle transacional, caso a segunda operação fosse interrompida por um problema qualquer, a primeira operação não teria como saber do ocorrido e os dados ficariam errados. Tomando os valores do exemplo acima, o saldo da conta A ficaria R$ 130,00 e o da conta B os mesmos R$ 120,00. Ou seja, R$ 70,00 teriam desaparecido.

Uma solução seria que o programador fizesse esse controle pela aplicação, mas essa tarefa seria muito trabalhosa caso o número de operações fosse grande.

Diante disso, e para garantir a integridade dos dados a maioria dos SGBDs modernos implementa o controle transacional. Nele, uma transação começa de modo subentendido, quando cada comando se inicia e ao seu final, a transação também é finalizada.

Porém, o padrão SQL definiu que o início e o final de uma transação possam ser feitos de forma explicita, ou seja, de forma declarada. Assim, todas as operações entre o início da transação e o seu término fazem parte da mesma unidade lógica e, em caso de falha, podem ser desfeitos automaticamente.

Para especificar o início de cada transação utiliza-se a declaração `begin transaction` (em alguns casos a forma simplificada `begin tran` é aceita).

As transações são terminadas por uma das seguintes declarações SQL:
- `COMMIT WORK` – efetiva toda as operações da transação corrente;
- `ROLLBACK WORK` – desfaz todas as operações da transação corrente;

A palavra work é opcional (desnecessária) em ambas as declarações. Utilizando essa funcionalidade, a realização de transferência de valores entre as contas A e B do exemplo anterior deveria ter os comandos de inicialização e término de uma transação, conforme mostrado abaixo:

```
BEGIN TRAN

UPDATE ContaCorrente
SET Saldo = Saldo - 70
Where Conta = A;


UPDATE ContaCorrente
SET Saldo = Saldo = Saldo + 70
Where Conta = B;

COMMIT;
```

RESUMO

Segundo Silberschatz (2004), “Uma linguagem de consulta é a linguagem por meio da qual os usuários obtêm informações do banco de dados”.

Apesar de ter sido criada como uma linguagem de consulta, a SQL oferece cinco grupos de comandos:

Grupo	Funções
Comandos de definição de dados (Data Definiton Language - DDL)	• Criação do banco de dados e seus objetos: tabelas, colunas e restrições; • Alterações dos objetos do banco de dados; • Exclusão de objetos;
Comandos de consulta de dados (Data Query Language - DQL)	Recuperação de dados e informações do banco de dados;
Comandos de manipulação de dados (Data Manipulation Language - DML)	Inclusão, atualização e exclusão de dados das tabelas do banco de dados;
Comandos de controle de acesso de dados (Data Control Language - DCL)	Definição das permissões de acesso a dados e de operações sobre os dados;
Comandos de controle transacional (Data Transaction Language - DTL)	Definição do início e término do conjunto de operações que devem pertencer a mesma transação;

Atividades de aprendizagem

1 – Escreva os comandos para criação das tabelas do banco de dados apresentadas a seguir.

Tabela EMPREGADO					
Nome	RG	CIC	Depto.	RG Supervisor	Salário
João Luiz	10101010	11111111	1	NULO	3.000,00
Fernando	20202020	22222222	2	10101010	2.500,00
Ricardo	30303030	33333333	2	10101010	2.300,00
Jorge	40404040	44444444	2	20202020	4.200,00
Renato	50505050	55555555	3	20202020	1.300,00
Marcos	60606060	6666666	3	20202020	1350,00
Sandra	70707070	7777777	2	10101010	2300,00
Maria	80808080	8888888	2	10101010	2300,00

Tabela DEPARTAMENTO		
Nome	Número	RG Gerente
Contabilidade	1	10101010
Engenharia Civil	2	30303030
Engenharia Mecânica	3	20202020

Tabela PROJETO		
Nome	Número	Localização
Financeiro 1	5	São Paulo
Motor 3	10	Rio Claro
Prédio Central	20	Campinas

Tabela DEPARTAMENTO_PROJETO	
Número Depto.	Número Projeto
2	5
3	10
2	20

Tabela EMPREGADO_PROJETO		
RG Empregado	Número Projeto	Horas
20202020	5	10
20202020	10	25
30303030	5	35
40404040	20	50
50505050	20	35
60606060	10	25
70707070	20	30
80808080	10	30

Tabela DEPENDENTES				
RG Responsável	Nome Dependente	Dt. Nascimento	Relação	Sexo
10101010	Jorge	27/12/86	Filho	Masculino
10101010	Luiz	18/11/79	Filho	Masculino
20202020	Fernanda	14/02/69	Cônjuge	Feminino
20202020	Angelo	10/02/95	Filho	Masculino
30303030	Adreia	01/05/90	Filho	Feminino

Observação: Os campos sublinhados correspondem ao(s) atributo(s) determinante(s) da tabela.

CAPÍTULO

7

PROJETO DE UM BANCO DE DADOS

Objetivo

O capitulo 7 é um capítulo prático. Nele, utilizaremos todos os conceitos apreendidos na construção de um banco de dados em um ambiente computacional.

Existe no mercado de informática uma série de ferramentas de modelagem e gerenciadores bancos de dados relacionais capazes de criar um banco de dados.

Optamos por utilizar nesse curso uma SGBD bastante versátil. Entretanto, todos são muito parecidos. Diante disso, nosso foco será nas ações que estaremos realizando e não como realizá-la nesta ou em outra ferramenta.

Projeto do banco de dados.

Nosso objetivo é projetar e criar fisicamente um banco de dados capaz de suportar um sistema de informações para informatizar a J&J Vídeo. Assim sendo, foram feitos alguns levantamentos sobre as necessidades que os proprietários identificaram para esse sistema, parte das quais está especificada a seguir:

1 – Existem 3 classes de filmes: Lançamento, Especial e Catalogo;

2 – As locações podem ser feitas pelo cliente principal ou por algum de seus dependentes, desde que ele esteja cadastrado;

3 – De acordo com a classificação o preço e o prazo de devolução mudam, segundo a seguinte tabela:

Classe de Filme	Preço	Prazo
Lançamento	R$ 7,00	24 horas
Especial	R$ 6,00	48 horas
Catálogo	R$ 5,00	24horas x Nº itens da locação

4 – Um cliente pode adquirir antecipadamente Vales-Locação. Tais vales são usados para fazer o pagamento das locações. Atualmente eles são adquiridos nas seguintes quantidades/valor:

Quantidade	Valor
10	R$ 65,00
20	R$ 120,00
30	R$ 165,00

5 – As principais informações sobre os filmes são:

a) Titulo em Português
b) Título Original
c) Atores principais
d) Estilo (drama, ação, suspense, comédia, etc.)
e) País de origem
f) Ano de lançamento

6 – Os principais dados de uma locação são:

a) Filme locado
b) Data da locação
c) Hora da locação
d) Cliente que realizou a locação
e) Atendente que efetuou o registro da locação
f) Data da devolução de cada filme

7 – O pagamento das locações pode ser feito em uma das seguintes formas:

a) no momento da locação, em dinheiro, com desconto de 10%;
b) em dinheiro na devolução sem desconto;
c) no momento da locação, em Vale-Locação adquirido antecipadamente;

8 – Os proprietários desejam saber qual o valor recebido em cada locação para avaliar os resultados ao longo do mês.

Observação: Esses requisitos compõem o mínimo que deve existir no sistema. Use seu conhecimento pessoal para agregar mais informações que julgue necessária ou útil ao processo de locação de filmes.

Aguarde a 2ª Parte dos requisitos no ambiente virtual!

Referências Bibliográficas

CHEN, Peter. The Entity-Relationship Model--Toward a Unified View of Data. In: ACM Transactions on Database Systems . ACM-Press, 1976;

DATE, Christopher J. Introdução a Sistemas de Banco de Dados. Rio de Janeiro: Campus, 2004.

ELMASRI, Ramez; NAVATHE, Shamkant. B. Sistemas de Banco de Dados. São Paulo: Pearson Addison Wesley, 2005.

HARRINGTON, Jan L. Projeto de Banco de Dados Relacionais. 2ª Edição. Campus / Elsevier, 2002.

SETZER, Valdemar W.; Silva, Flávio Soares Corrêa da . Bancos de Dados. São Paulo: Edgard Blücher, 2005.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHA, S.. Sistema de Banco de Dados. São Paulo: Makron Books,2004.

WATSON, Richard T. Data Management: Banco de Dados e Organizações. Rio de Janeiro: LTC ,2002