

Aula 1 – Conceitos de Bancos de Dados

Objetivos

Definir conceitos básicos de Bancos de Dados.

Identificar os problemas relativos à tecnologia da informação que induziram ao desenvolvimento de Sistemas Gerenciadores de Banco de Dados (SGBDs).

Enumerar os principais objetivos da tecnologia de Banco de Dados.

Classificar os diferentes tipos de usuários de Banco de Dados.

1.1 Definição

A humanidade sempre se dedicou à produção de signos. Um signo ou símbolo corresponde a qualquer coisa que represente algo para alguém. Da mesma forma que um mapa representa um dado terreno, um brasão pode representar o time de futebol. Uma canção pode fazê-lo lembrar uma fase específica de sua vida ou uma paixão. Tudo isso e muitos outros são signos. Nesse sentido, o que para você nada significa pode representar algo para outros. Portanto, todo signo é subjetivo e pode ser de difícil interpretação.

Até hoje, pesquisadores debruçam-se sobre pinturas feitas em cavernas há pelo menos 15.000 anos e debatem sobre seus significados. Apesar de representarem animais ou cenas de caçadas na pré-história, escapa-nos sua motivação. Faziam parte de algum ritual religioso, eram o registro de uma importante atividade social ou apenas a expressão da subjetividade de seu pintor?

A oralidade primária precede a invenção da escrita e se caracteriza pelo uso da palavra falada como a única forma de gerir a memória social das comunidades. A cultura narrativa valorizava as parábolas, fábulas e mitos como veículos naturais do conhecimento que se pretendia preservar de uma geração para outra. Histórias e imagens eram associadas a algum fato que se buscava memorizar.



O mito codifica, sob forma de narrativa, algumas das representações que parecem essenciais aos membros de uma sociedade. Dado o funcionamento da memória humana, e na ausência de técnicas de fixação da informação como a escrita [...], as representações que têm mais chances de sobreviver em um ambiente composto quase que unicamente por memórias humanas são aquelas que estão codificadas em narrativas dramáticas, agradáveis de serem ouvidas, trazendo uma forte carga emotiva e acompanhadas de músicas e rituais diversos. (LÉVY, 1993, p. 82).

Ao desenvolverem a escrita, as primeiras civilizações melhoraram a utilidade dos símbolos. Afinal, o surgimento do Estado trouxe consigo a inevitável cobrança de impostos. Com a escrita, a contabilidade veio a possuir meios objetivos para registrar tributos devidos e valores pagos. Desde então, o monarca passou a contar com registros precisos de seu tesouro e dos súditos devedores.

Com o capitalismo e o consequente progresso técnico, a administração das organizações tornou-se mais complexa, demandando, por exemplo, maior controle da atividade produtiva: do gerenciamento de estoques, recursos humanos e financeiros. O grande volume de informações registradas em papel dificultava consideravelmente seu gerenciamento e atualização. Então, com os primeiros computadores migraram-se essas informações para dispositivos eletrônicos.

De início, a migração foi implementada de modo pouco organizado, usando-se de sistemas de arquivos tradicionais. Cada aplicação do sistema de informações era tratada isoladamente pela equipe de desenvolvedores. Então, cada aplicação tinha seus próprios arquivos e a redundância de informações era normal. Uma aplicação para o controle da frequência dos funcionários, por exemplo, tinha seu próprio arquivo com dados dos empregados em atividade. Esse arquivo podia não ser compartilhado com a aplicação de controle das férias desses mesmos empregados.

Por isso, dados como nome, número de matrícula e departamento de trabalho podiam facilmente estar duplicados nos diferentes arquivos. Com a multiplicação de aplicações e, assim, de arquivos com redundância de dados, o risco de inconsistências de dados entre eles crescia exponencialmente. Considere, por exemplo, uma funcionária que se casou e mudou de nome. Uma eventual falha humana podia levar a uma situação em que seu nome fosse alterado apenas em alguns desses arquivos, mas não em todos eles.





Os primeiros **Bancos de Dados** surgiram no mercado como uma resposta a esses problemas. Um **Sistema de Gerenciamento de Banco de Dados** (SGBD) consiste em um conjunto de arquivos estruturados e de programas que respondem pelo acesso e manipulação de tais arquivos. Diferente dos sistemas de arquivos tradicionais, o Banco de Dados (BD) favorece o inter-relacionamento dos arquivos; portanto, pode ser definido como “uma coleção de dados inter-relacionados e um conjunto de programas para acessá-los” (KORTH; SILBERSCHATZ; SUDARSHAN, 2006, p. 1).

Simplificando, pode-se definir SGBD como um *software* desenvolvido para gerenciar grandes volumes de informações. O objetivo principal reside em superar problemas comuns aos sistemas de arquivos tradicionais. Tais problemas ou desvantagens (KORTH; SILBERSCHATZ; SUDARSHAN, 2006) são:

- (1) Redundância e inconsistência de dados.** Equipes de desenvolvedores criam diferentes aplicações/sistemas ao longo do tempo. De maneira análoga, em uma mesma organização, diferentes linguagens de programação, por exemplo, podem ser usadas no desenvolvimento de diversos sistemas de informações. Logo, dados distintos podem estar duplicados. Tal redundância conduz a altos custos de armazenamento e dificuldade de atualização das informações.
- (2) Dificuldade no acesso aos dados.** Dados espalhados em arquivos isolados não apresentam as facilidades de acesso e processamento das informações dos BD. Há pouca flexibilidade em relação a demandas que não tenham sido antecipadas quando o sistema foi projetado. Por exemplo, uma vez desenvolvido o sistema, caso haja a necessidade de gerar relatórios com os nomes de todos os empregados com idade igual ou superior a 40 anos, a ausência de uma aplicação específica para esse objetivo traz sérios inconvenientes. Isso porque quando os usuários do sistema solicitarem esse novo aplicativo, sua implementação demandará tempo e recursos dos programadores para a geração de um relatório muito específico que raramente será usado. Outra solução seria conseguir a impressão de uma listagem com todos os empregados existentes para posterior extração manual da informação desejada. Essa solução também não é satisfatória, pois o trabalho manual está sujeito a erros humanos.
- (3) Isolamento de dados.** Dados espalhados em diferentes arquivos e formatos dificultam a criação de novos aplicativos para sua recuperação.

A-Z

Banco de Dados (BD)

É um conjunto de dados integrados reunidos com o intuito de suportar o funcionamento de sistemas de informação.

Sistema Gerenciador de Banco de Dados (SGBD)

É um *software* de caráter geral para a manipulação eficiente de grandes coleções de informações estruturadas e armazenadas de uma forma consistente e integrada.





(4) Anomalias de acesso concorrente. Inúmeros sistemas de informação permitem que múltiplos usuários acessem e atualizem dados simultaneamente. A inexistência de sofisticados mecanismos de gerenciamento de atualizações concorrentes pode resultar em dados inconsistentes. Considere o exemplo de uma conta bancária conjunta com saldo de R\$ 600,00, caso dois clientes saquem dinheiro dessa mesma conta simultaneamente. Suponha que o cliente A saque R\$ 50,00 ao mesmo tempo em que o cliente B saca R\$ 100,00. Então, o programa aplicativo lê o saldo da conta de R\$ 600,00. Em seguida, o valor do saque é diminuído do saldo lido, gerando assim o saldo atualizado. Então, para o aplicativo de A, temos $(600 - 50) = 550$, enquanto para o aplicativo de B, temos $(600 - 100) = 500$. Supondo que o aplicativo de A atualize o arquivo antes do aplicativo de B, o saldo da conta registrará R\$ 500,00. Afinal, no arquivo, os dados do aplicativo de B sobreporão os dados do aplicativo de A. Repare que essa informação está errada – inconsistente – pois havia R\$ 600,00 e foram retirados R\$ 150,00. Portanto, o saldo deveria ser de R\$ 450,00.



A tecnologia de BD trouxe maior produtividade para o desenvolvimento de sistemas de informação. Afinal, nos sistemas tradicionais de arquivos, quando implementadas, as características inauguradas com o surgimento dos BD eram incorporadas às aplicações isoladamente. Logo, a complexidade do desenvolvimento de sistemas de informação aumentava consideravelmente, resultando em:

(1) Prazos maiores para a conclusão dos sistemas de informação e, consequentemente, elevação dos custos envolvidos no desenvolvimento.

(2) Obrigação de as equipes desenvolvedoras (analistas e programadores) dedicarem mais tempo a programas que não tratavam diretamente do sistema de informação em questão, mas sim de funcionalidades que forneceriam suporte a esse mesmo sistema.

(3) Ocorrência de problemas não previstos, favorecidos pela ausência de padronização das funcionalidades de suporte.

(5) Problemas de segurança. O acesso a determinados dados deve ser restrito para alguns usuários do sistema de informações. Os dados relativos ao contracheque dos empregados não podem ser disponibilizados a todos os usuários indistintamente, mas apenas aos usuários responsáveis pela folha de pagamento. Entretanto, quando programas aplicativos são instalados de maneira arbitrária, ou sem acompanhamento e controle necessários, não há como assegurar tais restrições de segurança.

(6) Problemas de integridade. Restrições de consistência são impostas aos dados armazenados. Elas são normalmente regras de negócio. O saldo de uma conta bancária, por exemplo, não deve cair abaixo de um valor predeterminado. Restrições de consistência como essa, nos sistemas de arquivos tradicionais, são incorporadas aos códigos dos programas aplicativos. Um problema grave ocorre quando a adição de novas restrições implica a necessidade de alteração de vários programas aplicativos. Sempre há o risco de esquecimento de algum desses programas, ameaçando a integridade dos dados.

Essas seis dificuldades levaram ao desenvolvimento dos SGBD. Ao longo do curso veremos algumas das estratégias criativamente engendradas para a solução das desvantagens dos sistemas de arquivos tradicionais.





Os sistemas de BD atualmente existentes no mercado aliviam as equipes desenvolvedoras de preocupações do que não esteja relacionado aos seus objetivos, permitindo assim a codificação de sistemas de informação mais robustos e confiáveis.

Crie pelo menos dois possíveis exemplos para as desvantagens/problemas trazidos pelo uso de sistemas tradicionais de arquivos.



1.2 Objetivos de Banco de Dados

Sistemas Gerenciadores de Bancos de Dados (SGBD) têm o objetivo de prover mecanismos adequados ao armazenamento e acesso seguro e eficiente de dados em Sistemas de Informação (SI). Mais detalhadamente, os BD têm por objetivo:

- fornecer interfaces amigáveis e padronizadas para o armazenamento e acesso aos dados, poupando os usuários dos detalhes da implementação interna;
- assegurar a privacidade dos dados através de medidas de segurança como: atribuição de permissões de acesso; criação de visões; e fornecimento de senhas de acesso, evitando o acesso a dados por pessoas não autorizadas;
- administrar acessos concorrentes aos dados, permitindo que diferentes usuários compartilhem simultaneamente a mesma coleção de dados;
- prover mecanismos para a recuperação de dados em caso de eventuais paradas e falhas do sistema, as quais podem ocorrer por causa de erros de *software*, interrupção no suprimento de energia, defeito de *hardware*, queda na comunicação com o servidor, etc. Qualquer falha pode resultar na perda de dados processados pelo SGBD no momento da parada. A perda desses dados pode levar o BD a uma condição de inconsistência que, se não evitada, torna a tecnologia pouco confiável. Portanto, durante uma venda, o estoque deve ser atualizado e, se uma falha ocorre após a venda, mas antes da atualização do estoque, o BD ficará inconsistente. Afinal, a quantidade em estoque não será real para o produto em questão.

Um SGBD foi definido como “uma coleção de dados inter-relacionados e um conjunto de programas para acessá-los” (KORTH; SILBERSCHATZ; SUDARSHAN, 2006, p. 1). Entretanto, os sistemas tradicionais de arquivo que precederam a tecnologia de BD falhavam exatamente nesses dois aspectos: (I) manter dados





inter-relacionados e (II) fornecer um conjunto de programas voltados à manutenção de tais dados.

- a) Comente as vantagens que a existência de **dados inter-relacionados** traz aos sistemas de informação.
- b) Uma das principais funcionalidades de um Banco de Dados é a chamada **restrição de consistências**. Explique os riscos que a ausência dessa funcionalidade trazia aos sistemas tradicionais de arquivos.

1.3 Usuários de Bancos de Dados

Basicamente são quatro os tipos de usuários de sistemas de Bancos de Dados:

- **Usuários leigos:** profissionais de outras áreas cujos conhecimentos de informática se restringem ao básico e que interagem com o BD por meio de aplicações escritas pelos programadores de aplicações. As aplicações fazem a intermediação entre esses usuários e o BD, pois através de telas ou páginas tal usuário acessa seus dados.
- **Usuários avançados:** usuários com maior nível de independência em relação aos programadores de aplicações: interagem com os Bancos de Dados por meio de interfaces disponíveis no ambiente e escrevem consultas para relatórios com certa facilidade, sem a necessidade de um programador escrever uma nova aplicação.
- **Programadores de aplicações:** profissionais formados em computação que constroem aplicações/programas de computador com interfaces intuitivas e amigáveis (formulários e relatórios acessando o BD) para os usuários leigos.
- **Analistas de sistemas:** profissionais responsáveis por traduzir as necessidades dos usuários leigos e avançados em uma especificação racional de um sistema de informação. Definem o projeto do sistema de informação que especifica as aplicações e a estrutura do Banco de Dados e que será rigorosamente seguido pelos programadores, desenvolvendo programas que acessem o Banco de Dados;
- **Administrador de BD (Database Administrator – DBA):** usuário mais especializado responsável por administrar as bases de dados; ocupa-se com:





- (1) a atribuição de permissões de acesso adequadas a cada usuário;
- (2) a geração de cópias de segurança (*backups*) como contingência contra falhas;
- (3) o monitoramento do ambiente como forma de assegurar a disponibilidade do Banco de Dados pelo maior tempo possível;
- (4) a otimização de recursos de infraestrutura (disco, memória, processador) para assegurar o desempenho satisfatório do BD;
- (5) o suporte à equipe de desenvolvimento (analistas e programadores).

Em resumo, o **DBA** deve ser o profissional que zela pela implementação adequada do BD, assegurando um funcionamento eficiente que prime pelo desempenho, escalabilidade, flexibilidade e confiabilidade.

1.4 Modelos de Bancos de Dados

Os modelos de BD definem a forma como os dados estão organizados internamente. Em ordem cronológica, são assim classificados: em redes, hierárquicos, relacionais, objeto-relacionais e orientados a objetos.

1.4.1 Modelo em rede

Um BD em rede é uma coleção de registros concatenados uns aos outros por meio de ligações. Semelhantes ao conceito de ponteiros, elas são entendidas como endereços de memória que “apontam” a localização de um registro associado.

Na Figura 1.1, há um modelo em rede com dois diferentes tipos de registros: cliente e conta. O registro de cliente apresenta três atributos: nome, cidade e sexo. Por sua vez, o registro de conta possui apenas dois atributos: número e saldo.



Algumas organizações preferem manter um profissional que acumule as funções do analista de sistemas e do programador de aplicações. Isso lhes permite reduzir custos trabalhistas e evitar “ruídos de comunicação” que comprometam a eficiência dos sistemas de informações. Afinal, sempre há a possibilidade de que o analista de sistemas não seja plenamente entendido pelo programador de aplicações, gerando um sistema de informação muito diferente do que o usuário requisitara inicialmente. Assim, as aplicações teriam de ser reescritas, tornando o processo mais caro e menos eficiente.



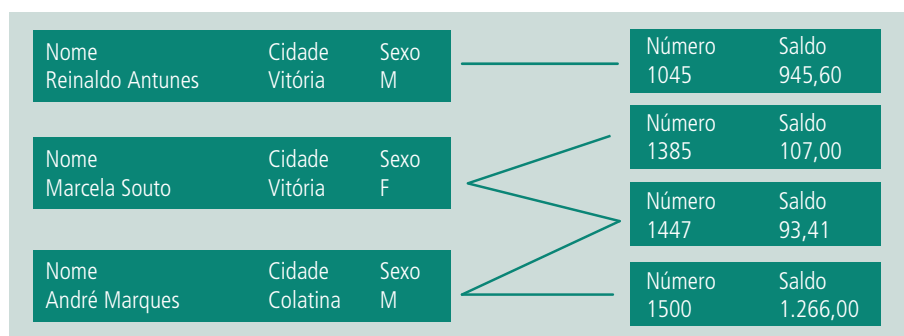


Figura 1.1: Exemplo de Banco de Dados – modelo em rede

Fonte: Elaborada pelo autor

Ligações associam registros de clientes aos de contas. Então, sabemos que o cliente Reinaldo Antunes tem uma conta de número 1045 e saldo de R\$ 945,60; e Marcela Souto e André Marques possuem uma conta conjunta de número 1447, com R\$ 93,41. Porém, Marcela tem ainda outra conta número 1358 e saldo R\$ 107,00; e André Marques também tem outra de número 1500 com R\$ 1.266,00.

Um conjunto de diferentes tipos de registros relacionados entre si por meio de um emaranhado de ligações forma uma estrutura de dados semelhante a uma rede.



Seguindo o modelo da Figura 1.1, elabore um exemplo de BD no modelo em rede envolvendo os seguintes dados: Funcionário {matrícula, nome, salário, função} e Filial {cidade, bairro, telefone}. O objetivo dessa coleção de dados é armazenar informações sobre todos os funcionários de um supermercado e de suas filiais de diversas localidades. Por meio dessa coleção deve ser possível determinar as filiais em que cada funcionário está lotado.

Esse exemplo deverá apresentar pelo menos duas filiais com um mínimo de cinco funcionários por elas distribuídos. Use sua criatividade para dar nomes aos funcionários e preencher os outros dados sobre eles e as filiais nas tabelas. Lembre-se de que um funcionário somente poderá estar lotado em uma filial.

1.4.2 Modelo hierárquico

Uma evolução do modelo em rede, os BD hierárquicos são uma coleção de registros relacionados uns aos outros por meio de ligações semelhantes a ponteiros. Porém, se diferenciam de seu antecessor na organização seus registros. Enquanto no modelo em rede os registros são distribuídos conforme a lógica de grafos arbitrários, no hierárquico eles são dispostos como uma coleção de árvores. A Figura 1.1 ficaria então traduzida como mostra a Figura 1.2 a seguir.



Árvores binárias são um dos temas tratados na disciplina referente à Programação em Estrutura de Dados.

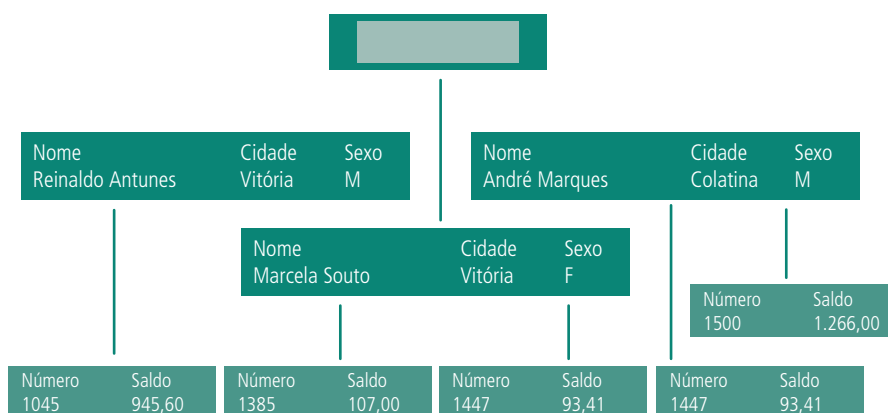


Figura 1.2: Exemplo de Banco de Dados – modelo hierárquico

Fonte: Elaborada pelo autor

Um registro isolado no topo da Figura 1.2 encontra-se associado a todos os registros de Clientes. Esse é o registro do tipo “raiz”, o ponto de partida da árvore de registros. Ele está no nível mais elevado da estrutura de dados e pode ser ligado a nenhum, um ou vários registros no nível inferior. Em nosso exemplo, os registros conectados ao registro “raiz” são sempre registros de Clientes que, por sua vez, estão interligados a registros de Contas no nível imediatamente abaixo.

Generalizando, existem camadas, níveis ou hierarquias em uma árvore. Os registros de um nível sempre se associam aos do imediatamente inferior e nunca aos do mesmo nível. No primeiro nível da árvore há sempre um único registro, a “Raiz”, que representa o ponto de partida para acessar os demais registros.

Cada nível da árvore comporta apenas um tipo de registro. No exemplo, o nível abaixo da raiz comporta registros do tipo Cliente; e o seguinte, do tipo Conta. Ainda poderiam existir outros níveis se houvesse registros de outros tipos.

Na árvore, um registro de nível K pode estar associado a, no máximo, um registro do nível imediatamente acima ($K - 1$). Contudo, um registro de nível superior pode estar associado a mais de um registro do nível imediatamente abaixo. Note que o registro da cliente Marcela Souto liga-se a dois outros registros de conta, mas a conta de número 1447 teve de ser duplicada para que a regra fosse preservada, replicando o registro para que André Marques compartilhasse essa conta com Marcela Souto, já que um registro de conta não pode se associar simultaneamente a dois de clientes.

A estrutura hierárquica do BD define o caminho de acesso a registros. A busca ao registro sempre começa pela raiz até o nível correspondente ao tipo procurado.



Converta o exemplo de BD no modelo em rede da atividade anterior em um exemplo equivalente no modelo hierárquico.

1.4.3 Modelo relacional

Na década de 1970, o modelo relacional de BD estabeleceu-se como o preferencial para aplicações comerciais. Na década seguinte já era o padrão no mercado corporativo.



É comum encontrar na literatura as denominações Pai e Filho para distinguir a relação existente entre registros de diferentes camadas da árvore. Registros de uma camada K geralmente são designados Filhos dos registros de uma camada (K – 1). Em contrapartida, os dessa camada (K – 1) são considerados Pais dos registros da camada K que a eles estiverem conectados. Portanto, em nosso exemplo da Figura 1.2, os registros de Conta são Filhos dos registros de Clientes.

No modelo relacional, os registros são organizados em tabelas e cada linha representa uma relação entre os valores armazenados em diferentes colunas. Assim, na tabela de CLIENTE (Figura 1.3), seguindo o exemplo desenvolvido até aqui, temos cada registro subdividido em três colunas: NOME, CIDADE e SEXO. Toda linha existente na tabela de CLIENTES representa um conjunto de valores inter-relacionados. Ou seja, cada linha da tabela armazena os dados de um cliente específico organizados em diferentes colunas.

CLIENTE		
NOME	CIDADE	SEXO
Reinaldo Antunes	Vitória	M
Marcela Souto	Colatina	F
André Marques	Colatina	M

Figura 1.3: Exemplo de tabela CLIENTE em um modelo relacional de Banco de Dados
Fonte: Elaborada pelo autor



Em 1970, Edgar Frank Codd, um pesquisador da IBM, propôs o modelo relacional de BD tratado aqui. Esse novo modelo trouxe uma importante contribuição ao dissociar a estrutura lógica do BD dos métodos de armazenamento físico dos dados – algo impossível para os modelos em rede e hierárquico. Ao fazê-lo, tornou os SGBD mais “amigáveis” (fáceis de utilizar).

Assim, sabemos que o cliente “Reinaldo Antunes” mora em “Vitória” e é do sexo masculino (“M”), pois esses dados estão na mesma linha da tabela. Também sabemos que os clientes “Marcela Souto” e “André Marques” residem na cidade de “Colatina” e são, respectivamente, do sexo feminino e masculino.

O modelo relacional de BD apresenta esse nome em razão da relação existente entre as colunas de uma mesma tabela, e também da possibilidade de estabelecer relacionamentos entre diferentes tabelas. Para que o exemplo de clientes e contas fique completo, precisamos relacionar registros de uma

tabela com os de outra. Caso contrário, não teríamos como saber com precisão a quais clientes cada conta pertence. Porém, as associações entre registros não são implementadas mediante “ponteiros”. Na verdade, o modelo relacional emprega a duplicação de uma ou mais colunas em uma tabela distinta daquela a que pertencem originalmente.

Ponteiros é tema da disciplina de Programação em Estrutura de Dados.



Na Figura 1.4 temos duas tabelas – FILME e GENERO – relacionadas entre si através da coluna COD_GENERO. Essa coluna é original da tabela GENERO, mas foi duplicada na tabela FILME para estabelecer uma associação lógica entre os registros das duas tabelas. Logo, sabemos que o filme “*Star Trek*” é classificado como ficção científica, pois apresenta o valor “FIC” em sua coluna COD_GENERO, e o mesmo valor na coluna correspondente da tabela GENERO está associada à descrição “Ficção Científica”.

NUMERO	TITULO	ANO	COD_GENERO
1099	El Cid	1961	DRA
1100	Star Wars	1977	FIC
1101	Star Trek	2009	FIC
1103	Transformers 2	2009	FIC

COD_GENERO	DESCRICAO
DRA	Drama
FIC	Ficção

Figura 1.4: Exemplo de relacionamento entre as tabelas FILME e GENERO

Fonte: Elaborada pelo autor

Há quem não compreenda a necessidade de duas tabelas em situações como essa. Afirmam ser uma solução complicada e perguntam: A coluna descrição não poderia simplesmente existir apenas na tabela FILME para não haver a tabela GENERO e nem as colunas duplicadas de COD_GENERO? A princípio essa parece uma solução mais simples e, portanto, melhor, mas a Figura 1.5 procura exemplificar sua vulnerabilidade.

FILME			
NUMERO	TITULO	ANO	DESCRICAO_GENERO
1099	El Cid	1961	Drama
1100	Star Wars	1977	Ficção - Científica
1101	Star Trek	2009	Ficção
1103	Transformers 2	2009	Ficção - Científica

Figura 1.5: Vulnerabilidade da fusão equivocada das tabelas FILME e GENERO

Fonte: Elaborada pelo autor

Os valores da coluna DESCRICAO_GENERO podem se repetir em diferentes registros (linhas). Na tabela da Figura 1.5 há três filmes do gênero ficção científica. Contudo, cada filme apresenta valores diferenciados para essa coluna. No filme “*Transformers 2*” não há hífen, mas em “*Star Wars*” esse caractere está presente. Já no filme “*Star Trek*”, não somente houve a omissão do hífen como também a supressão da palavra “Científica”. Algo assim é perfeitamente possível de acontecer. Usuários diferentes podem ter cadastrado cada um desses filmes, ou ainda um mesmo usuário em diferentes ocasiões. Acontece que, em um momento, esse usuário encontrava-se um tanto cansado e registrou apenas “ficção”. Mais tarde, por um erro de digitação, passou a usar o hífen.

A possibilidade de ocorrer tal situação demonstra a vulnerabilidade da solução baseada em uma única tabela. Caso uma consulta seja efetuada, o resultado pode diferir da realidade que a base de dados deveria espelhar. Suponha que um usuário submeta uma consulta ao BD usando como critério de busca a coluna DESCRICAO_GENERO com o valor “Ficção Científica”. O resultado da consulta não incluirá os registros de “*Star Trek*” e “*Star Wars*” e ele pensará que existe apenas um filme desse gênero: “*Transformers 2*”.

Ao tratarmos do conceito de integridade referencial, a solução inicial (Figura 1.5) é a mais indicada para evitar tais problemas. Ocorre que a tecnologia de BD relacionais possui recursos suficientes para assegurar que a coluna COD_GENERO da tabela FILME apresente somente valores já existentes na coluna COD_GENERO da tabela GENERO. Logo, nenhum usuário poderá cadastrar um novo filme ou alterar um velho filme com um código inexistente na tabela de origem (GENERO). O SGBD relacional simplesmente não permitirá: não se consegue usar um novo código de gênero para um Filme

sem antes criá-lo na tabela de GÊNERO. No máximo, um usuário descuidado poderia lançar um código errado para um filme.

Retomando o exemplo do BD para clientes e contas bancárias, a Figura 1.6 apresenta a configuração ideal para o modelo relacional. Uma terceira tabela CONTAS_CLIENTE precisou ser criada para relacionar CLIENTE e CONTA. No exemplo sobre FILME e GÊNERO, não foi necessário, bastou replicar a coluna COD_GÊNERO na tabela FILME. Porém, a duplicação da coluna ID_CLIENTE na tabela CONTA ou a da coluna NUMERO_CONTA na tabela CLIENTE não seria uma solução eficiente. Por isso, criamos uma tabela apresentando duas colunas e nenhuma delas é originária da nova tabela (CONTAS_CLIENTE). As colunas ID_CLIENTE e NUMERO_CONTA são replicadas respectivamente nas tabelas Cliente e Conta.

CLIENTE			
ID_CLIENTE	NOME	Cidade	SEXO
10	Reinaldo Antunes	Vitória	M
15	Marcela Souto	Colatina	F
37	André Marques	Colatina	M

CONTAS_CLIENTE		CONTA	
NUMERO_CONTA	ID_CLIENTE	NUMERO_CONTA	SALDO
1045	10	1045	945,60
1385	15	1385	107,00
1447	15	1447	93,41
1447	37	1500	1.266,00
1500	37		

Figura 1.6: Exemplo um de Banco de Dados – modelo relacional

Fonte: Elaborada pelo autor

Caso duplicássemos ID_CLIENTE na tabela CONTA, haveria um problema ao cadastrar a conta número 1447, como mostrado na Figura 1.7. Essa conta pertence a dois clientes, é uma conta conjunta: André (ID_CLIENTE = 37) e Marcela (ID_CLIENTE = 15). Como a coluna ID_CLIENTE na tabela CONTA comporta apenas um valor por linha, teríamos de cadastrar duas linhas para a Conta de número 1447.

CONTAS_CLIENTE		
NUMERO_CONTA	SALDO	ID_CLIENTE
1045	945,60	10
1385	107,00	15
1447	93,41	15
1447	93,41	37
1500	1.266,00	37




Figura 1.7: O problema de duplicar ID_CLIENTE na tabela CONTA

Fonte: Elaborada pelo autor

Por outro lado, se optássemos por duplicar NUMERO_CONTA na tabela CLIENTE, teríamos outro problema para registrar as duas contas de “Marcela Souto” – números 1385 e 1447 (Figura 1.8), pois teríamos de cadastrar duas linhas exatamente iguais para Marcela na tabela CLIENTE, exceto pelo conteúdo da coluna duplicada de NUMERO_CONTA.

CONTAS_CLIENTE				
ID_CLIENTE	NOME	ID_CLIENTE	SEXO	NUMERO_CONTA
10	Reinaldo Antunes	Vitória	M	1045
15	Marcela Souto	Colatina	F	1385
15	Marcela Souto	Colatina	F	1447
37	André Marques	Colatina	M	1447
7	André Marques	Colatina	M	1500



Figura 1.8: O problema de duplicar NUMERO_CONTA na tabela CONTA

Fonte: Elaborada pelo autor

As consequências mais sérias para essas tentativas equivocadas de relacionamento das tabelas CLIENTE e CONTA seriam:

- (1) Uso ineficiente dos recursos de armazenamento: redundância de dados consumiria desnecessariamente o espaço disponível no disco rígido.
- (2) Maior complexidade para a manutenção das informações: a atualização do saldo em uma conta (Figura 1.7) ou a mudança de cidade para um cliente (Figura 1.8) poderia alterar mais de um registro simultaneamente.

Por mais que a nova tabela CONTAS_CLIENTE (Figura 1.6) também seja uma redundância de dados, quando comparada às outras situações (Figuras 1.7 e 1.8), ela representa uma replicação controlada e, portanto, mais eficiente.



Na Figura 1.8, a duplicação de dados abrange quatro colunas (ID_CLIENTE, NOME, CIDADE e SEXO), e na Figura 1.6, envolve duas colunas (ID_CLIENTE e NUMERO_CONTA), mas nenhuma combinação se repete na tabela CONTAS_CLIENTE.

Como a duplicação de dados é inerente ao modelo relacional – o que é inevitável – a situação representada pela Figura 1.6 mostra-se plenamente aceitável e preferível às demais situações representadas pelas Figuras 1.7 e 1.8.

A maneira de definir quais colunas e em quais tabelas as mesmas deverão ser replicadas será tratada adiante quando abordarmos conceitos como chaves primárias, chaves estrangeiras e integridade referencial. Também veremos com detalhes a linguagem de consulta e manipulação de BDs relacionais conhecida como *Structured Query Language* (SQL), padrão desde 1980.



Converta o exemplo de BD das Atividades anteriores em um exemplo equivalente no Modelo Relacional.



1.4.4 Modelo objeto-relacional

SGBDs que adotam o modelo Objeto-Relacional (OR) aproveitam a estrutura básica do modelo relacional com algumas características próprias da orientação a objetos. Porém, esse modelo híbrido não deve ser confundido com o Orientado a Objetos. Dentre as características da orientação a objetos incorporadas pelo modelo Objeto-Relacional merecem destaque: a herança de tipos e tabelas e a definição de novos tipos complexos.

O modelo Objeto-Relacional é conhecido como modelo relacional estendido. Sua linguagem de consulta foi adaptada para abranger objetos, atributos multivalorados, dados abstratos, métodos e funções como predicados de busca.

O padrão ANSI SQL-99 ou SQL-3, caracterizado como SQL orientada a objetos, trouxe inovações em relação à SQL-92. Ela é a base de vários SGBDs OR, como o Oracle 11g, o IBM DB2 Universal Database e o Informix Universal Server.

1.4.5 Modelo orientado a objetos

Os modelos em rede, hierárquico e relacional trouxeram importantes contribuições para a tecnologia de BD, principalmente os comerciais. Qualquer um desses modelos foi um avanço em relação aos sistemas tradicionais de arquivos. Dentre os três modelos acima, o relacional merece destaque, pois, desde a década de 1980, tornou-se padrão de mercado no desenvolvimento de aplicações





comerciais, como controle de estoques, contas a pagar e a receber, frente de loja, recursos humanos, etc. Contudo, a complexidade de novas demandas tecnológicas como os sistemas de informações geográficas e multimídias evidenciaram as limitações desses modelos. As novas aplicações precisavam suportar estruturas complexas de dados para objetos, assim como o armazenamento de imagens digitalizadas e textos muito longos (ELMASRI; NAVATHE, 2005).

Outro fator que pressionou o desenvolvimento de um modelo Orientado a Objeto (OO) para BD é a predominância de linguagens de Programação Orientadas a Objeto (POO). Logo, programadores de aplicações convivem com os dois paradigmas de desenvolvimento: modelo relacional para BD e orientação a objetos para os programas que trabalham com esses mesmos bancos. Tudo seria mais simples se aplicações e bancos que dão suporte à persistência de seus dados fossem igualmente orientados a objetos, não sendo mais necessário o mapeamento objeto-relacional para combinar as duas tecnologias.

Infelizmente, BDs orientados a objetos ainda não foram bem aceitos no mercado. Tal rejeição é explicada pela simplicidade e popularidade do modelo relacional junto aos profissionais de informática. Portanto, muitos desenvolvedores optam por uma solução híbrida como o modelo Objeto-Relacional ou modelo relacional estendido.



Modelo Orientado a Objetos é abordado na disciplina referente à Análise e Projeto de Sistemas.

Resumo

Nesta aula você se familiarizou com conceitos básicos de Banco de Dados. Esses conceitos serão desenvolvidos com mais detalhes ao longo do material. Vimos como as exigências modernas de um maior controle sobre os dados armazenados levaram ao surgimento de sistemas de informação; inicialmente, desenvolvidos como sistemas de arquivos tradicionais que constituíam um grande avanço em relação os mecanismos manuais de registro das informações, mas apresentavam sérias limitações que somente foram corrigidas com a tecnologia de Banco de Dados. Também abordamos os diferentes tipos de usuários para Bancos de Dados. Os sistemas gerenciadores de Bancos de Dados (SGBDs) foram implementados em diferentes modelos ao longo do tempo. Inicialmente surgiram os SGBDs de modelo em rede. Com a evolução tecnológica, sucederam-no os modelos hierárquico, relacional e o objeto-relacional. Um modelo orientado a objetos é esperado para breve, mas ainda não se estabeleceu no mercado.



Atividades de aprendizagem

1. Liste em ordem cronológica os modelos de BD.
2. Cite causas que levaram ao surgimento do modelo Orientado a Objetos de BD.
3. Por que BDs orientados a objetos não se tornaram popular no mercado? Poderíamos afirmar que a tecnologia de BD estagnou no modelo relacional incorporando poucas inovações desde o início da década de 1970?
4. As tabelas mostradas na Figura 1.9 correspondem ao modelo relacional de BD.

Funcionários				Cargos	
MATRICULA	NOME	CPF	CARGO	CODIGO	NOME_CARGO
01	Ana	123	02	01	Programador
02	Maria	234	01	02	Topógrafo
03	José	245	03	03	Engenheiro
04	Pedro	125	01		

Figura 1.9: Exemplo de BD de uma empresa

Fonte: Elaborada pelo autor

Converta esses dados:

- a) para o modelo redes, e
 - b) para o modelo hierárquico (lembre-se de começar dos dados mais gerais para os mais específicos dentro da árvore).
5. (FCC 2006 – SEFAZ PB – Auditor Fiscal de Tributos Estaduais) Um gerenciador de Banco de Dados relacional:
- a) Identifica a relação entre seus registros a partir de ponteiros no sentido filho-pai, unicamente.
 - b) Identifica a relação entre dois ou mais registros a partir da sua justaposição.
 - c) Não contempla a definição de dados pertinentes às tabelas.
 - d) Identifica a relação entre seus registros a partir de ponteiros no sentido pai-filho, unicamente.
 - e) Deve possibilitar a identificação única de uma linha de uma tabela.

