Capítulo 1 - Conceitos, arquitetura e modelo de dados relacional

Este capítulo tem como objetivo:

- Introduzir os principais conceitos usados em sistemas de banco de dados; descrever a arquitetura de SGDB conhecida como ANSI/SPARC, que permite três níveis de esquema; e apresentar os conceitos de modelagem, estrutura de dados e as restrições do modelo relacional.

Este capítulo descreve os conceitos introdutórios básicos do modelo de dados relacional e a arquitetura ANSI/SPARC. Inicialmente, apresentaremos os principais conceitos usados em sistemas de banco de dados. Em seguida, mostraremos a arquitetura de SGDB conhecida como ANSI/SPARC, que permite três níveis de esquema. Por fim, são apresentados os conceitos de modelagem, estrutura de dados e as restrições do modelo relacional. O objetivo principal é que ao final deste capítulo vocês tenham aprendido os conceitos introdutórios básicos necessários para um bom conhecimento dos modelos de banco de dados, sistemas e linguagens.

1. Introdução

Os bancos de dados e os sistemas de banco de dados se tornaram componentes essenciais da vida moderna. No nosso cotidiano nos deparamos com diversas atividades que envolvem alguma interação com um banco de dados. Por exemplo, quando vamos ao banco para depositar ou realizar um saque, fazemos uma reserva de voo ou de hotel, declaramos o imposto de renda anual, acessamos o catálogo de uma biblioteca virtual para procurar uma bibliografia, ou compramos algo on-line de um fornecedor, provavelmente essas atividades envolverão algum programa de computador que acessa um banco de dados. Inclusive a compra de produtos em um supermercado atualiza automaticamente o banco de dados que mantém o controle de estoque dos produtos.

Até recentemente bases de dados costumavam armazenar unicamente dados alfanuméricos (tipicamente cadeias de caracteres e valores numéricos). Atualmente elas estão armazenando também imagens, gráficos, e até objetos multimídia (som e vídeo), o que aumenta enormemente as necessidades de armazenamento e a complexidade de recuperação e processamento de dados. A última aplicação acima, por exemplo, poderia incluir informações geográficas da rede de cabos da

Companhia Telefônica, e seria um caso típico de uma nova especialização de banco de dados, os chamados *bancos de dados geográficos* (GUIMARÃES, 2003).

Os sistemas de bancos de dados e a sua tecnologia favorecem o uso crescente dos computadores, desempenhando um papel crítico em quase todas as áreas em que os computadores são usados, tais como, comércio eletrônico, economia, engenharia, medicina, genética, direito e educação. Aliás, o termo *banco de dados* (do inglês, *database*) é tão utilizado que precisamos começar por sua definição.

Um **banco de dados** é um conjunto de dados integrados que tem por objetivo atender a uma comunidade específica (HEUSER, 2004). Com **dados**, queremos dizer fatos conhecidos que podem ser registrados e possuem significado implícito. Por exemplo, considere os nomes, números de telefone e endereço das pessoas que você conhece. Você pode ter registrado esses dados em uma agenda ou, talvez, os tenha armazenado em um disco rígido, usando um computador pessoal e um software como *Microsoft Access* ou *Microsoft Excel*. Essa coleção de dados relacionados, com significado implícito é um banco de dados.

Em outras palavras, um banco de dados tem alguma fonte da qual o dado é derivado, algum grau de interação com eventos do mundo real (minimundo) e um público que está ativamente interessado em seu conteúdo. Os usuários finais de um banco de dados podem realizar transações comerciais (por exemplo, um cliente compra uma câmera) ou eventos podem acontecer (por exemplo, uma funcionária tem um filho), fazendo que a informação do banco de dados mude. Para que um banco de dados seja preciso e confiável o tempo todo, ele precisa ser um reflexo verdadeiro do minimundo que representa; portanto, as mudanças precisam ser refletidas no banco de dados o mais breve possível.

Um banco de dado pode ser muito simples ou muito complexo ou de tamanhos que podem variar ordens de magnitude de um banco de dados para outro. Por exemplo, um banco de dados poderia ser a relação dos nomes e telefones das pessoas conhecidas de um indivíduo ou a base de dados dos contribuintes da Receita Federal. Bancos de dados podem ser criados e mantidos manualmente, ou podem ser computadorizados. Por exemplo, um catálogo de cartão de biblioteca é um banco de dados que pode ser criado e mantido manualmente. Um banco de dados pode ser criado e mantido por um grupo de programas de aplicação, escritos especificamente para essa tarefa, ou por um sistema gerenciador de banco de dados.

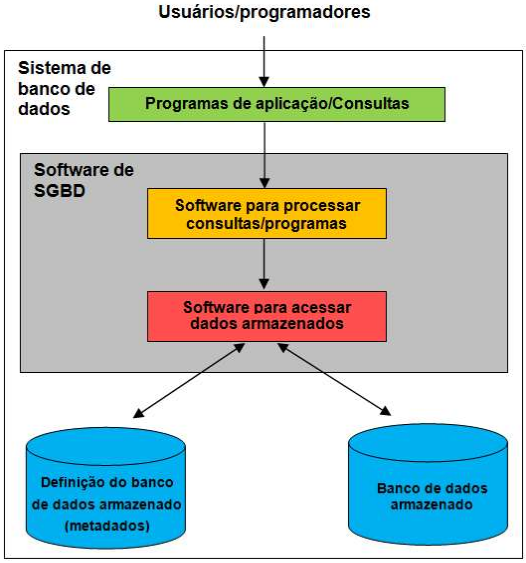
Segundo Elmasri & Navathe (2011), um **sistema gerenciador de banco de dados** (**SGBD –** Database Management System) é uma coleção de programas que permite aos usuários criar e manter um banco de dados. O SGBD é um *sistema de uso geral* que facilita o processo de *definição*, *construção*, *manipulação e compartilhamento* de banco de dados entre diversos usuários e aplicações.

Definir um banco de dados envolve especificar os tipos, estruturas e restrições dos dados a serem armazenados. A definição ou informação descritiva do banco de dados é armazenada pelo SGBD na forma de um catálogo ou dicionário, chamado de **metadados.** O catálogo é usado pelo software de SGBD e também pelos usuários do banco de dados que precisam de informações sobre a estrutura do banco de dados. Um pacote de software de SGBD de uso geral não é escrito para uma aplicação específica. Portanto, ele precisa consultar o catálogo para conhecer a estrutura dos arquivos em um banco de dados específico, como o tipo e o formato de dados que ele acessará.

A **construção** do banco de dados é o processo de armazenar os dados em algum meio controlado pelo SGBD. A **manipulação** de um banco de dados inclui funções como consulta ao banco de dados para recuperar dados específicos, atualização do banco de dados para refletir mudanças no minimundo e geração de relatórios com base nos dados. O **compartilhamento** de um banco de dados permite que diversos usuários e programas acessem-no simultaneamente.

Por convenção, costuma-se chamar a união do banco de dados com o software de SGBD de **sistema de banco de dados.** A Figura 1 ilustra um ambiente de sistema de banco de dados.

Figura 1 – Diagrama simplificado de um ambiente de sistema de banco de dados



Fonte: Elmasri & Navathe (2011)

2. Um exemplo

Como exemplo, considere um banco de dados para manter informações referentes a clientes, contas e agências em um ambiente bancário.

A Figura 2 mostra a estrutura e alguns exemplos de dados para o banco de dados BANCO. O banco está organizado como seis arquivos, e cada um armazena **registros de dados** do mesmo tipo. O arquivo BANCO armazena dados sobre cada banco, o arquivo AGENCIA armazena dados sobre cada agência de um banco, o arquivo CLIENTE armazena dados sobre cada cliente, os arquivos CONTA e TELEFONE_CLIENTE armazenam dados sobre cada conta e telefones de um cliente, respectivamente; e o arquivo HISTORICO armazena as contas de cada cliente.

Para definir esse banco de dados, precisamos especificar a estrutura dos registros de cada arquivo, determinando os diferentes tipos de **elementos de dados** a serem armazenados em cada registro. Na Figura 2 cada registro de BANCO contém os dados que representam o Codigo e Nome (o nome do banco); cada registro de AGENCIA contém Numero_agencia (o número da agência ou código de compensação bancária), Endereco, Cod_banco (o código do banco que a agência pertence); cada registro de CLIENTE contém os dados que representam o Cpf, Nome, Sexo e Endereco; cada registro de CONTA contém os dados que representam o Numero_conta (o número da conta), Saldo, Tipo_conta (como corrente igual a '1', poupança igual a '2', e salário igual a '3') e Num_agencia (a agência que a conta pertence); cada registro de HISTORICO contém os dados que representam o Cpf, o Num_conta (o número da conta do cliente) e Data_inicio (a data de abertura da conta); cada registro de TELEFONE_CLIENTE contém os dados que representam o Cpf_cli e Telefone_cli (Cpf e telefone do cliente).

Observe que mais de um registro de HISTORICO pode ter o mesmo Cpf. Isso significa que um cliente tem mais de uma conta. De modo semelhante, mais de um registro de HISTORICO pode ter o mesmo Num_conta, indicando que uma conta pode ser de propriedade coletiva (conta conjunta).

Figura 2 – Exemplo de um banco de dados que armazena informações de cliente e conta

BANCO

Codigo	Nome
1	Banco do Brasil
2	Itaú
3	Bradesco
4	CEF

AGENCIA

Numero_agencia	Endereco	Cod_banco
3676	Rua Joaquim Teixeira Alves, 1750	3
464	Av. Marcelino Pires, 2830	2
0562	Rua Joaquim Teixeira Alves, 1555	4
4336	Av. Weimar G. Torres, 2965	1
3153	Av. Marcelino Pires, 1960	1

CLIENTE

Cpf	Nome	Sexo	Endereço
444.555.666-77	João B Silva	M	Rua Arapongas, 1234
222.333.444-55	Paulo A Lima	M	Rua Ipiranga, 678
111.222.333-44	Jennifer B Souza	F	Rua Cuiabá, 1050
666.777.888-99	Caetano K Lima	M	Rua Ivinhema, 879
333.888.666-22	Alice Maciel	F	Rua dos Missionários, 1830
555.444.777-33	Silvia Macedo	F	Rua Estados Unidos, 735
999.666.111-88	Robson Soares	M	Rua dos Ingleses, 3245

CONTA

Numero_conta	Saldo	Tipo_conta	Num_agencia	Codigo_banco
62548-6	5468.45	2	4336	1
23584-7	3879.12	1	0562	4
98723-4	6854.96	3	464	2
78963-2	4654.73	1	3676	3
13879-3	20809.67	1	4336	1
86340-2	763.05	2	3153	1
35480-9	8791.20	3	4336	1

HISTORICO

Cpf	Num_conta	Data_inicio
444.555.666-77	98723-4	12-08-1979
444.555.666-77	78963-2	04-03-1980
111.222.333-44	23584-7	17-12-1997
666.777.888-99	23584-7	17-12-1997
222.333.444-55	62548-6	15-01-1994
999.666.111-88	13879-3	03-09-2013
555.444.777-33	86340-2	29-11-2010
333.888.666-22	35480-9	12-04-1985

TELEFONE_CLIENTE

Cpf_cli	Telefone_cli
444.555.666-77	(67)3421-1122
444.555.666-77	(67)3910-3344
444.555.666-77	(67)9999-5566
111.222.333-44	(67)3422-7788
666.777.888-99	(67)3423-9900
666.777.888-99	(67)8121-8833
333.888.666-22	(67)9971-6644
999.666.111-88	(67)3427-2255

Fonte: o autor

Também precisamos especificar um **tipo de dado** para cada elemento de dado de um registro. Por exemplo, podemos especificar que o Nome de CLIENTE é uma sequência de caracteres alfabéticos; Codigo de BANCO é um inteiro, e Sexo de CLIENTE é um único caractere do conjunto {'F','M'}. Também podemos usar um esquema de codificação para representar os valores de um item de dados. Por exemplo, na Figura 2, representamos Tipo_Conta de uma CONTA como 1 para corrente, 2 para poupança e 3 para conta salário.

Para *construir* o banco de dados BANCO, armazenamos dados para representar cada banco, agência, cliente, conta e histórico como um registro no arquivo apropriado. Observe que os registros podem estar relacionados. Por exemplo, o registro para '444.555.666-77' no arquivo HISTORICO está relacionado a dois registros no mesmo arquivo, que especifica as contas de 'João B Silva'. De modo semelhante, o registro para a conta '23584-7' no arquivo HISTORICO, relaciona-se a

dois registros de clientes que são titulares da mesma conta. A maioria dos bancos de dados de tamanho médio e grande inclui muitos registros e possui *muitos relacionamentos* entre os registros.

Um sistema de banco de dados fornece uma **linguagem de definição de dados** (**DDL**) para especificar o esquema de banco de dados e uma **linguagem de manipulação de dados** para expressar as consultas e atualizações de banco de dados. Na prática, as linguagens de definição de dados e de manipulação de dados não são duas linguagens separadas, mas simplesmente formam partes de uma única linguagem de banco de dados, como a amplamente usada linguagem SQL (SILBERSCHATZ, 2006).

A *manipulação* do banco envolve consulta e atualização. Alguns exemplos de consultas são as seguintes:

- Recuperar uma lista de todas as contas de 'João B Silva'.
- Listar os nomes dos clientes que possuem conta na agência '3676'.
- Listar as agências do 'Banco do Brasil'.

Alguns exemplos de atualização incluem:

- Alterar o endereço de cliente de 'João B Silva' para 'Rua Cafelândia, 1350'.
- Criar outra conta para o cliente de 'Caetano K Lima'.
- Inserir uma agência '0391' para 'Banco do Brasil' no endereço 'Rua Joaquim Teixeira Alves, 1796'.

Segundo Heuser (2004), o projeto de um novo banco de dados dá-se em três fases, descritas a seguir.

1. *Modelagem conceitual*

   Nesta primeira fase, é construído um modelo conceitual, na forma de um diagrama entidade-relacionamento. Este modelo captura as necessidades da organização em termos de armazenamento de dados de forma independente de implementação.
2. *Projeto lógico*

   A etapa de projeto lógico objetiva transformar o modelo conceitual obtido na primeira fase em um modelo lógico. O modelo lógico define como o banco de dados será implementado em um SGBD específico.

3. *Projeto físico*

   Na etapa de projeto físico, o modelo do banco de dados é enriquecido com detalhes que influenciam no desempenho do banco de dados, mas não interferem em sua funcionalidade. O modelo obtido neste passo é o modelo físico do banco de dados.

   Alterações neste modelo não afetam as aplicações que usam o banco de dados, já que o modelo não envolve aspectos funcionais do banco de dados. Na prática o projeto físico é um processo contínuo, que ocorre mesmo depois de o banco de dados já estar implementado e em funcionamento. Este processo normalmente é chamado de *sintonia* ("tuning") de banco de dados.

Em outras palavras, o projeto de um banco de dados ocorre usualmente em três etapas. A primeira etapa, a **modelagem conceitual** ou **projeto conceitual**, procura capturar formalmente os requisitos de informação de um banco de dados. (Apresentaremos um modelo denominado Entidade-Relacionamento, no Capítulo 3, que é usado para essa finalidade.) A segunda etapa, o **projeto lógico**, objetiva definir, no nível do SGBD, as estruturas de dados que implementarão os requisitos identificados na modelagem conceitual. (No mercado há vários tipos de SGBDs. Concentramo-nos, nesta disciplina, em um tipo de SGBD que domina o mercado da atualidade, o SGBD relacional[1].) A terceira etapa, o **projeto físico**, define parâmetros físicos de acesso ao banco de dados, procurando otimizar o desempenho do sistema como um todo.

3. Usuários de banco de dados

Segundo Elmasri & Navathe (2011) para um pequeno banco de dados pessoal uma pessoa normalmente define, constrói e manipula o banco de dados, sem compartilhamento. Porém, em grandes organizações, muitas pessoas estão envolvidas no projeto, no uso e na manutenção de um grande banco de dados, com centenas de usuários. Nesta seção, identificamos as pessoas cujas funções envolvem o uso diário de um grande banco de dados.

[1] SGBDs relacionais (SGBDRs) populares atuais incluem o DB2 e Informix Dynamic Server (da IBM), o Oracle e Rdb (da Oracle), o Sybase SGBD (da Sybase) e o SQLserver e Acess (da Microsoft). Além disso, vários sistemas de código aberto, como MySQL e PostgreSQL, estão disponíveis.

3.1 Administradores de banco de dados

Em qualquer organização onde muitas pessoas utilizam os mesmos recursos, há uma necessidade de um administrador principal para supervisionar e gerenciar recursos. Em um ambiente de banco de dados, o recurso principal é o próprio banco de dados, e o recurso secundário é o SGBD e os softwares relacionados. A administração desses recursos é de responsabilidade do **administrador de banco de dados** (**DBA –** *database administrator*). O DBA é responsável por autorizar o acesso ao banco de dados, coordenar e monitorar seu uso e adquirir recursos de software e hardware conforme a necessidade. Também é responsável por problemas como falhas na segurança e demora no tempo de resposta do sistema. Em grandes organizações, ele é auxiliado por uma equipe que executa essas funções.

3.2 Projetista de banco de dados

Os **projetistas de banco de dados** são responsáveis por identificar os dados a serem armazenados e escolher estruturas apropriadas para representar e armazenar esses dados. Essas tarefas são realizadas principalmente antes que o banco de dados esteja realmente implementado e populado com dados. É responsabilidade dos projetistas de banco de dados se comunicar com todos os potenciais usuários a fim de entender suas necessidades e criar um projeto que as entenda. Em muitos casos, os projetistas estão na equipe de DBAs e podem receber outras responsabilidades após o projeto do banco de dados estar concluído. Os projetistas de banco de dados normalmente interagem com cada potencial grupo de usuários e desenvolvem **visões** do banco de dados que cumprem os requisitos de dados e processamento desses grupos. Cada visão é então analisada e *integrada* às visões de outros grupos de usuários. O projeto final do banco de dados precisa ser capaz de atender às necessidades de todos os grupos de usuários.

3.3 Usuários Finais

Os **usuários finais** são pessoas cujas funções exigem acesso ao banco de dados para consultas, atualizações e geração de relatórios. O banco de dados existe primariamente para atender os usuários finais.

3.4 Analistas de sistemas e programadores de aplicações (engenheiros de software)

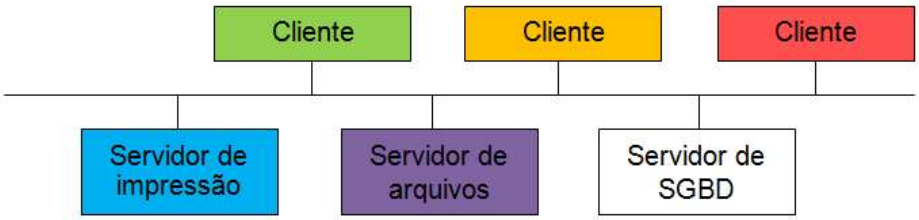
Analistas de sistemas identificam as necessidades dos usuários finais e definem as especificações das transações padrão que atendam a elas. Os **programadores de aplicações** implementam essas especificações como programas; depois, eles testam, depuram, documentam e mantêm essas transações programadas. Esses analistas e programadores – também conhecidos como **engenheiros de softwares** e **desenvolvedores de sistemas de software –** devem estar familiarizados com todo o conjunto de capacidades fornecido pelo SGBD para realizarem suas tarefas.

4 Arquitetura dos sistemas de banco de dados

A arquitetura dos SGBDs tem evoluído desde os primeiros sistemas monolíticos, nos quais todo o software SGBD era um sistema altamente integrado, até os mais modernos, que têm um projeto modular, com arquitetura de sistema cliente/servidor. Essa evolução espelha as tendências na computação, em que grandes computadores *mainframes* centralizados estão sendo substituídos por centenas de estações de trabalho distribuídas e computadores pessoais, conectados por redes de comunicações a vários tipos de máquinas servidoras – servidor Web, servidores de banco de dados, servidores de arquivos, servidores de aplicações, e assim por diante.

Em uma arquitetura básica de SGBD cliente/servidor a funcionalidade do sistema é distribuída entre dois módulos. O **módulo cliente** normalmente é projetado para executar em uma estação de trabalho ou computador pessoal. Em geral, os programas de aplicação e interfaces com o usuário que acessam o banco de dados executam no módulo cliente. Logo, esse módulo se encarrega da interação do usuário e oferece interfaces amigáveis, como formulários ou GUIs (interfaces gráficas do usuário) baseadas em menu. O outro tipo de módulo, chamado **módulo servidor,** é normalmente responsável pelo armazenamento de dados, acesso, pesquisa e outras funções, conforme ilustrado na Figura 3.

Figura 3 – Arquitetura cliente/servidor lógica em duas camadas



Fonte: Elmasri & Navathe (2011)

Os programas da interface com o usuário e os programas de aplicação podem ser executados no lado cliente. Quando é necessário acessar o SGBD, o programa estabelece uma conexão com o SGBD (que está no lado servidor); quando a conexão é criada, o programa cliente pode se comunicar com o SGBD. Um padrão denominado **Conectividade de Banco de Dados Aberta** (**ODBC –** *Open Database Connectivity*) oferece uma **interface de programação de aplicações** (**API –** *Application Programming Interface*), que permite que os programas do cliente chamem o SGBD, desde que as máquinas cliente e servidor tenham o software necessário instalado. A maioria dos vendedores de SGBD oferece drivers ODBC para seus sistemas. Um programa cliente pode se conectar a vários SGBDRs e enviar solicitações de consulta e transação usando a API da ODBC, que são processadas nos servidores. Quaisquer resultados de consulta são enviados de volta ao programa cliente, que pode processar e exibir os resultados conforme a necessidade. Foi definido também um padrão para a linguagem Java, chamado **JDBC.** Isso permite que programas cliente em Java acessem um ou mais SGBDs por meio de uma interface padrão (ELMASRI & NAVATHE, 2011).

4.1 Arquitetura ANSI/SPARC

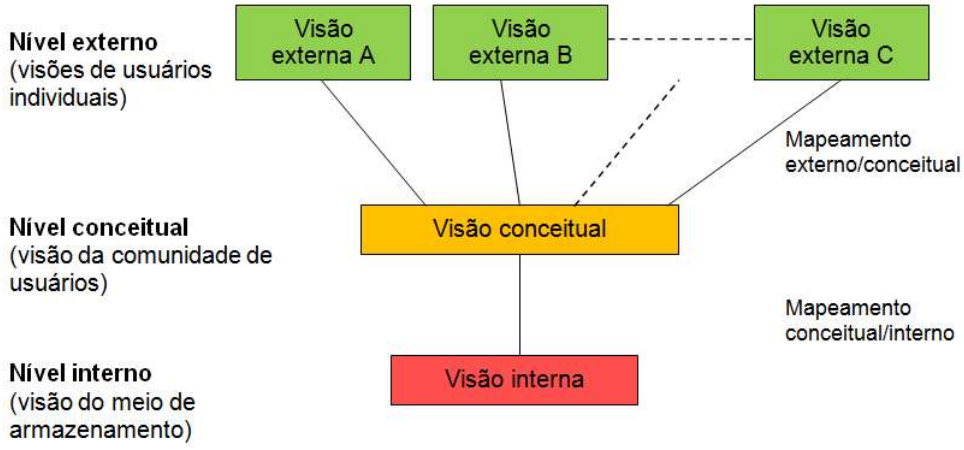
Nesta seção, apresentamos uma arquitetura geral para os sistemas de banco de dados, denominada **ANSI/SPARC**[2], proposta para ajudar a alcançar a **independência dos dados –** capacidade de alterar o esquema de um nível do sistema de banco de dados sem ter de alterar o esquema no nível mais alto.

O objetivo da arquitetura ANSI/SPARC, ilustrada na Figura 4, é separar as aplicações do usuário do banco de dados físico. Conforme Date (2004), a arquitetura ANSI/SPARC se divide em três níveis, a seguir:

[2] A arquitetura ANSI/SPARC também é conhecida como arquitetura de três esquemas e possui o mesmo nome do comitê que a propôs (TSICHRITZIS & KLUG, 1978).

1. O **nível interno** (também conhecido como nível de *armazenamento*) é o mais próximo do meio de armazenamento físico – ou seja, é aquele que se ocupa do modo como os dados são fisicamente armazenados dentro do sistema.
2. O **nível externo** (também conhecido como nível *lógico* do *usuário*) é o mais próximo dos usuários – ou seja, é aquele que se ocupa do modo como os dados são vistos por usuários individuais.
3. O **nível conceitual** (também conhecido como nível *lógico de comunidade*, ou às vezes apenas nível *lógico*, sem qualificação) é um nível "indireto" entre os outros dois.

Figura 4 – Os três níveis da arquitetura



Fonte: adaptado de Date (2004)

Observe que o nível externo se preocupa com as percepções dos usuários individuais, enquanto o nível conceitual está preocupado com uma percepção da *comunidade* de usuários. De modo geral, a maior parte dos usuários não estará interessada no banco de dados inteiro, mas somente em alguma parte restrita dele; assim, haverá muitas "visões externas" distintas, cada qual consistindo em uma representação mais ou menos abstrata de alguma parte do banco de dados completo, e haverá exatamente uma "visão conceitual", consistindo em uma representação igualmente abstrata do banco de dados em sua totalidade. Do mesmo modo, haverá exatamente uma "visão interna", representando o modo como o banco de dados está armazenado internamente. Observe que os níveis externo e conceitual são níveis de *modelo*[3], enquanto o nível interno é um nível de *implementação*[4]; em outras

[3] Um **modelo de dados** é uma definição abstrata, autônoma e lógica dos objetos, operadores e outros elementos que, juntos constituem a *máquina abstrata* com a qual os usuários interagem. Os objetos nos permitem modelar a *estrutura* dos dados. Os operadores nos permitem modelar seu *comportamento* (DATE, 2004).

palavras, os níveis externo e conceitual são definidos em termos de construções voltadas para o usuário, como registros e campos, enquanto o nível interno é definido em termos de construções voltadas para a máquina, como bits e bytes (DATE, 2004).

A arquitetura ANSI/SPARC é uma ferramenta com a qual o usuário pode visualizar os níveis de esquema em um sistema de banco de dados. A maioria dos SGBDs não separa os três níveis de esquema completa e explicitamente, mas dá suporte a eles de alguma forma. A arquitetura ANSI/SPARC de três níveis tem um lugar importante no desenvolvimento da tecnologia de banco de dados, pois separa, claramente, o nível externo dos usuários, o nível conceitual do banco de dados e o nível de armazenamento interno no projeto de um banco de dados. Ainda hoje ela é bastante aplicável no projeto de SGBDs. Na maioria dos SGBDs que têm suporte a visões do usuário, os esquemas externos são especificados no mesmo modelo de dados que descreve a informação no nível conceitual (por exemplo, um SGBD relacional como o MySQL utiliza SQL para isso).

Note que os três esquemas são apenas *descrições* dos dados; os dados armazenados que realmente existem estão apenas no nível físico. Em um SGBD baseado na arquitetura ANSI/SPARC, cada grupo de usuários recorre ao seu próprio esquema externo. Assim, o SGBD precisa transformar uma solicitação especificada em um esquema externo em uma solicitação no esquema conceitual, e depois em uma solicitação no esquema interno para o processamento no banco de dados armazenado. Se a solicitação for uma recuperação, os dados extraídos do banco de dados armazenado devem ser reformatados para corresponder à visão externa do usuário. Os processos de transformação de requisições e os resultados entre os níveis são chamados de **mapeamentos.** Esses mapeamentos são necessários para transformar solicitações entre os níveis e ajudar a alcançar a independência lógica[5] e a independência física[6] dos dados.

---

[4] Uma **implementação** de um determinado modelo de dados é uma representação física em uma máquina real dos componentes da máquina abstrata que, juntos, constituem esse modelo (DATE, 2004).

[5] *Independência lógica de dados* é a capacidade de alterar o esquema conceitual sem ter de alterar os esquemas externos de aplicação. Por exemplo, podemos modificar o esquema conceitual para alterar restrições.

[6] *Independência física de dados* é a capacidade de alterar o esquema interno sem ter de alterar o esquema conceitual ou os esquemas externos. Por exemplo, podemos criar índices para melhorar o

Sempre que temos um SBGD de múltiplos níveis, seu catálogo deve ser expandido para incluir informações sobre como mapear solicitações e dados entre os diversos níveis. O SGBD usa software adicional para realizar esses mapeamentos, recorrendo à informação de mapeamento no catálogo. A independência de dados ocorre porque, quando o esquema é alterado em algum nível, o esquema no próximo nível mais alto permanece inalterado; somente o *mapeamento* entre os dois níveis é alterado. Logo, os programas de aplicação que fazem referência ao esquema de nível mais alto não precisam ser alterados.

5. Modelo de dados relacional

Esta seção aborda os bancos de dados relacionais. O modelo de dados relacional foi introduzido inicialmente por Ted Codd, da IBM Research, em 1970, em um artigo clássico (Codd, 1970), que atraiu atenção imediata devido a sua simplicidade e base matemática. O modelo usa o conceito de *relação matemática* – que se parece com uma tabela de valores – como seu bloco de montagem básico, e sua base teórica reside em uma teoria de conjunto e lógica de predicado de primeira ordem. Nesta seção, discutiremos as características básicas do modelo e suas restrições.

Por causa da importância do modelo relacional, toda a disciplina Banco de Dados I é dedicada a esse modelo e alguma das linguagens associadas a ele. No Capítulo 2, descreveremos a linguagem de consulta SQL, que é o *padrão* para SGBDs relacionais comerciais. Outros aspectos do modelo relacional são apresentados em outras partes da Mídia Integrada Convergente. O Capítulo 3 abordará as estruturas de dados do modelo relacional para construções do modelo ER, e apresentará algoritmos para projetar um esquema de banco de dados relacional mapeando um esquema conceitual no modelo ER para uma representação relacional. Esses mapeamentos são incorporados em muitas ferramentas de projeto de banco de dados e CASE[7]. O Capítulo 4 apresentará outro aspecto do modelo relacional, a saber, as restrições formais das dependências funcionais e multivaloradas. Essas

---

tempo de resposta de uma consulta sem que seja preciso alterá-la, embora a consulta seja executada com mais eficiência pelo SGBD ao utilizar o novo caminho de acesso.

[7] CASE significa Computer-Aided Software Engineering (engenharia de software auxiliada por computador).