



# Banco de dados I



# **Banco de dados I**

Leonardo de Marchi Ferrareto  
Roberto Yukio Nishimura

© 2018 por Editora e Distribuidora Educacional S.A.

Todos os direitos reservados. Nenhuma parte desta publicação poderá ser reproduzida ou transmitida de qualquer modo ou por qualquer outro meio, eletrônico ou mecânico, incluindo fotocópia, gravação ou qualquer outro tipo de sistema de armazenamento e transmissão de informação, sem prévia autorização, por escrito, da Editora e Distribuidora Educacional S.A.

**Presidente**

Rodrigo Galindo

**Vice-Presidente Acadêmico de Graduação**

Mário Ghio Júnior

**Conselho Acadêmico**

Alberto S. Santana

Ana Lucia Jankovic Barduchi

Camila Cardoso Rotella

Cristiane Lisandra Danna

Danielly Nunes Andrade Noé

Emanuel Santana

Grasiele Aparecida Lourenço

Lidiâne Cristina Vivaldini Olo

Paulo Heraldo Costa do Valle

Thatiane Cristina dos Santos de Carvalho Ribeiro

**Revisão Técnica**

Márcio Aparecido Artero

**Editorial**

Adilson Braga Fontes

André Augusto da Andrade Ramos

Cristiane Lisandra Danna

Diogo Ribeiro Garcia

Emanuel Santana

Erick Silva Griepe

Lidiâne Cristina Vivaldini Olo

---

**Dados Internacionais de Catalogação na Publicação (CIP)**

---

F375b Ferrareto, Leonardo De Marchi  
Banco de dados I / Leonardo De Marchi Ferrareto,  
Roberto Yukio Nishimura. – Londrina : Editora e Distribuidora  
Educacional S.A., 2018.  
136 p.

ISBN 978-85-522-0291-2

1. Banco de dados – gerência. 2. Internet – programas de  
computador. I. Nishimura, Roberto Yukio. II. Título.

---

CDD 005.3

---

2018

Editora e Distribuidora Educacional S.A.  
Avenida Paris, 675 – Parque Residencial João Piza  
CEP: 86041-100 – Londrina – PR  
e-mail: editora.educacional@kroton.com.br  
Homepage: <http://www.kroton.com.br/>

# Sumário

<b>Unidade 1   Fundamentos e estrutura de banco de dados</b>	<b>9</b>
<b>Seção 1 - Fundamentos</b>	
1.1   Visão geral	12
1.2   Dado, informação e conhecimento	12
1.3   O valor da informação	13
1.4   Fundamentos	15
1.5   Emprego de um banco de dados	16
1.5.1   Natureza autodescritiva	17
1.5.2   Isolamento dos dados e abstração de dados	18
1.5.3   Múltiplas visões de dados	19
1.5.4   Compartilhamento de dados e transações distribuídas	19
1.6   Usuários	20
1.6.1   Administradores de Banco de Dados (DBA)	20
1.6.2   Desenvolvedores	20
1.6.3   Usuários	20
1.7   Dados Operacionais	21
1.8   Por que usar banco de dados	22
<b>Seção 2 - Estrutura de dados</b>	
2.1   Estruturas de Dados	25
2.2   Tipos de Dados	26
2.2.1   Char, Varchar, nchar e nvarchar	26
2.2.2   Int, bigint, smallint, tinyint e bit	27
2.2.3   Decimal e Numeric	27
2.2.4   Date, datetime e smalldatetime	28
2.3   Níveis de Abstração de dados	28
<b>Unidade 2   Sistemas de gerenciamento de banco de dados (SGBD)</b>	<b>35</b>
<b>Seção 1 - Sistema de Gerenciamento de Banco de Dados (SGBD), ou Sistema Gerenciador de Banco de Dados</b>	
1.1   Definição	38
<b>Seção 2 - Ferramentas de administração do banco de dados</b>	
2.1   Definição	43
2.2   Linguagem de acesso	44
2.2.1   Linguagem de Definição de Dados (DDL)	44
2.2.2   Linguagem de Manipulação de Dados (DML)	44
2.2.3   Linguagem de Controle de Dados (DCL)	45
2.3   Ferramentas para administração do banco de dados	45
2.3.1   Como são as ferramentas para administração do banco de dados	45
<b>Seção 3 - Instância, esquema e independência de dados</b>	
3.1   Instância de Banco de Dados	49
3.1.1   Definição	49
3.2   Esquema de Banco de Dados	51
3.2.1   Definição	51
3.3   Independência de Dados	55
3.4   Independência Lógica de Dados	57
3.5   Independência Física de Dados	58
3.6   Propriedades ACID de um banco de dados	59

3.6.1   Propriedade Atomicidade	60
3.6.2   Propriedade Consistência	60
3.6.3   Propriedade Isolamento	60
3.6.4   Propriedade Durabilidade	61
<b>Unidade 3   Modelo de dados e modelagem entidade relacionamento</b>	<b>67</b>
<b>Seção 1 - Modelo de dados: conceitos, elementos e funcionalidades</b>	<b>70</b>
1   Modelo de dados	70
1.1   Definição	70
1.2   Projeto Conceitual	72
1.3   Projeto Lógico	72
1.4   Projeto Físico	73
<b>Seção 2 - Modelo Entidade Relacionamento – MER</b>	<b>76</b>
2   Ferramentas de Administração do Banco de Dados	76
2.1   Definição	76
2.2   Entidade	76
2.3   Relacionamento	78
2.3.1   Tipos de Relacionamento	79
2.3.2   AutoRelacionamento	79
2.3.3   Integridade Referencial	80
2.4   Atributo	81
<b>Seção 3 - Cardinalidades e Tipos de Atributos</b>	<b>84</b>
3   Cardinalidades e Tipos de Atributos	84
3.1   Cardinalidades	84
3.1.1   Definição	84
3.1.2   Cardinalidade 1 - 1	84
3.1.3   Cardinalidade 1 - N	85
3.1.4   Cardinalidade N - N	85
3.1.5   Cardinalidade N - 1	86
3.1.6   Sentido da Leitura	86
3.1.7   Cardinalidade mínima	87
3.2   Tipos de Atributos	87
3.2.1   Definição	87
3.2.2   Tipo Determinante	88
3.2.3   Tipo Composto	89
3.2.4   Tipo Multivalorado	89
3.2.5   Tipo Calculado	90
<b>Unidade 4   Linguagem de acesso a dados</b>	<b>97</b>
<b>Seção 1 - Linguagem de banco de dados</b>	<b>100</b>
1.1   Linguagem de banco de dados	100
1.1.1   DDL (Data Definition Language) – Linguagem de definição de dados	101
1.1.2   DML (Data Manipulation Language) – Linguagem de manipulação de dados	102
1.1.3   DQL (Data Query Language) – Linguagem de consulta de dados	102
1.1.4   DTL (Data Transaction Language) – Linguagem de transação de dados	103
1.1.5   DCL (Data Control Language) - Linguagem de controle de dados	104
<b>Seção 2 - Comandos SQL</b>	<b>107</b>
2.1   Comandos DDL	107
2.1.1   CREATE TABLE	107
2.1.2   ALTER TABLE	109
2.1.3   DROP TABLE	112
2.1.4   PRIMARY KEY – Chave Primária	113
2.1.5   FOREIGN KEY – Chave Estrangeira	115

2.16   Not Null – Não Nulo	117
2.17   CHECK	118
2.18   UNIQUE	118
2.19   DEFAULT	120
2.2   Operadores	120
2.2.1   Comparação	120
2.2.2   Lógicos e de Conjunto	121
2.3   Comando SELECT	121
2.4   Comando INSERT	122
2.5   Comando DELETE	123
2.6   Comando UPDATE	123



# Apresentação

Este material didático possui a função de apresentar ao estudante do curso de Análise e Desenvolvimento de Sistemas os conceitos relacionados a banco de dados.

Conheceremos a fundo os principais conceitos sobre banco de dados e os componentes que o envolvem, como dado, informação e conhecimento.

O presente material não apresentará todos os conceitos relacionados a banco de dados, visto que, dependendo do uso do sistema de gerenciamento de banco de dados, cada fabricante pode ter sua particularidade implementada na ferramenta, portanto abordará os principais conceitos baseados no padrão ANSI.

Através das Unidade 1 e 2, conseguiremos visualizar os fundamentos que envolvem a construção de um banco de dados e os objetos dos quais ele é criado, bem como a aplicação que fará o gerenciamento do sistema de banco de dados.

Já na Unidade 3, entenderemos o modelo criado para a construção de um banco de dados relacional, todas as regras de modelagem e os conceitos que serão utilizados para a construção de um modelo lógico de dados.

Por fim, na Unidade 4, entraremos no universo da linguagem SQL, conhecendo os grupos de comandos da linguagem e nos familiarizando com os principais comandos do padrão ANSI.

Através desta estrutura de aprendizagem, esperamos garantir ao aluno um início sólido e seguro, criando uma base capaz de ajudá-lo em outros futuros desafios, relacionados ao uso de banco de dados.

Bons estudos!



# Fundamentos e estrutura de banco de dados

*Leonardo de Marchi Ferrareto*

## Objetivos de aprendizagem

- Entender sobre os conceitos iniciais.
- Entender as estruturas.
- Compreender sobre as diferenças entre banco e sistema de banco.
- Visualizar todos os componentes.

## Seção 1 | Fundamentos

Serão apresentados os componentes importantes para o conhecimento de banco de dados, como: dado, informação e conhecimento; valor da informação; fundamentos, como confiabilidade, integridade, visões, segurança, interface, independência de dados, autodescrição e concorrência; capacidade de rodar de forma distribuída e alta performance; em que o banco de dados é empregado; usuários de um banco de dados; e dados operacionais, sendo cada um deles descritos de forma clara e objetiva, mostrando sua importância dentro da aplicação.

## Seção 2 | Estrutura de dados

Apresentação dos conceitos de estrutura de dados e níveis de abstração de dados.



# Introdução à unidade

Nesta unidade, iremos abordar os principais conceitos relacionados a banco de dados, tudo que precisamos entender para poder compreender de maneira clara a importância do uso do banco de dados.

Na Seção 1, abordaremos os conceitos de dado, informação e conhecimento, valor da informação, fundamentos sobre o banco de dados, emprego de banco de dados e suas características, conheceremos os usuário envolvidos no banco de dados e compreenderemos o que são dados operacionais. Na Seção 2, estudaremos as estruturas de dados, os tipos de dados e a abstração de dados, sendo esta a forma que o banco de dados é apresentado aos usuários, e como ele se divide internamente. Todos esses conceitos são essenciais para que possamos entender e aproveitar melhor o banco de dados.

# Seção 1

## Fundamentos

### Introdução à seção

Vamos entender os principais conceitos, bem como seus componentes, e avaliar o papel de cada um, ver a importância que a informação tem para o usuário que manipula seus dados para gerar informação. Qual é o valor desta informação? Por que o usuário necessita tanto assim desta informação? Analisando toda essa importância, teremos a possibilidade de entender que as regras de um banco de dados visam, basicamente, trabalhar a informação para o usuário.

### 1.1 Visão geral

A busca pela informação, tanto no âmbito comercial como no pessoal, cresce demasiadamente a cada dia, e esse crescimento traz a necessidade de armazenamento e manipulação de dados. Toda manipulação de dados tem um único fim: gerar informação sobre um determinado assunto ou negócio.

Para suprir tal necessidade de armazenamento e, posteriormente, de manipulá-los, foi desenvolvido um banco de dados capaz de sanar essa necessidade.

Antes de conhecermos mais sobre banco de dados, vamos entender melhor as definições de dado, como é gerada a informação e de onde vem o conhecimento.

### 1.2 Dado, informação e conhecimento

Segundo Mülbert e Ayres (2007, p. 22), “dados é um conjunto de caracteres que, organizados entre si, não transmitem nenhum tipo de informação, é a forma bruta de toda a informação”. Sozinho, esse elemento não traz nenhuma informação, não mostra nada, apenas um pedaço da informação que será gerada quando combinarmos diversos dados, assim terá um sentido. Podemos, então, resumir que dado é uma parte da informação essencial para que ela seja gerada e compreendida.

Já informação origina-se do latim *informare*, ou dar forma. Conforme Mülbert e Ayres (2007, p. 23),

é um conjunto de dados registrados, organizados e ordenados que tem um significado lógico, transmitem uma mensagem, portanto informação é um conjunto de dados com sentido lógico, são dados ordenados.

Quando combinamos dados, temos uma informação. E quando combinamos informação?

De acordo com Mülbert e Ayres (2007, p. 23):

Já o conhecimento, tem praticamente o mesmo conceito de informação, porém quando desejamos ter um conhecimento, combinamos informação sobre vários assuntos. Então conhecimento é quando combinamos a informação disponível de um determinado fato ou assunto.

Uma forma simples de exemplificar um conhecimento é analisarmos uma biblioteca, na qual temos setores divididos por áreas, e os livros são as informações que precisamos obter para gerar o conhecimento sobre uma área/assunto que estamos necessitando apreender.

Figura 1.1 | Papel da informação dentro da empresa



Fonte: Laudon e Laudon (1999, p. 27).

### 1.3 O valor da informação

Quanto vale a informação de uma empresa atualmente? Como podemos analisar o valor de toda a informação gerada por uma empresa?

Podemos dizer que a informação que uma empresa possui é um ativo dela. Segundo Dantas (2011, p. 21),

**todo o ativo da empresa compreende um conjunto de bens, portanto isso acaba sendo ampliado, pois, hoje, a informação tem um valor para a empresa, sendo também um ativo.**

A informação tem ocupado um destaque nas empresas, e também tem adquirido um potencial de valoração para as organizações.

Proteger a informação é uma maneira de zelar pelo ativo da empresa, para o negócio, e maximizar o retorno sobre o investimento e as oportunidades de negócio.

Conforme a norma ABNT NBR ISO/IEC 27002:2005:

**Mostra que pode existir várias formas de informação, podendo ser impressa, armazenada em modo eletrônico, transmitidas pelos meios comuns de correio ou por meios eletrônicos, pode ser coletada através de vídeos como filmes ou até em conversas. Independente da maneira que essa informação é coletada, devemos protegê-la adequadamente.**

Para que a empresa use uma informação como ativo, deve-se analisar se esta respeita algumas características. Stair (1998, p. 6) demonstra essas características na tabela a seguir.

Tabela 1.1 | Características da Informação

Característica	Definição
Precisa	Tem que ser clara e com procedência. Quando uma informação é gerada de forma errada, imprecisa, significa que a entrada de dados foi feita de maneira errada.
Completa	Quando ela nos remete a tudo aquilo que procuramos, todos os fatos e acontecimentos que precisamos saber quando solicitamos a sua geração.
Econômica	Quando a informação traz dados para os tomadores de decisão, assim eles podem usar e analisar a informação gerada para administrar e tomar decisões.
Flexível	Quando gerada a informação produzida, pode ser utilizada em diversos ambientes ou por diversos usuários, sua finalidade não acaba sendo somente um único destino.
Confiável	Verificar se a fonte desta informação possui dados que foram analisados e coletados de forma correta, respeitando a precisão da informação.
Relevante	Leva ao tomador da informação aquilo que realmente é necessário para a condução do seu negócio.

Simples	Essa característica mostra que não se deve exagerar no conteúdo produzido pela informação, deve-se levar somente o que é pedido. Um exemplo disso é quando há a necessidade de um relatório de contas a pagar e junto é enviada a informação de data de aniversário do fornecedor. Para que essa informação foi útil para a condução do negócio? Simplicidade tende a ser somente a informação de contas a pagar.
Em tempo	Disponível quando necessário, usuário ou tomador de decisão não pode esperar para ter informação para conduzir um negócio, ela tem que estar à disposição em um curto prazo de tempo e não levar horas ou dias para produzi-la.
Verificável	Checkar se a fonte da informação é de origem certa, se os dados produzidos para gerar a informação foram analisados e confirmados.

Fonte: Stair (1998, p. 6).

As informações ajustam-se rapidamente às mudanças. É necessário que elas tenham qualidade, sejam oportunas e confiáveis. O seu valor pode ser medido pela maneira como é eficaz para as empresas no uso de tomada de decisão.

## 1.4 Fundamentos

Consoante Silberschatz e Korth (2006, p. 2):

Quando temos uma coleção de dados inter-relacionados gerando informação sobre um determinado assunto, domínio, podemos dizer que temos um banco de dados, lembrando que essas informações geradas, sejam sempre de um domínio específico, outros autores, definem de uma forma diferente, mas todos com uma ideia de coleção ou conjunto de dados que servem ou não em algumas situações específicas.



Algumas considerações sobre banco de dados:

- Não podemos definir banco como um conjunto aleatório de dados.
- Os dados são fatos, pedaços de uma informação de um assunto ou domínio específico.
- O banco de dados é essencial em sistemas de controle e gerenciamento. Um simples sistema de cadastro de clientes ou vendas necessita de um banco para armazenar os dados e gerar informações sobre o negócio.
- O armazenamento dos dados permite o compartilhamento e

a manutenção de dados consistentes por diversas aplicações.

Segundo Elmasri e Navathe (2005, p. 3), “a sociedade ficou dependente de informação, seja em qualquer momento do dia a dia, a necessidade de armazenamento de dados em um sistema de banco de dados torna-se essencial”.

Pode-se notar a presença do seu uso em nosso dia a dia, como: retirar um extrato bancário, acessar uma rede social, realizar uma compra em um supermercado, entre outros exemplos. Muitas dessas atividades envolvem o seu uso como forma de armazenamento e geração de informação, como no caso mencionado de uma compra em um supermercado, na qual os itens que são registrados em um cupom fiscal são resgatados do banco, na sua maioria, através do código de barras, o qual é a chave de identificação de um determinado produto, em que estão armazenadas suas características, bem como seu preço. Cada compra realizada também é armazenada em banco de dados, pois são dados essenciais para a geração de relatórios gerenciais ou fiscais.

Para que os bancos de dados possam ser úteis, eles devem oferecer:

- Confiabilidade.
- Integridade.
- Visões.
- Segurança.
- Interface.
- Independência de dados.
- Autodescrição.
- Concorrência.
- Capacidade de rodar de forma distribuída.
- Alta performance.

## 1.5 Emprego de um banco de dados

Existem características que diferem as aplicações que utilizam banco daquelas que, tradicionalmente, utilizam a programação e os arquivos.

Segundo Elmasri e Navathe (2005, p. 7):

Tradicionalmente o processamento de arquivos é definido quando o usuário programa uma aplicação, com isso o processamento de arquivos fica vinculado como parte específica da aplicação. Por exemplo, um usuário final, pode manter dados em arquivos de alunos de uma faculdade, sendo esses dados como notas ou presença, a aplicação pode imprimir um histórico do aluno e adicionar novos dados como notas, presença, solicitações de documentos. Um outro usuário de um departamento financeiro, pode controlar a vida financeira como as mensalidades e datas de pagamentos realizados pelos alunos.

Nesta definição, todos os usuários têm um único objetivo: os dados produzidos pelo sistema. Cada um mantém suas informações em arquivos separados, e os programadores manipulam esses arquivos. Essa redundância na definição e no armazenamento dos dados cria um problema de espaço de armazenamento desperdiçado, pois há uma redundância desnecessária dos dados, ocasionando também um problema grave na manutenção dos dados, pois muitos podem ficar desatualizados.

Quando utilizamos um único repositório de dados, temos concentrados todos os dados produzidos pelos usuários em um lugar só.

Vejamos algumas das principais características do banco de dados em relação ao processamento de arquivos:

- Natureza autodescritiva.
- Isolamento e abstração de dados.
- Múltiplas visões de dados.
- Compartilhamento de dados e transações distribuídas.

### 1.5.1 Natureza autodescritiva

Conforme Elmasri e Navathe (2005, p. 7):

Quando abordamos banco de dados, uma característica fundamental não é apenas o banco por si só, é também uma série de outros fatores que estruturam esse banco e todas as regras de restrições que o mesmo possui. Esse catálogo complexo é armazenado em um sistema de

gerenciamento chamado de sistema de gerenciamento de banco de dados, SGBD, o qual possui toda a estrutura, tipo e formato de tudo que está armazenado bem como suas restrições sobre cada dado.

O descrito em um banco de dados é usado pelos usuários que precisam das informações sobre a estrutura descrita dele. O pacote de software de um banco de dados tem um propósito geral: nele não vem descrita uma aplicação específica, portanto é necessário acessar o catálogo do banco de dados em questão para conhecer e avaliar a sua estrutura, como os tipos e o formato dos dados.

No processamento tradicional através de arquivos, o próprio programa gera a definição dos dados a serem utilizados pelo usuário, isso tudo é muito restrito ao programa.

### 1.5.2 Isolamento dos dados e abstração de dados

Na forma tradicional, toda alteração na estrutura de arquivos de dados está embutida nos programas de acesso a este arquivo.

Os bancos de dados possuem uma propriedade de independência de dados, que é o isolamento entre os dados e os programas que os acessam. Quando, neste caso, uma estrutura de dados é alterada, não há nenhuma demanda de alteração nos programas que o acessam, isso é possível porque as camadas são separadas.

Elmasri e Navathe (2005, p. 8) nos mostram que:



Os bancos de dados orientado a objeto e objeto-relacional, fazem suas operações sobre os dados como parte na definição dos mesmos, uma operação de chamada de função ou até mesmo um método é em duas partes. Toda operação inclui seu nome e os tipos dos dados levam consigo seus argumentos. Todo método é definido separadamente e sua alteração não afeta a interface. As aplicações dos usuários operam os dados chamando suas operações por argumentos desconsiderando as operações implementadas, esse acontecimento chama-se de independência programa operação.

A característica que permite a independência entre programa e dados e programa e operação é definida como abstração de dados.

### 1.5.3 Múltiplas visões de dados

Elmasri e Navathe (2005, p. 9) afirmam:

Um banco de dados típico tem muitos usuários, e cada qual pode solicitar diferentes perspectivas ou visões do banco de dados. Uma visão pode ser um subconjunto de um banco de dados ou conter uma visão virtual dos dados, derivados dos arquivos do banco de dados, mas não, explicitamente, armazenados. Alguns usuários podem não saber se os dados a que eles se referem são armazenados ou derivados.

Este suporte permite que os usuários tenham consultas especializadas de parte dos dados de um banco, com isso também temos a segurança da informação que será apresentada, podendo distribuir os dados somente para os usuários que têm direito de visão.

### 1.5.4 Compartilhamento de dados e transações distribuídas

Como o nome mesmo implica, devemos permitir que vários usuários tenham acesso, ao mesmo tempo, aos dados. O compartilhamento traz a vantagem de que não é necessário manter o mesmo dado para distintas operações que o usam.

Para garantir que vários usuários realizem operações de manipulação dos dados, o software de banco de dados deve possuir um controle de concorrência. Por exemplo, quando muitos vendedores realizam operações de venda ou emissão de pedidos, por exemplo, devemos garantir que cada número de pedido seja associado ao seu cliente, no caso, quando emitir o pedido de número 100, este deve ser de posse, única e exclusivamente, do cliente que está associado a ele. Esse tipo de aplicação é denominado de aplicações de processamento de transação.

A transação tornou-se fundamental em aplicações de banco de dados pois ela controla toda a execução dos processos acionados pelos programas, como inclusão de dados,

alteração e leitura. Tudo isso deve ser controlado de forma correta para que cada transação não interfira em uma outra transação em execução. (ELMASRI; NAVATHE, 2005, p. 9)

O isolamento da transação deve ser de forma que nenhuma transação possa impactar sobre outra transação, mesmo acontecendo dezenas e dezenas de transações simultaneamente.

## 1.6 Usuários

Podemos dividir os utilizadores do banco de dados em três grupos:

- Administrador de Banco de Dados (DBA).
- Desenvolvedores.
- Usuários.

### 1.6.1 Administradores de Banco de Dados (DBA)

É o responsável por manter e zelar por todas as bases de dados. Também controla toda a parte de segurança e controle de acesso, concedendo aos usuários as permissões que cada um pode ter na manipulação e no acesso aos dados.

Tem um papel essencial para que o banco de dados possa garantir acesso rápido e eficaz, analisando todos os objetos e programas dentro dele, e também realizando rotinas de backup e restauração para evitar perdas em caso de falha.

### 1.6.2 Desenvolvedores

São responsáveis por implementar e modelar a base de dados. Seu principal papel é desenvolver aplicações (programas escritos em outras linguagens, como C#, PHP, Java, entre outras), conectando ao banco de dados do sistema, podemos, assim, tornar o acesso e a manipulação de dados mais simples ao usuário final.

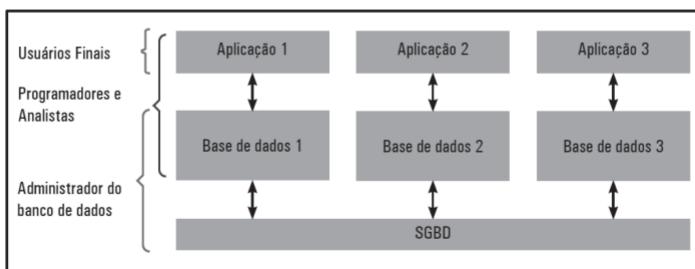
### 1.6.3 Usuários

Dentre os três grupos apresentados, este é o mais importante, pois tem um papel fundamental dentro do banco de dados, pois é ele quem

dá vida aos dados, gera a informação e faz manutenção nos dados de forma que fiquem sempre atualizados.

Esse grupo trabalha diariamente com a aplicação desenvolvida pelos desenvolvedores, sendo responsável por dar entrada de dados e alterar os já existentes, produzindo novas informações. Esses usuários não precisam ter nenhum conhecimento sobre o banco de dados, pois, para eles, ele já é transparente, só interessa a informação que está salva nele e que possam recuperá-la.

Figura 1.2 | Acesso dos usuários ao banco de dados



Fonte: Angelotti (2010, p. 14).

## 1.7 Dados Operacionais

São os dados existentes no banco de dados, os quais não incluem dados de entrada ou saída, filas de trabalho ou, sem dúvida, qualquer informação puramente transiente. Segundo Date (1984, p. 30), “os dados de entrada, podem provocar modificações nos dados operacionais do banco de dados, eles entram no banco através do mundo exterior, com uma simples alteração em um dado ou a inclusão de outro novo, mudando assim a informação gerada”. Já os dados de saída são os relatórios e as informações gerados pelo sistema através dos dados contidos no banco de dados, mas por si só não fazem parte do banco de dados.

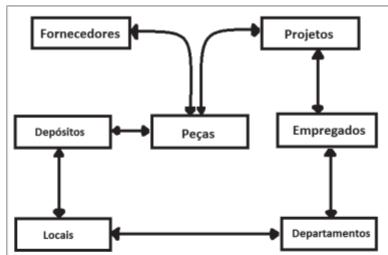
Vamos considerar o caso de uma companhia de manufatura, com um pouco mais de detalhes. Vejamos a seguir a necessidade que a empresa tem em uma operação e a exemplificação na figura:

- Manter os dados dos projetos realizados pela empresa.
- Manter os dados das peças usadas em cada projeto.
- Manter os dados dos empregados que trabalham em cada projeto.
- Manter os depósitos onde estão localizadas as peças.

- Manter os locais de cada depósito e departamento.
- Manter os departamentos que cada empregado está vinculado.
- Manter os fornecedores de cada peça usada pelo projeto.

Um analista, ao se deparar com essa situação, poderá desenhar uma solução sobre a situação desejada pela empresa.

Figura 1.3 | Exemplo de dados operacionais



Fonte: elaborada pelo autor.

Essa representação demonstra os inter-relacionamentos dos dados, mantidos de acordo com cada necessidade especificada pelo cliente. Os dados mantidos neste esquema, quando ligados entre si, relacionados, poderão gerar informações, como uma nota fiscal de entrada, um relatório de empregados por projetos, entre outros dados. Isso exemplifica os dados de saída gerados pelos dados operacionais do sistema. Os dados gerados não estão mantidos no banco de dados, eles foram criados quando relacionados aos dados operacionais.

### 1.8 Por que usar banco de dados

A grosso modo, Date (1984, p. 31) justifica isso de uma forma simples e clara:



**o uso do banco de dados permite que toda as operações sobre os dados sejam centralizadas, permitindo que a empresa tenha um controle dos dados operacionais, que hoje é um dos seus principais ativos, o mais valioso.**

Podemos, então, pontuar algumas vantagens na centralização dos dados:

- **Controle de redundância:** manter somente o que é necessário para que o sistema funcione corretamente, evitando armazenamento desnecessário. O uso do inter-relacionamento dos dados mantidos pelo sistema o torna mais eficaz e livre dessa redundância. Há casos

em que a redundância é necessária, como em um sistema de Business Intelligence.

- **Inconsistência:** são dados sem sentido, mantidos sem a necessidade, pois não têm relacionamento e não são capazes de produzir informação. Quando encontrada a inconsistência de dados, é um sinal que alguma coisa grave aconteceu, corrompeu a base, sendo necessário recuperá-la.
- **Compartilhamento:** como visto no subtítulo 1.5.4, significa que todas as aplicações existentes podem compartilhar os dados do banco, como também que as novas aplicações podem usar esses dados já existentes, não há necessidade de novos dados para novas aplicações.
- **Padrões criados no banco de dados:** com o controle centralizado, o DBA do banco de dados pode criar padrões de acordo com as regras específicas em cada empresa ou companhia. A padronização pode facilitar o intercâmbio de dados ou a migração dos sistemas.
- **Restrições de segurança:** as restrições ajudam a manter o nível de segurança sobre os dados armazenados no banco, e até mesmo os dados de saída gerados pelo banco. O DBA, usuário responsável em manter e administrar, pode conceder ou não acessos a determinados dados e estruturas do banco de dados de acordo com a necessidade de cada usuário.
- **Integridade:** a integridade dos dados garante a qualidade dos dados mantidos no banco de dados. Segundo Date (1984, p. 33), "o problema da integridade é o de se garantir que os dados no banco de dados sejam precisos. A inconsistência entre duas entradas representando o mesmo 'fato' é um exemplo de perda de integridade".



**Para saber mais**

Para saber mais a respeito dos fundamentos de banco de dados, o site DevMedia produziu um conteúdo sobre banco de dados que pode ser acessado através do link a seguir: <<http://www.devmedia.com.br/conceitos-fundamentais-de-banco-de-dados/1649>>. Acesso em: 28 ago. 2017.



## Questão para reflexão

Sabendo da importância da informação em nosso dia a dia, indique onde o banco de dados está presente, constantemente, em uma atividade no seu dia.

### Atividades de aprendizagem

- 1.** O ser humano tem três papéis fundamentais dentro de um sistema de banco de dados, sendo cada um deles essencial para que todos os seus processos possam ser desempenhados de forma correta, desde a obtenção dos dados até mesmo a segurança de todo o sistema de banco de dados. Assinale a alternativa que mostra os três papéis do ser humano dentro do sistema de banco de dados.
  - a) Gerente, Desenvolvedor e Tester.
  - b) Usuário, DBA e Desenvolvedor.
  - c) DBA, Desenvolvedor e Analista.
  - d) Analista, DBA e Tester.
  - e) Gerente, Usuário e Desenvolvedor.
  
- 2.** A informação tem um papel importante dentro do meio empresarial, por isso ela acaba sendo valorizada. Para que a informação seja gerada, necessitamos de dados verificados e de procedência que sejam úteis ao negócio. Qual é a diferença entre dado, informação e conhecimento?

# Seção 2

## Estrutura de dados

### Introdução à seção

Vamos entender sobre os níveis de abstração de dados; o que será apresentado em cada camada do banco de dados; como o usuário vê os dados e como eles são armazenados; e os conceitos de estrutura de dados, vendo os principais tipos de dados e suas características, podendo perceber a maneira que a informação é processada.

### 2.1 Estruturas de Dados

Uma estrutura de dados consiste em uma organização de dados em um dispositivo de armazenamento ou memória, de modo que possam ser acessados e utilizados da forma mais eficiente possível. Para Laureano (2012, p. 1),

“na maioria dos casos as estruturas de dados são baseadas nos tipos de armazenamento vistos, ou seja, a transformação da forma de armazenamento tradicional convertida para o mundo computacional. Cada tipo possui suas vantagens e desvantagens e atuam cada uma em sua área.”

Os dados manipulados por algoritmos ou no banco de dados têm natureza distinta, podendo ser número, letra, frase etc. Dependendo da natureza de um dado, algumas operações podem ou não fazer sentido, por exemplo, a somatória entre letras. Laureano (2012, p.1) afirma que, “para que possamos distinguir os dados para que cada um possa realizar operações distintas com eles, as aplicações utilizam o conceito de tipo de dados”.

Mas como se dá o processamento dos dados? Vejamos o esquema a seguir, o qual é uma representação do processamento dos dados/informação.

Figura 1.4 | Esquema de processamento de informação



Fonte: elaborada pelo autor.

## 2.2 Tipos de Dados

Os tipos de dados podem definir um conjunto de valores que podem se assumir, por exemplo, o tipo de dado data pode assumir valores, como "20/01/1978", "07/12/1982", "24/09/1980" etc. Esse tipo de dado está associado ao modo como o computador interpreta os dados. Outra maneira de visualização dos dados é de acordo com o que o usuário desejar fazer, por exemplo, somar dois números, ordenar uma lista de inteiros etc. Isso é conhecido como tipo abstrato de dados.

Segundo Balieiro (2015, p. 16):



Um conjunto de tipos de dados e um de funções são formadores dos tipos abstratos de dados. Para a implementação de cada tipo abstrato são utilizadas as estruturas, essas representam as relações lógicas entre os dados. As relações linear e hierárquicas (conhecidas como não lineares) e as operações efetuadas sobre estes dados, são representadas pelas estruturas de dados.

Os tipos dos dados são definidos na criação das tabelas quando o banco é planejado. Cada coluna criada em uma tabela terá um tipo associado a ela, fazendo com que essa coluna somente aceite dados do tipo que foi estruturado na tabela, a qual foi associada a esta, independentemente da sua arquitetura.

Vamos analisar alguns tipos de dados que podemos encontrar no banco de dados SQL Server da Microsoft, o qual é um dos mais conhecidos do mercado:

- Char e Varchar.
- Nchar e nvarchar.
- Int, bigint, smallint, tinyint e bit.
- Numeric e decimal.
- Date, Datetime e smalldatetime.

Cada um dos tipos acima está associado a um grupo, sendo eles: caractere, números e data/hora.

### 2.2.1 Char, Varchar, nchar e nvarchar

A diferença entre os dois é o fato de que o char possui um tamanho fixo e o varchar tem seu tamanho variável. Veja o exemplo: caso seja

definido um campo do tipo char(10), significa que serão alocados 10 espaços do armazenamento para esse campo, usando ou não seu tamanho total; do mesmo modo, se for definido o campo usando varchar(10), será armazenado somente o tamanho usado pelo conjunto de caracteres, ou seja, se escrever a palavra "amor" dentro desse campo, seu tamanho será de 4 caracteres, e não de 10, como foi declarado.

As variações nchar e nvarchar seguem as mesmas regras, porém seu tamanho máximo é de 4.000 caracteres, sendo os tipos char e varchar com tamanho máximo de 8.000 caracteres. Outra diferença é que os campos char e varchar não suportam o formato Unicode, já os formatos nchar e nvarchar sim. O padrão Unicode permite representar e manipular, de forma consistente, um texto de qualquer sistema de escrita.

### 2.2.2 Int, bigint, smallint, tinyint e bit

São os números inteiros. Nesse tipo de dado não é possível o armazenamento de números com casas decimais. Segue um quadro com as características de cada tipo:

Quadro 1.1 | Tipo de dados do banco de dados

Tipo	Descrição
Bigint	Aceita valores entre $-2^{63}$ e $2^{63}-1$ , sendo que esse datatype ocupa 8 bytes
Int	Os valores aceitos aqui variam entre $-2^{31}$ a $2^{31}-1$ . Ocupa 4 bytes
Smallint	Aceita valores entre -32768 até 32767 e ocupa 2 bytes
Tinyint	Os valores aceitos aqui variam entre 0 e 255, ocupa apenas 1 byte
Bit	É um tipo de dado inteiro (conhecido também como booleano), cujo valor pode corresponder a <b>NULL</b> , 0 ou 1. Podemos converter valores de string <b>TRUE</b> e <b>FALSE</b> em valores de bit, sendo que TRUE corresponde a 1 e FALSE a 0.

Fonte: DevMedia.

### 2.2.3 Decimal e Numeric

Esse tipo de dado armazena números não inteiros e inteiros. O número de casas decimais a serem armazenadas será definido na criação do campo. No quadro a seguir, temos o exemplo de decimal e numeric.

Quadro 1.2 | Tipos de dados não inteiros

Tipo	Descrição
Decimal (P,S)	Os valores aceitos variam entre -999999,99 até 999999,99, sendo que o espaço ocupado varia de acordo com a precisão. Se a precisão for de 1 a 9, o espaço ocupado é de 5 bytes. Se a precisão é de 10 a 19, o espaço ocupado é de 9 bytes; já se a precisão for de 20 a 28, o espaço ocupado é de 13 bytes; e se a precisão for de 29 a 38, o espaço ocupado é de 17 bytes. (P,S) -> Sendo P o número de casas inteiro e S o número de casas decimais.
Numeric (P,S)	Considerado um sinônimo do datatype decimal, o numérico também permite valores entre -10^38-1 e 10^38-1 e o espaço ocupado é o mesmo do anterior. (P,S) -> Sendo P o número de casas inteiro e S o número de casas decimais.

Fonte: DevMedia.

## 2.2.4 Date, datetime e smalldatetime

Esse tipo de dado armazena os dados baseados em data e hora. Vejamos as características de cada um no quadro a seguir:

Quadro 1.3 | Tipos de dados data e hora

Tipo	Descrição
Date	AAAA-MM-DD - Armazena somente dados do tipo data compreendidos entre 0001-01-01 a 9999-12-31, com uma precisão de 1 dia, seu armazenamento é de 3 bytes
Datetime	AAAA-MM-DD hh:mm:ss[.nnn] - Armazena o tipo data e hora juntos, compreendidos entre 1753-01-01 a 9999-12-21, com uma precisão de 0,00333 segundo, seu armazenamento é de 8 bytes
Smalldatetime	AAAA-MM-DD HH:mm:SS – Assim como datetime, o smalldatetime armazena data e hora mas não armazena os milissegundos, sua precisão é de 1 minuto e seu tamanho de armazenamento é de 4 bytes.

Fonte: MSDN Microsoft.

## 2.3 Níveis de Abstração de dados

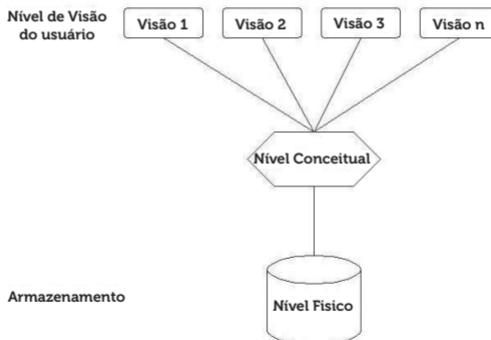
Uma camada de abstração de dados é uma interface de aplicações que unifica a comunicação do hardware com o banco de dados. Cada banco de dados tem uma característica específica, e com isso cada um fornece uma interface própria que deixa o desenvolvedor implementar os códigos para todas as interfaces de banco de dados.

Para garantir uma visão totalmente abstrata do banco de dados para o usuário, pouco importa a ele em qual unidade de armazenamento estão os dados que está consumindo para gerar informação, basta apenas que esta esteja disponível a ele quando solicitar.

Existem três níveis de abstração de dados:

- Nível de visão do usuário.
- Nível Conceitual.
- Nível Físico.

Figura 1.5 | Níveis de abstração de dados



Fonte: DevMedia.

Segundo Date (1984, p. 38):

O nível mais próximo ao armazenamento físico é o nível interno, este, está voltado para como os dados estão realmente armazenados. Já o nível mais próximo ao usuário é o nível externo, nele o usuário tem a visão mais clara sobre os dados. O nível que simula os níveis interno e externo é o nível conceitual.

O nível externo, ou nível de visão de usuário, é a maneira que o usuário enxerga os dados no banco de dados; já o nível conceitual é a maneira que o usuário solicitou o banco de dados, como ele deve funcionar.

A visão interna, ou visão de armazenamento, é como o banco de dados armazena os dados, fato que pouco importa ao usuário final.

Date (1984, p. 38) afirma: “ao exemplificar uma representação abstrata, simplesmente queremos dar o significado que ela está voltada para o usuário, ao invés de ser voltada para a máquina”.

Segundo Silberschatz (2006, p. 4),

“[...] o sistema de banco de dados tem uma finalidade essencial em fornecer uma visão mais abstrata aos usuários, o sistema deve ocultar detalhes de armazenamento que não são necessários aos usuários, somente os dados operacionais”.

Imagino que você tenha percebido a complexidade de um banco de dados, todos os componentes existentes e cada processo que ele mantém durante o seu uso. Essa complexidade é necessária para que todo o sistema de banco de dados possa oferecer segurança e rapidez na manipulação dos dados, podendo, assim, gerar toda a informação necessária ao seu usuário.



### Para saber mais

Para saber mais sobre a estrutura de dados e os níveis de abstração de dados, o professor André Rodrigo Sanches, da USP, produziu um material sobre Arquitetura de banco de dados, disponível no link: <<https://www.ime.usp.br/~andrers/aulas/bd2005-1/aula4.html>>. Acesso em: 28 ago. 2017.



### Questão para reflexão

Sobre a abstração de dados, qual nível é visível ao usuário? O que ele enxerga neste nível?

## Atividades de aprendizagem

1. O isolamento de detalhes internos do banco para o usuário está definido na capacidade de abstração dos dados pelo banco. Cite e explique os três níveis de abstração de dados.
2. O tipo de dado que cada campo pode assumir mostra a qual grupo de dados ele pertence. O grupo de dados que pode assumir valores inteiros e não inteiros são, respectivamente:
3. Em qual nível se concentra o armazenamento no banco de dados? Explique.

## Fique ligado

Nesta unidade, estudamos:

- Dado, informação e conhecimento.
- Valor da informação.
- Fundamentos de banco de dados.

- Emprego de um banco de dados.
- Usuários de um banco de dados.
- Dados operacionais.
- Por que usar banco de dados.
- Estrutura de dados.
- Tipos de dados.
- Níveis de abstração de dados.

## Para concluir o estudo da unidade

Caro aluno, esta unidade fica por aqui! Através dela você pôde observar a importância que um banco de dados tem nos dias atuais. Nossa objetivo central foi mostrar o valor da informação, por isso, o uso de uma aplicação de gerenciamento de dados que proporcione segurança e rapidez na geração da informação. Na sequência, iremos ver como é projetado um banco de dados para a solução de uma necessidade.

## Atividades de aprendizagem da unidade

**1.** A informação é um ativo essencial tanto no âmbito comercial como no pessoal, e seu valor pode variar de acordo com a maneira que é utilizada. Para produzirmos informação, há a necessidade de uma maior organização dos dados e de uma maior segurança no acesso a eles. Assinale a alternativa que define este conceito de segurança e organização.

- a) Firewall de rede.
- b) Active Director.
- c) Arquivos criptografados.
- d) Banco de dados.
- e) Sistemas de gestão.

**2.** No princípio, a saída de dados era demonstrada por luzes que acendiam e apagavam em um determinado painel, no qual cada luz, em sua posição, representava um valor específico, e essa combinação era o resultado do processamento de dados. A evolução caminhava velozmente a passos inimagináveis. Surgiram cartões perfurados, teclados, monitores de vídeo, impressoras etc. Dentro dessa evolução, surgiu o conceito de arquivos de dados. De acordo com esse conceito, assinale a alternativa correta.

- a) Questionário de dados.
- b) Armazenamento de dados.
- c) Segurança da informação.
- d) Mineração de dados.
- e) Requisitos de software.

**3.** A informação que uma empresa possui é um dos seus principais ativos. Ela tem ocupado um destaque nas empresas, por isso muitos dedicam várias horas para elaborar e verificar uma maneira de zelar por este ativo. Para que a empresa use a informação como ativo, deve-se respeitar algumas regras. A regra que demonstra que o tomador da informação tem como objetivo o uso da mesma para a condução do seu negócio é:

- a) Precisa.
- b) Completa.
- c) Econômica.
- d) Relevante.
- e) Flexível.

**4.** Muitos usuários acessam os dados de um banco de dados corporativo ao mesmo tempo, utilizando-os para geração de informação ou até mesmo para a atualização e inserção de dados. Para garantir que cada usuário possa realizar suas operações com total segurança de que elas serão finalizadas, temos o conceito de:

- a) Transações distribuídas.
- b) Múltiplas visões de dados.
- c) Abstração de dados.
- d) Natureza autodescritiva.
- e) Estrutura de armazenamento.

**5.** Abstração de dados é uma interface de aplicações que unifica a comunicação do hardware com o banco de dados. Cada banco de dados tem uma característica específica e, com isso, cada um fornece uma interface própria que deixa o desenvolvedor implementar os códigos para todas as interfaces de banco de dados. Existem três níveis de abstração de dados. Assinale a alternativa que mostra em qual nível os dados são armazenados no banco de dados:

- a) Nível de encapsulamento.
- b) Nível de usuário.
- c) Nível conceitual.
- d) Nível de armazenamento.
- e) Nível de rede.

# Referências

- ANGELOTTI, Elaini Simoni. **Banco de Dados**. Curitiba: Editora do Livro Técnico, 2010.
- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR ISSO/IEC 27002**. ABNT: Rio de Janeiro, 2005.
- BALIEIRO, Ricardo. **Estrutura de dados**. Rio de Janeiro: SESES, 2015.
- Conceitos Fundamentais de Banco de Dados. DEV MEDIA. Disponível em: <<http://www.devmedia.com.br/conceitos-fundamentais-de-banco-de-dados/1649>>. Acesso em: 19 jul. 2017.
- DANTAS, Marcus Leal. **Segurança da Informação**: uma abordagem focada em gestão de risco. Olinda, 2011.
- DATE, C. J. **Introdução a Sistemas de Banco de Dados**. Rio de Janeiro: Campus, 1984.
- DAVENPORT, Thomas; PRUSAK, Laurence. **Conhecimento Empresarial**: como as organizações gerenciam o seu Capital Intelectual. Rio de Janeiro: Campus, 1998.
- ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistema de banco de dados**. São Paulo: Editora Pearson, 2005.
- HEUSER, C. A. **Projeto de banco de Dados**. Porto Alegre: Editora Sagra Luzzatto, 1998.
- LAUDON, Kenneth C.; LAUDON, Jane P. **Sistemas de informação**. 4. ed. Rio de Janeiro: LTC, 1999.
- LAUREANO, Marcos. **Estrutura de dados com Algoritmos e C**. Rio de Janeiro: BRASPORT, 2012.
- MÜLBERT, A. L.; AYRES, N. M. **Sistema de Informações Gerenciais no Varejo e Serviços**. 2. ed. Palhoça: Unisul Virtual, 2007.
- SILBERSCHATZ, Abraham; KORTH, Henry F. **Sistema de Banco de Dados**. Rio de Janeiro: Elsevier, 2006;
- SQL SERVER – Comandos Básicos, Objetos, Tipos de Dados e Criação de Database. DEV MEDIA. Disponível em: <<http://www.devmedia.com.br/sql-server-comandos-basicos-objetos-tipos-de-dados-e-criacao-de-database/17052>>. Acesso em: 19 jul. 2017.
- STAIR, M. R. **Princípios de sistemas de informação**. Rio de Janeiro: LTC, 1998.
- Tipos de Dados. **MSDN Microsoft**. Disponível em: <[https://msdn.microsoft.com/pt-br/library/bb630352\(v=sql.120\).aspx](https://msdn.microsoft.com/pt-br/library/bb630352(v=sql.120).aspx)>. Acesso em: 19 jul. 2017.



# Sistemas de gerenciamento de banco de dados (SGBD)

Roberto Yukio Nishimura

### Objetivos de aprendizagem

- Estudar os conceitos, os elementos e as funcionalidades de um SGBD.
- Estudar as ferramentas de um SGBD.
- Estudar as estruturas de um SGBD.

### Seção 1 | Sistema de Gerenciamento de Banco de Dados (SGBD) ou Sistema Gerenciador de Banco de Dados

Nesta seção, apresentaremos a definição do que é um Sistema de Gerenciamento de Banco de Dados (SGBD) ou Sistema Gerenciador de Banco de Dados, e como funcionam seus principais componentes funcionais. Também serão mostrados os detalhes que compõem a sua estrutura. Complementarmente, deixaremos alguns links disponíveis para o aluno, a fim de ampliar o seu conhecimento no assunto.

### Seção 2 | Ferramentas de administração do banco de dados

Nesta seção, apresentaremos as ferramentas de administração do banco de dados, as quais realizam as principais tarefas para a manutenção funcional e operacional do banco de dados. Também serão descritas as ferramentas de apoio complementar, como realizador de backup do banco de dados e realizador da recuperação do banco de dados em casos de *disaster-recovery*.

### Seção 3 | Instância, esquema e independência de dados

Nesta seção, explanaremos o que é uma instância de banco de dados, um esquema de banco de dados, a independência lógica de dados e a independência física de dados. Propriedades ACID.



# Introdução à unidade

Caro(a) aluno(a), nesta unidade apresentaremos os conceitos relacionados ao Sistema de Gerenciamento de Banco de Dados ou Sistema Gerenciador de Banco de Dados (SGBD), seu funcionamento interno e como cada componente interage com a estrutura do banco de dados, do sistema operacional e do administrador de banco de dados. Em seguida, será a vez das ferramentas que compõem o ambiente de um sistema gerenciador de banco de dados autêntico, com suas tarefas e funções específicas. Depois, a contextualização e explicação sobre instância de banco de dados, esquema de banco de dados, passando pela independência lógica de dados e pela independência física de dados, finalizando com as propriedades ACID de um banco de dados.

Na Seção 1, na qual abordaremos o SGBD, teremos algumas definições que apresentarão os componentes de um banco de dados em funcionamento, como é o seu comportamento dentro da memória de um servidor e a sua interação com o sistema operacional. Existem cenários nos quais o SGBD interage e depende totalmente do sistema operacional, e cenários em que o SGBD é quem controla os periféricos de armazenamento, como as unidades de disco rígido (HD – hard disk).

Já na Seção 2, o assunto a ser abordado são as ferramentas de administração do banco de dados, pois através delas é que são realizadas as principais tarefas funcionais e operacionais, por exemplo, a criação do banco de dados, a criação das tabelas e dos relacionamentos entre as tabelas, a criação dos usuários do banco de dados e a criação dos grupos de acesso e seus perfis de funcionamento. São citadas as ferramentas de operacionais para backup (cópia de segurança) e recovery (recuperação) do banco de dados. Lembramos que o backup é uma rotina de proteção e segurança dos dados, mas fundamental para a recuperação do banco de dados em casos de disaster-recovery (quando ocorre a perda do banco de dados e é necessária a reconstrução dele).

A última seção desta unidade – Seção 3 – trata de diversos itens complementares para uma boa conceituação de SGBD, como instância, esquema, independência de dados (lógica e física) e propriedades ACID.

# Seção 1

## Sistema de Gerenciamento de Banco de Dados (SGBD), ou Sistema Gerenciador de Banco de Dados

### Introdução à seção

Caro(a) aluno(a), nesta Seção 1 vamos abordar os conceitos relacionados ao Sistema de Gerenciamento de Banco de Dados, ou Sistema Gerenciador de Banco de Dados (SGBD). Serão tratados dois conceitos básicos (instância e esquema), os quais, para alguns SGBDs de mercado, recebem outros nomes, inclusive, os termos “instância” e “esquema” também são utilizados com outros propósitos.

#### 1.1 Definição

Segundo Navathe e Elmasri (2005, p. 19),



a arquitetura dos pacotes SGBD (Sistemas Gerenciadores de Banco de Dados) evoluiu dos sistemas pioneiros e monolíticos, em que o pacote de software SGBD era um bloco único formando um sistema fortemente integrado, para os modernos pacotes SGBDs modulares por projeto, com uma arquitetura cliente/servidor.

O Sistema Gerenciador de Banco de Dados (SGBD) também é conhecido no mercado pela sigla DBMS, que vem do inglês DataBase Management System, e a palavra *management* já diz tudo: vamos “gerenciar” um banco de dados utilizando um sistema.

Desta forma, o SGBD é como se fosse uma caixa de ferramentas, pois é uma coleção de programas de computador com um propósito específico, o qual é muito mais do que controlar ou gerenciar arquivos convencionais, tarefa esta que é destinada aos Sistemas Operacionais dos computadores.

Um Sistema Operacional no quesito de gerenciamento do armazenamento em disco divide tudo em dois tipos de itens: ou é um arquivo, ou é um diretório (pasta). Desta forma, esse gerenciamento é executado de uma forma mais tranquila, com controles mais simples,

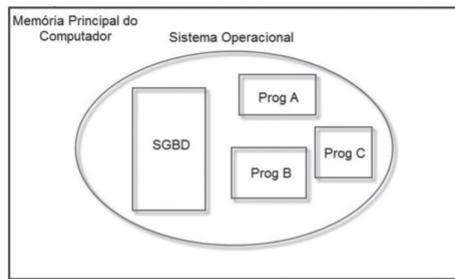
como direito de gravação, direito de leitura, direito de regravação ou direito de remoção. Sistemas Operacionais baseados em UNIX implementam o conceito de direito de usuário, direito de grupo e direito de outros. Assim, aumentam um pouco mais a segurança no acesso aos arquivos convencionais.

O LINUX é um Sistema Operacional de código fonte aberto, baseado nos conceitos do Sistema Operacional UNIX, o qual, por sua vez, é fechado, e cada fabricante guarda o seu kernel (código fonte) bem fechado, mas com interoperabilidade entre Sistemas Operacionais semelhantes (ou seja, os comandos são semelhantes entre si e o comportamento dos comandos também é muito semelhante entre si, mas não são totalmente iguais).

E é exatamente nesta hora que o SGBD entra em ação. Pelo fato de ser um conjunto de programas ou ferramentas com um propósito específico, que é o gerenciamento de banco de dados, possui muito mais funcionalidades e operacionalidades para garantir total segurança aos dados.

A Figura 2.1 apresenta a Memória Principal do Computador, que é controlada pelo Sistema Operacional. A elipse é para destacar que, tanto o SGBD quanto os vários programas (Prog A, Prog B e Proc C) que fazem uso do banco de dados, estão todos carregados, executados e gerenciados na Memória Principal do Computador, em locais distintos da memória.

Figura 2.1 | Memória Principal do Computador



Fonte: elaborada pelo autor.

O SGBD é um software ou conjunto de ferramentas que manipula e controla todos os acessos ao banco de dados, proporcionando a interface entre os programas aplicativos dos usuários, ou mesmo os acessos interativos do usuário e o banco de dados propriamente dito.

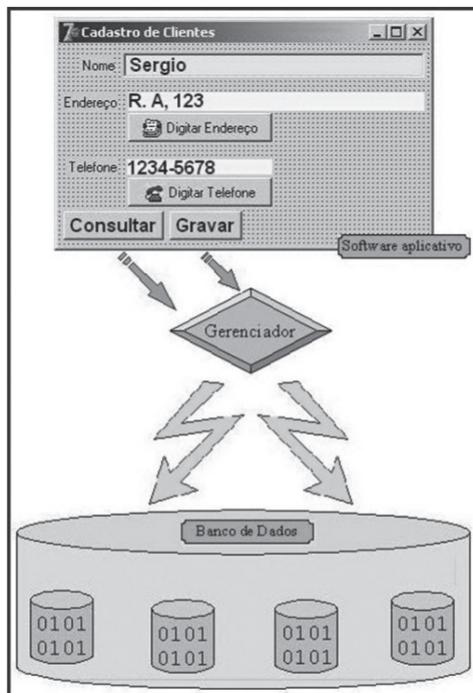
Nesta interação, podemos considerar desde a fase de criação do

banco de dados, criação das tabelas e seus relacionamentos, criação dos usuários, criação dos grupos de usuários, criação das regras de acesso às tabelas e criação das regras de negócio, até a execução de rotinas de backup para a proteção dos dados e rotinas de *recover* em casos de *disaster-recovery* do banco de dados.

Não podemos esquecer dos programas dos usuários acessando e interagindo diretamente com o banco de dados, sempre através do “gerenciador”.

A Figura 2.2, a seguir, representa como o “gerenciador”, ou SGBD, interage entre o software aplicativo e o banco de dados. O software aplicativo é, para o usuário, a tela do computador, na qual ele vai preencher os campos com os dados que ele pretende inserir no sistema. O “gerenciador”, ou SGBD, vai receber esses dados do software aplicativo e fará a gravação ou leitura de dados dentro do banco de dados. Este, por sua vez, está armazenado dentro dos discos rígidos do computador.

Figura 2.2 | Sistema de Banco de Dados



Fonte: elaborada pelo autor.



Para saber mais

Vale uma pesquisa na enciclopédia livre Wikipédia.

Disponível em: <[https://pt.wikipedia.org/wiki/Sistema\\_de\\_gerenciamento\\_de\\_banco\\_de\\_dados](https://pt.wikipedia.org/wiki/Sistema_de_gerenciamento_de_banco_de_dados)>. Acesso em: 2 ago. 2017.

O site Devmedia também é uma referência no mundo da Tecnologia da Informação.

Disponível em: <<http://www.devmedia.com.br/gerenciamento-de-banco-de-dados-analise-comparativa-de-sgbds/30788>>. Acesso em: 2 ago. 2017.

Você também pode acessar os links a seguir:

- Tutorial – Sistema Gerenciador de Banco de Dados – Prof. Eduardo Henrique Gomes. Disponível em: <<http://ehgomes.com.br/disciplinas/bdd/sqbd.php>>. Acesso em: 2 ago. 2017.
- Tutorial – Dicas de programação. Disponível em: <<http://www.dicasdeprogramacao.com.br/o-que-e-um-sqbd/>>. Acesso em: 2 ago. 2017.



Questão para reflexão

O mundo dos SGBDs começou com as grandes marcas (IBM e Burroughs) e, depois, chegaram as marcas independentes (Oracle, Sybase e Microsoft). Tudo isso era pago e muito caro, portanto era acessível somente para as grandes empresas, mas a comunidade mundial do software livre começou a lançar produtos gratuitos, como MySql, Paradox, Ingress e Postgree, os quais permitiram às pequenas empresas utilizarem softwares com as mesmas características dos SGBDs pagos. Hoje em dia, as grandes marcas possuem versões gratuitas, porém limitadas para uso, e quando o usuário chega nesse limite, tem que comprar a versão paga.

O que o você acha dessa prática de mercado das grandes marcas?

## Atividades de aprendizagem

1. Considerando que, quando surgiu, o Sistema Gerenciador de Banco de Dados (SGBD) era monolítico, ou seja, praticamente um único programa, e que, ao longo do tempo, vem evoluindo. Desta forma, hoje em dia, o SGBD é composto por diversos programas de computador que permitem

o gerenciamento do banco de dados. Quem é o responsável pela interação entre transferir os dados da memória do computador e os discos de armazenamento de dados?

- a) Processador.
- b) Memória RAM.
- c) Sistema Operacional.
- d) EPROM.
- e) Barramento de Dados.

**2.** Quando é realizada uma interação entre um programa aplicativo e o banco de dados, o sistema operacional não pode interferir nessa etapa. Neste momento, o Sistema Gerenciador de Banco de Dados (SGBD) entra em ação, pois todo o gerenciamento e a administração do banco de dados é de sua responsabilidade. Em qual local de um servidor o SGBD executa todo esse gerenciamento?

- a) Processador.
- b) Memória Principal.
- c) Disco Rígido.
- d) Disco Estado Sólido.
- e) Processador Binário.

# Seção 2

## Ferramentas de administração do banco de dados

### Introdução à seção

Caro(a) aluno(a), na Seção 2, o assunto tratado será as ferramentas de administração do banco de dados. Vamos abordar a linguagem de acesso utilizada nas ferramentas de administração e, em seguida, detalharemos algumas das principais funcionalidades, como backup e *disaster-recovery*.

### 2.1 Definição

Conforme Navathe e Elmasri (2005, p. 27), “além dos módulos de software descritos, muitos SGBDs têm utilitários de banco de dados que auxiliam o DBA no gerenciamento do sistema de banco de dados”.

Um Sistema Gerenciador de Banco de Dados (SGBD) é composto por diversas ferramentas ou programas de computador para executarem as tarefas da administração do banco de dados.

Como já foi dito anteriormente, as tarefas administrativas podem ser de manutenção funcional do banco de dados ou de manutenção operacional do banco de dados.

As tarefas de manutenção funcional podem ser entendidas como as tarefas para criação do banco de dados, criação das tabelas e relacionamentos, criação dos usuários, criação dos grupos de usuários, criação das regras de negócio, criação de regras de acesso etc.

Já as tarefas de manutenção operacional são aquelas nas quais executamos ações, como carregar o banco de dados, colocar o banco de dados disponível no ar, suspender a execução do banco de dados, retirar o banco de dados do ar, executar a realização de backup do banco de dados (tarefa esta que pode ser feita com o banco de dados no ar ou com o banco de dados fora do ar) e executar rotinas de recuperação do banco de dados (que podem ser do banco de dados inteiro ou de apenas parte do banco de dados, inclusive controlando o tempo em que se deseja encerrar o processo de recuperação).

Rotinas de diagnóstico da saúde dos dados também podem ser executadas, rotinas de carga de dados ou rotinas de extração de dados.

## 2.2 Linguagem de acesso

Para acessar o banco de dados, o SGBD utiliza uma linguagem própria. É através dessa linguagem própria, a qual apresenta características semânticas (codifica-se do jeito que se fala, ou seja, ao realizar a leitura do comando, é perfeitamente possível a compreensão do que se deseja fazer), que os SGBDs ganharam uma parte da confiança entre os analistas e programadores.

Esse padrão semântico dos comandos fez com que a linguagem de acesso ao banco de dados recebesse o nome de Linguagem de Consulta Estruturada, ou *Structure Query Language*, ou simplesmente SQL.

A linguagem própria é dividida em três categorias, com propósitos específicos. Para a definição das estruturas de armazenamento e organização e manutenção, denominamos de Linguagem de Definição de Dados (DDL), do inglês *Data Definition Language*; para a manipulação dos dados que estão armazenados nas tabelas do banco de dados, Linguagem de Manipulação de Dados (DML), do inglês *Data Manipulation Language*; e para o controle e a segurança do banco de dados, Linguagem de Controle de Dados (DCL), do inglês *Data Control Language*.

### 2.2.1 Linguagem de Definição de Dados (DDL)

Segundo Silberschatz, Korth e Sudarshan (2004), DDL é uma linguagem que define esquemas de relações.

Na Linguagem de Definição de Dados (DDL), temos um grupo de comandos padrão SQL (*Structure Query Language*) que permite a criação e manutenção das estruturas de armazenamento, organização e acesso aos dados.

Os comandos básicos são: CREATE, ALTER e DROP. Através desses comandos são possíveis a criação, a alteração e a remoção de tabelas, campos, chaves primárias, chaves estrangeiras, regras de negócio, índices, pacotes, funções, procedimentos, gatilhos (*triggers*) e visões (*view*).

### 2.2.2 Linguagem de Manipulação de Dados (DML)

Linguagem de Manipulação de Dados (DML) é uma linguagem de consulta baseada na álgebra relacional e no cálculo relacional de tuplas

(SILBERSCHATZ; KORTH; SUDARSHAN, 2004).

Na DML, temos um grupo de comandos padrão SQL que fazem a manipulação dos dados propriamente dita. São os comandos utilizados no dia a dia pelos analistas, programadores, administradores de banco de dados e até mesmo por usuários finais que possuem um conhecimento estruturado.

Os comandos básicos são: SELECT, INSERT, UPDATE e DELETE. Esses comandos podem ser conhecidos como CERA (criar, eliminar, recuperar, atualizar) ou DICA (deletar, inserir, consultar e atualizar).

### **2.2.3 Linguagem de Controle de Dados (DCL)**

A Linguagem de Controle de Dados (DCL) é uma terceira categoria e mais atual, na qual temos alguns comandos que auxiliam o gerenciamento e o controle do banco de dados e não interferem propriamente nas estruturas de armazenamento ou nos dados. São comandos que ficam restritos ao DBA (administrador de banco de dados).

Os comandos básicos são: CREATE, ALTER e DROP, mas concatenados com USER, ROLE e VIEW. E complementando: STARTUP, OPEN DATABASE, SHUTDOWN, STOP e CLOSE.

## **2.3 Ferramentas para administração do banco de dados**

As ferramentas para administração do banco de dados, geralmente, estão restritas ao DBA (administrador de banco de dados), pois através delas é possível colocar o banco de dados no ar (disponível aos usuários), retirar o banco de dados do ar, executar rotinas de backup, executar rotinas de verificação de performance, extrair relatórios de estatísticas de acesso aos dados, executar procedimentos para compactação de espaço e executar procedimentos para a recuperação do banco de dados em casos de *disaster-recovery*.

### **2.3.1 Como são as ferramentas para administração do banco de dados**

As ferramentas para administração do banco de dados eram todas client/server, ou seja, programas instalados no microcomputador ou notebook do DBA (administrador do banco de dados), ou direto no

servidor do banco de dados, e assim a performance dependia do equipamento em que estavam instaladas as ferramentas.

A partir da propagação da arquitetura de três camadas, essas ferramentas passaram a ser em ambiente *WEB*, ou seja, funcionam através dos *browsers* de navegação internet e, com isso, podem ser acessadas de qualquer lugar e com qualquer dispositivo (inclusive tablets e smartphones). Uma nova geração de ferramentas vem surgindo recentemente, agora na forma de Apps para smartphones, prometendo mobilidade e acesso total.

Todo SGBD possui ferramentas específicas para as rotinas de backup, que são cópias de segurança dos dados. Normalmente, as cópias de segurança dos dados ficam armazenadas em fitas, ou caso essas cópias sejam muito utilizadas, podem ficar armazenadas em disco rígido também. Atualmente, algumas empresas estão utilizando cópias de segurança em nuvem (computação em nuvem).

As ferramentas para recuperação do banco de dados em casos de *disaster-recovery* são invocadas a partir de comandos de administração, nos quais, normalmente, o banco de dados está fora do ar (indisponível) e em estado de recuperação.

Independentemente do fabricante do SGBD, existem vários fornecedores de ferramentas para backup também, os quais, muitas vezes, não se prendem a um único banco de dados, sendo, assim, uma ferramenta multiplataforma.



Para saber mais

**IBM:** ferramentas para gerenciamento de Dados IBM InfoSphere, especificamente para o SGBD DB2 nos sistemas operacionais UNIX, LINUX e WINDOWS.

Disponível em: <[https://www.ibm.com/support/knowledgecenter/pt-br/SSEPGG\\_10.5.0/com.ibm.db2.luw.idm.tools.doc/doc/c0055013.html](https://www.ibm.com/support/knowledgecenter/pt-br/SSEPGG_10.5.0/com.ibm.db2.luw.idm.tools.doc/doc/c0055013.html)>. Acesso em: 2 ago. 2017.

**QUEST:** uma ferramenta de gerenciamento de banco de dados independente de fabricante de SGBD e que funciona para diversos SGBDs de mercado, sendo que, no momento da aquisição oficial, o cliente diz para qual SGBD ele quer comprar. Disponível em: <<https://www.quest.com/br-pt/database-management/>>. Acesso em: 2 ago. 2017.

Você também pode acessar os links a seguir:

- GUJ Caelum. Disponível em: <<http://www.guj.com.br/t/ferramentas-para-gerenciar-banco-de-dados-windows-linux/2564>>. Acesso em: 2 ago. 2017.
- MySQL - Administração e ferramentas de desenvolvimento. Disponível em: <<http://www.pinceladasdaweb.com.br/blog/2010/01/11/mysql-administracao-e-ferramentas-de-desenvolvimento/>>. Acesso em: 2 ago. 2017.
- SQL Manager. Disponível em: <<http://www.sqlmanager.com.br/>>. Acesso em: 2 ago. 2017.



### Questão para reflexão

Um DBA (administrador de banco de dados) é quem faz uso das ferramentas de administração e gerenciamento de banco de dados. Porém, com o avanço da Inteligência Artificial (IA), é possível prever, programar e automatizar uma série de tarefas rotineiras da administração do banco de dados. Será que o futuro do administrador de banco de dados está no fim?

## Atividades de aprendizagem

1. Um Sistema Gerenciador de Banco de Dados (SGBD) é uma caixa de ferramentas com diversos softwares para a administração e manutenção de um banco de dados. Uma das tarefas é a realização de rotinas de backup, visando à proteção dos dados. Quando ocorre um sinistro com o banco de dados, são necessárias a reconstrução e a recuperação do banco de dados. Qual é o nome dado para este cenário?
  - a) *Disaster-Reconstruct*.
  - b) *Disable-Enable*.
  - c) *Disaster-Recovery*.
  - d) *Crunch-Recovery*.
  - e) *Disaster-StartUp*.
2. Considerando que o acesso ao banco de dados é feito utilizando uma linguagem de alto nível, padronizada e dentro de uma semântica. O padrão SQL (*Structure Query Language*) veio para consolidar esta forma de linguagem. Nas definições e classificações, qual é o nome da categoria que engloba os comandos de criação e manutenção das estruturas de armazenamento, como tabelas e índices?

- a) DDL – Linguagem de Definição de Dados.
- b) DML – Linguagem de Manipulação de Dados.
- c) DCL – Linguagem de Controle de Dados.
- d) DTL – Linguagem de Transação de Dados.
- e) DPL – Linguagem de Programação de Dados.

# Seção 3

## Instância, esquema e independência de dados

### Introdução à seção

Caro(a) aluno(a), na Seção 3, vamos abordar a definição e conceituação de instância e esquema, principalmente porque os SGBDs de mercado utilizam estes dois termos com propósitos diferentes. A independência de dados também será tratada, com suas variações lógica e física. Essa característica do banco de dados é uma grande facilitadora para o dia a dia dos analistas e programadores, por minimizar a quantidade de vezes que um programa de computador precisa ser compilado ou não. Finalizamos com as propriedades ACID (atomicidade, consistência, isolamento e durabilidade) de um banco de dados. Graças a essas propriedades que o SGBD é muito superior aos gerenciadores de arquivos convencionais.

### 3.1 Instância de Banco de Dados

#### 3.1.1 Definição

Conforme Navathe e Elmasri (2005, p. 21),

os dados no banco de dados, em um determinado momento, são chamados de estado do banco de dados ou instantâneos (snapshot). Também chamado o conjunto corrente de ocorrências ou instâncias: no banco de dados. Em um dado estado do banco de dados, cada construtor do esquema tem seu próprio conjunto corrente de instâncias.

Quando um Sistema Gerenciador de Banco de Dados (SGBD) é colocado em execução, vários programas de computador são carregados na memória do servidor de banco de dados. Feito isso, dizemos que o SGBD está apto a colocar um banco de dados no “ar”, ou seja, disponibilizá-lo para uso.

Através de ferramentas de administração de banco de dados é executado um comando de “startup” ou “open”, e o banco de dados invocado é carregado na memória do servidor de banco de dados.

Após o banco de dados estar carregado e liberado para uso, dizemos que a instância está disponível para uso, ou seja, os computadores ou

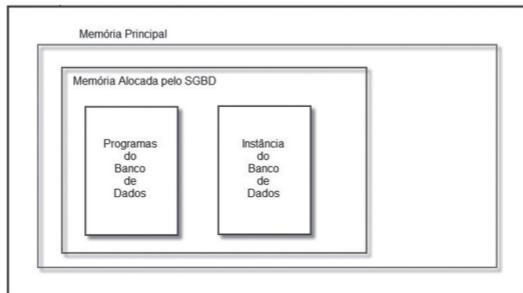
os usuários já podem fazer uso dos dados.

Ao conjunto dos programas de banco de dados, mais os dados carregados do banco de dados específico, chamamos de instância do banco de dados.

É como se fosse uma fotografia momentânea do banco de dados que foi tirada naquele instante. No momento seguinte, independentemente de os dados na memória terem sido alterados ou não, chamamos de uma nova instância do banco de dados.

A Figura 2.3 representa como a memória alocada pelo SGBD está dentro da memória principal, e que a memória utilizada pelos programas do banco de dados e a instância do banco de dados fazem parte exatamente desta memória alocada pelo SGBD.

Figura 2.3 | Instância de banco de dados



Fonte: elaborada pelo autor.



#### Para saber mais

Bancos chamam de instância um “banco de dados” carregado na memória. Pesquise na internet quais são os termos utilizados por estes SGBDs de mercado.

Para ter mais conhecimento a respeito do assunto estudado até aqui, você pode acessar os links a seguir:

- Instância do mecanismo de banco de dados. Disponível em: <[https://msdn.microsoft.com/pt-br/library/hh231298\(v=sql.120\).aspx](https://msdn.microsoft.com/pt-br/library/hh231298(v=sql.120).aspx)>. Acesso em: 30 jul. 2017.
- O que é banco de dados, instância, schema e datafile. Disponível em: <<http://aprenderoracle.com/2011/03/21/13bancodedados/>>. Acesso em: 30 jul. 2017.
- Banco de dados: entenda o tema. Disponível em: <<http://dicasbd.com.br/banco-de-dados-entenda-o-tema/>>.

[blogspot.com.br/2011/02/entender-o-termo-instancia-e-banco-de.html](http://blogspot.com.br/2011/02/entender-o-termo-instancia-e-banco-de.html). Acesso em: 30 jul. 2017.

- Arquitetura do SQL SERVER. Disponível em: <<http://www.devmedia.com.br/arquitetura-do-sql-server/24271>>. Acesso em: 30 jul. 2017.



### Questão para reflexão

Como um SGBD convive com o Sistema Operacional do computador?

## 3.2 Esquema de Banco de Dados

### 3.2.1 Definição

No esquema de banco de dados, “a descrição do banco de dados é intitulada esquema de banco de dados. Definido durante o projeto do banco de dados e não se espera que seja alterado frequentemente” (NAVATHE; ELMASRI, 2005, p. 21).

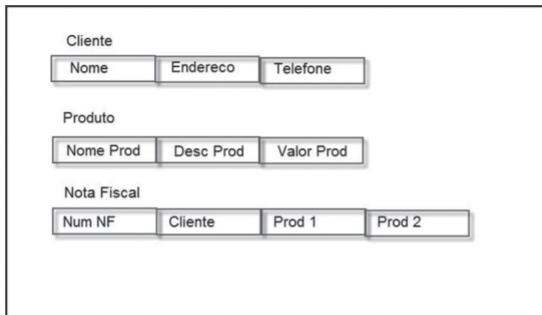
Durante um projeto de banco de dados, a modelagem de dados é fundamental para que a estrutura do banco de dados projetado tenha a melhor organização possível, sem perder os relacionamentos entre as entidades identificadas e armazenadas.

A técnica de modelagem entidade relacionamento (MER) apresenta o projeto de banco de dados com três etapas distintas, nas quais é possível identificar os resultados dessa modelagem. São os projetos conceitual, projeto lógico e projeto físico do banco de dados.

No projeto conceitual, são representadas as entidades, os atributos das entidades e os relacionamentos. Neste nível da modelagem, é apresentado um diagrama de forma a demonstrar a realidade mais próxima entre o mundo real e o modo como serão armazenados os dados. A representação das entidades e dos atributos é feita de forma livre, sem a necessidade de determinar o tipo do dado ou atributo, ou de estipular o tamanho do dado.

A Figura 2.4 apresenta como um projeto conceitual mapeia os dados que serão modelados, já separando as entidades, os relacionamentos e seus atributos. As entidades Cliente e Produto e o relacionamento Nota Fiscal são identificados, e os atributos de cada entidade e relacionamento também são apresentados. Não é feita nenhuma referência ao tipo do atributo ou ao tamanho do atributo.

Figura 2.4 | Projeto Conceitual

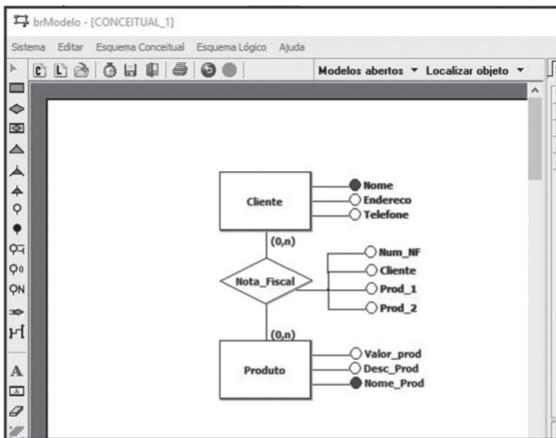


Fonte: elaborada pelo autor.

A Figura 2.5 demonstra como o projeto conceitual pode ser modelado dentro da ferramenta BrModelo, gerando, assim, um Diagrama Entidade Relacionamento (DER). As entidades Cliente e Produto são representadas com um retângulo, e o relacionamento Nota Fiscal, com um losango. Os atributos são representados por um círculo.

Esta proposição de representação gráfica foi feita por Chen (1990), que é considerado o “pai” da técnica de modelagem de dados Entidade Relacionamento, ou MER.

Figura 2.5 | Projeto conceitual na ferramenta BrModelo



Fonte: elaborada pelo autor.

No projeto lógico são representadas as tabelas, os campos e os relacionamentos. Agora as entidades já são referenciadas como tabelas, os atributos viram os campos e os relacionamentos continuam como

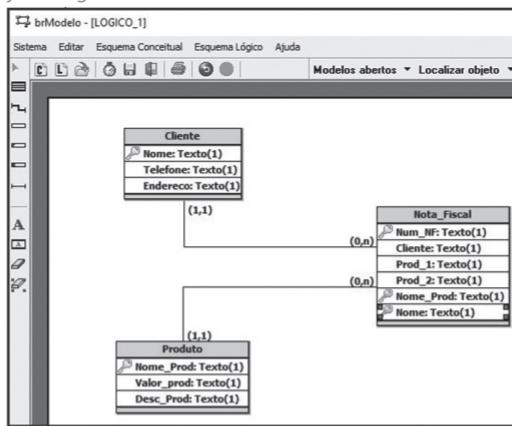
relacionamentos, porém já apontam quem são as chaves primárias e as chaves estrangeiras relacionadas.

Outro ponto importante é a possibilidade de tipificar os campos e atribuir tamanhos específicos, por exemplo: número (6) – campo numérico de seis posições inteiras, caractere (10) – campo caractere de 10 posições no máximo, alfanumérico (10) – campo alfanumérico de 10 posições no máximo, date – campo do tipo data, aceitando a máscara que for escolhida (dd/mm/aa, dd/mm/aaaa, dd/mmm/aa, aa/mm/dd).

O projeto lógico é agnóstico de banco de dados, ou seja, não representa nenhum SGBD de mercado, portanto é totalmente independente.

A Figura 2.6 apresenta o projeto lógico já com as tabelas e os campos. Como o relacionamento entre as entidades Cliente e Produto possui uma cardinalidade N-N, a própria ferramenta BrModelo já resolve esse problema transformando o relacionamento em uma nova entidade/tabela.

Figura 2.6 | Projeto Lógico



Fonte: elaborada pelo autor.

No projeto físico, são representadas as tabelas, os campos e os relacionamentos. Agora, sob o ponto de vista dos SGBDs, as entidades já são referenciadas como tabelas, os atributos viram os campos e os relacionamentos continuam como relacionamentos, porém já apontam quem são as chaves primárias e as chaves estrangeiras relacionadas.

Neste projeto, a sintaxe própria do SGBD é utilizada com a nomenclatura correta para uso nos comandos SQL, que será aplicada

para a geração do banco de dados.

Por exemplo, um atributo NOME do tipo genérico caractere de 10 posições pode ser representado assim:

NOME STRING(10)

Ou

NOME CHAR(10)

Ou

NOME VARCHAR(10)

Essas sintaxes podem ser aplicadas nos SGBDs de mercado, como ORACLE, MS SQL SERVER ou IBM DB2.

A Figura 2.7 apresenta os códigos padrão SQL ANSI gerados pela ferramenta BrModelo através do processo de engenharia progressiva, o qual transformou o Projeto Lógico em um Projeto Físico. Os comandos que estão no Projeto Físico estão aptos a serem executados em um SGBD que adote o padrão ANSI.

Figura 2.7 | Projeto Físico

– Sql ANSI 2003 - brModelo.

```
CREATE TABLE Cliente (
    Nome String(1) PRIMARY KEY,
    Telefone String(1),
    Endereco String(1)
)

CREATE TABLE Produto (
    Nome_Prod String(1) PRIMARY KEY,
    Valor_prod String(1),
    Desc_Prod String(1)
)

CREATE TABLE Nota_Fiscal (
    Num_NF String(1) PRIMARY KEY,
    Cliente String(1),
    Prod_1 String(1),
    Prod_2 String(1),
    Nome_Prod String(1),
    Nome String(1),
    FOREIGN KEY(Nome_Prod) REFERENCES Produto (Nome_Prod)
)
```

Fonte: elaborada pelo autor.



Para saber mais

O SGBD de mercado da Microsoft, MS-SQL Server, utiliza em seus padrões de nomenclatura a palavra "SCHEMA", ou esquema, para se referenciar a uma instância de banco de dados. Em suas ferramentas de administração, várias instâncias ou esquemas podem ser administrados

de um único ponto.

Já o SGBD de mercado ORACLE utiliza a palavra “SCHEMA” para se referenciar a “OWNERS” ou donos de tabela. Uma instância de banco de dados ORACLE pode conter diversos usuários “OWNERS”, e as tabelas estarem vinculadas a estes usuários. Desta forma, é possível dividir ou organizar diversos módulos de sistemas diferentes em “OWNERS” ou “SCHEMA” diferentes, aumentando a segurança do banco de dados.

Você também pode acessar os links a seguir para obter mais conhecimento:

- Esquema em Oracle. Disponível em: <<http://aprenderoracle.com/2011/03/21/13bancodedados/>>. Acesso em: 2 ago. 2017
- Revista SQL Magazine. Disponível em: <<http://www.devmedia.com.br/artigo-sql-magazine-57-esquemas-administrando-objetos-do-banco-de-dados/10859>>. Acesso em: 2 ago. 2017.
- Lucid Chat. Disponível em: <<https://www.lucidchart.com/pages/pt/o-que-%C3%A9-um-esquema-de-banco-de-dados>>. Acesso em: 2 ago. 2017.



### Questão para reflexão

Consulte a enciclopédia livre Wikipédia e veja a definição de esquema.

Disponível em: <[https://pt.wikipedia.org/wiki/Esquema\\_de\\_banco\\_de\\_dados](https://pt.wikipedia.org/wiki/Esquema_de_banco_de_dados)>. Acesso em: 2 ago. 2017.

### 3.3 Independência de Dados

Uma característica que ajuda a marcar a diferença entre arquivos convencionais (flat files) e os bancos de dados é a interface entre os programas compilados que irão acessar os dados e os dados propriamente ditos.

Um programa de computador, quando é escrito ou codificado, precisa passar por um processo de “compilação”, que consiste, basicamente, em transformar um programa de computador escrito em uma linguagem de programação de alto nível ou código fonte (que o ser humano consegue entender) e aplicar o compilador dessa linguagem específica, a qual vai gerar um código objeto ou programa de computador executável pelo sistema operacional do computador.

Neste processo de compilação, toda referência a arquivos convencionais ou ao banco de dados é verificada e checada para saber se todas as declarações de localização dos dados, bem como o formato dos dados que serão acessados, estão correspondendo aos arquivos convencionais ou ao banco de dados.

Uma vez conferidas essas referências, o programa objeto ou executável já tem essas informações garantidas e que não poderão ser alteradas depois, sendo que a única forma de alterar é através de uma nova compilação do código fonte.

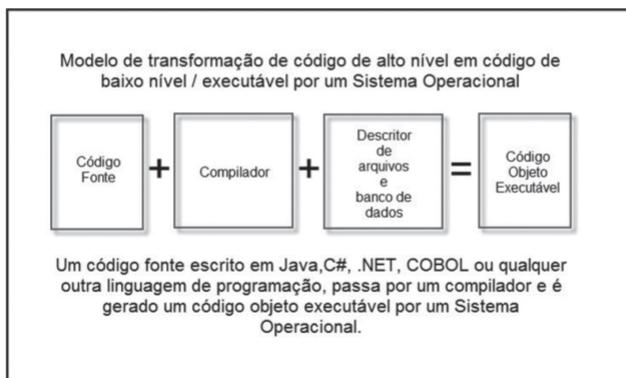
O que acontece se uma definição for modificada? Simplesmente, é necessário realizar os ajustes no programa código fonte e realizar uma nova compilação.

Essa ação seria simples se o sistema tiver poucos programas de computador, pois assim o trabalho do programador seria facilmente desenvolvido por ele ou até mesmo por um estagiário.

Agora, imagine se o sistema possuir uma centena ou milhares de programas. Todos os programas teriam que ser corrigidos e recompilados, perdendo-se muito tempo e ainda correndo o risco de deixar programas para trás, sem alteração/correção, que seriam somente descobertos na hora de usar, pois daria erro de versão.

A Figura 2.8 apresenta um modelo de transformação ou compilação de um código fonte (programa de computador em alto nível) em um código objeto executável (programa de computador em baixo nível, comprehensível pelo Sistema Operacional).

Figura 2.8 | Código Fonte + Compilador + Descritor de Banco de Dados = Código Objeto Executável



Fonte: elaborada pelo autor.

A independência de dados possui duas divisões, as quais serão estudadas a seguir, são elas: a Independência Lógica de Dados e a Independência Física de Dados. Cada uma delas apresenta a sua característica própria e que a torna única para diferenciar um banco de dados de um gerenciador de arquivos convencional.

### 3.4 Independência Lógica de Dados

Silberschatz, Korth e Sudarshan (2004) consideram a Independência Lógica a modificação do esquema lógico sem que os programas precisem ser reescritos.

A Independência Lógica de Dados é a capacidade de se modificar o esquema conceitual de um banco de dados sem afetar os programas de computador que já foram compilados para acessar o referido banco de dados.

Um exemplo seria a inclusão de novas entidades e de novos relacionamentos, os quais irão se tornar novas tabelas no banco de dados.

Se for gerenciar arquivos convencionais, seria acrescentar novos arquivos a serem acessados.

Como um programa código fonte, no caso de acessos a um banco de dados, faz referência apenas ao banco de dados, novas tabelas não afetam essa definição, com isso não é necessário recompilar os programas que acessam o banco de dados.

Já no caso dos arquivos convencionais, dentro do programa código fonte são relacionadas todas as descrições de onde encontraremos os arquivos convencionais, um por um, ou seja, se um novo arquivo convencional for incluído no sistema, ele deverá ser referenciado e descrito em todos os programas que fizerem acesso ao referido arquivo convencional. É o que nós podemos chamar de *file description*, ou seja, descriptor de arquivos dentro dos programas de computador.

Só essa característica já nos permite realizar inúmeras manutenções nos projetos de banco de dados sem ter que ficar verificando todos os programas e os recompilando um a um.



Para saber mais

A Independência Lógica de Dados é a mais difícil de ser conseguida, principalmente porque, quando os programas código fonte começam a ser escritos ou codificados, o banco de dados já está praticamente definido. E se surgirem modificações no projeto do banco de dados, geralmente, elas envolverão um nível de modificação que a propriedade de independência de dados não consegue cumprir.

### 3.5 Independência Física de Dados

De acordo com Silberschatz, Korth e Sudarshan (2004), a Independência Física consiste em modificar o esquema físico sem ter que reescrever os programas aplicativos.

Ela é a capacidade de se modificar o esquema físico de um banco de dados sem afetar os programas de computador que já foram compilados para acessar o referido banco de dados.

Um exemplo seria trocar uma tabela do banco de dados de local de armazenamento, passar de um diretório de disco para outro diretório de outro disco, ou mesmo mudar o banco de dados de lugar de armazenamento físico.

Se for em gerenciador de arquivos convencionais, no local onde é descrita toda a referência do arquivo convencional, precisa ser corrigida para o novo local de armazenamento dos arquivos a serem acessados e, consequentemente, o programa precisa ser recompilado.

Ex. "file A (C:/sistema contabil / arquivos / clientes.arq)"

Mudar para:

"file A (D:/sistema geral / arquivos / clientes.arq)"

Como um programa código fonte, no caso de acessos a um banco de dados, faz referência apenas ao banco de dados, não importando onde o banco de dados está armazenado, se o banco de dados mudar de lugar, isso não afeta os programas de computador já compilados. E assim não é necessário recompilar todos os programas que acessam o banco de dados.

No caso dos arquivos convencionais, como a localização dos arquivos precisa ser corrigida, todos os programas passarão por essa manutenção e, consequentemente, por uma nova recompilação.



Para saber mais

A Independência Física de Dados não contempla modificações físicas de armazenamento, por exemplo, acrescentar um novo campo na tabela, alterar o tamanho de um campo já existente na tabela, alterar o tipo de dado de um campo já existente em uma tabela, criação de um novo índice de acesso a uma tabela, modificação em chaves primárias ou chaves estrangeiras de uma tabela existente.

Todos esses tipos de modificação física descritos afetarão diretamente os programas de computador, alterando as características de acesso aos dados, a leitura das informações e a transferência do dado para as variáveis locais dos programas de computador, por isso será necessário adaptar os programas código fonte para esta nova realidade e, consequentemente, será necessário recompilar os programas código fonte, gerando novos códigos objetos executáveis.

Para obter mais conhecimento, você pode acessar os links a seguir:

- Dataprix independência de dados. Disponível em: <<http://www.dataprix.net/pt-pt/42-independencia-dados>>. Acesso em: 2 ago. 2017.
- Wilton Poleska: independência de dados. Disponível em: <<http://wiltonpolesca.blogspot.com.br/2012/04/banco-de-dados-independencia-de-dados.html>>. Acesso em: 2 ago. 2017.
- Carlos Eduardo: independência de dados. Disponível em: <<https://pt.linkedin.com/pulse/banco-de-dados-independ%C3%A3ncia-carlos-eduardo>>. Acesso em: 2 ago. 2017.



Questão para reflexão

Por que os sistemas operacionais não evoluíram o seu mecanismo de controle de arquivos convencionais para poder concorrer com os SGBDs?

### 3.6 Propriedades ACID de um banco de dados

Ainda descrevendo sobre características que o SGBD possui e o gerenciador de arquivos convencionais não possui, podemos citar as propriedades ACID (atomicidade, consistência, isolamento e durabilidade).

Essas propriedades incidem sobre os dados que estão armazenados no banco de dados e sobre as transações que são realizadas com os dados.

As propriedades ACID foram o grande fator de confiabilidade que o SGBD apresentou ao mercado consumidor de tecnologia e ajudaram na transformação dos sistemas de acesso a arquivos convencionais em sistemas de acesso a banco de dados nas décadas de 1970 e 1980.

### 3.6.1 Propriedade Atomicidade

A propriedade atomicidade está diretamente ligada à transação realizada pelo SGBD. No momento que uma transação inicia, várias operações podem ser realizadas nela (*insert, update e delete*), contemplando diversos dados diferentes e, ao final delas, um comando *Commit* finaliza a transação. A quantidade de operações dentro da mesma transação, geralmente, está ligada a uma Regra de Negócio, a qual é exatamente sustentada pela propriedade atomicidade, que vai garantir que ou todas as operações realizadas dentro da transação são executadas até o fim, ou nenhuma operação será executada, portanto não existe a “meia transação”. É o “tudo ou nada” da transação de banco de dados.

### 3.6.2 Propriedade Consistência

A propriedade consistência está intimamente ligada às Regras de Negócio que podem ser estabelecidas dentro do banco de dados. Podem ser regras simples, como atribuir valores pré-determinados a um campo (ex.: masculino e feminino), ou a um campo tipo DATA (ex.: 30 de fevereiro não passa como valor válido), ou tentar gravar caracteres em campo numérico, ou atribuir um valor diferente de *true* ou *false* para um campo booleano, ou uma regra na qual um determinado campo numérico não pode ser inferior a 10.

A partir do momento em que um valor qualquer atribuído a um campo de um registro de uma tabela violar uma regra de negócio, a propriedade consistência entra em ação, bloqueando a transação, independentemente da sua origem (acesso interativo direto no banco de dados ou programa de computador, executando uma transação).

O dado é consistente antes de iniciar a transação e deverá continuar consistente ao finalizar a transação.

### 3.6.3 Propriedade Isolamento

A propriedade isolamento já afeta o cenário no qual dois ou mais

programas de computador estejam realizando transações simultâneas no mesmo banco de dados. Como sabemos, o SGBD tem a capacidade de controlar várias transações realizadas ao mesmo tempo no mesmo banco de dados (mesma instância).

O que a propriedade isolamento prega é que uma transação não afeta a outra transação, desde que sejam em dados distintos. Se ocorrer de uma transação necessitar dos dados que estejam sendo manipulados em outra transação, aquela transação vai esperar para que esta transação termine e libere o dado desejado, sem provocar interferência na transação ou mesmo fazer a leitura do dado antigo e que não refletirá mais a nova realidade do dado.

### 3.6.4 Propriedade Durabilidade

A propriedade durabilidade tem relação com a perenidade dos dados, ou seja, não importa quando a transação foi realizada, se ela está registrada no banco de dados, este dado será eterno até que outra transação o modifique ou remova do banco de dados.

Uma vez que o dado foi inserido no banco de dados, somente outra transação tem a capacidade de modificar ou remover o dado. Sem que exista uma transação, o dado não pode sumir, desaparecer ou modificar o seu valor ou conteúdo de forma espontânea.

Se um dado foi gravado há 10 anos e ninguém mais o alterou ou removeu durante todo esse período, ele deverá continuar o mesmo da época em que foi gravado inicialmente e não pode se perder com o tempo.



Para saber mais

Consulte a enciclopédia livre Wikipédia para saber mais.

Disponível em: <<https://pt.wikipedia.org/wiki/ACID>>. Acesso em: 2 ago. 2017.

Você também pode acessar os links a seguir:

- TechNet: Transações. Disponível em: <[https://technet.microsoft.com/pt-br/library/ms190612\(v=sql.105\).aspx](https://technet.microsoft.com/pt-br/library/ms190612(v=sql.105).aspx)>. Acesso em: 2 ago. 2017.
- SQL para todos: modelo ACID SQL. Disponível em: <<http://sqlparatodos.com.br/modelo-acid-sql/>>. Acesso em: 2 ago. 2017.



## Questão para reflexão

Por que os sistemas operacionais não evoluíram o seu mecanismo de controle de arquivos convencionais para poder concorrer com os SGBDs?

## Atividades de aprendizagem

- 1.** Considerando que um Sistema Gerenciador de Banco de Dados (SGBD) é composto por uma série de programas de computador, que são ferramentas para a construção e manutenção dos bancos de dados. Quando estas ferramentas estão carregadas na memória do servidor e o banco de dados também está carregado, qual é o nome dado ao conjunto de ferramentas + banco de dados carregados que reflete um momento do banco de dados?
  - a) Esquema.
  - b) Cache.
  - c) Instância.
  - d) *Memory Buss*.
  - e) Barramento de Dados.
  
- 2.** No paradigma da independência de dados, temos duas vertentes: a independência lógica de dados e a independência física de dados. É através dessas características que os programas de computador que acessam banco de dados são compilados muito menos vezes do que os programas de computador que acessam os arquivos convencionais, principalmente no momento de mudanças lógicas ou físicas dos dados. No caso em que um banco de dados precisa ser realocado de um “file system” para outro “file system” que fica em um disco diferente, qual é o tipo de independência de dados apresentado?
  - a) Independência Lógica de Dados.
  - b) Independência Física de Dados.
  - c) Independência Lógica de Tabelas.
  - d) Independência Física de *File Systems*.
  - e) Independência Física de *Row Devices*.

## Fique ligado

Na Unidade 2, apresentamos alguns dos conceitos que fazem com que o SGBD tenha uma maior confiabilidade no armazenamento dos dados do que os arquivos convencionais.

Relembrando que termos, como “instância” e “esquema”, possuem uma definição clássica, mas podem ser utilizados com outras finalidades pelos SGBDs de mercado.

Quando o assunto é independência de dados, tanto a independência lógica como a independência física são características que facilitam muito o dia a dia dos analistas e programadores, principalmente no assunto manutenção de sistemas.

E as propriedades ACID (atomicidade, consistência, isolamento e durabilidade) fecham um conjunto de características e propriedades que tornam um SGBD (sistema gerenciador de banco de dados) definitivamente superior aos gerenciadores de arquivos convencionais ou mesmo gerenciadores de tabelas (Ms-Access Microsoft).

Mas se tem uma coisa no mundo da tecnologia da informação que é certeira é que tudo muda, ou seja, o que é um conceito certo hoje, amanhã pode não ser mais.

## Para concluir o estudo da unidade

Para concluir esta unidade, esperamos que você tenha alcançado o seu objetivo inicial de estudo e, assim, concluir a sua meta de aprendizado. Aproveite para pesquisar na internet sobre os assuntos abordados aqui. E se desejar, aproveite para rever os links, vídeos ou outros livros, consulte as seções *Para saber mais* e as nossas referências bibliográficas.

## Atividades de aprendizagem da unidade

**1.** Os SGBDs de mercado apresentam um conjunto de propriedades que auxiliam e definem os controles sobre as transações que ocorrem em um banco de dados e também sobre o armazenamento dos dados dentro do banco de dados. Essas propriedades são conhecidas pela sigla ACID.

Qual é a propriedade que garante que um dado que foi gravado no banco de dados não terá o seu valor alterado de maneira espontânea, mas somente através de uma nova transação?

- a) Atomicidade.
- b) Consistência.
- c) Isolamento.
- d) Durabilidade.
- e) Comprometimento.

**2.** Durante um projeto de banco de dados, o analista de sistema identificou uma situação na qual foi necessária a inclusão de mais uma tabela no banco de dados. Quando ele conversou com o administrador do banco de dados, solicitando essa inclusão, aproveitou para avisar que, assim que estivesse pronta a manutenção, ele iria recompilar todos os programas do sistema aplicativo. Porém, o administrador do banco de dados o tranquilizou, dizendo que não seria necessária essa recompilação total, pois o banco de dados tinha uma característica que o auxiliaria.

Qual é a característica que o administrador do banco de dados se referiu?

- a) Independência Lógica de Dados.
- b) Independência Física de Dados.
- c) Independência Lógica de Tabelas.
- d) Independência Física de File Systems.
- e) Independência Física de Row Devices.

**3.** Considerando que a Linguagem de Manipulação de Dados (DML – *Data Manipulation Language*) apresenta os comandos mais utilizados dentro dos programas de computador ou mesmo pelos usuários do banco de dados (analistas e programadores) e que é composta, basicamente, por quatro comandos padrão SQL. Sabemos que um dos comandos não atualiza o banco de dados, provoca apenas a leitura deles. Já os outros três comandos provocam alterações e modificações no conteúdo do banco de dados.

Quais seriam os três comandos padrão SQL que provocam as alterações no banco de dados?

- a) Select, Insert e Delete.
- b) Create, Alter e Drop.
- c) Insert, Update e Delete.
- d) Insert, Create e Shutdown.
- e) Shutdown, Startup e Recovery.

**4.** Considerando que um autêntico SGBD deve ter como um dos princípios de funcionamento as propriedades ACID de uma transação, e que graças a estas propriedades os SGBDs superaram os gerenciadores de tabelas e arquivos, apresentando uma confiabilidade extrema nos dados armazenados; Qual é a propriedade que nos garante que os dados armazenados no banco de dados estarão de acordo com as regras de negócio estabelecidas, com as lógicas inteiras e coerentes com o projeto estabelecido?

- a) Atomicidade.
- b) Compatibilidade.
- c) Isolamento.
- d) Durabilidade.
- e) Consistência.

**5.** Considere que um banco de dados já exista há mais de 2 anos; que os backups são realizados todo final de semana; e que os arquivos de log das transações também sejam guardados e protegidos, mas um incidente/sinistro acontece com o servidor do banco de dados e vários discos rígidos, nos quais o banco de dados estava armazenado, ficam em estado de falha total, sendo necessária a troca por discos rígidos novos. Após o técnico realizar a troca dos discos rígidos, o banco de dados não está armazenado nos discos novos.

Qual é o nome do procedimento a ser executado para a recuperação e restauração do banco de dados através das cópias de segurança (backup)?

- a) *Disaster-Reconstruct.*
- b) *Catastrophic-Recovery.*
- c) *Disaster-Recovery.*
- d) *Detonation-Reconstruct.*
- e) *Begin-Rebuild-DataBase.*

# Referências

CHEN, Peter. **Modelagem de Dados**. São Paulo: McGraw-Hill, 1990.

NAVATHE, S. B.; ELMASRI, R. **Sistemas de banco de dados**. 4. ed. São Paulo: Pearson Education do Brasil Ltda., 2005.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistema de banco de dados**. 4. ed. São Paulo: Makron Books, 2004.

# Modelo de dados e modelagem entidade relacionamento

Roberto Yukio Nishimura

## Objetivos de aprendizagem

- Estudar os conceitos, elementos e funcionalidades do modelo de dados.
- Estudar os conceitos, elementos, funcionalidade da modelagem entidade relacionamento.

## Seção 1 | Modelo de dados: conceitos, elementos e funcionalidades

Nesta seção, apresentaremos a definição de modelo de dados, que é um pré-requisito para a modelagem de dados. Iniciamos pelo projeto conceitual de dados, passamos pelo projeto lógico de banco de dados e finalizaremos no projeto físico de banco de dados.

## Seção 2 | Modelo Entidade Relacionamento – MER

Nesta seção, apresentaremos o conceito do MER (modelo entidade relacionamento), detalhando sobre as entidades, os relacionamentos e os atributos. O DER (diagrama entidade relacionamento) é explicado e como utilizá-lo.

## Seção 3 | Cardinalidades e Tipos de Atributos

Nesta seção, apresentaremos as cardinalidades de relacionamento, como realizar a sua leitura e compreensão e também todos os tipos de classificação de atributos.



# Introdução à unidade

Caro (a) aluno (a), nesta unidade apresentaremos os conceitos relacionados a modelo de dados e a modelagem entidade relacionamento (MER), todos os seus componentes e funcionalidades, a forma de aplicar a técnica de modelagem gerando o diagrama entidade relacionamento (DER). Ao estabelecer os relacionamentos entre as entidades, surge a cardinalidade do relacionamento e finalizamos com os tipos de classificação de atributos das entidades.

Na Seção 1 iniciamos apresentando o conceito de modelo de dados, suas principais características e funcionalidades. O projeto conceitual de dados, o projeto lógico de banco de dados e o projeto físico de banco de dados são detalhados e comparados, demonstrando onde são independentes dos SGBDs (sistema gerenciador de banco de dados) de mercado e onde são totalmente dependentes deste.

Na Seção 2 começamos conceituando a técnica de modelagem de dados modelo entidade relacionamento (MER). Entidades, relacionamentos e atributos são detalhados e comparados com o mundo real. Para poder expressar e representar o resultado da técnica aplicada, o diagrama entidade relacionamento (DER) é demonstrado.

Na Seção 3 explicamos a cardinalidade do relacionamento, que permite fazer a leitura de todos os relacionamentos de acordo com as regras de negócio. Explicando as relações 1-1, 1-N e N-N. Finalizamos apresentando os tipos possíveis de atributos e sua utilização.

# Seção 1

## Modelo de Dados

### Introdução à seção

Caro (a) aluno (a), aqui na Seção 1 vamos iniciar com o processo de modelagem de banco de dados, ele é dividido em 3 partes (o projeto conceitual de dados, depois o projeto lógico de dados e finaliza com o projeto físico de dados). Cada uma destas etapas tem as suas características próprias e seus objetivos distintos também.

Vamos apresentar as características que tornam os SGBDs (sistema gerenciador de banco de dados) de mercado independentes e os pontos onde são totalmente dependentes deste.

### 1 Modelo de dados

#### 1.1 Definição

“Um projeto de banco de dados pode ser dividido em duas etapas: projeto lógico e projeto físico” CHEN (1990, p. 5). De acordo com NAVATHE e ELMASRI (2005), um projeto de banco de dados inicia-se pelo projeto conceitual, passa pelo projeto lógico e finaliza com o projeto físico. Desta forma, iniciamos o modelo de dados que vai dar origem ao banco de dados.

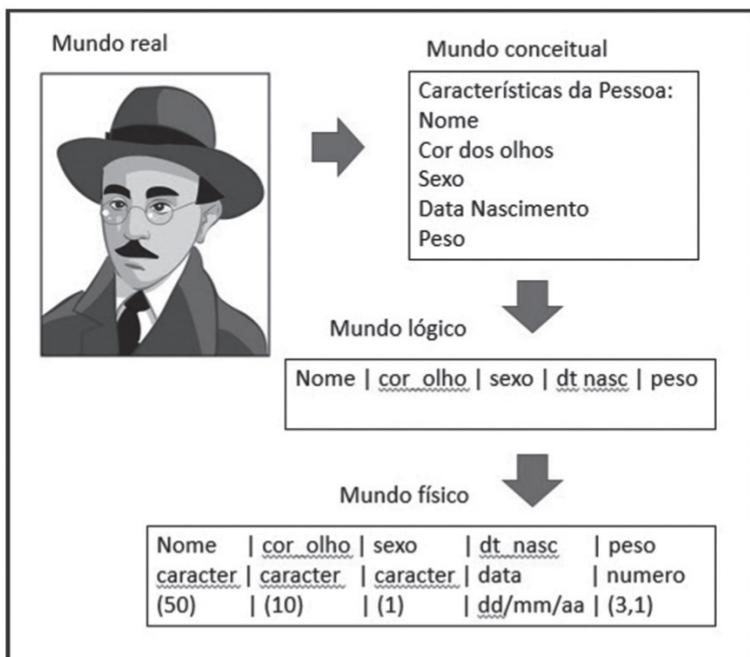
O projeto conceitual tem a finalidade de tentar aproximar o mundo real (que vai ser informatizado) do mundo computacional. Através de regras de modelagem, o mundo real começa a tomar formas lógicas ao mundo da computação.

“O projeto lógico de banco de dados é o processo de planejar a estrutura lógica de dados para o banco de dados” CHEN (1990, p. 5). O projeto lógico de banco de dados apresenta as tabelas já definidas com seus campos e o tipo do campo atribuído, ficando opcional o estabelecimento do tamanho dos campos. Porém quando utilizamos uma ferramenta CASE (*Computer-Aided Software Engineering*) e já definirmos o tamanho dos campos, isto facilitará a transformação do projeto lógico em um projeto físico.

"O projeto físico de banco de dados é o processo de selecionar uma estrutura física de dados para uma dada estrutura lógica de dados" CHEN (1990, p. 5). O projeto físico de banco de dados já tem como objetivo aproximar-se ao SGBD (sistema gerenciador de banco de dados) escolhido, uma vez que a representação se faz pelos comandos padrão SQL que irão materializar as tabelas, campos e relacionamentos no banco de dados.

A Figura 3.1 – Modelo de dados demonstra uma sequência da modelagem de dados, onde iniciamos no mundo real, passamos pelo mundo conceitual, depois pelo mundo lógico e finalizamos no mundo físico.

Figura 3.1 | Modelo de dados



Fonte: elaborada pelo autor.

## 1.2 Projeto Conceitual

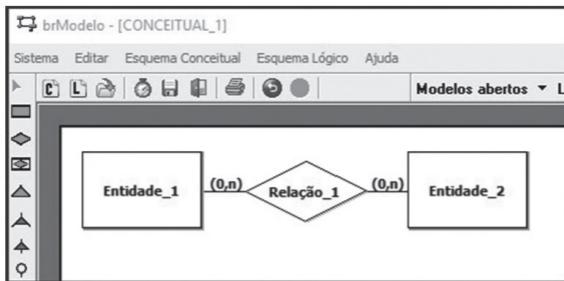
“O esquema conceitual é uma descrição concisa dos requisitos de dados dos usuários e inclui descrições detalhadas de tipos de entidade, relacionamentos e restrição” NAVATHE e ELMASRI (2005, p. 36).

O projeto conceitual tem por objetivo apresentar uma representação conceitual do armazenamento de dados que vai refletir o mundo real. Ou seja, olhando-se para o cenário a ser informatizado, identificam-se os dados que serão armazenados no banco de dados e propõe-se uma organização lógica obedecendo as regras de negócio do cenário.

Neste contexto, surgem as estruturas de armazenamento que dependendo da técnica de modelagem a ser utilizada, são chamadas de entidades ou de objetos.

A Figura 3.2 – Projeto Conceitual apresenta como um projeto conceitual é representado dentro da ferramenta Br-Modelo.

Figura 3.2 | Projeto Conceitual



Fonte: elaborada pelo autor.

## 1.3 Projeto Lógico

“O projeto lógico de banco de dados é o processo de planejar a estrutura lógica de dados para o banco de dados” CHEN (1990, p. 5).

O projeto lógico tem por objetivo apresentar uma representação lógica do armazenamento dos dados que estão sendo modelados. Nesta fase importante do projeto, já temos as tabelas definidas, os campos das tabelas e os relacionamentos que envolvem as tabelas.

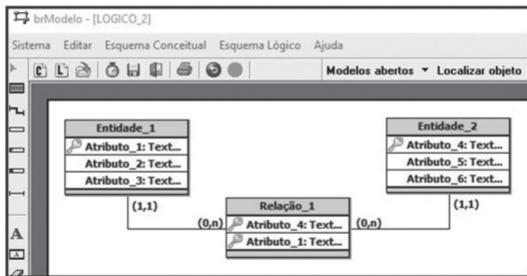
O tipo de dado de cada campo já está definido e o tamanho de cada campo. Nesta fase do projeto é importante definir se um campo será caractere, numérico, booleano, etc. mas não nos importamos com a sintaxe que será utilizada (ex. Varchar, Character, String).

Por apresentar estas características, um projeto lógico é

independente de fornecedor ou tipo ou marca de banco de dados / SGBD (sistema gerenciador de banco de dados).

A Figura 3.3 – Projeto Lógico apresenta como um projeto lógico é representado dentro da ferramenta Br-Modelo.

Figura 3.3 | Projeto Lógico



Fonte: elaborada pelo autor.

## 1.4 Projeto Físico

"O projeto físico de banco de dados é o processo de selecionar uma estrutura física de dados para uma data estrutura lógica de dados" CHEN (1990, p. 5).

O projeto físico de um banco de dados tem por objetivo representar as tabelas, os campos e os relacionamentos já com os comandos específicos de um SGBD (sistema gerenciador de banco de dados) de mercado, ou seja, os comandos que permitirão a criação ou alteração ou mesmo a remoção das estruturas de armazenamento (tabelas).

Geralmente um projeto físico é gerado por uma ferramenta CASE (*computer-aided software engineering*) e com isto está livre de erros de sintaxe. Quando construímos um projeto físico na mão, geralmente erros de sintaxe acontecem e podem atrapalhar o processo de manutenção de um banco de dados.

A Figura 3.4 – Projeto Físico apresenta como um projeto físico é representado dentro da ferramenta Br-Modelo.

Figura 3.4 | Projeto Físico

```
-- Geração de Modelo físico  
-- Sql ANSI 2003 - brModelo.  
  
CREATE TABLE Entidade_1 (  
Atributo_1 Varchar2(1) PRIMARY KEY,  
Atributo_2 Varchar2(1),  
Atributo_3 Varchar2(1));  
  
CREATE TABLE Entidade_2 (  
Atributo_4 Varchar2(1) PRIMARY KEY,  
Atributo_5 Varchar2(1),  
Atributo_6 Varchar2(1));  
  
CREATE TABLE Relação_1 (  
Atributo_4 Varchar2(1),  
Atributo_1 Varchar2(1),  
FOREIGN KEY(Atributo_4) REFERENCES Entidade_2 (Atributo_4));
```

Fonte: elaborada pelo autor



### Para saber mais

Vale uma pesquisa na encyclopédia livre Wikipédia, disponível em: <[https://pt.wikipedia.org/wiki/Modelagem\\_de\\_dados](https://pt.wikipedia.org/wiki/Modelagem_de_dados)>. Acesso em: 8 ago. 2017.

Link para o livro *Projeto de Banco de Dados* do prof. Dr. Carlos Alberto Heuser: <[http://www.fernandozaidan.com.br/pit-grad/Diversos/Livros\\_Disciplinas/Projeto\\_de\\_Banco\\_de\\_Dados\\_-\\_Carlos\\_Alberto\\_Heuser.pdf](http://www.fernandozaidan.com.br/pit-grad/Diversos/Livros_Disciplinas/Projeto_de_Banco_de_Dados_-_Carlos_Alberto_Heuser.pdf)>. Acesso em: 2 set. 2017.

Link para o material de José Antonio da Cunha, *Projeto de Banco de Dados* - <<https://docente.ifrn.edu.br/josecunha/disciplinas/banco-dados/material-de-aula/introducao-bd>>. Acesso em: 2 set. 2017.

O site Devmedia também é uma referência no mundo da Tecnologia da Informação.

<<http://www.devmedia.com.br/introducao-a-modelagem-dados/24953>>. Acesso em: 8 ago. 2017.



### Questão para reflexão

"O processo de construção de um banco de dados deve ser muito bem realizado, pois, depois de implementado e populado, se forem necessárias mudanças estruturais profundas em um banco de dados pode ser muito trabalhoso. É como a construção de uma casa, deve-se fazer todos os projetos antes da construção, pois, uma vez construída, qualquer reforma geralmente dá muito trabalho".

Por este motivo, a construção de um banco de dados é parte fundamental para o sucesso de um bom sistema de informação.

**Links:**

- Tutorial – Planeje seu modelo de dados: <http://www.linhadecodigo.com.br/artigo/332/planeje-o-seu-modelo-de-dados.aspx> - Acesso em: 8 ago. 2017.
- Tutorial – Modelo de dados - UFSC: <<http://www.inf.ufsc.br/~r.mello/ine5423/3-modeloRelacional.pdf>>. Acesso em: 8 ago. 2017.

## Atividades de aprendizagem

**1.** Um projeto de banco de dados consiste em 3 etapas distintas, projeto conceitual, projeto lógico e projeto físico. Cada uma destas etapas possui a sua finalidade específica e apresenta seus resultados próprios.

Em qual etapa tratamos o armazenamento dos dados com estruturas de armazenamento denominados “entidades”?

- a) Projeto Conceitual e Projeto Lógico.
- b) Projeto Conceitual.
- c) Projeto Lógico.
- d) Projeto Lógico e Projeto Físico.
- e) Projeto Físico.

**2.** Considere que um projeto de banco de dados é composto por 3 partes distintas. O projeto total se inicia pelo projeto conceitual de dados onde são feitas as comparações com o mundo real. Passamos pelo projeto lógico onde as tabelas e campos e os relacionamentos são apresentados, E finalizamos com o projeto físico que tem os comandos em padrão SQL para a construção das tabelas e relacionamentos dentro do banco de dados. Qual é a etapa dos projetos onde o SGBD que será utilizado é relevante e extremamente importante?

- a) Projeto Conceitual.
- b) Projeto Lógico.
- c) Projeto Físico.
- d) Em todos os projetos (conceitual, lógico e físico).
- e) Em nenhum dos projetos (conceitual, lógico e físico).

# Seção 2

## Modelo Entidade Relacionamento - MER

### Introdução à seção

Caro (a) aluno (a), na Seção 2 o assunto tratado será o MER (modelo entidade relacionamento). Vamos abordar cada um dos componentes, suas definições, características e representação geométrica.

### 2 Ferramentas de Administração do Banco de Dados

#### 2.1 Definição

"Modelo Entidade Relacionamento" (MER), que é um modelo de dados conceitual de alto nível, além de muito popular" NAVATHE e ELMASRI (2005, p. 35).

A técnica de modelagem diz que devemos pegar o cenário dos dados que serão informatizados e nele identificar as entidades, os atributos das entidades e os relacionamentos entre as entidades.

Este cenário do mundo real deve ser descrito na forma de um texto, apresentando tudo que o analista identificar.

Após realizar esta etapa, tudo isto deverá ser expresso através de um DER (diagrama entidade relacionamento) que é a forma de apresentar os componentes identificados dentro de um diagrama.

#### 2.2 Entidade

"O objeto básico que o modelo ER representa é uma entidade, algo do mundo real, com uma existência independente" NAVATHE e ELMASRI (2005, p. 39).

Uma entidade pode ser identificada através dos substantivos dentro da descrição textual narrativa feita pelo analista de sistemas. Ou seja, todo substantivo é um potencial candidato a ser uma entidade dentro do seu projeto conceitual. Vale observar que nem todos os substantivos serão uma entidade dentro do seu banco de dados, pois muita coisa inútil será descrita no texto.

A entidade é representada por um retângulo dentro do DER

(diagrama entidade relacionamento).

Entidade forte é aquela entidade que possui atributos próprios o suficiente para provocar a sua existência sem a dependência de nenhuma outra entidade.

Entidade fraca é aquela entidade que não tem todos os atributos necessários para uma existência sozinha, ou seja, depende de outra entidade e de um relacionamento entre elas para que a mesma tenha sentido de existência válido. É representada por um retângulo dentro de outro retângulo.

Entidade concreta é aquela que representa um substantivo real, materializável, tocável. Exemplo: carro, árvore, pessoa física, animal.

Entidade abstrata é aquela que representa um substantivo irreal, uma ideia, não possível tocá-la. Exemplo: conta corrente bancária, nota fiscal, documento de identidade, pessoa jurídica.

A Figura 3.5 – Entidade apresenta como uma entidade é representada dentro da ferramenta Br-Modelo.

Figura 3.5 | Entidade



Fonte: elaborada pelo autor.

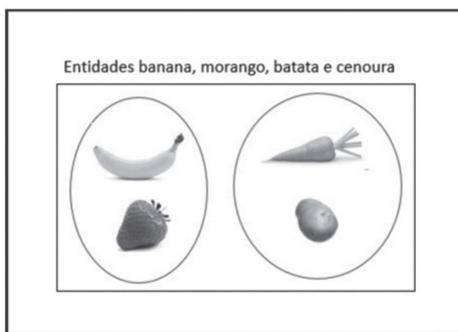
Conjunto de ocorrências – as entidades que possuem características semelhantes ficam agrupadas umas com as outras e a esta característica chamamos de Conjunto de Ocorrências de uma Entidade ou simplesmente Entidade.

Ex. temos 4 entidades diferentes, uma banana, uma batata, um morango e uma cenoura. Se formos agrupar estas entidades por características em comum, seria o óbvio agrupar a entidade banana com a entidade morango e ao conjunto destas entidades chamar de entidade Fruta e as entidades batata e cenoura em outro conjunto de nome Legumes ou Tubérculos.

A Figura 3.6 – Conjunto de Entidades apresenta a ideia que as

entidades que possuem características em comum devem ficar juntas umas com as outras, formando assim o conjunto de entidade.

Figura 3.6 | Conjunto de Entidades



Fonte: elaborada pelo autor.

### 2.3 Relacionamento

"Quando um atributo de uma entidade refere-se a uma outra entidade há um relacionamento" NAVATHE e ELMASRI (2005, p. 43).

O relacionamento é o que une duas entidades ou mais, representa uma regra de negócio estabelecida entre as entidades. Na descrição textual narrativa pode ser identificado através dos verbos. No momento da modelagem, devemos observar que somente as regras de negócio mais significativas têm relevância para o modelo de dados, vamos desprezar as regras desnecessárias ou irrelevantes.

O relacionamento é representado por um losango no DER (diagrama entidade relacionamento).

A Figura 3.7 – Relacionamento apresenta como um relacionamento é representado dentro da ferramenta Br-Modelo.

Figura 3.7 | Relacionamento



Fonte: elaborada pelo autor.

### 2.3.1 Tipos de Relacionamento

Segundo CHEN (1990), os relacionamentos podem ser entre duas entidades ou entre mais de duas entidades.

NAVATHE e ELMASRI (2005), determinam o tipo de relacionamento pelo grau, grau dois de binário e grau três de ternário.

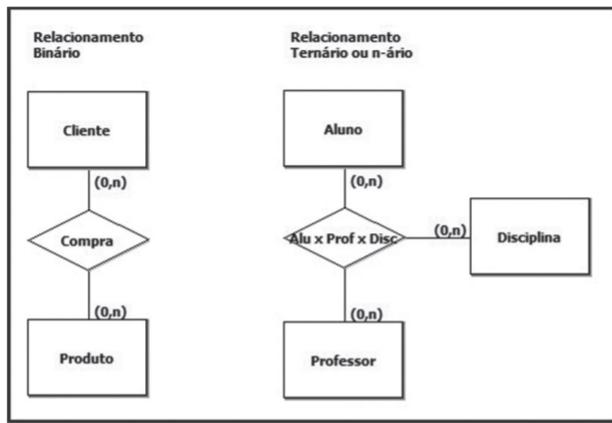
Relacionamento binário é quando temos um relacionamento entre duas entidades e somente duas.

Relacionamento ternário ou n-ário é quando temos mais de duas entidades participando do mesmo relacionamento.

Vários SGBDs (sistema gerenciador de banco de dados) não conseguem implementar fisicamente um relacionamento ternário, para resolver este problema, aplicamos o MRN (modelo relacional normalizado) que é um conjunto de regras que transforma o nosso MER (modelo entidade relacionamento) desnormalizado em um MRN (modelo relacional normalizado) e pronto para ser implementado no banco de dados.

A Figura 3.8 – Tipos de Relacionamento apresenta como são representados no DER (diagrama entidade relacionamento) um relacionamento binário e um ternário, isto tudo no projeto conceitual.

Figura 3.8 | Tipos de Relacionamento



Fonte: elaborada pelo autor.

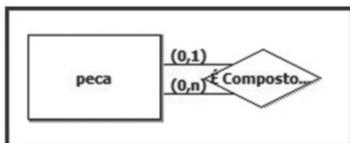
### 2.3.2 AutoRelacionamento

Segundo CHEN (1990), é possível uma entidade se relacionar com ela mesma através de um relacionamento que inicia na entidade e finaliza na mesma entidade.

AutoRelacionamento é quando uma entidade se relaciona consigo mesma. Podemos exemplificar desta forma: uma entidade peça se relaciona consigo mesma através de um relacionamento “é composto por” outras peças. Seria um relacionamento de 1 peça é composto por N peças.

A Figura 3.9 – AutoRelacionamento apresenta um exemplo de autorelacionamento, em que uma peça é composta por várias peças.

Figura 3.9 | AutoRelacionamento



Fonte: elaborada pelo autor.

### 2.3.3 Integridade Referencial

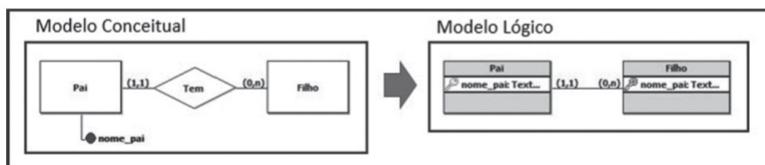
Segundo CHEN (1990), os relacionamentos são identificados pelo uso de identificadores das entidades envolvidas no relacionamento.

A integridade referencial é uma regra implícita imposta por um relacionamento entre uma entidade Forte e uma entidade Fraca. Neste relacionamento a chave primária (*Primary Key*) (lógico e físico) da entidade forte será doada como uma chave estrangeira (*Foreign Key*) e fazendo parte da chave primária da entidade Fraca.

No projeto conceitual, isto não aparece, só fica evidente no projeto lógico e físico.

A Figura 3.10 – Integridade Referencial apresenta em dois momentos distintos, no modelo conceitual e no modelo lógico, como ficam representados e demonstrados a integridade referencial entre as entidades participantes do relacionamento.

Figura 3.10 | Integridade Referencial



Fonte: elaborada pelo autor.

## 2.4 Atributo

"Atributo – propriedades particulares que a descrevem" NAVATHE e ELMASRI (2005, p.39).

O atributo representa o dado que será armazenado no banco de dados, geralmente pertence a uma entidade, mas existem relacionamentos que podem ter atributos também (no projeto conceitual). Na descrição textual narrativa podemos identificar um atributo como sendo os adjetivos de um substantivo, ou seja, as qualidades do substantivo.

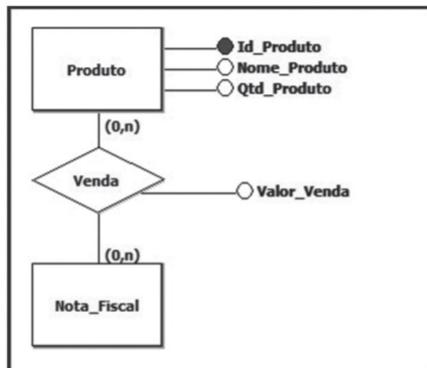
O atributo é representado por uma elipse no DER (diagrama entidade relacionamento).

A ferramenta BrModelo utiliza um círculo para representar o atributo (círculo vazio é um atributo normal e um círculo preto é um atributo determinante).

Como o atributo é o dado em sua essência, o mesmo pode ter características como caractere, alfanumérico, numérico inteiro ou com decimais, data (em qualquer formato), booleano ou número real.

A Figura 3.11 – Atributo apresenta uma forma de representação das entidades, dos relacionamentos e dos atributos, utilizada pela ferramenta BrModelo, em que os atributos são os atributos são uma circunferência pequena, ligadas a uma entidade ou a um relacionamento.

Figura 3.11 | Atributo



Fonte: elaborada pelo autor.



Para saber mais

Vale uma pesquisa na enclopédia livre Wikipédia, disponível em: <[https://pt.wikipedia.org/wiki/Modelo\\_entidade\\_relacionamento](https://pt.wikipedia.org/wiki/Modelo_entidade_relacionamento)>. Acesso em: 10 ago. 2017.

Devmedia: Modelo Entidade Relacionamento (MER) e Diagrama Entidade Relacionamento (DER), disponível em: <<http://www.devmedia.com.br/modelo-entidade-relacionamento-mer-e-diagrama-entidaderelacionamento-der/14332>>. Acesso em: 10 ago. 2017.



Questão para reflexão

"O gerenciamento de dados tornou-se uma das atividades mais importantes em muitas organizações" CHEN (1990, p1). Com esta afirmação, Peter Chen inicia a apresentação da técnica de modelagem Entidade-Relacionamento na década de 70.

A técnica de modelagem de dados MER (modelo entidade relacionamento) inicialmente adotada pela IBM em seu SGBD (sistema gerenciador de banco de dados) dos mainframes. O modelo relacional caiu no gosto da comunidade de analistas e programadores ao ponto de tornar-se um padrão de mercado. Na década de 80 e 90, outros SGBDs de mercado ajudaram a propagar o MER (modelo entidade relacionamento) nas plataformas abertas UNIX, dentre estas empresas destacamos duas, a ORACLE e a SYBASE. A ORACLE segue até hoje como uma das principais fornecedoras de tecnologia da informação para o mercado consumidor e a SYBASE pegou o seu produto (SQL-SERVER) e desfez a sua sociedade com a MICROSOFT, cada uma seguiu o seu caminho. A SYBASE foi comprada pela SAP (uma das gigantes do mundo de software) em 2010. A MICROSOFT depois que desfez a sociedade lançou a sua versão MS SQL-SERVER.

Será que o modelo orientado a objeto conseguirá superar a modelo entidade relacionamento?

#### Links:

- O que é um MER: <<https://www.lucidchart.com/pages/pt/o-que-%C3%A9-um-modelo-entidade-relacionamento>>. Acesso em: 10 ago. 2017.
- O Modelo Entidade Relacionamento – UFSC – prof. Renato Fileto: <<http://www.inf.ufsc.br/~r.fileto/Disciplinas/INE5423-2010-1/Aulas/02-MER.pdf>>. Acesso em: 10 ago. 2017.

## Atividades de aprendizagem

- 1.** A técnica de modelagem de dados MER (modelo entidade relacionamento) inicia com uma abordagem da observação do cenário que se pretende informatizar. Neste levantamento de dados, através de uma entrevista do analista de sistemas com os usuários, uma descrição textual narrativa é descrita. Para a identificação das entidades, dos atributos e dos relacionamentos, quais são as características que facilitam este apontamento, respectivamente para entidade, atributo e relacionamento?
- a) Substantivo, Adjetivo e Verbo.
  - b) Substantivo, Pronome e Verbo.
  - c) Substantivo, Adjetivo e Provérbio.
  - d) Pronome, Provérbio e Processo.
  - e) Pronome, Prerrogativa e Verbo.
- 2.** Na técnica de modelagem MER (modelo entidade relacionamento) nós aproximamos o ambiente real dos dados que serão informatizados com a melhor forma de organização e armazenamento dos dados possível. Para que este armazenamento seja o mais correto possível, além de identificar as entidades e os seus atributos, precisamos identificar os relacionamentos entre as entidades. O que estes relacionamentos representam para o MER (modelo entidade relacionamento)?
- a) Representam a interatividade entre entidades e os atributos.
  - b) Representam o estado de classe entre entidades e atributos.
  - c) Representam as regras de negócio existentes entre as entidades.
  - d) Representam a iteração entre atributos e relacionamentos.
  - e) Representam a transição entre entidades e relacionamentos.

# Seção 3

## Cardinalidades e Tipos de Atributos

### Introdução à seção

Caro (a) aluno (a), na Seção 3 vamos abordar as cardinalidades que apresentam os relacionamentos entre entidades e também os diversos tipos de atributos possíveis para uma entidade.

A cardinalidade é uma propriedade que surge com o relacionamento, auxilia no entendimento do relacionamento e na leitura correta do relacionamento. Normalmente é a regra de negócio que determina o valor que será aplicado à cardinalidade. É necessário estabelecer valores mínimos e máximos em cada ponta da cardinalidade.

Os atributos, quando são atrelados às entidades, podem ter características próprias e diferentes umas das outras, inclusive dependendo do tipo do atributo escolhido, pode influenciar no processo de MRN (modelo relacional normalizado).

### 3 Cardinalidades e Tipos de Atributos

#### 3.1 Cardinalidades

##### 3.1.1 Definição

"A razão de cardinalidade para um relacionamento binário especifica o número máximo de instâncias de relacionamento em que uma entidade pode participar" NAVATHE e ELMASRI (2005, p. 46).

Quando um SGBD (sistema gerenciador de banco de dados) é colocado em execução, vários programas de computador são carregados na memória do servidor de banco de dados.

##### 3.1.2 Cardinalidade 1 - 1

A cardinalidade 1 – 1 representa que uma entidade de A se relaciona com 1 somente 1 entidade de B.

"Um homem é casado com uma mulher".

A Figura 3.12 – Cardinalidade 1 – 1 apresenta como o relacionamento "é casado com" entre as entidades Homem e Mulher com cardinalidade

de 1 – 1 fica representado no diagrama entidade relacionamento (DER).

Figura 3.12 | Cardinalidade 1 – 1



Fonte: elaborada pelo autor.

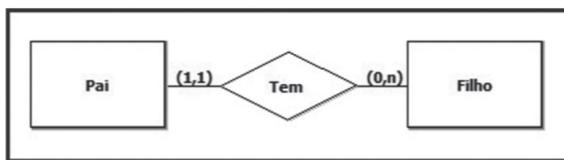
### 3.1.3 Cardinalidade 1 - N

A cardinalidade 1 – N representa que uma entidade de A se relaciona com várias entidades de B.

"Um pai tem vários filhos".

A Figura 3.13 – Cardinalidade 1 – N apresenta como o relacionamento "tem" entre as entidades Pai e Filho com cardinalidade de 1 – N fica representado no diagrama entidade relacionamento (DER).

Figura 3.13 | Cardinalidade 1 - N



Fonte: elaborada pelo autor.

### 3.1.4 Cardinalidade N - N

A cardinalidade N – N representa que várias entidades de A se relacionam com várias entidades de B.

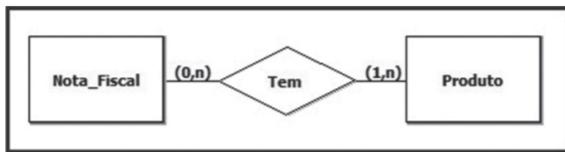
Este tipo de cardinalidade pode ser lido como uma entidade de A se relaciona com várias entidades de B e uma entidade de B se relaciona com várias entidades de A.

Ou seja, um relacionamento de 1 – N nos dois sentidos.

"Uma nota fiscal tem vários produtos, um produto está em várias notas fiscais".

A Figura 3.14 – Cardinalidade N – N apresenta como o relacionamento "tem" entre as entidades Nota\_Fiscal e Produto com cardinalidade de N – N fica representado no diagrama entidade relacionamento (DER).

Figura 3.14 | Cardinalidade N - N



Fonte: elaborada pelo autor.

### 3.1.5 Cardinalidade N - 1

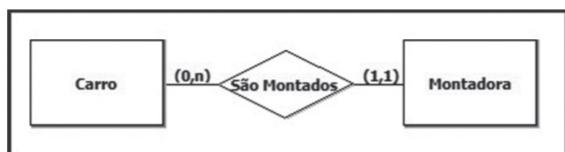
A cardinalidade N – 1 é semelhante à cardinalidade 1 – N, apenas representa que a leitura está sendo feita em sentido contrário à regra de negócio do relacionamento.

Recomenda-se não fazer este tipo de leitura como cardinalidade principal do relacionamento.

"Vários carros são montados por uma montadora".

A Figura 3.15 – Cardinalidade N – 1 apresenta como o relacionamento "são montados" entre as entidades Carro e Montadora com cardinalidade de N – 1 fica representado no diagrama entidade relacionamento (DER).

Figura 3.15 | Cardinalidade N – 1



Fonte: elaborada pelo autor.

### 3.1.6 Sentido da Leitura

A leitura é sempre feita no sentido "1 entidade de A se relaciona com 1 ou N entidades de B" e a volta da leitura do relacionamento deve ser sempre "1 entidade de B se relaciona com 1 entidade de A".

Caso esta leitura reversa apresente um cenário "1 entidade de B se relaciona com N entidades de A", ou o relacionamento em questão é N – N ou você está fazendo uma leitura no sentido errado do relacionamento, podendo até ser necessário a troca do nome do relacionamento (verbo).

### 3.1.7 Cardinalidade mínima

A cardinalidade mínima é representada sempre entre parênteses (0 : 1) ou (0 : N) ou (1 : 1) ou (1 : N). Esta característica determina o mínimo de entidades que devem participar obrigatoriamente do relacionamento.

(0 : 1) no mínimo 0 e no máximo 1.

(0 : N) no mínimo 0 e no máximo N.

(1 : 1) no mínimo 1 e no máximo 1.

(1 : N) no mínimo 1 e no máximo N.



### Questão para reflexão

Já vimos que a cardinalidade de um relacionamento é reflexo direto de uma regra de negócio, o que pode acontecer com um relacionamento se uma regra de negócio for alterada depois que o banco de dados já estiver construído e as tabelas cheias de dados?

#### Links:

- Vale uma pesquisa na enciclopédia livre Wikipédia, disponível em: <[https://pt.wikipedia.org/wiki/Cardinalidade\\_\(modelagem\\_de\\_dados\)](https://pt.wikipedia.org/wiki/Cardinalidade_(modelagem_de_dados))>. Acesso em: 10 ago. 2017.
- Tecnologia de Banco de Dados e Modelagem de Dados Parte 2: <<http://www.devmedia.com.br/tecnologias-de-banco-de-dados-e-modelagem-de-dados-parte-2/1871>>. Acesso em: 10 ago. 2017.

## 3.2 Tipos de Atributos

### 3.2.1 Definição

"Diversos tipos de atributos ocorrem no modelo entidade relacionamento (MER): simples versus composto, univvalorado versus multivvalorado, e armazenado versus derivado" NAVATHE e ELMASRI (2005, p. 39).

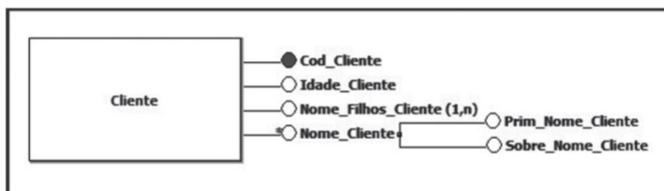
Durante um projeto de banco de dados, a modelagem de dados é fundamental para que a estrutura do banco de dados projetado tenha a melhor organização possível sem perder os relacionamentos entre as entidades identificadas e armazenadas.

A técnica de modelagem entidade relacionamento (MER) apresenta o projeto de banco de dados com 3 etapas distintas onde se é possível

identificar resultados desta modelagem. São os projetos conceitual, projeto lógico e projeto físico do banco de dados.

A Figura 3.16 – Tipos de Atributo no BrModelo apresenta a forma como são representados os atributos dentro da ferramenta BrModelo.

Figura 3.16 | Tipos de Atributo no BrModelo



Fonte: elaborada pelo autor.

Atributos do tipo data, aceitando a máscara que for escolhida (dd/mm/aa, dd/mm/aaaa, dd/mmm/aa, aa/mm/dd).

O projeto lógico é agnóstico de banco de dados, ou seja, não representa nenhum SGBD (sistema gerenciador de banco de dados) de mercado, é totalmente independente.

### 3.2.2 Tipo Determinante

"Os atributos escolhidos devem ser capazes de identificar de forma absoluta as entidades" CHEN (1990, p. 27).

O atributo determinante é aquele cuja entidade não sobrevive sem (o atributo determinante). É o atributo ou conjunto de atributos que permite a identificação individual de cada ocorrência dentro do seu conjunto de ocorrências. Ou seja, é através deste atributo ou conjunto de atributos que cada registro dentro da tabela pode ser identificado e localizado.

O seu conteúdo não pode ser modificado, ou seja, se após inserir o registro, você desejar mudar o conteúdo de um atributo identificador, não poderá fazê-lo. Será necessário remover o registro original e um novo registro ser inserido no lugar.

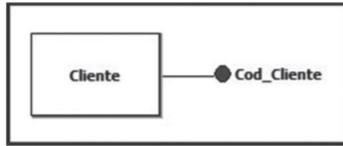
No projeto lógico e físico, este atributo recebe o nome de chave primária ou primary key.

O atributo determinante não pode conter dados duplicados ou mesmo estar vazio ou nulo.

A Figura 3.17 – Atributo Determinante apresenta como o mesmo é

representado dentro da ferramenta BrModelo.

Figura 3.17 | Atributo Determinante



Fonte: elaborada pelo autor

### 3.2.3 Tipo Composto

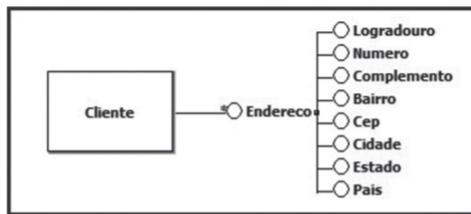
"Os atributos compostos podem ser divididos em subpartes menores, que representam a maioria dos atributos básicos com significados independentemente" NAVATHE e ELMASRI (2005, p. 39).

O atributo composto é aquele que isoladamente tem o seu significado próprio e pode também ser subdividido em outras partes menores, em que cada parte tem o seu significado próprio também.

Ex. um atributo endereço, que isoladamente já tem o seu sentido e este endereço pode ser subdividido em logradouro, número, complemento, bairro, cep, cidade, estado e país.

A Figura 3.18 – Atributo Composto apresenta como o mesmo é representado dentro da ferramenta BrModelo.

Figura 3.18 | Atributo Composto



Fonte: elaborada pelo autor

### 3.2.4 Tipo Multivalorado

"Em alguns casos, um atributo pode ter um conjunto de valores para a mesma entidade" NAVATHE e ELMASRI (2005, p. 40).

O atributo multivalorado é aquele que pode ocorrer várias vezes na mesma entidade, sem uma quantidade específica previamente apontada.

Ex. um atributo optional\_carro onde o cenário poderia ser o seguinte:

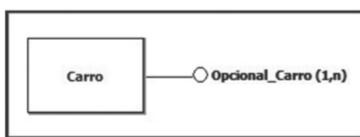
`opcional_carro(1) = "vidro elétrico", opcional_carro(2) = "ar-condicionado", opcinal_carro(3) = "direção elétrica"` e assim por diante.

Desta forma, cada ocorrência da entidade Carro, poderia ter os opcionais que fossem necessários.

No processo de MRN (modelo relacional normalizado), já na Primeira Forma Normal, esta situação seria resolvida com a criação de uma nova entidade.

A Figura 3.19 – Atributo Multivalorado apresenta como o mesmo é representado dentro da ferramenta BrModelo.

Figura 3.19 | Atributo Multivalorado



Fonte: elaborada pelo autor.

### 3.2.5 Tipo Calculado

Segundo NAVATHE e ELMASRI (2005) um atributo calculado ou derivado pode ser o resultado de algum cálculo ou contagem de atributos.

O atributo calculado é um atributo comum como qualquer outro, porém o que muda a sua característica é a origem do dado. Um atributo calculado é dependente de cálculos realizados através de outros atributos da mesma entidade ou de outras entidades.

Em algumas literaturas pode ser denominado como atributo derivado.

Ex. um atributo `Idade_Cliente` é dependente do cálculo da data corrente menos a data de nascimento do cliente.

A vulnerabilidade deste tipo de atributo é a sua consistência, pois “pode” ter uma valor válido e verdadeiro somente no momento da sua gravação e logo depois ter o seu conteúdo defasado.

Um outro exemplo seria um atributo `saldo_conta_corrente` ou `valor_total_nota_fiscal`, vejam que são atributos que dependem de cálculos e que os cálculos podem ser refeitos a qualquer momento.



Para saber mais

Os SGBDs (sistema gerenciador de banco de dados) utilizam os tipos de dados padrão de mercado, desta forma a troca de informações entre eles é mais tranquila. Pesquise a origem, a forma de uso e funcionalidades do padrão XML.



Questão para reflexão

Consulte a enciclopédia livre Wikipedia e veja a definição de tipo de dado. Disponível em: <[https://pt.wikipedia.org/wiki/Tipo\\_de\\_dado](https://pt.wikipedia.org/wiki/Tipo_de_dado)>. Acesso em: 10 ago. 2017.

Portal da Educação – Tipos de atributos. Disponível em: <<https://www.portaleducacao.com.br/conteudo/artigos/informatica/tipos-de-atributos/66721>>. Acesso em: 10 ago. 2017.

#### Links:

- Consulte a enciclopédia livre Wikipedia e veja a definição de tipo de atributo. Disponível em: <[https://pt.wikipedia.org/wiki/Atributo\\_\(programa%C3%A7%C3%A3o\)](https://pt.wikipedia.org/wiki/Atributo_(programa%C3%A7%C3%A3o))>. Acesso em: 10 ago. 2017.
- IBM – Lista de tipos de atributos: Disponível em: <[https://www.ibm.com/support/knowledgecenter/es/SSCKZ6\\_9.1.1/MktOps/FormsEditor/AG\\_Attribute\\_types.html](https://www.ibm.com/support/knowledgecenter/es/SSCKZ6_9.1.1/MktOps/FormsEditor/AG_Attribute_types.html)>. Acesso em: 10 ago. 2017.
- Luis Blog – Entidade: atributos: Disponível em: <<http://www.luis.blog.br/analise-de-entidade-atributos-simples-compostos-multivvalorados.aspx>>. Acesso em: 10 ago. 2017.

## Atividades de aprendizagem

**1.** Um analista de sistema está trabalhando em um projeto de banco de dados e durante o seu levantamento de requisitos junto aos usuários, identificou uma situação em que um determinado dado que precisa ser armazenado no banco de dados possui um valor absoluto sozinho e que pode ser subdividido em outros dados complementares menores.

Segundo a classificação de tipos de dados, podemos classificar esta situação apresentada pelo analista de sistemas como um atributo do tipo:

- a) Multifórmico.
- b) Calculado.

- c) Composto.
- d) Derivado.
- e) Multivalorado.

**2.** Um dos meios que o SGBD (sistema gerenciador de banco de dados) utiliza para um rápido acesso aos dados é o uso de chaves primárias. Além de acelerar uma busca, a chave primária permite identificar individualmente cada uma das ocorrências da entidade. Qual é a classificação recebida pelo atributo chave primária de uma entidade?

- a) Composto.
- b) Determinante.
- c) Indutivo.
- d) Ordenador.
- e) Classificador.

## Fique ligado

Na Unidade 3 apresentamos o conceito de modelo de dados com seus componentes diretos (projeto conceitual, projeto lógico e projeto físico) e a técnica de modelagem de dados MER (modelo entidade relacionamento).

Para detalhar o MER (modelo entidade relacionamento) conceituamos entidade, relacionamento e atributo. Em seguida apresentamos as cardinalidades do relacionamento e os tipos de atributos.

As imagens foram geradas utilizando-se a ferramenta CASE (Computer-Aided Software Engineering) chamada BrModelo, que é um software gratuito na internet e utiliza a representação gráfica proposta por Peter Chen.

Não podemos esquecer que ao utilizar a técnica de modelagem entidade relacionamento, é necessário expressar o resultado e para isto é utilizado o DER (diagrama entidade relacionamento).

## Para concluir o estudo da unidade

Nesta unidade você estudou a técnica de modelagem de dados Modelo Entidade Relacionamento (MER) e viu que a representação da modelagem é feita através de um diagrama entidade relacionamento (DER).

Caracterizamos os componentes deste modelo, que são as Entidades, os Relacionamentos e os Atributos (que podem ser de uma entidade ou de um relacionamento).

Os relacionamentos apresentam a cardinalidade do relacionamento que nada mais é do que a aplicação de uma regra de negócio específica.

Apresentamos também os diversos tipos de atributo: determinante, composto, multivvalorado e calculado.

Para concluir esta unidade, esperamos que você tenha alcançado o seu objetivo inicial de estudo e assim concluir a sua meta de aprendizado.

Aproveite para pesquisar na internet sobre os assuntos aqui abordados. se desejar, aproveite para rever os links, vídeos ou outros livros, consulte as seções “Para Saber Mais” e as nossas “Referências Bibliográficas”.

## Atividades de aprendizagem da unidade

**1.** Considerando que o processo de modelagem de dados precisa utilizar alguma técnica que oriente os analistas na condução da melhor organização possível, podemos destacar as técnicas de Modelo Entidade Relacionamento, Modelo Orientado a Objeto, Modelo Hierárquico, Modelo de Redes e mais recentemente o Modelo Objeto Relacional.

Como a técnica Modelo Entidade Relacionamento vem da década de 70 e consolidado na década de 80 e 90 nas empresas, independentemente da técnica aplicada, a modelagem de dados pura e simples indica que todo projeto deve ter 3 etapas distintas, quais são estas etapas?

- a) Projeto Inicial, Projeto Intermediário e Projeto Final.
- b) Projeto Visionário, Projeto Lógico e Projeto de Tabelas.
- c) Projeto Conceitual, Projeto Lógico e Projeto Físico.
- d) Projeto Visão, Projeto Lógico e Projeto Complementar.
- e) Projeto Independente, Projeto Conceitual e Projeto Físico.

**2.** As ferramentas CASE (*Computer-Aided Software Engineering*) podem realizar as 3 etapas de um projeto de banco de dados de maneira integrada e automatizando a passagem entre cada uma das etapas.

Com esta afirmação já podemos imaginar que existem diversas ferramentas CASE no mercado e elas não são propriamente dos fabricantes de SGBD (sistema gerenciador de banco de dados), com isto, podemos dizer que as ferramentas CASE são multi-banco de dados ou multiplataforma.

O que as ferramentas CASE devem fazer na fase do projeto físico de dados?

- a) Devem se preocupar com as tabelas primárias do banco de dados.
- b) Devem se preocupar com a sintaxe do SGBD escolhido para receber a implantação do projeto.
- c) Devem se preocupar com a sintaxe dos programas de computador que irão acessar o banco de dados.
- d) Devem se preocupar com as entidades embutidas que não foram resolvidas até o momento.
- e) Devem se preocupar se os índices declarados nesta fase batem com os índices das duas fases anteriores.

**3.** Considerando que a técnica de modelagem de dados MER (modelo entidade relacionamento) conceitua e divide o cenário em entidades, relacionamentos e atributos, em que cada uma cada uma tem a sua função e especificidade definida, as entidades podem ser compreendidas como um objeto que possui diversas características e dados que desejamos armazenar.

Quando fazemos a leitura de uma descrição textual narrativa, as entidades são identificadas como substantivos nas frases.

Conceitualmente temos entidades classificadas como:

- a) Integras e Dismorfas.
- b) Simples e Compostas.
- c) Concretas e Abstratas.
- d) Simples e Complexas.
- e) Concretas e Superficiais.

**4.** No processo de modelagem de dados, os dados propriamente ditos são classificados como atributos e fazem parte das entidades ou eventualmente estão ligados aos relacionamentos.

Qual é o tipo de atributo que aceita diversos valores diferentes na mesma ocorrência e que não tem um número limitado de repetições pré-estabelecida?

- a) Atributo Multivalorado.
- b) Atributo Multiplicador.
- c) Atributo Multicontador.
- d) Atributo Multicampo.
- e) Atributo Multiespacial.

**5.** A cardinalidade de um relacionamento é a expressão das regras de negócio do relacionamento entre as entidades. Em muitos dos casos, uma regra depois de estabelecida e as entidades transformadas em tabelas e implementadas no banco de dados, não podem sofrer alterações severas pois os dados armazenados não suportarão estas novas regras.

Qual alternativa apresenta um cenário em que uma mudança deste tipo seria de difícil implementação?

- a) Cardinalidade de 1 - 1 transformado em 1 – N.
- b) Cardinalidade de 1 – 1 transformada em N – N.
- c) Cardinalidade de 1 – N transformada em N - N.
- d) Cardinalidade de 1 – 10 transformada em 1 – N.
- e) Cardinalidade de 1 – N transformado em 1 – 1.

# Referências

Elmasri, Ramez. Shamkant B., Navathe. **Sistemas de banco de dados**. 4. ed. São Paulo: Pearson Education do Brasil Ltda, 2005.

Chen, Peter. **Modelagem de dados**. 1. ed. São Paulo: McGraw-Hill : Makron, 1990.

# Linguagem de acesso a dados

*Leonardo de Marchi Ferrareto*

## Objetivos de aprendizagem

- Entender sobre os conceitos de linguagem de dados.
- Compreender sobre a linguagem de banco.
- Entender os grupos de comandos de linguagem de banco de dados.
- Conhecer os comandos principais da linguagem de acesso a banco de dados padrão ANSI.

## Seção 1 | Linguagem de banco de dados

Serão apresentados os conceitos iniciais da linguagem SQL, como a divisão dos comandos em grupos de acordo com a sua característica, se ele manipulará dados ou fará um controle de acesso, transação ou definição de dados.

## Seção 2 | Comandos SQL

Nesta seção, nos aprofundaremos um pouco mais na linguagem SQL, conhecendo os principais comandos do padrão ANSI, como select, insert, update e delete, e os comandos DDL, como create table, alter table e drop table.



# Introdução à unidade

Nesta unidade, iremos nos familiarizar com os grupos de comandos SQL (*Structured Query Language*) e entender a finalidade de cada um. Todos os conceitos apresentados serão abordados dentro da plataforma de banco de dados SQL Server, porém será respeitado o padrão internacional ANSI (*American National Standards Institute*), adotado pela linguagem SQL. Nossa maior preocupação é demonstrar os principais conceitos da linguagem de banco de dados e os comandos que a compõem.

# Seção 1

## Linguagem de banco de dados

### Introdução à seção

Vamos conhecer e entender os principais conceitos sobre a linguagem de banco de dados, suas características e seus comandos seguindo o padrão ANSI (*American National Standards Institute*); também classificaremos os comandos apresentados em grupos de acordo com a característica de cada um.

### 1.1 Linguagem de banco de dados

Em um sistema de banco de dados, temos duas linguagens utilizadas para o seu uso. Uma delas é a linguagem de definição de dados (DDL), a qual especifica o esquema de banco de dados, e a outra linguagem é a que manipula dados (DML), como atualização, consultas, inserção e exclusão de dados.

Segundo Angelotti (2010, p. 74):



A linguagem SQL (*Structured Query Language*), desenvolvida para banco de dados relacional, teve sua criação na década de 70 e passa por constante aprimoramento e padronização. As duas entidades responsáveis são a ANSI (*American Standards Institute*), os padrões mais recentes do SQL são SQL-99 e o SQL-2003.

Na verdade, essas duas linguagens não são separadas, mas sim uma especificação da linguagem SQL (*Structured Query Language*, ou Linguagem de Consulta Estruturada).

Conforme Angelotti (2010, p. 74):



Nenhum Sistema de Gerenciamento de Banco de dados adota integralmente o padrão ANSI, as empresas desenvolvedoras dos SGBDs customizam suas ferramentas, porém, a cada evolução da mesma, elas absorvem mais ainda o padrão ANSI.

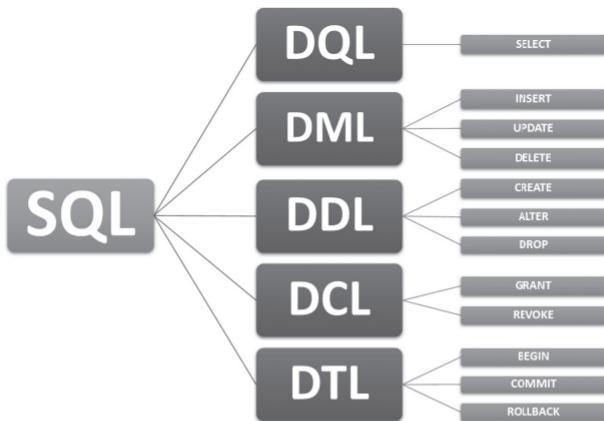
A linguagem de consulta SQL possui vários recursos. Além de consultas em banco de dados, ela pode trabalhar com definição da estrutura de dados, modificação, inserção, exclusão de dados, especificação de segurança e demais controles gerenciais.

O SQL possui uma subdivisão em grupos de comandos por função que exercem. Analisaremos os comandos e as características de cada subconjunto adotando o padrão do banco de dados SQL SERVER.

Os grupos da linguagem SQL são:

- DDL (*Data Definition Language*) – Linguagem de definição de dados.
- DML (*Data Manipulation Language*) – Linguagem de manipulação de dados.
- DQL (*Data Query Language*) – Linguagem de consulta de dados.
- DCL (*Data Transaction Language*) – Linguagem de transação de dados.
- DTL (*Data Control Language*) – Linguagem de controle de dados.

Figura 4.1 | Grupos de Comandos SQL



Fonte: <<http://www.devmedia.com.br/guia/sql/38314>>. Acesso em: 16 ago. 2017.

### 1.1.1 DDL (*Data Definition Language*) – Linguagem de definição de dados

Este grupo permite que o usuário possa definir a estrutura de armazenamento e os elementos associados.

Segundo Silberschatz (2006, p. 9), “O subconjunto de comandos DDL permite a criação de tabelas, restrições de integridade etc.”

Quando utilizamos os comandos DDL para criação de uma tabela, por exemplo, além de criar a tabela no banco de dados, também atualiza o dicionário de dados, onde ficam armazenados os metadados, que são os dados da estrutura de armazenamento.

### 1.1.2 DML (*Data Manipulation Language*) – Linguagem de manipulação de dados

Este grupo é um subconjunto da linguagem SQL que realiza as operações de manipulação de dados, como consulta, inclusão, alteração e exclusão. Essas operações podem ser executadas sob vários registros, em várias tabelas ao mesmo tempo, graças ao controle de transação e compartilhamento de dados.

De acordo com Silberschatz (2006, p. 8), “A linguagem SQL (linguagem de consulta estruturada) não é procedural. Ela tem diversas entradas (podendo ser apenas uma) e retornará uma tabela única”.

Veja, a seguir, um exemplo de uma consulta SQL que encontra nomes de todas as pessoas que residem na cidade de São Paulo:

Figura 4.2 | Comando Select

```
SELECT Pessoa.Nome  
FROM dbo.Pessoa  
WHERE Cidade = 'São Paulo'
```

Fonte: elaborada pelo autor.

Neste exemplo, a consulta traz as linhas em que o campo **CIDADE** são iguais a **SÃO PAULO**, porém o resultado apresentado será somente o campo NOME. Mais especificamente, o resultado dessa consulta é uma tabela com uma única coluna, rotulada **NOME**, e um conjunto de linhas contendo somente o nome das pessoas que moram em **SÃO PAULO**.

### 1.1.3 DQL (*Data Query Language*) – Linguagem de consulta de dados

Este subconjunto possui apenas um comando, o qual é o mais utilizado na linguagem SQL. O comando único deste grupo é o **SELECT**, que permite ao usuário especificar uma query (consulta) para busca de um conjunto de dados desejado. Sua descrição pode ser composta de várias cláusulas e opções, podendo elaborar consultas simples ou até mesmo mais complexas.

A Figura 4.2 é um exemplo de um comando Select.

#### 1.1.4 DTL (*Data Transaction Language*) – Linguagem de transação de dados

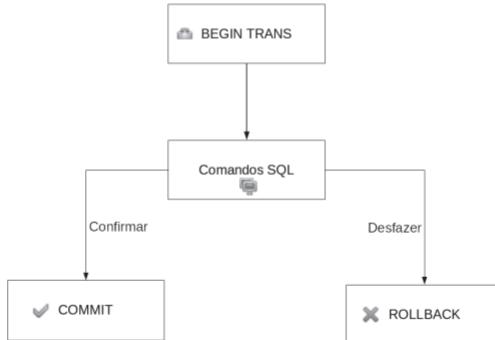
Transação é um conjunto de ações realizadas em um banco de dados. Cada transação é única, sendo assim, elas devem garantir que o seu objetivo seja concluído com segurança, respeitando as restrições em um banco de dados. Quando ela se inicia, o banco de dados deve ser consistente para que ela seja concluída. (SILBERSCHATZ, 2006, p. 14)

Neste subconjunto de comandos, estão presentes os comandos que gerenciam diferentes transações ocorridas em um banco de dados. Basicamente, são três comandos:

- BEGIN TRAN (ou BEGIN TRANSACTION): este marca o início de uma transação no banco de dados que pode ou não ser completada.
- COMMIT: responsável por confirmar a transação aberta e dar como realizada.
- ROLLBACK: praticamente, é uma segurança que nos é proporcionada. Ele faz com que as transações realizadas possam ser desfeitas, e somente funcionará se tiver a transação em aberto.

Os comandos COMMIT e ROLLBACK terminam qualquer transação aberta, liberando o cadeado que é ligado aos dados.

Figura 4.3 | Exemplo de funcionamento do subconjunto DTL



Fonte: elaborada pelo autor.

### 1.1.5 DCL (*Data Control Language*) - Linguagem de controle de dados

Consoante Silberschatz (2006, p. 18):



O Administrador de Banco de Dados, DBA, tem o controle central sobre o sistema de banco de dados, suas responsabilidades são: Definição de esquema do banco; estruturação de armazenamento e definição dos métodos de acesso; organização física e modificação do esquema de dados; rotinas de manutenção e concessão de autorização aos usuários, podendo este autorizar ou não aos usuários acesso para controle de diversas partes do banco de dados bem como a negação aos mesmos também.

O subconjunto DCL (*Data Control Language*, ou Linguagem de Controle de Dados) controla as autorizações que cada usuário pode fazer em um banco de dados. Ele é muito utilizado pelos Administradores de Banco de Dados (DBA – *Database Administrator*). Os comandos que compõem esse subconjunto são:

- GRANT: utilizado para dar permissões aos usuários para realizar operações em um banco de dados. As autorizações podem variar, de acordo com a necessidade, de apenas conexão em um banco a até mesmo exclusão.
- DENY: utilizado para negar a um usuário ou grupo acessos ao banco de dados e também para realizar operações em determinados objetos.
- REVOKE: é utilizado para reverter as ações dos comandos GRANT e DENY.

Com relação a este assunto, Takahashi (2009, p. 159) afirma:



Nos bancos relacionais, o comando GRANT é utilizado para conceder certas permissões para um determinado usuário, essas permissões podem variar de acordo com a necessidade que o mesmo deseja. O comando REVOKE já é o oposto do comando GRANT, ele remove permissões concedidas há um usuário, já o comando DENY é bem parecido com o comando REVOKE porém, ele não remove as permissões concedidas e sim bane o usuário.

Uma maneira prática de se entender o subconjunto DCL quando um usuário necessita de permissão para acessar e realizar operações do tipo select em uma determinada tabela do banco de dados é vendo o exemplo a seguir.

É necessário que o usuário de testes de uma empresa de desenvolvimento tenha acesso de select à tabela de clientes do banco de dados. **Veja como ficaria o comando:**

```
/* Seta o banco para o banco padrão do SQL Server */
```

```
USE master
```

```
GO
```

```
/* Cria o usuário no banco SQL Server */
```

```
CREATE LOGIN [User_Teste] WITH PASSWORD=N'user@123' MUST_CHANGE,  
    DEFAULT_DATABASE=[master],           CHECK_EXPIRATION=ON,CHECK_  
    POLICY=ON;
```

```
/* Acessa o banco o qual o usuário receberá o acesso */
```

```
USE [BancoDeDados]
```

```
GO
```

```
/* Cria o login para o banco de acesso */
```

```
CREATE USER [User_Teste] FOR LOGIN [User_Teste];
```

```
/* Concede a permissão de select para a tabela de Clientes */
```

```
GRANT SELECT ON TabCliente TO User_Teste;
```

Caso seja necessário remover a permissão de select concedida para o usuário, será utilizado o seguinte comando:

```
/* Remove a permissão de select para a tabela de Clientes */
```

```
REVOKE SELECT ON TabCliente TO User_Teste;
```

Neste caso, pode ser ainda que o usuário tenha acesso de select à tabela de clientes se ele estiver em um grupo dentro do banco de dados que tem esse direito. Para confirmar definitivamente o corte ao acesso de select à tabela, utilizaremos o seguinte comando:

```
/* Bani a permissão de select para a tabela de Clientes */
```

```
DENY SELECT ON TabCliente TO User_Test;
```



Para saber mais

Para saber mais a respeito dos grupos de comandos SQL, o site do professor Eduardo Henrique Gomes possui um conteúdo com as principais informações.

Disponível em: <<http://ehgomes.com.br/disciplinas/bdd/sql.php>>. Acesso em: 30 set. 2017.



Questão para reflexão

A linguagem SQL é dividida em subconjuntos de acordo com as operações que queremos efetuar sobre um banco de dados. Quando manipulamos os dados que existem em uma tabela, por exemplo, estamos usando os comandos dos grupos DML (*Data Manipulation Language*), os quais realizam essa operação como *INSERT*, *UPDATE* e *DELETE*. Em relação ao subconjunto DCL, expresse de maneira clara e objetiva qual é a sua finalidade em relação ao banco de dados e à segurança dele.

## Atividades de aprendizagem

**1.** O subconjunto de comandos que permite realizar a criação e alteração de objetos em um banco de dados, sendo eles de armazenamento ou de indexação de dados, é:

- a) Linguagem de manipulação de dados
- b) Linguagem de definição de dados.
- c) Linguagem de consulta de dados
- d) Linguagem de controle de dados
- e) Linguagem de transação de dados

**2.** Para que os dados sejam validados em um banco de dados de forma que possam produzir informação, o banco de dados deve garantir que, para cada operação realizada, esta possa ser concluída, respeitando o seu tomador de posse desta informação. Qual é o mecanismo responsável por essa operação?

- a) Controle de Acesso.
- b) Dados Compartilhados.
- c) Transação.
- d) Armazenamento.
- e) Modificação de dados.

# Seção 2

## Comandos SQL

### Introdução à seção

Nesta seção, seremos apresentados aos comandos SQL respeitando o subconjunto dos comandos pertencentes; conheceremos os operadores lógicos e de comparação; analisaremos novamente os grupos de comandos DML E DDL e veremos quais comandos cada um tem, tudo isso apresentado na plataforma do banco de dados SQL Server.

### 2.1 Comandos DDL

Como vimos na Seção 1 deste livro, os comandos DDL são responsáveis pela definição de armazenamento dos dados no banco de dados.

Segundo Angelotti (2010, p. 74), “A linguagem de definição de dados – DDL – disponibiliza os comandos de criação, exclusão e alteração dos objetos do esquema do banco de dados”.

Abordaremos os comandos que criam, excluem e apagam uma tabela, os quais definem restrições e índices.

#### 2.1.1 CREATE TABLE

O comando CREATE TABLE é o que ocupa um lugar de destaque no grupo de comandos DDL. Ele é responsável pela criação das tabelas em um banco de dados, a qual é feita através de comandos da linguagem SQL. Também é possível definir restrições no ato da criação da tabela.

Segundo Elmasri (2005, p. 150):

O comando **CREATE TABLE** é usado para criar uma tabela em um banco de dados, especificando as suas colunas, atributos e as restrições que cada uma poderá ter. Para cada coluna criada é necessário dar um nome, um tipo de dados que a mesma irá suportar e, se necessário, alguma

**restrição de atributo, por exemplo NOT NULL, que não permite a inserção de valores do tipo nulo.**

A sintaxe do comando CREATE TABLE deve ser bem detalhada, a fim de deixar especificado o que cada coluna armazenará. Vejamos a seguir:

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
    ....
);
```

O parâmetro **table\_name** indica o nome da tabela a ser criada. Os parâmetros **column** deverão receber o nome dos campos (colunas) que irão armazenar os dados referentes à tabela criada. O **datatype** identificará o tipo de dados que a coluna armazenará, por exemplo: varchar, char, date, int, numeric(18,2) etc. A descrição das colunas e seus tipos deverá ser feita dentro do parêntese aberto após o nome da tabela e fechado ao final da definição da última coluna ou restrição.

O exemplo a seguir cria uma tabela chamada “Pessoa”, a qual contém 5 atributos (colunas): PessoalD, Nome, Endereco, Telefone e Cidade.

```
CREATE TABLE Pessoa (
    PessoalD int,
    Nome varchar(255),
    Endereco varchar(255),
    Telefone varchar(255),
    cidade varchar(255 );
```

Após a execução deste comando, será criada uma tabela vazia

com o nome Pessoa com os atributos: PessoalID do tipo int, Nome, Endereço, Telefone e Cidade, todos do tipo varchar, com comprimento máximo de 255 caracteres.

Figura 4.4 | Tabela Pessoa

Pessoa *			
Nome da Coluna	Tipo de Dados	Permitir Nulos	
PessoalID	int	<input checked="" type="checkbox"/>	
Nome	varchar(255)	<input checked="" type="checkbox"/>	
Endereco	varchar(255)	<input checked="" type="checkbox"/>	
Telefone	varchar(255)	<input checked="" type="checkbox"/>	
cidade	varchar(255)	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Fonte: elaborada pelo autor.

### 2.1.2 ALTER TABLE

A instrução ALTER TABLE é utilizada para modificar uma tabela já existente no banco de dados. Essa modificação pode ser uma inclusão, uma modificação ou uma exclusão, tanto para uma coluna quanto para uma restrição existente ou nova restrição.

As definições de uma tabela ou definições de um esquema podem ser alteradas através do comando ALTER. As ações sobre a tabela poderão ser: alteração, adição e exclusão de uma coluna bem com os mesmo três procedimentos em relação as restrições aplicadas as colunas de uma tabela. (ELMASRI, 2005, p. 156)

A sintaxe do comando ALTER pode variar de acordo com a ação que será tomada.

- Na adição de uma coluna a uma tabela existente:

```
ALTER TABLE table_name ADD column_name datatype;
```

- Na exclusão de uma coluna de uma tabela existente:

```
ALTER TABLE table_name DROP COLUMN column_name datatype;
```

- Na alteração de uma coluna de uma tabela existente:

```
ALTER TABLE table_name ALTER COLUMN column_name datatype;
```

Vamos nos basear no exemplo da tabela Pessoa, criada no item 2.1.1 desta seção. É necessário adicionar uma coluna de nome Estado do tipo char(2).

```
ALTER TABLE Pessoa ADD Estado CHAR(2);
```

Esse comando adicionará a coluna mencionada com o seu tipo à tabela já existente, lembrando que o comando ALTER TABLE somente funcionará para tabelas já criadas no banco de dados.

Figura 4.5 | Alteração na tabela Pessoa

Pessoa		
Nome da Coluna	Tipo de Dados	Permitir Nulos
PessoalID	int	<input checked="" type="checkbox"/>
Nome	varchar(255)	<input checked="" type="checkbox"/>
Endereco	varchar(255)	<input checked="" type="checkbox"/>
Telefone	varchar(255)	<input checked="" type="checkbox"/>
cidade	varchar(255)	<input checked="" type="checkbox"/>
Estado	char(2)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Fonte: elaborada pelo autor.

Podemos notar que foi incluído o novo campo solicitado, mantendo os demais da maneira que foram criados.

Digamos que ainda há algumas necessidades de alteração na tabela, então, deve-se modificar a coluna telefone para varchar de tamanho 100, com isso também usaremos o comando ALTER TABLE, porém utilizando a ação ALTER COLUMN.

```
ALTER TABLE Pessoa ALTER COLUMN Telefone VARCHAR(100);
```

Esse comando alterará a tabela já existente Pessoa, modificando o tipo da coluna Telefone para varchar, com tamanho máximo de 100 caracteres.

Figura 4.6 | Modificação do tipo de dados da coluna telefone

Pessoa			
Nome da Coluna	Tipo de Dados	Permitir Nulos	
PessoaID	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Nome	varchar(255)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Endereco	varchar(255)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Telefone	varchar(100)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
cidade	varchar(255)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Estado	char(2)	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Fonte: elaborada pelo autor.

Analisando a tabela contida na Figura 4.7, nota-se que a coluna Preço foi inserida na tabela Pessoa, com isso há uma necessidade de remover essa coluna.

Figura 4.7 | Tabela Pessoa com coluna a ser removida

Pessoa *			
Nome da Coluna	Tipo de Dados	Permitir Nulos	
PessoaID	int	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Nome	varchar(255)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Endereco	varchar(255)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Telefone	varchar(100)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
cidade	varchar(255)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Estado	char(2)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Preço	numeric(18, 2)	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Fonte: elaborada pelo autor.

Para remover a coluna Preço da tabela Pessoa, também é usado o comando ALTER TABLE, com a condição DROP COLUMN.

```
ALTER TABLE Pessoa DROP COLUMN Preço;
```

Não há necessidade de especificar o tipo de dados da coluna, apenas o nome da coluna a ser removida. Na conclusão do comando, teremos a seguinte situação, demonstrada na Figura 4.8.

Figura 4.8 | Tabela Pessoa sem a coluna Preço

Pessoa		Tipo de Dados	Permitir Nulos
Nome da Coluna			
PessoaID		int	<input checked="" type="checkbox"/>
Nome		varchar(255)	<input checked="" type="checkbox"/>
Endereco		varchar(255)	<input checked="" type="checkbox"/>
Telefone		varchar(100)	<input checked="" type="checkbox"/>
cidade		varchar(255)	<input checked="" type="checkbox"/>
Estado		char(2)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Fonte: elaborada pelo autor.

### 2.1.3 DROP TABLE

O comando DROP faz a exclusão de uma tabela no banco de dados. Deve-se tomar um cuidado maior com relação a este comando, pois ele apaga tudo que estiver vinculado às colunas (restrições) da tabela.

De acordo com Elmasri (2005, p. 155), "O comando DROP é utilizado para excluir uma tabela existente de um banco de dados, juntamente com as restrições e os dados vinculados a ela".

Sintaxe do comando DROP.

DROP TABLE *table\_name*;

Veja o exemplo da Figura 4.9.

Figura 4.9 | Tabela Pessoa e Cliente

Cliente *		Pessoa	
Nome da Coluna	Tipo de Dados	Nome da Coluna	Tipo de Dados
ClienteID	int	PessoaID	int
Nome	varchar(255)	Nome	varchar(255)
Endereco	varchar(255)	Endereco	varchar(255)
Telefone	varchar(255)	Telefone	varchar(100)
cidade	varchar(255)	cidade	varchar(255)
Estado	char(2)	Estado	char(2)

Fonte: elaborada pelo autor.

Tem-se a representação de duas tabelas com, praticamente, a mesma finalidade, com isso, devemos excluir a tabela Cliente, a qual será substituída pela tabela Pessoa, lembrando que, ao efetuar a exclusão da tabela Cliente, todos os dados e restrições vinculados a ela serão excluídos também.

O comando usado para a exclusão desta tabela é o seguinte:

```
DROP TABLE Cliente;
```

Ao executar esse comando, a tabela Cliente será removida do banco de dados, permanecendo somente a tabela Pessoa.

#### 2.1.4 PRIMARY KEY – Chave Primária

Chaves primárias são restrições criadas em uma tabela do banco de dados para identificar a linha da tabela. Elas não permitirão que possam ocorrer valores duplicados, ou seja, que a mesma linha seja inserida novamente. Isso só ocorrerá caso o valor da chave primária for alterado.

Conforme Angelotti (2010, p. 78), “A Primary Key é utilizada para identificar a linha da tabela, é a chave primária, ela não pode se repetir e também não aceita valor nulo”.

Suas principais características são:

- Não pode ser nula.
- Identificará a linha.
- Terá um índice definido.
- Poderá ser referenciada em outra tabela, a fim de definir integridade referencial.
- Pode ser simples (possui um único campo) ou composta (possui dois ou mais campos).
  - Se a chave primária é composta, os valores de cada campo podem se repetir, mas não a combinação desses valores.
- Toda chave primária criará um objeto chamado Constraint (restrição).

Essa restrição pode ser criada junto ao comando CREATE TABLE. Caso a tabela já exista e seja necessária a criação da chave primária, deve-se utilizar o comando ALTER TABLE com a condição ADD CONSTRAINT. As sintaxes são assim definidas: sintaxe de criação da chave primária junto ao comando CREATE TABLE e ALTER TABLE e sintaxe de exclusão de uma chave primária.

- Criação com o comando CREATE TABLE.

```
CREATE TABLE table_name (  
    column1 datatype,  
    column2 datatype,  
    column3 datatype,  
    column4 datatype,  
    .....  
    CONSTRAINT key_name PRIMARY KEY (columns1, column2, ...));
```

- Criação com o comando ALTER TABLE.

```
ALTER TABLE table_name ADD CONSTRAINT key_name  
    PRIMARY KEY (column1, column2, ...);
```

- Exclusão com o comando ALTER TABLE.

```
ALTER TABLE table_name DROP CONSTRAINT key_name;
```

Vamos tomar como exemplo a tabela Pessoa, apresentada na Figura 4.4. Nessa tabela é necessário criar uma chave primária no campo PessoalD. Neste caso, como a tabela já existe no banco de dados, devemos utilizar o comando de modificação da tabela ALTER TABLE, adicionando a constraint PRIMEY KEY na coluna PessoalD.

```
ALTER TABLE Pessoa ADD CONSTRAINT PK_PESSOA PRIMARY KEY (PessoalD);
```

O comando acima criará a chave primária na tabela Pessoa, tomando o campo PessoalD como chave primária da tabela Pessoa. Deve-se atentar-se para que o campo PessoalD tem que ser declarado com NOT NULL, pois uma das condições da chave primária é que o campo não pode receber valores nulos.

## 2.1.5 FOREIGN KEY – Chave Estrangeira

Esta restrição é utilizada para manter o relacionamento entre duas tabelas em um banco de dados. Esse relacionamento é imposto através de campos contidos nas duas tabelas.

Conforme afirma Angelotti (2010, p. 78),

**Chave estrangeira garante um dos principais conceitos do banco de dados que é a integridade referencial, onde um campo primário de uma tabela Pai se relaciona com um outro campo de uma tabela Filho.**

Algumas de suas características são:

- O campo de relação da tabela Pai deverá ser a chave primária da tabela Pai.
- As colunas de relação, necessariamente, precisam ser do mesmo tipo.
- Uma chave estrangeira poderá compor uma chave primária.
- Nem toda chave estrangeira será uma chave primária.
- Pode ter o valor nulo.
- Toda chave estrangeira irá criar um objeto chamado Constraint.

A chave estrangeira pode ser criada junto ao comando CREATE TABLE, ou pelo comando ALTER TABLE em tabelas já existentes. Para a exclusão, assim como na chave primária, devemos utilizar o comando ALTER TABLE com a condição DROP CONSTRAINT.

- Sintaxe para criação de chave estrangeira em tabela inexistente, junto ao comando CREATE TABLE.

```
CREATE TABLE table_name{
```

```
column1 datatype,
```

```
column2 datatype,
```

```
column3 datatype,
```

```
.....
```

```
CONSTRAINT key_name FOREIGN KEY column1 REFERENCES table_ref (column));
```

- Sintaxe de criação de chave estrangeira em tabelas já existentes utilizando o comando ALTER TABLE.

```
ALTER TABLE table_name ADD CONSTRAINT key_name
    FOREIGN KEY column REFERENCES table_ref(column);
```

- Sintaxe de exclusão de uma chave estrangeira.

```
ALTER TABLE table_name DROP CONSTRAINT key_name;
```

A Figura 4.10 mostra duas tabelas: a Tabela Estado e a Tabela Cidade. Para que seja criado um relacionamento entre as duas tabelas, devemos levar a chave primária da tabela Estado para se relacionar com a coluna de mesmo tipo de dados da tabela Cidade, pois para cada estado temos várias cidades, e para cada cidade temos somente um estado.

Figura 4.10 | Tabela Estado e Tabela Cidade

TabEstado *		
Nome da Coluna	Tipo de Dados	Permitir Nulos
estID	int	<input type="checkbox"/>
estUF	char(2)	<input type="checkbox"/>
		<input type="checkbox"/>
		<input type="checkbox"/>

TabCidade *		
Nome da Coluna	Tipo de Dados	Permitir Nulos
cidID	int	<input type="checkbox"/>
cidNome	varchar(200)	<input type="checkbox"/>
cid_estID	int	<input type="checkbox"/>
		<input type="checkbox"/>
		<input type="checkbox"/>

Fonte: elaborada pelo autor.

Para que o relacionamento seja feito entre as duas tabelas, devemos relacionar o campo estID da tabela Estado com o campo cid\_estID da tabela Cidade, através do comando SQL:

```
ALTER TABLE TabCidade ADD CONSTRAINT FK_Estado_Cidade
    FOREIGN KEY (cid_estID) REFERENCES TabEstado (estID);
```

Após a execução deste comando, o relacionamento será criado entre as duas tabelas e o diagrama ficará da seguinte maneira.

Figura 4.11 | Chave estrangeira entre as tabelas Estado e Cidade



Fonte: elaborada pelo autor.

### 2.1.6 Not Null – Não Nulo

Define que a coluna de uma tabela não poderá receber valores nulos. Este tipo de restrição poderá ser criado no ato da criação da tabela através do comando CREATE TABLE, ou pelo comando ALTER TABLE, redefinindo a coluna desejada.

“A condição not null, aplicada a uma coluna, não permite que a mesma receba valores nulo, ela obriga a coluna a receber um valor qualquer de acordo com o seu tipo” (ANGELOTTI, 2010, p. 77).

- Sintaxe para criação da restrição NOT NULL utilizando o comando CREATE TABLE.

```
CREATE TABLE table_name(  
    column1 datatype not null,  
    column2 datatype not null,  
    .....);
```

- Sintaxe utilizando o comando ALTER TABLE

```
ALTER TABLE table_name ALTER COLUMN column_name datatype not null;
```

De acordo com a Figura 4.4 – Tabela Pessoa –, será criado uma constraint chave primária no campo PessoalID, para isso, deve-se alterar o campo para não receber nulo sem alteração do seu tipo de dados.

```
ALTER TABLE Pessoa ALTER COLUMN PessoalID int not null;
```

Caso seja necessário reverter a situação do mesmo campo, fazendo

com que ele possa receber valores nulos, devemos novamente usar o comando ALTER TABLE, porém identificando que ele possa receber valores nulos.

```
ALTER TABLE Pessoa ALTER COLUMN PessoalD int null;
```

### 2.1.7 CHECK

Define que a coluna de uma tabela deverá respeitar uma condição para cada valor que for inserido nela.

Segundo Angelotti (2010, p. 78), “A restrição check cria uma regra na coluna, fazendo com que todo novo registro inserido na tabela tenha que respeitar essa condição”.

A restrição CHECK pode ser criada utilizando o comando CREATE TABLE ou pelo comando ALTER TABLE.

- Sintaxe de criação da restrição CHECK no comando CREATE TABLE.

```
CREATE TABLE table_name(  
    column1 datatype,  
    column2 datatype not null,  
    constraint check_name check (column condition)  
    .....);
```

Veja, na Figura 4.8 – Tabela Pessoa –, que o campo Estado poderá receber somente dados iguais a PR, SC e RS, para isso devemos criar uma restrição nesta coluna para que seja respeitada essa condição.

```
ALTER TABLE Pessoa ADD CONSTRAINT CHK_Estado  
    CHECK (Estado in ('PR', 'SC', 'RS'));
```

### 2.1.8 UNIQUE

A restrição UNIQUE garante que os valores contidos em uma coluna, na qual a restrição foi aplicada, não irão se repetir.

Esta restrição é bem parecida com a chave primária, porém ela não tem poder de identificar uma linha da tabela.

Conforme Angelotti (2010, p. 78), “É utilizado para manter dados inseridos com valores únicos. Se essa restrição estiver definida em um campo, nenhum valor irá se repetir”.

A sua criação pode ser feita pelo comando CREATE TABLE ou alterando a tabela pelo comando ALTER TABLE.

```
CREATE TABLE table_name(  
    column1 datatype,  
    column2 datatype,  
    constraint unique_name unique (column)  
    .....);
```

Na tabela “Pessoa”, apresentada na Figura 4.12, foi incluído um campo de nome CPF, o qual deverá receber a restrição UNIQUE, para que não aceite valores duplicados.

Figura 4.12 | Tabela Pessoa modificada

Pessoa *			
Nome da Coluna	Tipo de Dados	Permitir Nulo	Pessoa
PessoalID	int	<input checked="" type="checkbox"/>	
Nome	varchar(255)	<input checked="" type="checkbox"/>	
Endereco	varchar(255)	<input checked="" type="checkbox"/>	
Telefone	varchar(100)	<input checked="" type="checkbox"/>	
cidade	varchar(255)	<input checked="" type="checkbox"/>	
Estado	char(2)	<input checked="" type="checkbox"/>	
CPF	varchar(50)	<input checked="" type="checkbox"/>	

Fonte: elaborada pelo autor.

Para tornar o campo CPF como chave única, UNIQUE, devemos alterar a tabela adicionando uma nova constraint para o campo mencionado.

```
ALTER TABLE Pessoa ADD CONSTRAINT UK_CPF UNIQUE (CPF);
```

## 2.1.9 DEFAULT

A restrição DEFAULT é utilizada para inserir um valor em uma coluna, quando ela não receber nenhum valor.

"A restrição DEFAULT é usada para atribuir um valor padrão para as colunas que o usuário não especificar um valor" (ANGELOTTI, 2010, p. 78).

A criação da restrição DEFAULT pode ser feita na criação da tabela utilizando o comando CREATE TABLE, ou pelo comando ALTER TABLE para tabelas já existentes.

```
CREATE TABLE table_name(  
    column1 datatype DEFAULT(valor),  
    column2 datatype,  
    .....);
```

Usando a Figura 4.12 como exemplo, pede-se que seja criado uma restrição DEFAULT para a coluna Endereco. Quando o usuário não atribuir um valor a essa coluna, ele deverá atribuir a palavra inexistente automaticamente.

```
ALTER TABLE Pessoa ADD CONSTRAINT DF_ENDERECHO DEFAULT ('INEXISTENTE')  
FOR Endereco;
```

## 2.2 Operadores

### 2.2.1 Comparação

São usados para realizar a comparação de valores passados como parâmetros nas colunas de uma tabela. Os operadores não alteram os valores, apenas criam um filtro na apresentação dos dados.

Os operadores de comparação que temos são os seguintes:

Quadro 4.1 | Operadores de Comparação

Operador	Descrição
>	Maior: traz somente valores maiores ao comparado
<	Menor: traz somente valores menores ao comparado
=	Igual: traz somente valores iguais ao comparado
<>	Diferente: traz somente valores diferentes ao comparado
>=	Maior Igual: traz os valores que são maiores e iguais ao comparado
<=	Menor Igual: traz os valores que são menores e iguais ao comparado

Fonte: elaborado pelo autor.

## 2.2.2 Lógicos e de Conjunto

Utilizados como uma regra a ser cumprida, a qual é criada através dos operadores de comparação. Por exemplo, quando é necessário retornar uma faixa de valores compreendida entre dois operadores de comparação.

Os operadores lógicos e de conjunto são:

Quadro 4.1 | Operadores de Comparação

Operador	Descrição
AND	Obriga que as regras impostas sejam todas verdadeiras. Ex.: Idade > 18 <b>AND</b> Idade < 30
OR	Retorna os valores respeitando uma das regras impostas. Ex.: Profissao = 'Analista' <b>OR</b> Profissao = 'DBA'
NOT	É um operador de negação, ele nega o filtro aplicado. Ex.: preco <b>NOT</b> (valor = 10).
IN	Define um conjunto de valores a serem comparados. Ex.: cor <b>IN</b> ('preta','branca').
BETWEEN	Define uma faixa de valores a serem retornados. Ex.: ANO <b>BETWEEN</b> 2000 and 2002.

Fonte: elaborado pelo autor.

## 2.3 Comando SELECT

É um comando do grupo DQL (*Data Query Language*) utilizado para executar consultas em uma ou várias tabelas do banco de dados. O caractere \* representa que o retorno da consulta deverá trazer todas as colunas da tabela; a cláusula FROM determinará a tabela que será consultada.

Consoante Angelotti (2010, p. 91), “Este comando é utilizado para buscar os atributos desejados em uma tabela, retornando os dados contidos nela”.

O comando WHERE pode ser usado junto ao select para utilizar operadores de comparação, lógicos e de conjunto, a fim de refinar mais o resultado a ser apresentado.

Sua sintaxe básica é:

```
SELECT <<column>>,<<column>>, ...  
FROM Table WHERE column operador condition
```

Vamos tomar como exemplo a Figura 4.4 da tabela Pessoa. É necessário retornar os dados das pessoas que moram na cidade de Londrina.

```
SELECT * FROM Pessoa WHERE cidade = 'Londrina'
```

Neste exemplo podemos notar que o comando select utilizou um operador de comparação igual, com isso foi criado um filtro para especializar ainda mais os dados a serem retornados.

Pode-se ainda incluir os operadores lógicos para obedecerem a mais de uma regra.

```
SELECT * FROM Pessoa WHERE cidade = 'Londrina' and nome = 'Jose da Silva'
```

## 2.4 Comando INSERT

É um comando do grupo DML (*Data Manipulation Language*) utilizado para executar inclusões de dados em uma tabela já existente. Podem ser informados valores através de um resultado do comando select.

De acordo com Angelotti (2010, p. 86), “o comando insert é usado quando desejamos inserir dados em uma tabela já existente. Podemos especificar uma linha a ser inserida ou até mesmo escrever várias linhas, um conjunto de linhas a ser inserido”.

Sintaxe do comando INSERT.

```
INSERT INTO table_name (atributo1, atributo2, .....)  
VALUES (valor1, valor2, .....);
```

Tomando a Figura 4.4 – tabela Pessoa –, desejamos inserir dados na tabela de pessoa.

```
INSERT INTO Pessoa (PessoalD, Nome, Endereco, Cidade, Estado)  
VALUES (1, 'Jose da Silva', 'Rua do conhecimento, 100', 'Londrina', 'PR');
```

## 2.5 Comando DELETE

É um comando do grupo DML, o qual é utilizado para excluir linha de dados de uma tabela no banco de dados.

"O comando delete é utilizado para excluir linhas de uma tabela do banco de dados. Ele não exclui dados de uma coluna em específico" (ANGELOTTI, 2010, p. 88).

Sintaxe do comando DELETE.

```
DELETE table_name WHERE condition;
```

Deve-se ter muita atenção quando utilizar esse comando, pois sem o filtro WHERE ele irá apagar todas as linhas de uma tabela.

Usando a tabela de Pessoa da Figura 4.4, deseja-se excluir todos os registros nos quais o campo CPF estiver vazio (nulo).

```
DELETE Pessoa WHERE CPF = null;
```

## 2.6 Comando UPDATE

É utilizado para manipular os dados de uma tabela, mais precisamente, quando é necessário fazer uma atualização nos dados existentes. Este comando irá atualizar a coluna especificada, com isso deverá ser analisada a necessidade ou não de utilização de cláusula WHERE; sem esta condição, o comando irá atualizar todos os dados contidos na coluna mencionada.

Segundo Angelotti (2010, p. 90), "o comando UPDATE altera os

valores de uma determinada coluna especificada na cláusula SET".

Sua sintaxe é composta da seguinte maneira.

```
UPDATE table_name
SET column_name = value
WHERE condition;
```

Para atualizarmos os dados da coluna CPF da tabela Pessoa, da Figura 4.4, devemos usar o comando UPDATE, e na cláusula SET, definirmos a coluna CPF como a coluna que terá os seus dados modificados. Veja um exemplo: deseja-se modificar os valores que sejam nulos na coluna CPF para a palavra 'SEM CADASTRO'.

```
UPDATE Pessoa SET CPF = 'SEM CADASTRO'
WHERE CPF = NULL
```

Como no exemplo foi solicitado para que fossem alterados somente os campos nos quais o valor da coluna CPF fosse nulo, é necessário utilizar a cláusula WHERE para que os dados que tenham que ser alterados correspondam ao que é pedido.



**Para saber mais**

Para se aprofundar um pouco mais neste universo da linguagem de banco de dados SQL, o site W3Schools tem um tutorial completo e detalhado sobre os comandos da linguagem, com exemplos em diversas plataformas de banco de dados.

Disponível em: <<https://www.w3schools.com/sql/default.asp>>. Acesso em: 30 set. 2017.



**Questão para reflexão**

Os bancos de dados são utilizados no armazenamento de diversos tipos de dados, os quais, relacionados entre si, geram informações que são importantes para as empresas e as pessoas que os utilizam. Podemos implementar diversos controles que restringem o acesso a esses dados, concedendo ou não a um usuário, por exemplo, acesso a um grupo

de dados. Em relação ao grupo de comandos DCL (*Data Control Language*), qual é a sua importância em relação ao nível de segurança em um banco de dados?

## Atividades de aprendizagem

- 1.** Baseando-se no quadro a seguir, quantas linhas serão retornadas na seguinte consulta?

*SELECT nome, DataNascimento FROM Pessoa where Sexo = 'M' and between DataNascimento '01/01/1980' and '31/12/1980'*

NOME	DataNascimento	Sexo
José Carlos	20/01/1978	M
Maria do Carmos	24/09/1984	F
João Pedro	16/05/1979	M
Ana Claudia	29/07/2010	F

- a) 1.
- b) 4.
- c) 5.
- d) 6.
- e) 0.

- 2.** Considere a sequência de comandos a seguir:

```
CREATE TABLE pessoa ( ID int not null, nome varchar(100));  
INSERT INTO pessoa ( ID, NOME) values (1,'Leonardo');  
INSERT INTO pessoa ( ID, NOME) values (NULL, 'JOAO');  
SELECT * FROM pessoa;
```

Quantas linhas serão retornadas ao final no comando select?

- a) 10.
- b) 2.
- c) 1.
- d) 3.
- e) 4.

## Fique ligado

Nesta unidade, estudamos:

- Conceitos da linguagem SQL.
- Grupos de comandos SQL.
- Principais comandos do padrão ANSI.

## Para concluir o estudo da unidade

Caro aluno, esta unidade fica por aqui! Através dela podemos observar que as linguagens e acesso nos proporciona uma facilidade na manutenção e na geração da informação para uso em diversas áreas. Procure explorar, cada dia mais, outras possibilidades de geração de informação e automatização dos processos de manutenção de informação, como, o uso de procedimentos armazenados no banco de dados.

## Atividades de aprendizagem da unidade

1. Suponha uma tabela criada com os seguintes comandos:

```
CREATE TABLE Tabela1 (nome varchar(100),
                      Telefone varchar(30),
                      Sexo char(1),
                      Estado char(2));
```

```
INSERT INTO Tabela1 (nome, telefone, sexo, estado)
VALUES ('Leonardo', '4333222020','M', 'PR'),
       ('Pedro', '4333556677','M', 'SC'),
       ('Joao', '4333222222','M', 'PR'),
       ('Marcelo', '4333222222','M', 'SC'),
       ('Eduardo', '4333222222','M', 'SP'),
       ('Carlos', '3245673322,'M', 'PR')
```

Após a execução de todos os comandos, foi feita uma consulta com o seguinte comando:

```
SELECT * FROM Tabela1 WHERE Estado = 'PR' OR Telefone = '4333222222'
```

Quantas linhas serão retornadas para o usuário após a execução do comando?

- a) 1.
- b) 2.
- c) 5.
- d) 6.
- e) 3.

**2.** Em um banco de dados relacional, temos uma importante restrição que pode ser formada por mais de uma coluna. Essa restrição tem um papel fundamental quando for necessário criar um relacionamento entre duas ou mais tabelas. Ela recebe o nome de:

- a) FOREIGN KEY.
- b) UNIQUE.
- c) CHECK.
- d) PRIMARY KEY.
- e) NOT NULL.

**3.** Considere a tabela DebTrab a seguir:

Tabela DebTrab

NroProcesso	Principal	Juros	FGTS	Honorários
111/15	26345,00	3801,75	7933,00	4755,00
777/15	125600,00	18870,00	57966,87	7543,00
333/15	6844,50	1326,67	4233,55	1781,00
555/15	327631,00	65528,20	104863,78	11523,00
444/15	5072,00	1014,40	895,14	700,00

Um analista de banco de dados identificou que há uma necessidade de armazenar um campo nomeado DataPartida com o tipo date. Essa identificação veio através de uma necessidade de armazenamento da data de início dos trabalhos de cada processo, para poder, posteriormente, manter um controle de dias em que o processo está ativo. Qual é o comando correto para adicionar esta nova coluna à tabela existente?

- a) ALTER TABLE DebTrab ALTER COLUMN DataPartida date
- b) ALTER TABLE DebTrab ADD DataPartida date
- c) ALTER TABLE DebTrab INSERT COLUMN DataPartida date
- d) ALTER TABLE DebTrab DROP COLUMN DataPartida date
- e) ALTER TABLE DebTrab REBUILD COLUMN DataPartida date

**4.** Quando desejamos fazer uma alteração nos dados de uma tabela, utilizamos o comando UPDATE. Considere a seguinte expressão escrita na linguagem SQL em um banco de dados relacional:

UPDATE Produtos SET Preco = 15.00 WHERE cor = 'AZUL'

O resultado esperado após a execução do comando é:

- a) Alteração no preço para o valor de 15.00 quando a cor for igual a AZUL.
- b) Exclusão de todos os preços quando a cor for igual a AZUL.
- c) Inserção de um preço para os produtos de cor AZUL.
- d) Alteração no preço, adicionando mais 15.00 quando a cor for AZUL.
- e) Alteração em todos os preços para 15.00, independentemente da cor.

**5.** Através da instrução Insert, podemos adicionar uma ou mais linhas para uma tabela do banco de dados. Sua sintaxe básica é:

```
INSERT INTO <nome_tabela> (coluna1, coluna2, coluna3,.....)  
VALUES (valor1, valor2, valor3, ....)
```

Pode-se também inserir dados em uma tabela a partir do resultado de um comando SELECT.

O comando SQL para inserir todos os registros da tabela Produto na tabela Mercadoria é:

- a) SELECT \* FROM PRODUTO INSERT INTO MERCADORIA
- b) INSERT INTO MERCADORIA SELECT \* FROM PRODUTO
- c) INSERT INTO PRODUTO VALUES MERCADORIA
- d) UPDATE MERCADORIA SET MERCADORIA = PRODUTO
- e) INSERT INTO PRODUTO WHERE PRODUTO = MERCADORIA

# Referências

- ANGELOTTI, Elaini Simoni. **Banco de Dados**. Curitiba: Editora do Livro Técnico, 2010.
- BEAULIEU, Alan. **Aprendendo SQL**. São Paulo: Novatec, 2010.
- BEIGHLEY, Lynn. **Use a Cabeça SQL**. Rio de Janeiro: Alta Books, 2008.
- DATE, C. J. **Introdução a Sistemas de Banco de Dados**. Rio de Janeiro: Campus, 1984.
- ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistema de Banco de Dados**. São Paulo: Pearson, 2005.
- HALVORSEN, Hans-Petter. **Structuted Query Language**. University College os Southeast Norway, 2016.
- HEUSER, C. A. **Projeto de banco de Dados**. Porto Alegre: Sagra Luzzatto, 1998.
- KORTH, H. F.; SILBERSCHATZ, A. **Sistemas de Banco de Dados**. São Paulo, 2006.
- MSDN Blogs. **Guia de Referência: SQL Server Community FAQs Manual**. Disponível em: <<https://blogs.msdn.microsoft.com/sqlforum/2011/06/09/sql-server-community-faqs-ebook-download/>>. Acesso em: 16 ago. 2017.
- SILBERSCHATZ, Abraham; KORTH, Henry F. **Sistema de Banco de Dados**. Rio de Janeiro: Elsevier, 2006.
- TAKAHASHI, Mana. **Guia mangá do banco de dados**. São Paulo: Novatec, 2009.

## Anotações





# unopar

ISBN: 978-85-522-0291-2

A standard linear barcode representing the ISBN number.

9 788552 202912 >