



Modelagem de Dados

Modelagem de Dados

Claudia Werlich

© 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 e de Educação Básica
Mário Ghio Júnior

Conselho Acadêmico
Ana Lucia Jankovic Barduchi
Camila Cardoso Rotella
Danielly Nunes Andrade Noé
Grasiele Aparecida Lourenço
Isabel Cristina Chagas Barbin
Lidiâne Cristina Vivaldini Olo
Thatiane Cristina dos Santos de Carvalho Ribeiro

Revisão Técnica
Isabella Alice Gotti
Vanessa Cadan Scheffer

Editorial
Camila Cardoso Rotella (Diretora)
Lidiâne Cristina Vivaldini Olo (Gerente)
Elmir Carvalho da Silva (Coordenador)
Letícia Bento Pieroni (Coordenadora)
Renata Jéssica Galdino (Coordenadora)

Dados Internacionais de Catalogação na Publicação (CIP)

Werlich, Claudia
W489m Modelagem de dados / Claudia Werlich. – Londrina :
Editora e Distribuidora Educacional S.A., 2018.
216 p.

ISBN 978-85-522-1154-9

1. SGDB (Sistema Gerenciador de Banco de Dados).
2. DER (Diagrama Entidade-Relacionamento). 3. Entidades.
4. Levantamento de requisitos. I. Werlich, Claudia. II. Título.

CDD 005.74

Thamiris Mantovani CRB-8/9491

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

Unidade 1 Fundamentos de Bancos de Dados	7
Seção 1.1 - Introdução a Sistemas Gerenciadores de Bancos de Dados	9
Seção 1.2 - Dados (SGDB)	9
Seção 1.3 - Banco de Dados Relacional	27
Dados como apoio a tomada de decisão	42
Unidade 2 Modelos de banco de dados	57
Seção 2.1 - Modelos de banco de dados	59
Seção 2.2 - Modelagem de dados através do modelo entidade-relacionamento	75
Seção 2.3 - Diagrama de Entidade-Relacionamento (DER)	92
Unidade 3 Abordagem entidade-relacionamento	113
Seção 3.1 - Modelagem de dados através do modelo entidade-relacionamento usando DER	115
Seção 3.2 - Modelagem de dados através do modelo entidade-relacionamento usando UML	129
Seção 3.3 - Ferramentas CASE's de modelagem do diagrama de entidade-relacionamento (DER)	144
Unidade 4 Normalização de dados	161
Seção 4.1 - Normalização de dados na computação	163
Seção 4.2 - Transformação 1FN - 2FN	179
Seção 4.3 - Transformação 3FN - 4FN	194

Palavras do autor

Atualmente, vivemos com nossas informações inseridas em vários bancos de dados. Ao nascer, os dados do bebê são inseridos em diversas bases de dados: na maternidade, no cartório (agora o recém-nascido também possui CPF), na receita federal, entre outros. Diariamente, alimentamos os bancos de dados com nossas informações, ao acessar um site, ao fazer compra em uma loja ou até mesmo ao começar um novo emprego. Os bancos de dados fazem parte de praticamente todos os sistemas computacionais.

Estudar como eles funcionam, criar e modelar os dados corretamente pode impulsionar sua carreira profissional. Conhecer as regras que regem a modelagem de dados traz um diferencial fundamental no desenvolvimento de um software: a qualidade do software.

Você vai estudar nesta disciplina os fundamentos dos bancos de dados e como realizar uma modelagem eficaz. Além disso, conhecerá e realizará os relacionamentos entre entidades de um banco de dados, possibilitando aplicar no projeto de novos softwares e criar soluções arrojadas para os clientes.

Na Unidade 1, você vai conhecer os fundamentos de bancos de dados. Aprenderá os conceitos sobre o Sistema Gerenciador de Banco de Dados (SGBD), um software fundamental para os desenvolvedores de sistemas. Será apresentado aos elementos de um banco de dados relacional e aos softwares de tomada de decisão.

Na Unidade 2, você conhecerá os procedimentos essenciais para modelagem de banco de dados, regras que precisarão ser aplicadas nos modelos a serem implementados. Poderá também observar as várias notações gráficas dos diagramas.

Estratégias de modelagens e a utilização do UML serão abordadas na Unidade 3. As funcionalidades das ferramentas CASEs (do inglês *Computer-Aided Software Engineering*) serão demonstradas, permitindo que a modelagem tenha um resultado mais profissional. Na Unidade 4, estudaremos a normalização dos dados, aplicando regras que serão utilizadas frequentemente em projetos de modelagem de dados.

É de fundamental importância que você desenvolva um plano de estudos e acompanhe as aulas, sua participação é essencial no processo de aprendizagem. As atividades sugeridas possibilitarão uma melhor fixação do conhecimento adquirido neste livro e nas aulas.

Seja bem-vindo ao estudo de Modelagem de Dados!

Fundamentos de Bancos de Dados

Convite ao estudo

Prezado aluno, seja bem-vindo!

A modelagem de dados é uma área muito abrangente. Precisamos ter muitas competências e habilidades para conseguir criar um banco de dados correto e seguro. Para alcançar este objetivo, estudaremos nesta seção o SGBD, entendimento muito importante para podermos auxiliar nossos clientes no desenvolvimento de softwares. Conheceremos os conceitos iniciais sobre os processos de modelagem de um banco de dados e a importância dos dados como apoio a tomada de decisão, algo primordial para as empresas de vários segmentos.

Para iniciarmos nossa jornada de estudos, você foi contratado por uma empresa que desenvolve softwares para vários clientes. Ela cria sistemas atendendo desde microempresas até empresas de grande porte. Sua função é de programador, mas sempre sonhou com a possibilidade de ser um analista de sistemas. Sua chance chegou! Como a demanda por software está muito alta, a empresa está abrindo novas oportunidades. Você participará do atendimento a futuros clientes e terá responsabilidades novas, como fazer visitas aos clientes e criar relatórios, sendo que um desses relatórios exigirá que você analise as características do modelo de banco de dados relacional.

Alguns clientes não sabem ao certo se o investimento em um software mais robusto vale a pena e estão com medo do valor. O time de analistas da empresa do qual você fará parte deverá esclarecer as dúvidas dos clientes e agradá-los com propostas atraentes e eficazes.

Nesta unidade, você terá a oportunidade de estudar conceitos sobre os SGBDs e de conhecer alguns elementos essenciais de um banco de dados. Além disso, deverá reconhecer a necessidade de estabelecer políticas de segurança em um banco de dados. Os conceitos são fundamentais para compreensão e aprendizado sobre a modelagem de banco de dados e, principalmente, para auxiliar no entendimento dos problemas dos clientes, propondo modelos mais seguros e adequados às suas necessidades.

Para iniciarmos, você precisará indicar o SGBD mais adequado para o cliente que deseja um software para uma pequena empresa. Depois, outro desafio será atender a uma empresa de médio porte com necessidades diferenciadas. Finalmente, um cliente de uma empresa de porte grande precisa receber um relatório sobre as características do banco de dados para poder ter uma ideia inicial de investimento e para os analistas da empresa poderem determinar a dimensão do banco de dados e, consequentemente, do porte do software a ser desenvolvido.

Ao final da unidade, você aplicará os conhecimentos, gerando um relatório das características do modelo de banco de dados relacional dos clientes atendidos e de das necessidades dos seus projetos. Ao todo, serão três clientes com necessidades distintas.

Esta fase inicial é muito importante, e você perceberá que precisa se manter atualizado sempre.

Bons estudos!

Seção 1.1

Introdução a Sistemas Gerenciadores de Bancos de Dados (SGDB)

Diálogo aberto

Nesta seção, você estudará conceitos sobre o SGBD, importantes para a compreensão do processo de modelagem pelos quais os dados devem passar, para depois serem armazenados nestes softwares. Quando projetamos um banco de dados, precisamos saber quais serão as aplicações que utilizarão o banco projetado. É necessário um dimensionamento a fim de poder indicar o SGBD mais apropriado para o cliente que deseja o software.

Para darmos início à situação proposta, você recebeu uma oportunidade de crescimento na empresa, ou seja, recebeu uma promoção, e sua nova função é de analista de sistemas júnior. Você terá mais responsabilidades e a chance de obter novos conhecimentos, além de precisar visitar clientes, levantar os requisitos do software e projetar soluções para os clientes da empresa em que você trabalha.

Você deverá analisar as necessidades dos clientes e indicar soluções para cada um, levando em conta os conhecimentos obtidos nesta seção. Através de estudos sobre os SGBDs, qual seria o mais indicado para a empresa de cada cliente? Como poderemos determinar o tipo de SGBD ideal para uma determinada empresa? Esse software pode ser um gratuito ou somente os softwares proprietários podem ser utilizados?

Sabendo que seu primeiro cliente é uma empresa familiar de pequeno porte e que atende em média uns trinta clientes mensais, crie um relatório com informações sobre o cliente e suas necessidades para a utilização do SGBD mais apropriado, juntamente com uma análise das características de cada modelo relacional indicado. O cliente precisará receber informações completas sobre que tipo de investimento deverá fazer. Quanto ele irá gastar? Poderemos utilizar ferramentas freeware? Até onde é seguro para o cliente?

Não pode faltar

Os bancos de dados se tornaram essenciais para o sucesso de grandes corporações. Analisar informações de consumo de clientes pode trazer mais lucratividade para as empresas. Desta forma, atualmente, os bancos de dados se tornaram essenciais no desenvolvimento de softwares. De acordo com Guimarães (2003), ao se projetar um banco de dados, deve-se ter em mente um conjunto de aplicações que deverão utilizar estes dados. Os bancos de dados podem ser simples ou extremamente complexos. O analista de sistemas deverá propor soluções, levando em consideração o volume de informações que deverá ser armazenado.

Podemos afirmar que banco de dados é um conjunto de dados ou informações relacionadas entre si. Conforme Heuser (2009), banco de dados são coleções inter-relacionados de arquivos e que visam auxiliar vários. Exemplificando esse conceito, suponha que você vá fazer a matrícula numa faculdade. Primeiramente, você irá até a secretaria e lá passará os seus dados pessoais, informações acadêmicas, entre muitas outras informações, que deverão ser cadastradas em um banco de dados. Para gerar boletos de pagamento, você não precisará informar seus dados novamente, pois o departamento financeiro já possui suas informações no banco de dados (com as que foram fornecidas no ato da matrícula). São dois sistemas diferentes usando o mesmo banco de dados ou também, podemos dizer, usando a mesma base de dados.

Conforme afirma Date (2003), um banco de dados é uma coletânea de dados duráveis e acessíveis a vários softwares da empresa. O termo persistente faz referência ao armazenamento dos dados que só podem ser apagados do banco de dados através de alguma função específica e, claro, por alguém que tenha a permissão de exclui-los do banco de dados.



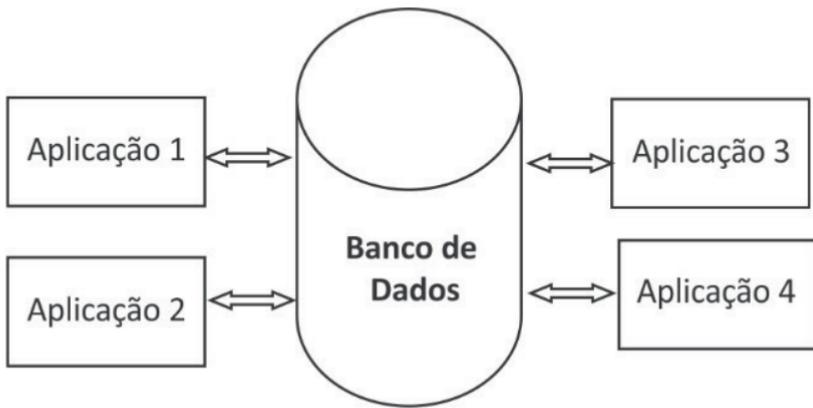
Assimile

Um SGBD é um software cuja finalidade de gerenciar as informações de um banco de dados (também chamada de base de dados) segundo Navathe e Ramez (2005), e que também devem organizar, acessar,

controlar e proteger as informações contidas no banco de dados. O SGBD tem por objetivo facilitar a vida do programador ou analista, deixando livre para pensar na modelagem e não ficar pensando em questões técnicas de armazenamento de dados (sendo esta uma das funções do SGBD).

Quando nos referenciamos ao termo aplicação, estamos mencionando os softwares que estarão se beneficiando dos dados inseridos em um banco de dados. Por exemplo, como citado anteriormente, o sistema do setor financeiro de uma faculdade utiliza as informações arquivadas no banco de dados do sistema de controle acadêmico da secretaria da faculdade, ou seja, podemos dizer que existe um banco de dados único e centralizado para diversas aplicações utilizarem, conforme a Figura 1.1.

Figura 1.1 | Aplicações em um banco de dados

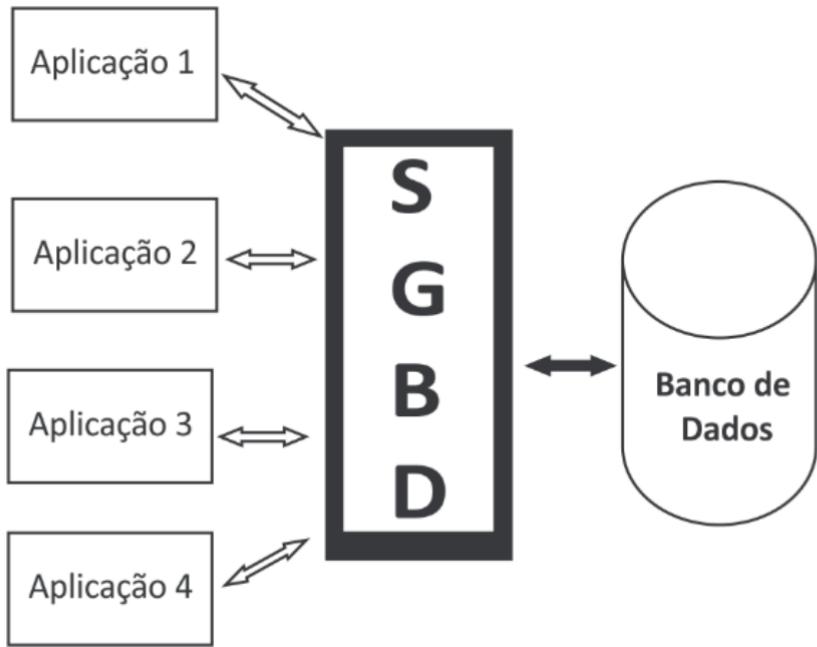


Fonte: elaborada pela autora.

Na Figura 1.1 podemos observar uma imagem central: o banco de dados. O acesso ao banco de dados por diversas aplicações necessita de regras específicas para garantir tanto a segurança quanto a integridade das informações inseridas. São diversos programas que executam essas garantias, conhecidos como SGBD

ou Sistema Gerenciador de Banco de Dados. Observe na Figura 1.2 como é o esquema do SGBD em relação às aplicações e ao banco de dados.

Figura 1.2 | SGBD em um banco de dados



Fonte: elaborada pela autora.

Segundo Korth, Silberschatz e Sudarshan (2012, p. 19), um Sistema Gerenciador de Banco de Dados é constituído por um conjunto de dados associados a um conjunto de programas para acesso a esses dados. O SGBD é projetado para gerenciar grandes volumes de informações, chegando a 1.152.921.504.606.846.976 bytes ou exabyte. O SGBD tem como finalidade a garantia de que as informações que foram inseridas no banco de dados estejam seguras, protegendo de ataques indevidos quanto ao seu acesso ou problemas ocasionados por erros de software ou hardware.

O SGBD pode ser distribuído por diversos computadores, no mesmo local ou até em locais diferentes (espaços, cidades, países). Caso o SGBD esteja em locais físicos diferentes, cada um passa a

receber o nome de nó, e uma operação realizada no banco de dados pode ser executada em um ou em mais nós. Os computadores e seus SGBDs se comunicam através de diversos protocolos de comunicação, de acordo com Navathe e Ramez (2005).



Exemplificando

A integridade é a garantia de que a informação armazenada no banco de dados esteja correta. Precisamos, na medida do possível, prevenir erros de integridade no banco de dados. Por exemplo: é necessário que o usuário informe a sua cidade de nascimento. A cidade que deveria ser informada é São Paulo, mas o usuário digita San Paolo. Caso seja feita uma pesquisa para informar os nascidos em São Paulo, esse usuário ficará fora do resultado. A forma de evitar que isso aconteça é solicitar o estado de nascimento e, a partir dessa informação, apresentar somente as cidades do estado que o usuário informou.

O objetivo geral de um banco de dados é centralizar as informações em determinado computador (servidor ou servidores), permitindo o compartilhamento dos dados entre os mais diversos sistemas, conforme a Figura 1.1. Com muitos usuários acessando os dados, podem ocorrer acessos concomitantes à mesma informação, por exemplo, no banco de dados de uma loja de eletrodomésticos, há disponível a última geladeira de uma determinada marca no estoque. Dois vendedores acessam simultaneamente o registro e vendem a geladeira para seus clientes. Com certeza, um cliente ficaria sem a geladeira, gerando muitos conflitos tanto para o cliente, quanto para a loja. Para este tipo de evento damos o nome de controle de concorrência, uma das finalidades essenciais de um SGBD e que, segundo Korth, Silberschatz e Sudarshan (2012), são técnicas utilizadas para garantir a propriedade de isolamento de transações que estão sendo executadas ao mesmo tempo.

Navathe e Ramez (2005) afirmam que um SGBD possui as funções de permitir aos seus usuários a pesquisa em um banco de dados para recuperar uma determinada informação, alterar e gerar relatórios das informações. Outras funções que podemos destacar do SGBD são a proteção e a recuperação dos dados quando houver problemas de hardware ou software, a segurança a acessos indevidamente

autorizados, a possibilidade de compartilhar dados, a administração da redundância e a restrição de integridade dos componentes do banco. Conforme Guimarães (2003), o conjunto de requisitos de um SGBD recebe o nome de ACID dos termos em inglês **Atomicity**, **Consistency**, **Isolation**, **Durability** ou, respectivamente, **Atomicidade**, **Consistência**, **Isolamento** e **Durabilidade**. O SGBD escolhido pela empresa deve possuir os fatores ACID para garantir que uma transação no banco de dados seja realizada com sucesso. Antes de vermos cada uma das quatro características de um SGBD, precisamos compreender o real significado de uma transação. O SGBD deve garantir várias propriedades durante uma transação.

Segundo Navathe e Ramez (2005), uma transação é um processo ou um determinado programa que pode incluir vários bancos de dados ou somente uma parte do banco de dados, realizando atividades como consultas, alterações e até exclusão de informações da base de dados. Para Korth, Silberschatz e Sudarshan (2012, p. 393), uma transação é uma consequência da efetivação de um programa (ou uma rotina) que acessa e possivelmente atualiza vários itens de dados. A transação é o resultado da execução de um programa de usuário escrito em uma linguagem de manipulação de alto nível ou em uma linguagem de programação, como Java, C# ou SQL, entre outras.



Assimile

O que é um log de transação? O SGBD, para recuperar-se de uma transação com falhas, possui um log para registrar todas as operações realizadas em dados. Funciona como um histórico das modificações. Caso haja erro, através do log, haverá a recuperação dos dados para que eles voltem ao estado inicial.

Agora que sabemos o que é uma transação, vamos entender os requisitos de um SGDB:

- A **atomicidade** garante que nenhuma ou a totalidade das operações da transação sejam realizadas com sucesso. Suponha que estamos aumentando os salários dos funcionários (este aumento é uma alteração em uma tabela e, neste caso é uma transação) e que durante a atualização faltou luz. Somente uma parte dos funcionários

receberá o aumento no salário, caso não haja a verificação de atomicidade. Conforme Korth, Silberschatz e Sudarshan (2012), a ideia por trás da garantia de atomicidade é que o sistema de banco de dados mantenha um registro (em disco) dos antigos valores de quaisquer dados a serem alterados. Caso haja algum problema durante a realização da transação, o SGBD reestabelece os dados antigos, como se nunca tivessem sido modificados.

- A **consistência** preserva as regras impostas no banco de dados. Assim que a transação for finalizada, todos os dados devem estar íntegros. Um exemplo seria a soma de dois valores. Após uma transação, os valores iniciais não podem ser alterados, mas, é claro, se esta for a regra determinada no banco de dados. A consistência é a garantia de manter os dados íntegros durante e com a finalização da transação realizada no banco de dados.
- O **isolamento** é a segurança de que uma transação não interfira no trabalho de outra. Somente após o término de uma transação, ela estará liberada para receber outras. Korth, Silberschatz e Sudarshan (2012) afirma que, mesmo asseguradas as propriedades de atomicidade e consistência para cada transação, a intercalação das operações de várias transações concorrentes pode resultar em inconsistências (erros nos resultados e/ou nos dados). Alterações feitas por transações simultâneas precisam ser isoladas das alterações feitas por qualquer outra transação simultânea.
- A **durabilidade** é a certeza de que após uma transação ser realizada com sucesso, os resultados fiquem gravados no banco de dados, mesmo se algum problema tenha ocorrido, como a queda do sistema. A durabilidade ou persistência (como também é conhecida) em um meio de armazenamento confiável e seguro é um dos requisitos mais importantes de um Sistema Gerenciador de Banco de Dados.

Atualmente, é difícil criar um projeto de banco de dados para uma única aplicação. Por mais que isso ocorra, cabe ao analista de sistema pensar e deixar o banco de dados modelado para futuras mudanças e adaptações. As principais características do uso de um banco de dados, conforme Navathe e Ramez (2005), são as seguintes:

- Natureza autodescritiva do SGBD.

- Isolamento entre os programas, os dados e a abstração dos dados.
- Suporte a diversas visões dos dados inseridos no banco de dados.
- Transações para diversos usuários do banco e a possibilidade de compartilhar os dados da base de dados.

Uma característica essencial de um SGBD é possuir uma ampla gama de possibilidades para definir a estrutura da base de dados e poder aplicar restrições no banco. Os programas de aplicação que irão acessar a base de dados devem ser criados independentemente da estrutura do banco. De acordo com Navathe e Ramez (2005), um SGBD oferece aos usuários uma representação conceitual de dados, omitindo vários detalhes, por exemplo, como os dados realmente são guardados ou como as transações são realizadas no banco de dados. Essa representação de modelo de dados é informalmente conhecida como abstração de dados.

Uma visão (ou *view*) pode ser uma parte de uma base de dados, podendo ser resultantes de pesquisas que retornam parte das informações armazenadas. Um SGBD com suporte a múltiplas visões deve proporcionar facilidades para a definição de diversas visões. O controle de concorrência é o fator primordial para que o compartilhamento de dados e as transações sejam realizados com sucesso para todos aqueles que utilizam o banco de dados. Ao criar visões, podemos criar filtros protegendo certas colunas e tornando o código mais simplificado.



Exemplificando

Uma visão (ou *view*) pode ser considerada como uma tabela virtual ou uma consulta armazenada, permitindo mais do que somente visualizar os dados, implementar algumas restrições. Por exemplo: um professor só tem acesso aos dados de seus alunos matriculados na sua disciplina. O docente, ao fazer uma pesquisa, não precisa ver as notas dos alunos em outras disciplinas e muito menos os dados pessoais ou financeiros dos alunos.

Os Sistemas Gerenciadores de Banco de Dados evoluíram muito nessas últimas três décadas. Muitos deles foram desenvolvidos

por diversas empresas ao longo das últimas décadas, vários ainda estão no mercado e muitos ficaram obsoletos. Segundo Korth, Silberschatz e Sudarshan (2012), os primeiros bancos de dados foram desenvolvidos a partir do sistema de arquivos e eram programados manualmente. As empresas que desenvolvem SGBDs possuem uma fatia de mercado muito valorizada, envolvendo milhões e até bilhões de dólares. Basicamente, os bancos de dados podem ser classificados como:

- Banco de dados usando sistemas hierárquicos e de rede.
- Banco de dados relacional.
- Banco de dados relacional/objetos.
- Banco de dados para Web com XML.
- Banco de dados para nuvem.

As primeiras aplicações de banco de dados mantinham as informações de grandes empresas, como multinacionais, hospitais, universidades e bancos. Navathe e Ramez (2005) afirmam que grande parte dos sistemas que utilizava os bancos de dados desta época, usava computadores de grande porte: os *mainframes*. Esses computadores eram muitos caros e possuíam somente uma *interface* para a linguagem de programação, e cada sistema desenvolvido levava muito tempo para ser programado, pois todas as transações realizadas no banco de dados eram programadas, testadas e depuradas.

Nesta primeira fase dos SGBDs, em meados de 1960 indo até meados de 1980, os bancos de dados eram nos modelos hierárquico e de rede. Os modelos hierárquicos possuíam uma estrutura semelhante à de uma árvore, muito rígida. Caso fosse necessária a adição de uma nova informação (tabela ou campo), o banco de dados em sua totalidade precisaria ser reorganizado ou redefinido. Os bancos de dados em rede assemelhavam-se com uma teia, tecida por uma aranha ou uma rede conectada de informações (registros), porém existia uma restrição de um determinado número de relacionamentos que poderiam ser realizados entre os registros.

Os bancos de dados relacionais começaram a surgir comercialmente a partir de 1980. Navathe e Ramez (2005) afirmam que os bancos de dados relacionais foram originalmente projetados para separar a forma de armazenamento, diferenciando o projeto físico do projeto conceitual do banco de dados. Ofereceram uma flexibilidade maior no desenvolvimento dos sistemas para os mesmos clientes da fase anterior, tornando-se rapidamente uma tendência para todos os desenvolvedores daquela época até os dias atuais e com a possibilidade de ser utilizado em uma infinidade de aplicações nos softwares, por exemplo, para servidores em grandes sistemas, pequenas empresas e em muitos sites na internet.

Conforme pode ser observado em Date (2003), os bancos de dados relacionais/objetos são uma evolução do banco de dados relacional. Com o aparecimento das linguagens orientadas a objetos, a evolução dos bancos de dados foi naturalmente surgindo. Inicialmente, os bancos de dados orientados a objetos surgiram como uma concorrência ao relacional, mas na prática é usado com a possibilidade de utilização de recursos da orientação a objetos como herança, encapsulamento e tipos de dados abstratos, de acordo com Navathe e Ramez (2005).

A internet possibilitou que muitos sistemas ficassem on-line, com sistemas de banco de dados distintos trocando informações entre si. O XML (*eXtensible Markup Language* ou linguagem de marcação que permite a integração de dados) possui um formato que, independentemente da plataforma que o sistema utiliza, permite a comunicação entre sistemas de banco de dados diferentes. Navathe e Ramez (2005) afirmam que a linguagem XML combina os conceitos de modelos empregados nos sistemas de documentos com os conceitos de modelos de banco de dados.

A computação em nuvem está revolucionando a forma de armazenamento, de processamento dos dados e principalmente no quesito de infraestrutura, com pagamentos relacionados ao uso dos recursos disponibilizados. Os SGBDs estão migrando para os serviços em nuvens, diminuindo os custos com equipamentos e softwares, principalmente os e-commerce.



Pesquise mais

A computação em nuvem está revolucionando a criação de aplicativos e serviços de banco de dados. Os SGBDs estão evoluindo e se tornando mais baratos em função desta tecnologia.

BOSCAROLI, C. et al. Uma reflexão sobre banco de dados orientados a objetos. In: CONGRESSO DE TECNOLOGIAS PARA GESTÃO DE DADOS E METADADOS DO CONE SUL, 4., 2006, Ponta Grossa. *Anais...* Ponta Grossa: CONGED, 2006. Disponível em: <<http://conged.deinfo.uepg.br/artigo4.pdf>>. Acesso em: 4 maio 2018.

Um SGBD, de acordo com Heuser (2009), deve ser um software que possui as funções que permita definir, recuperar e alterar as informações contidas em banco de dados. Outra tarefa do sistema é permitir somente acesso autorizado aos dados, então os usuários são registrados no SGBD e possuem seu acesso protegido por senha. Desta forma, os dados são acessíveis somente a uma determinada categoria de usuários, impondo limitação de acessos, conforme afirma Guimarães (2003). O recurso de backup (cópia de segurança) é um fator fundamental que o SGBD deve possuir, permitindo a gravação em um meio de armazenamento e possibilitando a recuperação posterior dos dados.



Refletá

Qual é o melhor modelo de banco de dados? A resposta é simples: aquele que melhor atende às necessidades do cliente. É necessário levar em conta fatores como a infraestrutura, os recursos disponíveis e principalmente as reais necessidades de cada cliente.

Alguns SGBDs antigos estão sendo utilizados no mercado, em sistemas antigos. Isso acontece, pois a empresa pode não ter se atualizado, devido ao custo, porém, na medida em que os sistemas são atualizados, o SGBD mais atualizado começa a ser aplicado, trazendo mais inovação e principalmente segurança. Podemos citar alguns SGBDs que mais se destacam atualmente: Oracle, SQL Server, MySQL, Postgree, entre outras opções existentes no mercado.

Oracle é um SGBD proprietário e sua licença precisa ser adquirida, portanto, não é freeware. É utilizado em médias e grandes empresas e foi projetado para sistemas que requerem alto desempenho e segurança. As versões atuais possuem recursos para computação em nuvem, *big data* (grandes volumes de informação), multiplataformas e também muitas ferramentas de administração e de desenvolvimento de aplicações que servem como *interface*, possibilitando mais facilidades no acesso ao banco de dados.

O **SQL Server** pertence à empresa Microsoft e possui versões gratuitas e pagas, sendo que as pagas são de valores bem inferiores ao seu principal concorrente (visto anteriormente, o Oracle). É utilizado em diversos segmentos de empresas que precisam de um SGBD estável e seguro (e claro, não tão caro!). Um dos problemas deste SGBD era sua plataforma que funcionava somente com o sistema operacional Windows. As novas versões permitem que o SGBD funcione no LINUX e em Container Docker (tecnologia que oferece um conjunto de ferramentas empacotadas e isoladas, como se estivessem em um contêiner). Atualmente, o SQL Server, é um dos SGBDs mais utilizados no mercado, porém está perdendo espaço para outros que têm código aberto.

O **MySQL** é *Open Source* ou código aberto e possui licenças GNU/GPL (Licença Pública Geral – General Public License) permitindo que qualquer usuário edite o seu código fonte de forma que atenda aos requisitos de uma determinada aplicação que está sendo implementada. Atualmente, pertence ao Oracle, cujo objetivo, com a disponibilização *freeware*, é a fomentação do uso desta tecnologia. Sua capacidade de processamento de transações é muito grande e pode ser utilizado por grandes empresas. Um fator a ser considerado é o quesito segurança. Por ser código aberto, o cuidado com falhas de segurança deve ser redobrado. O MySQL possui uma versão bem mais robusta, porém paga, com diversos recursos avançados, como auditoria, *firewalls*, monitoramento e backups avançados.

Postgree é um SGBD muito utilizado por rodar em várias plataformas de desenvolvimento como código aberto (*Open Source*) e de desenvolvimento livre. É usado em sistemas mais robustos, em que a base de dados é muito grande, em empresas

corporativas. Possui funcionalidades de controle de concorrência mais elaboradas e sofisticadas.

Existem muitos outros SGBDs, como o **Access** da Microsoft, que está integrado no pacote do Office, ideal para pequenas aplicações no ambiente Windows. Outro exemplo é o **Firebird**, gratuito e capaz de gerenciar desde bases pequenas até extensos volumes de dados, disponível para vários sistemas operacionais. Há, ainda, o DB2 da IBM, que perdeu muito mercado para os concorrentes, mas ainda é utilizado em sistemas que vão de celulares a *mainframes*.



Pesquise mais

Neste artigo, você pode conhecer mais sobre o termo *Big Data* e entender um pouco mais sobre os novos desafios dos profissionais de informática com o grande volume de dados, gerados e utilizados por grandes corporações. E, que deverão ser gerenciados por SGBDs. SILVEIRA, M.; MARCOLIN, C. B.; FREITAS, H. M. R. O big data e seu uso corporativo: uma revisão de literatura. In: SIMPÓSIO INTERNACIONAL DE GESTÃO DE PROJETOS, INOVAÇÃO E SUSTENTABILIDADE, 4., 2015, São Paulo. **Anais...** São Paulo: SINGEP, 2015. Disponível em: <<https://singep.org.br/4singep/resultado/245.pdf>>. Acesso em: 4 maio 2018.

Os SGBDs possuem diversas peculiaridades que podem conquistar os desenvolvedores, mas todos têm muitas características e funcionalidades em comum, destacando:

- Permite inclusão, exclusão, seleção, ordenação e junção de registros de entidades.
- Possibilita a cópia e a exclusão de entidades.
- Estabelece relações entre as entidades e a criação de chaves.
- Permite a importação ou exportação de dados entre outras bases de dados.
- Possibilita a alteração da estrutura de campos e entidades.
- Permite consultas e relatórios da base de dados.

- Possibilita a criação de usuários com permissão de acesso individualizados.

O profissional que trabalha com os Sistemas Gerenciadores de Banco de Dados precisa acompanhar a constante evolução destes softwares. A atualização deve ser uma constante, possibilitando mais embasamento na utilização do SGBD.

Sem medo de errar

Prezado aluno, agora que você estudou sobre os conceitos de um banco de dados e sobre o SGBD, iremos retomar a nossa situação-problema. Recentemente, você recebeu uma promoção. Em suas novas funções como analista de sistemas júnior, você deverá realizar uma análise do levantamento das necessidades de cada cliente. É justamente nesta hora que precisamos colocar em prática os conhecimentos adquiridos nesta seção sobre o Sistema Gerenciador de Banco de Dados.

O cliente possui uma empresa familiar que atende a cerca de trinta clientes mensalmente, possuindo somente três funcionários. O objetivo principal é controlar as encomendas solicitadas. O cliente precisará receber as seguintes informações detalhadas:

- Qual tipo de investimento deverá fazer na questão de hardware?
- Qual será o valor do investimento em software?
- Poderá utilizar os SGBDs freeware?
- Quão seguro é para o cliente utilizar um SGBD freeware?

O primeiro ponto importante que você deve levar em consideração é o porte da empresa e suas movimentações. Esse ponto impacta diretamente nos custos de software e hardware. Apresente para seu cliente duas opções de orçamento de hardware, lembrando sempre que esse investimento deverá ser feito em uma configuração muito boa. Com relação ao software, apresente

duas opções de orçamento, uma usando um SGBD freeware e outra usando um pago. A escolha do SGBD impactará fortemente no valor final do software. Justifique para ele essa diferença nos valores, mostrando os benefícios e as desvantagens de cada SGBD. Apresente uma terceira opção usando um SGBD na nuvem e não se esqueça dos valores, das vantagens e das desvantagens.

Prepare uma apresentação em slide para seu chefe com suas indicações e justificativas.

Avançando na prática

Análise das necessidades do SGBG para empresa de encomendas

Descrição da situação-problema

Dona Catarina, proprietária de uma empresa de bolos, docinhos e salgados, está com problemas de gerenciamento de seu negócio e quer informatizar sua rotina de trabalho, porém está receosa quanto ao valor de investimento. No último final de ano ela deixou de atender vários clientes, causando prejuízos. A empresa precisa sair do processo manual e informatizar todo o processo da empresa (da encomenda até a entrega e cobrança). Como ela não está mais conseguindo gerenciar as encomendas e precisa de um software, procurou a nossa empresa pedindo ajuda e conselhos e querendo saber no que terá que investir. De uma coisa ela tem certeza: quer que todas as informações fiquem na empresa e de forma bem segura. Você como analista de sistemas deverá auxiliar a Dona Catarina.

Através dos estudos sobre os SGBDs, qual seria o mais indicado para a empresa dessa cliente? Como poderemos determinar o tipo de SGBD ideal para uma pequena, média ou grande base de dados? Verifique os conceitos, caso haja dúvidas.

Após estudar sobre os SGBDs você poderá compreender que, como analistas, devemos tomar cuidado ao indicar um sistema

desnecessário às necessidades do cliente, é responsabilidade de nossa profissão ajudá-lo com ideias claras e corretas.

Resolução da situação-problema

Para a resolução do problema da empresa de Dona Catarina, temos que verificar alguns pontos que devem ser levados em conta para a análise das necessidades deste cliente:

1. Qual a quantidade de clientes da empresa? É uma empresa pequena, portanto, automaticamente descartamos o SGBD Oracle. A versão SQL Server somente é viável se a opção for freeware, mas essa decisão depende diretamente da ferramenta de desenvolvimento.
2. O sistema é todo manual, então haverá a necessidade de investimento em infraestrutura com a compra de equipamentos. Além disso, uma máquina deverá ser configurada para o servidor. Lembre-se de que Dona Catarina deseja que as informações fiquem bem seguras na própria empresa.
3. Como a empresa é de encomendas de bolos e salgados e tudo é feito de forma manual, por que não indicar um sistema integrado à internet? Nesta situação, o MySQL pode perfeitamente ser utilizado.
4. Outra opção seria um serviço em nuvem. Neste caso, teríamos que demonstrar as vantagens e desvantagens deste tipo de serviço: preço *versus* segurança.

Com os conhecimentos adquiridos nesta seção, você deverá escolher e indicar o melhor SGBD. Existem outras opções disponíveis no mercado, mas o fundamental é ter o conhecimento para não indicar algo que inviabilize a realização do software que o cliente deseja. Conquistar o cliente é uma das características desejadas de um analista de sistemas.

Faça valer a pena

1. Leia a sentença abaixo sobre o Sistema Gerenciador de Banco de Dados: Um Sistema Gerenciador de Banco de dados ou simplesmente SGBD é

um _____ de softwares que possuem o objetivo de _____ as informações armazenadas em um banco de dados (também conhecida como base de dados). Devem _____, acessar, _____ e dar _____ às informações contidas no banco de dados.

Com base na sentença, assinale a alternativa que apresenta as palavras que completam a frase corretamente.

- a) banco – gerenciar – processar – controlar – sustentação.
- b) conjunto – agrupar – reformar – inviabilizar – proteção.
- c) sistema – agrupar – processar – marcar – finalização.
- d) conjunto – gerenciar – organizar – controlar – proteção.
- e) banco – capturar – compreender – sistematizar – finalização.

2. Um Sistema Gerenciador de Banco de Dados é um software que possui muitas funcionalidades, entre elas podemos destacar:

- I. Permitem aos seus usuários pesquisas para encontrar uma determinada informação.
- II. Podem gerar relatórios com diversas informações da base de dados.
- III. Protegem e recuperam os dados em caso de falha de software ou de hardware.
- IV. Permitem o compartilhamento de dados e o controle da redundância.

Analise cada item e marque a alternativa correta referente às funcionalidades de um SGBD:

- a) Somente as opções I e IV estão corretas, as outras estão incorretas.
- b) Somente a opção III está incorreta.
- c) Somente as opções I e III estão corretas, as outras estão incorretas.
- d) Todas as opções estão incorretas.
- e) Todas as opções estão corretas.

3. Conforme Guimarães (2003), o conjunto de requisitos de um SGBD recebe o nome de ACID ou, respectivamente, Atomicidade, Consistência, Isolamento e Durabilidade. É fundamental verificar se o SGBD escolhido possui esses quatro fatores. Sobre o conceito de Atomicidade, podemos afirmar que:

- I. É a garantia que todas as informações sejam corretamente cadastradas, inviabilizando dados cadastrados erroneamente, no momento em que o usuário final utiliza o software de aplicação.

II. É a garantia de que todas as transações sejam realizadas com sucesso. Caso contrário, tudo o que foi alterado deverá voltar ao seu estado original antes de a transação ser executada.

III. É a certeza de que, a cada exclusão realizada no banco de dados, um arquivo reserva mantenha o item apagado por tempo indeterminado.

IV. É a certeza de que, a cada inclusão nova no banco de dados, uma nova cópia seja criada como backup provisório da inclusão realizada.

Analise atentamente as afirmativas e marque a opção correta referente ao conceito de Atomicidade de um SGBD.

- a) Somente as alternativas II e IV estão corretas, as demais estão incorretas.
- b) Somente a alternativa II está correta.
- c) Somente as alternativas I e II estão incorretas, as demais estão corretas.
- d) Somente a alternativa III está correta.
- e) Todas as alternativas estão corretas.

Seção 1.2

Banco de Dados Relacional

Diálogo aberto

Seja bem-vindo a mais um conteúdo que norteia a modelagem de dados. O profissional que analisa e projeta o banco de dados precisa sempre estar bem informado e preparado. Antes de conversar com o cliente, você deverá obter o máximo possível de informações sobre o modelo de negócio do cliente, além de ficar a par de como tudo funciona, saber um pouco de legislação e, principalmente, estar sempre bem informado sobre as novas tendências tecnológicas.

O processo de modelagem de banco de dados é algo que deve ser realizado com cautela. A cada diagrama elaborado, a experiência é refletida diretamente no resultado. Como estamos no início deste processo, você precisa ficar atento aos conceitos abordados a partir de agora.

Você deverá atender a um novo cliente, a empresa de aluguel de decoração de festas infantil, Decora Feliz, que nos procurou, pois deseja um software. O desafio é auxiliar no controle de reserva de aluguel dos diversos itens de decoração e brinquedos para festa infantil. Como a empresa é pequena, o pagamento deverá ficar fora deste processo. O objetivo é controlar o local da festa, os clientes e quais equipamentos e decorações serão necessários. Você deverá realizar um levantamento das entidades iniciais que farão parte do banco de dados.

Quais seriam os passos iniciais do processo de modelagem dessa empresa? Por onde começar? Você precisará fazer uma pesquisa sobre a estrutura do banco de dados, mais precisamente, determinar as principais entidades que irão compor o banco de dados a ser modelado.

Observe que algumas entidades que não foram mencionadas pelo nosso cliente aparecerão. Por que isso ocorre? Estudando os conceitos do processo de modelagem, podemos observar que as entidades sempre serão de um número maior do que o problema apresenta. Nessa seção você começará a aprender sobre

o processo de modelagem de dados em banco de dados relacional. O que devemos fazer para criar modelos de dados corretos? Você conhecerá alguns dos elementos mais importantes do banco de dados e ainda começará a compreender mais sobre as entidades e seus atributos.

Como analista de sistemas, você recebeu um novo desafio. Um novo cliente precisa de um software e você, com os conhecimentos adquiridos nesta seção, deverá ajudá-lo no processo de modelagem do banco de dados. Vamos continuar?

Bons estudos!

Não pode faltar

A modelagem de uma base de dados precisa obedecer a certos princípios técnicos. Vale ressaltar que a modelagem é reflexo da experiência de cada analista de sistemas ou programador. Após a realização de vários softwares, o profissional da área de TI consegue visualizar os diagramas mentalmente e vai aperfeiçoando-os com auxílio das técnicas de modelagem de dados. O ponto de vista da modelagem é outro fator a ser considerado, e um exemplo seria o sistema de uma padaria. Basicamente, sabemos como funciona uma padaria, mas precisamos saber que cada empresa possui seus padrões e procedimentos que podem influenciar diretamente no processo de modelagem de um banco de dados.



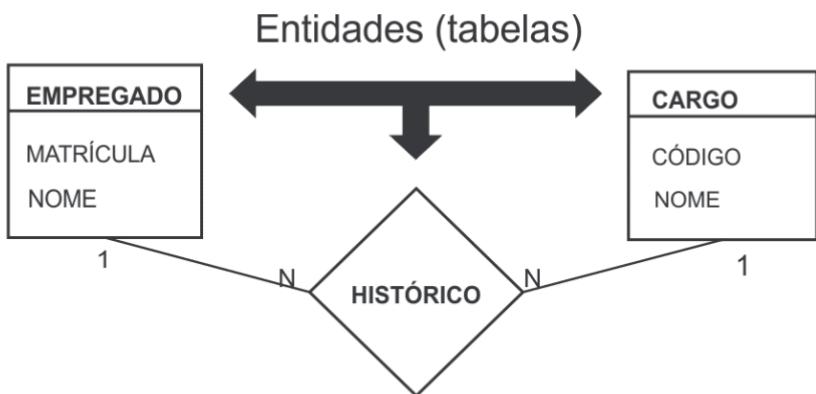
Assimile

A expressão base de dados está relacionada a uma coleção de informações. Podemos afirmar que uma base de dados são os dados armazenados em um banco de dados.

O modelo de dados relacional foi proposto na década de 70 por Peter P. Chen e, desde então, é utilizado para a modelagem de dados, de acordo com Date (2003). Com o passar dos anos, foi aperfeiçoado, porém seu princípio básico ainda é o mesmo. Esse modelo foi baseado na teoria de conjuntos da álgebra relacional. Um banco de dados pode ser representado por um modelo relacional, baseado em uma coleção de relações entre seus integrantes.

Segundo Cougo (1997), um modelo é a representação abstrata e simplificada de um sistema real, gerando um modelo gráfico. Conforme Guimarães (2003), essa forma gráfica (resultante da modelagem relacional) teve grande aceitação por ser um meio de comunicação do projeto conceitual de fácil compreensão por usuários finais de banco de dados. Ao apresentar a modelagem para clientes leigos na área de informática, através dos diagramas, os usuários podem facilmente compreender e ajudar a encontrar problemas na modelagem. Observe a Figura 1.3, em que, mesmo sem saber os conceitos de entidades e relacionamentos, podemos facilmente identificar uma relação entre empregado e cargos e o histórico que é o resultado de vários cargos ocupados por um empregado em alguma empresa. Por ora, não nos concentremos nos diagramas, conforme avançarmos no processo de modelagem, dedicaremos uma seção inteira para esse importante tema.

Figura 1.3 | Exemplo modelo relacional



Fonte: elaborada pela autora.

Date (2003) afirma que o modelo relacional não é algo estático, ele evolui e se expande, assim como a própria matemática. A proposta do modelo relacional baseia-se na ideia de que as informações em uma base de dados podem ser representadas em tabelas cujas linhas apresentam as informações cadastradas. A teoria dos conjuntos se aplica no modelo relacional, pois as operações realizadas nas tabelas são baseadas na álgebra relacional, como seleção, união, junção, subtração, produto cartesiano e projeção.

Tabela 1.1 | Exemplo tabela aluno

Matrícula	Nome	Dt. Nasc.	Curso
1515	Ana Lee	18/06/1989	Direito
1819	Pedro Luz	21/07/2000	Veterinária
2125	Karla Keen	01/02/1999	Direito

Fonte: elaborada pela autora.

Conforme Korth, Silberschatz e Sudarshan (2012), o modelo relacional usa um conjunto de tabelas (ou entidades) para representar tanto os dados como as relações entre eles. Na Tabela 1.1, foram armazenadas quatro informações na tabela Aluno: matrícula, nome, data de nascimento (abreviado: Dt Nasc) e o Curso em que o aluno está matriculado. Neste primeiro momento, analisaremos a disposição das informações: encontram-se em colunas (observe que cada coluna contém uma categoria de informação) e em linhas (observe que em cada linha há informações de um determinado aluno).

Um modelo relacional é composto de muitas tabelas. Date (2003) descreve o modelo relacional como tendo três aspectos básicos:

- Estrutural: os dados inseridos no banco de dados são reconhecidos pelos usuários como tabelas.
- De integridade: as tabelas precisam satisfazer as restrições de integridades (será abordado na próxima unidade).
- Manipulador: são as operações que poderemos realizar com as tabelas, com a intenção de juntar, selecionar, excluir, entre outras operações.

A Tabela 1.1 representa a estrutura de uma tabela com informações sobre alunos. Podemos afirmar, com certeza, que haveria muito mais informações para serem armazenadas. O ideal é criar a estrutura da tabela com o máximo de precisão, mas os SGBDs atuais permitem, com até certa facilidade, adicionar novas informações nas tabelas.



Refilita

Quais seriam as vantagens do modelo relacional de banco de dados? Por que após tantos anos de sua criação ainda é um dos procedimentos mais utilizados no desenvolvimento de sistemas?

Criar um modelo de banco de dados é uma tarefa que, na maioria das vezes, não deve ser feita por uma única pessoa e de forma isolada com a sua própria experiência de vida. Se o projeto for pequeno, é provável que isso ocorra. Porém, suponha que você necessite fazer um banco de dados para uma clínica de estética, mas nunca sequer pisou em uma. Não pode simplesmente fazer algumas consultas na internet e supor o que é necessário para o sistema. O passo inicial, segundo Eslmari e Navathe (2005), é o levantamento e a análise dos requisitos, realizando entrevistas para entender e documentar as necessidades solicitadas pelos usuários. Cougo (1997) afirma que o processo de modelagem implica na existência de um objeto ou em um ambiente a ser observado.

Monteiro (2014) afirma que um requisito é uma condição ou uma capacidade que um software deverá possuir. Requisito é um levantamento e análises das necessidades que o sistema precisa atender. Para atingir esses objetivos serão necessárias reuniões com os clientes envolvidos e analistas de sistemas e que resultará em um documento de especificação com as funcionalidades que o banco de dados precisará possuir.

A fase inicial do projeto de banco de dados, segundo Korth, Silberschatz e Sudarshan (2003), é caracterizar completamente as necessidades de dados dos prováveis usuários do banco de dados. As etapas da modelagem possuem algumas funções que podem ser classificadas como:

- **Concepção:** é um entendimento da necessidade do cliente com relação ao software e é quando serão estabelecidos os objetivos principais da solução desejada.
- **Elicitação:** são as conversas com os usuários do software com o objetivo de colher mais informações sobre os procedimentos realizados e que deverão estar presentes no software.
- **Elaboração:** criação de modelos para a formalização dos requisitos (aqui entra a nossa disciplina). Com o modelo é possível encontrar falhas ou esquecimentos dos clientes ou do próprio analista de sistemas.
- **Negociação:** com o modelo apresentado, os clientes podem querer mais itens, é necessário verificar as viabilidades das sugestões.

A qualidade do banco de dados desenvolvido depende diretamente dos requisitos levantados, mas não para por aí. Ao projetarmos um banco de dados, precisamos nos anteceder às necessidades do cliente. Muitas vezes, criamos e programamos muito além do que foi solicitado, visando a qualidade final do software para este para futuros clientes (caso o sistema seja vendido para um outro cliente). Devemos realizar vários testes como forma de garantir que as relações criadas estejam corretas, antes de implantar o banco de dados. Usamos alguns exemplos (dados) típicos da empresa para determinar se algum ponto crucial da modelagem pode estar com problemas.



Pesquise mais

Saiba um pouco mais sobre os requisitos de software no artigo *Definição de requisitos de software baseada numa arquitetura de modelagem de negócios*.

AZEVEDO JUNIOR, D. P.; CAMPOS R. Definição de requisitos de software baseada numa arquitetura de modelagem de negócios. **Produção**, São Paulo. Associação Brasileira de Engenharia de Produção, vol. 18, n. 1, 2008, p. 26-46. Disponível em: <<http://www.redalyc.org/pdf/3967/396742032003.pdf>>. Acesso em: 4 maio 2018.

Um Sistema Gerenciador de Banco de Dados, de acordo com Korth, Silberschatz e Sudarshan (2012), é constituído por um conjunto de dados associados a um conjunto de programas para acesso a estes mesmos dados, proporcionando um ambiente perfeito para o armazenamento e a recuperação destes dados. Podemos classificar os elementos que compõem um banco de dados em: dados, hardware, software e usuários.

O SGBD foi projetado para trabalhar com um volume muito grande de dados. Esses dados armazenados nos bancos de dados serão compartilhados entre os diversos usuários dos sistemas. É em função deste armazenamento que algumas regras de modelagem deverão ser rigorosamente seguidas (elas serão vistas nas próximas unidades deste livro).

O hardware é um elemento importante, pois determina como e onde os dados serão processados e armazenados. Devemos

orientar corretamente os nossos clientes quanto à necessidade de investimentos nesta área. De nada adianta uma modelagem perfeita se o banco de dados rodar em uma máquina obsoleta. Conforme visto na seção anterior, o armazenamento em nuvem está se destacando atualmente, é uma tendência, mas ainda existem muitas empresas que preferem ter seu próprio servidor e deixar a base de dados em sua propriedade física.

O software em questão como elemento essencial de um banco de dados é sem dúvida o Sistema Gerenciador de Banco de Dados. De acordo com Date (2003), o SGBD é de longe o software mais importante, por isolar o acesso dos dados pelos usuários leigos do acesso ao seu armazenamento no hardware. Existem, ainda, os softwares de aplicação, que são ambientes próprios do SGBD ou linguagens de programação que permitem criar uma *interface* amigável entre o usuário e o SGBD.

Os usuários de banco de dados são os atores (as pessoas) que realizam operações de manipulação na base de dados e podem ser classificados, conforme Korth, Silberschatz e Sudarshan (2012), como:

- Programadores de aplicação: são os programadores do software que podem ou não criar a base de dados, podem simplesmente criar aplicações para acessar a base já existentes ou podem desenvolver o banco de dados e a aplicação que irá utilizar o banco;
- Usuários sofisticados: são usuários treinados para poder manipular a base de dados através de consultas ao banco de dados. Geralmente, são analistas de sistemas especializados em determinado software (ou base de dados);
- Usuários especialistas: são desenvolvedores que projetam aplicações em determinada base de dados específica e complexa, como dados gráficos ou cartográficos;
- Usuários navegantes (comuns ou leigos): são os usuários comuns que fazem uso das informações do banco de dados. Algumas vezes, este tipo de usuário nem sabe que está acessando uma base de dados, procura as informações em um software (de aplicação) e a informação é apresentada na tela (vinda de um banco de dados).

Korth, Silberschatz e Sudarshan (2012) enfatizam que uma das principais razões para usarmos os Sistemas Gerenciadores de Banco de Dados é ter um controle central sobre os dados e sobre os programas que os acessam. É necessário que alguém tenha um controle total sobre o sistema, e este usuário é o Administrador do Banco de Dados ou simplesmente DBA (do inglês *Database Management System*). O Administrador do Banco de Dados possui diversas funções que envolvem a instalação, a configuração e a manutenção do SGBD. O DBA também precisa estabelecer regras de acesso aos dados do servidor, monitorar o banco e realizar manutenções preventivas e corretivas.

Entre algumas funções do DBA, segundo Korth, Silberschatz e Sudarshan (2012), destacam-se:

- Definição do esquema: é o DBA que cria, modifica e atualiza o esquema do banco de dados, executando um conjunto de instruções.
- Concessão de autorização ao acesso aos dados: define quem pode visualizar determinada informação no banco, estabelece juntamente com os administradores da empresa a quais dados cada usuário comum poderá ter acesso.
- Manutenção de rotina: liberar espaços no servidor, realizar backup e monitorar as tarefas no servidor (evitando possíveis gargalos de acessos).

É papel do Administrador de Banco de Dados ficar atento às novas tecnologias que surgem. É uma área que requer muito investimento, e os dados de uma empresa são cruciais. Imagine uma empresa multinacional ficar o dia inteiro parada porque o servidor de banco de dados estragou e não havia um esquema de reserva de equipamentos ou de recuperação de backup. É desse profissional a responsabilidade de garantir que a empresa sofra o mínimo possível em caso de problemas de hardware ou de software.

É a figura do analista de sistema que geralmente desempenha a função de projetista de banco de dados. Korth, Silberschatz e Sudarshan (2012) afirmam que o projetista de banco de dados precisa interagir extensivamente com especialistas do domínio e usuários para realizar essa tarefa, pois ele é quem criará uma representação gráfica dos requisitos do cliente. Neste livro, trataremos do papel do projetista como função do analista de sistemas.



Os usuários dos bancos de dados podem possuir várias formas de acessos sobre os dados e podem ter autorização para: somente leitura, inserir novos dados, atualizar e excluir os dados. Um usuário pode ter um ou mais acessos, sendo que poucas pessoas devem possuir acessos à exclusão de dados.

O modelo relacional usa um conjunto de tabelas para representar os dados como a relação entre eles, cada tabela possui múltiplas colunas e cada uma possui um nome único, como afirmam Korth, Silberschatz e Sudarshan (2012). As tabelas também podem ser denominadas como: entidades, cadastros ou arquivos. Elmasri e Navathe (2005) declaram que as entidades podem representar objetos com existência física, como uma pessoa, um carro, um animal. Podem também representar um objeto com existência conceitual, como um projeto, um departamento, um trabalho acadêmico.

Para Guimarães (2003), uma entidade é um objeto ou indivíduo do mundo real que possui existência própria e cujas características ou propriedades desejamos registrar. Podemos representar graficamente uma tabela usando retângulos ou losangos. A Figura 1.4 mostra o exemplo de três entidades: duas estão em um retângulo e uma num losango. A diferença entre essas formas de representação será detalhada com conceitos que aprenderemos em breve. O mais importante, neste momento, é saber que a entidade "Histórico" irá receber os campos que não podem estar nem na entidade Empregado e nem na entidade Cargo, por exemplo: data de início e data de fim da ocupação de determinado cargo pelo empregado.

Figura 1.4 | Exemplo de entidades



Fonte: elaborada pela autora.

As entidades possuem características próprias e que podem variar na quantidade conforme a necessidade de cada sistema. Suponha que tenhamos uma entidade chamada Animal. Nesta entidade podemos armazenar várias informações básicas de acordo com a necessidade do sistema a ser realizado. Digamos, por exemplo, que se fosse no sentido de venda do animal, na entidade Animal teríamos as seguintes informações: data de nascimento, sexo, raça, porte e valor de venda. Agora, se o objetivo não fosse para a venda do animal e sim para algum serviço a uma clínica veterinária, as informações armazenadas poderiam ser vacinas tomadas, peso, altura, histórico de doenças e muito mais.



Assimile

A nomenclatura para as entidades e os atributos é padronizada pelas empresas de desenvolvimento de software. As entidades devem estar no singular e começar com letra maiúscula. Já o atributo poderá ser abreviado, por exemplo: nm para nome. Algumas empresas utilizam o nome da entidade atrás do nome do atributo (campo): nm_aluno. No SGBD não é aconselhável utilizar espaços em branco, números ou caracteres especiais como nome de entidades e atributos (mas para efeitos didáticos, iremos usá-los neste livro).

As informações que armazenamos em tabelas podem ser agrupadas e são o que chamamos de atributos, campos ou colunas. Na Tabela 1.2 podemos observar que a tabela Aluno possui quatro campos: matrícula, nome, data de nascimento e curso. Cada coluna representa uma categoria de informação, portanto nesta coluna somente será permitido inserir o campo desta categoria, por exemplo, na coluna nome somente o nome do aluno deverá ser digitado. Um conjunto de atributos logicamente dispostos em uma entidade (ou tabela) são conhecidos como registros, linhas ou tuplas. Uma tabela poderá ter milhares de registros. Observe que na tabela possuímos um campo chamado curso, ele está disposto nesta tabela somente como exemplo ilustrativo. Na próxima unidade, veremos que esse campo virá de uma tabela (através do relacionamento), precisaremos tomar certos cuidados para evitar que tabelas se escondam em outras tabelas como forma de campos e, este é um dos segredos da modelagem de dados: descobrir essas tabelas escondidas.

Tabela 1.2 | Exemplo de campo ou atributo

campo ou atributo



Matrícula	Nome	Dt. Nasc.	Curso
1515	Ana Lee	18/06/1989	Direito
1819	Pedro Luz	21/07/2000	Veterinária
2125	Karla Keen	01/02/1999	Direito

Fonte: elaborada pela autora.



Exemplificando

Os atributos ou campos de uma tabela possuem tipos que devem ser cuidadosamente declarados para evitar desperdício de armazenamento. Por exemplo: o campo nome deverá ser do tipo texto, mas se o campo for quantidade de filhos, o correto é declarar com inteiro (ou byte) e não como texto.

O processo de modelagem de dados visa buscar informações para criar o banco de dados. Saber identificar corretamente uma entidade é um dos primeiros passos para obter sucesso no desenvolvimento do software. Um analista de sistemas ou projetista do banco de dados deve sempre estar atento aos detalhes que uma entidade ou um atributo pode revelar: pode se tornar outras entidades.

Sem medo de errar

Nesta seção você poderá conhecer os primeiros passos da modelagem de dados e, com o conhecimento adquirido, vamos relembrar nosso desafio. Uma empresa de aluguéis de decoração para festas deseja um software para o auxílio no gerenciamento de reservas de equipamentos para seus clientes. Quais seriam os passos

iniciais do processo de modelagem dessa empresa? Por onde começar a modelagem? Nesse desafio você deverá realizar um levantamento da estrutura do banco de dados, mais precisamente, determinar as principais entidades que irão compor o banco de dados a ser modelado.

Para a resolução desse problema, será necessário compreender os conceitos sobre entidades e atributos, juntamente com os levantamentos dos requisitos da modelagem. Observe que algumas entidades que não foram mencionadas pelo nosso cliente aparecerão. Por que isso ocorre? Estudando os conceitos do processo de modelagem vistos nesta seção, podemos observar que as entidades sempre serão de um número maior do que o problema apresenta.

Para iniciarmos a solução, observe novamente as necessidades que o cliente apresentou, dando especial atenção às possíveis entidades. Uma dica importante é observar se é um substantivo e, se for, verificar se podemos guardar informações. Neste contexto podemos destacar as seguintes entidades e informações básicas que podem ser armazenadas no banco de dados:

1. Cliente: nome, endereço, telefone, e-mail e CPF.
2. Local da Festa: nome do local, endereço e telefone.
3. Equipamentos: nome, quantidade, preço e tamanho.
4. Decoração: nome (ou tema), quantidade, preço, cor e tamanho.
5. Aluguel: data do aluguel e da festa, horário de entrega, horário de retirada, valor de entrada e saldo a pagar.

Crie uma versão gráfica das entidades utilizando uma das ferramentas do Office e adicione mais atributos. Prepare uma apresentação em slides para mostrar ao seu cliente.

Avançando na prática

Modelagem de dados para uma escola

Descrição da situação-problema

Seu chefe necessita de ajuda no levantamento das entidades envolvidas no próximo banco de dados a ser desenvolvido. Você

precisa fazer um levantamento das entidades para uma escola de ensino fundamental e médio que necessita um banco de dados para um controle acadêmico. Será necessário guardar informações dos alunos, professores, disciplinas, cursos e departamentos. Um problema da escola é a geração dos horários dos alunos, que podem fazer várias disciplinas em diferentes horários.

Quais serão as entidades? Observe que não há muitos detalhes sobre as informações a serem guardadas. Crie uma lista de possíveis atributos para cada entidade, levando em consideração sua vida acadêmica.

Caso seja necessário, verifique novamente os conceitos sobre entidades e atributos.

Resolução da situação-problema

Para começar a resolver a situação problema, primeiramente precisamos apontar as entidades, que são: Aluno, Professor, Disciplina, Curso, Departamento e Horário. Observe que neste caso não faz sentido criar uma entidade chamada escola, visto que o banco de dados é para a própria escola. As entidades (tabelas) e seus campos (atributos) podem ficar da seguinte forma:

- Aluno: matrícula-aluno, nome, endereço, e-mail, telefone, dt-nasc, foto, etc.
- Professor: número-identificação, nome, e-mail, fone, dt-nasc, dt-admissão, etc.
- Curso: sigla-curso e nome-curso.
- Disciplina: código-da-disciplina, nome e carga-horária.
- Departamento: código-do-departamento, nome e sigla.
- Horário: código-horário, horário, sala, sigla-curso e sigla-disciplina.

Observações importantes: note que na entidade Horário, apareceram dois campos que são de outras entidades. Isso ocorre devido ao relacionamento entre as entidades (na próxima unidade esse exemplo ficará mais claro). Outra situação são os nomes dos atributos: primeiro, eles não foram abreviados para que você possa compreender melhor (mas na realidade esses atributos ficam

sempre abreviados) e, segundo, você poderá incluir novos atributos sem problemas.

Faça valer a pena

1. A modelagem de dados relacional baseia-se na ideia de que as informações em uma base de dados são representadas em tabelas (ou entidades). Desta forma, podemos considerar:

- I. Atributos são as linhas de uma tabela.
- II. Todas as linhas de uma tabela também são conhecidas como categoria.
- III. Cada informação cadastrada em um banco de dados é conhecida como tupla.
- IV. Um campo ou atributo é a coluna de uma tabela, conhecida também como categoria.
- V. Um registro é um conjunto lógico de campos que são conhecidos como tuplas.

Analise cada item e marque a alternativa correta que demonstra os conceitos corretos referentes às partes integrantes da tabela (entidade):

- a) Somente as assertivas I e III estão corretas.
- b) Somente as assertivas II e IV estão corretas.
- c) Somente as assertivas III e V estão corretas.
- d) Somente as assertivas II e III estão corretas.
- e) Somente as assertivas IV e V estão corretas.

2. O levantamento dos requisitos é o primeiro item a ser realizado em um novo projeto e, para atingir esse objetivo, são necessárias entrevistas para entender e documentar as necessidades solicitadas pelos usuários, conseguindo desta forma começar a elaborar o desenvolvimento do software desejado.

Analise atentamente as alternativas e marque a correta que demonstra o conceito sobre os requisitos de um projeto de banco de dados.

- a) Um requisito é um único item que, após ser diagnosticado, acaba se transformando em uma tabela associativa.

- b) Um requisito é a descrição detalhada de cada item e componente de cada tabela, estabelecendo os tipos de dados que serão inseridos na base de dados.
- c) Um requisito é uma condição ou capacidade que um software deverá possuir. É um levantamento e uma análise das necessidades às quais o sistema precisa atender.
- d) Um requisito é criação da tupla de uma tabela, definindo os campos que a compõem.
- e) Um requisito é a criação da base de dados no SGBD.

3. O Administrador de Banco de Dados ou simplesmente DBA é um profissional muito valorizado no mercado de trabalho, justificando, desta forma, a sua constante atualização profissional. As atribuições do DBA são fundamentais para as organizações, pois a funcionalidade do banco de dados depende deste profissional.

Considerando as atribuições de um DBA, marque a alternativa correta referente às funções de um Administrador de Banco de Dados.

- a) Realiza somente a instalação do banco de dados.
- b) Projeta a modelagem do banco de dados e não participa da manutenção do banco de dados.
- c) Faz a manutenção do banco de dados, mas não tem permissão para instalá-lo.
- d) Instala, configura, monitora e realiza manutenção preventiva e corretiva do banco de dados.
- e) Realiza a manutenção do banco somente quando o computador trava, afetando os negócios da empresa.

Seção 1.3

Dados como apoio a tomada de decisão

Diálogo aberto

Nesta seção falaremos sobre como os dados podem influenciar na tomada de decisão das empresas. Novos conceitos serão apresentados, abrindo um enorme leque de pesquisas que você poderá realizar, além de conhecer novas tecnologias. Um analista de sistemas precisa reciclar seus conhecimentos e manter-se sempre bem informado sobre as tendências do mercado de TI.

Nas seções anteriores, você estudou os conceitos do SGBD (Sistema Gerenciador de Banco de Dados) e as políticas de segurança de um banco de dados, agora pode distinguir alguns dos principais SGBDs do mercados. Você ajudou duas empresas, uma de porte pequeno, familiar e que precisava de indicação de um SGBD, e a outra de porte médio, cujas entidades você identificou para iniciar o processo de modelagem do banco de dados. Agora, uma empresa de porte grande precisará ser atendida. Este cliente gerencia planos de saúde e deseja estudar o comportamento de seus clientes. Eles possuem um ótimo banco de dados, mas querem um estudo com estatísticas mais apuradas para estabelecer um perfil de comportamento e de possíveis doenças (e, claro, futuros gastos com o plano de saúde). Como você poderá ajudar essa empresa? Como decidir no que investir para obter resultados mais estratégicos? A empresa Saúde e Vida teme também que essas informações preciosas acabem nos concorrentes, então precisa que você a auxilie nesta questão.

Estudando os conceitos de OLTP (Processamento de Transações em Tempo Real) e OLAP (Processamento Analítico On-line), você deverá indicar uma solução para a empresa. Para clientes de grande porte, a necessidade pode ser de bancos de dados mais robustos, portanto, estabeleça uma breve política de segurança para sua base de dados.

Como desafio final, você deverá realizar um relatório para sua empresa. Como foram três clientes de portes diferentes, crie um

relatório sobre as características do modelo de banco de dados relacional e das necessidades de cada projeto realizado, indique o melhor SGBD para cada cliente e justifique sua resposta, de acordo com as respectivas necessidades. Para essa tarefa, retome os estudos desta unidade.

Bons estudos!

Não pode faltar

A finalidade de utilizar um banco de dados está muito além do simples cadastro de informações que depois serão utilizadas por outros softwares. Um banco de dados é uma mina de ouro, e um estudo minucioso do seu conteúdo pode revelar informações preciosas para a empresa, que podem ser utilizadas para tomarem decisões estratégicas. Administradores de grandes corporações precisam tomar decisões baseadas em informações, estatísticas, gráficos de consumo, etc. Um exemplo da aplicação de tomada de decisão são as empresas de planos de saúde. Por que a porcentagem de aumento do plano de saúde é diferente de uma empresa para outra? Simples, são emitidos vários relatórios dos gastos feitos pelos funcionários. Se o plano de saúde foi muito utilizado por eles durante o ano, no ano seguinte a porcentagem de reajuste desta empresa será maior. Os gestores utilizam as informações contidas no banco de dados para decidirem se haverá aumento, a porcentagem que será aplicada e o tipo de doença que está ocasionado tantos gastos com o plano de saúde.

Conforme Date (2003), sistemas de apoio à decisão são sistemas que ajudam na análise de informação do negócio. A meta é auxiliar os gestores a tomarem decisões e a apontarem problemas existentes e até futuros que possam ocorrer. Korth, Silberschatz e Sudarshan (2012, p. 559) afirmam que os sistemas de apoio à decisão visam obter informações de alto nível a partir de informações detalhadas armazenadas nos SGBDs, possibilitando que os administradores tomem decisões sobre determinados problemas. Qual produto estocar? O que vender primeiro? Qual produto deve entrar em promoção para ser vendido primeiro? São perguntas que podem ser respondidas nos sistemas de apoio à decisão.

Um *Data Warehouse* ou Depósito de Dados, segundo Korth, Silberschatz e Sudarshan (2012, p. 560), é um arquivo ou repositório de informações obtidas de várias origens (bancos de dados) e armazenados em um único local e com um esquema unificado, permitindo consultas para o apoio à decisão. Date (2003) afirma que um *Data Warehouse* é um tipo especial de banco de dados, um depósito de dados orientado por assunto, integrado, não volátil e que pode variar com o tempo, utilizado para ajudar na tomada de decisão. Este modelo surgiu pela necessidade de preservar o banco de dados original da empresa (lembrando que um banco de dados sofre alterações diariamente).

Se existem os *Data Warehouse* para armazenar os dados, por que não criar estratégias para analisar esses dados? A mineração de dados ou *Data Mining* é o processo de analisar grandes bancos de dados de forma semiautomática e de responder a perguntas estratégicas em um período de tempo curto. É a descoberta do conhecimento nos bancos de dados, como afirmam Korth, Silberschatz e Sudarshan (2012, p.16). O *Data Mining* refere-se à mineração ou à descoberta de novas informações em função de regras ou padrões em grandes quantidades de dados, segundo Navathe e Ramez (2005, p. 624), e pode ser aplicado em pesquisas científicas ou em empresas com o objetivo de aumentar significativamente a lucratividade. Korth, Silberschatz e Sudarshan (2012, p. 559) descrevem que um *Data Mining* deve ser capaz de explorar grandes quantidades de dados para encontrar padrões, mudanças, anomalias e associações que tenham relevância no objeto de estudo do banco de dados, utilizando-se de estatísticas, inteligência artificial, reconhecimento de padrões e recuperando informações da base de dados.

Um *Data Mining* possui muitas regras para poder obter o conhecimento desejado. Por exemplo, uma grande fabricante de carro deseja saber qual modelo deveria ser indicado para homens de meia idade e com faixa de seis a dez salários mínimos. Uma análise pode ser realizada no banco de dados da empresa, levando em consideração os clientes dos últimos dez anos. O resultado pode ser uma lista de carros mais escolhidos e mais recusados. Em cima de análises complementares, os gestores podem tomar decisões sobre o que e para quem vender determinado produto.



Pesquise mais

Leia mais sobre Mineração de Dados no artigo a seguir:

RAMOS, I.; SANTOS, M. Y. Data Mining no suporte à construção de conhecimento organizacional. CONFERÊNCIA DA ASSOCIAÇÃO PORTUGUESA DE SISTEMAS DE INFORMAÇÃO, 4, 2003, Porto. **Anais...** Porto: CAPSI, 2003. Disponível em: <http://repository.sdu.m.uminho.pt/bitstream/1822/2302/1/CAPSI2003_IMR_MYS.pdf>. Acesso em: 11 jun. 2018.

O termo OLTP quer dizer *On-line Transaction Processing* ou simplesmente Processamento de Transações em Tempo Real e representa as operações realizadas no SGBD, que permitem executar transações na base de dados de forma repetitiva e a nível operacional e administrativo, de acordo com Date (2003). O OLTP permite consultas do dia a dia da empresa, por exemplo, uma consulta personalizada do RH, com o objetivo de mostrar todos os funcionários cuja renda é acima de dez salários mínimos, que não têm filhos e nem mestrado. As transações realizadas no banco de dados utilizam comandos do SQL (*Structured Query Language* ou Linguagem de Consulta Estruturada), como INSERT, UPDATE e DELETE. Estas transações são feitas em tempo real e não armazenam um histórico das consultas realizadas no banco de dados, dificultando o auxílio à tomada de decisões, conforme Korth, Silberschatz e Sudarshan (2012, p. 393) afirmam.

Date (2003) conceitua OLAP (*Online Analytical Processing* ou Processamento Analítico On-line) como o processo interativo de criar, gerenciar, analisar e gerar relatórios sobre os dados de bancos de dados. Os dados coletados são armazenados em uma tabela multidimensional (ou *arrays multidimensionais*) para posterior análise de algoritmos e softwares específicos. Para fazer as análises, os dados são coletados do OLTP e isso acontece de acordo com a necessidade da empresa. Uma comparação entre OLTP e OLAP pode ser vista no Quadro 1.1.

Quadro 1.1 | Quadro Comparativo entre OLTP e OLAP

OLTP	OLAP
Operações de Rotina	Operações Analíticas
Desempenho Baixo em Consultas	Alto Desempenho em Consultas
Sem Histórico das Consultas	Permite Histórico das Consultas
Usado a Nível Operacional	Usado a Nível de Administradores
Baixo Volume de Dados	Envolve Altos Volumes de Dados
Armazenamento Convencional dos Dados	Armazenamento em Data Warehouse
Dados Voláteis	Dados Históricos e Não Voláteis
Permissão de Leitura, Inserção	Permissão de Leitura
Permissão de Modificação e Exclusão	Não a Permissão de Inserção e Exclusão
Utilizado a Todo Instante	Baixa Frequência de Uso e Pesquisa

Fonte: adaptado de Korth, Silberschatz e Sudarshan (2012).



Assimile

O termo *Business Intelligence* (Inteligência de Negócios ou BI) é o processo de coleta, análise, monitoria e compartilhamento de informações para a gestão de negócios. Esse conceito pode se confundir com o de *Data Mining*. O BI analisa dados brutos operacionais para encontrar informação útil e auxiliar a tomada de decisão. Já o *Data Mining* utiliza ferramentas como agrupamentos, hipóteses, regras e árvores de decisão atuando em nível mais estratégico e fornecendo à empresa conhecimento útil acerca do ambiente para decisões a longo prazo, segundo Korth, Silberschatz e Sudarshan (2012, p. 563).

Conforme afirmam Korth, Silberschatz e Sudarshan (2012, p. 120), "a maioria dos fornecedores de Sistemas Gerenciadores de Banco de Dados fornece ferramentas de OLAP como parte do SGBD ou como adicionais". Existem empresas que desenvolvem softwares específicos para tomada de decisão e auxílio na gestão de informação da base de dados. Algumas companhias que desenvolvem OLAP são IBM, Microsoft, ORACLE, entre outras.

Uma pergunta que podemos fazer é: quando utilizar o OLTP ou o OLAP? É possível exemplificar com a seguinte situação: uma secretaria faz a matrícula de um aluno no sistema de uma determinada faculdade, inserindo suas informações, então, neste caso, é usado o OLTP. Se o aluno se matricular em um curso, arrependendo-se e quiser trocar, a secretaria poderá modificar isso no sistema. Neste caso também é usado o OLTP, por se tratar de uma

consulta de alteração. No final no bimestre, o diretor desta mesma faculdade precisa de um relatório com o perfil dos alunos dos últimos três anos, no qual será analisado: idade, curso, semestres cancelados e situação financeira. Este é um exemplo de OLAP, trazendo informações que servirão para oferta de novos cursos para o perfil de aluno desejado.



Assimile

Ad-hoc é uma consulta (ou transação) no banco de dados realizada pelo próprio usuário com parâmetros criados pela necessidade específica do consultante. A geração de consultas de acordo com as necessidades do negócio do cliente pode levar a resultados e descobertas casuais.

O grande número de dados e a modelagem de um banco de dados podem levar a redundâncias, ocasionando futuros problemas. O controle da redundância (repetição) de um banco de dados é uma tarefa que deve ser realizada a partir da modelagem do banco de dados. Korth, Silberschatz e Sudarshan (2012, p. 158) afirmam que uma análise dos atributos das entidades deve ser realizada para evitar redundâncias, por exemplo o atributo cidade em entidades. Imagine que num banco de dados de uma universidade precisamos armazenar a cidade de nascimento do aluno. O atributo cidade precisa ser armazenado em outras entidades, como funcionários e fornecedores. É a mesma informação sendo armazenada em várias entidades diferentes, levando muitas vezes à inconsistência do banco de dados.



Exemplificando

A inconsistência em um banco de dados ocorre, principalmente, quando há redundâncias, por exemplo: o nome da cidade de São Paulo é digitado em três entidades diferentes. Na primeira está São Paulo (sem acento), na segunda está SP (abreviado) e na terceira está São Paul (com erro de digitação). Os dados são da mesma cidade, porém de formas bem diferentes. Esse tipo de situação pode ser contornado ao criarmos uma entidade chamada cidade.

Segundo Navathe e Ramez (2005, p. 12), um SGBD deve permitir a redundância controlada. Em algumas situações específicas, até é permitida a repetição de algum atributo em uma entidade, mas isso deve ser feito de forma bem consciente pelo analista ou projetista do banco de dados, pois pode aumentar significativamente o espaço de armazenamento do banco de dados. Por causa do backup, existem várias cópias físicas do banco de dados, gerando redundância, cujo tipo de redundância, também é uma redundância controlada, visto que, as cópias são necessárias para manter a segurança do banco de dados. Em caso de falhas, a cópia backup do banco de dados é realizada.

Um SGBD deve prever facilidades para a recuperação após falhas, de acordo com Navathe e Ramez (2005, p. 13), porém, a empresa precisa se prevenir estabelecendo uma política de backup. Essa política é um conjunto de regras estabelecido pelo DBA (Administrador do Banco de Dados) junto com os gestores da empresa, que determinam as respostas dos seguintes itens:

- Responsabilidades: quem fará o backup? Quem terá acesso ao backup?
- Meios: de que forma será realizado o backup? Qual mídia ou nuvem usar? Qual software? Qual hardware?
- Período: qual o intervalo dos backups? Diariamente, semanalmente, mensalmente?
- Retenção: quanto tempo o backup deve ficar armazenado na mesma mídia?

Reflita

Um banco de dados é o bem mais precioso da empresa? Como uma companhia pode ter prejuízos por causa do banco de dados? Ela poderia ir à falência pela perda ou invasão de seu banco de dados?

- Armazenamento: onde serão armazenados os backups? Quais locais seguros deverão ser indicados?



Reflita

Um banco de dados é o bem mais precioso da empresa? Como uma companhia pode ter prejuízos por causa do banco de dados? Ela poderia ir à falência pela perda ou invasão de seu banco de dados?

Segundo Date (2003), o bem mais valioso de uma empresa é a informação. Um SGBD permite que várias formas de segurança possam ser implementadas para proteger o banco de dados e preservar o seu conteúdo. Quando ocorrem falhas de segurança, nem sempre é um ato proposital. Por exemplo, um empregado do RH, de forma equivocada, pode apagar as informações contidas nas entidades dos empregados. Por que isso ocorreu? Aconteceu pelo fato de o funcionário ter permissão para apagar (neste caso, o comando DELETE) dados na entidade dos funcionários.

É fundamental estabelecer políticas de permissões e acessos às tabelas. Korth, Silberschatz e Sudarshan (2012) declararam que o primeiro ponto da segurança de um banco de dados é reforçar a autenticação do usuário e garantir que tenham permissão para acessar e executar somente os trabalhos com autorização prévia, conforme seu nível hierárquico dentro da empresa. Em um outro cenário, suponha que a companhia desenvolva projetos de peças mecânicas e está trabalhando em peças revolucionárias. Se não houver um controle de privilégios aos acessos destas informações, o projeto pode ser roubado (copiado por alguém) ou ser apagado por alguém que não deveria ter acesso e achou que não era algo importante. As consequências do acesso indevido são muitas e podem causar prejuízos muito altos. A empresa investe para desenvolver um projeto e, por causa de falhas em suas políticas, pode ter perdas significativas.

A segurança da informação ganhou notoriedade nestas últimas décadas. Riscos de ataques internos e externos podem afetar os dados e a própria administração da empresa, então é necessário criar regras que garantam a segurança aos seus dados, ou seja, a política de segurança da empresa. As regras são criadas conforme as necessidades de cada companhia, estabelecendo critérios de acesso físico e remoto ao banco de dados. Navathe e Ramez (2005, p. 525-527) citam que um dos itens mais importantes do SGBD são as senhas que controlam o acesso ao banco de dados. Uma política de segurança de um banco de dados deve possuir os seguintes itens:

- Integridade: garantia de que as informações serão mantidas de forma íntegra e sem modificações indevidas de pessoas não autorizadas.
- Confiabilidade: garantia de que as informações armazenadas no banco somente serão acessadas por pessoas autorizadas previamente.

- Disponibilidade: garantia de disponibilizar a informação somente às pessoas com permissão de acesso e modificação.

Korth, Silberschatz e Sudarshan (2012, p. 87) demonstram que podemos atribuir a um usuário várias formas de autorizações a um banco de dados:

- Somente de leitura dos dados.
- Para inserir novos dados.
- Para atualizar novos dados.
- Para excluir dados.

As autorizações dadas aos usuários são chamadas de privilégios e podemos estendê-los de um usuário para todos. Os privilégios também podem ser uma combinação de autorizações. É nesta hora que entra a política de segurança do banco de dados da empresa: quando um usuário realiza uma consulta ao banco de dados, primeiramente o SGBD verifica se o usuário possui a autorização necessária para realizar a ação que pretende, de acordo com Korth, Silberschatz e Sudarshan (2012, p. 88). A maior autoridade sobre o banco de dados é o DBA (Administrador do Banco de Dados), pois ele tem acesso tanto ao esquema do banco (diagramas) quanto aos dados armazenados, além de ser quem autoriza os acessos ao banco de dados, limitando e até criando novos usuários com permissões distintas.



Pesquise mais

O SQL *Injection* é uma das principais formas de ataques ao banco de dados. Conheça um pouco mais sobre essa tecnologia acessando o vídeo abaixo:

ROSA, Alex. **Entendendo o SQL Injection – Vídeo 1**. 2016. Disponível em: <<https://www.youtube.com/watch?v=OmDCgJ9eQDw>>. Acesso em: 11 jun. 2018.

Chegamos ao fim da primeira unidade deste livro. Conceitos essenciais foram apresentados sobre os SGBDs e suas tecnologias, além de subsídios para poder indicar uma ferramentas para um cliente, conforme a necessidade. Um analista de sistemas precisa estar ciente de que suas decisões podem afetar positiva ou

negativamente o negócio das empresas. A confiança que o cliente deposita nos conselhos que recebe ao solicitar um software é muito grande. Na próxima unidade você aprenderá sobre os princípios da modelagem de dados, para que possam ser criados nos SGBDs vistos nesta unidade.

Sem medo de errar

Nesta seção estudamos assuntos que têm uma essencial importância no futuro do banco de dados da empresa. Aprendemos sobre os conceitos e a importância de OLTP e OLAP para as companhias. A redundância e a necessidade de estabelecer regras para garantir um banco de dados mais seguro também foram apresentadas.

Agora, como desafio, a empresa de planos de saúde, Saúde e Vida, deseja estudar o comportamento de seus clientes. Para isso, ela gostaria de utilizar a base de dados com as informações dos usuários de seus planos de saúde para tomar decisões estratégicas. Como você poderá ajudar essa empresa? Como decidir no que investir para obter resultados mais estratégicos? Além disso, a organização teme que essas informações preciosas acabem nos concorrentes, então precisa que você a auxilie nesta questão.

Você precisará indicar uma ferramenta OLAP para a empresa, portanto, poderá realizar uma pesquisa sobre as ferramentas de sistemas em nuvem. Empresas como a Amazon e Microsoft possuem soluções interessantes, como o *Amazon Web Service (AWS)* ou o *Azure*. Essas ferramentas disponibilizam vários serviços de análise de dados, permitindo análises, buscas de padrões e ajuda na tomadas de decisões (com base nos dados do banco de dados). O *Amazon Redshift* é um *Data Warehouse* que pode auxiliar a empresa Saúde e Vida.

Como a companhia também deseja investir na melhoria da segurança do banco de dados, pesquise como podemos utilizar recursos como a criptografia, por exemplo, no banco de dados. Faça uma pesquisa sobre Engenharia Social e como ela pode ser prejudicial à segurança dos dados da empresa. Sugira ações que a instituição pode tomar para prevenir ataques às informações.

Para o backup das informações da empresa, como é de grande porte, pesquise um pacote de serviços em nuvem que possa armazenar grandes quantidades de dados, procure por serviços especializados que possuam uma boa confiabilidade no mercado e com serviços de criptografia e pesquise outras opções, como optar por um servidor exclusivo para o backup.

Faça uma cotação com algumas empresas desenvolvedoras desta tecnologia e aponte os equipamentos necessários: servidores, espaço em nuvem ou outros dispositivos extras. Crie uma apresentação em slide, a fim de apresentar e apontar as melhorias para seu cliente.

Prepare um relatório final para sua empresa sobre os três clientes (de pequeno, médio e grande porte) apresentados nesta unidade, em que deverão ser demonstradas as características do modelo de banco de dados relacional, indicando qual SGBD você recomendaria para cada empresa. Além disso, faça um quadro comparativo dos SGBDs justificando o uso para cada tipo de empresa.

Avançando na prática

Estabelecendo Políticas de Segurança e Backup

Descrição da situação-problema

Você foi contratado por uma empresa de advocacia Irmãos Pedro S.A. que está precisando de uma assessoria e que tem ganhado muitos clientes, aumentando assim a quantidade de informações de processos judiciais. Consequentemente, os sócios estão com receio de perder as informações de seus clientes. A empresa possui, ao todo, doze pessoas trabalhando diariamente no prédio, onde há uma sala de informática com dois servidores localizada no andar térreo, nos fundos do prédio de dois andares. O que você poderia propor para ajudar nos dois seguintes itens: criar uma política de segurança e uma de backup?

Resolução da situação-problema

Para a política de segurança, poderíamos propor algumas sugestões:

1. Revisar no sistema da empresa as permissões de acessos (os privilégios) de cada funcionário que utiliza os softwares e estabelecer critérios de permissão mais rígidos para processos que requerem uma segurança maior.
2. Criar regras de acesso à sala dos servidores, impedindo o livre acesso de funcionários não autorizados no local.
3. Verificar se algum funcionário tem acesso ao SGBD, quem é e se ele é um profissional capacitado.
4. Como a sala do servidor fica no primeiro andar, verificar a possibilidade de risco natural, como inundação, e, se sim, sugerir que a sala seja transferida para um andar superior. Além disso, analisar possibilidade incêndio e ver se os extintores estão corretos.
5. Estabelecer regras na questão da limpeza, orientando o pessoal responsável, a fim de evitar danos acidentais durante o trabalho.

Para a política de backup, poderíamos propor algumas sugestões:

1. Determinar a frequência do backup, verificar se o volume de informações é muito alto e, caso seja, fazer o backup diariamente ou verificar o limite máximo de dias sem backup.
2. Determinar se forma de backup será em mídia ou em nuvem. A nuvem talvez ficaria mais acessível, porém precisaria ter requisitos de segurança. Como os dados são sigilosos, verificar se a empresa não gostaria de um sistema de cópias do banco locais, então, sugerir outro HD ou Fita Dat.
3. Determinar a quantidade de cópias do banco (o ideal é trabalhar com no mínimo três cópias) e sempre ir sobrepondo a cópia ou a mídia mais antiga.
4. Determinar a rotina do backup, ou seja, qual software será usado, os procedimentos que serão realizados e quem o fará.
5. Escolher três lugares distintos para armazenar as cópias do banco de dados e determinar quem será o guardião responsável pelas cópias.

Faça valer a pena

1. Conforme Date (2003), sistemas de apoio à decisão são sistemas que ajudam na análise de informação do negócio. Com a constante concorrência entre as empresas, atualmente os administradores utilizam esta tecnologia para diversas finalidades em suas companhias.

Assinale a alternativa correta referente aos objetivos do uso dos sistemas de apoio à decisão:

- a) Ajudar os gestores a tomarem decisões, porém não é possível prever nenhum tipo de problema com a análise dos dados da empresa.
- b) Aumentar a lucratividade das empresas e a sua consequente valorização no mercado de ações.
- c) Ajudar os gestores a tomarem decisões e a apontarem existentes e até futuros problemas que possam ocorrer.
- d) Encontrar erros no SGBD e estabelecer uma rápida correção para evitar problemas entre entidades fundamentais na estrutura do banco de dados.
- e) Definir a política de segurança do banco de dados e criar uma política de backup do SGBD.

2. Um Sistema Gerenciador de Banco de Dados é um software que possui muitas funcionalidades, sendo uma delas a questão de segurança. Caso ocorra uma falha de hardware ou software, a integridade do banco é mantida. A empresa que mantém um banco de dados precisa ficar atenta e também tomar muitas precauções em relação à segurança do banco de dados, por isso, precisa ter uma política de backup.

Analise com cuidado cada alternativa e escolha a opção correta referente à política de backup que a empresa deve possuir para preservar a integridade de seu banco de dados.

- a) É uma formalidade que às vezes deve ser seguida pela equipe responsável pelo SGBD.
- b) Ela deve determinar o diretor ou gerente administrativo da empresa que deverá realizar o backup diariamente.
- c) Ela deve definir um único local para armazenar as cópias do banco de dados, preferencialmente ao lado do servidor para facilitar o acesso, caso haja um problema no banco de dados.

- d) É estabelecida somente para grandes corporações, pois somente grandes volumes de dados podem ocasionar problemas no banco de dados.
- e) São regras que devem ser estabelecidas para preservar o banco de dados, pois, caso seja necessário recuperar as informações, elas estarão em lugar seguro.

3. As duas siglas OLTP e OLAP são bastante usadas no mundo de *Business Intelligence* e pode-se afirmar que as duas são importantes e se complementam. Com isso, é possível dizer que:

- I. OLTP é voltado para o sistema de transações, regras de negócios aplicadas no sistema do dia a dia da empresa.
- II. OLAP é voltado para a análise das informações, ou seja, cálculos mais complexos, um sistema de modelagem voltado para sistemas de apoio à decisão.
- III. OLTP é voltado para sistemas de grandes proporções e usados para tomar decisões complexas no mercado financeiro.
- IV. OLAP é voltado para a análise de pequenas informações do dia a dia de uma empresa, sendo que empregados operacionais podem fazer essas consultas.

Analise atentamente cada assertiva e marque o item correto, referente às finalidades do OLTP e OLAP.

- a) Somente a assertiva I está correta.
- b) Somente a assertiva II está correta.
- c) Somente as assertivas I e II estão corretas.
- d) Somente as assertivas III e IV estão corretas.
- e) Somente as assertivas II e III estão corretas.

Referências

COUGO, P. **Modelagem conceitual e projeto de bancos de dados**. Rio de Janeiro: Elsevier: 1997.

DATE, C. J. **Introdução a sistemas de banco de dados**. 8. ed. Rio de Janeiro: Elsevier, 2003.

GUIMARÃES, C. C. **Fundamentos de bancos de dados: modelagem, projeto e linguagem SQL**. Campinas, SP: Editora da Unicamp, 2003.

KORTH, H. F.; SILBERSCHATZ, A.; SUDARSHAN, S. **Sistema de banco de dados**. Rio de Janeiro: Elsevier, 2012.

MONTEIRO, M. A. **Introdução à organização dos computadores**. 5. ed. Rio de Janeiro: LTC, 2014.

NAVATHE, S. B.; RAMEZ, E. **Sistemas de banco de dados**. 4. ed. São Paulo: Addison Wesley, 2005.

Modelos de banco de dados

Convite ao estudo

Caro aluno, seja bem-vindo a esta segunda unidade!

Com tanta demanda de *software*, você agora possui sua própria empresa. Você se tornou uma pessoa jurídica, em busca de novos clientes e novas conquistas. Suas responsabilidades são maiores, principalmente no quesito conhecer melhor seu cliente. Para conhecer melhor o funcionamento do processo a ser modelado, serão necessárias algumas informações para conhecer o negócio do cliente e também os usuários do software que será produzido.

Seu novo cliente é uma oficina mecânica de automóveis, o proprietário, Sr. Ruddy, precisa urgentemente do controle dos clientes, que trazem seus automóveis para revisões e consertos. A oficina é de pequeno porte, especializada em uma determinada marca de automóveis importados, possui quatro mecânicos e mais dois funcionários, um para o serviço de limpeza e outro para o setor administrativo.

A modelagem de dados é uma metodologia utilizada para a especificar os requisitos que um determinado software deverá possuir. Essa metodologia faz parte do processo desenvolvimento de um *software*. Modelar dados envolve uma série de aplicações teóricas e práticas, visando construir um esquema banco de dados que possa ser utilizado em qualquer Sistema Gerenciador de Banco de Dados (SGBD).

Para atingir o objetivo da construção de um esquema de banco de dados, nesta unidade você aprenderá as diferenças

entre os modelos conceitual, lógico e físico, além de ver com detalhes os elementos usados na construção da modelagem.

Fique atento aos conceitos que serão demonstrados, eles serão essenciais para a compreensão da disciplina.

Vamos em busca de novas conquistas!

Seção 2.1

Modelos de banco de dados

Diálogo aberto

A tecnologia invadiu o cotidiano das pessoas. Utilizar aplicativos em computadores, *tablets* ou *smartphones*, para os mais diversos fins, faz parte do dia a dia de todos. Várias empresas estão usando aplicações para oferecerem serviços diferenciados para seus clientes. Nesse universo, podemos citar os aplicativos de sistemas bancários, nos quais você pode fazer pagamentos, transferências e diversos outros serviços. O sucesso dessas aplicações depende de um sistema muito bem estruturado, e uma das etapas primordiais nesse processo é a modelagem do banco de dados: um equívoco no modelo pode acarretar em funcionalidades incorretas e ocasionar problemas para a empresa e, como consequência, problemas para seus clientes.

Como analista de sistemas, você precisará fazer o levantamento dos requisitos para saber as necessidades do cliente e, desta forma, começar o processo de modelagem. A oficina mecânica de carros importados é de pequeno porte, sendo necessário o controle de peças usadas nos serviços efetuados. Para realizar essa tarefa, é preciso conhecer a rotina de trabalho da empresa, agendando visitas para conversar com os funcionários e entender como realmente funciona o fluxo de trabalho da empresa. Após algumas visitas, foram levantadas as seguintes informações sobre a oficina mecânica:

- Será necessário um cadastro de clientes e de seus carros.
- Foi constatado um pequeno estoque de peças (de alto giro).
- Há a necessidade de um controle sobre as atividades dos funcionários, o que requer o cadastro dos mesmos.
- A cada serviço deverá ser verificado quem o executou, que materiais usou e quanto tempo levou.
- Não há agendamentos prévios, o cliente chega e é atendido por ordem de chegada.

- Há uma lista de clientes VIPs, que possuem preferência de atendimento.

Com essa lista, você precisará responder às seguintes questões: quais entidades (tabelas) podem ser listadas para realização do modelo conceitual? Quais informações poderemos guardar em cada tabela? Qual SGBD você recomendaria?

Chegamos num dos itens mais importantes da carreira de um analista de sistemas: a modelagem de dados. Ser eficiente nesse item é um dos destaques desta profissão. Poder planejar um sistema de forma correta, prevendo futuros problemas é um dos destaques de sucesso dos analistas de sistemas no mercado tecnológico.

Chegou o momento de começar a modelagem de dados.

Bons estudos!

Não pode faltar

A modelagem de dados, de acordo com Coronel e Rob (2011), é um processo que irá se repetir de forma progressiva, começando com uma compreensão simples de um determinado problema e, logo que haja um melhor entendimento do problema a ser modelado, o nível de detalhes do modelo também irá se ampliar. Uma das fases mais cruciais no desenvolvimento de um software é justamente a elaboração de um projeto de banco de dados. Criar um banco de dados sem um estudo e de forma apressada pode causar vários problemas, tais como desempenho abaixo do esperado, consultas complicadas e, ainda, resultados equivocados, o que impacta de forma negativa a performance do software.

Cougo (1997) descreve que um modelo de dados é um detalhamento dos tipos de informações que serão guardadas em um banco de dados. Para a construção de modelos de dados, usa-se uma linguagem de modelagem de dados, essas linguagens são qualificadas conforme a forma de apresentar modelos, podendo ser: a linguagem textual ou linguagem gráfica. Esta forma de representação de um modelo de dados, por meio de uma linguagem de modelagem de dados, é conhecida como: esquema de banco de dados (KORTH; SILBERSCHATZ; SUDARSHAN, 2012).

Um modelo de dados, de acordo com Coronel e Rob (2011), é uma reprodução gráfica de estruturas de dados de situações reais. Um modelo pode representar uma abstração de alguma coisa real ou de algum objeto. Utilizar um modelo tem como finalidade auxiliar na compreensão das situações reais. Em um sistema que utiliza banco de dados, o modelo irá representar as estruturas das tabelas, os dados e seus relacionamentos.

Utilizamos os modelos de forma gráfica para conseguirmos simplificar a comunicação com o usuário. Na Figura 2.1, podemos observar uma representação gráfica que, mesmo sem muita noção sobre banco de dados ou qualquer área da computação, pode ser facilmente compreendida: uma receita de bolo possui diversos ingredientes e, com um pouco de analogia iria perceber que os ingredientes podem ou não ser usados nas receitas de bolo. Os modelos gráficos como forma de comunicação com os usuários são uma excelente ferramenta. Com poucas explicações, os usuários já se tornam aptos a darem suas opiniões.

Figura 2.1 | Exemplo de representação gráfica



Fonte: elaborada pela autora.



Exemplificando

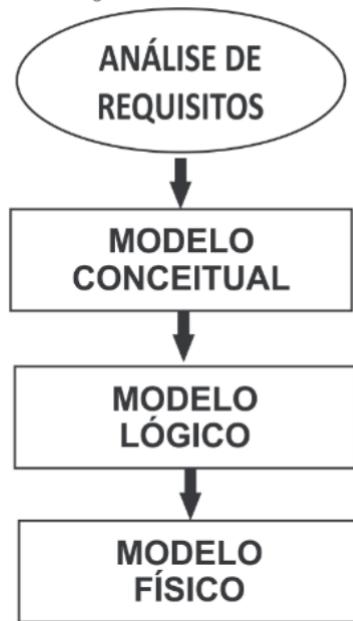
Objetos ou eventos do mundo real podem ser modelados e transformados em entidades (tabelas). São exemplos de objetos do mundo real que podem ser modelados: clientes, empresas, funcionários, produtos etc. Eventos reais são ações que geram atributos e que devem ser preservados, tais como: reservas, atendimentos, locação, dentre outros.

Abreu e Machado (2004) afirmam que o projeto de um sistema de informações é uma atividade complexa, que inclui planejamentos, especificações e desenvolvimento de vários componentes. É necessário estabelecer uma sequência de atividades para guiar o processo de modelagem do banco de dados, sendo elas:

- **Análise dos requisitos:** levantamento das necessidades do cliente.
- **Modelo conceitual:** não contém detalhes sobre como será representado em meio físico; representa as informações no nível da realidade do que será modelado.
- **Modelo lógico:** descreve as estruturas que estarão contidas no banco de dados, de acordo com a abordagem da modelagem a ser utilizada.
- **Modelo físico:** descreve o detalhamento ao nível do SGBD; nível físico de criação dos componentes do banco de dados.

Segundo Coronel e Rob (2011), o projeto de banco de dados focaliza em como a estrutura do banco será utilizada para armazenar e gerenciar todos os dados do sistema do usuário final. Na Figura 2.2, podemos observar a sequência do processo de modelagem de um banco de dados.

Figura 2.2 | Processo de modelagem de dados



Fonte: elaborada pela autora.

O processo de inicialização de um banco de dados de sucesso começa no procedimento de levantamento e de análise de requisitos

(CORONEL; ROB, 2011). Devemos estudar o domínio do problema que o banco de dados deverá solucionar. Por meio dos requisitos, é realizado um levantamento das necessidades que o software deverá possuir. Diversas reuniões com o cliente deverão ser realizadas para que o analista possa detectar as reais necessidades do cliente e também para conhecer as rotinas de trabalho da empresa. Desta forma, a chance de haver problemas na modelagem por causa de um requisito esquecido será reduzida.

Cougo (1997) destaca alguns pontos importantes que devem ser considerados, na análise de requisitos:

- Abrangência: determinar o escopo do projeto para poder determinar o que será realizado e qual processo terá a necessidade de ser observado (na empresa) para realizar a modelagem do banco de dados.
- Nível de detalhamento: definir qual o nível de detalhamento que o projeto deverá possuir.
- Tempo para a produção do modelo: é necessário estabelecer um tempo para realizar a modelagem. Nesta fase, muitos problemas podem ser encontrados e rapidamente solucionados.
- Recursos disponíveis: estabelecer a quantidade de mão de obra para desenvolver o software solicitado.

Korth, Silberschatz e Sudarshan (2012) relatam que, em geral, nenhum usuário entende todas as necessidades de uma aplicação. O analista de sistemas precisa interagir com os usuários a fim de identificar e criar as regras de negócio, pois, caso essa parte seja mal executada, corremos o risco de ter que refazer a modelagem. Como resultado da análise dos requisitos, são produzidos diversos documentos de especificação do projeto que deverão ser validados com o cliente, mas não nos aprofundaremos sobre tal documentação, nos atendo ao foco desta disciplina, o processo de modelagem.



Exemplificando

Um requisito se refere as funcionalidades que o software deverá possuir, a regra de negócio determina como o software deverá

se comportar e quais restrições deverá possuir. Tanto o requisito quanto, a regra de negócio possui um foco diferente. São exemplos de requisito e de regra de negócio, respectivamente: matricular o aluno na disciplina de estágio; o aluno somente será matriculado na disciplina de estágio caso não haja dependência no primeiro e no segundo semestre do curso.

Assim que os requisitos do software forem levantados, o próximo passo é a modelagem conceitual. Navathe e Ramez (2005) afirmam que a modelagem conceitual é uma descrição concisa das informações que o software deverá possuir, de acordo com seus requisitos. É uma representação do que precisa ser realizado (não é a solução do problema). Após o levantamento dos requisitos é gerado um esquema conceitual ou uma visão mais abrangente e geral do banco de dados.

Podemos utilizar as linguagens textuais ou linguagens gráficas, sem nos preocuparmos com as regras de modelagem de dados, que serão demonstradas mais adiante. A linguagem textual é menos utilizada por ser mais complexa, caso haja muitas informações. Na Figura 2.3, podemos observar, em linguagem textual as entidades Ator e Filme, com alguns campos que caracterizam cada uma.

Figura 2.3 | Linguagem textual

Autor: nome do ator, valor do cachê,
data de nascimento, altura

Filme: nome do filme, data de lançamento,
orçamento, duração

Fonte: elaborada pela autora.

As entidades representadas na Figura 2.3 possuem poucos atributos, mas é comum uma entidade ter muitos deles, o que dificulta a visualização por parte do usuário. Na figura 2.4, a linguagem gráfica é demonstrada de forma simplificada.

Figura 2.4 | Linguagem gráfica simplificada



Fonte: elaborada pela autora.

Utilizamos os modelos de entidade relacionamento para descrever esses esquemas e assim apresentar aos usuários para sanar eventuais dúvidas de entendimento do software. Por exemplo, estamos realizando uma modelagem para um sistema de uma universidade e é nesta hora que definimos algumas situações:

- O aluno poderá se matricular em mais de um curso?
- Um professor pode lecionar em vários cursos?
- Uma turma pode ser composta por alunos de cursos diferentes?

As perguntas anteriores não podem ser respondidas pelo analista de sistemas, mas sim pelo cliente, que irá determinar essas situações. Nessa fase de modelagem conceitual, não é levado em conta como essas informações serão armazenadas no Sistema Gerenciador de Banco de Dados (SGBD).

Segundo Coronel e Rob (2011), o modelo conceitual traz algumas vantagens importantes: fornece a visão de nível macro, de forma simplificada e independente de hardware e de software, não sendo necessária a realização de adaptações em função de determinado SGBD ou equipamento que serão adotados posteriormente.



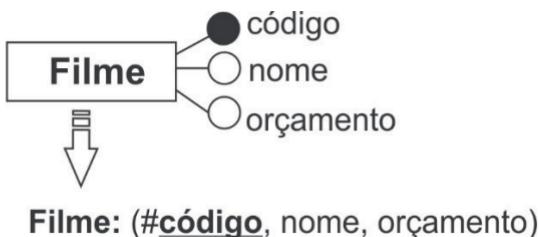
Assimile

O alvo do modelo conceitual é a definição do problema e não a sua solução. Mostra o que precisará existir no banco de dados de uma forma geral, sem se preocupar de que forma isso será realizado no SGBD.

O modelo lógico do banco de dados é a etapa em que mapeamos o conceito de modelos de entidade e relacionamentos com o foco na criação do banco de dados. Nessa etapa, as entidades são transformadas em tabelas para armazenar as informações, os relacionamentos são estabelecidos, as regras são e os de tipos

de dados para cada campo da tabela são determinados (KORTH; SILBERSCHATZ; SUDARSHAN, 2012). O modelo lógico se transforma de um modelo mais abstrato (conceitual) para um modelo com mais detalhes de implementação, ou seja, mais próximo do que será de fato implementado. Cougo (1997) define como modelo lógico de dados aquele em que os objetos, suas características e seus relacionamentos sejam representados de acordo com as regras de implementação e com os limites impostos por alguma tecnologia de determinado SGBD. No modelo lógico serão determinados os relacionamentos e as chaves entre as tabelas, assunto que abordaremos na Seção 2.2. Além da forma gráfica, podemos utilizar a forma textual (Figura 2.5), que ajuda a detalhar os atributos da tabela, deixando a forma gráfica mais “limpa”, somente com a figura e com os seus respectivos nomes das tabelas.

Figura 2.5 | Forma Gráfica e Textual



Fonte: elaborada pela autora.



No modelo da Figura 2.5, é criado um campo chave chamado código. Observe que foi utilizado o sinal # para determinar que esse atributo é diferente dos demais e que possui um papel importante, conforme veremos mais adiante.

Uma boa prática de desenvolvimento de banco de dados é um modelo lógico de dados ser criado a partir do modelo conceitual. Contudo, o modelo lógico pode ser construído diretamente, sem o modelo conceitual, a desvantagem é que a participação do usuário pode ficar comprometida e a possibilidade de erros é grande no ambiente de negócios da empresa.

Na última fase do projeto de banco de dados é realizada a modelagem física. Navathe e Ramez (2005) afirmam que é justamente nesta fase que são determinadas as estruturas de armazenamento interno, as chaves (ou índices) e os diversos caminhos de acessos a base de dados. Paralelamente às atividades de modelagem física, são criados os softwares de aplicação que irão interagir com o banco de dados implementado. Nessa fase, as regras de cada SGBD devem ser utilizadas e as regras de segurança devem ser implementadas, tais como as políticas de *backup* e as permissões de acessos de cada usuário do banco de dados.

Korth, Silberschatz e Sudarshan (2012) descrevem que, na modelagem física, é utilizada a linguagem *Structured Query Language* ou Linguagem de Consulta Estruturada (SQL), que tem como principal objetivo a manipulação dos bancos de dados relacionais e é utilizada para interagir com o usuário e com o SGBD, permitindo inserir, consultar, gerenciar, controlar transações, entre outras opções.

Cougo (1997) enfatiza que durante essa fase são utilizadas ferramentas CASE, que auxiliam nas tarefas do desenvolvimento de software, desde a análise de requisitos e modelagem, até a programação e os testes na elaboração dos modelos, levando-se sempre em conta o SGBD que será utilizado.



Assimile

Software de aplicação são conjuntos de programas de computador com os quais o usuário pode realizar determinadas tarefas. Por exemplo: o software acadêmico que a secretaria utiliza é o software de aplicação e este software acessa o banco de dados, criados em um SGBD.

Para exemplificar um projeto de banco de dados, vejamos o seguinte estudo de caso como demonstração da diferença entre os modelos conceitual e lógico: uma escola de ensino fundamental bilíngue necessita de um software para seu gerenciamento acadêmico. Após algumas entrevistas, o analista de sistemas levantou os seguintes requisitos essenciais para o projeto de banco de dados:

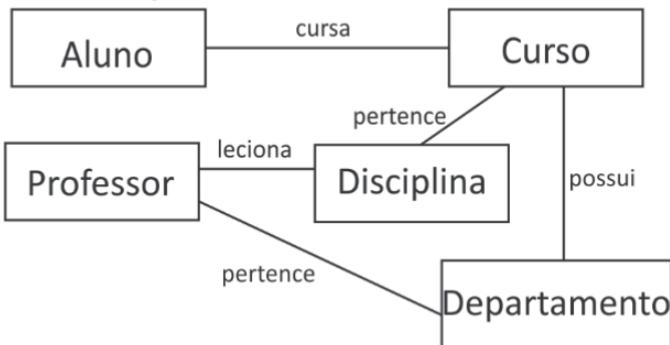
- A escola possui diversos departamentos, divididos entre as grandes áreas de conhecimento: matemática, estudo da linguagem etc.
- Um departamento pode oferecer diversas disciplinas, mas uma disciplina pertence a somente um departamento.
- Um aluno somente pode estar matriculado em um único curso.
- Uma mesma disciplina pode constar no currículo de diversos cursos.
- Todo professor pertence a um departamento e poderá lecionar em diversas disciplinas.

O analista de sistemas também fez um levantamento sobre quais informações são essenciais e deverão estar armazenadas nas entidades, gerando os seguintes atributos:

- Professores: código, nome, formação, endereço, telefone.
- Curso: código, nome, sigla.
- Disciplinas: código, denominação, sigla, ementa.
- Departamentos: código, denominação.
- Aluno: matrícula, nome, endereço, telefone, filiação e data de nascimento.

Conforme os requisitos determinado pelo cliente, foi elaborado um modelo conceitual simplificado para uma apresentação (Figura 2.7), lembrando que o objetivo é mostrar a estrutura macro do banco de dados.

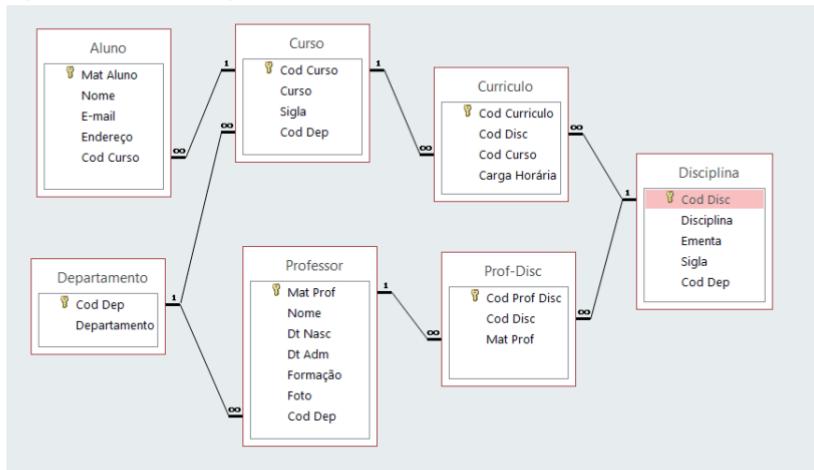
Figura 2.7 | Modelo Lógico de uma Escola



Fonte: elaborada pela autora.

O modelo lógico do estudo de caso pode ser analisado na Figura 2.8. Observe que novas entidade apareceram: currículo e prof-disc (nas próximas seções o surgimento destas entidades ficará mais compreensível). Para a criação do modelo lógico, que foi simplificado para fins didáticos, foi utilizado o SGBD Access®, pela sua facilidade de compreensão.

Figura 2.8 | Modelo Lógico em um SGBD



Fonte: captura de tela no Microsoft Access, elaborada pela autora.



Refletia

O modelo lógico é sempre mais complexo que o modelo conceitual. Observando as Figuras 2.7 e 2.8, onde o boletim de cada aluno se encaixaria? E o diário do professor, de que forma poderia ser representado? A formação do professor não poderia ser uma tabela?

A última etapa, o modelo físico, dependerá do SGBD a ser utilizado. E, somente para termos uma ideia dos comandos SQL utilizados nesta etapa, observe a figura 2.9 que tem a finalidade de criar uma tabela, no caso, a tabela Departamento.

Figura 2.9 | Exemplo comando SQL

```
CREATE TABLE Departamento (
cod_dep int primary key not null,
departamento varchar(150));
```

Fonte: elaborada pela autora.



Pesquise mais

SQL é uma linguagem muito utilizada no banco de dados pelo DBA e por usuários mais capacitados na empresa. Conheça mais sobre SQL e conheça os seus comandos básicos.

EDSON PIMENTEL. **Sql** – DDL – Introdução. 29 nov. 2009. Disponível em: <<https://www.youtube.com/watch?v=FHflxAyQlt8>>. Acesso em: 19 abr. 2018.

As primeiras etapas da modelagem do banco de dados (conceitual e lógica) são de grande relevância para atender às necessidades do cliente, enquanto a última etapa, de modelagem física, está voltada diretamente ao SGBD escolhido para ser utilizado na criação do banco de dados. Cada etapa possui a sua importância, mas projetar um banco de dados é vital para o sucesso do software que está sendo desenvolvido.

Sem medo de errar

Você agora é um empreendedor em busca do sucesso. Seu novo cliente é uma oficina mecânica de automóveis. A sua missão é realizar um modelo conceitual para apresentar ao sr. Ruddy, o proprietário da oficina. O passo inicial para criar um modelo conceitual é observar os requisitos levantados, por meio dos quais poderemos retirar várias informações. Os requisitos da oficina, para a realização do software são:

- Será necessário um cadastro de clientes e de seus carros.
- Foi constatado um pequeno estoque de peças (de alto giro).

- Há a necessidade de um controle sobre as atividades dos funcionários, o que requer o cadastro dos mesmos.
- A cada serviço deverá ser verificado quem o executou, que materiais usou e quanto tempo levou.
- Não há agendamentos prévios, o cliente chega e é atendido por ordem de chegada.
- Há uma lista de clientes VIPs, que possuem preferência de atendimento.

Vale ressaltar, conforme visto anteriormente na seção, que uma tabela deverá guardar informações. Em cada linha dos requisitos, deveremos buscar por substantivos fortes e que indicam que poderemos guardar informações em um banco de dados. Observe o primeiro item: será necessário um cadastro de clientes e de seus carros. Por meio desse requisito, podemos guardar informações muito importantes sobre duas tabelas: Cliente e Carro. E quais informações poderemos guardar sobre clientes e carros? Diversas, observe algumas:

Cliente: nome, CPE, endereço, CEP, telefone fixo, celular, e-mail, data de nascimento.

Carro: placa, modelo, ano, tipo de combustível, quilometragem.

Observe que nas duas tabelas (Cliente e Carro) há dois campos sublinhados. Estes campos estão em destaque, pois ajudarão a identificar as informações de cada tabela.

Agora é com você:

- Encontre mais campos para as tabelas Cliente e Carro.
- Observe atentamente os demais requisitos e encontre mais tabelas e seus respectivos campos.
- Em cada tabela, sublinhe um campo que você considera o mais importante e que servirá para ajudar a encontrar o registro da tabela.

Crie o modelo conceitual de forma textual (continue conforme as duas tabelas de exemplo). Crie mais campos nas tabelas e sublinhe o campo mais importante da tabela, que ajudaria a identificar o registro dela. Aponte também qual SGBD você indicaria para ser

utilizado no desenvolvimento do software. Crie uma apresentação para apresentar ao cliente.

Avançando na prática

Modelagem conceitual para uma Floricultura

Descrição da situação-problema

A Flores Belas é uma floricultura que trabalha com entregas de flores e presentes. Nas datas especiais, há um acúmulo de encomendas e entregas e, para agilizar esse processo, será desenvolvido um software no ambiente Web. A sua missão será elaborar um modelo conceitual do banco de dados da floricultura. Uma lista de requisitos básicos já foi elaborada:

- Será necessário cadastrar clientes, encomendas, local de entrega e produtos.
- Um cliente poderá realizar várias encomendas, mas cada encomenda pertence a apenas um cliente.
- Cada encomenda poderá ter vários produtos.
- Os produtos estão classificados por tipo: flores, chocolates, presentes, cartão etc.
- Uma encomenda sempre deverá ter um local de entrega.
- Não será necessário o controle de pagamento, pois o mesmo será feito via cartão de crédito.

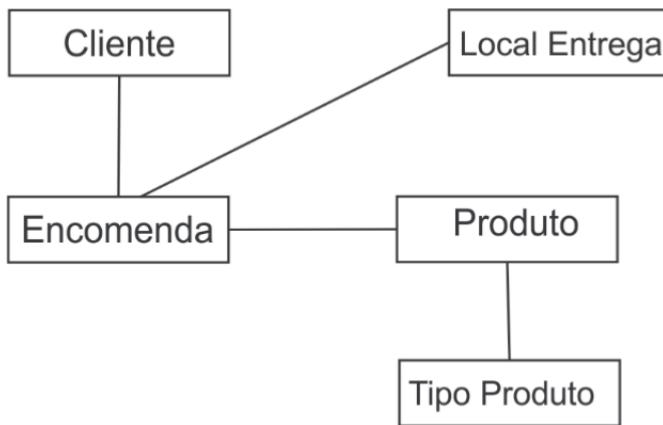
De acordo com os requisitos listados, você deverá criar o modelo conceitual gráfico e responder aos seguintes questionamentos:

- Quais são as tabelas que irão compor o modelo conceitual?
- Como ficará o relacionamento entre essas tabelas, no modelo conceitual e de forma gráfica?

Resolução da situação-problema

Uma sugestão de modelo conceitual pode ser observada na Figura 2.10, em que podemos ter uma visão macro do funcionamento da floricultura, conforme os requisitos apresentados. Esse modelo servirá de partida para a próxima etapa de modelagem, que é o modelo lógico.

Figura 2.10 | Modelo conceitual da floricultura Flores Belas



Fonte: elaborada pela autora.

Faça valer a pena

1. Segundo Coronel e Rob (2011), o modelo conceitual simula uma visão global do banco de dados, disponibilizando uma visão geral de como o banco de dados é na realidade, sendo representado por meio de um esquema do banco. É uma das ferramentas de comunicação entre o pessoal de desenvolvimento do software e o cliente.

Marque a alternativa que demonstra o nome do modelo que deve ser realizado após a criação do modelo conceitual:

- a) Modelo físico.
- b) Modelo de entidades.
- c) Modelo de requisitos.
- d) Modelo lógico.
- e) Modelo particional.

2. Korth, Silberschatz e Sudarshan (2012) descrevem que, na _____, é utilizada a linguagem *Structured Query Language*, ou Linguagem de Consulta Estruturada (SQL), que tem como principal objetivo a manipulação dos bancos de dados relacionais e é utilizada para interagir com o usuário e o _____, permitindo inserir, consultar, gerenciar, controlar transações, entre outras opções.

Analise os itens atentamente e marque a opção correta e que complete a sentença respectivamente:

- a) entidade – SGBD.
- b) classe – software.
- c) modelagem física – SGBD.
- d) modelagem conceitual – software.
- e) modelagem lógica – programa de aplicação.

3. O sucesso de um banco de dados começa muito antes do desenvolvimento do software. Várias etapas devem ser realizadas para atender às expectativas e necessidades do cliente. Uma etapa principal é entender o domínio do problema antes de efetivamente desenvolver o software.

Analise cuidadosamente os itens e marque a alternativa correta que demonstra o objetivo de “entender o domínio do problema”.

- a) Ajuda a equipe de usuários a criar novos atributos para as entidades.
- b) Ajuda a equipe de desenvolvimento a compreender questões relacionadas com a construção do sistema.
- c) É utilizado em último recurso, caso haja problemas no desenvolvimento do software.
- d) É utilizado diretamente na modelagem física do banco de dados, com auxílio do SGBD.
- e) Serve para estabelecer as restrições de acesso físico ao banco de dados.

Seção 2.2

Modelagem de dados através do modelo entidade-relacionamento

Diálogo aberto

Começamos a parte mais emocionante da disciplina: a modelagem de um banco de dados. Nesta hora, o modelo conceitual servirá de apoio à criação do modelo lógico do banco de dados. Analisaremos mais de perto cada requisito do banco de dados para atingir um modelo mais eficaz. Você começará a aprender a criar o modelo relacional utilizando notação gráfica. Com a aprendizagem desta seção você será capaz de estabelecer os relacionamentos entre as tabelas e as suas cardinalidades, aprimorando a técnica de modelagem de dados.

Agora, como empreendedor, você tem muitas responsabilidades, uma delas é desenvolver um software com eficiência. Dando continuidade ao processo de modelagem de dados da oficina mecânica do Sr. Ruddy, você deverá criar o Diagrama de Entidade-Relacionamentos, que será o modelo lógico do banco de dados da oficina. Para relembrar: a oficina precisa controlar seus clientes, que trazem seus automóveis para revisões e consertos. Será necessário, também, controlar as peças usadas nos serviços efetuados, quem consertou o automóvel, o que foi consertado e quanto tempo o carro levou para ficar pronto. Resgate a análise de requisitos e o modelo conceitual que você criou na última seção e crie o Diagrama de Entidade-Relacionamentos. Você precisará, ainda, considerar a necessidade de controlar os tipos de peças, tipos de clientes (empresas terão descontos especiais) e as ordens de serviço de conserto dos automóveis. O controle de pagamento do conserto foi postergado, ou seja, o cliente ainda não quer esse tipo de controle.

Observe atentamente as regras dos relacionamentos e aplique-as de acordo com as necessidades do cliente para responder as seguintes questões:

- Quais são as novas tabelas que irão surgir?

- Como ter certeza que os relacionamentos e cardinalidades propostos estão corretos?

Avance nos estudos e fique atento aos conceitos apresentados, para poder solucionar a realização deste desafio.

Bons estudos!

Não pode faltar

O Modelo de Entidade-Relacionamentos (MER) foi desenvolvido para aperfeiçoar o projeto do banco de dados, permitindo a especificação do modelo conceitual (KORTH; SILBERSCHATZ; SUDARSHAN, 2012). É o modelo mais utilizado pelos Sistemas Gerenciadores de Banco de Dados e foi elaborado por Edgar F. Codd, em 1970, mas somente a partir de 1987 começou a ser adotado pelas empresas de desenvolvimento de software.

O modelo lógico, ou seja, o modelo relacional de um banco de dados, é criado a partir do levantamento de requisitos e do modelo conceitual. O processo de mapeamento dos dados entre os modelos conceitual e lógico é chamado de modelagem relacional.

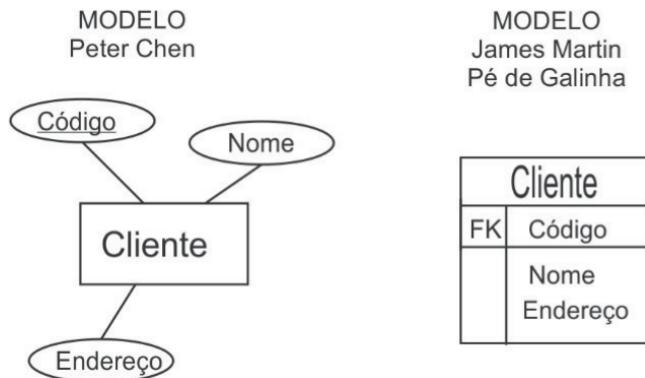
A abordagem relacional, segundo Abreu e Machado (2009), parte do princípio que as informações em uma base de dados podem ser consideradas como relações matemáticas e que devem ser representadas em formas de tabelas. Os autores também destacam as principais vantagens da abordagem relacional:

- Independência total dos dados;
- Visão múltipla dos dados;
- Melhor comunicação entre analistas e usuários comuns;
- Redução e melhor gerenciamento do tempo de desenvolvimento e de manutenção;
- Melhor segurança dos dados;
- Maior agilidade no gerenciamento da informação.

A representação gráfica da modelagem relacional é a forma de representação dos componentes do modelo lógico de um banco de dados. Essa representação é uma parte muito importante da

compreensão do esquema do banco de dados. Uma representação simples e intuitiva é fundamental para o entendimento e comunicação das pessoas envolvidas na criação do modelo do banco de dados. De acordo com Korth, Silberschatz e Sudarshan (2012), existe uma série de notações alternativas para a realização da modelagem, sendo as mais utilizadas a Peter Chen, a IDEF1X, a James Martin (com o famoso Pé de Galinha) e a UML. Existem diversos softwares para a modelagem da representação gráfica, que serão vistos na próxima unidade. Na Figura 2.11, podemos observar a tabela Cliente sendo representada de duas formas (notação) diferentes.

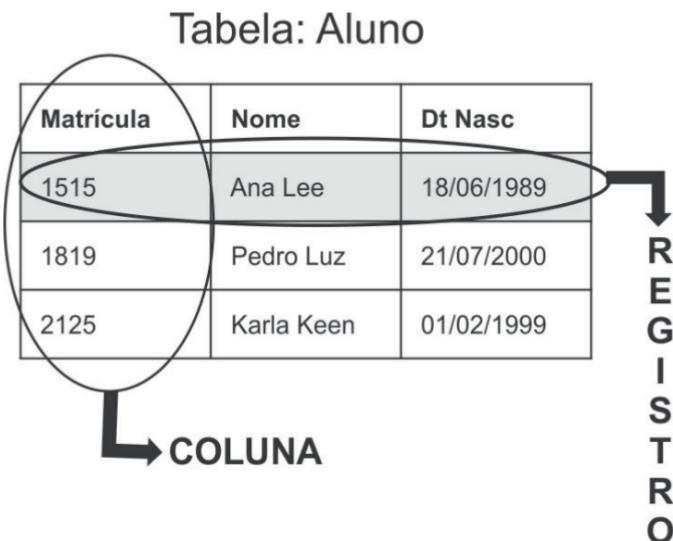
Figura 2.11 | Representação Gráfica Peter Chen e James Martin



Fonte: elaborada pela autora.

Conforme afirmam Coronel e Rob (2011), o fundamento do modelo relacional é um conceito matemático conhecido como relação, no qual dois conjuntos numéricos possuem seus termos relacionados entre si. No modelo conceitual, um conjunto é chamado de entidade, já no modelo lógico é chamado de tabela. Cada tabela é definida com um conjunto de atributos, também conhecidos como campos, que descrevem suas características particulares. Na Figura 2.12, podemos observar uma tabela que possui três campos, que são as colunas da tabela.

Figura 2.12 | Representação gráfica da tabela Aluno



Fonte: elaborada pela autora.

É na fase de projeto de um banco de dados que os campos de uma tabela são definidos. Cada campo possui uma classificação conforme o seu tipo, podendo ser: inteiro, texto, decimal, monetário, lógico, etc. Um tipo de dado interessante no SGBD é o tipo de autoincremento, que tem como característica principal que o seu valor aumenta automaticamente ao se inserir um registro, o que é muito útil para o campo que será usado para a identificação de um registro na tabela, o que aprenderemos em breve.



Exemplificando

Arquivos como fotos, sons, vídeos, PDF, etc. podem ser armazenados num campo do tipo *Binary Large Objects* (BLOB). O tipo BLOB também pode ser utilizado para backup de banco de dados, permitindo armazenar os dados e recuperar as informações, caso seja necessário.

Cada linha de uma tabela representa um conjunto de campos conhecidos como registros ou *tuplas*. Uma tabela pode ter milhares de registros (milhares de linhas) e quem limita a quantidade de linhas

de uma tabela é o SGBD. Os registros de uma tabela são todos do mesmo tipo (nesse caso, é o tipo da tabela), ou seja, num registro em que guardamos informações sobre alunos não poderemos guardar informações sobre empresas.

Um banco de dados é formado por um conjunto de tabelas relacionadas entre si. Cada tabela do banco de dados deve ter um nome único e significativo, por exemplo: uma tabela que guarda informações de automóveis pode ter como nome “automóvel”, e não “Tabela_A”. As tabelas possuem algumas características (CORONEL; ROB, 2011):

- A tabela é vista como uma estrutura composta de linhas e colunas (bidimensional).
- Cada linha ou registro representa uma única ocorrência da entidade no interior do conjunto da entidade.
- Cada coluna da tabela representa um atributo e possui nome diferente dos demais atributos da mesma tabela.
- Cada intersecção entre linha e coluna representa um único valor, que é o dado da tabela.
- Todos os valores em uma coluna devem possuir o mesmo formato.
- A ordem das colunas e das linhas é insignificante para um SGBD.
- Cada tabela deve representar um atributo (chamado de chave, que será estudado mais adiante) ou uma combinação de atributos que identifique exclusivamente cada linha.

Cada tabela que representa uma entidade do modelo conceitual pode ser classificada em: Entidade Forte ou Entidade Fraca. Conforme Cougo (1997), essa distinção se dá por meio da análise de existência de duas condições básicas: dependência de existência ou dependência de identificador. Uma entidade é fraca se um desses dois tipos de dependência existir no relacionamento entre duas entidades. Segundo Navathe e Ramez (2005), uma entidade fraca sempre possui uma restrição de participação total, uma dependência de existência a uma determinada entidade. Korth, Silberschatz e Sudarshan (2012) e Navathe e Ramez (2005) classificam os tipos de entidades como:

- **Entidades Fortes:** são uma tabela autônoma que não depende de outra para sua existência. Alguns exemplos: Aluno, Curso, Cliente, Empresa, Paciente. Na análise de requisitos, são facilmente encontradas pois são substantivos fortes e significativos.
- **Entidades Fracas ou Dependentes:** são uma tabela que necessita de outra para realmente existir e somente existe por causa da Entidade Forte. Entidades fracas podem ser representadas por meio de retângulos com bordas duplas (na maioria dos modelos atuais ela é representada somente com o retângulo). Por exemplo: a tabela Dependente só existe por que existe a tabela Funcionário, pois, para que exista um dependente cadastrado, tem que existir um funcionário que "possui" esse dependente. Se não existisse a tabela Funcionário, certamente não existiria a tabela Dependente.
- **Entidades Agregadas:** são criadas quando temos um conjunto de campos que se repetem em mais de uma entidade, por exemplo: Aluno e Professor têm dados de endereço; para evitar repetições, podemos criar uma nova entidade agregada chamada Endereço para guardar esse tipo de dado.
- **Entidades Subordinadas:** representam uma especialização em que uma entidade supertipo possui várias entidades subordinadas que são especializadas com atributos específicos. Por exemplo, podemos ter dois tipos de clientes: Pessoa Física e Pessoa Jurídica. Os dois têm campos em comum que ficariam na entidade supertipo Cliente, e nas entidades Pessoa Física e Pessoa Jurídica somente estariam os campos específicos de cada tipo de cliente.

Entidades fortes e fracas podem ser conectadas. Por exemplo, a entidade "aluno" pode estar conectada à entidade "cursos". Essa conexão entre as entidades é chamada de relacionamento. Um relacionamento descreve uma associação entre entidades (CORONEL; ROB, 2011), e relacionamentos envolvendo tabelas fracas resultam em uma tabela associativa e que deve ser representada por meio de um losango com bordas duplas.

- **Entidades Associativas:** somente existem em razão do tipo de relacionamento que existe entre as tabelas. O nome desse tipo de tabela deve ser algo significativo, como Contrato ou Histórico, e é comum podermos nomeá-la com a mistura de nomes entre duas tabelas (como veremos mais adiante). Nos requisitos de um banco

de dados, esse tipo de tabela denota a um verbo ou tempo verbal, por exemplo: atender, contratar, prescrever, entre outras.

Quando temos um relacionamento entre duas entidades, o número de ocorrências de uma entidade que está associada a ocorrências de outra entidade determina o **grau de relacionamento** ou **de cardinalidade** entre as tabelas (ABREU; MACHADO, 2009). O grau de relacionamento é a quantidade de entidades que estão ligadas ao relacionamento, podendo ser:

- Relacionamento unário (grau 1): uma entidade se relaciona com ela mesma.
- Relacionamento binário (grau 2): é um relacionamento que liga dois tipos diferentes de entidades. É o evento mais comum dos tipos de relacionamentos.
- Relacionamento ternário (grau 3): é um relacionamento em que três entidades estão conectadas.
- Relacionamento quaternário (grau 4): é um relacionamento em que quatro tabelas estão conectadas.
- Relacionamento n-ário: é um relacionamento com mais de quatro tabelas envolvidas. Esse tipo de relacionamento é o menos aconselhável, visto que a possibilidade de redundâncias no banco pode ser maior.

Para exemplificar, considere um relacionamento entre as entidades Paciente e Doença. Quais as considerações que precisamos fazer para estabelecê-lo? Algumas sugestões de perguntas:

- Podemos guardar informações dos pacientes e das doenças?
- Um paciente pode ter uma ou mais doenças?
- O paciente pode tratar uma doença e ela voltar a aparecer?
- A doença pode aparecer em outros pacientes?

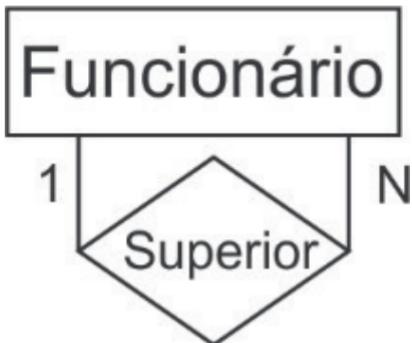
As respostas das perguntas dependem muito do problema a ser modelado, porém, precisamos estabelecer para cada relacionamento a sua cardinalidade. A cardinalidade atribui um valor específico ao relacionamento, expressando a faixa de ocorrências permitidas (mínimas e máximas) entre as tabelas, que, segundo Coronel e Rob (2011), podem ser:

- Auto relacionada; Um para Um (1 - 1);

- Um para Muitos (1 - N);
- Muitos para Um (N - 1);
- Muitos para Muitos (N - N).

O **Auto relacionamento** é um tipo de relacionamento unário, envolvendo somente uma tabela. Nele, os elementos de uma entidade se relacionam a outros elementos dessa mesma entidade, conforme pode ser observado na Figura 2.13, na tabela Funcionário. Todo funcionário possui um chefe ou superior que, por sua vez, também é um funcionário e que supervisiona vários empregados.

Figura 2.13 | Auto relacionamento



Fonte: elaborada pela autora.

Os relacionamentos binários envolvem as cardinalidades: Um para Um, Um para Muitos e Muitos para Muitos. A cardinalidade **Um-para-Um (1 para 1)** tem como característica que cada tabela terá somente uma única ocorrência da outra tabela, por exemplo, em uma agência de empregos **um** Candidato pode cadastrar somente **um** Currículo e o Currículo pertence a somente **um** Candidato, como mostrado na Figura 2.14, em que utilizamos a notação gráfica de Peter Chen.

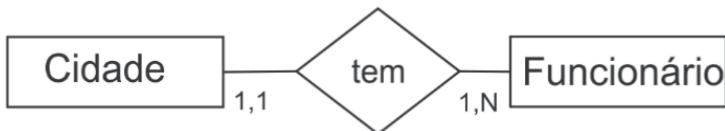
Figura 2.14 | Um-para-Um



Fonte: elaborada pela autora.

Na cardinalidade **Um-para-muitos (1 para N)** ou **Muitos-para-um (N para 1)**, uma das entidades pode referenciar várias unidades da outra, porém, do outro lado, só pode ser referenciada uma única vez. Por exemplo, o Funcionário precisa informar a cidade de seu nascimento e ele só pode indicar uma única Cidade, mas a mesma cidade pode ser referenciada várias vezes por outros Funcionários. Veja a Figura 2.15, que devemos interpretar da seguinte forma: um Funcionário pode estar associado a no mínimo 1 e no máximo a somente 1 Cidade; observe agora a tabela Cidade, ao se relacionar com a tabela Funcionário, uma Cidade precisa ter no mínimo 1 Funcionário e no máximo N (muitos Funcionários).

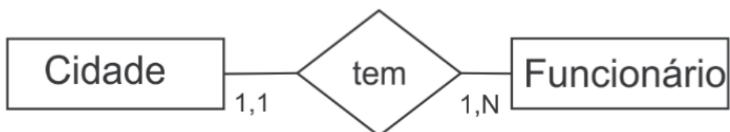
Figura 2.15 | Um-para-Muitos



Fonte: elaborada pela autora.

No relacionamento **Muitos-para-muitos (N para N)**, cada entidade, de ambos os lados, pode referenciar múltiplas ocorrências. O relacionamento resultante da cardinalidade N para N geralmente é um verbo, como atender, consultar, realizar, entre outros. Vejamos um exemplo: um aluno pode cursar vários cursos e o mesmo curso pode possuir vários alunos, como mostrado na Figura 2.16, que devemos interpretar como um Curso pode ter 0 ou N ocorrências de Alunos, e, um Aluno pode cursar 1 ou N (vários) Cursos.

Figura 2.16 | Muitos-para-Muitos

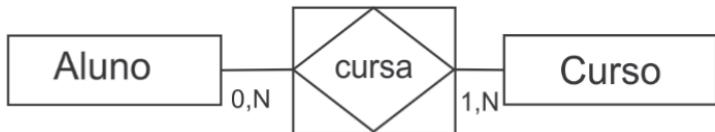


Fonte: elaborada pela autora.

Toda vez que possuímos um relacionamento N para N (na cardinalidade máxima), devemos “quebrar” o relacionamento e inserir uma tabela associativa, dando origem a uma nova tabela (HEUSER,

2011). Na Figura 2.17, podemos ver que do relacionamento entre as tabelas Aluno e Curso surge uma nova tabela, a figura do losango agora recebe um retângulo, indicando que é uma tabela associativa.

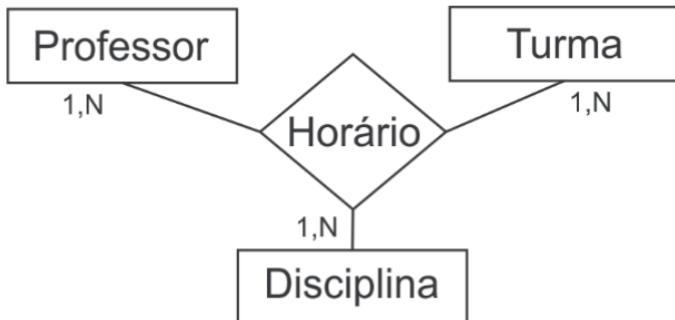
Figura 2.17 | Muitos-para-Muitos com Tabela Associativa



Fonte: elaborada pela autora.

No relacionamento ternário, três tabelas estão interligadas por uma tabela associativa. Na Figura 2.18, podemos observar um exemplo bem comum: um professor leciona para várias turmas e leciona várias disciplinas, tanto para turmas diferentes como para a mesma turma também. Esse tipo de relacionamento é utilizado para gerar o horário do professor. Nesse caso, a tabela associativa Horário é uma tabela que depende das três tabelas: Professor, Turma e Disciplina.

Figura 2.18 | Relacionamento Ternário



Fonte: elaborada pela autora.

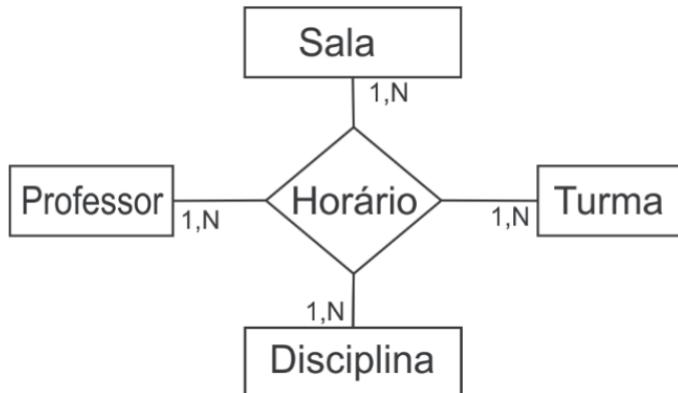


Refletia

Na Figura 2.18, podemos analisar o relacionamento entre três tabelas: Professor, Turma e Disciplina. A tabela Horário é uma tabela associativa. Quais campos essa tabela poderia ter? Lembre-se que os campos devem ser comuns entre as tabelas: Professor, Turma e Disciplina.

No relacionamento quaternário, quatro entidades são envolvidas, apoiadas pela tabela associativa. Na Figura 2.19, observamos a tabela Horário dependendo de quatro tabelas: Professor, Turma, Disciplina e Sala.

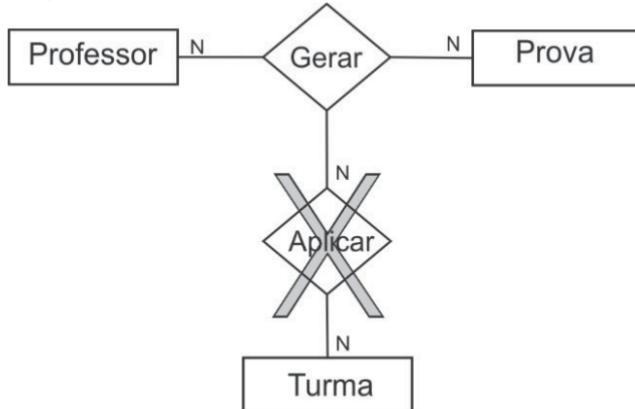
Figura 2.19 | Relacionamento Quaternário



Fonte: elaborada pela autora.

Suponha que um professor precise gerar uma prova e que uma prova pode ser gerada por mais de um professor (em tabelas, ficaria: Professor-Gerar-Prova). Além de gerar a prova, o professor precisará aplicar a prova gerada para uma determinada turma. A tabela Aplicar, nessa situação, ficará relacionada com a tabela Gerar, que representa a prova gerada, conforme aparece na Figura 2.20.

Figura 2.20 | Relacionamento Errado entre Tabelas Associativas



Fonte: elaborada pela autora.

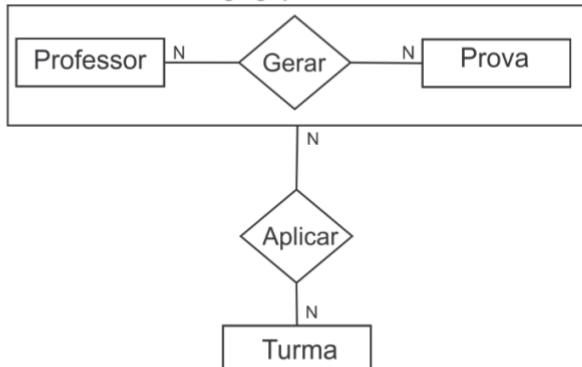


Na Figura 2.20, observe que, nos relacionamentos, somente aparece o N de muitos e no outro lado não há nada. Sempre que em um relacionamento não for inserido um valor, significa que a cardinalidade é 1. Caso você se esqueça de inserir a cardinalidade em um relacionamento, você estará indicando que o relacionamento entre as tabelas é de 1 para 1.

Uma restrição do Diagrama Entidade-Relacionamentos é a proibição de se relacionar duas tabelas associativas ou relacionamentos entre relacionamentos. Na Figura 2.20, observe que as duas tabelas associativas, Gerar e Aplicar, estão relacionadas. O relacionamento está incorreto, pois não podemos relacionar duas tabelas associativas.

As entidades e os relacionamentos, em alguns casos, podem ser agregados de forma a facilitar a compreensão. A estrutura de agregação, segundo Cougo (1999), é um recurso que pode ser aplicado aos modelos relacionais para facilitar o entendimento semântico e tornar mais claros os graus de relacionamentos ternários ou de maior número. Na Figura 2.21, a agregação é representada por um retângulo ao redor das tabelas e da relação envolvida. Portanto, a agregação ocorre quando um conjunto de tabelas e relacionamentos se comportam como se fossem uma única tabela, podendo assim se relacionar com outras tabelas.

Figura 2.21 | Relacionamento de Agregação



Fonte: elaborada pela autora.

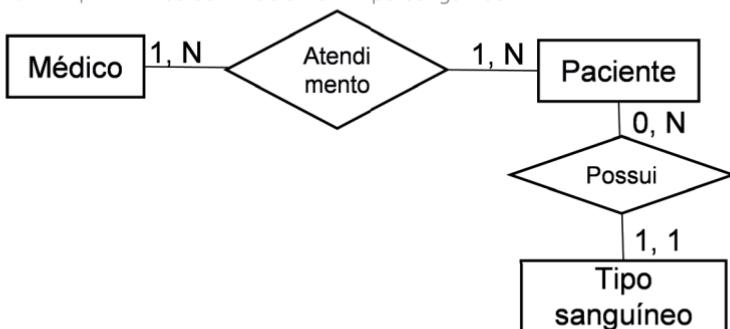


O artigo *Um roteiro para facilitar o ensino e o aprendizado na elaboração de projetos conceituais de bancos de dados* propõe um roteiro para a elaboração de modelos conceituais de banco de dados, abordando a busca por tabelas e estabelecendo relacionamentos entre as tabelas.

PRATA, J. F. Um roteiro para facilitar o ensino e o aprendizado na elaboração de projetos conceituais de bancos de dados. *Exacta*, São Paulo, v. 4, n. 1, p. 113-121, jan./jun. 2006. Disponível em: <<http://www.redalyc.org/pdf/810/81040111.pdf>>. Acesso em: 30 abr. 2018.

Para determinar a cardinalidade entre as tabelas, é necessário muita atenção. Perguntamos sempre se existe a possibilidade da repetição, e se existir será N, mas não havendo possibilidade de repetição, será 1. Por exemplo, um médico pode atender mais de um paciente? Claro que sim! E o paciente? Pode ser atendido por mais de um médico? Sim, e em várias ocasiões. Fica claro que o relacionamento entre as tabelas Médico e Paciente terá a cardinalidade de N para N. Agora, se o paciente precisar informar o seu tipo sanguíneo? Ele não poderá informar mais de um tipo, mas esse mesmo tipo sanguíneo pode repetir em outros pacientes? Com certeza! Pronto, teremos um relacionamento N para 1 entre as tabelas Paciente e Tipo Sanguíneo. Observe na Figura 2.22 com fica o Diagrama de Entidade – Relacionamentos (DER) dessa situação descrita: Médico, Paciente e Tipo Sanguíneo.

Figura 2.22 | DER: Médico – Paciente – Tipo Sanguíneo



Fonte: elaborada pela autora.

A definição da cardinalidade é de fundamental importância para o sucesso do banco de dados. Após realizar uma modelagem de um banco de dados, considere sempre mostrar o diagrama para seus colegas. Nessa hora, a opinião dos colegas pode levantar problemas que na hora de modelar foram esquecidos. E, claro, o cliente deverá sempre ser consultado em casos que a cardinalidade for muito duvidosa.

Sem medo de errar

Com a missão de criar um Diagrama de Entidade-Relacionamentos para a oficina mecânica do Sr. Ruddy, você precisará rever os requisitos que foram levantados na última seção e que foram os seguintes:

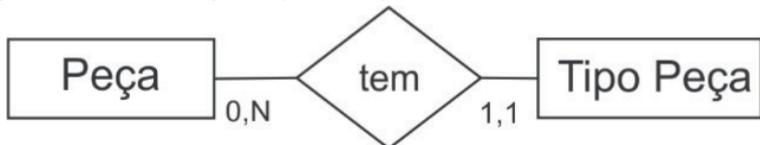
- Cadastrar os clientes, seus carros e as peças (de alto giro).
- Cadastrar os funcionários, e, a cada serviço, verificar quem o executou, o que usou e quanto tempo levou.
- Não há agendamentos prévios, e o atendimento é feito por ordem de chegada, mas há uma lista de clientes VIP que possuem prioridade no atendimento.

Para a criação do Diagrama de Entidade-Relacionamentos, foram ainda levantados mais alguns requisitos, que são: classificar as peças por seus tipos e criar um controle da ordem de serviço do conserto do automóvel.

- Quais são as novas tabelas que irão surgir? As tabelas Tipo de Peça e Ordem de Serviço;
- Como ter certeza de que os relacionamentos e cardinalidades propostos estão corretos? Deveremos criar o modelo e validar o esquema do modelo relacional com o cliente.

Para solucionar essa situação problema, primeiramente vamos ver as duas tabelas: Peças e Tipos de Peça. Uma peça do estoque deverá ser classificada em um Tipo de Peça. Um Tipo de Peça por sua vez, pode ter nenhuma peça no estoque ou muitas peças no estoque, como mostra a Figura 2.23.

Figura 2.23 | Tabelas Peças e Tipos de Peça



Fonte: elaborada pela autora.

Agora é com você! Continue e complete o diagrama. Encontre as entidades fortes e procure nos requisitos por substantivos fortes, que quase sempre são futuras entidades. Algumas entidades já apareceram em seu modelo conceitual: Cliente, Funcionário, Automóvel e Atendimento. Você deverá relacionar as tabelas e determinar as cardinalidades. Crie o uma apresentação para o seu Diagrama de Entidade-Relacionamento mostre para seu cliente, para que ele possa validar o diagrama.

Avançando na prática

DER Casa de Repouso

Descrição da situação-problema

Com o avanço da terceira idade, estão surgindo muitas casas de repouso. As pessoas internadas precisam ter um atendimento especializado por profissionais capacitados, pois muitos deles precisam tomar diversos tipos de medicamentos e em horários específicos. Tudo precisa ser muito bem controlado. Você precisará realizar a modelagem do banco de dados da Casa de Repouso para controlar o atendimento aos idosos ali internados. Uma análise de requisitos já foi previamente realizada e foram apontados os seguintes aspectos fundamentais para o banco de dados:

- Cadastrar os idosos e seus responsáveis.
- Médicos e enfermeiros não são funcionários, devendo ser cadastrados separadamente, pois serão terceirizados.
- Toda vez que um idoso precisar de atendimento médico, sempre haverá um enfermeiro para ajudar.

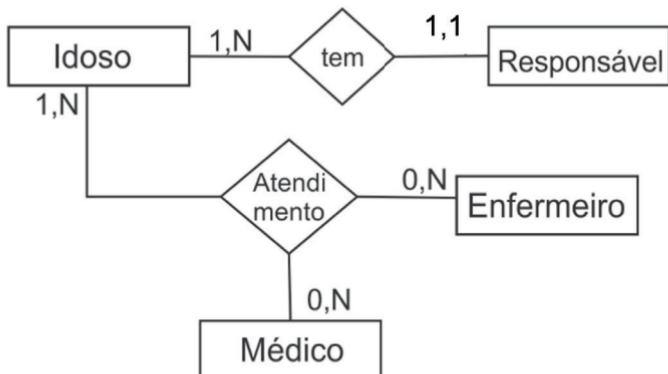
Crie o Diagrama de Entidade-Relacionamentos, seus relacionamentos e aplique as cardinalidades.

Resolução da situação-problema

Para resolver a situação problema devemos, primeiramente, destacar as tabelas que são facilmente identificadas, nos requisitos apresentados: Idoso, Responsável, Médico, Enfermeiro, e Atendimento e as cardinalidades no diagrama.

Uma sugestão de solução pode ser observada na Figura 2.24, em que o diagrama mostra que cada idoso terá um responsável. O responsável, por sua vez, poderá ter mais de um idoso sob sua responsabilidade. No atendimento, teremos as tabelas: Idoso, Médico e Enfermeiro envolvidos.

Figura 2.24 | DER Casa de Repouso



Fonte: elaborada pela autora.

Faça valer a pena

1. As entidades, também conhecidas como tabelas, possuem características próprias, que podem variar na quantidade de características conforme a necessidade de cada sistema. Representam objetos reais ou abstratos e podem possuir diversos atributos de diferentes tipos.

Considerando o conceito de entidade, marque a alternativa correta referente à classificação das entidades em um modelo relacional.

- a) Entidade aguda, entidade inerte, entidade disjuntiva.
- b) Entidade firme, entidade fraca, entidade cooperativa.
- c) Entidade forte, entidade simples, entidade conjuntiva.
- d) Entidade forte, entidade fraca, entidade associativa.
- e) Entidade composta, entidade simples, entidade subjuntiva.

2. Segundo Korth, Silberschatz e Sudarshan (2012), um banco de dados é uma coleção de dados inter-relacionados, representando informações sobre um domínio específico. Um banco de dados possui diversos componentes, entre eles podemos citar:

- I. As tabelas, que são compostas de diversos registros.
- II. Os registros, que são compostos de diversos campos.
- III. Os campos, que podem ser classificados como categorias de informação.
- IV. Os dados, que são os itens armazenados no banco de dados.

Analise as afirmativas e marque a alternativa correta referente ao banco de dados:

- a) Somente a assertiva IV está correta.
- b) Somente as assertivas I e III estão corretas.
- c) Somente as assertivas II e IV estão corretas.
- d) Somente as assertivas II, III e IV estão corretas.
- e) Todas as assertivas estão corretas.

3. A relação entre tabelas se dá através da cardinalidade. A cardinalidade é um número que expressa o comportamento (número de ocorrências) de determinada entidade associada a uma ocorrência da entidade em questão por meio do relacionamento.

Sobre os relacionamentos entre tabelas, é correto afirmar que:

- a) Todo relacionamento entre tabelas é sempre "N para N".
- b) Todo relacionamento entre tabelas é sempre "1 para 1".
- c) Cada relacionamento deve ser analisado para que seja possível determinar a sua cardinalidade.
- d) Somente o usuário final do banco de dados pode determinar o relacionamento entre uma tabela.
- e) Não há diferenças entre os relacionamentos "1 para N" ou "N para N" e, como não há consequências no banco de dados, é melhor sempre deixar N para N.

Seção 2.3

Diagrama de Entidade-Relacionamento (DER)

Diálogo aberto

Participar do processo de modelagem de banco de dados é muito enriquecedor, pois agrega experiência à vida profissional. Um fator importante é a coleta de dados com o cliente, portanto, o analista deverá ir até a empresa, conversar e buscar soluções, uma tarefa bastante desafiadora. O mais importante é sempre escutar o cliente e ser muito observador, para conseguir captar informações que não foram ditas. Um aspecto que devemos sempre levar em conta é a possibilidade de ocorrem mudanças no futuro. Hoje o cliente não quer algo, mas, e se ele mudar de opinião? Seu projeto estará preparado para adaptações? Para responder a essas questões, devemos seguir as regras da modelagem, que existem justamente para possibilitar futuros ajustes. Um banco de dados pode ser modelado de diversas maneiras, inclusive não respeitando as regras que aqui serão mostradas. O impacto de uma modelagem sem seguir os padrões estabelecidos aparecerá justamente quando o cliente precisar consultar ou extrair informações do banco de dados, ou adicionar requisitos, afetando a performance do banco de dados e, provavelmente, gerando retrabalho.

Como analista de sistemas, você estará trabalhando no banco de dados da oficina mecânica do Sr. Ruddy. Precisamos relembrar as entidades que foram encontradas nas duas últimas seções, quais sejam: clientes, peças, tipos de peças, funcionários, serviços realizados e ordem de serviço. Deverá ser criado o modelo entidade-relacionamentos de forma gráfica e textual, e o modelo deverá responder às seguintes questões:

- Quais serão os principais campos de cada tabela?
- Quais as chaves de cada tabela?
- Será mesmo necessário utilizar a chave estrangeira?
- Haverá alguma tabela associativa?

Para responder a essas perguntas e criar o modelo lógico do banco de dados da oficina, leia atentamente os conceitos dessa seção. Lembre-se que esse modelo será utilizado para a próxima etapa de desenvolvimento e também será apresentado ao cliente.

Como desafio final, resgate todos os diagramas criados para a oficina mecânica. Modifique o Modelo Entidade-Relacionamento (MER), levando em consideração a criação de uma agenda de horários (para os atendimentos com hora marcada) e a geração de nota fiscal dos serviços realizados e peças utilizadas. Temos um grande desafio. Vamos lá?

Boa aula!

Não pode faltar

O modelo de dados relacional é amplamente utilizado no projeto e no desenvolvimento de banco de dados para os mais diversos tipos de software. Devido a sua simplicidade e a sua facilidade de aprendizado, esse modelo se difundiu entre os desenvolvedores de software.

O Diagrama de Entidade-Relacionamento tem como objetivo a preparação para a implementação física do banco de dados no Sistema Gerenciador de Banco de Dados (SGBD). A fase de modelagem física requer a aplicação de comandos para a criação das tabelas e campos, e depende diretamente da modelagem lógica. É de fundamental importância ter o projeto lógico o mais correto possível para a criação física do banco de dados. Imagine a seguinte situação na construção de um edifício: um construtor vai contratando pedreiros e aumentando os andares do prédio conforme surgem as ideias. Agora, imagine se os pedreiros chegassem na construção do quinto andar e alguém quisesse mudar o local do elevador? Teriam que refazer tudo o que construíram? Teriam que improvisar? Sabemos que não é assim que um edifício é construído. São necessárias várias plantas do edifício para que o prédio possa começar a sair do papel. A analogia da construção de um banco de dados com a construção de um edifício é perfeita no sentido de planejamento e criação de um projeto antes da sua efetiva construção.

O modelo de dados lógico de um banco de dados possui as seguintes características:

- Todas as tabelas e os relacionamentos entre elas.
- Descrição de todos os atributos de cada tabela.
- Identificação de um atributo chave para cada tabela.
- Determinação de relacionamentos por meio de chaves.

O modelo relacional trabalha com esquemas compostos de tabelas. Para que uma tabela realmente exista, é necessário que possua diversas propriedades, que nada mais são do que os seus atributos ou campos. Cada campo, por sua vez, possui uma descrição de seu tipo de dado.

Suponha que seja necessário guardar informações sobre um cliente de uma loja. O primeiro questionamento a ser feito é: existem informações que podem ser guardadas sobre um determinado cliente? Certamente. Podemos citar: nome, CPF, RG, data de nascimento, cidade natal. Essas informações cadastradas sobre o cliente serão armazenadas em uma tabela chamada *Cliente*. Observe, na Figura 2.25, como algumas informações cadastradas na tabela ficariam armazenadas.

Figura 2.25 | Tabela Cliente

Tabela: Cliente

Nome	RG	CPF	DT Nasc	Cidade
André Marco	7555333	77799944411	13/08/198	Curitiba
Jonny Lucca	1222333	44455566622	28/02/199	São Paulo
Lia Leme	6333000	99977766644	09/03/198	Curitiba

Fonte: elaborada pela autora.

Korth, Silberschatz e Sudarshan (2012) afirmam que, no modelo relacional, o termo relação é designado para se referir a uma tabela, o termo *tupla* refere-se a uma determinada linha da tabela e como o atributo é conhecido como a coluna de uma tabela. O termo “instância de relação” é designado a um determinado conjunto de

tuplas ou a um determinado número de registros (algumas linhas selecionadas de uma tabela). Na Figura 2.25, os três registros que aparecem na tabela Cliente podem ser considerados uma instância de uma relação e, para mostrar a tabela Cliente, foram selecionados esses três exemplos pois, certamente, uma tabela de clientes contém centenas e até milhares de registros.



Segundo Coronel e Rob (2011), em banco de dados, a palavra relação também é conhecida como um *Dataset*. O termo relação é fundamentado na teoria da matemática dos conjuntos e está se referindo à tabela e não aos relacionamentos, como muitos podem “achar” devido à semelhança da escrita entre as palavras.

Agora devemos pensar na seguinte situação: em um determinado banco de dados existe uma tabela de clientes com centenas de registros cadastrados. Será que existe a possibilidade de procurar um determinado cliente pelo seu nome e aparecer mais de um registro como resultado? Há possibilidades de duplicidade de nomes em uma tabela? Se a possibilidade de repetição sempre existe, como garantir que o cliente certo seja encontrado?

Para solucionar esse dilema, no banco de dados relacional, há a necessidade de se estabelecer um ou mais campos para serem uma chave de identificação do registro armazenado. Alguns valores do registro até poderão ser repetidos, como duas ou mais pessoas com exatamente o mesmo nome, mas, obrigatoriamente, deve haver um campo na tabela que nunca se repete. Esse campo será a **chave da tabela**.

Coronel e Rob (2011) explicam que uma chave consiste em um ou mais atributos que determinam a existência de outros atributos. Elas são utilizadas para estabelecer os relacionamentos entre as tabelas e estabelecer a integridade referencial dos dados. Os tipos de chaves em uma tabela podem ser: Chave Primária; Chave Concatenada ou Composta; Chave Substituta ou Surrogada; Chave Secundária; ou Chave Estrangeira.

Um dos fundamentos primordiais de um banco de dados é que em cada tabela exista uma chave primária. A chave primária também

é conhecida como *Primary Key* (PK). Devemos escolher um dos campos da tabela para ser a chave primária. Na Figura 2.25 foi escolhido o campo CPF como chave primária, pois temos certeza que uma pessoa não pode ter o CPF igual ao de outra pessoa. O campo nome seria uma péssima escolha, visto que é comum termos pessoas com nomes e sobrenomes exatamente iguais. O campo data de nascimento também é uma péssima escolha, pela sua possibilidade de repetição. A cidade de nascimento é uma escolha muito ruim, no próprio exemplo da Figura 2.26 há a repetição da cidade Curitiba.

Figura 2.26 | Chave Primária

Tabela: Cliente

Nome	RG	CPF	DT Nasc	Cidade
André Marco	7555333	77799944411	13/08/198	Curitiba
Jonny Lucca	1222333	44455566622	28/02/199	São Paulo
Lia Leme	6333000	99977766644	09/03/198	Curitiba



Chave Primária

Fonte: elaborada pela autora.

Na forma textual, a chave primária deve sempre ficar em evidência, em negrito, sublinhada e com sinal # na sua frente. Observe como ficaria a tabela Cliente da Figura 2.26: Cliente (**#CPF**, Nome, RG, Dt Nasc, Cidade). A ordem dos campos não prejudica em nada os campos das tabelas, porém, para fins didáticos e para facilitar a identificação, nos exemplos desse livro, a chave primária ficará sempre como o primeiro campo da tabela.

A Figura 2.27 mostra o uso do RG, seria esse um bom campo para usar como chave primária? Afinal ele também não repete, certo? Digamos que isso é questionável. O documento do RG é emitido em órgãos diferentes em diferentes estados. Existe a possibilidade de repetição do número do RG? Sim, existe! Caso

quiséssemos realmente usar o RG, deveríamos solicitar também o órgão expedidor do documento, essas duas informações em conjunto não se repetem, e é justamente esse o conceito de chave concatenada ou composta. Um ou mais campos que juntos não se repetem.

Figura 2.27 | Chave Composta ou Concatenada

Tabela: Cliente

Nome	RG	Órgão Expedidor	CPF	DT Nasc
André Marco	7555333	SSP-PR	77799944411	13/08/19
Jonny Lucca	1222333	SSP-SP	4445556662	28/02/19
Lia Leme	6333000	SSP-PR	9997776664	09/03/19


Chave Composta

Fonte: elaborada pela autora.

A forma textual da tabela Cliente demonstrada na Figura 2.27 ficaria da seguinte forma: Cliente (**#RG, # Órgão Expedidor**, Nome, CPF, DT Nasc). Podemos utilizar vários campos que juntos não irão se repetir, mas o ideal é que não ultrapassem quatro campos.

A chave substituta ou surrogada, também conhecida como *Surrogate Key* é uma chave primária criada exclusivamente para impedir que os registros da tabela venham a se repetir e que não tenham um campo como a chave primária. Esse tipo de chave (Figura 2.28) é um campo criado de valor inteiro, que tem um auto incremento. A cada registro adicionado na tabela, o campo recebe um incremento e assim sucessivamente (o próprio SGBD se encarrega do incremento). Esse tipo de campo nunca pode ser alterado, bem como nunca pode ser reutilizado em casos em que o registro for apagado, não havendo como reutilizar o valor de uma chave substituta. Veja a forma textual da tabela Encomenda da Figura 2.28: Encomenda (**#Código**, Encomenda, Quantidade, Preço Unitário).

Figura 2.28 | Chave Surrogada ou Substituta

Tabela: Encomenda

Código	Encomenda	Quantidade	Preço Unitário
1	Rosas Vermelhas	12	R\$ 3,50
2	Chocolate ao Leite	2	R\$ 8,00
3	Urso de Pelúcia	1	R\$ 23,00
4	Cartão Florido	1	R\$ 9,80

 Chave Surrogada

Fonte: elaborada pela autora.

De acordo com Coronel e Rob (2011), a chave secundária é uma chave utilizada para fins de recuperação de informação. É uma chave que auxilia na recuperação de um registro. Por exemplo, digamos que um paciente de um consultório médico tenha o CPF como chave primária e, ao chegar no consultório, ele perceba que esqueceu o documento e também não sabe o número, como encontrar o registro do paciente? Nesse caso, podemos usar algum mecanismo de busca, utilizando o campo do sobrenome ou de data de nascimento como chaves secundárias, para realizar uma pesquisa no banco de dados. Devem aparecer vários registros, mas ficaria mais fácil encontrar o registro do paciente.

Korth, Silberschatz e Sudarshan (2012) descrevem a chave estrangeira, também conhecida como *Foreign Key* (FK), como uma chave primária de outra tabela. É por meio dessa chave que ocorrem os relacionamentos entre as tabelas de um banco de dados. Na Figura 2.26 da tabela Cliente, temos um campo chamado Cidade. Observe que foram cadastradas duas vezes a cidade de Curitiba. Em um banco de dados, normalmente, é necessário informar a cidade para várias tabelas: funcionário, cliente, aluno, fornecedor. Todos precisam informar a cidade em que nasceram ou em que estão estabelecidos. Para evitar essas redundâncias, criamos uma tabela Cidade e a relacionamos com a tabela Cliente, sendo que um

Cliente somente poderá cadastrar uma única Cidade, mas a Cidade pode ser relacionada por vários Clientes, como mostra a Figura 2.29.

Figura 2.29 | Relacionamento Cliente e Cidade



Fonte: elaborada pela autora.

O relacionamento 1 para N é estabelecido pelo uso da chave estrangeira. Coronel e Rob (2011, p.75) dizem que uma chave estrangeira de uma tabela (do lado N) é sempre uma chave primária de outra tabela (do lado 1), e que seus valores deverão coincidir com a sua chave primária ou serem valores nulos, estabelecendo, desta forma, a integridade referencial.

Figura 2.30 | Chave Estrangeira

Tabela: Cidade

CodCidade	Cidade
1	Curitiba
2	São Paulo
3	Campinas
4	Blumenau

Tabela: Cliente

CPF	Nome	DT Nasc	CodCidade
77799944411	André Marco	13/08/1980	1
44455566622	Jonny Lucca	28/02/1999	2
99977766644	Lia Leme	09/03/1987	1

Fonte: elaborada pela autora.

Com o relacionamento entre as tabelas Cliente e Cidade, conforme a Figura 2.29, teremos a seguinte forma textual entre as duas tabelas:

Cidade (#CodCidade, Cidade)

Cliente (#CPF, Nome, DT Nasc, &CodCidade)



Exemplificando

Observe que para identificar a chave estrangeira devemos utilizar o caractere **&** na frente do campo. Em uma tabela, podemos ter diversas chaves estrangeiras, e o mais importante, ela poderá se repetir uma infinidade de vezes, conforme a necessidade. Por exemplo, podemos cadastrar vários clientes e cidades e, no relacionamento cliente-cidade, o código da cidade aparecerá em vários clientes.

Segundo Coronel e Rob (2011), a integridade referencial tem como exigência básica a sua existência em outra tabela como chave primária. Estabelecer a integridade referencial é justamente garantir que, ao relacionar uma tabela com outra, haverá a garantia de que a chave estrangeira foi cadastrada antes como chave primária de outra tabela que compõe o relacionamento. E, para estabelecer a integridade referencial, precisamos seguir os seguintes passos:

- 1º Passo: observar no diagrama os relacionamentos. Procure por cardinalidades do tipo N nas tabelas.
- 2º Passo: existe uma ou mais cardinalidades do tipo N? Caso a resposta seja afirmativa, então haverá chaves estrangeiras. Pode haver vários Ns nas tabelas e, consequentemente, várias chaves estrangeiras.
- 3º Passo: a tabela do lado 1 deverá receber novos campos, para criar o relacionamento. Insira a chave primária da tabela correspondente ao relacionamento do lado N.



Refita

A integridade referencial em um banco de dados relacional é, na verdade, uma restrição (uma regra) para que dados incorretos não entrem no banco de dados, por exemplo, evitar que o cliente insira

uma cidade inexistente. Mas como isso realmente acontece? E quais são as chances de o usuário cadastrar uma cidade errada no banco de dados? O que podemos fazer para evitar que cidades inexistentes sejam cadastradas, provocando erros futuros?

Para representar as cardinalidades no modelo gráfico de um Diagrama de Entidade-Relacionamento, podemos utilizar diversas notações. Uma notação gráfica é a forma como algo é criado, desenhado ou projetado; são os padrões que deverão ser adotados no desenvolvimento de algo. Cougo (1997) descreve a notação "Pé-de-Galinha", ou *Crow's Foot* (do inglês, pata de corvo), criada por James Martin. Rob e Coronel (2011) apontam a notação de Bachman, desenvolvida por Charles William Bachman, que, após algumas modificações, se transformou na notação de Setas. A notação de Peter Chen é amplamente utilizada nos livros de banco de dados (assim como neste livro) para explicar os relacionamentos entre tabelas. Na Figura 2.31, podemos observar um relacionamento 1 para N sendo representado em três notações diferentes.

Figura 2.31 | Relacionamento 1 para N com notação diferente

Notação de Peter Chen



Notação de Pé-de-Galinha



Notação de Bachman



Fonte: elaborada pela autora.

A notação de Bachman é utilizada em muitos softwares para a criação de modelos lógicos de um banco de dados. Esse modelo sofreu algumas alterações e se transformou na notação de Setas, adotada por vários softwares. Observe-a no Quadro 2.1.

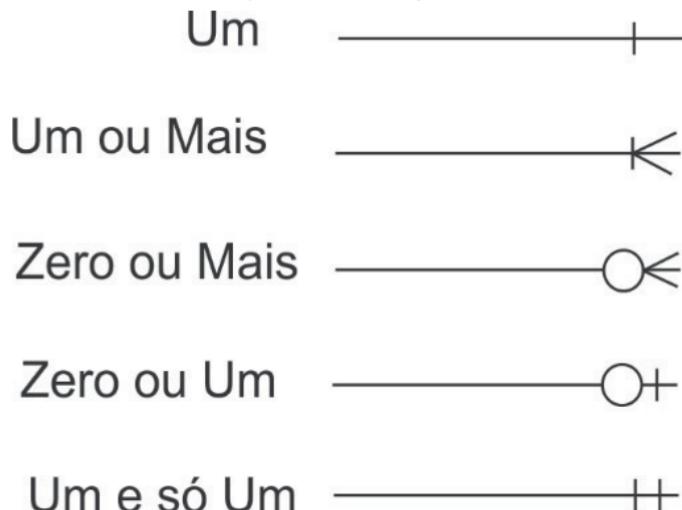
Quadro 2.1 | Notação de Bachman vs Notação de Setas

Cardinalidade	Notação Original de Bachman	Notação de Setas
1:1	—	↔
1:N	→	↔→
N:1	←	←→
M:N	↔	↔→

Fonte: elaborada pela autora.

A notação de James Martin e seu famoso diagrama de “Pé-de-Galinha” também são muito populares entre as ferramentas de criação de modelos gráficos de esquemas de banco de dados. Na Figura 2.32, podemos observar as notações para a definição das cardinalidades.

Figura 2.32 | Relacionamento 1 para N com notação diferente



Fonte: elaborada pela autora.

Observe que o termo “Pé-de-Galinha” vem justamente do relacionamento do lado N que aparenta um pé-de-galinha. É

importante destacar que uma notação pode ser difundida por meio de um software. No caso do SGBD Microsoft Access, por exemplo, o lado N sempre é representado pelo sinal do infinito. Existem diversas notações que podem ser encontradas e utilizadas para demonstrar as cardinalidades entre as tabelas, tais como: UML, IDEF e OMT.



Pesquise mais

A modelagem de dados deste livro está abordando o modelo relacional, mas existem diversas outras abordagens que podem ou não virar tendência de mercado. Acesse a dissertação abaixo, que apresenta uma proposta de mapeamento de bancos de dados para um domínio semântico, e leia as páginas 20 a 30 e conheça um pouco mais sobre os modelos de banco de dados.

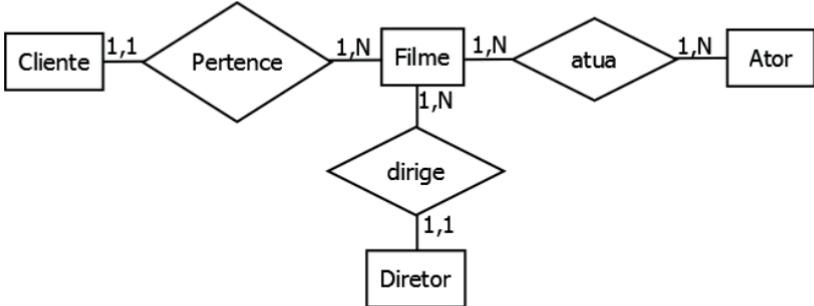
CRUZ, J. A. G. C. **Mapeamento de Bancos de Dados para Domínios Semânticos.** 2015. Dissertação (Mestrado em Ciências da Computação) - Instituto de Informática, Universidade Federal de Goiás, Goiânia, GO. Disponível em: <<http://repositorio.bc.ufg.br/tede/handle/tede/4639>>. Acesso em: 10 maio 2018.

Para exemplificar a criação de um modelo lógico de um banco de dados, iremos resolver um estudo de caso para a elaboração do Diagrama de Entidade-Relacionamento para um pequeno estúdio de gravação de vídeos publicitários. Com a demanda de clientes aumentando, o processo de produção dos vídeos precisa ser melhor controlado. Após algumas visitas realizadas, os seguintes requisitos para a elaboração do modelo conceitual do banco de dados foram elencados:

- Será necessário cadastrar os dados sobre o filme e os atores.
- Cada filme deverá ter somente um diretor.
- O filme sempre pertencerá a um único cliente.

Na Figura 2.33, um modelo conceitual foi criado para ser apresentado ao cliente.

Figura 2.33 | Modelo Conceitual do Estudo de Caso



Fonte: elaborada pela autora.



Refletia

Na figura 2.33, o Diagrama de Entidade-Relacionamento foi criado com base nos requisitos apresentados pelo cliente. Nas tabelas associativas, foram inseridas as chaves estrangeiras. Poderíamos usar as chaves estrangeiras para criar uma chave primária? Como poderíamos usar uma chave concatenada?

Após a apresentação do modelo, o cliente ainda solicitou os seguintes requisitos:

- Controlar os locais onde foi gravado o filme publicitário.
- As informações sobre o cliente devem ser guardadas e cada filme pertence somente a um cliente.
- Cada filme sempre é sobre somente um produto alvo.
- Será necessário controlar os atores que atuaram nos filmes, sendo que o ator pode trabalhar no mesmo filme, atuando como um personagem diferente no próprio filme.

Como resultado, foi elaborado o modelo lógico do banco de dados (Figura 2.34). Observe, na forma textual, os campos das tabelas e suas respectivas chaves:

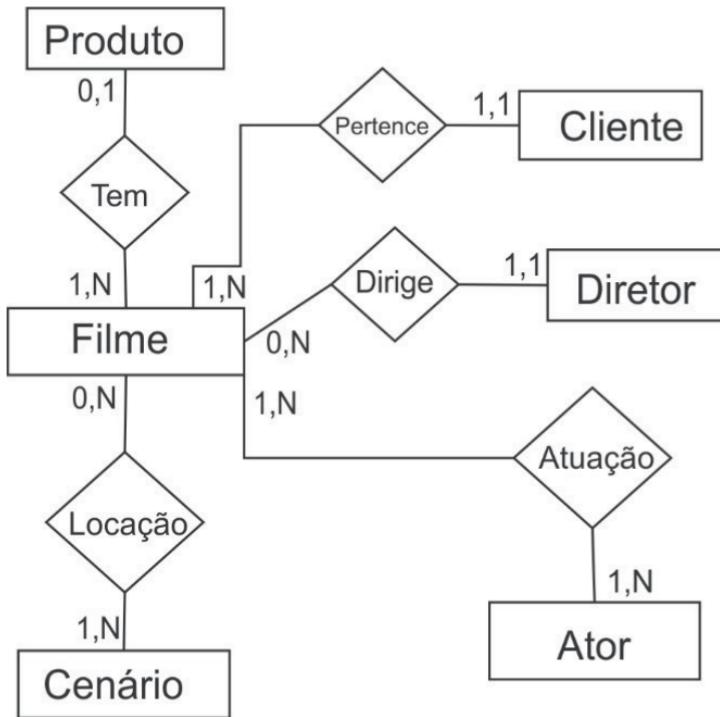
Filme (#IdFilme, nome, duração, dt_início, dt_fim, &IdCliente, &IdDiretor, &CodProduto)

Cliente (#IdCliente, nome, endereço)

Produto (#CodProduto, nome)

Diretor (#IdDiretor, nome)
 Ator (#CodAtor, nome, foto, sexo, dt_nasc, cachê)
 Cenário (#IdCenário, Local, descrição)
 Locação (#IdLocação, &IdFilme, &IdCenário, dt_início, dt_fim, valor_gasto)
 Atuação (#CodAtuação, &CodAtor, &IdFilme, personagem)

Figura 2.34 | Modelo de Entidade-Relacionamento do Estudo de Caso



Fonte: elaborada pela autora.



Assimile

Os passos para a elaboração do Modelo de Entidade-Relacionamento devem ser: 1) identificar as tabelas; 2) identificar os relacionamentos e as cardinalidades; 3) nos relacionamentos N para N, criar a tabela associativa; 4) criar o modelo textual com os campos de cada tabela; 5) achar as chaves primárias; e 6) inserir as chaves estrangeiras das tabelas que estão relacionadas e possuem o N.

Esta unidade trouxe subsídios para o início do processo de modelagem de dados. Aprendemos sobre os modelos conceitual e físico de bancos de dados para conseguir criar os Modelos de Entidade-Relacionamentos. A prática é a amiga da perfeição, portanto, quanto mais modelagens você fizer, seus diagramas serão cada vez melhores. Na próxima unidade, demonstraremos como melhorar os diagramas, com mais técnicas e estudos de casos. Lembre-se de que esta é uma profissão que precisa de estudo e atualização constantes.

Sem medo de errar

Você aprendeu nesta unidade o processo de modelagem de dados para realizar o modelo conceitual e o modelo lógico de um banco de dados. Reconhecer uma tabela, estabelecer os relacionamentos e aplicar as cardinalidades são, sem sombra de dúvida, passos importantes para a criação do banco de dados de qualquer aplicação.

No desafio desta unidade, você conheceu os requisitos para realizar a modelagem de um banco de dados para uma oficina mecânica de pequeno porte. Foi necessário encontrar as tabelas dos clientes, de peças, de tipos de peças, de funcionários, de serviços realizados e de ordem de serviço. Agora devemos encontrar os campos e as chaves de cada tabela. Deverá ser criado o Modelo de Entidade-Relacionamento de forma gráfica e textual, onde o modelo deverá responder às seguintes questões: quais serão os principais campos de cada tabela? Quais são as chaves de cada tabela? Será mesmo necessário utilizar a chave estrangeira? Haverá alguma tabela associativa?

Respondendo às questões acima:

- Toda tabela deverá ter mais de um campo, tais como: nome, quantidade, preço, endereço, etc.
- Toda tabela deve possuir uma chave primária. Primeiramente, procure por um campo que possa servir como identificador único. Você poderá criar um campo novo e denominar o campo como código (Cod) ou identificador (Id). Observe o exemplo: Cliente (#codCliente, Nome, [...]). Procure sempre deixar o nome da tabela

atrás da chave primária, isso é uma boa prática de modelagem e ajudará no futuro.

- Observe o relacionamento entre as tabelas: a tabela tem N? Se sim, haverá chave estrangeira nessa tabela. Tem mais de um N? Se sim, haverá mais de uma chave estrangeira na tabela.

Para exemplificar o uso da chave estrangeira, vamos analisar as tabelas Peça e Tipo de Peça. Uma Peça é classificada por um Tipo de Peça e um Tipo de Peça poderá ter várias Peças, as tabelas ficariam assim:

Tipo de Peça (#**idTipoPeça**, Tipo de Peça)

Peça (#**codPeça**, NomePeça, ..., &IdTipoPeça) → observe que inserimos “&IdTipoPeça” no lado do N (coloque o símbolo & no início dela para indicar a chave estrangeira).

Agora é com você! Para finalizar o desafio desta unidade, resgate todos os diagramas criados para a oficina mecânica. Modifique o Modelo de Entidade-Relacionamento levando em consideração a criação de uma agenda de horários (para os atendimentos com hora marcada) e a geração de nota fiscal, dos serviços realizados e das peças utilizadas. Crie uma apresentação e mostre a evolução de seus diagramas de entidade-relacionamentos.

Avançando na prática

DER - Relacionamentos, Cardinalidades e Chaves

Descrição da situação-problema

Você possui dois cenários (independentes) e deverá criar o Diagrama de Modelagem-Relacionamento, determinando os relacionamentos, as cardinalidades e as chaves de cada cenário.

1º Cenário: um país possui vários estados, e um estado possui muitas cidades. Um estado só poderá estar relacionado a um país e uma cidade só poderá estar vinculada a um estado.

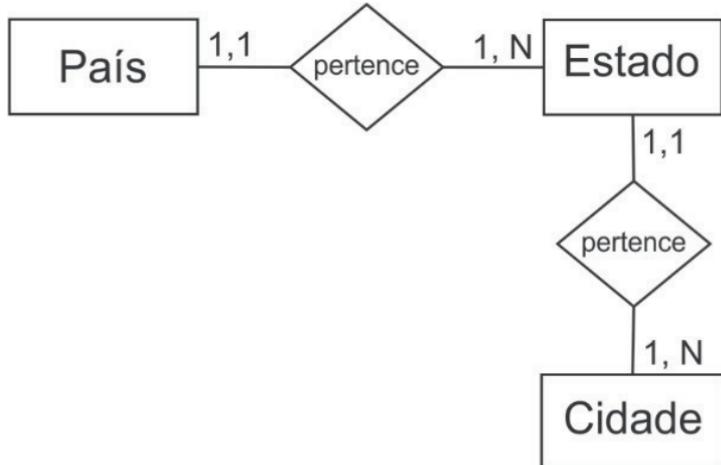
2º Cenário: um encontro de eventos de empreendedorismo pode ter muitos palestrantes e um palestrante pode palestrar em mais de um evento.

Crie o Diagrama de Entidade-Relacionamento gráfico e textual para cada cenário. Ilustre com campos em todas as tabelas. Caso haja tabela associativa, procure campos exclusivos dessa tabela.

Resolução da situação-problema

Para o 1º Cenário, devemos observar que um país poderá ter mais de um estado e o estado não poderá pertencer a outro país, sendo, assim, um relacionamento 1 para N. Um estado pode ter várias cidades, mas a cidade nunca pertencerá a outro estado, sendo este, então, um relacionamento 1 para N também. O diagrama ficará conforme a Figura 2.35.

Figura 2.35 | Modelo Entidade-Relacionamentos do 1º Cenário



Fonte: elaborada pela autora.

A forma textual ficará da seguinte forma:

País (**#IdPaís**, NomePaís, bandeira, idioma)

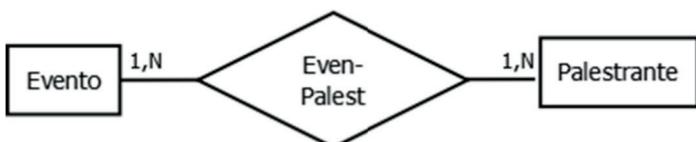
Estado (**#IdEstado**, NomeEstado, &IdPaís)

Cidade (**#IdCidade**, NomeCidade, &IdEstado)

O 2º Cenário possui o relacionamento entre eventos e palestrantes. Um evento pode ter muitos palestrantes e o palestrante pode palestrar em mais de um evento? A resposta é sim para as duas perguntas, sendo o relacionamento N para N. Lembre-se de que em

todo relacionamento N para N devemos criar a tabela associativa. Observe na Figura 2.36 o diagrama com a solução.

Figura 2.36 | Modelo de Entidade-Relacionamento do 2º Cenário



Fonte: elaborada pela autora.

Na tabela associativa, muitas vezes misturamos os nomes das duas tabelas que originaram o relacionamento. A forma textual deve ficar da seguinte forma:

Evento (#CodEvento, Evento, Data, Local)

Palestrante (#CodPalestrante, Nome, Foto, Descrição da Formação)

Even-Palest (#IdEven-Palest, &CodEvento, &CodPalestrante, data, horário, sala)

Faça valer a pena

1. Segundo Navathe e Ramez (2005), o objeto básico de um Modelo de Entidade-Relacionamento é uma entidade (ou tabela) que representa alguma coisa do mundo real e que possa ter informações para serem armazenadas. As informações, por sua vez, podem ser classificadas por categorias, sendo esses os campos da entidade ou tabela. Podemos apontar um dos campos como um campo chave da tabela.

Marque a alternativa correta, que apresenta os principais tipos de chave encontrados em um Modelo de Entidade-Relacionamento.

- a) Chave primária e chave composta.
- b) Chave composta e chave estrangeira.
- c) Chave única e chave mestra.
- d) Chave primária e chave estrangeira.
- e) Chave simplificada e chave composta.

2. A integridade _____ em um banco de dados relacional é, na verdade, uma _____ para que dados _____ não entrem no banco de dados.

Marque a alternativa que preenche corretamente as lacunas da afirmativa.

- a) substancial – obrigação - imperfeitos.
- b) comportamental – alternativa – armazenados.
- c) referencial – restrição - incorretos.
- d) estrutural – migração – incompatíveis.
- e) relacional – facilidade – destorcidos.

3. Conforme Coronel e Rob (2011), uma chave consiste em um ou mais atributos que determinam outros atributos, ou seja, é um campo que pode identificar outros campos. Existem diversos tipos de chaves.

Considerando o contexto, avalie as afirmativas a seguir e marque V (Verdadeiro) ou F (Falso):

- () Uma chave primária é obrigatória nas tabelas, mas existe a possibilidade de deixar o seu valor como nulo, inserindo o valor da chave somente nos momentos de pesquisas no banco de dados.
- () Uma chave secundária tem como objetivo o conjunto de várias chaves primárias, que juntas poderão formar uma única chave primária.
- () Uma chave estrangeira é uma chave que, obrigatoriamente, é uma chave primária em outra tabela e deverá se relacionar com a tabela que possui a chave estrangeira.

- a) V – F – V.
- b) V – V – F.
- c) F – V – V.
- d) F – F – V.
- e) F – V – F.

Referências

- ABREU, M. P.; MACHADO, F. N. R. **Projeto de banco de dados: uma visão prática.** 16. ed. rev. e atual. São Paulo: Érica, 2009.
- CODD, E. F. **The relational model for database management:** version 2. [S.l.]: Reading Addison-Wesley, 1990.
- CORONEL, C.; ROB, P. **Sistema de banco de dados:** projeto, implementação e administração. São Paulo: Cengage Learning, 2011.
- COUGO, P. **Modelagem conceitual e projeto de bancos de dados.** Rio de Janeiro: Elsevier, 1997.
- DATE, C. J. **Introdução a sistemas de bancos de dados.** Rio de Janeiro: Elsevier, 2003.
- GUIMARÃES, C. C. **Fundamentos para bancos de dados: modelagem, projeto e linguagem SQL.** Campinas: Ed. da Unicamp, 2003.
- HEUSER, C. A. **Projeto de Banco de Dados,** 6. ed. [S.l.]: Bookman, 2011.
- KORTH, H. F. SILBERSCHATZ, A. SUDARSHAN, S. **Sistema de banco de dados.** Rio de Janeiro: Elsevier, 2012.
- NAVATHE, S. B. RAMEZ, E. **Sistemas de banco de dados.** 4. ed. São Paulo: Addison Wesley, 2005.

Abordagem entidade-relacionamento

Convite ao estudo

Prezado aluno, seja bem-vindo à nossa terceira unidade.

Para ser um analista de sistemas, precisamos estar atentos às tecnologias que surgem a todo momento. Você deve ter percebido que criar os modelos no papel é o primeiro passo para fazer a modelagem, mas observe que precisamos de ferramentas apropriadas para criar esses modelos e apresentá-los aos nossos clientes e a outros analistas. Para isso, utilizamos as ferramentas CASE, que serão estudadas nesta unidade.

Sua empresa está começando a fazer sucesso e seus clientes estão aumentando. Além de ser empreendedor, você é o analista de sistemas da empresa e, para ajudar na execução das atividades, é necessário delegar tarefas. Para atender seu novo cliente, você contratará estagiários para ajudar no desenvolvimento do banco de dados e precisará estabelecer padrões no desenvolvimento dos softwares. As ferramentas descritas nesta unidade permitirão que você padronize e controle seus estagiários no desenvolvimento da modelagem do banco de dados.

Para seu novo cliente, você precisará elaborar o diagrama de entidade-relacionamento de forma completa. O software a ser modelado será para um salão de beleza de médio porte, ou seja, possui muitos clientes e atendimento personalizado para clientes de alto padrão.

Você deverá mapear todas as entidades e gerar uma documentação do modelo que foi criado. Dando prosseguimento, serão utilizados os conceitos de UML. Vamos resgatar o modelo criado das entidades e usar a modelagem UML para ajustes na modelagem do banco de dados do salão

de beleza. No último passo, você utilizará uma ferramenta CASE e, então, criará o diagrama de entidade-relacionamento do salão de beleza, para poder apresentar ao cliente.

Bons estudos!

Seção 3.1

Modelagem de dados através do modelo entidade-relacionamento usando DER

Diálogo aberto

As novas tecnologias surgem a todo instante e, com elas, novas oportunidades de emprego e de empreendedorismo. Um profissional da área de tecnologia da informação precisa fazer pesquisas, atualizar-se e estar sempre atento às mudanças e às necessidades do mercado. Porém, uma habilidade realmente fundamental é a adaptação às novas situações, às necessidades dos clientes e da sua empresa, pois isto vai determinar o seu sucesso. Um projeto despretensioso e pequeno pode se tornar uma ótima forma de empreender, por isso, fique sempre atento às modelagens que realizará. Com elas, novas ideias de negócios podem trazer ótimas oportunidades para sua carreira.

Nesta seção, serão apresentadas as estratégias de modelagem de banco de dados e a necessidade de documentar os diagramas criados. Lembrando que você é um empreendedor que contratará estagiários para auxiliar no processo de modelagem do banco de dados e precisará criar padrões de desenvolvimento para que, ao final da modelagem, o banco de dados possa ser implementado de forma eficaz.

Seu novo cliente é o salão de beleza da Dona Áurea, que deseja informatizar o atendimento e o controle dos clientes e dos funcionários. Deverão ser criados o modelo de entidade-relacionamentos e a documentação das tabelas geradas no modelo lógico. Os requisitos da modelagem são os seguintes:

- O cliente deve ser cadastrado e ter seus serviços principais armazenados, pois futuramente o estabelecimento pretende criar um cartão de fidelidade.
- O salão possui diversos fornecedores, de quem são comprados os produtos, e que podem ser também profissionais que prestam serviços como afiação de instrumentos, eletricidade, etc. Desta forma, precisamos classificar os tipos de fornecedores.

Com os requisitos acima listados, você precisará responder: como dona Áurea, proprietária do salão, poderá saber qual serviço um determinado funcionário realizou e como controlar o agendamento de horários dos clientes?

Será necessário documentar todo o modelo de banco de dados produzido. Isso é importante pelos seguintes motivos: organização, apresentação e controle (caso seja necessário incluir campos ou mesmo entidades). Após a elaboração do diagrama entidade-relacionamento (DER), deverá ser realizado o mapeamento das entidades, juntamente com a documentação completa das tabelas, neste caso, um dicionário de dados.

Bons estudos!

Não pode faltar

No desenvolvimento de qualquer software devemos sempre considerar o seu ciclo de vida, que nada mais é do que o início do software através do estudo e do planejamento de sua viabilidade até o seu término na fase da manutenção ou do abandono do software. Em um projeto de banco de dados também há um ciclo de vida que determinará o começo do projeto até o seu final (que neste caso é a manutenção ou a evolução do banco de dados). Coronel e Rob (2011) destacam as seis fases e suas respectivas ações do ciclo de vida de um banco de dados:

- **Estudo inicial do banco de dados:** estudo dos requisitos do problema e suas restrições e definição dos objetivos, escopo e fronteiras do banco de dados.
- **Projeto do banco de dados:** criação do projeto conceitual, escolha do sistema de gerenciamento do banco de dados (SGBD) que deverá ser usado e criação do projeto lógico e físico do banco de dados.
- **Implementação e carga:** instalação do SGBD, criação do banco de dados e carregamento ou conversão dos dados que serão armazenados no banco.
- **Teste e avaliação:** realização de testes na base de dados para encontrar possíveis erros.

- **Operação:** o banco entra em funcionamento nos aplicativos desenvolvidos em paralelo.
- **Manutenção e evolução:** assim que entra em operação, o banco de dados deve sempre receber manutenção para ficar o máximo possível em plena operação e a evolução do banco de dados acontece assim que novas necessidades do usuário surgem.



Exemplificando

As manutenções que acontecem em um banco de dados podem ser:

- Preventiva: por causa do backup.
- Corretiva: se houver necessidade de recuperação de informação.
- Adaptativa: para melhor o desempenho, para acrescentar tabelas ou campos ou para dar permissões de acessos.

Conforme afirmam Coronel e Rob (2011), há duas abordagens clássicas tradicionais que podem ser adotadas como estratégia de modelagem em um diagrama de entidade-relacionamentos:

- Estratégia *top-down*: inicializa-se identificando os conjuntos de dados e, em seguida, são definidos os elementos de cada um desses conjuntos. O processo envolve a identificação de diferentes tipos de entidades e a definição de cada atributo. Esta técnica é utilizada em banco de dados maiores.
- Estratégia *bottom-up*: primeiramente são identificados os elementos de dados, ou seja, os itens, e então eles são agrupados em conjuntos de dados. Neste caso, os atributos são identificados primeiro e, ao agrupá-los, teremos as tabelas. Geralmente, esta técnica é utilizada em banco de dados pequenos.

As abordagens *top-down* e *bottom-up* acabam se complementando, pois muitas vezes um analista ou projetista aplica as duas técnicas no mesmo banco de dados a ser modelado, surgindo então uma abordagem mista, denominada *middle-up-down*.

A modelagem conceitual em um projeto de banco de dados é considerada de alto nível, pois possui como finalidade a fácil compreensão entre os usuários envolvidos na modelagem, como ressaltam Korth, Silberschatz e Sudarshan (2012). O foco da modelagem conceitual (Quadro 3.1) é detalhar e discutir o funcionamento do negócio do cliente e não o uso de determinada tecnologia, descartando dados de como as informações serão armazenadas e depois recuperadas em banco de dados.

Quadro 3.1 | Comparaçāo entre as modelagens conceitual e lógica

	Modelagem conceitual	Modelagem lógica
Entidades	Somente as importantes	Incluem todas as entidades (chamadas de tabelas)
Atributos	Não são especificados	Incluem todos os atributos (chamados de campos)
Chaves	Não são especificadas	Especificadas as chaves primárias e estrangeiras
Relacionamentos	Somente os importantes	Incluem todos os relacionamentos entre as tabelas

Fonte: elaborado pela autora.

Assim que o modelo lógico começar a ser implementado, o modelo conceitual servirá de apoio à construção do esquema do banco de dados. Navathe e Ramez (2005) afirmam que durante ou mesmo ao término do esquema conceitual, as operações básicas do modelo de dados podem ser usadas para especificar as operações de alto nível do usuário e servem para verificar se o modelo possui todos os requisitos listados pelo cliente.

Para criar um modelo lógico e mais coeso do banco de dados, são necessárias várias revisões na descrição do modelo conceitual (de alto nível) e, desta forma, encontrar:

- Tabelas: em substantivos, objetos reais e objetos que podem armazenar informações.

- Campos: em características específicas de algum objeto ou adjetivos.
- Relacionamentos: em verbos que “ligam” uma tabela a outra.
- Cardinalidades: a quantidade de vezes que cada tabela pode estar relacionada com outra.

Algumas normas precisam ser adotadas durante a criação do modelo lógico do banco de dados, na criação do diagrama de entidade-relacionamentos. Estão elencadas abaixo as principais regras que norteiam os fundamentos da modelagem de dados, conforme adaptado de Heuser (2011):

- Em casos de relacionamento 1 para N: a chave primária do lado 1 sempre deverá estar na tabela do lado N como uma chave estrangeira.
- Em casos de relacionamento N para N: o relacionamento passa a ser implementado como tabela própria que possui campos específicos relacionados entre as duas tabelas que deram origem a esta nova tabela, chamada tabela associativa.
- As tabelas devem ter o número reduzido de chaves primárias ao mínimo possível, ou seja, sempre que possível, uma tabela deverá ter somente um identificador único, evitando chaves alternativas.



Os modelos de dados podem estar classificados como: modelo de alto nível, modelo intermediário e modelo de baixo nível. O de alto nível é o modelo conceitual, que fornece uma visão mais próxima de como os usuários realmente enxergam os dados. O modelo físico fornece uma visão mais detalhada de como realmente o dado será armazenado dentro do SGBD. O modelo intermediário está entre os dois tipos (alto nível e baixo nível) e faz uma espécie de ligação. Usa-se o modelo conceitual para a construção de um esquema lógico de banco de dados e que será depois implementado no SGBD (pelo modelo físico).

Na maioria dos projetos existe uma grande quantidade de tabelas e campos envolvidos. É necessário criar padrões de desenvolvimento para evitar problemas de conflito de nomes de

atributos, por exemplo. Em uma modelagem em que dois ou três analistas ou programadores estejam trabalhando, caso não haja um padrão, o mesmo campo pode ser criado e referenciado com nomes diferentes, dificultando uma consulta ou alguma manutenção realizada posteriormente. É necessário criar o dicionário de dados para estabelecer uma padronização e uma documentação sobre cada tabela criada. Korth, Silberschatz e Sudarshan (2012) mencionam um dicionário como uma descrição de dados, ou seja, contém *metadados* que são detalhes dos dados armazenados na tabela. Observe no Quadro 3.2 um exemplo simplificado de dicionário de dados.

Quadro 3.2 | Dicionário de dados da tabela funcionário

Tabela: funcionário				
	Campo	Descrição	Tipo	Tamanho
PK	Cd_Func	Código do funcionário	VARCHAR	20
	Nm_Func	Nome do funcionário	VARCHAR	100
	CPF_Func	CPF do funcionário	VARCHAR	15
	Dt_Nasc_Func	Data de nascimento funcionário	Date	-
FK	Id_Cidade	Cidade do funcionário	Inteiro	-

Fonte: elaborado pela autora.

Cada empresa de desenvolvimento de software possui o seu próprio padrão de dicionário de dados, porém, Korth, Silberschatz e Sudarshan (2012) afirmam que um dicionário de dados deve ter várias informações, como as seguintes:

- Descrição dos nomes das tabelas, relações e atributos.
- Tipos dos dados (domínio) e seus respectivos tamanhos.
- Descrição detalhada das chaves utilizadas.
- Nomes dos usuários com suas permissões sobre a tabela.

O nível de detalhamento do dicionário de dados é opcional, mas ele acaba se tornando um documento fundamental, sendo muito requisitado quando o banco de dados apresenta problemas e precisa sofrer manutenção. É comum que a pessoa que realizará a manutenção não seja a mesma que criou o banco de dados, por isso a documentação dos campos é fundamental para acelerar o processo de manutenção. O Quadro 3.3 mostra outra forma de dicionário de dados, mais robusta e com mais informações.

Quadro 3.3 | Dicionário de dados da tabela funcionário modo completo

Tabela: funcionário					
Atributos					
Atributo	Campo	Tipo de dado	Tamanho	Descrição	Restrição
Código	Cd_Func	VARCHAR	20	Código do funcionário	Chave primária
Nome	Nm_Func	VARCHAR	100	Nome do funcionário	Nome completo
CPF	CPF_Func	VARCHAR	15	CPF do funcionário	CPF válido
Data Nasc	Dt_Nasc_Func	Date	-	Data de nascimento funcionário	Data formato dd/mm/aaaa
Cidade	Id_Cidade	Inteiro	-	Cidade do funcionário	Chave estrangeira da tabela cidade obrigatória

Fonte: elaborado pela autora.



Pesquise mais

Saiba mais sobre o uso de *metadados* como forma de padronização, sendo utilizados em bibliotecas digitais como ambiente propício para a recuperação da informação.

CASTRO, F. F.; SANTOS, P. L. V. A. C. **Os metadados como instrumentos tecnológicos na padronização e potencialização dos recursos informacionais no âmbito das bibliotecas digitais na era da web semântica.** Inf. & Soc.: Est., João Pessoa, v. 17, n. 2, p. 13-19, maio/ago. 2007. Disponível em: <<https://bit.ly/2Awx2Dw>>. Acesso em: 1 ago. 2018.

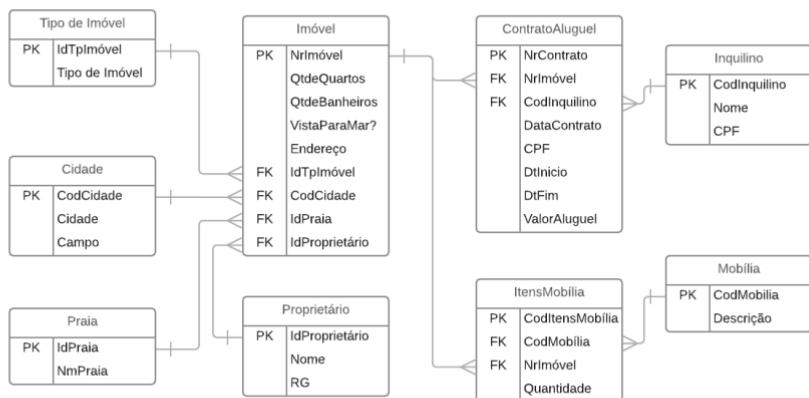
Para exemplificar um modelo lógico de entidade-relacionamento, vamos modelar um estudo de caso. Uma imobiliária especializada em aluguel de casas e apartamentos do litoral de Santa Catarina necessita de um software para ajudar no gerenciamento dos aluguéis e oferecer melhores ofertas para seus clientes. Após diversos contatos com a imobiliária, ficou estabelecido que os seguintes requisitos deveriam ser atendidos pelo banco de dados:

- Para cada imóvel deverá ter registrado: seu tipo (casa ou apartamento), quantidade de quartos e banheiros, se possui vista para o mar e preço da diária.
- As informações dos proprietários e dos inquilinos deverão ser armazenadas separadamente. Os proprietários podem ter vários imóveis que podem ser alugados para vários inquilinos.
- Além das informações sobre o município ao qual o imóvel pertence, deverá também ser informado o nome da praia mais próxima a ele.
- Os imóveis são todos os itens que compõem a mobília, e os mais verificados são: cama, geladeira, freezer, televisor, ar-condicionado, entre outros. Neste caso, é importante que seja informada a quantidade de cada item.
- Deverá ser realizado e registrado um contrato exclusivo para os aluguéis com os inquilinos e os imóveis respectivamente alugados por eles.

Com os requisitos listados, podemos identificar primeiramente as entidades que se destacam: imóvel, tipo de imóvel, cidade, praia, proprietário, inquilino, contrato de aluguel e mobília.

Para expressar graficamente o diagrama de entidade-relacionamentos, utilizamos a notação “Pé de Galinha” de James Martin, de acordo com Coronel e Rob (2011). Para entender o diagrama da Figura 3.1, observe as entidades e seus atributos.

Figura 3.1 | DER imobiliária



Fonte: elaborada pela autora.

Para melhor entendimento do diagrama da imobiliária, considere os seguintes itens:

- Observe as setas que ligam as tabelas, o lado que tem três pontas está representando o lado N, o que possui uma ponta é o lado 1.
- Observe que há as siglas PK e FK em alguns campos.
- PK significa *Primary Key* ou a chave primária da tabela.
- FK significa *Foreign Key* ou a chave estrangeira da tabela.

O dicionário de dados da tabela imóvel pode ser observado no Quadro 3.4. Para fins didáticos, estamos criando o dicionário de somente uma tabela e com poucos campos. Vale ressaltar que os acentos das palavras foram preservados, mas na criação do modelo físico no SGBD, os acentos deverão ser evitados.

Quadro 3.4 | Dicionário de dados da tabela imóvel

Tabela: imóvel				
	Campo	Descrição	Tipo	Tamanho
PK	Nrlmóvel	Número do imóvel	Varchar	20
	QtdeQuartos	Quantidade de quartos	Inteiro	-
	QtdeBanheiros	Quantidade de banheiros	Inteiro	-
	VistaParaMar?	Tem vista para o mar?	Boolean	True / False
	Endereço	Endereço completo	Varchar	150
FK	IdTpImóvel	Tipo do imóvel (apart, casa)	Inteiro	-
FK	CodCidade	Cidade	Inteiro	-
FK	IdPraia	Praia mais próxima	Inteiro	-
FK	IdProprietário	Proprietário	Inteiro	-

Fonte: elaborado pela autora.



Refita

Na Figura 3.1 é demonstrado o diagrama de entidade-relacionamentos como sendo uma solução inicial para a modelagem do banco de dados da imobiliária. Em um DER sempre podemos agregar mais tabelas, além das que aparecem explicitamente no enunciado. Uma situação que o cliente não levantou foi a questão do contrato que o proprietário deve ter em relação ao seu imóvel. O que você faria para criar um contrato para o proprietário? Outra situação é que a imobiliária pode expandir suas atividades para outros estados. Como você poderia criar uma relação entre cidade, estado e praias?

Nesta seção você pode conhecer mais alguns aspectos do processo de modelagem de dados, como as estratégias de modelagem *top-down* e *bottom-down* e a documentação do desenvolvimento de um diagrama de entidade-relacionamentos.

Em sua vida profissional você conhecerá e usará modelos diferentes, adaptados pelas empresas de acordo com suas necessidades de desenvolvimentos. A necessidade de apartação das novas tecnologias determinará o seu sucesso profissional. Na próxima seção, abordaremos os padrões de modelagem com UML, uma linguagem de modelagem unificada e muito utilizada na programação orientada a objetos.

Sem medo de errar

Esta seção trouxe novas ferramentas para aperfeiçoar o desenvolvimento das modelagens de banco de dados de sua empresa: os dicionários de dados poderão auxiliar na organização, na apresentação e no controle de cada campo em uma tabela. Será possível adotar uma estratégia de modelagem que melhor se enquadre nos padrões de desenvolvimento de sua empresa.

Agora com um novo cliente, você contará com ajuda de estagiários, que vão auxiliar na modelagem do banco de dados. Novamente, é necessário organização e controle nessas atividades.

Seu novo cliente, o salão de beleza da Dona Áurea, deseja informatizar o atendimento e controle dos clientes e dos funcionários. Os requisitos da modelagem são os seguintes:

- O cliente deve ser cadastrado e ter seus serviços principais também armazenados, pois futuramente o salão pretende criar um cartão de fidelidade.
- O salão possui diversos fornecedores, de quem são comprados os produtos, e que podem ser também profissionais que prestam serviços como afiação de instrumentos, eletricidade, etc. Desta forma, precisamos classificar os tipos de fornecedores.

Deverá ser criado o modelo de entidade-relacionamentos e a documentação das tabelas criadas no modelo lógico. Com os requisitos apontados, duas perguntas precisarão ser respondidas: qual serviço um determinado funcionário realizou e como controlar o agendamento de horários dos clientes? Como resposta à primeira pergunta, deveremos criar três tabelas (atendimento, serviços e funcionários) e a relação

entre elas. Em relação ao segundo questionamento, é necessário criar a tabela agenda, devendo conter o cliente, o serviço que a pessoa deseja realizar e o funcionário requisitado.

Na solução da situação-problema, vamos elencar as tabelas que apareceram nos questionamentos e nos requisitos: cliente, serviço, funcionário, agenda, atendimento, fornecedores e produtos.

Com as tabelas listadas, agora é com você, faça o DER da seguinte maneira:

- Encontre o relacionamento entre as tabelas.
- Indique as cardinalidades entre as tabelas.
- Encontre campos para as tabelas.
- Identifique as chaves primárias para cada tabela.
- Verifique os relacionamentos e insira as chaves estrangeiras nas tabelas que possuam a cardinalidade N.

Após a elaboração do DER, deverá ser realizado o mapeamento das entidades na forma textual, juntamente com a documentação completa das tabelas, neste caso, um dicionário de dados.

Pesquise uma ferramenta gráfica que permita a criação do modelo gráfico de forma mais prática e, então, crie um slide de apresentação para seu cliente, contendo o modelo lógico gráfico, o modelo textual e o dicionário de dados de todas as tabelas.

Avançando na prática

DER para uma biblioteca comunitária

Descrição da situação-problema

No bairro onde você mora há um centro comunitário que recebeu alguns livros usados de todos os tipos: romances, pesquisas científicas, escolares, infantis, etc. Então, surgiu a ideia de criar uma pequena biblioteca para disponibilizar os livros. Logo as crianças do bairro começaram a querer pegá-los emprestados, causando uma certa preocupação pela falta de controle. Em paralelo, novas doações de livros chegaram ao centro comunitário, juntamente com um computador para ser utilizado na biblioteca do centro. O que você poderá fazer para ajudar nessa situação?

Resolução da situação-problema

Observe que, por ser um centro comunitário, possivelmente não haverá verbas para a compra de licença de softwares. O ideal é instalar um sistema operacional Linux e um SGBD freeware, como o Kexi ou o Base (concorrentes das versões proprietárias, como o Access), e que, além de criar o banco de dados, permitem criar telas intuitivas para usuários leigos. Para ajudar no controle de empréstimos dos livros, poderá ser desenvolvido um banco de dados pequeno. O modelo lógico textual do banco de dados ficará da seguinte forma:

Livro (**#codLivro**, Título, &codTipoLivro, &IdAutor, &IdEditora)

Tipo de Livro (**#codTipoLivro**, Tipo de Livro)

Autor (**#IdAutor**, Nome)

Editora (**#IdEditora**, Editora)

Pessoa (**#matrículaPessoa**, Nome, DtNasc, NomeResponsável, Endereço, foto)

Empréstimo (**#CodEmpréstimo**, DtRetirada, DtDevolução, &codLivro, &matrículaPessoa)

Como sugestão, você poderá:

- Pesquisar e criar uma versão gráfica das tabelas e dos campos listados na versão textual do modelo lógico do banco de dados.
- Criar o dicionário de dados de todas as tabelas mostradas na versão textual do modelo lógico do banco de dados.

Faça valer a pena

1. Conforme afirmam Coronel e Rob (2011), há duas abordagens clássicas tradicionais que podem ser adotadas como estratégia de modelagem em um diagrama de entidade-relacionamentos: *top-down* (que se inicializa identificando os conjuntos de dados e, então, são definidos os elementos de cada um desses conjuntos) e *bottom-up* (são identificados os elementos de dados ou seja, os itens, que são agrupados em conjuntos de dados).

Marque a alternativa correta que identifica o meio termo entre as estratégias de modelagem *top-down* e *bottom-down*.

- a) *middle-down*.
- b) *middleware*.
- c) *middle-up-down*.
- d) *up-down-safe*.
- e) *safe-up-down*.

2. O ciclo de vida de um software descreve as atividades desde sua concepção até sua última fase, que é a manutenção ou extinção. Em um projeto de banco de dados também há um ciclo de vida que vai determinar o projeto do começo ao fim (neste caso, a manutenção ou a evolução). Destacam-se as seguintes fases do ciclo de vida de um banco de dados:

- I. Estudo dos requisitos do problema e suas restrições, definição dos objetivos, escopo e fronteiras do banco de dados.
- II. Criação do projeto conceitual, escolha do SGBD que deverá ser usado, criação do projeto lógico e físico do banco de dados.
- III. Instalação do SGBD, criação do banco de dados, carregamento ou conversão dos dados que serão armazenados no banco.
- IV. Realização de testes na base de dados para encontrar possíveis erros.

Analizando cuidadosamente as afirmativas apresentadas, é correto o que se afirma em:

- a) Apenas as afirmativas II, III e IV estão corretas.
- b) Apenas as afirmativas I, III e IV estão corretas.
- c) Apenas as afirmativas I, II e III estão corretas.
- d) Apenas as afirmativas I, II e IV estão corretas.
- e) As afirmativas I, II, III e IV estão corretas.

3. Utilizamos os requisitos para criar os modelos de banco de dados. Geralmente as tabelas são encontradas através dos _____, os campos são as _____ e o relacionamentos são os _____ que ligam uma tabela a outra.

Assinale a alternativa que completa as lacunas corretamente:

- a) verbos – alternativas – meios.
- b) substantivos – alternativas – tipos.
- c) meios – chaves – tipos.
- d) substantivos – características – verbos.
- e) verbos – chaves – substantivos.

Seção 3.2

Modelagem de dados através do modelo entidade-relacionamento usando UML

Diálogo aberto

Desenvolver um software requer o conhecimento de diversas áreas, e uma das que mais crescem em TI é o desenvolvimento orientado a objetos, que tem como base a reutilização de componentes em outros softwares. A reutilização é uma palavra muito importante no desenvolvimento de sistemas. Poder aproveitar códigos, classes e partes inteiras de um sistema em outro é redução de custos. Para isso realmente acontecer, é preciso planejamento e padronização dos processos de desenvolvimento de um software. Nesta seção vamos aprender sobre uma ferramenta de padronização com UML, amplamente utilizada por analistas de sistemas e que pode ser usada no processo de modelagem de banco de dados.

Retomando a nossa situação-problema, você está no processo de modelagem de um banco de dados para o salão de beleza da Dona Áurea, que deseja informatizar os serviços. As seguintes tabelas já foram encontradas: Cliente, Serviço, Funcionário, Agenda, Atendimento, Fornecedores e Produtos.

Para dar conta de tantas tarefas, você contratou dois estagiários e precisa estabelecer padrões de modelagens de seus softwares. Será que poderá utilizar o padrão UML para o modelo lógico de banco de dados do salão? Qual tabela poderá receber o relacionamento generalização e especialização? Como saber quais tabelas serão acrescentadas no modelo lógico?

Você precisará criar o Diagrama Entidade-Relacionamentos (DER) utilizando a notação UML.

Leia com atenção esta seção pois, certamente, esse conteúdo será novamente abordado em novas disciplinas do seu curso de computação.

Bons estudos!

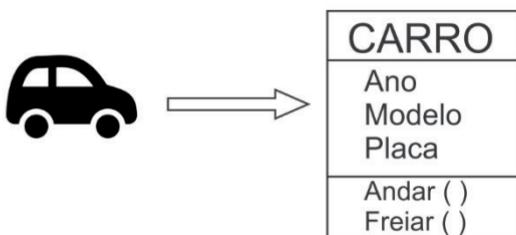
Não pode faltar

Atualmente, a maioria das linguagens de programação é orientada a objetos, entre elas, podemos citar Java e C# que, apesar de serem diferentes, compartilham dos mesmos conceitos e paradigmas da programação orientada a objetos. Um software desenvolvido nestas linguagens utilizando classes pode perfeitamente armazenar as informações em bases de dados relacionais. Porém, antes de mais nada, o que é um objeto? Um objeto pode ser uma pessoa, alguma coisa, um processo real ou abstrato e que possua informações e funcionalidades, podendo representar:

- Entidades físicas: um cliente, uma bicicleta, um imóvel.
- Entidade conceitual: um diagrama entidade-relacionamentos de um sistema.
- Entidade de software: um botão de confirmação de um software.

Podemos agrupar as informações comuns de um objeto, surgindo então a classe, que pode ser classificada como uma entidade teórica usada para modelar um objeto. A classe deve ser exclusivamente criada para guardar informações de um determinado objeto. A Figura 3.2 demonstra o exemplo do objeto Carro, que tem as suas informações armazenadas em uma classe Carro.

Figura 3.2 | Exemplo da classe Carro

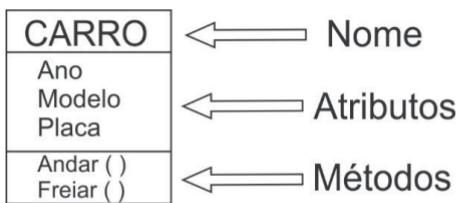


Fonte: elaborada pela autora.

Uma classe é dividida em três partes: o nome, os atributos e os seus métodos, como na Figura 3.3. O nome deve ser referente ao que a classe representa, isto é, ao ler o nome da classe, já devemos identificar o que ela vai armazenar. Os atributos são as

características do objeto. Os métodos são os comportamentos que um objeto poderá assumir. Fazendo uma analogia à linguagem de programação C++, os métodos são as funções criadas para realizar alguma tarefa referente aos seus atributos ou ao objeto.

Figura 3.3 | Classe Carro



Fonte: elaborada pela autora.

Conforme Mizrahi (2008), uma classe pode ser conceituada como um tipo de dado, assim como os tipos predefinidos que existem em compiladores de diversas linguagens de programação ou do próprio Sistema de Gerenciamento de Banco de Dados (SGBD). Uma classe é formada por dados e comportamentos. Para a definição dos dados de uma classe são utilizados os atributos e, para definir os comportamentos, usamos os métodos (funções que manipulam os atributos da classe). Medeiros (2004) afirma que uma instância de uma classe é a criação de um objeto baseado em uma determinada classe. Ao instanciarmos um objeto, criamos esses objetos na memória.

Você deve estar pensando: “mas o que isso tem a ver com a modelagem de dados?”. A resposta é que os objetos serão persistidos em algum banco de dados e, por isso, também precisam ser modelados de acordo com tais regras que estamos aprendendo. Para auxiliar nesse processo de modelagem de classes, existe a Linguagem de Modelagem Unificada ou, como mais utilizada, a sigla UML (*Unified Modeling Language*), uma ferramenta que auxilia na modelagem de sistemas orientados a objetos. A finalidade da UML é proporcionar uma padronização nos projetos de sistemas, abrangendo aspectos conceituais, como regras de negócios, e artefatos concretos, como as classes, escritas em linguagens de programação, esquemas de banco de dados e componentes de software reutilizáveis, segundo Medeiros (2004).

A Linguagem UML, conforme Fowler (2004), é composta por diversos diagramas, destacando-se os seguintes:

- Diagrama de classes: é o mais usado da Linguagem UML, pois pode representar um conjunto de classes e seus relacionamentos.
- Diagrama de objetos: demonstra como na realidade as informações do objeto podem ficar armazenadas na classe.
- Diagrama de caso de uso: é um complemento do diagrama de classes, utilizado principalmente na fase de especificação dos requisitos do sistema, pois demonstra os usuários e as funcionalidades do software.
- Diagrama de sequência: demonstra uma visão ou perspectiva, norteada por tempo, da colaboração entre os objetos, principalmente com a ordem temporal em que as mensagens são trocadas.
- Diagrama de atividades: determina o fluxo de tarefas que podem ser executadas pelo software ou por um ator.
- Diagrama de estados: pode representar um conjunto de estados que um objeto pode ter e os "gatilhos" que possam estimular a mudança ou a transição do objeto de um estado para outro.
- Diagrama de componentes: demonstra os componentes do software e seus relacionamentos, podendo ser: bibliotecas, arquivos de ajuda e classes que podem ser anexadas ao software.

A cardinalidade utilizada em um diagrama de classe é semelhante ao diagrama de entidade-relacionamentos, porém, podemos determinar o número exato de ocorrências, por exemplo, no mínimo 1 ocorrência e no máximo 5. No diagrama DER usamos a letra *N* para representar "muitos", já no diagrama de classes é utilizado o sinal de asterisco (*) para a mesma função. Observe que a classe "Estado" na Figura 3.4 possui dois atributos do tipo *string* e um método. Já a classe "Cidade" possui três atributos, sendo dois do tipo inteiro e um do tipo *string*, além de um método. É importante ressaltar que a quantidade de atributos e métodos varia de projeto para projeto, pois o desenvolvedor deverá modelar a classe da forma que achar mais adequada.

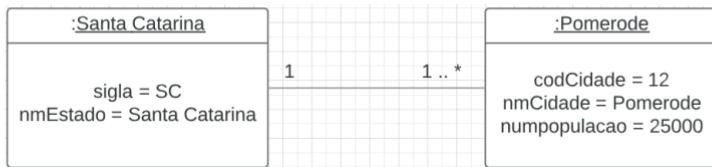
Figura 3.4 | Exemplo de relacionamento entre duas classes



Fonte: elaborada pela autora.

Quando uma classe é instanciada (ação de criá-la), tem-se um objeto na memória de trabalho do computador. Nesse caso, os atributos da classe são “preenchidos” com dados. Assim como existe o diagrama de classes, também existe o diagrama de objetos para mostrar as informações que serão armazenadas na classe. A Figura 3.5 apresenta um exemplo de como ficarão os atributos apresentados no diagrama da Figura 3.4.

Figura 3.5 | Exemplo de diagrama de objetos



Fonte: elaborada pela autora.



A UML originalmente foi proposta para realizar a modelagem de sistemas orientados a objetos, mas ela pode ser utilizada como uma notação em projetos de banco de dados, deixando a modelagem de todo o sistema de forma padronizada. O diagrama de classes, como visto na Figura 3.2, pode ser usado para representar o projeto lógico de um banco de dados (geralmente, o modelo conceitual é realizado com o diagrama de entidade-relacionamentos).

Uma classe, quando instanciada, guarda as informações na memória RAM do computador. Toda vez que ela precisa armazenar suas

informações para depois recuperá-las, é necessário gravar em arquivos ou em banco de dados. Este processo é chamado de persistência. Uma classe de persistência tem como finalidade guardar as informações em um meio persistente, para que possam ser recuperadas posteriormente.



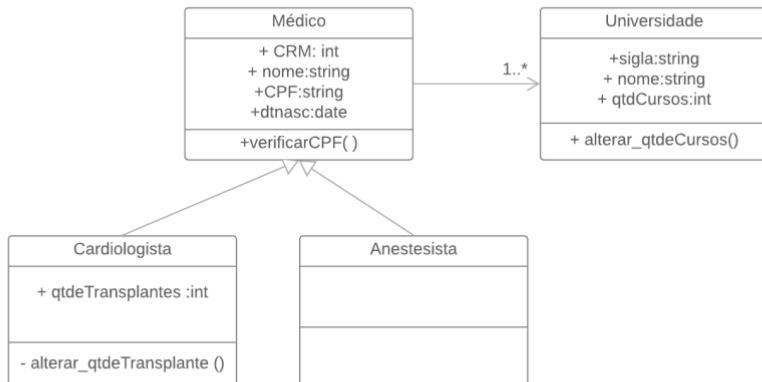
Exemplificando

A persistência de objetos pode ser definida como a capacidade que uma linguagem de programação tem de permitir que os dados e objetos resistam a um processo, para mais tarde utilizá-los em outro. Existem várias técnicas de persistência, como: Padrão DAO (*Data Access Object*), Força Bruta, Active Record, Persistência Robusta (*Framework*) e Padrão JDO (*Java Data Object*).

Alguns desses padrões você verá nas aulas de programação orientada a objetos. Aproveite para pesquisar sobre eles!

Mizrahi (2008) afirma que na programação orientada a objetos podemos relacionar classes com outras classes através de hierarquias. Uma subclasse pode herdar as características de outra classe (neste caso, seria a classe mãe ou superclasse). Na estrutura de herança, as classes compartilham suas funções e características comuns, e as subclasses podem receber outras particularidades exclusivas. A Figura 3.6 mostra uma estrutura de herança entre as classes.

Figura 3.6 | Exemplo de herança



Fonte: elaborada pela autora.

Na Figura 3.6, a classe Médico é a superclasse e suas duas classes filhas (Cardiologista e Anestesista) herdam os atributos e métodos da classe mãe. Isso significa que os atributos CRM, nome, CPF e dtnasc não precisam ser escritos nas classes filhas, pois serão herdados da superclasse. A classe Anestesista não possui nenhum atributo ou método além dos herdados, porque podemos definir os atributos e métodos conforme necessário (mas precisamos criar a classe no diagrama).



Pesquise mais

Saiba mais sobre a modelagem de dados utilizando o UML. Este artigo apresenta uma modelagem de uma aplicação web, demonstrando como as classes persistentes definidas na UML podem ser traduzidas no modelo relacional de um banco de dados.

TANAKA, S. A. et al. **Modelagem de uma aplicação web a partir de um framework de agenda de tarefas**. Londrina, [s.d.]. Disponível em: <<http://bit.ly/2MauuQ9>>. Acesso em: 9 ago. 2018.

Estabelecer a herança entre classes é um recurso da orientação a objetos que pode ser empregado na modelagem de dados relacionais, a fim de permitir um melhor suporte a relacionamentos entre estruturas de classes (mãe e filhas).

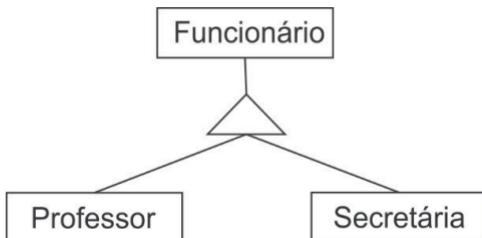
A herança tem como principal fundamento a possibilidade de criar subclasses que possam herdar características da classe mãe. Esta mesma analogia é aplicada aos modelos de entidade-relacionamentos e, a este processo, em que várias entidades (tabelas) são agrupadas em uma única entidade genérica, damos o nome de **generalização**. **Especialização** é o processo inverso, em que novas entidades são criadas com atributos que acrescentam detalhes à entidade genérica.

Korth, Silberschatz e Sudarshan (2012) afirmam que em diagrama de entidade-relacionamentos, a generalização e a especialização são um tipo de relacionamento entre entidades que determina que uma entidade contém a outra, isto quer dizer que uma entidade superior contém um ou mais conjuntos de entidades inferiores.

Cougo (1998) destaca que em um diagrama de entidade-relacionamentos, as estruturas de generalização e especialização são úteis para que sejam retiradas a complexidade de entidades, e o seu uso

está diretamente ligado ao nível de detalhamento que desejamos em um modelo lógico. Na Figura 3.7 criamos uma tabela generalizada chamada Funcionário e também as tabelas especializadas chamadas Professor e Secretária. Nesse cenário, as entidades Professor e Secretária possuem atributos da entidade Funcionário, mais as suas específicas.

Figura 3.7 | Exemplo de generalização-especialização

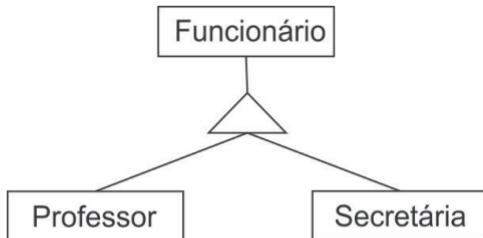


Fonte: elaborada pela autora.

A estrutura de generalização e especialização é representada pelo triângulo que une as entidades. Quando devemos usar estruturas de generalização e especialização em um diagrama de entidade-relacionamentos? Quando existem atributos (campos das tabelas) específicos para determinada entidade. Uma estrutura de generalização e especialização pode ser classificada em **Parcial** ou **Total**. Na Figura 3.8 podemos observar que existe um “p” ao lado do triângulo.

Uma generalização e especialização parcial vista na Figura 3.8 indica que nem todo funcionário é professor ou secretária e, neste caso, não é toda ocorrência da entidade generalizada que possui um entidade especializada correspondente.

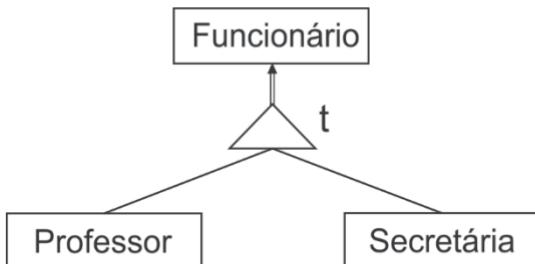
Figura 3.8 | Exemplo de generalização e especialização parcial



Fonte: elaborada pela autora.

Podemos ter uma generalização e especialização total quando, a cada ocorrência da entidade generalizada, houver, obrigatoriamente, a entidade especializada. Na Figura 3.9 podemos observar um “t” que indica a totalidade e uma seta dupla apontando para a entidade generalizada. Neste exemplo, a seta está indicando que necessariamente um funcionário será ou professor ou secretária.

Figura 3.9 | Exemplo de generalização e especialização total



Fonte: elaborada pela autora.



Refletia

Quais os benefícios da utilização de herança (generalização e especialização) em um diagrama de entidade-relacionamentos? Como podemos detectar essa necessidade em nossos modelos?

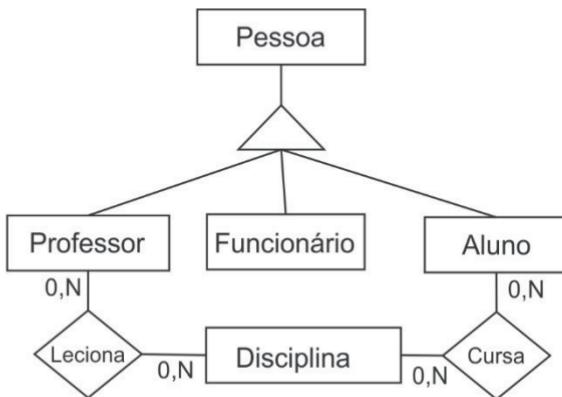
Um exemplo prático da utilização de UML em um diagrama de entidade-relacionamentos é a aplicação da generalização e especialização, que nada mais é do que usar o conceito de herança no modelo. Digamos que, para uma determinada universidade, precisamos guardar as informações das disciplinas ministradas por professores e as disciplinas que um determinado aluno cursa. Outras informações das tabelas deverão ser armazenadas:

- **Tabela Funcionário:** nome, endereço, RG, CPF, data de nascimento, data de admissão, data de demissão, salário, nome da mãe e nome do pai.

- **Tabela Professor:** nome, endereço, RG, CPF, data de nascimento, nome da mãe, nome do pai, valor da hora da aula e quantidade de horas/aula.
- **Tabela Aluno:** nome, endereço, RG, CPF, data de nascimento, data de entrada faculdade, data de formatura, salário, nome da mãe e nome do pai.

A tabela Disciplina estará relacionada com a tabela Professor e com a tabela Aluno. As tabelas Funcionário, Professor e Aluno têm algo em comum? Algum atributo se repete? Se você leu com atenção, percebeu a repetição dos seguintes atributos: nome, endereço, RG, CPF, data de nascimento, nome da mãe e nome do pai. O que fazer para evitar a repetição? Podemos criar uma entidade chamada Pessoa e nela inserir os atributos repetidos nas demais tabelas. Observe na Figura 3.10 o exemplo de como o modelo conceitual ficará.

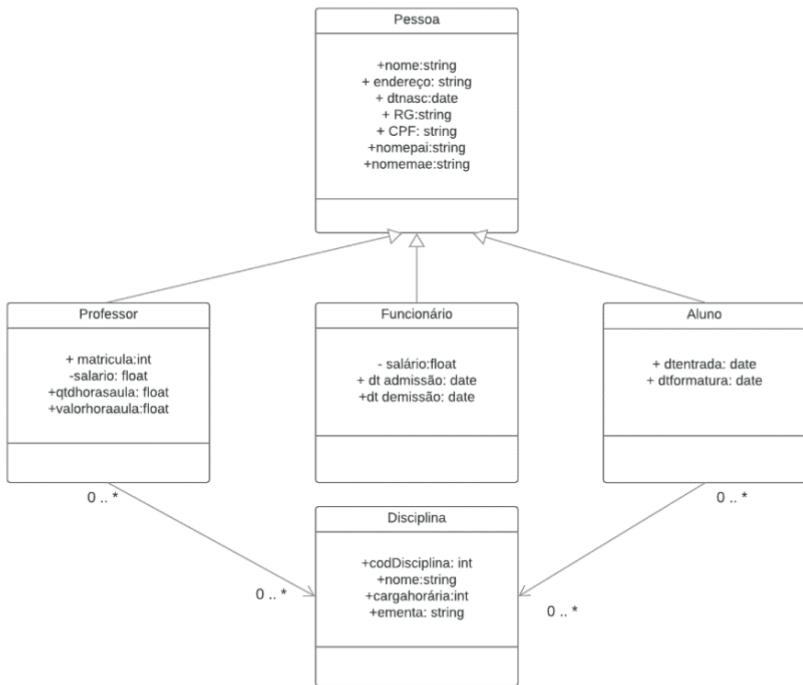
Figura 3.10 | Exemplo do modelo conceitual com generalização e especialização



Fonte: elaborada pela autora.

Korth, Silberschatz e Sudarshan (2012) descrevem que o diagrama de classes pode ser utilizado como ferramenta gráfica para criar o modelo lógico de banco de dados. Observe na Figura 3.11 como ficará o DER com a notação UML.

Figura 3.11 | Exemplo do DER com a notação UML



Fonte: elaborada pela autora.



Refletia

Observando o DER da Figura 3.11, como você acredita que ficarão as chaves estrangeiras entre as tabelas Aluno e Disciplina – Professor e Disciplina? Como “arrumar” o diagrama e contemplar as chaves estrangeiras?

A modelagem de dados abstraiu os conceitos da programação orientada a objetos como forma de evolução e acompanhamento dos conceitos mais atuais de programação. Nesta seção você conheceu a aplicação de herança (generalização e especialização) no modelo de entidade-relacionamentos. Como reflexão de aprendizagem, procure nas unidades anteriores os diagramas que podem receber este novo tipo de relacionamento entre as tabelas.

O fundamental é sempre perceber nas constantes repetições de atributos, esse é o indício mais forte da necessidade de estabelecer a generalização e a especialização entre as entidades.

Sem medo de errar

Nesta seção você conheceu uma nova forma de relacionamentos: a generalização e especialização, processo semelhante ao de herança da orientação a objetos. O padrão UML pode ser utilizado como ferramenta para auxiliar na modelagem do banco de dados. Como alguns softwares possuem seus projetos no padrão UML, é comum as empresas criarem o modelo lógico também com a notação UML. O diagrama de entidade-relacionamentos é muito semelhante ao diagrama de classes e pode ser facilmente adaptado.

Lembrando que você foi contratado pelo salão de beleza da dona Áurea para desenvolver um software para informatizar o processo de atendimento ao cliente. As seguintes tabelas já foram identificadas na seção anterior: Cliente, Serviço, Funcionário, Agenda, Atendimento, Fornecedor e Produto.

Para iniciar a solução desta situação-problema, temos primeiramente a seguinte situação: será que podemos utilizar o padrão UML para o modelo de banco de dados do salão? Com certeza podemos usar. Como sua empresa é nova, servirá inclusive de teste para verificar a viabilidade de雇用 esse padrão em outros projetos. É importante que seus dois colaboradores saibam o padrão UML e, se for o caso, vale a pena investir em treinamento para que todos entendam a padronização do modelo lógico.

O outro questionamento é: qual tabela poderá receber o relacionamento generalização e especialização e como saber quais tabelas serão acrescentadas no modelo lógico? Em visitas ao salão, ficou determinado que os clientes serão classificados em: VIP, casual e mensal. O VIP terá descontos exclusivos e facilidades de pagamentos que outros clientes não terão. O cliente mensal tem a possibilidade de optar por determinado profissional para seu atendimento e o casual não tem nenhuma especificação, pois é

aquele cliente que vai esporadicamente ou, em muitos casos, somente uma vez.

Agora é com você: crie o DER usando a notação UML. Pesquise uma ferramenta online que suporte o UML ou utilize uma ferramenta CASE, como o Astah. Crie a estrutura de generalização e especialização, utilizando as seguintes tabelas: Cliente (será a tabela mãe) e VIP, Casual e Mensal serão as tabelas filhas.

Observe que Cliente e Funcionário possuem campos semelhantes. Aprimore ainda mais seu modelo, verificando a possibilidade de criar mais uma estrutura de herança.

Elabore um slide de apresentação para seu cliente.

Bom trabalho!

Avançando na prática

Estabelecendo a necessidade do relacionamento generalização e especialização

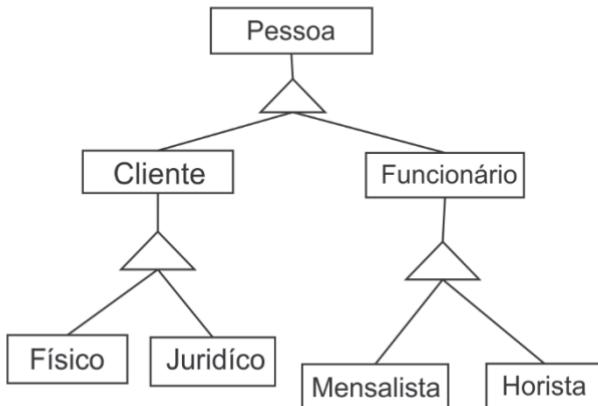
Descrição da situação-problema

Durante a modelagem de dados de uma loja de conveniências, o programador percebeu que dois atributos, nome e endereço, começaram a aparecer em quase todas as tabelas que ele estava modelando. A loja precisará criar uma separação entre os tipos de clientes (pessoa física e pessoa jurídica) para poder aplicar uma política de descontos. Além disso, será necessário classificar os empregados como horistas e mensalistas, pois, além de receber de forma diferenciada, os mensalistas recebem alguns benefícios extras. Como ficará a estrutura de entidades utilizando o relacionamento generalização e especialização?

Resolução da situação-problema

Após fazer algumas pesquisas, o programador resolveu essa situação utilizando os conceitos de heranças no modelo lógico de banco de dados. A Figura 3.12 mostra como ficaria a estrutura de generalização e especialização entre as entidades.

Figura 3.12 | Exemplo de generalização e especialização



Fonte: elaborada pela autora.

Na entidade Pessoa, podemos inserir o nome e o endereço, e as demais entidades poderão herdar esses atributos. Observe que a entidade Cliente foi generalizada para podermos classificar o Cliente entre Cliente Físico e Cliente Jurídico. Já a entidade Funcionário pode ser generalizada para que possamos criar a especialização Funcionário Horista e Funcionário Mensalista.

Faça valer a pena

1. A Linguagem de Modelagem Unificada ou UML (*Unified Modeling Language*) é uma ferramenta que auxilia na modelagem de sistemas orientados a objetos.

Marque a afirmativa correta sobre a UML:

- a) Serve para a programação somente de banco de dados.
- b) É utilizada como um SGBD pelo fato de permitir a orientação a objetos.
- c) Permite padronizar projetos de sistemas orientados a objetos e de banco de dados.
- d) É uma linguagem de programação orientada a objetos reutilizável.
- e) É utilizada somente para a criação do modelo lógico de banco de dados de um SGBD.

2. Os atributos ou campos são características de determinadas classes ou entidades. Eles devem ter nomes apropriados ao sentido do que vão armazenar e um tipo de dados que deverá ser declarado, podendo ser: inteiro, caractere, decimal, entre outros.

Assinale a alternativa correta referente à diferença entre classes e entidades.

- a) Uma classe é somente um conjunto de campos, exatamente igual as entidades dos modelos conceituais de banco de dados.
- b) Uma das vantagens da classe é possuir métodos que podem manipular os seus atributos, já as entidades possuem o objetivo de armazenar as informações em seus atributos.
- c) As classes e as entidades possuem métodos que permitem, de forma moderada, a alteração de seus atributos.
- d) Uma entidade possui a vantagem de ter métodos que podem ser utilizados para modificar os atributos da própria tabela, algo que em classe ainda não é permitido.
- e) Uma vez criados os atributos em classes ou em entidades, não é possível alterar o conteúdo armazenado, isto é, assim que o atributo for criado, recebe um valor e não pode mais ser alterado.

3. Um exemplo prático da utilização de UML em um diagrama de entidade-relacionamentos é a aplicação da generalização e especialização, que nada mais é do que usar o conceito de herança no modelo lógico do banco de dados.

Assinale a alternativa correta que mostra os tipos de generalização e especialização.

- a) Duplo e Simples.
- b) Composto e Unitário.
- c) Duplo e Isolado.
- d) Total e Parcial.
- e) Composto e Simples.

Seção 3.3

Ferramentas CASE's de modelagem do diagrama de entidade-relacionamento (DER)

Diálogo aberto

Olá, seja bem-vindo!

Nesta seção vamos utilizar ferramentas para auxiliar no processo de modelagem de dados. Você conhecerá a ferramenta CASE, cujo principal objetivo é a criação do modelo gráfico do diagrama entidade-relacionamentos. Além disso, com as informações vistas nesta seção, você deverá terminar o processo de modelagem do banco de dados do salão de beleza da Dona Áurea. Vamos relembrar o que já foi realizado?

O salão de beleza precisa de um banco de dados para o controle de clientes, funcionários, serviços realizados, fornecedores e agendamento de horários. Encontramos as tabelas Cliente, Serviço, Funcionário, Agenda, Atendimento, Fornecedores e Produtos. No segundo momento, aplicamos os recursos de herança nos tipos de Cliente e classificamos os clientes em VIP, Casual e Mensal.

A dona Áurea deseja manter um histórico de atendimento de seus clientes. Isso permitirá que ela analise o perfil de seus clientes, determine os serviços mais procurados no salão e verifique os funcionários mais requisitados. Para isso, precisaremos resolver as seguintes situações:

- Como manter um histórico dos atendimentos de cada cliente?
- Qual ferramenta CASE poderemos utilizar para criar o diagrama de entidade-relacionamentos?
- O dicionário de dados é algo extenso e trabalhoso. Existe alguma forma de tornar esse processo automático?

Você precisará criar o DER desta modelagem usando uma ferramenta CASE e, para cumprir esse objetivo, nesta seção serão apresentadas algumas ferramentas CASEs que poderão ser utilizadas.

Bom trabalho!

Não pode faltar

Na maioria dos projetos de banco de dados para softwares de médio e grande porte, vários analistas e programadores acabam trabalhando no projeto inteiro ou em parte dele. Projetar e modelar banco de dados requer disciplina. É necessário estabelecer padrões de desenvolvimento, isto é, estabelecer regras para que todos os envolvidos modelem da mesma forma, e é nessa hora que entra em cena a ferramenta CASE.

CASEs (*Computer Aided Software Engineering* ou, em português, Engenharia de Software Auxiliada por Computador) são ferramentas que apresentam uma série de serviços que auxiliam no desenvolvimento de software e podem minimizar o tempo de desenvolvimento do software modelado.

De acordo com Navathe e Ramez (2005), as primeiras ferramentas CASEs surgiram na década de 80 e eram classificadas em:

- *Lower CASE*: com suporte nas fases de análise e projeto de sistemas.
- *Upper CASE*: com suporte nas fases de construção e análise de sistemas.

Hoje em dia as ferramentas CASEs são classificadas como *Integrated CASE* com a união das ferramentas *Lower CASE* e *Upper CASE*, atendendo a praticamente todas as fases de um projeto de sistemas.

Todos os tipos de ferramentas CASEs têm em comum a possibilidade da representação gráfica de elementos do projeto, podendo ser: o diagrama de entidade-relacionamentos, os diagramas de classes, casos de usos, etc. Existem várias ferramentas CASEs disponíveis como freeware (com opções básicas) e as proprietárias (pagas) com muitos recursos.

As ferramentas CASEs são utilizadas para automatizar várias tarefas, por exemplo:

- Geração de códigos: de forma automática, os códigos podem ser gerados a partir do diagrama gráfico.
- Geração de documentação: permite padronização nos processos.
- Execução de testes: possibilita validações nas especificações formais de desenvolvimento.
- Geração de relatórios: permite o acompanhamento do planejamento e do gerenciamento do projeto.



A geração de códigos automáticos das ferramentas CASEs para banco de dados são conhecidas como scripts, que ajudam a tarefa de criar o banco de dados fisicamente no Sistema de Gerenciamento de Banco de Dados (SGBD), gerando todos os comandos em SQL de criação de tabelas, campos e chaves.

Nas atividades da engenharia de software, a ferramenta CASE é fundamental, auxiliando em todos os processos. Podemos citar Rational Rose, Astah, Genexus, Multicase, Clarify, entre outras.

As ferramentas mencionadas podem ser utilizadas para a modelagem de banco de dados, porém, existem ferramentas CASEs específicas para criar os diagramas de entidade-relacionamentos, como Oracle Designer, DBDesigner, Erwin, Embarcadero e MySQL Workbench. Uma tendência são as ferramentas CASEs online, como Draw.IO ou Lucidchart. A grande maioria das ferramentas (inclusive as online) possui versões freeware com algumas limitações, ideais para pequenos diagramas. Caso a pessoa ou empresa se interessar, poderá adquirir a versão oficial e com mais recursos. As ferramentas CASEs para banco de dados possuem as seguintes características, conforme Coronel e Rob (2011):

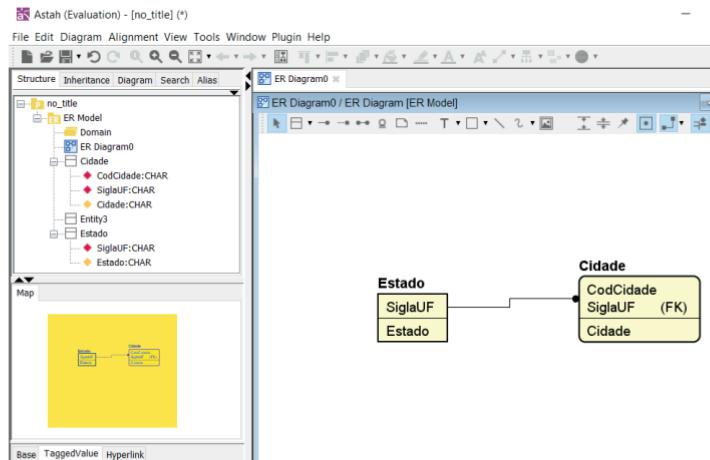
- Suporte à criação de diagramas gráficos.
- Utilização de alguma notação de modelagem de banco de dados.
- Geração de scripts SQL (*Structured Query Language – Linguagem de Consulta Estruturada*).
- *Forward Engineer*: permite, a partir do DER (modelo gráfico), conectar de forma automática o banco de dados e criar automaticamente o modelo físico.
- *Reverse Engineer*: permite, a partir do modelo físico criado no banco de dados, a geração do modelo gráfico (DER) do banco de dados.
- Documentação: conforme os atributos são criados nas tabelas, a ferramenta CASE já cria o dicionário de dados de forma automática.

O Astah é uma ferramenta CASE para criar diagramas UML e possui as seguintes versões:

- *Community*: gratuita para projetos UML (com algumas limitações).
- *Professional*: versão completa e paga (ou disponível de forma trial).

É uma ferramenta ideal para os desenvolvedores Java, pois gera os scripts em Java, acelerando o processo de desenvolvimento do software. Na versão *Professional* há a possibilidade de criar diagramas de entidade-relacionamentos, utilizando a notação IDEF1X, conforme a Figura 3.13, cuja bolinha preta representa o *N* de muitos. Você poderá fazer o download diretamente no site: <<http://astah.net/download>>. Acesso em: 14 ago. 2018.

Figura 3.13 | Exemplo ferramenta CASE Astah



Fonte: captura de tela do Astah, elaborada pela autora.



Exemplificando

Para criar o diagrama da Figura 3.13, siga o passo a passo a seguir:

1. Vá ao Menu e escolha *Diagram* → *ER Diagram*.
2. Selecione a figura da tabela e dê dois cliques. Adicione os campos e insira as chaves primárias e estrangeiras na parte superior da tabela.
3. Para criar o relacionamento, selecione a tabela do lado 1, escolha a linha para relacionar as tabelas, arraste e solte em cima da chave estrangeira.

Uma das vantagens da ferramenta Astah é a possibilidade de criar automaticamente o dicionário de dados, basta exportar para a ferramenta Microsoft Excel, e o dicionário de dados é gerado com todas as tabelas, conforme a Figura 3.14. Para exportar o dicionário de dados, clique no menu do Astah, em *Tools*, entre na opção *ER Diagram* e selecione a quarta opção: *Export Entity Definition Report*. Você deverá escolher as tabelas com que deseja criar o dicionário de dados e salvar com um nome.

Figura 3.14 | Exemplo dicionário de dados gerado no Astah

The screenshot shows an Excel spreadsheet with the following structure:

- Header Row:** H1 (containing file navigation icons) and a row of letters A through Z.
- Entity Section:** Contains three rows:
 - Row 1: Entity (highlighted in blue), Logic Name (Cidade), Physical Na (empty).
 - Row 2: Kind (highlighted in blue), empty.
 - Row 3: Definition (highlighted in blue), empty.
- Empty Rows:** Rows 4 through 7 are empty.
- Attribute Section:** Contains three rows:
 - Row 8: No, Attribute(Logic), Attribute(Physical), PK, FK, AK, IE, NN, DataType, Length&Precision, Initialvalue, Definition.
 - Row 9: 1, CodCidade, , Y, , Y, CHAR, 40, Chave Primária da Tabela.
 - Row 10: 2, SiglaUF, Sigla, Y, Y, Y, CHAR, 2, Chave Estrangeira da Tabela Estado.
 - Row 11: 3, Cidade, , Y, , Y, CHAR, 50, Nome da Cidade.
- Empty Rows:** Rows 12 and 13 are empty.

Fonte: captura de tela do Microsoft Excel, elaborada pela autora.



Pesquise mais

Saiba mais sobre a Linguagem Unificada de Modelagem, a UML, nas páginas 505 a 517 do livro:

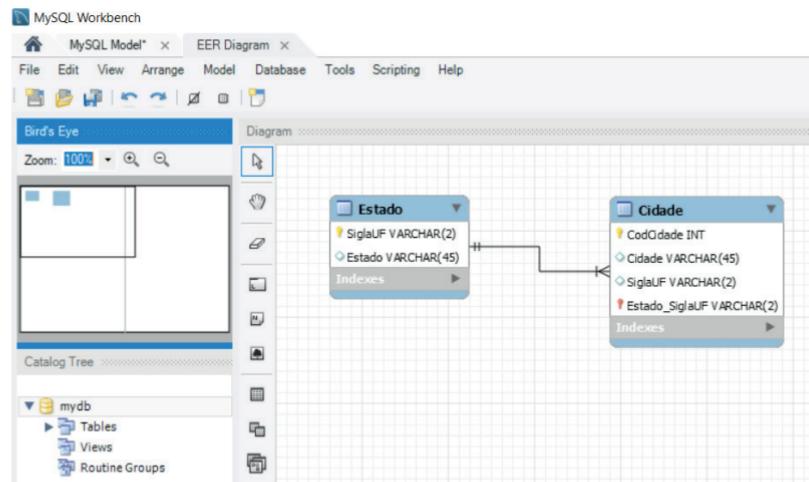
LEE, R. C.; TEPFENHART, W. M. **UML e C++**: guia prático de desenvolvimento orientado a objeto. São Paulo: Pearson eBooks, 2001.

Esse livro também encontra-se disponível em: <<https://biblioteca-virtual.com/detalhes/eds/cat05587a/kae.9788534613644>>. Acesso em: 14 ago. 2018.

O MySQL Workbench® é de propriedade da empresa Oracle e é uma ferramenta CASE gratuita que gera scripts para o SGBD MySQL. O foco dela é a modelagem física do banco de dados, acelerando

o processo de criação da base de dados. O site para download é: <<https://www.mysql.com/products/workbench/>>, acesso em: 14 ago. 2018. Observe na Figura 3.15 que o relacionamento utiliza a notação do "Pé-de-Galinha".

Figura 3.15 | Exemplo ferramenta CASE MySQL Workbench



Fonte: captura de tela do MySQL Workbench, elaborada pela autora.



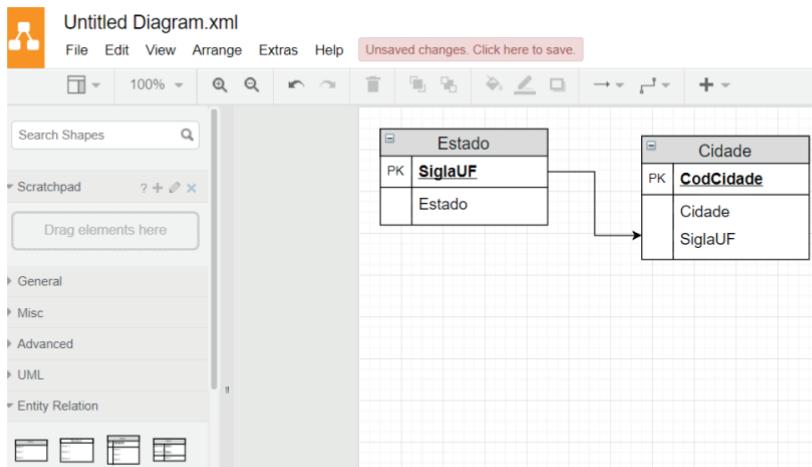
Exemplificando

Para criar o diagrama da Figura 3.15, siga o passo a passo a seguir:

1. Vá ao Menu e escolha *File* → *New Model* → *Add New Diagram*.
2. Para inserir uma tabela, aperte a letra **T** e pressione *enter*. Dê dois cliques na tabela para abrir o editor de campos e adicione-os, juntamente com seus tipos, e insira as chaves primárias e estrangeiras.
3. Para definir os relacionamentos entre tabelas, escolha as ligações que estão na barra lateral esquerda do diagrama criado.
4. Depois de criado o diagrama, é possível exportá-lo para um script SQL ou até inseri-lo diretamente no SGBD. Acesse o menu *File* → *Export* → *Forward Engineer SQL Create Script*.

Ferramentas online também são uma opção para a criação de modelos gráficos de desenvolvimento de software. Draw.IO é uma ferramenta fácil, que tem como requisito estar conectado à Internet. Está disponível em: <<https://www.draw.io/>>, acesso em: 14 ago. 2018. A ferramenta ainda disponibiliza uma grande quantidade de *templates* de modelos para servirem de exemplos de modelagem. Na Figura 3.16, foi realizada a modelagem com a notação de setas. O uso da ferramenta é bem intuitivo: ao lado esquerdo estão as categorias de diagrama, então, você pode acessar a *Entity Relation* para ter acesso às opções de tabela e clicar em uma delas para adicioná-la a área de trabalho. Os campos podem ser editados diretamente na tabela adicionada. Para relacionar as tabelas, escolha as opções de seta e ligue a chave primária diretamente sobre a chave estrangeira.

Figura 3.16 | Exemplo ferramenta CASE online Draw.IO



Fonte: captura de tela do Draw.IO, elaborada pela autora.



Um *template* é um exemplo ou um modelo que pode servir de base de criação para algum determinado objetivo. Possui uma estrutura predefinida que facilita o desenvolvimento e a criação do conteúdo a partir de algo que já foi construído previamente.

Outra ferramenta CASE online é a Lucidchart, disponível em: <<https://www.lucidchart.com/>>, acesso em: 14 ago. 2018. No site, é possível criar gratuitamente um modelo com até 60 objetos, acima disso, é necessário pagar. Essa ferramenta é ideal para pequenos e médios projetos. Veja um exemplo na Figura 3.17, em que foi utilizada a notação “Pé-de-Galinha”. Na ferramenta, você utilizará os recursos da categoria padrão e entidade-relacionamentos. Na categoria de entidade-relacionamentos, você deverá selecionar o modelo de tabela que desejar e inserir os campos diretamente na figura da tabela. Para criar o relacionamento entre as tabelas, use a ferramenta de “seta” na categoria, ligando sempre a chave primária com a sua chave estrangeira (na outra tabela).

Figura 3.17 | Exemplo ferramenta CASE online Lucidchart



Fonte: captura de tela do Lucidchart, elaborada pela autora.

Um dos destaques dessa ferramenta é a possibilidade de gerar scripts para os seguintes Sistemas de Gerenciadores de Banco de Dados: MySQL, PostgreSQL, SQL Server e Oracle, conforme pode ser observado na Figura 3.18. Para gerar o script, acesse a opção “Exportar” no menu esquerdo do software, na categoria de entidade-relacionamentos. Ao selecioná-la, você terá acesso à tela da Figura 3.18 e poderá escolher para qual banco deseja gerar o script. Também será possível exportar o DER para uma imagem, para isso, selecione Arquivo → Baixar como → e escolha a opção que desejar.

Figura 3.18 | Exemplo exportando para SQL no Lucidchart

Exporte Para SQL

Qual sistema de gerenciamento de banco de dados (DBMS) você está usando?

Vamos alterar a sintaxe dos nossos comandos SQL para corresponder ao seu DBMS.

- MySQL
- PostgreSQL
- SQL Server
- Oracle

Exporte para SQL

Copie estes comandos para aplicá-los em seu próprio banco de dados. Antes de usar os comandos gerados, pode ser necessário adicionar tipos de dados, índices e chaves estrangeiras.

Copiar para a área de transferência

```
CREATE TABLE [Cidade] (
    [CodCidade] <type>,
    [Cidade] <type>,
    [SiglaUF] <type>,
    PRIMARY KEY ([CodCidade])
);

CREATE INDEX [FK] ON [Cidade] ([SiglaUF]);

CREATE TABLE [Estado] (
    [SiglaUF] <type>,
```

Concluir

Fonte: captura de tela do Lucidchart, elaborada pela autora.



Exemplificando

Um exemplo de um script gerado automaticamente após criar a tabela Estado traz o código em SQL totalmente pronto, observe:

```
CREATE TABLE Estado (
    Sigla CHAR(2) NOT NULL,
    Estado CHAR(40)
);
```

Um diagrama de entidade-relacionamentos geralmente possui muitas entidades, e utilizar uma ferramenta CASE para criar as tabelas e relacionamentos facilita muito o trabalho do desenvolvedor. Agora que já conhecemos algumas ferramentas CASEs disponíveis, vejamos um estudo de caso para que possa ser implementado em uma das ferramentas:

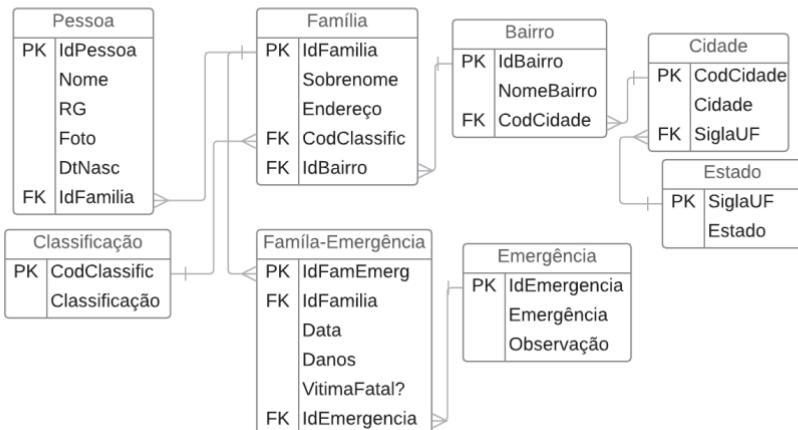
Após diversas situações de emergência devido a inundações, geadas e vendavais, a defesa civil de uma cidade no sul do país precisa de um banco de dados com informações sobre as vítimas, para poder controlar as emergências que ocorreram e a quantidade de pessoas envolvidas. Uma emergência pode atingir cidades vizinhas e a defesa civil pode intervir em outras cidades.

Os requisitos para realizar o banco de dados são os seguintes:

- É necessário anotar o endereço da família em situação de risco.
- Deve haver um cadastro de cada membro de uma família. Os dados pessoais e uma foto são fundamentais.
- A família poderá ser classificada por tipo: em risco, alto risco, extremo risco.
- Deve haver uma tabela para cadastro das emergências classificadas em: inundações, vendaval, desmoronamentos, entre outras opções.
- Uma família pode sofrer com diversas emergências e uma emergência pode atingir diversas famílias. É necessário armazenar a data da emergência e os danos causados por ela.

Para criar o diagrama de entidade-relacionamentos, conforme a Figura 3.19, foi utilizada a ferramenta CASE online Lucidchart. Acesse o site <<https://www.lucidchart.com/>> e selecione o local em que deseja salvar o diagrama. Depois, escolha a opção Arquivo → Novo Documento e selecione Entidade Relacionamento (ERD). Escolha a figura para a tabela (pode ser a que você achar mais interessante), adicione os campos diretamente da tabela, dê dois cliques nela e digite o nome e os campos da tabela. Para relacionar: crie antes todas as tabelas. No menu superior, escolha as setas para relacionar a chave primária com a sua chave estrangeira.

Figura 3.19 | Estudo de caso com Lucidchart



Fonte: elaborada pela autora.



Refletia

Observe o DER da Figura 3.19. Na tabela Família-Emergência há o campo Danos. Os danos podem se repetir para outras famílias? Como poderemos impedir que essa repetição seja muito frequente no banco de dados?

Utilizar ferramentas CASEs para a modelagem gráfica de banco de dados é muito importante no processo de desenvolvimento de qualquer software, com destaque na apresentação para o cliente, pois esses diagramas, após alinhados, sempre causam boa impressão. Para a equipe de desenvolvimento, o destaque fica com a possibilidade de uma melhor comunicação (todos podem entender o que está sendo modelado) e agilidade de gerar os scripts que poderão ser utilizados para criar o banco de dados fisicamente em um SGBD.

Chegando à nossa última unidade, veremos que não adianta ter somente um modelo gráfico bonito, ele precisa estar 100% correto!

Sem medo de errar

Você conheceu nesta seção as ferramentas que poderão auxiliar na modelagem de dados. As ferramentas CASEs podem facilitar e acelerar o trabalho de desenvolvimento de softwares. E agora será necessário aplicar esse conhecimento para o banco de dados do salão de beleza. Nas duas últimas seções desta unidade, você trabalhou na modelagem do banco de dados e agora vai usar uma ferramenta CASE para modelar as tabelas Cliente, Serviço, Funcionário, Agenda, Atendimento, Fornecedores, Produtos, Cliente VIP, Cliente Casual e Cliente Mensal.

O salão de beleza da dona Áurea está com problemas, então, será necessário manter um histórico de atendimento de seus clientes. Isso permitirá uma análise do perfil dos clientes, determinação dos serviços mais procurados no salão e verificação dos funcionários mais requisitados. Para isso, precisaremos resolver as seguintes situações.

Primeiro, devemos verificar como poderemos manter um histórico dos atendimentos de cada cliente. A resposta é: criando mais uma tabela chamada de Histórico. Observe: um Cliente poderá solicitar vários Serviços e um Serviço pode ser solicitado por diversos Clientes, então, ficando um relacionamento N para N. A tabela Histórico pode ser a tabela associativa entre as tabelas Cliente e Serviço.

Agora fica a dúvida: qual ferramenta CASE utilizar para criar o diagrama de entidade-relacionamentos? A solução é simples: aquela que você preferir, a que tiver mais recursos ou estiver mais acessível. Porém, e o dicionários de dados? Poderemos criar de forma automática? Sim, poderemos! A ferramenta CASE Astah Professional tem essa possibilidade. Pesquise outras ferramentas que possuam essa funcionalidade e que você possa utilizar.

Para ajudar na solução, as seguintes tabelas precisam estar no diagrama entidade-relacionamentos: Cliente, Tipos de Clientes, Serviço, Funcionário, Agenda, Atendimento, Fornecedores, Produtos, Cidade e Estado.

Utilize uma ferramenta CASE para criar o DER do salão de beleza da Dona Áurea. Estabeleça padrões de criação dos campos das tabelas, depois, crie o dicionário completo de todas as tabelas. Apresente em forma de slide para seu cliente.

Criação do DER para horta comunitária

Descrição da situação-problema

Os moradores de seu bairro possuem uma horta comunitária em um terreno cedido pela prefeitura. Cada morador que demonstrou real interesse em cultivar hortaliças ganhou um lote. Cada lote tem um tamanho diferenciado: famílias maiores ganham espaços maiores, mas isso não é uma regra, visto que é possível que o morador queira um lote pequeno, somente para plantar temperos. Cada um poderá plantar o que desejar, desde que o item seja cadastrado e aprovado (esse controle se faz necessário para evitar incômodos futuros). A única regra é: proibição total de qualquer defensivo químico. A equipe que gerencia e fiscaliza os lotes precisa de um banco de dados:

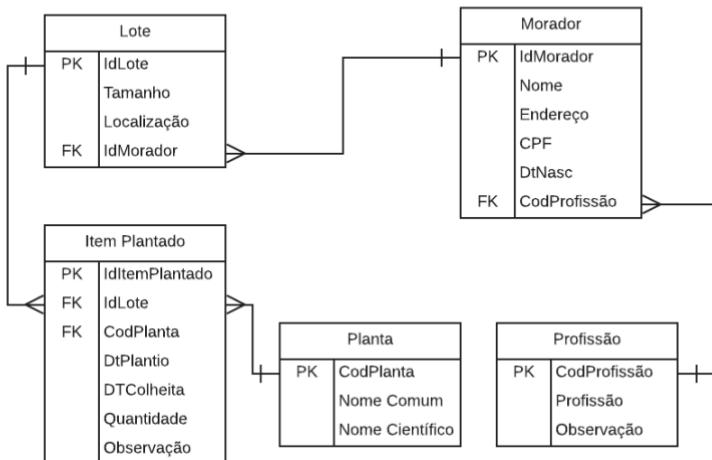
- Para saber quem é o morador responsável por cada lote, sendo que um morador pode ser responsável por mais de um lote. É importante saber a profissão de cada pessoa (isso ajudará em caso de um morador precisar de ajuda de alguém de determinada profissão).
- Assim que o residente plantar algo no lote, ele deve informar para um controle de itens de hortaliças plantadas (que devem ser cadastradas).
- A data da plantação e da colheita também será controlada, a fim de ajudar a manter uma estatística de controle da horta.

Como você poderá ajudar a criar uma modelagem do banco de dados?

Resolução da situação-problema

Para resolver essa situação-problema, devemos nos atentar às solicitações da equipe que gerencia a horta. Com isso, já podemos identificar as seguintes tabelas: Morador, Profissão, Lote, Plantas e Itens plantados. Podemos utilizar uma ferramenta CASE para criar o DER. Observe na Figura 3.20 o DER elaborado na ferramenta CASE online Lucidchart.

Figura 3.20 | DER horta comunitária



Fonte: elaborada pela autora.

Lembre-se de que o PK representa a chave primária e o FK representa a chave estrangeira da tabela.

Faça valer a pena

1. Uma das vantagens de utilizar uma ferramenta CASE para a modelagem de banco de dados é a possibilidade da criação automática de scripts. Este procedimento acelera o trabalho de administrador do banco de dados, e os comandos dos scripts são executados todos juntos, criando tabelas e relacionamentos, além de poderem inserir dados no banco de dados.

Assinale a alternativa correta sobre a finalidade dos scripts em uma ferramenta CASE para banco de dados.

- a) Os scripts são os desenhos das tabelas em uma ferramenta CASE.
- b) Os scripts servem exclusivamente para a definição automática das chaves primárias e estrangeiras de uma tabela.
- c) O script é a linguagem de programação do SGBD utilizada para criar graficamente o modelo físico do banco de dados.
- d) O script é a versão orientada a objetos que pode ou não ser utilizada como alternativa de modelagem de um banco de dados.
- e) Os scripts geram todos os comandos em SQL de criação de tabelas, campos e chaves.

2. Utilizar ferramentas CASEs para a modelagem gráfica de banco de dados é muito importante no processo de desenvolvimento de qualquer software. Esta ferramenta permite que os diagramas criados tenham um padrão, o que ajuda no processo de comunicação e na qualidade final do projeto de banco de dados.

Assinale a alternativa correta que aponta como a ferramenta CASE pode ajudar no processo de modelagem de um banco de dados.

- a) Ajuda na programação dos relatórios do banco de dados.
- b) Pode ajudar na criação gráfica das páginas HTML que vão apresentar o banco de dados.
- c) Deixa disponíveis todas as informações que o usuário armazenará no banco de dados.
- d) Pode ajudar na padronização da modelagem do banco de dados.
- e) Ajuda no processo de coleta de informação, acelerando o mecanismo de acesso aos dados dos clientes.

3. As ferramentas CASEs (*Computer Aided Software Engineering* ou Engenharia de Software Auxiliada por Computador) apresentam uma série de serviços que auxiliam no desenvolvimento de software e podem minimizar o tempo de desenvolvimento do software modelado.

Assinale a alternativa correta referente às ferramentas CASE's para modelagem de banco de dados.

- a) Todas a ferramentas CASEs são freeware, o principal motivo de ter sido usado nos processos de modelagem de banco de dados.
- b) A ferramenta MySQL Workbench, por pertencer à empresa Oracle, é proprietária e não permite a criação de modelagens grátis, além de seus scripts serem exclusivos para o SGBD ORACLE.
- c) Existe a versão freeware, como o MySQL Workbench, e diversas ferramentas online que podem ou não ser freeware, mas a maioria permite a criação de pequenos projetos para a utilização da ferramenta.
- d) As ferramentas online possuem a desvantagem de não gerar scripts, fator que está dificultando o uso destas ferramentas nas modelagens de banco de dados.
- e) Todas as ferramentas CASEs são pagas e somente empresas podem utilizar (após cadastro realizado e aprovado), o que dificulta o acesso e o uso deste tipo de ferramenta por parte dos desenvolvedores de software.

Referências

- ABREU, M. P.; MACHADO, F. N. R. **Projeto de banco de dados:** uma visão prática. 16. ed. rev. e atual. São Paulo: Érica, 2009.
- BOOCH, G., RUMBAUGH, J., JACOBSON, I. **UML, Guia do Usuário.** 2. ed. Rio de Janeiro: Editora Elsevier, 2005.
- COUGO, P. **Modelagem conceitual e projeto de bancos de dados.** Rio de Janeiro: Elsevier: 1997.
- CORONEL, C.; ROB, Peter. **Sistema de banco de dados:** projeto, implementação e administração. São Paulo: Cengage Learning, 2011.
- DATE, C. J. **Introdução a sistemas de bancos de dados.** Rio de Janeiro, RJ: Elsevier, 2003.
- FOWLER, M. **UML essencial:** um breve guia para a linguagem-padrão de modelagem de objetos. 3. ed. São Paulo: Bookman, 2004.
- GUIMARÃES, C. C. **Fundamentos para bancos de dados:** modelagem, projeto e linguagem SQL. Campinas: Ed. da Unicamp, 2003.
- HEUSER, C. A. **Projeto de Banco de Dados.** 6. ed. Porto Alegre: Bookman, 2011.
- KORTH, H. F.; SILBERSCHATZ, A.; SUDARSHAN, S. **Sistema de banco de dados.** 5. ed. São Paulo: Makron, 2006.
- MEDEIROS, E. S. **Desenvolvendo software com UML 2.0:** definitivo. São Paulo: Pearson Makron Books, 2004. 264 p.
- MIZRAHI, V. V. **Treinamento em linguagem C++.** 2. ed. São Paulo: Pearson Prentice Hall, 2008.
- NAVATHE, S. B.; RAMEZ, E. **Sistemas de banco de dados.** 4. ed. São Paulo: Addison Wesley, 2005.

Normalização de dados

Convite ao estudo

Bem-vindo, chegamos a nossa última unidade.

A realidade de uma analista de sistemas é projetar sistemas para clientes que muitas vezes nem sabem ao certo o que desejam. Dessa forma, esse profissional deve ter uma visão geral sobre os diversos segmentos de mercado. Também é certo que o que o cliente deseja não venha numa descrição perfeita e com as entidades já definidas por um estudo de caso. Diante disso, o analista precisará visitar a empresa e realizar entrevistas com diversos funcionários. Além disso, é importante recolher cópias de documentos que são utilizados na empresa, como a nota fiscal, a requisição de compras, o controle de estoque, a ficha de contratação de empregados, entre outros. Esses documentos trazem informações preciosas e que muitas vezes passam despercebidas até pelo próprio cliente que está solicitando o software.

Nesta unidade utilizaremos documentos para projetar o banco de dados. Conhecer o conteúdo de cada seção desta unidade é primordial a fim de se ter a habilidade necessária para a modelagem a partir de documentos e assim poder aplicar a normalização dos dados.

Você está começando a ficar famoso com a sua empresa, sendo recomendado pelos clientes anteriores, satisfeitos com o software modelado que você realizou. E, por causa dessas recomendações, um novo cliente entrou em contato, pois precisará de um banco de dados. Um clube de futebol regional, Unidos Venceremos Futebol Clube, está procurando se modernizar, querem entrar para a segunda divisão. Com um banco de dados, realizado por você, o time poderá organizar-se para os campeonatos, mantendo um histórico dos jogos e dos jogadores.

Após algumas visitas à empresa, foram recolhidos vários documentos. Um documento em especial será utilizado para o controle de informações dos jogadores, jogos e resultados. Você precisará aplicar a normalização de dados para encontrar entidades e relacionamentos que podem estar escondidos em documentos, aplicando, dessa forma, os conhecimentos em algo realmente prático e que é uma rotina comum para um analista de sistemas.

Vamos lá?

Bons estudos.

Seção 4.1

Normalização de dados na computação

Diálogo aberto

Em um projeto de banco de dados existe uma regra fundamental: nunca misturar assuntos diferentes em uma mesma tabela. Precisamos lutar contra dois aspectos chaves: redundância de dados e inconsistência de dados. Para lidar com esses fundamentos, nesta seção, apresentaremos os conceitos sobre normalização de dados e dependência funcional. Um profissional da área de TI precisa conhecer os conceitos desta seção e, mais do que isso, aplicá-los em suas modelagens nos bancos de dados.

Para aplicar os conhecimentos adquiridos nesta seção realizaremos uma modelagem a partir de um documento. Você precisará modelar um banco de dados a partir do documento fornecido pelo seu mais novo cliente: Unidos Venceremos Futebol Clube. Veja na Figura 4.1 a atual ficha usada pelo clube.

Figura 4.1 | Ficha usada pelo Unidos Venceremos Futebol Clube

Unidos Venceremos Futebol Clube

JOGO Nº 114 DATA: 18/06/2018
ADVERSÁRIO: Cacoalense de RO
ESTÁDIO: ARENA DA FLORESTA CIDADE: Rio Branco
UF: AC
CAMPEONATO: Florestal Brasileiro
TÉCNICO:
PLACAR FINAL: UVFC _____ x _____ ADVERSÁRIO

JOGADORES:

Nº CAMISA	NOME	Nº de Gols na Partida

PRINCIPAIS EVENTOS DO JOGO:

TEMPO:	OCORRIDO:
2 min	Gol Nossa Time
3 min	Gol Adversário
15 min	Pênalti Nossa
16 min	Expulsão Time Adversário

HISTÓRICO DAS ÚLTIMAS GOLEADAS:

ADVERSÁRIOS	Qtde Gols Adv	Qtde Gols Casa
Pirambu - SE	1	2
Guarabira - PB	0	1
Luverdense - MT	0	2
Social - MG	2	3
Coroatá - MA	2	2

Fonte: elaborada pela autora.

O clube de futebol quer armazenar as fichas de controle de cada jogo para manter um histórico dos jogos realizados e da atuação dos seus jogadores.

Nessa etapa você precisará responder:

- Quais os possíveis campos que o documento apresenta?
- Podemos identificar alguma tabela no documento?

Criar um banco de dados a partir de documentos é algo muito comum em qualquer modelagem de banco de dados. É muito mais fácil para o cliente fornecer uma cópia de seus documentos do que ficar detalhando passo a passo o funcionamento de uma rotina de trabalho. Essa atividade é prática e usual, aproveite!

Bons estudos.

Não pode faltar

A modelagem de dados tem como sua essência mais pura o refinamento de processos. Isso significa que sempre podemos melhorar o que já foi realizado. Uma vez criado o Diagrama Entidade-Relacionamento (DER) já podemos implementá-lo em um Sistema Gerenciador de Banco de Dados? Não. O correto é revisar, procurar imperfeições e melhorar, prevendo possíveis problemas. Em um banco de dados o maior problema é a redundância, pois ela pode causar danos enormes e pode acontecer disso ser notado somente quando o banco de dados já estiver sendo usado pela empresa. Os danos que a redundância pode causar e que geram mais problemas é a repetição da mesma informação em várias tabelas, ocasionando, além de duplicidade, possíveis erros em relatórios.



Exemplificando

Na Figura 4.2 podemos observar três tabelas: a do cliente, a do fornecedor e a do funcionário. Repare que todas possuem um campo em comum: cidade. Se deixarmos do jeito que está, a mesma cidade pode ser cadastrada três vezes e em tabelas diferentes.

Figura 4.2 | Exemplo de redundância

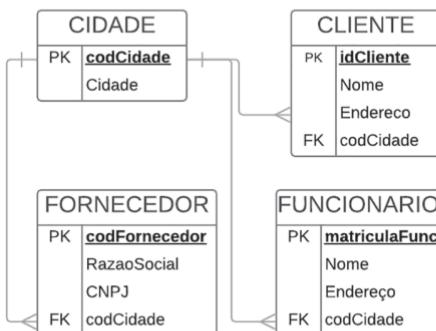
CLIENTE		FORNECEDOR		FUNCIONÁRIO	
PK	<u>idCliente</u>	PK	<u>codFornecedor</u>	PK	<u>matriculaFunc</u>
	Nome Endereço CidadeResid		RazãoSocial CNPJ Cidade		Nome CPF CidadeNasc

Fonte: elaborada pela autora.

E o que poderia ser pior que a repetição vista na Figura 4.2? O problema de inconsistência no banco de dados, que nada mais é do que a mesma informação sendo cadastrada de forma errônea, por exemplo, a cidade de Joinville pode ser cadastrada como: "Joinville", "Joinvile" ou ainda "Jlle". São três formas de cadastros que darão problemas quando precisarmos consultar, no banco de dados, informações como: "mostrar todos os funcionários da cidade de Joinville". Caso a cidade tenha sido cadastrado de forma errada, esses funcionários ficarão fora do relatório gerado.

Quando o usuário cobra uma informação em uma tabela não podemos, por exemplo, prever que ele irá inserir o seu nome errado no banco de dados. Mas, no caso da cidade é mais simples: criamos uma tabela chamada *Cidade*, relacionando as tabelas que precisam dessa informação, observe a Figura 4.3 que ilustra o correto relacionamento entre a tabela *Cidade* e as demais que a utilizam.

Figura 4.3 | Resolvendo a redundância e a inconsistência entre as tabelas



Fonte: elaborada pela autora.

Às vezes é interessante manter uma informação redundante no banco de dados. Por questões de desempenho de alguma pesquisa ou software a redundância pode ocorrer, fato esse que é chamado de **redundância controlada** e é recomendado quando o campo recebe uma grande quantidade de consultas e poucas alterações. Um exemplo é uma tabela de nota fiscal, o campo “valor_total_da_nota”, que poderia ser obtido automaticamente, como o resultado da soma de todos os itens vendidos e multiplicados pelo seu preço. Entretanto, é comum criarmos manualmente o campo “valor_total_da_nota” para evitar que, caso haja alteração do preço do produto vendido, o valor da nota não seja alterado (causando problemas contábeis). É uma redundância que sabemos que existe, mas é necessária.

O Quadro 4.1 apresenta um outro exemplo de redundância controlada. Observe, na tabela *Funcionário* há o código do departamento (*CodDepartamento*) e o nome do departamento (*NomeDepartamento*), causando a redundância e que pode ser ideal para um determinado software, mas para outro software pode ser uma péssima ideia. É nessa hora que o projetista do banco de dados ou o analista de sistemas deverá decidir se realmente será benéfico deixar a redundância acontecer.

Quadro 4.1 | Exemplo tabela com redundância controlada

Tabela: funcionário				
Matricula	Nome	Valor_Hora	CodDepartamento	NomeDepartamento
13467-4	Marco Antonio Liz	R\$ 18,22	DP - 450	Expedição
34562-5	Anna Pietro	R\$ 18,22	DP - 450	Expedição
76321-0	Carlos Werner	R\$ 13,50	DP - 450	Expedição
58309-3	Sandro Lopez	R\$ 28,70	DA - 780	Contabilidade

 REDUNDÂNCIA

Fonte: elaborado pela autora.

Segundo Coronel e Rob (2011), a normalização é uma técnica para avaliar e corrigir estruturas e tabelas ao modo de tornar mínimas as redundâncias de dados, reduzindo assim as chances de haver problemas. Normalizar um banco de dados é aplicar regras para todas as suas tabelas, com os objetivos, além de reduzir a redundância e eliminar campos que não dizem respeito à tabela. Um bom exemplo disso é uma tabela que armazena informações sobre a cidade. De forma alguma poderemos armazenar informações sobre um funcionário dentro da tabela.

Podemos listar alguns objetivos e vantagens da normalização de um esquema de banco de dados:

- Diminuição de dados repetitivos deixando o banco de dados mais compacto.
- Aumento da performance no Sistema Gerenciador de Banco de Dados.
- Armazenamento dos dados de forma lógica.
- Facilidade na criação de consultas.
- Permite a concatenação de índices (chaves) de acordo com a quantidade de tabelas envolvidas.
- Facilidade na manutenção do banco de dados.

Normalização de dados é um processo rígido e formal que deve ser seguido passo a passo examinando os campos de uma tabela, a fim de evitar irregularidades observadas na inclusão, exclusão e alteração de registros. Observe o Quadro 4.2 que demonstra a tabela *Produto* não normalizada, observe que nos campos *TipoProduto* e no *Fornecedor* a mesma informação foi inserida erroneamente, entre os registros.

Quadro 4.2 | Exemplo tabela *Produto* não normalizada

Tabela: produto					
idProd	Produto	Preço	TipoProduto	CodForn	Fornecedor
1415	Sabão	R\$ 4,71	Limpeza	708	Tem Tudo
7841	Álcool	R\$ 5,80	Limpezas	708	Tem de Tudo
8543	Arroz	R\$ 7,84	Grão	516	Compra Boa
9124	Trigo	R\$ 5,45	Grãos	516	Compra B.

Fonte: elaborado pela autora.

Uma das regras básicas da normalização é verificar se determinado campo realmente pertence à tabela. No caso do Quadro 4.2, há os campos:

- *TipoProduto*: podemos criar uma tabela para armazenar os tipos de produtos.
- *CodForn* e *Fornecedor*: devemos criar uma tabela chamada fornecedor.

Na sequência de Quadros 4.3, 4.4 e 4.5, é mostrado como devem ser as tabelas normalizadas. O Quadro 4.3 mostra que foram retirados da tabela *Produto* alguns campos, para que ela fique normalizada. As informações que permaneceram e que agora fazem parte de outra tabela ficaram como chaves estrangeiras das tabelas: *TipoProduto* e *Fornecedor*, observe:

Quadro 4.3 | Exemplo tabela *Produto* normalizada

Tabela: <i>produto</i>				
idProd	Produto	Preço	idTipoProduto	CodFornecedor
1415	Sabão	R\$ 4,71	23	708
7841	Álcool	R\$ 5,80	23	708
8543	Arroz	R\$ 7,84	18	516
9124	Trigo	R\$ 5,45	18	516

Fonte: elaborado pela autora.

No Quadro 4.4, a tabela *TipoProduto* foi criada e será nesta tabela que todos os tipos de produtos deverão ser criados, para depois serem escolhidos na tabela *Produto* (via chave estrangeira).

Quadro 4.4 | Exemplo tabela *TipoProduto* normalizada

Tabela: <i>TipoProduto</i>	
idTipoProduto	TipoProduto
23	Limpeza
18	Grãos

Fonte: elaborado pela autora.

No Quadro 4.5, a tabela fornecedor foi criada para receber as informações dos fornecedores e, por meio da chave estrangeira na tabela *Produto*, vincularemos o *Fornecedor* ao *Produto*, evitando que o nome do mesmo fornecedor seja cadastrado de forma errada.

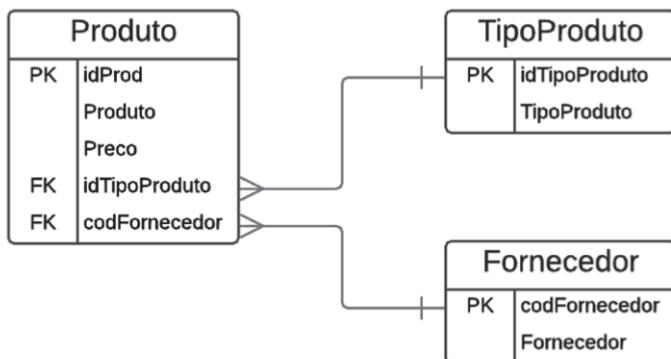
Quadro 4.5 | Exemplo tabela *Fornecedor* normalizada

Tabela: fornecedor	
codFornecedor	Fornecedor
708	Tem Tudo
516	Compra Boa

Fonte: elaborado pela autora.

Geralmente os analistas de sistemas já criam as tabelas *TipoProduto* e *Fornecedor* automaticamente num modelo de banco de dados. Mostramos as tabelas dos quadros supracitados para dar ênfase à necessidade da normalização, principalmente quando estamos começando na modelagem de banco de dados. Na Figura 4.4 você poderá observar o Diagrama de Entidade-Relacionamento entre as três tabelas: *Produto* – *Fornecedor* e *TipoProduto*. No diagrama você pode observar, que um produto é classificado a um determinado tipo de produto, mas um determinado tipo de produto poderá estar relacionado com muitos produtos. Para esse exemplo um produto possui somente um fornecedor e o fornecedor poderá fornecer muitos produtos.

Figura 4.4 | DER *Produto* – *TipoProduto* - *Fornecedor*



Fonte: elaborada pela autora.

Korth, Silberschatz e Sudarshan (2012) afirmam que as primeiras técnicas de normalização foram criadas em 1972 por Edgar Frank Codd. Após ter dado o primeiro passo, Codd propôs, junto de Raymond Boyce, um novo significado, que ficou conhecido como Forma Normal Boyce-Codd (ou FNBC). Todas elas se baseiam na dependência funcional entre os atributos de uma entidade do banco de dados e nas chaves primárias. Segundo Coronel e Rob (2011), as formas normais mais populares, são:

- A primeira forma normal ou 1FN.
- A segunda forma normal ou 2FN.
- A terceira forma normal ou 3FN.
- A quarta forma normal ou 4FN.

Heuser (2001) afirma que a forma normal é uma regra que deve ser satisfeita por uma entidade para que ela seja avaliada como uma tabela “projetada com exatidão”. São várias formas normais, com regras que vão se tornando mais rigorosas, com o objetivo de averiguar nas tabelas a existência de redundância ou dependências funcionais. No entanto, pelo menos quatro formas normais (como supracitadas) são consideradas essenciais para a construção de um bom projeto de banco de dados.



Refletia

Aplicando as formas normais em um projeto de banco de dados e respeitando suas regras, será possível ter um banco de dados mais coeso e com possibilidades de sucesso. Mas será que estará livre de erros? Quais erros podem ocorrer nas últimas fases de um projeto de banco de dados?

Podemos descrever a normalização como um processo de organizar os dados em um modelo de banco de dados. Para realizar essas tarefas são criadas tabelas e estabelecidos relacionamentos entre tabelas de acordo com as regras empregadas para proteger os dados e para tornar o banco de dados mais maleável se adaptando

para acabar com a redundância e a dependência inconsistente (campo errado na tabela errada) dos dados que serão armazenados no banco de dados.



A redundância de dados desperdiça espaço de armazenamento, com informações duplicadas, e geralmente ocasiona problema de manutenção. Caso haja alteração e os dados estejam em várias tabelas, por exemplo, será necessário procurar esse dado e alterá-lo, gerando uma perda significativa de tempo na manutenção do banco de dados.

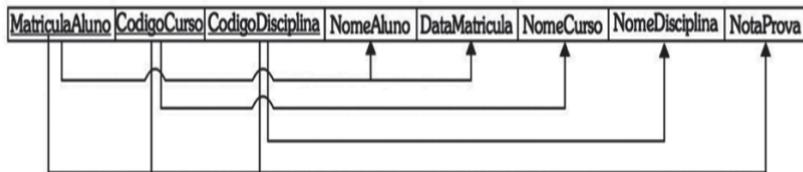
Para compreender as formas normais, que serão apresentadas nas próximas seções, precisamos primeiramente compreender o significado de dependência funcional, que nada mais é consistir em uma restrição entre dois ou mais conjuntos de atributos de uma mesma tabela ou relacionamento, conforme afirma Alves (2014).

Conforme Coronel e Rob (2011) podemos explicar a dependência funcional com o apoio na teoria de conjuntos, dados dois conjuntos de atributos X e Y de uma entidade pode-se afirmar que:

- Y é dependente funcional de X ou
- X determina Y ou
- Y depende de X, logo
- Podemos representar a dependência funcional como:
 $X \rightarrow Y$

A dependência $X \rightarrow Y$ ocorrerá se a cada valor de X estiver associado um e somente um valor de Y. Um exemplo de dependência funcional é a tabela *Aluno*. Nela temos a matrícula e o nome do aluno. O nome do aluno depende diretamente da matrícula do aluno. A Figura 4.5 demonstra o esquema de relacionamento de uma tabela. Observe que os três primeiros campos sublinhados: *MatriculaAluno*, *CodigoCurso* e *CodigoDisciplina* representam a chave primária da tabela (no caso, uma chave composta ou concatenada). Nesse exemplo o conjunto dos três campos formam um índice único (formando a chave primária).

Figura 4.5 | Tabela exemplificando dependência funcional



Fonte: Alves (2014, p. 109).

A Figura 4.5 apresenta a notação de dependências funcionais dos campos, observe:

- Campo *MatriculaAluno*: possui como dependência funcional os campos *NomeAluno* e *DataMatricula*.
- Campo *CódigoCurso*: possui como dependência funcional o campo *NomeCurso*.
- Campo *CódigoDisciplina*: possui como dependência funcional o campo *NomeDisciplina*.
- Campos *MatriculaAluno*, *CódigoCurso*, *CódigoDisciplina*: determinam o valor da *NotaProva*.

A dependência funcional pode ser classificada em: transitiva ou indireta, total ou parcial. A dependência funcional transitiva acontece quando um determinado campo da tabela, além de depender da chave primária da tabela, depende também de outro campo ou de outros campos que são integrantes da mesma tabela. A dependência transitiva ocorre quando Y depende de X e Z depende de Y; logo, Z também depende de X. Podemos representar esse tipo de dependência como: $X \rightarrow Y \rightarrow Z$.

No Quadro 4.6, observe a tabela *Aluno* para um sistema de uma universidade. Temos a necessidade de guardar a escola de origem do aluno e o endereço dessa escola. O endereço da escola de origem é dependente da escola de origem, que depende da chave primária que é a matrícula do aluno. Esse é um exemplo de tabela que precisará sofrer um processo de normalização para resolver essa dependência entre os campos com esse tipo de dependência.

Quadro 4.6 | Dependência funcional transitiva ou indireta

Tabela: aluno			
Matrícula	Nome	Escola de Origem	Endereço da Escola Origem
1407	Lucca Lewis	E.B. Amigos dos Estudos	R. das Montanhas, 450.
5789	Karyn Cruz	E.B. Estudar é Preciso	R. Ventos Fortes, 715.
1587	Jane Flores	E.B. Futuro Melhor	R. Pardal Solitário, 957.

Fonte: elaborado pela autora.

A dependência funcional total ou completa (como também é conhecida) ocorre quando um atributo que não faz parte da chave primária depende diretamente de todos os outros atributos que fazem parte da chave primária. Sempre ocorre quando a tabela possui chaves concatenadas (mais de uma chave primária). No Quadro 4.7, há um exemplo em que os campos *Cidade* e o *Bairro* são chaves concatenadas e o campo *Fiscal Responsável* depende dos dois campos para “existir”. A dependência funcional total pode ser representada da seguinte forma: *Cidade, Bairro* → *Fiscal Responsável*.

Quadro 4.7 | Dependência Funcional Total

Tabela: fiscalização		
Cidade	Bairro	Fiscal Responsável
Blumenau	Garcia	Werner Klaus
São Paulo	Ibirapuera	Antônio Luiz
São Paulo	Bom Retiro	Cristina Laís

Fonte: elaborado pela autora.

A dependência funcional parcial ocorre quando um campo ou atributo que não faz parte da chave primária tem dependência funcional de apenas alguns dos atributos que fazem parte da chave primária. No Quadro 4.8 observe que a tabela *Medição da Temperatura*, possui três chaves primárias: UF, Cidade e Região. O campo *Temperatura* possui uma dependência funcional parcial apenas de parte das chaves primárias visto que se o campo *Região* não existisse ou se fosse removido não afetaria o campo *Temperatura*.

Quadro 4.8 | Dependência funcional parcial

Tabela: medição da temperatura			
UF	Cidade	Região	Temperatura
SC	Urubici	Sul	10º
SP	São Carlos	Sudeste	28º
RN	Natal	Nordeste	35º

Fonte: elaborado pela autora.



Pesquise mais

Você pode ler mais a respeito de dependência funcional, a partir da página 97 a 105 do livro *Sistema de banco de dados: uma abordagem introdutória e aplicada*:

CARDOSO, G.; CARDOSO, V. **Sistema de banco de dados: uma abordagem introdutória e aplicada**. São Paulo: Saraiva, 2012. Esse livro encontra-se disponível na biblioteca virtual <<https://bibliotecavirtual.uk>>.

A normalização de tabelas é um conjunto de procedimentos que permitem encontrar erros em um projeto de banco de dados, encontrando inconsistências que podem ser informações duplicadas e dependências funcionais mal resolvidas ou mal elaboradas. Normalizar é converter uma tabela em tabelas de graus e cardinalidades menores até que quase não haja redundâncias e nem dependências funcionais. Você poderá perceber que o objetivo principal da normalização não é eliminar totalmente as inconsistências, mas controlá-las. Identificar as dependências funcionais nas tabelas é o primeiro passo para saber que precisamos normalizar as tabelas em um banco de dados.

Sem medo de errar

Retomando os objetivos desta seção, você aprendeu que existe mais um processo de verificação de qualidade da modelagem de dados: a normalização das tabelas.

Como desafio, você, que possui uma empresa, atenderá o clube de futebol Unidos Venceremos Futebol Clube. Para realizar essa tarefa, você recebeu uma ficha de cadastro de jogos que, até então, é preenchida manualmente. O objetivo é informatizar essa ficha e armazenar os dados em um banco de dados.

Para resolver essa situação-problema primeiramente precisamos observar com cuidado as informações que a ficha fornece. Como encontrar os campos das tabelas? Faça uma listagem de todas as informações que podem ser cadastradas, por exemplo, podemos cadastrar as informações sobre: número do jogo, data do jogo, nome do adversário, estádio, etc.

E as tabelas? Como encontrar? A intuição nesse primeiro instante ajudará (visto que não vimos ainda as técnicas de normalização). Observe algumas tabelas: jogadores, cidade, estado, etc.

Crie um slide de apresentação com todos os possíveis campos da ficha de cadastro de jogos e a relação de todas as tabelas encontradas. Como desafio, procure relacionar as tabelas encontradas, criando um diagrama de entidades-relacionamento.

Avançando na prática

Modelagem a partir de um documento para uma loja de material de construção

Descrição da situação-problema

Após fazer compras em uma pequena loja de material de construção, você percebeu que o controle dos produtos levados pelos clientes era realizado em um caderno com fichas coladas para serem preenchidas de forma manual. Somente clientes antigos e de confiança possuem a regalia de pegar os produtos e pagar depois. Ao saberem que você é da área de informática, pediram que fizesse um banco de dados para agilizar o processo de controle. Para tanto, você recebeu uma cópia da ficha de controle, que permite a retirada de produtos da loja. Observe a Figura 4.6.

Figura 4.6 | Ficha de retirada de materiais

Fonte: elaborada pela autora.

Observando o documento da Figura 4.6, quais são os campos e tabelas que podem ser inseridos em um banco de dados?

Resolução da situação-problema

Primeiramente, devemos observar atentamente o documento. O que pode ser um campo? Tudo aquilo que o usuário pode informar e ser armazenado no banco de dados. A lista de campos será:

- Nº de controle da ficha
 - Data da nota
 - Nome cliente
 - RG cliente
 - CPF cliente
 - UF
 - Cidade
 - Código produto
 - Descrição produto
 - Quantidade
 - Preço unitário
 - Preço item
 - Valor total da nota

E como podemos encontrar as tabelas? Uma dica é observar a lista de campos, por exemplo, os campos: *Cod Produto* e *Descrição do Produto*. Esses dois campos já indicam que existe a tabela Produto. O próprio documento, no caso a ficha de controle, também é uma tabela (geralmente o documento modelado acaba virando uma tabela). Podemos listar as seguintes tabelas:

- Ficha de controle
- Cliente
- Produto
- Cidade
- Estado

Lembre-se de que após o relacionamento entre as tabelas poderão aparecer as tabelas associativas.

Faça valer a pena

1. Desenvolver um software de qualidade é o objetivo de todo analista de sistemas. Existem diversas técnicas que podem auxiliar a identificar problemas no software projetado. Modelar uma banco de dados não é uma exceção. É necessário realizar refinamentos nas tabelas buscando repetições e erros que passaram despercebidos.

Marque a alternativa correta que ajuda a melhorar a qualidade do projeto de um banco de dados.

- a) Modelagem de dados.
- b) Normalização de dados.
- c) Refinamento de dados.
- d) Reciclagem de dados.
- e) Estabilização dos dados.

2. Korth, Silberschatz e Sudarshan (2012) afirmam que as primeiras técnicas de normalização foram criadas em 1972 por Edgar Frank Codd. Após ter dado o primeiro passo, Codd propôs junto com Raymond Boyce, um novo significado, que ficou conhecido como Forma Normal Boyce-Codd (ou FNBC). Normalizar um banco de dados é aplicar regras para

todas as tabelas do banco de dados, com os objetivos, além reduzir a redundância, eliminar campos que não dizem respeito à tabela.

Marque a alternativa correta que demonstra uma vantagem de ter nenhuma ou pouca redundância em um banco de dados.

- a) Redução do número de tabelas no banco de dados.
- b) Diminuição de chaves estrangeiras nas tabelas de um banco de dados.
- c) Diminuição de dados repetidos deixando o banco de dados mais compacto.
- d) Redução de campos nas tabelas, sendo substituídos por chaves primárias concatenadas.
- e) Aumento da quantidade de chaves primárias e estrangeiras para conseguir diminuir a quantidade de relacionamentos entre as tabelas.

3. Identificar as dependências funcionais nas tabelas é o primeiro passo para saber que precisamos normalizar as tabelas em um banco de dados. A dependência funcional pode ser classificada em: transitiva ou indireta, total ou parcial.

Assinale a alternativa correta que demonstra o conceito para uma dependência funcional.

- a) Uma dependência funcional é uma nova tabela criada para estabelecer a normalização dos dados de uma tabela.
- b) Uma dependência funcional é a criação de chaves concatenadas para evitar as redundâncias entre as tabelas relacionadas.
- c) Uma dependência funcional é um relacionamento entre dois bancos de dados que podem interferir diretamente na performance do SGBD.
- d) Uma dependência funcional é um relacionamento entre dois ou mais atributos de forma que o valor de um atributo identifique o valor para cada um dos outros atributos, ou seja, um atributo está relacionado a outro.
- e) Uma dependência funcional é um novo campo que deve ser acrescentado na tabela para receber as informações duplicadas, dessa forma, todas as dependências são centralizadas nesse novo campo criado.

Seção 4.2

Transformação 1FN - 2FN

Diálogo aberto

Olá, seja bem-vindo!

Nesta seção começaremos a normalizar as tabelas de um banco de dados. Algumas das regras iniciais do processo de normalização já são aplicadas desde o começo deste livro. Você já foi orientado a criar as tabelas da forma correta. Agora, com as regras da normalização, caso haja alguma dúvida no processo de criação das tabelas e campos, você terá as regras para ajudá-lo.

Sua empresa tem um cliente especial, um clube de futebol. Os diretores do clube entregaram o documento que era preenchido à mão. O documento serve para cadastrar as informações de cada jogo realizado pelo time. Com a informatização do processo de cadastro dessas informações, o time de futebol espera melhorar suas estratégias nos jogos e verificar os jogadores que mais contribuem para o melhor desempenho da equipe.

Na seção anterior você recebeu um documento e a partir desse documento o banco de dados deverá ser criado. Os campos das tabelas já foram listados por você (como missão da última seção).

O desafio agora será responder: podemos aplicar a Primeira Forma Normal (1FN) e a Segunda Forma Normal (2FN) nas tabelas que foram encontradas a partir do documento? Como fazer?

Observe novamente o documento que o time de futebol deixou como exemplo:

Figura 4.7 | Ficha usada pelo Unidos Venceremos Futebol Clube

Unidos Venceremos Futebol Clube

JOGO Nº 114 DATA: 18/06/2018
ADVERSÁRIO: Cacoalense de RO
ESTÁDIO: ARENA DA FLORESTA CIDADE: Rio Branco
UF: AC
CAMPEONATO: Florestal Brasileiro
TÉCNICO:
PLACAR FINAL: UVFC _____ x _____ ADVERSÁRIO

JOGADORES:

Nº CAMISA	NOME	Nº de Gols na Partida

PRINCIPAIS EVENTOS DO JOGO:

TEMPO:	OCORRIDO:
2 min	Gol Nossa Time
3 min	Gol Adversário
15 min	Pênalti Nossa
16 min	Expulsão Time Adversário

HISTÓRICO DAS ÚLTIMAS GOLEADAS:

ADVERSÁRIOS	Qtde Gols Adv	Qtde Gols Casa
Pirambu - SE	1	2
Guarabira - PB	0	1
Luverdense - MT	0	2
Social - MG	2	3
Coroatá - MA	2	2

Fonte: elaborada pela autora.

Após responder essas perguntas, você deve fazer um slide para apresentar para seu cliente. Para essa apresentação, faça uma lista das tabelas e liste seus campos, identificando as chaves e os relacionamentos, encontrados a partir da tabela *Jogo* e dos campos listados. Utilize uma ferramenta CASE para criar o DER das tabelas encontradas. Pronto para começar?

Não pode faltar

Quando uma empresa investe na criação de um software, ela espera, além do funcionamento, rapidez nos resultados. As demandas por consultas ao banco de dados sempre são grandes e há constantemente necessidades de novas consultas. Um banco de dados mal projetado pode exigir muito tempo de espera por resultados e o pior poderá acontecer: erros duplicados e imprecisos. A normalização de tabelas, segundo Coronel e Rob (2011), é um método para avaliar e corrigir estruturas de tabelas com o propósito de reduzir as redundâncias de dados, reduzindo dessa forma a

possibilidades de erros e anomalias em uma tabela. Para atingir os objetivos da normalização, conforme Coronel e Rob (2011), as tabelas precisam ter as seguintes propriedades:

- Cada tabela deverá tratar de somente um único assunto, por exemplo: uma tabela com informações sobre remédio, não poderá ter informações de um médico.
- O mesmo campo não poderá ser armazenado, desnecessariamente, em mais de uma tabela. Essa é uma garantia de que não será necessária a atualização do mesmo campo, em mais de uma tabela.
- Os campos de uma tabela são dependentes da chave primária dessa tabela e de mais nenhum campo.
- A tabela deverá estar livre de anomalias de inserção, atualização e exclusão, garantindo a integridade e a consistência dos dados, por exemplo: na tabela *Cliente* será necessário informar a cidade de seu nascimento, não devemos deixar ele informar a cidade, mas escolher a cidade dentre as cidades previamente cadastradas ou por meio de uma busca do CEP (que trará o endereço completo).

Para aplicar as regras da normalização, um dos alvos a ser observado são os campos (ou atributos) que fazem parte das tabelas. Podemos classificar os atributos, conforme Korth, Silberschatz e Sudarshan (2012), de um Modelo Entidade-Relacionamento como:

- **Atributo simples ou atômico:** é o atributo que não é divisível, possui um sentido único, como o RG ou o CPF de uma pessoa, esses dois exemplos mostram que tanto o RG quanto o CPF não podem ser divididos em dois outros campos.
- **Atributo composto:** é um atributo que pode ser dividido em várias partes, um bom exemplo é o endereço. Podemos dividir esse atributo em: rua, número, complemento, bairro.
- **Atributo monovalorado:** é um atributo que possui apenas um valor para a tabela, como a matrícula de um aluno, esse número não poderá se repetir na tabela.

- **Atributo multivalorado:** é um atributo que pode receber mais de uma informação, o melhor exemplo é o telefone que pode receber mais de um valor.
- **Atributo derivado:** o valor desse tipo de atributo pode originar de outra tabela ou de outros campos. Digamos que para um cardiologista seja necessário saber a idade (em anos e dias). Podemos calculá-la a partir da data de nascimento e da data de atendimento no ato da consulta médica.
- **Atributo chave:** é o atributo escolhido ou criado para que possa indicar o registro (a linha) da tabela.

A Primeira Forma Normal, ou simplesmente 1FN, possui a seguinte regra: uma tabela estará na Primeira Forma Norma se, e somente se, todos os seus atributos forem atômicos, não possuindo grupos repetitivos ou colunas que possuam mais de um valor. Para estar na 1FN os seguintes passos devem ser realizados:

- Identificar a chave primária da tabela.
- Identificar a coluna que possua dados repetidos.
- Remover a coluna que tenha dados repetidos.
- Criar uma nova tabela para armazenar os dados repetidos.
- Criar um relacionamento entre a tabela que está sendo normalizada e a sua tabela secundária.

No Quadro 4.9, podemos observar a tabela *Funcionário* que não está normalizada e que possui diversos campos.

Quadro 4.9 | Tabela não normalizada de funcionário

Nome	Idade	Valor da Hora	Cidade	Departamento	Data de Admissão
Jonas Pedro	25	R\$ 18,54	Curitiba	Contabilidade	15/01/2018
Lívia Marta	19	R\$ 16,70	São Paulo	Produção	21/11/2017
Carlo Andes	22	R\$ 20,15	Santo André	RH	03/04/2018

Fonte: elaborado pela autora.



No Quadro 4.9 a tabela *Funcionário* não está normalizada. Analisando a tabela, quais os campos que apresentam problemas? Todos os campos realmente pertencem a essa tabela? Existem campos que podem ser chaves estrangeiras e que podem ser representados em outra tabela?

Para deixar a tabela *Funcionário* na Primeira Forma Normal (1FN) primeiramente devemos verificar: alguns dos campos podem ser a chave primária? Pelo que observamos, é melhor criar um campo novo e indicar como a chave primária. Uma boa sugestão é o campo: matrícula ou código do funcionário, observe como ficará a tabela, no modo textual:

Funcionário: (#matrículaFunc, nome, idade, data de admissão, valor da hora, cidade, departamento).

Podemos melhorar mais alguma coisa? Observe que há os campos: cidade e departamento. Para normalizar a tabela *Funcionário* para que fique na 1FN, devemos fazer o seguinte:

- Primeiro: criar uma tabela chamada *Cidade*.
- Segundo: inserir a chave estrangeira da tabela *Cidade* na tabela *Funcionário*.



Observe como ficaram as tabelas após a 1FN sendo aplicada:

Cidade (#idCidade, Cidade)

Funcionários (#matrículaFunc, nome, idade, valordahora, dtadmissão, Departamento, &idCidade)

No Quadro 4.10, observe a tabela *Funcionário* que agora está na Primeira Forma Normal (1FN).

Quadro 4.10 | Tabela Funcionário na 1FN

matrículaFunc	Nome	Idade	Valor da hora	idCidade	Departamento	dtAdmissão
123	Jonas Pedro	25	R\$ 18,54	12	Contabilidade	15/01/2018
456	Lívia Marta	19	R\$ 16,70	18	Produção	21/11/2017
789	Carlo Andes	22	R\$ 20,15	19	RH	03/04/2018

Fonte: elaborado pela autora.

Verificando mais uma vez os campos da tabela, observe o campo *idade*. Não está errado guardar, porém, a cada mudança de idade de um funcionário, uma atualização deverá ser realizada na tabela. Isso não é nem de longe prático e ideal para um banco de dados. Então, como resolver? Criar o campo *data de nascimento* em vez do campo *idade*. Agora a tabela ficará da seguinte forma: Funcionários (**#matrículaFunc**, nome, dtNascimento, valordahora, dtadmissão, Departamento, &idCidade).



Assimile

Para uma tabela estar na Primeira Forma Normal, todos os seus campos ou atributos deverão ser monovalorados e atômicos.

A Segunda Forma Normal, ou 2FN, deve obedecer a seguinte regra: uma tabela está na 2FN se, e somente se, estiver na 1FN e todas as suas colunas que não são chaves, dependam exclusivamente da chave primária (de toda a chave primária e não só de parte dela). Para estar na 2FN devemos aplicar as seguintes ações:

- Identificar as colunas que não são funcionalmente dependentes da chave primária da tabela.
- Remover o campo da tabela e criar uma nova tabela com esses dados.

Utilizando a tabela *Funcionário*, observe os seus campos: Funcionários (**#matrículaFunc**, nome, idade, valordahora,

`dtadmissão, Departamento, &idCidade)`. Para aplicar a 2FN, devemos criar uma tabela *Departamento* e inserir a chave estrangeira na tabela *Funcionário*. As tabelas irão ficar da seguinte forma:

Departamento (**#codDepart**, Departamento).

Funcionário (**#matrículaFunc**, nome, dtNascimento, valorHora, `dtadmissão, &codDepart, &idCidade`).

No Quadro 4.11, observe a tabela *Funcionário*:

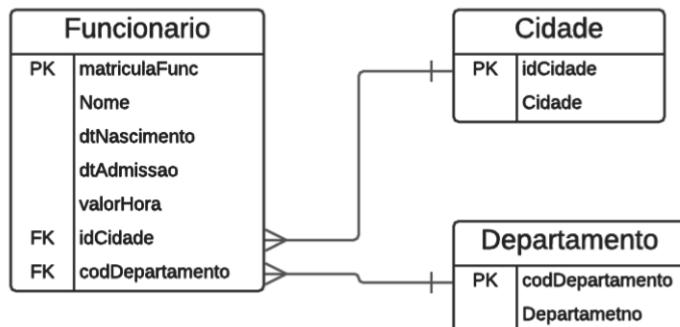
Quadro 4.11 | Tabela *Funcionário* na 2FN

matrículaFunc	Nome	dtNascimento	Valor Hora	idCidade	codDepart	dtAdmissão
123	Jonas Pedro	01/01/2000	R\$ 18,54	12	13	15/01/2018
456	Lívia Marta	13/10/1999	R\$ 16,70	18	21	21/11/2017
789	Carlo Andes	09/05/1998	R\$ 20,15	19	25	03/04/2018

Fonte: elaborado pela autora.

A Figura 4.8 mostra o Diagrama Entidade-Relacionamento resultante após aplicarmos a 1FN e a 2FN. Antes tínhamos somente uma única tabela, chamada *Funcionário*. Agora temos três tabelas.

Figura 4.8 | DER Funcionários na 1FN e na 2FN



Fonte: elaborada pela autora.



Pesquise mais

Baixe o artigo que apresenta uma ferramenta de apoio ao processo de normalização de tabelas, visando contribuir principalmente com projetos

bottom-up de bancos de dados relacionais. Por meio da análise de dados, a ferramenta descobre dependências funcionais existentes que devem ser removidas.

ÁVILA, M. L. de; MELLO, R. S. **Uma Ferramenta de Apoio à Normalização de Tabelas Relacionais Baseada na Análise de Dados.**

Disponível em: <<https://goo.gl/dyfJCQ>>. Acesso em: 2 jul. 2018.

Observe os campos apresentados sobre uma tabela de alunos: informações do aluno, endereço completo, telefones, notas. Como ficaria a tabela pelas informações passadas? Observe os campos na forma textual: aluno (informações, endereço, telefones, notas). Você como analista de sistemas ou programador não pode olhar esses campos e achar que está tudo certo. Vamos analisar os campos dessa tabela:

- Informações: que informações são essas? Nome, data de nascimento?
- Endereço: é o endereço completo com o bairro, cidade e estado?
- Telefones: quantos números armazenar?
- Notas: quantas notas? A quantidade pode variar? Há um limite de notas?

Observe a quantidade de perguntas realizadas somente sobre os campos da tabela aluno? E, é justamente essa a primeira tarefa de quem modela uma tabela: questionar o que puder sobre cada campo existente na tabela. A tabela *Aluno* não está normalizada e pode ser vista na Figura 4.9.

Figura 4.9 | Tabela Aluno não normalizada

Aluno	
	Informações
	Endereços
	Telefones
	Notas

Fonte: elaborada pela autora.

Vamos aplicar a Primeira Forma Normal na tabela *Aluno*. O primeiro passo é descobrir o que o campo *Informações* irá armazenar e nesse caso será: o CPF, RG, o nome, e-mail, foto e a data de nascimento. Aproveitando, podemos também criar uma chave primária para a tabela, como existe o CPF, iremos utilizar esse campo como chave primária. Agora a tabela ficará com novos campos, observe a Figura 4.10.

Figura 4.10 | Tabela Aluno na 1FN – 1^a parte

Aluno	
PK	CPF
	nome
	email
	dtNascimento
	foto
	RG
	Telefones
	Notas
	Endereço

Fonte: elaborada pela autora.

Resolvemos dois problemas na tabela: criamos a chave primária e “arrumamos” o campo *informações* e que agora são os campos: CPF, RG, nome, e-mail, data de nascimento e a foto do aluno. Um campo acabou virando outros seis campos. Terminou? Não! Será necessário verificar o campo *endereço* e saber: o que ele representa? O endereço deverá ter: rua, número, complemento, CEP, bairro, cidade e estado. Na Figura 4.11, verifique como ficará a tabela *Aluno*.

Figura 4.11 | Tabela Aluno na 1FN – 2^a parte

Aluno	
PK	CPF
	nome
	email
	dtNascimento
	foto
	RG
	Telefones
	Notas
	Rua
	Numero
	Complemento
	CEP
	Bairro
	Cidade
	Estado

Fonte: elaborada pela autora.

O problema do endereço na tabela *Aluno* foi resolvido, porém, sobraram os outros dois campos: notas e telefones. Quantas notas? Quantos telefones? Para a tabela *Aluno* vamos determinar 4 notas para serem cadastradas (para esses campos, sobre as notas, futuramente teremos que aplicar outras Formas Normais para resolver algumas dependências funcionais). E, para os telefones vamos classificar e guardar como: residencial, celular e do trabalho. A tabela *Aluno* ficará com os seguintes campos: *Aluno* (#CPF, Nome, E-mail, RG, Foto, dtNascimento, Rua, Número, Complemento, CEP, Bairro, Cidade, Estado, Nota1, Nota2, Nota3, Nota4, Tefefone_Residencial, Telefone_Celular, Telefone_Trabalho).



Assimile

Uma forma de deixar a sua tabela sempre na 1FN é sempre criar a chave primária e analisar o conteúdo que cada campo irá armazenar. Outra dica importante é sobre os campos: *Cidade* e *Estado*. Tanto *Cidade* quanto *Estado* sempre serão tabelas. Você nunca deverá deixar como um simples campo em uma tabela, caso faça isso o usuário poderá armazenar qualquer coisa como nome de cidade e de estado, gerando inconsistência nos dados armazenados.

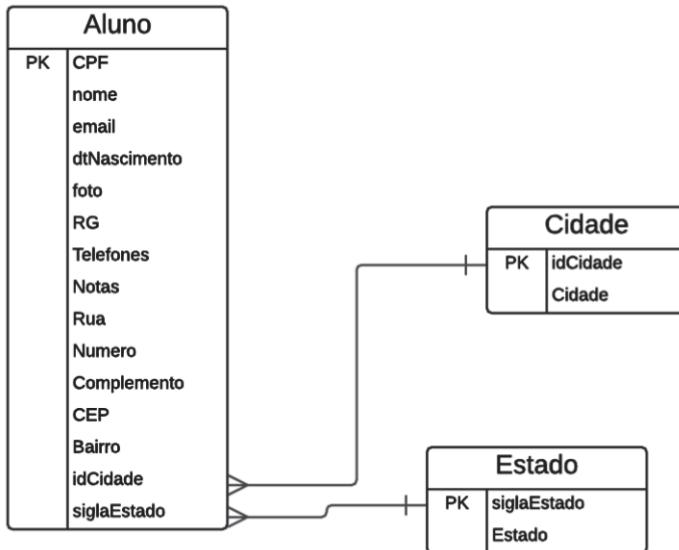
É possível aplicar a 2FN na tabela *Aluno*? Sim, porque existem campos que dependem da chave primária e que poderiam estar em outra tabela, são os seguintes campos: *Cidade* e *Estado*. Devemos tirar esses dois campos da tabela *Aluno* e criar duas novas tabelas: *Cidade* e *Estado*.



Exemplificando

Na Figura 4.12, você pode verificar o resultado final do exemplo de aplicação da 1FN e 2FN na tabela *Aluno*. Veja que em vez de termos uma tabela, já são três.

Figura 4.12 | DER *Aluno* na 2FN



Fonte: elaborada pela autora.

As duas primeiras formas normais que foram apresentadas, 1FN e a 2FN, já foram implicitamente implementadas no decorrer deste livro. Sempre que aplicamos o processo de modelagem de dados precisamos ficar atentos para inserir campos nas tabelas que realmente são dessa tabela. Não podemos “misturar” os campos das tabelas e essa é uma regra básica e essencial para a modelagem de dados.

Sem medo de errar

Você recebeu um documento como base para criar um banco de dados. Isso é comum? Sim e é muito comum! Os documentos são ótimos, pois a partir deles podemos realmente saber o que é utilizado pelos clientes. E, é justamente na hora de modelar um banco de dados a partir de um documento que a normalização deve ser empregada.

Para relembrar, o clube de futebol Unidos Venceremos Futebol Clube, precisa agilizar as estatísticas sobre a atuação do time nos campeonatos que participa. Precisa também verificar o desempenho de cada jogador. Como aplicar a Primeira Forma Normal e a Segunda Forma Normal nas tabelas que foram encontradas a partir do documento?

Para ajudar na solução, os campos encontrados sobre cada jogo são: Número do Jogo, Data do Jogo, Adversário, Estádio, Cidade, UF, Campeonato, Técnico, Placar Final, Jogadores (Número da Camisa, Nome, Número de Gols na Partida), Eventos do Jogo (Tempo, Evento Ocorrido).

Para aplicar a Primeira Forma Normal devemos primeiro inserir uma chave primária na tabela Jogo, nesse caso o melhor campo para ser chave primária seria o **Número do Jogo**. A tabela Jogo ficará assim: Jogo (**#Número do Jogo**, Data do Jogo, Adversário, Estádio, Cidade, UF, Campeonato, Técnico, Placar Final, Jogadores (Número da Camisa, Nome, Número de Gols na Partida), Eventos do Jogo (Tempo, Evento Ocorrido).

Você consegue visualizar novas tabelas “escondidas como campos” na tabela Jogo?

Utilize uma ferramenta gráfica e comece a elaborar o diagrama.

Avançando na prática

Modelagem de dados a partir de um documento (continuação)

Descrição da situação-problema

Vamos continuar a modelagem do documento da loja de material de construção. Para relembrar: após fazer compras em

uma pequena loja de material de construção, você percebeu que o controle dos produtos levados pelos clientes era cadastrado em um caderno com fichas coladas para serem preenchidas de forma manual. Somente clientes antigos e de confiança possuem a regalia de pegar os produtos e pagar depois. Ao saberem que você é da área de informática, pediram que você fizesse um banco de dados para agilizar o processo de controle. Para tanto, você recebeu uma cópia da ficha de controle, que permite a retirada de produtos da loja. Observe o Quadro 4.12.

Quadro 4.12 | Ficha de controle

FICHA DE CONTROLE DE RETIRADA DE MATERIAIS

Fonte: elaborado pela autora.

Os campos encontrados na última seção foram: Nº de controle da ficha, Data da nota, Nome Cliente, RG Cliente, CPF Cliente, UF, Cidade, Código Produto, Descrição produto, Quantidade, Preço unitário, Preço item, Valor total da nota.

As tabelas encontradas foram: *Ficha de Controle*, *Cliente*, *Produto*, *Cidade*, *Estado*.

Como aplicar a 1FN e a 2FN nas tabelas encontradas?

Resolução da situação-problema

Para aplicar a 1FN primeiro devemos inserir a chave primária nas tabelas:

Ficha (#numControle, DtNota, valorTotal, Cidade, Estado)

Cliente (#CPF, Nome, RG, Endereço)

Produto (#codProduto, Descricao, precoUnitario, quantidade, precoTotal)

Para aplicar a 2FN primeiro devemos criar as tabelas: *Cidade* e *Estado* e inserir as chaves estrangeiras na tabela *Ficha*.

Ficha (#numControle, DtNota, valorTotal, &idCidade, &siglaEstado)

Cidade (#idCidade, Cidade)

Estado (#siglaEstado, Estado)

Faça valer a pena

1. Para realizar a normalização das tabelas, o primeiro passo é analisar os atributos ou campos como também são conhecidos. Muitas vezes um atributo pode esconder diversas informações. Um exemplo é o atributo endereço. O endereço é composto das seguintes informações: nome da rua, número da casa, complemento, bairro. É necessário verificar se essas informações serão mantidas como campos ou se podem até virar uma tabela chamada *Endereço*.

O atributo que não é divisível possui um sentido único, e não pode ser dividido em dois outros campos e um conceito de:

- a) Atributo monovalorado.
- b) Atributo simples ou atômico.
- c) Atributo multivalorado.
- d) Atributo regular.
- e) Atributo normal.

2. A Primeira Forma Normal ou simplesmente 1FN possui a seguinte regra: uma tabela estará na Primeira Forma Norma se, e somente se, todos seus atributos forem atômicos, não possuindo grupos repetitivos ou colunas que possuam mais de um valor.

Marque a alternativa correta que indica um dos primeiros passos para deixar uma tabela na 1FN.

- a) Estabelecer os relacionamentos entre as tabelas.
- b) Criar no mínimo quinze campos a mais na tabela.

- c) Identificar ou criar uma chave primária na tabela.
- d) Reduzir a quantidade de campos para que fiquem abaixo de vinte campos.
- e) Dividir a tabela em no mínimo três tabelas novas.

3. A normalização é o processo de organizar os dados em um banco de dados. Isso inclui criar tabelas e estabelecer relacionamentos entre essas tabelas de acordo com as regras criadas para proteger os dados e para tornar o banco de dados mais flexíveis, eliminando a redundância e dependência inconsistente. Existem várias formas normais, entre as principais estão: 1FN, 2FN, 3FN e a 4FN.

Marque a alternativa correta que mostra a regra para uma tabela estar na 2FN.

- a) Uma tabela está na 2FN se, e somente se, não estiver na 1FN e todas as suas colunas, que não são chaves, não dependam exclusivamente da chave primária.
- b) Uma tabela está na 2FN somente se não estiver na 1FN e todas as suas chaves estrangeiras dependam exclusivamente da chave primária.
- c) Uma tabela está na 2FN, automaticamente, se estiver na 1FN e não precise de nenhuma chave primária, somente das chaves estrangeiras.
- d) Uma tabela está na 2FN, automaticamente, se não estiver na 1FN e não precise de nenhuma chave primária, somente de uma chave estrangeira.
- e) Uma tabela está na 2FN se, e somente se, estiver na 1FN e todas as suas colunas, que não são chaves, dependam exclusivamente da chave primária.

Seção 4.3

Transformação 3FN - 4FN

Diálogo aberto

Olá, seja bem-vindo a nossa última seção.

Nesta seção finalizaremos as regras de normalização, que servem como apoio para verificar se há inconsistências na modelagem do banco de dados. Com certeza após aplicar algumas vezes essas regras, você fará a modelagem de forma automática.

Retomaremos o contexto profissional em que você se encontra, concluindo seu trabalho nesta seção. Para relembrar, você foi contratado para criar um banco de dados para um time de futebol. O documento selecionado para servir como apoio para a criação do banco de dados é uma ficha de controle dos jogos realizados. Nas seções anteriores já realizamos, a partir do documento apresentado, o levantamento dos campos e aplicamos a Primeira Forma Normal e a Segunda Forma Normal nas tabelas.

O banco de dados a ser elaborado para o time de futebol precisa estar muito bem modelado, e você não tem certeza se as tabeladas criadas estão realmente corretas. Como você pode garantir que seu trabalho foi realizado de maneira correta? Poderemos aplicar a Terceira Forma Normal e a Quarta Forma Normal nas tabelas encontradas? Será possível criar o Diagrama Entidade-Relacionamento a partir das tabelas normalizadas?

O primeiro passo é novamente observar o documento fornecido pelo time de futebol, na Figura 4.13.

Figura 4.13 | Documento time de futebol

Unidos Venceremos Futebol Clube

**JOGO N° 114
de RO**

DATA: 18/06/2018 ADVERSÁRIO: Cacoalense

ESTÁDIO: ARENA DA FLORESTA CIDADE: Rio Branco UF: AC

CAMPEONATO: Florestal Brasileiro TÉCNICO:

PLACAR FINAL: UVFC _____ x _____ ADVERSÁRIO

JOGADORES:

Nº CAMISA	NOME	Nº de Gols na Partida

PRINCIPAIS EVENTOS DO JOGO:

TEMPO:	OCORRIDO:
2 min	Gol Nossa Time
3 min	Gol Adversário
15 min	Pênalti Nossa
16 min	Expulsão Time Adversário

Fonte: elaborada pela autora.

Para realizar a modelagem do banco de dados para o cliente, você precisará aplicar todas as Quatro Formas Normais para conseguir garantir que as tabelas criadas a partir do documento estejam corretas. A partir das etapas que você já realizou nas seções anteriores, liste todas as tabelas na forma textual, separe cada Forma Normal e liste as tabelas criadas. Após, crie o Diagrama de Entidade-Relacionamento das tabelas e campos, utilizando uma ferramenta CASE de sua preferência, crie um dicionário de dados dos campos e finalize com um slide para apresentar ao seu cliente.

Não pode faltar

A normalização é um processo que visa diminuir a redundância no banco de dados. A ideia central é identificar e reduzir de forma gradual as anomalias que podem aparecer em tabelas ou nos relacionamentos. De forma geral, precisamos retirar um ou mais campos de uma tabela e criar novas tabelas para receber esses campos retirados. Conforme afirmam Navathe e Ramez (2005), o procedimento de normalização proporciona a quem for modelar um banco de dados, as seguintes ações:

- Uma estrutura formal para a análise dos relacionamentos entre as tabelas, com base em suas chaves (primárias e estrangeiras) e das dependências funcionais entre os campos da tabela.
- Um conjunto de testes de Formas Normais para ser realizado em cada esquema de relação, de forma que o modelo de banco de dados seja normalizado no grau desejado, aplicando as Formas Normais até o limite que for mais conveniente para a modelagem do banco de dados.



O método de normalização envolve a aplicação de um conjunto de regras, chamadas de Formas Normais. Ao avaliarmos um banco de dados e examinarmos que ele respeita as regras da Primeira Forma Normal, podemos, dessa forma, afirmar que o banco está na 1FN (Primeira Forma Normal).

Na medida que o analista de sistemas, programador ou o projetista de banco de dados ganha experiência na modelagem de dados, erros comuns que são encontrados na Primeira Forma Normal e na Segunda Forma Normal são evitados antes mesmos de serem criados. Um exemplo que já foi citado na seção anterior é a tabela *Cidade*. Quem faz a modelagem de dados de tabelas já sabe de antemão que jamais deverá inserir um campo chamado *Cidade* nas tabelas. A regra é simples: se em uma tabela aparecer o campo *Cidade*, devemos criar uma tabela *Cidade*.

Mas qual a vantagem de aplicar a normalização? A primeira é a redução do trabalho na manutenção do banco de dados. Usando o exemplo do campo *Cidade*, imagine se uma cidade trocar de nome, para atualizar o banco de dados e se ele não for normalizado, será necessário:

- Localizar o campo *Cidade* em todas as tabelas.
- Realizar um procedimento de manutenção para a troca do nome.

Se o banco de dados for normalizado, a cidade estará em uma tabela. Basta alterar o nome nessa tabela e todas as outras que

estiverem relacionadas com a tabela *Cidade* automaticamente ficarão atualizadas. Um banco de dados dentro das regras de normalização diminui o trabalho de manutenção e ajuda a evitar o desperdício do espaço de armazenamento.



Exemplificando

Uma tabela está na 1FN quando todos os campos contêm apenas um valor correspondente, singular e não existem grupos de atributos repetidos, isso quer dizer que não haverá repetições ou campos que tenham mais que um valor. O primeiro passo é identificar a chave primária da tabela. Após, é necessário reconhecer os campos repetitivos e removê-los da tabela. Em seguida, criamos uma nova tabela para receber os campos que estão repetindo na tabela que está sendo modelada.

Uma tabela está na Segunda Forma Normal se ela atender os requisitos da Primeira Forma Normal e se os campos que estão na tabela, que não sejam chaves, dependam da chave primária em sua totalidade e não em parte dela, causando irregularidades e prevenindo a redundância. Para solucionar, devemos identificar esses campos que dependem parcialmente da chave primária e criar uma nova tabela para eles, criando um relacionamento entre as duas tabelas. No Quadro 4.13 é demonstrado a tabela *Cliente*, a tabela está na 1FN e precisa que a 2 FN seja aplicada.

Quadro 4.13 | Tabela *Cliente* não normalizada na 2FN

#idCliente	Cliente	Endereço	Nr Nota Fiscal	Valor da Nota
5412	Jonas Pedro	R. das Pedras, 15.	1456	R\$ 178,35
8532	Lena Luz	R. das Flores, 558.	1488	R\$ 587,30
4588	Caio Luiz	R. Beira Rio, 47.	1502	R\$ 358,00

Fonte: elaborado pela autora.



Exemplificando

Quando a tabela *Cliente* foi normalizada para a 2FN constatou-se que a *Nota Fiscal* pode ser uma nova tabela, observe o Quadro 4.14.

Quadro 4.14 | Tabela Nota Fiscal

Nr Nota Fiscal	Valor da Nota
1456	R\$ 178,35
1488	R\$ 587,30
1502	R\$ 358,00

Fonte: elaborado pela autora.

Na tabela *Cliente* é necessário criar uma chave estrangeira, fazendo a referência à tabela *Nota Fiscal*, observe o Quadro 4.15. Lembre-se de que a nota fiscal só pode pertencer a um único cliente e um cliente poderá possuir várias notas fiscais, observe a tabela *Cliente* com a chave estrangeira da tabela *Nota Fiscal*.

Quadro 4.15 | Tabela *Cliente* normalizada

#idCliente	Cliente	Endereço	&Nr Nota Fiscal
5412	Jonas Pedro	R. das Pedras, 15.	1456
8532	Lena Luz	R. das Flores, 558.	1488
4588	Caio Luiz	R. Beira Rio, 47.	1502

Fonte: elaborado pela autora.

Uma tabela estará na Terceira Forma Normal somente se estiver na Segunda Forma Normal e todos os campos forem independentes, isso quer dizer que não poderá haver dependências funcionais entre os campos e todos os campos dependem da chave primária da tabela. Os campos da tabela precisam depender unicamente da chave primária da tabela. Para aplicar a Terceira Forma Normal é necessário:

- Reconhecer os campos que são funcionalmente dependentes das outras colunas não chaves.
- Eliminar as colunas dependentes.

Conforme Coronel e Rob (2011) para deixar uma tabela na Terceira Forma Normal é necessário eliminar todas as dependências transitivas, ou seja, eliminar todos os campos dependentes de outras tabelas. O Quadro 4.16 demonstra a tabela *Funcionário*, observe que um dos campos é a *Descrição*, mas é descrição do que? É a descrição do cargo que o funcionário ocupa e, nesse caso, será necessário aplicar a 3FN na tabela.

Quadro 4.16 | Tabela Funcionário

#cdFuncionário	Nome	idCargo	Descrição
148-9	Jane Anne	15	Professor I
721-4	Klaus Lins	16	Diretor
673-2	Sandra Costa	17	Professor II

Fonte: elaborado pela autora.



Assimile

Uma dependência funcional transitiva ocorre quando o valor de uma coluna é dependente de outra que não compõe a chave primária, dependendo indiretamente de outra chave primária. No Quadro 4.16 o campo *Descrição* depende do *Cargo* (que é outra tabela) e nesse caso há uma dependência funcional transitiva.

Ao aplicar a 3FN na tabela *Funcionário*, retiramos o campo *Descrição*, afinal esse é um campo que detalha o cargo e deve estar em uma tabela apropriada e que no caso será a tabela *Cargo*. O Quadro 4.17 demonstra a tabela *Funcionário* após aplicarmos a 3FN.

Quadro 4.17 | Tabela Funcionário normalizada

#cdFuncionário	Nome	&idCargo
148-9	Jane Anne	15
721-4	Klaus Lins	16
673-2	Sandra Costa	17

Fonte: elaborado pela autora.

A tabela *Cargo*, Quadro 4.18, agora possui a descrição do cargo do funcionário. O relacionamento entre as duas tabelas (*Funcionário* e *Cargo*) é 1 para N. Observe que na tabela *Funcionário* há um campo com o sinal &, isso indica que o campo é uma chave estrangeira e a tabela está relacionada com a tabela *Cargo*.

Quadro 4.18 | Tabela Cargo

#idCargo	Descrição
15	Professor I
16	Diretor
17	Professor II

Fonte: elaborado pela autora.



Observe a tabela **Ordem_de_Compra** na forma textual:

Ordem_de_Compra(**#idOrdem**, dtOrdem, &codProduto, Quantidade, Valor_Unitário, ValorTotal). Para aplicar a 3FN, qual campo não deveria estar na tabela? Um dos campos é resultado de um cálculo, consegue identificar qual é o campo?

Para realizar a normalização de uma tabela na Quarta Forma Normal (ou 4FN) é necessário que a tabela esteja na Terceira Forma Normal. A tabela somente estará na 4FN se não existir dependência multivalorada. Mas o que isso quer dizer? Dependência multivalorada é quando as informações inseridas nas tabelas podem ficar se repetindo e, claro, produzir redundâncias na tabela. Para evitar esse tipo de problema é melhor dividir a tabela evitando assim esse tipo de dependência. De acordo com Navathe e Ramez (2005), em uma tabela na 4FN, além de estar na 3FN, todo campo precisa ser atômico (não pode ser dividido em vários campos).

Os procedimentos para deixar uma tabela na 4FN são:

- Primeiro identificar os campos multivalorados (que causam repetições).
- Criar uma tabela para cada grupo multivalorado.
- Criar uma chave primária para a nova tabela.
- Inserir a chave estrangeira na tabela que está sendo normalizada (na 4FN) para criar o relacionamento entre as tabelas.

Para uma empresa é necessário armazenar as informações sobre os dependentes de todos os seus funcionários. Esse armazenamento se faz necessário por causa do imposto de renda, plano de saúde, etc. No Quadro 4.19 a tabela *Funcionário* possui dois campos: *Dependente* e *Parentesco*.

Quadro 4.19 | Tabela *Funcionário* com campo multivalorado

#cdFuncionário	Nome	&idCargo	Dependente	Parentesco
148-9	Jane Anne	15	Lucy Anne	Filho
721-4	Klaus Lins	16	Ana Lins	Espousa
673-2	Sandra Costa	17	Jonny Costa	Filho

Fonte: elaborado pela autora.

Algumas perguntas sobre a tabela *Funcionário*, mostrada no Quadro 4.19 precisam de respostas:

- Caso um funcionário tenha mais de um filho, como será o cadastro?
- Vamos cadastrar o funcionário duas vezes?
- E se o funcionário tiver cinco filhos?

A redundância na tabela *Funcionário* aparecerá com certeza. Para resolver esse problema, será necessário dividir a tabela *Funcionário* em duas tabelas: *Funcionário* e *Dependentes*. No Quadro 4.20 foi retirado totalmente os campos *Dependente* e *Parentesco*.

Quadro 4.20 | Tabela *Funcionário* normalizada

#cdFuncionário	Nome	&idCargo
148-9	Jane Anne	15
721-4	Klaus Lins	16
673-2	Sandra Costa	17

Fonte: elaborado pela autora.

O Quadro 4.21 apresenta a tabela *Dependente* que foi criada para evitar os campos multivalorados na tabela *Funcionário*.

Quadro 4.21 | Tabela *Dependente*

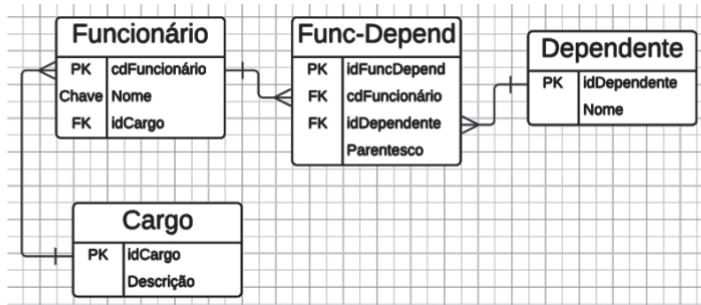
#idDependente	Nome	Parentesco
148-9	Lucy Anne	Filho
721-4	Ana Lins	Esposa
673-2	Jonny Costa	Filho

Fonte: elaborado pela autora.

Refletindo um pouco sobre as tabelas *Funcionário* e *Dependente*: seria possível que, em uma empresa, o mesmo dependente pudesse ser dependente de mais de um funcionário? E um funcionário poderia ter mais de um dependente? As respostas são todas afirmativas. Uma sugestão de modelagem dessas tabelas aparece na Figura 4.14. Foi criado mais uma

tabela para relacionar os funcionários que possuem mais de um dependente e os dependentes que estão relacionados com mais de um funcionário (gerando assim um relacionamento N para N).

Figura 4.14 | DER das tabelas normalizadas



Fonte: elaborada pela autora.

Korth, Silberschatz e Sudarshan (2012) afirmam que a Quarta Forma Normal não é a última instância da modelagem de um esquema de banco de dados. Os refinamentos podem continuar evoluindo, juntamente com o banco de dados. Alguns autores afirmam que um banco de dados normalizado até a Terceira Forma Normal já está num nível aceitável. Com a grande demanda de informações da nossa era, é importante pensar nos perigos que o excesso de redundância pode ocasionar ao banco de dados, afinal, guardar essas informações requer espaço em equipamentos e isso gera custos.



Pesquise mais

Você pode ver dois exemplos completos de modelagem aplicando a normalização de dados nas páginas 110 a 119 do livro:

CARDOSO, G.; CARDOSO, V. **Sistema de banco de dados**: uma abordagem introdutória e aplicada. São Paulo: Saraiva, 2012.

Esse livro encontra-se disponível na biblioteca virtual <<https://bibliotecavirtual/uk>>.

Chegamos ao final do nosso livro! Você aprendeu sobre a modelagem de dados de uma forma mais prática, aplicada para projetos de médio porte. A experiência fará com que os conceitos desse livro sejam diversas vezes aplicados no dia a dia do desenvolvimento de softwares. O nível de refinamento de suas modelagens irá refletir com a prática que você irá adquirir. Por isso, mãos à obra. Coloque em prática seus projetos, comece pelo banco de dados e realize a modelagem dos dados.

Sem medo de errar

O processo de normalização é importante para ajudar a encontrar problemas na modelagem do banco de dados. Muitas vezes nem percebemos que podemos transformar um campo em uma tabela. E pode ter certeza que o impacto de uma modelagem realizada erroneamente só é percebido após o banco de dados apresentar uma série de problemas. Dessa forma precisamos aplicar as Quatro Formas Normais nas tabelas.

Para criar um banco de dados para o time Unidos Venceremos Futebol Clube, você recebeu um documento para servir como base, como mostrado na Figura 4.13. Esse documento servirá de apoio inicial no processo de modelagem do banco de dados.

Nas seções anteriores, a partir do documento foram encontrados os seguintes campos: Número do Jogo, Data do Jogo, Adversário, Estádio, Cidade, UF, Campeonato, Técnico, Placar Final, Jogadores (Número da Camisa, Nome, Número de Gols na Partida), Eventos do Jogo (Tempo, Evento Ocorrido).

Para aplicar a 1FN devemos inserir a chave primária nas tabelas Jogo, Adversário, Estádio.

Para aplicar a 2FN devemos criar as tabelas Cidade e Estado e inserir as chaves estrangeiras na tabela Ficha.

Para aplicar a 3FN devemos eliminar todos os campos dependentes

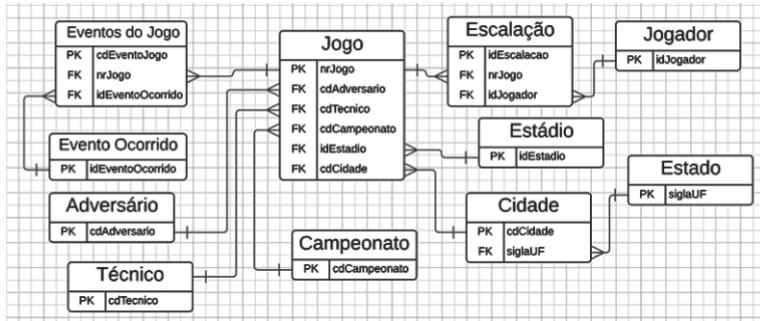
Surgem aqui as tabelas: Jogadores, Campeonato, Técnico e Eventos do Jogo.

Para aplicar a 4FN devemos verificar se não há campos multivvalorados e não atômicos

A tabela *Evento Ocorrido* aparece nessa etapa, pois os eventos de um jogo se repetem muitas vezes em um jogo e em outros jogos.

O DER desse documento deverá ficar conforme a Figura 4.15.

Figura 4.15 | DER – Ficha de controle



Fonte: elaborada pela autora.

O DER da Figura 4.15 precisa de alguns ajustes. Para otimização da imagem os campos foram suprimidos. Agora é hora de finalizar seu trabalho e é com você: utilize uma ferramenta CASE para criar o DER completo das tabelas encontradas (com todos os campos), crie o dicionário de dados das tabelas e crie um slide para apresentar ao seu cliente. Sucesso!

Avançando na prática

Modelagem de dados a partir de um documento (finalizando)

Descrição da situação-problema

Vamos finalizar a modelagem do documento da loja de material de construção. Para relembrar: após fazer compras em uma pequena loja de material de construção, você percebeu que o controle dos produtos levados pelos clientes eram cadastrados em um caderno com fichas coladas para serem preenchidas de forma manual. Somente clientes抗igos e, de confiança, possuem a regalia de pegar os produtos e pagar depois. Ao saberem que você

é da área de informática, pediram que fizesse um banco de dados para agilizar o processo de controle. Para tanto, você recebeu uma cópia da ficha de controle, que permite a retirada de produtos da loja, observe o Quadro 4.22.

Quadro 4.22 | Ficha de controle

FICHA DE CONTROLE DE RETIRADA DE MATERIAIS

Controle ficha nº:		Data:		
Cliente:	RG:	CPF:		
Endereço:	Cidade:	UF:		
Produtos				
Código	Descrição	Quantidade	Preço unitário	Preço total
			Valor total a pagar:	

Fonte: elaborado pela autora.

Como ficará o processo de normalização das tabelas? Qual o DER resultante da modelagem do documento?

Resolução da situação-problema

Nas seções anteriores foi realizada, primeiramente, a busca dos campos no documento e foram encontrados os seguintes: Nº de controle da ficha, Data da nota, Nome Cliente, RG Cliente, CPF Cliente, UF, Cidade, Código Produto, Descrição produto, Quantidade, Preço unitário, Preço item, Valor total da nota.

Após, foram encontradas as tabelas: *Ficha de Controle*, *Cliente*, *Produto*, *Cidade*, *Estado*.

Aplicamos a 1FN e a 2 FN nas tabelas encontradas.

Para aplicar a 1FN devemos inserir a chave primária nas tabelas:

Ficha (#numControle, DtNota, valorTotal, Cidade, Estado)

Cliente (#CPF, Nome, RG, Endereco)

Produto (#codProduto, Descricao, precoUnitario, quantidade, precoTotal)

Para aplicar a 2FN devemos criar as tabelas *Cidade* e *Estado* e inserir as chaves estrangeiras na tabela *Ficha*.

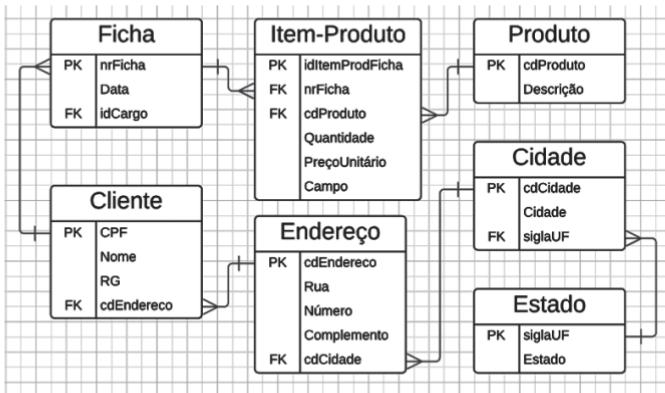
Ficha (#numControle, DtNota, valorTotal, &idCidade, &siglaEstado)
 Cidade (#idCidade, Cidade)
 Estado (#siglaEstado, Estado)

Para aplicar a 3FN devemos eliminar todos os campos dependentes, observando a tabela *Produto*, podemos retirar o campo *precoTotal* pois ele pode ser calculado a partir da quantidade do produto com o preço unitário do produto.

Para aplicar a 4FN devemos verificar se não há campos multivalorados e não atômicos. Nesse exemplo, temos o campo *endereço* que poderia ser inserido numa tabela, detalhando os campos que serão armazenados.

O DER resultante da modelagem do documento da Figura 4.13 pode ser analisado na Figura 4.16.

Figura 4.16 | DER – Ficha de controle



Fonte: elaborada pela autora.

Faça valer a pena

- Um dos primeiros passos no processo de normalização é a identificação das dependências funcionais. Uma dependência funcional é um relacionamento entre dois ou mais atributos de forma que o valor de um atributo identifique o valor para cada um dos outros atributos, ou seja, um atributo está relacionado a outro. A dependência funcional consiste em

uma restrição entre dois ou mais conjuntos de atributos de uma mesma tabela ou relacionamento.

Assinale a alternativa correta que determina quando uma tabela está na 3FN.

- a) Uma tabela estará na Terceira Forma Normal somente se estiver na Segunda Forma Normal e todos os campos forem dependentes funcionais entre as chaves primárias e estrangeiras.
- b) Uma tabela estará na Terceira Forma Normal somente se estiver na Quarta Forma Normal e todos os campos forem independentes da chave primária.
- c) Uma tabela estará na Terceira Forma Normal somente se estiver na Segunda Forma Normal e todos os campos forem independentes e não poderá haver dependências funcionais entre os campos.
- d) Uma tabela estará na Terceira Forma Normal somente se estiver na Primeira Forma Normal e todos os campos forem dependentes da chave primária e da chave estrangeira.
- e) Uma tabela estará na Terceira Forma Normal se os campos forem dependentes das chaves estrangeiras da tabela.

2. Uma tabela somente estará na 4FN se não existir dependência multivalorada. Quando as dependências multivaloradas ocorrem, as informações inseridas nas tabelas podem ficar se repetindo e, dessa forma, produzem redundâncias na tabela.

Assinale a alternativa correta que indica como resolver o problema da dependência multivalorada em uma tabela.

- a) Para evitar esse tipo de problema é melhor juntar todos os campos, concatenando os dois campos em um único campo, isso agiliza o processo de cadastro das informações.
- b) Para evitar esse tipo de problema é melhor dividir a chave primária em duas, tornando a chave numa chave composta, agilizando o armazenamento dos dados.
- c) Para evitar esse tipo de problema é melhor criar um campo a mais para cada campo da tabela. Ele servirá de backup para a tabela, acelerando a sua manutenção.
- d) Para evitar esse tipo de problema é melhor transformar em chave estrangeira cada campo duplicado, dessa forma, nunca mais haverá repetições na tabela.
- e) Para evitar esse tipo de problema é melhor dividir a tabela, quantas vezes for necessário, a fim de evitar esse tipo de dependência.

3. De acordo com Navathe e Ramez (2005), para uma tabela estar na Quarta Forma Normal, além de estar na Terceira Forma Normal, precisa que todo campo da tabela seja um campo atômico. Isso permite que repetições desnecessárias sejam evitadas, diminuindo transtornos de manutenção e problemas de redundâncias nos dados inseridos no banco de dados.

Marque a alternativa que demonstra o conceito correto de um atributo atômico.

- a) É um atributo que pode ser dividido em mais campos, facilitando a inserção de dados na tabela.
- b) É um atributo que não é divisível, possui um sentido único e não pode ser dividido em dois outros campos.
- c) É um atributo que “turbina” a tabela, aceitando quaisquer tipos de dados. É conhecido como um campo coringa para as tabelas.
- d) É o atributo que automaticamente vira a chave primária da tabela, caso tenha dois campos atômicos, teremos duas chaves primárias na tabela.
- e) É o atributo que é responsável por relacionar uma tabela com a outra tabela.

Referências

- ABREU, M. P.; MACHADO, F. N. R. **Projeto de banco de dados:** uma visão prática. 16. ed. rev. e atual. São Paulo: Érica, 2009.
- ALVES, W. P. **Banco de Dados.** São Paulo: Saraiva, 2014.
- COUGO, P. **Modelagem conceitual e projeto de bancos de dados.** Rio de Janeiro: Elsevier, 1997.
- CORONEL, C.; ROB, P. **Sistema de banco de dados:** projeto, implementação e administração. São Paulo: Cengage Learning, 2011.
- DATE, C. J. **Introdução a sistemas de bancos de dados.** Rio de Janeiro, RJ: Elsevier, 2003.
- GUIMARÃES, C. C. **Fundamentos para bancos de dados:** modelagem, projeto e linguagem SQL. Campinas: Ed. da Unicamp, 2003.
- KORTH, H. F.; SILBERSCHATZ, A.; SUDARSHAN, S. **Sistema de banco de dados.** 5. ed. São Paulo, SP: Makron, 2006.
- HEUSER, Carlos Alberto. **Projeto de Banco de Dados.** 6. ed. Bookman, 2011.
- NAVATHE, S. B.; RAMEZ, E. **Sistemas de banco de dados.** 4. ed. São Paulo: Addison Wesley, 2005.

ISBN 978-85-522-1154-9



9 788552 211549 >