Elmasri • Navathe

Sistemas de banco de dados

6ª edição

ALWAYS LEARNING

PEARSON

Elmasri • Navathe

Sistemas de banco de dados

6ª edição

Ramez Elmasri
Departamento de Ciência da Computação e Engenharia
Universidade do Texas em Arlington
Shamkant B. Navathe
Faculdade de Computação
Georgia Institute of Technology

Tradução:
Daniel Vieira
Revisão técnica:
Enzo Seraphim
Thatyana de Faria Piola Seraphim

Professores Doutores do Instituto de Engenharia de Sistemas e Tecnologias da Informação — Universidade Federal de Itajubá

PEARSON

abdr ASSOCIAÇÃO BRASILEIRA DE DIREITOS REPROGRÁFICOS
Respeite o direito autoral

© 2011 by Pearson Education do Brasil.
© 2011, 2007, 2004, 2000, 1994 e 1989 Pearson Education, Inc.

Tradução autorizada a partir da edição original, em inglês, *Fundamentals of Database Systems*, 6th edition, publicada pela Pearson Education, Inc., sob o selo Addison-Wesley.

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 Pearson Education do Brasil.

Diretor editorial: Roger Trimer
Gerente editorial: Sabrina Cairo
Supervisor de produção editorial: Marcelo Françozo
Editora plena: Thelma Babaoka
Editora de texto: Sabrina Levensteinas
Preparação: Paula Brandão Perez Mendes
Revisão: Elisa Andrade Buzzo
Capa: Thyago Santos sobre o projeto original de Lou Gibbs/Getty Images
Projeto gráfico e diagramação: Globaltec Artes Gráficas Ltda.

Dados Internacionais de Catalogação na Publicação (CIP)
(Câmara Brasileira do Livro, SP, Brasil)

Elmasri, Ramez

Sistemas de banco de dados / Ramez Elmasri e Shamkant B. Navathe ; tradução Daniel Vieira ; revisão técnica Enzo Seraphim e Thatyana de Faria Piola Seraphim. -- 6. ed. -- São Paulo : Pearson Addison Wesley, 2011.

Título original: Fundamentals of database systems.

ISBN 978-85-4301-381-7

1. Banco de dados I. Navathe, Shamkant B..II. Título.

10-11462 CDD-005.75

Índices para catálogo sistemático:
1. Banco de dados : Sistemas : Processamento de dados 005.75
2. Banco de dados : Fundamentos : Processamento de dados 005.75

3ª reimpressão – Julho 2014
Direitos exclusivos para a língua portuguesa cedidos à
Pearson Education do Brasil Ltda.,
uma empresa do grupo Pearson Education
Rua Nelson Francisco, 26
CEP 02712-100 – São Paulo – SP – Brasil
Fone: (11) 2178-8686 – Fax: (11) 2178-8688
vendas@pearson.com

capítulo 7

Modelagem de dados usando o modelo Entidade-Relacionamento (ER)

A modelagem conceitual é uma fase muito importante no projeto de uma aplicação de banco de dados bem-sucedida. Geralmente, o termo **aplicação de banco de dados** refere-se a um banco de dados em particular e aos programas associados que implementam as consultas e atualizações dele. Por exemplo, uma aplicação de banco de dados para um BANCO que controla contas de clientes incluiria programas que implementam atualizações ao banco de dados correspondentes a depósitos e saques de clientes. Esses programas oferecem interfaces gráficas com o usuário (GUIs) de fácil utilização, com formulários e menus para os usuários finais da aplicação — os caixas de banco, neste exemplo. Logo, uma parte importante da aplicação de banco de dados exigirá o projeto, a implementação e o teste desses programas de aplicação. Tradicionalmente, o projeto e o teste dos **programas de aplicação** têm sido considerados parte da *engenharia de software*, em vez do *projeto de banco de dados*. Em muitas ferramentas de projeto de software, as metodologias de projeto de banco de dados e de engenharia de software são interligadas, pois essas atividades estão fortemente relacionadas.

Neste capítulo, abordamos a técnica tradicional de concentrar nas estruturas e restrições de banco de dados durante seu projeto conceitual. O projeto de programas de aplicação normalmente é abordado em cursos de engenharia de software. Apresentamos os conceitos de modelagem do **modelo Entidade-Relacionamento (ER)**, que é um modelo de dados conceitual popular de alto nível. Esse modelo e suas variações costumam ser utilizados para o projeto conceitual de aplicações de banco de dados, e muitas ferramentas de projeto de banco de dados empregam seus conceitos. Descrevemos os conceitos e restrições básicas de estruturação de dados do modelo ER e discutimos seu uso no projeto de esquemas conceituais para aplicações de banco de dados. Também apresentamos a notação diagramática associada ao modelo ER, conhecido como **diagramas ER**.

As metodologias de modelagem de objeto, como a **Unified Modeling Language** (**UML**), estão se tornando cada vez mais populares no projeto e software de banco de dados. Essas metodologias vão além do projeto de banco de dados para especificar o projeto detalhado dos módulos de software e suas interações usando vários tipos de diagramas. Uma parte importante dessas metodologias — a saber, os *diagramas de classe*[1] — é semelhante, de muitas maneiras, aos diagramas ER. Nos diagramas de classe são especificadas *operações* sobre objetos, além da especificação da estrutura do esquema do banco de dados. As operações podem ser usadas para especificar os *requisitos funcionais* durante o projeto de banco de dados, conforme discutiremos na Seção 7.1. Apresentamos parte da notação e conceitos em UML para diagramas de classe que são particularmente relevantes ao projeto de banco de dados da Seção 7.8, e comparamos rapidamente estes com a notação e conceitos de ER. Notação e conceitos de UML adicionais serão apresentados na Seção 8.6 e no Capítulo 10.

Este capítulo está organizado da seguinte forma: a Seção 7.1 discute o papel dos modelos de dados conceituais de alto nível no projeto de banco de dados. Apresentamos os requisitos para uma aplicação de banco de dados de exemplo na Seção 7.2, para

[1] Uma **classe** é semelhante a um *tipo de entidade* de várias maneiras.

ilustrar o uso dos conceitos do modelo ER. Esse banco de dados de exemplo também é usado no decorrer do livro. Na Seção 7.3, apresentamos os conceitos de entidades e atributos, e gradualmente introduzimos a técnica diagramática para exibir um esquema ER. Na Seção 7.4, apresentamos os conceitos de relacionamentos binários e suas funções e restrições estruturais. A Seção 7.5 apresenta os tipos de entidade fraca. A Seção 7.6 mostra como um projeto de esquema é refinado para incluir relacionamentos. A Seção 7.7 analisa a notação para diagramas ER, resume os problemas e armadilhas comuns que ocorrem no projeto de esquema e discute como escolher os nomes para construções de esquema de banco de dados. A Seção 7.8 apresenta alguns conceitos de diagrama de classe UML, compara-os com os conceitos do modelo ER e os aplica ao mesmo banco de dados de exemplo. A Seção 7.9 discute tipos de relacionamentos mais complexos. No final há um resumo do capítulo.

O material nas seções 7.8 e 7.9 pode ser excluído de um curso introdutório. Se uma cobertura mais completa dos conceitos de modelagem de dados e projeto de banco de dados conceitual for desejada, o leitor deverá continuar até o Capítulo 8, onde descrevemos as extensões ao modelo ER que levam ao modelo EER (ER Estendido), o qual inclui conceitos como especialização, generalização, herança e tipos (categorias) de união. Também apresentaremos alguns conceitos adicionais e a notação de UML no Capítulo 8.

7.1 Usando modelos de dados conceituais de alto nível para o projeto do banco de dados

A Figura 7.1 mostra uma visão geral simplificada do processo de projeto de banco de dados. A primeira etapa mostrada é o **levantamento e análise de requisitos**. Durante essa etapa, os projetistas de banco de dados entrevistam os usuários esperados para entenderem e documentarem seus **requisitos de dados**. O resultado dessa etapa é um conjunto de requisitos dos usuários escrito de forma concisa. Esses requisitos devem ser especificados da forma mais detalhada e completa possível. Em paralelo com a especificação dos requisitos de dados, é útil determinar os conhecidos **requisitos funcionais** da aplicação. Estes consistem em **operações** (ou **transações**) definidas pelo usuário, que serão aplicadas ao banco de dados, incluindo recuperações e atualizações. No projeto de software, é comum usar *diagramas de fluxo de dados*, *diagramas de sequência*, *cenários* e outras técnicas para especificar requisitos funcionais. Não discutiremos essas técnicas aqui; elas normalmente são descritas em detalhes nos textos de engenharia de software. Daremos uma visão geral dessas técnicas no Capítulo 10.

Assim que os requisitos tiverem sido levantados e analisados, a próxima etapa é criar um **esquema conceitual** para o banco de dados, usando um modelo de dados conceitual de alto nível. Essa etapa é chamada de **projeto conceitual**. O esquema conceitual é uma descrição concisa dos requisitos de dados dos usuários e inclui detalhes dos tipos de entidade, relacionamentos e restrições; estes são expressos usando os conceitos fornecidos pelo modelo de dados de alto nível. Como não incluem descrições detalhadas de implementação, esses conceitos normalmente são mais fáceis de entender e podem ser usados para a comunicação com usuários não técnicos. O esquema conceitual de alto nível também pode ser utilizado como uma referência para garantir que todos os requisitos de dados dos usuários sejam atendidos e que não estejam em conflito. Essa técnica permite que os projetistas de banco de dados se concentrem em especificar as propriedades dos dados, sem se preocuparem com detalhes de armazenamento e implementação. Isso torna mais fácil criar um bom projeto de banco de dados conceitual.

Durante ou após o projeto do esquema conceitual, as operações básicas do modelo de dados podem ser usadas para especificar as consultas e operações do usuário de alto nível, identificadas durante a análise funcional. Isso também serve para confirmar se o esquema conceitual atende a todos os requisitos funcionais identificados. Modificações no esquema conceitual podem ser introduzidas, se alguns requisitos funcionais não puderem ser especificados usando o esquema inicial.

A próxima etapa no projeto de banco de dados é a implementação real do próprio banco de dados, usando um SGBD comercial. A maioria dos SGBDs comerciais utiliza um modelo de dados de implementação — como o modelo de banco de dados relacional ou objeto-relacional —, de modo que o esquema conceitual é transformado do modelo de dados de alto nível para o modelo de dados da implementação. Essa etapa é chamada de **projeto lógico** ou **mapeamento do modelo de dados**. Seu resultado é um esquema de banco de dados no modelo de dados da implementação do SGBD. O mapeamento do modelo de dados normalmente é automatizado ou semiautomatizado nas ferramentas de projeto do banco de dados.

A última etapa é a fase do **projeto físico**, durante a qual as estruturas de armazenamento internas, organizações de arquivo, índices, caminhos de acesso e parâmetros físicos do projeto para os arquivos do banco de dados são especificados. Em paralelo com essas atividades, os programas de aplicação são projetados e implementados como transações de banco de dados correspondentes às especificações da transação de alto nível. Discutiremos o processo de projeto do banco de dados com mais detalhes no Capítulo 10.

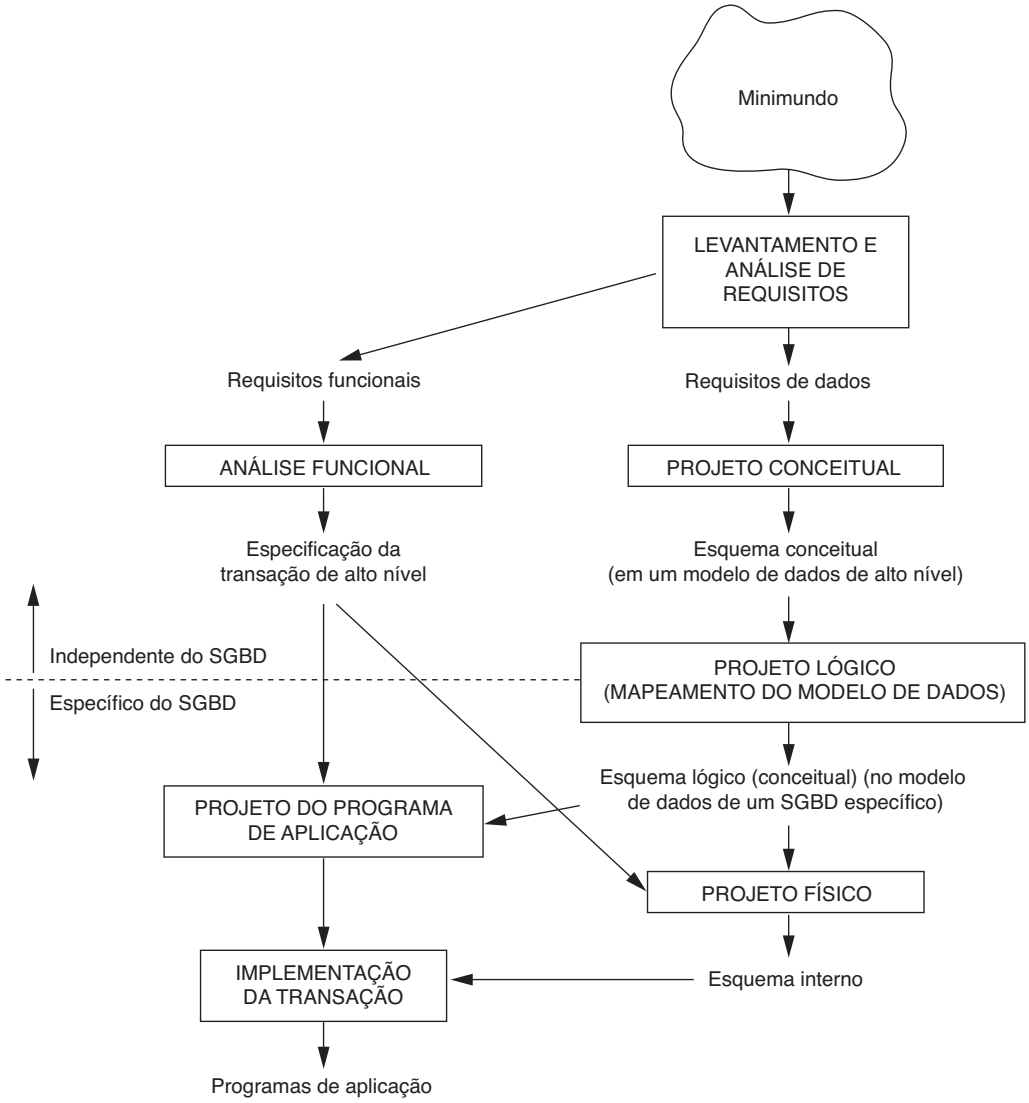

Figura 7.1
Um diagrama simplificado para ilustrar as principais fases do projeto de banco de dados.

Neste capítulo, apresentamos apenas os conceitos básicos do modelo ER para o projeto do esquema conceitual. Outros conceitos de modelagem serão discutidos no Capítulo 8, quando apresentaremos o modelo EER.

7.2 Exemplo de aplicação de banco de dados

Nesta seção, descrevemos um exemplo de aplicação de banco de dados, chamado EMPRESA, que serve para ilustrar os conceitos básicos do modelo ER e seu uso no projeto do esquema. Listamos os requisitos de dados para o banco de dados aqui, e depois criaremos seu esquema conceitual passo a passo, quando introduzirmos os conceitos de modelagem do modelo ER. O banco de dados EMPRESA registra os funcionários, departamentos e projetos de uma empresa. Suponha que, depois da fase de levantamento e análise de requisitos, os projetistas de banco de dados ofereçam a seguinte descrição do *minimundo* — a parte da empresa que será representada no banco de dados:

- A empresa é organizada em departamentos. Cada departamento tem um nome exclusivo, um número exclusivo e um funcionário em particular que o gerencia. Registramos a data inicial em que esse funcionário começou a gerenciar o departamento. Um departamento pode ter vários locais.
- Um departamento controla uma série de projetos, cada um deles com um nome exclusivo, um número exclusivo e um local exclusivo.

- Armazenamos o nome, número do Cadastro de Pessoa Física,[2] endereço, salário, sexo (gênero) e data de nascimento de cada funcionário. Um funcionário é designado para um departamento, mas pode trabalhar em vários projetos, que não necessariamente são controlados pelo mesmo departamento. Registramos o número atual de horas por semana que um funcionário trabalha em cada projeto. Também registramos o supervisor direto de cada funcionário (que é outro funcionário).
- Queremos registrar os dependentes de cada funcionário para fins de seguro. Para cada dependente, mantemos o nome, sexo, data de nascimento e parentesco com o funcionário.

A Figura 7.2 mostra como o esquema para essa aplicação de banco de dados pode ser exibido por meio da notação gráfica conhecida como **diagramas ER**. Essa figura será explicada gradualmente à medida que os conceitos do modelo ER forem apresentados. Descrevemos o processo passo a passo da derivação desse esquema com base nos requisitos declarados — e explicamos a notação diagramática ER — à medida que introduzirmos os conceitos do modelo ER.

7.3 Tipos de entidade, conjuntos de entidades, atributos e chaves

O modelo ER descreve os dados como *entidades*, *relacionamentos* e *atributos*. Na Seção 7.3.1, apresentamos os conceitos de entidades e seus atributos. Discutimos os tipos de entidade e os principais atributos na Seção 7.3.2. Depois, na Seção 7.3.3, especificamos o projeto conceitual inicial dos tipos de entidade para o banco de dados EMPRESA. Os relacionamentos são descritos na Seção 7.4.

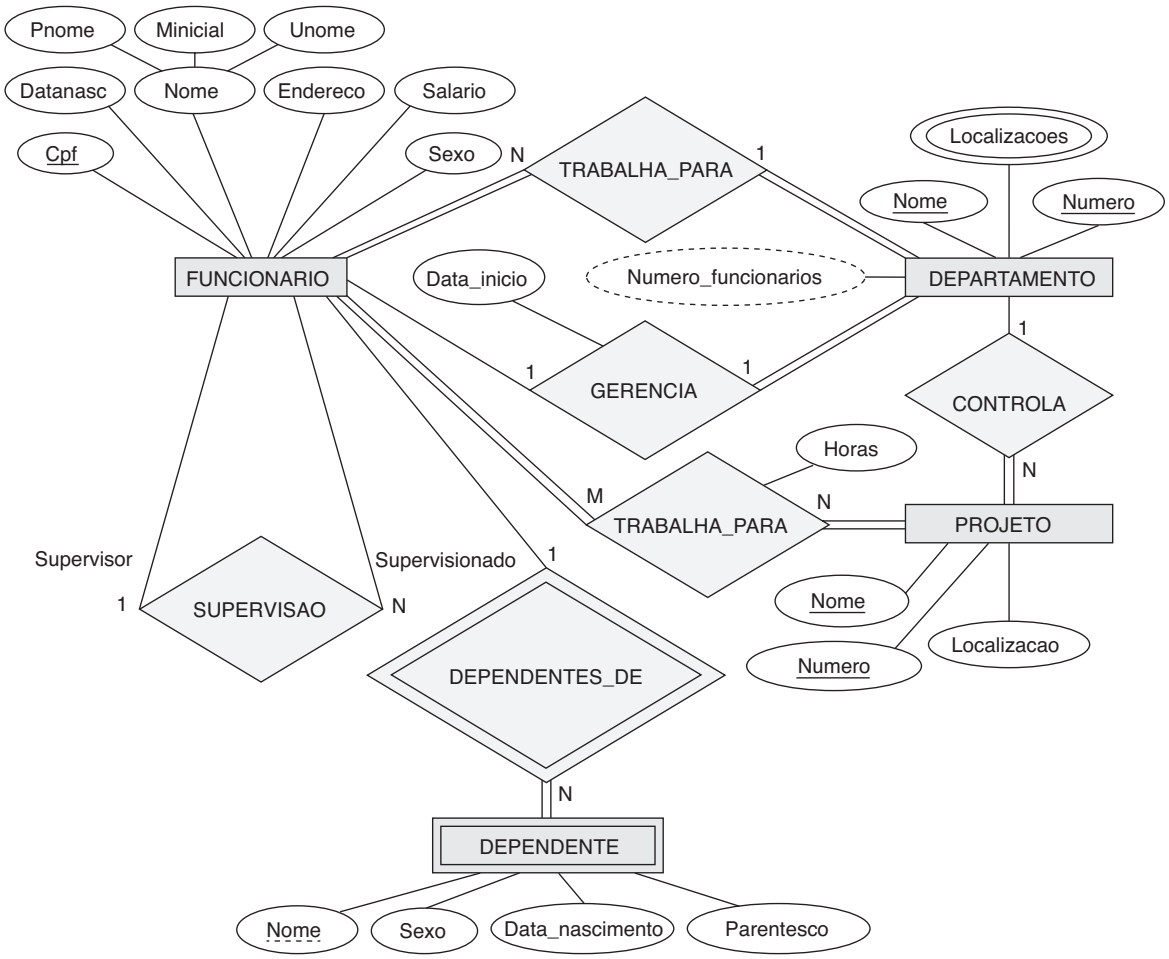

Figura 7.2
Um diagrama de esquema ER para o banco de dados EMPRESA. A notação diagramática é apresentada gradualmente no decorrer do capítulo e resumida na Figura 7.14.

[2] O número do Cadastro de Pessoa Física (CPF) é um identificador exclusivo de onze dígitos, atribuído a cada indivíduo no Brasil, para registrar seu emprego, benefícios e impostos. Outros países possuem esquemas de identificação semelhantes, como números do Seguro Social (SSN) e identificação civil.

7.3.1 Entidades e atributos

Entidades e seus atributos. O objeto básico que o modelo ER representa é uma **entidade**, que é algo no mundo real com uma existência independente. Uma entidade pode ser um objeto com uma existência física (por exemplo, uma pessoa em particular, um carro, uma casa ou um funcionário), ou pode ser um objeto com uma existência conceitual (por exemplo, uma empresa, um cargo ou um curso universitário). Cada entidade possui **atributos** — as propriedades específicas que a descrevem. Por exemplo, uma entidade FUNCIONARIO pode ser descrita pelo nome, idade, endereço, salário e cargo do funcionário. Uma entidade em particular terá um valor para cada um de seus atributos. Os valores de atributo que descrevem cada entidade tornam-se uma parte importante dos dados armazenados no banco de dados.

A Figura 7.3 mostra duas entidades e os valores de seus atributos. A entidade FUNCIONARIO f_1 tem quatro atributos: Nome, Endereco, Idade e Telefone_residencial; seus valores são 'João Silva,' 'Rua das Flores, 751, São Paulo, SP, 07700110', '55' e '13-4749-2630', respectivamente. A entidade EMPRESA e_1 tem três atributos: Nome, Matriz e Presidente; seus valores são 'Companhia Modelo', 'São Paulo' e 'João Silva', respectivamente.

Vários tipos de atributos ocorrem no modelo ER: *simples* versus *composto*, *valor único* versus *multivalorado*, e *armazenado* versus *derivado*. Primeiro, vamos definir esses tipos de atributo e ilustrar seu uso por meio de exemplos. Depois, discutiremos o conceito de um *valor* NULL para um atributo.

Atributos compostos *versus* simples (atômicos). **Atributos compostos** podem ser divididos em subpartes menores, que representam atributos mais básicos, com significados independentes. Por exemplo, o atributo Logradouro da entidade FUNCIONARIO mostrada na Figura 7.3 pode ser subdividido em Logradouro, Cidade, Estado e Cep,[3] com os valores 'Rua das Flores, 751', 'São Paulo', 'SP' e '07700110.' Os atributos não divisíveis são chamados **atributos simples** ou **atômicos**. Os atributos compostos podem formar uma hierarquia; por exemplo, Logradouro pode ser subdividido em três atributos simples: Numero, Rua e Numero_apartamento, como mostra a Figura 7.4. O valor de um atributo composto é a concatenação dos valores de seus componentes atributos simples.

Atributos compostos são úteis para modelar situações em que um usuário às vezes se refere ao atributo composto como uma unidade, mas outras vezes se refere especificamente a seus componentes. Se o atributo composto for referenciado apenas como um todo, não é necessário subdividi-lo em atributos componentes. Por exemplo, se não for preciso referenciar os componentes individuais de um endereço (CEP, rua etc.), então o endereço inteiro pode ser designado como um atributo simples.

Atributos de valor único *versus* multivalorados. A maioria dos atributos possui um valor único para uma entidade em particular; tais atributos são chamados de **valor único**. Por exemplo, Idade é um atributo de valor único de uma pessoa. Em alguns casos, um atributo pode ter um conjunto de valores para a mesma entidade — por exemplo, um atributo Cores para um carro, ou um atributo Formacao_academica para uma pessoa. Os carros com uma cor têm um único valor, enquanto os carros com duas cores possuem dois valores de cor. De modo semelhante, uma pessoa pode não ter formação acadêmica, outra pessoa pode ter, e uma terceira pode ter duas ou mais formações; portanto, diferentes pessoas podem ter distintos *números* de *valores* para o atributo Formacao_academica. Esses atributos são chamados de **multivalorados**. Um atributo multivalorado pode ter um limite mínimo e um máximo para restringir o *número de valores* permitidos para cada entidade individual. Por exemplo, o atributo Cores de

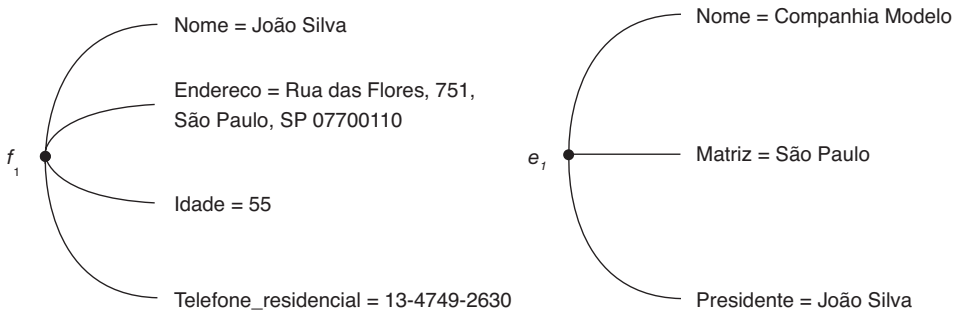

Figura 7.3
Duas entidades, FUNCIONARIO f_1 e EMPRESA e_1, e seus atributos.

[3] CEP (Código de Endereçamento Postal) é o nome usado no Brasil para um código postal com oito dígitos, como 07601-090.

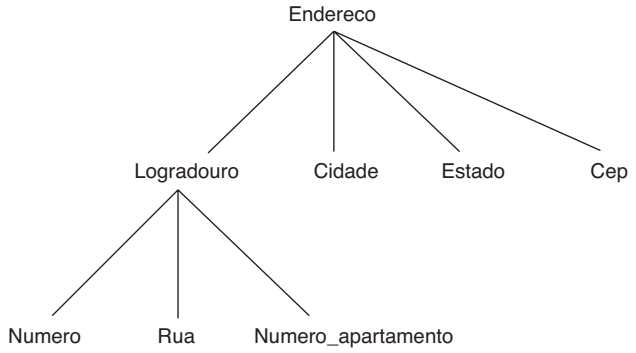

Figura 7.4
Uma hierarquia de atributos compostos.

um carro pode ser restrito a ter entre um e três valores, se considerarmos que um carro pode ter no máximo três cores.

Atributos armazenados *versus* derivados. Em alguns casos, dois (ou mais) valores de atributo estão relacionados — por exemplo, os atributos Idade e Data_nascimento de uma pessoa. Para uma entidade de pessoa em particular, o valor de Idade pode ser determinado pela data atual (hoje) e o valor da Data_nascimento dessa pessoa. O atributo Idade, portanto, é chamado de **atributo derivado** e considerado **derivável** do atributo Datanasc, que é chamado, por sua vez, de **atributo armazenado**. Alguns valores de atributo podem ser derivados de *entidades relacionadas*; por exemplo, um atributo Numero_funcionarios de uma entidade DEPARTAMENTO pode ser derivado contando-se o número de funcionários relacionados a (trabalhando para) esse departamento.

Valores NULL. Em alguns casos, uma entidade em particular pode não ter um valor aplicável para um atributo. Por exemplo, o atributo Numero_apartamento de um endereço só se aplica a endereços que estão em prédios de apartamento, e não a outros tipos de residências, como casas. De modo semelhante, um atributo Formacao_academica só se aplica a pessoas com esse tipo de formação. Para tais situações, foi criado um valor especial, chamado NULL. Um endereço de uma casa teria NULL para seu atributo Numero_apartamento, e uma pessoa sem formação acadêmica teria NULL para Formacao_academica. NULL também pode ser usado quando não conhecemos o valor de um atributo para determinada entidade — por exemplo, se não soubermos o número do telefone residencial de 'João Silva' na Figura 7.3. O significado do primeiro tipo de NULL é *não aplicável*, enquanto o significado do segundo é *desconhecido*. A categoria *desconhecido* de NULL pode ser classificada

{Endereço_telefone({Telefone(Codigo_area, Numero_telefone)},Endereco(Logradouro (Numero,Rua,Numero_apartamento),Cidade,Estado,Cep))}

Figura 7.5
Um atributo complexo: Endereço_telefone.

ainda em mais dois casos. O primeiro caso acontece quando se sabe que o valor do atributo existe, mas está *faltando* — por exemplo, se o atributo Altura de uma pessoa for listado como NULL. O segundo caso surge quando *não se sabe* se o valor do atributo existe — por exemplo, se o atributo Telefone_residencial de uma pessoa for NULL.

Atributos complexos. Observe que, em geral, os atributos compostos e multivalorados podem ser aninhados arbitrariamente. Podemos representar o aninhamento arbitrário ao agrupar componentes de um atributo composto entre parênteses ( ) e separá-los com vírgulas, e ao exibir os atributos multivalorados entre chaves { }. Esses atributos são chamados de **atributos complexos**. Por exemplo, se uma pessoa pode ter mais de uma residência e cada residência pode ter um único endereço e vários telefones, um atributo Endereço_telefone para uma pessoa pode ser especificado como na Figura 7.5.[4] Tanto Telefone quanto Endereco são atributos compostos.

7.3.2 Tipos de entidade, conjuntos de entidade, chaves e conjuntos de valores

Tipos de entidade e conjuntos de entidade. Um banco de dados em geral contém grupos de entidades que são semelhantes. Por exemplo, uma empresa que emprega centenas de funcionários pode querer armazenar informações semelhantes com relação a cada um dos funcionários. Essas entidades de funcionário compartilham os mesmos atributos, mas cada uma tem *o(s) próprio(s) valor(es)* para cada atributo. Um **tipo de entidade** define uma *coleção* (ou *conjunto*) de entidades que têm os mesmos atributos. Cada tipo de entidade no banco de dados é descrito por seu nome e atributos. A Figura 7.6 mostra dois tipos de entidade: FUNCIONARIO e EMPRESA, e uma lista de alguns dos atributos para cada um. Algumas entidades individuais de cada tipo também são ilustradas, junto com os valores de seus atributos. A coleção de todas as entidades de determinado tipo de entidade no banco de dados, em qualquer ponto no tempo, é chamada de **conjunto de entidades**. Normalmente, refere-se ao conjunto de entidades para usar o mesmo nome do

[4] Para aqueles acostumados com XML, devemos observar que os atributos complexos são semelhantes aos elementos complexos em XML (ver Capítulo 12).

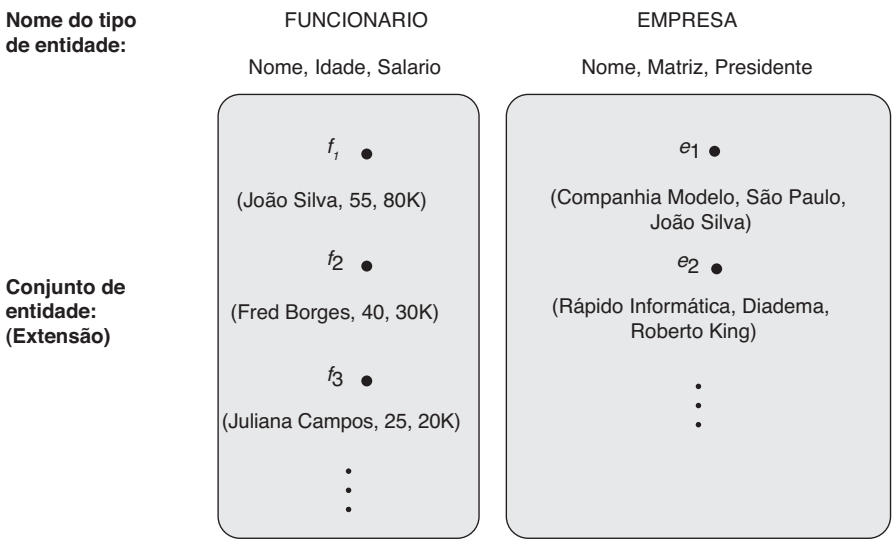

Figura 7.6
Dois tipos de entidade, FUNCIONARIO e EMPRESA, e algumas entidades membro de cada uma.

tipo de entidade. Por exemplo, FUNCIONARIO refere-se ao *tipo de entidade* e também ao conjunto atual *de todas as entidades de funcionário* no banco de dados.

Um tipo de entidade é representado nos diagramas ER[5] (ver Figura 7.2) como uma caixa retangular delimitando seu nome. Os nomes de atributo são delimitados em ovais, sendo ligados a seu tipo de entidade por linhas retas. Os atributos compostos são ligados aos seus atributos componentes por linhas retas. Os atributos multivalorados aparecem em ovais duplas. A Figura 7.7(a) mostra um tipo de entidade CARRO nessa notação.

Um tipo de entidade descreve o **esquema** ou **conotação** para um *conjunto de entidades* que compartilham a mesma estrutura. A coleção de entidades de determinado tipo é agrupada em um conjunto de entidades, que também é chamado de **extensão** do tipo de entidade.

Atributos-chave de um tipo de entidade. Uma restrição importante das entidades de um tipo de entidade é a **chave** ou **restrição de exclusividade** sobre os atributos. Um tipo de entidade normalmente tem um ou mais atributos cujos valores são distintos para cada entidade individual no conjunto de entidades. Esse atributo é denominado **atributo-chave**, e seus valores podem ser usados para identificar cada entidade de maneira exclusiva. Por exemplo, o atributo Nome é uma chave do tipo de entidade EMPRESA na Figura 7.6, pois duas empresas não podem ter o mesmo nome. Para o tipo de entidade PESSOA, um atributo-chave típico é o Cpf (Cadastro de Pessoa Física). Às vezes, vários atributos juntos formam uma chave, significando que a *combinação* dos valores de atributo deve ser distinta para cada entidade. Se um conjunto de atributos possui essa propriedade, o modo correto de representar isso no modelo ER que descrevemos aqui é definir um *atributo composto* e designá-lo como um atributo-chave do tipo de entidade. Observe que essa chave composta precisa ser *mínima*, ou seja, todos os atributos componentes precisam estar incluídos no atributo composto para ter a propriedade de exclusividade. Atributos supérfluos não devem ser incluídos em uma chave. Na notação diagramática ER, cada atributo-chave tem seu nome **sublinhado** dentro da oval, conforme ilustrado na Figura 7.7(a).

Especificar que um atributo é uma chave de um tipo de entidade significa que a propriedade anterior da exclusividade precisa ser mantida para *cada conjunto de entidades* do tipo de entidade. Logo, essa é uma restrição que proíbe que duas entidades tenham o mesmo valor para o atributo-chave ao mesmo tempo. Essa não é a propriedade de um conjunto de entidades em particular; em vez disso, é uma restrição sobre *qualquer conjunto de entidades* do tipo de entidade em qualquer ponto no tempo. Essa restrição-chave (e outras restrições que discutiremos mais adiante) é derivada das restrições do minimundo que o banco de dados representa.

Alguns tipos de entidade possuem *mais de um* atributo-chave. Por exemplo, cada um dos atributos Cod_veiculo e Registro do tipo de entidade CAR-

[5] Usamos a notação para diagramas ER, a qual é próxima da notação da proposta original (Chen, 1976). Muitas outras notações estão em uso; ilustraremos algumas delas mais adiante neste capítulo, quando apresentarmos os diagramas de classe UML, e no Apêndice A.

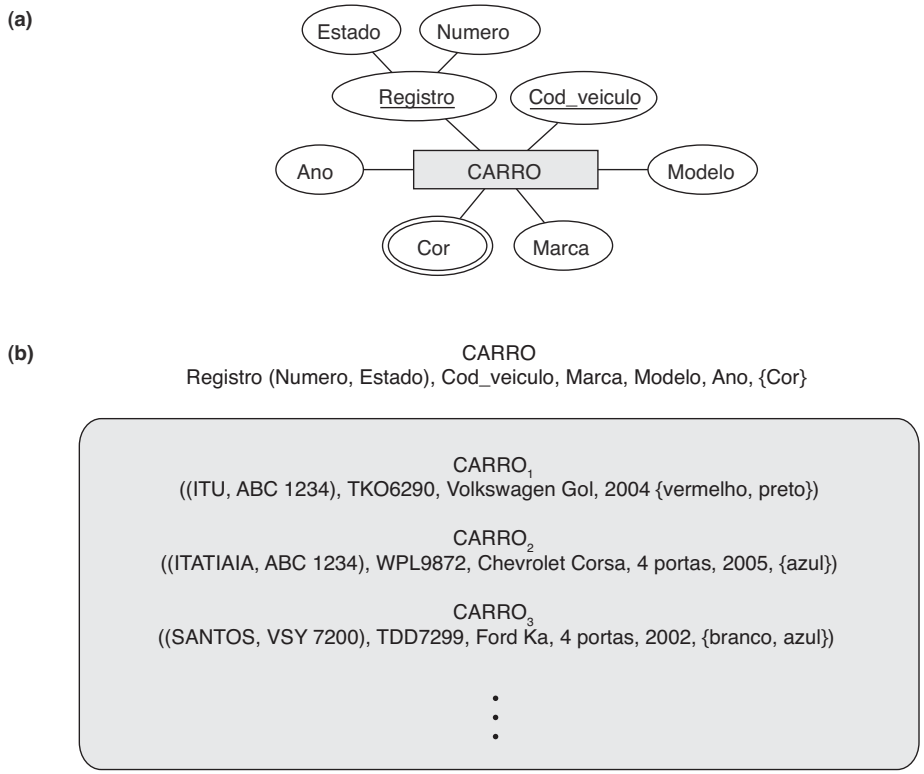

Figura 7.7
O tipo de entidade CARRO com dois atributos-chave, Registro e Cod_veiculo. (a) Notação do diagrama ER. (b) Conjunto de entidade com três entidades.

RO (Figura 7.7) é uma chave por si só. O atributo Registro é um exemplo de uma chave composta formada por dois atributos componentes simples, Estado e Numero, nenhum deles sendo uma chave por si só. Um tipo de entidade também *pode não ter chave*; nesse caso, ele é chamado de *tipo de entidade fraca* (ver Seção 7.5).

Em nossa notação diagramática, se dois atributos forem sublinhados separadamente, então *cada um é uma chave por si só*. Diferentemente do modelo relacional (ver Seção 3.2.2), não existe o conceito de chave primária no modelo ER que apresentamos aqui; a chave primária será escolhida durante o mapeamento para um esquema relacional (ver Capítulo 9).

Conjuntos (domínios) de valores dos atributos. Cada atributo simples de um tipo de entidade é associado a um **conjunto de valores** (ou **domínio** de valores), o qual especifica o conjunto de valores que podem ser designados a esse atributo para cada entidade individual. Na Figura 7.6, se o intervalo de idades permitidas para os funcionários estiver entre 16 e 70, podemos especificar o conjunto de valores do atributo Idade de FUNCIONARIO como sendo o conjunto de números inteiros entre 16 e 70. De modo semelhante, podemos especificar o conjunto de valores para o atributo Nome como sendo o conjunto de cadeias de caracteres alfabéticos separados por caracteres de espaço, e assim por diante. Os conjuntos de valores não são exibidos nos diagramas ER, e são com frequência especificados usando os **tipos de dados** básicos disponíveis na maioria das linguagens de programação, como inteiro, cadeia de caracteres, booleano, real, tipo enumerado, intervalo de valores, e assim por diante. Outros tipos de dados que representam tipos comuns no banco de dados, como data, hora e outros conceitos, também são empregados. Matematicamente, um atributo A do conjunto de entidades E, cujo conjunto de valores é V, pode ser definido como uma **função** de E ao conjunto de potência[6] $P(V)$ de V:

$$A : E \rightarrow P(V)$$

Referimo-nos ao valor do atributo A para a entidade e como $A(e)$. A definição anterior cobre tanto atributos de único valor quanto atributos multivalorados, bem como NULLs. Um valor NULL é representado pelo *conjunto vazio*. Para atributos de único valor, $A(e)$ é restrito a ser um *conjunto*

[6] O **conjunto de potência** $P(V)$ de um conjunto V é o conjunto de todos os subconjuntos de V.

singular[7] para cada entidade *e* em *E*, ao passo que não existe restrição sobre atributos multivalorados. Para um atributo composto *A*, o conjunto de valores *V* é o conjunto de potência do produto Cartesiano de $P(V_1)$, $P(V_2)$, ..., $P(V_n)$, onde V_1, V_2, ..., V_n são os conjuntos de valores dos atributos componentes simples que formam *A*:

$$V = P\,(P(V_1) \times P(V_2) \times \ldots \times P(V_n))$$

O conjunto de valores oferece todos os valores possíveis. Em geral, apenas um pequeno número desses valores existe no banco de dados em determinado momento. Esses valores representam os dados do estado atual do minimundo. Eles correspondem aos dados conforme realmente existem no minimundo.

7.3.3 Projeto conceitual inicial do banco de dados EMPRESA

Agora, podemos definir os tipos de entidade para o banco de dados EMPRESA, com base nos requisitos descritos na Seção 7.2. Após definir aqui vários tipos de entidade e seus atributos, refinamos nosso projeto na Seção 7.4, depois de introduzir o conceito de um relacionamento. De acordo com os requisitos listados na Seção 7.2, podemos identificar quatro tipos de entidade — uma correspondente a cada um dos quatro itens na especificação (ver Figura 7.8):

1. Um tipo de entidade DEPARTAMENTO com atributos Nome, Numero, Localizacoes, Gerente e Data_inicio_gerente. Localizacoes é o único atributo multivalorado. Podemos especificar que tanto Nome quanto Numero são atributos-chave (separados), pois cada um foi especificado como sendo exclusivo.
2. Um tipo de entidade PROJETO com atributos Nome, Numero, Localizacao e Departamento_gerenciador. Tanto Nome quanto Numero são atributos-chave (separados).
3. Um tipo de entidade FUNCIONARIO com atributos Nome, Cpf, Sexo, Endereco, Salario, Data_nascimento, Departamento e Supervisor. Tanto Nome quanto Endereco podem ser atributos compostos; no entanto, isso não foi especificado nos requisitos. Temos de voltar aos usuários para ver se algum deles irá se referir aos componentes individuais de Nome — Primeiro_nome, Inicial_meio, Ultimo_nome — ou de Endereco.
4. Um tipo de entidade DEPENDENTE com atributos Funcionario, Nome_dependente, Sexo, Data_nascimento e Parentesco (para o funcionário).

Até aqui, não representamos o fato de que um funcionário pode trabalhar em vários projetos nem o número de horas por semana que um funcionário trabalha em cada projeto. Essa característica é listada como parte do terceiro requisito na Seção 7.2, e pode ser representada por um atributo composto multivalorado de FUNCIONARIO, chamado Trabalha_em, com os componentes simples (Projeto, Horas). Como alternativa, ela pode ser representada como um atributo composto multivalorado de PROJETO, chamado Trabalhadores, com os componentes simples (Funcionario, Horas). Escolhemos a primeira alternativa na Figura 7.8, que mostra cada um dos tipos de entidade descritos. O atributo Nome de FUNCIONARIO aparece como um atributo composto, aparentemente após a consulta com os usuários.

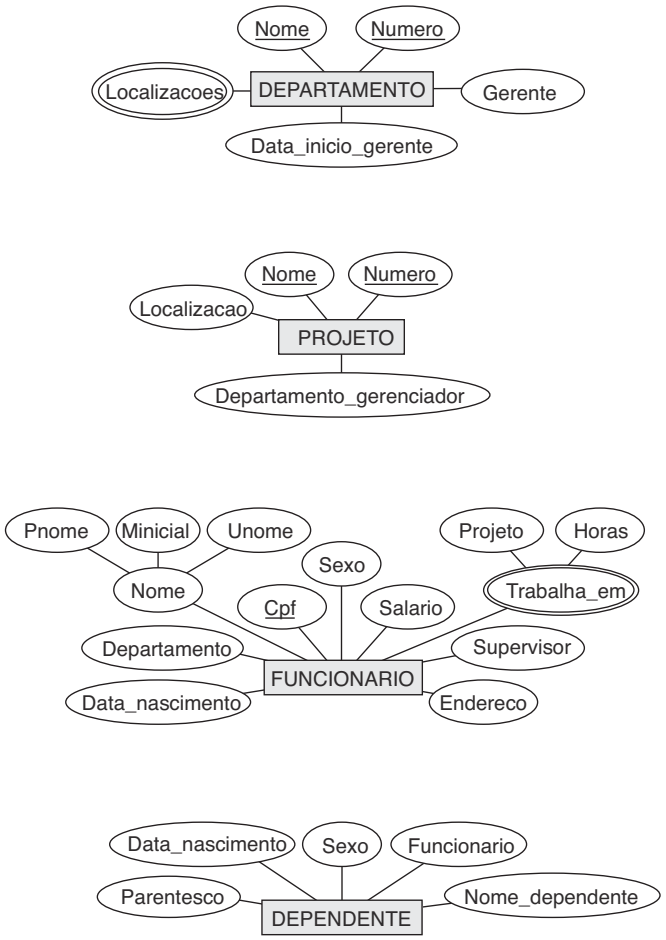

Figura 7.8
Projeto preliminar de tipos de entidade para o banco de dados EMPRESA. Alguns dos atributos mostrados serão refinados nos relacionamentos.

[7] Um conjunto **singular** (ou *singleton*) é um conjunto com apenas um elemento (valor).

7.4 Tipos e conjuntos de relacionamentos, papéis e restrições estruturais

Na Figura 7.8 existem vários *relacionamentos implícitos* entre os diversos tipos de entidade. De fato, sempre que um atributo de um tipo de entidade se refere a outro tipo de entidade, existe algum relacionamento. Por exemplo, o atributo Gerente de DEPARTAMENTO refere-se a um funcionário que gerencia o departamento; o atributo Departamento_gerenciador de PROJETO refere-se ao departamento que controla o projeto; o atributo Supervisor de FUNCIONARIO refere-se a outro funcionário (aquele que supervisiona esse funcionário); o atributo Departamento de FUNCIONARIO refere-se ao departamento para o qual o funcionário trabalha, e assim por diante. No modelo ER, essas referências não devem ser representadas como atributos, mas como **relacionamentos**, que são discutidos nesta seção. O esquema do banco de dados EMPRESA será refinado na Seção 7.6 para representar relacionamentos de maneira explícita. No projeto inicial dos tipos de entidade, os relacionamentos normalmente são capturados na forma de atributos. À medida que o projeto é refinado, esses atributos são convertidos em relacionamentos entre os tipos de entidade.

Este tópico é organizado da seguinte forma: a Seção 7.4.1 apresenta os conceitos dos tipos, conjuntos e instâncias de relacionamento. Definimos os conceitos de grau de relacionamento, nomes de função e relacionamentos recursivos na Seção 7.4.2, e depois discutimos as restrições estruturais sobre os relacionamentos — como as razões de cardinalidade e dependências de existência — na Seção 7.4.3. A Seção 7.4.4 mostra como os tipos de relacionamento também podem ter atributos.

7.4.1 Tipos, conjuntos e instâncias de relacionamento

Um **tipo de relacionamento** R entre n tipos de entidade $E_1, E_2, ..., E_n$ define um conjunto de associações — ou um **conjunto de relacionamento** — entre as entidades desses tipos de entidade. Assim como no caso dos tipos de entidade e conjuntos de entidade, um tipo de relacionamento e seu conjunto de relacionamento correspondente em geral são referenciados pelo *mesmo nome*, R. Matematicamente, o conjunto de relacionamento R é um conjunto de **instâncias de relacionamento** r_i, onde cada r_i associa-se a n entidades individuais $(e_1, e_2, ..., e_n)$, e cada entidade e_j em r_i é um membro do conjunto de entidades E_j, $1 \leq j \leq n$. Logo, um conjunto de relacionamento é uma relação matemática sobre $E_1, E_2, ..., E_n$. Como alternativa, ele pode ser definido como um subconjunto do produto cartesiano dos conjuntos de entidades $E_1 \times E_2 \times ... \times E_n$. Cada um dos tipos de entidade $E_1, E_2, ..., E_n$ é dito participar no tipo de relacionamento R; de modo semelhante, cada uma das entidades individuais $e_1, e_2, ..., e_n$ é dito **participar** na instância de relacionamento $r_i = (e_1, e_2, ..., e_n)$.

Informalmente, cada instância de relacionamento r_i em R é uma associação de entidades, onde a associação inclui exatamente uma entidade de cada tipo de entidade participante. Cada instância de relacionamento r_i desse tipo representa o fato de que as entidades participantes em r_i estão relacionadas de alguma maneira na situação do minimundo correspondente. Por exemplo, considere um tipo de relacionamento TRABALHA_PARA entre os dois tipos de entidade FUNCIONARIO e DEPARTAMENTO, que associa cada funcionário ao departamento para o qual o funcionário trabalha no conjunto de entidades correspondente. Cada instância de relacionamento no conjunto de relacionamentos TRABALHA_PARA associa uma entidade FUNCIONARIO a uma entidade DEPARTAMENTO. A Figura 7.9 ilustra esse exemplo, onde cada instância de relacionamento r_i aparece conectada às entidades FUNCIONARIO e DEPARTAMENTO que participam em r_i. No minimundo representado pela Figura 7.9, os funcionários f_1, f_3 e f_6 trabalham para o departamento d_1; os funcionários f_2 e f_4 trabalham para o departamento d_2; e os funcionários f_5 e f_7 trabalham para o departamento d_3.

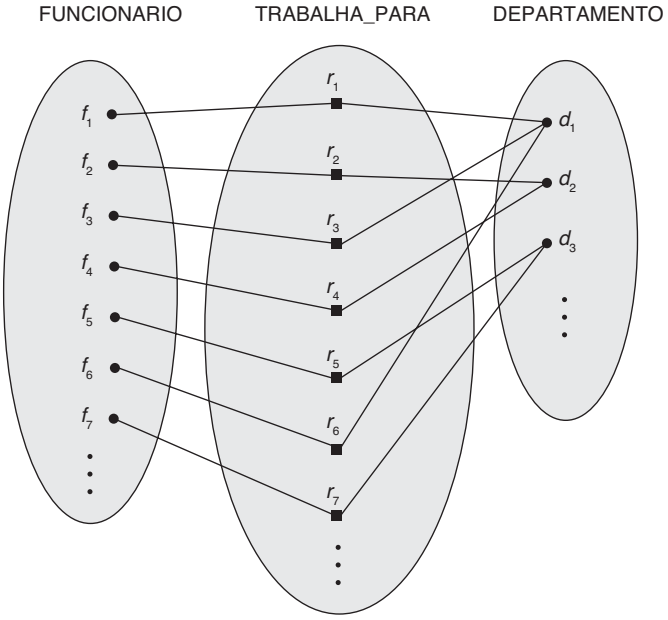

Figura 7.9
Algumas instâncias no conjunto de relacionamentos TRABALHA_PARA, que representa um tipo de relacionamento TRABALHA_PARA entre FUNCIONARIO e DEPARTAMENTO.

Nos diagramas ER, os tipos de relacionamento são exibidos como caixas em forma de losango, que são conectadas por linhas retas às caixas retangulares que representam os tipos de entidade participantes. O nome do relacionamento é exibido na caixa em forma de losango (ver Figura 7.2).

7.4.2 Grau de relacionamento, nomes de função e relacionamentos recursivos

Grau de um tipo de relacionamento. O **grau** de um tipo de relacionamento é o número dos tipos de entidade participantes. Logo, o relacionamento TRABALHA_PARA tem grau dois. Um tipo de relacionamento de grau dois é chamado de **binário**, e um tipo de grau três é chamado de **ternário**. Um exemplo de relacionamento ternário é FORNECE, mostrado na Figura 7.10, em que cada instância de relacionamento r_i associa três entidades — um fornecedor f, uma peça p e um projeto j — sempre que f fornece a peça p ao projeto j. Os relacionamentos geralmente podem ser de qualquer grau, mas os mais comuns são os relacionamentos binários. Relacionamentos de grau mais alto geralmente são mais complexos do que os binários. Nós os caracterizaremos mais adiante, na Seção 7.9.

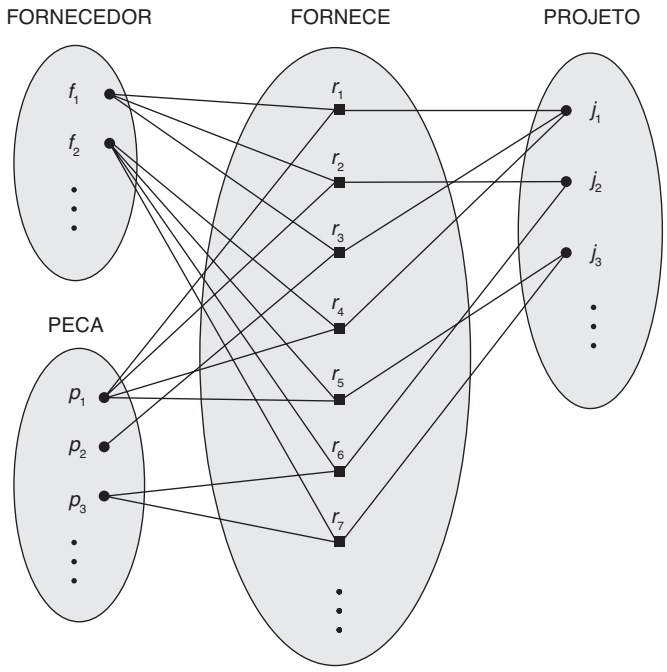

Figura 7.10
Algumas instâncias de relacionamento no conjunto de relacionamento ternário FORNECE.

Relacionamentos como atributos. Às vezes, é conveniente pensar em um tipo de relacionamento binário em termos de atributos, conforme discutimos na Seção 7.3.3. Considere o tipo de relacionamento TRABALHA_PARA da Figura 7.9. Pode-se pensar em um atributo chamado Departamento do tipo de entidade FUNCIONARIO, em que o valor do Departamento para cada entidade FUNCIONARIO é a (uma referência à) entidade DEPARTAMENTO para a qual esse funcionário trabalha. Logo, o conjunto de valores para esse atributo Departamento é o conjunto de *todas* as entidades DEPARTAMENTO, que é o conjunto de entidades DEPARTAMENTO. Foi isso que fizemos na Figura 7.8, quando especificamos o projeto inicial do tipo de entidade FUNCIONARIO para o banco de dados EMPRESA. Porém, quando pensamos em um relacionamento binário como um atributo, sempre temos duas opções. Neste exemplo, a alternativa é pensar em um atributo multivalorado Funcionario do tipo de entidade DEPARTAMENTO, cujos valores para cada entidade DEPARTAMENTO é o conjunto de entidades de FUNCIONARIO que trabalham para esse departamento. O conjunto de valores desse atributo Funcionario é o conjunto de potência do conjunto de entidades FUNCIONARIO. Qualquer um desses dois atributos — Departamento de FUNCIONARIO ou Funcionario de DEPARTAMENTO — pode representar o tipo de relacionamento TRABALHA_PARA. Se os dois forem representados, eles são restringidos a serem o inverso um do outro.[8]

Nomes de função e relacionamentos recursivos. Cada tipo de entidade que participa de um tipo de relacionamento desempenha nele uma função em particular. O **nome da função** significa a função que uma entidade participante do tipo de entidade desempenha em cada instância de relacionamento, e ajuda a explicar o que o relacionamento significa. Por exemplo, no tipo de relacionamento TRABALHA_PARA, FUNCIONARIO desempenha a função de *funcionário* ou *trabalhador*, e DEPARTAMENTO desempenha a função de *departamento* ou *empregador*.

Os nomes de função não são tecnicamente necessários nos tipos de relacionamento em que todos os tipos de entidade participantes são distintos, pois o nome de cada tipo de entidade participante pode ser usado como nome de função. Contudo, em algumas ocasiões, o *mesmo* tipo de entidade participa mais de uma vez em um tipo de relacionamento em *funções diferentes*. Nesses casos, o nome da função torna-se essencial para distinguir o significado da função que cada entidade

[8] Esse conceito de representar os tipos de relacionamento como atributos é usado em uma classe de modelos de dados chamada **modelos de dados funcionais**. Nos bancos de dados de objeto (ver Capítulo 11), os relacionamentos podem ser representados por atributos de referência, seja em uma direção ou nas duas direções como inversos. Em bancos de dados relacionais (ver Capítulo 3), as chaves estrangeiras são um tipo de atributo de referência usado para representar relacionamentos.

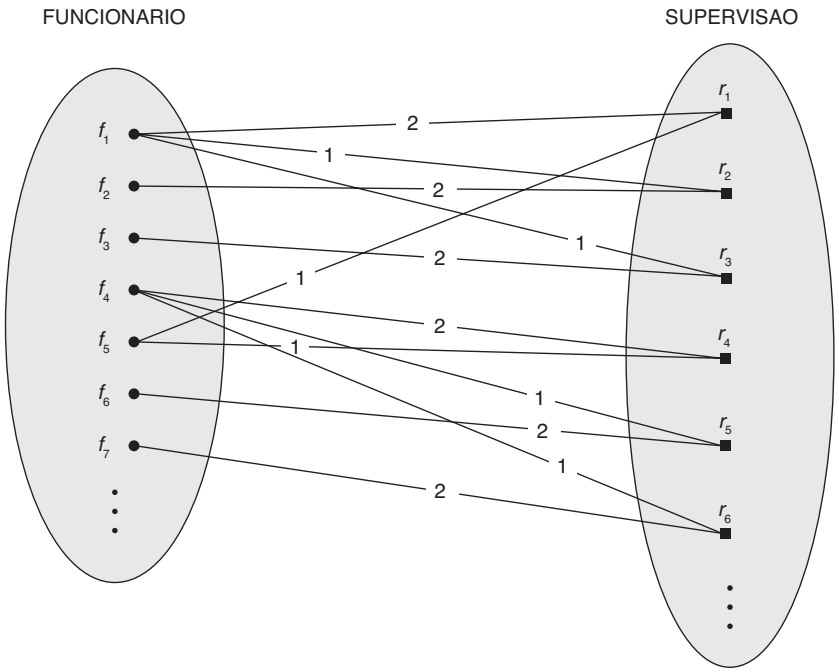

Figura 7.11
Um relacionamento recursivo SUPERVISAO entre FUNCIONARIO no papel de ***supervisor*** (1) e FUNCIONARIO no papel de ***subordinado*** (2).

[9] N significa ***qualquer número*** de entidades relacionadas (zero ou mais).

participante desempenha. Esses tipos de relacionamento são chamados de **relacionamentos recursivos**. A Figura 7.11 mostra um exemplo. O tipo de relacionamento SUPERVISAO relaciona um funcionário a um supervisor, no qual as entidades funcionário e supervisor são membros do mesmo conjunto entidade FUNCIONARIO. Logo, o tipo de entidade FUNCIONARIO *participa duas vezes* na SUPERVISAO: uma vez no papel de *supervisor* (ou *chefe*) e outra no papel de *supervisionado* (ou *subordinado*). Cada instância de relacionamento r_i na SUPERVISAO associa duas entidades de funcionário f_j e f_k, uma das quais desempenha a função de supervisor e a outra, a função de supervisionado. Na Figura 7.11, as linhas marcadas com '1' representam a função de supervisor, e aquelas marcadas com '2' representam a função de supervisionado; assim, f_1 supervisiona f_2 e f_3, f_4 supervisiona f_6 e f_7, e f_5 supervisiona f_1 e f_4. Neste exemplo, cada instância de relacionamento precisa ser conectada com duas linhas, uma marcada com '1' (supervisor) e a outra com '2' (supervisionado).

7.4.3 Restrições sobre tipos de relacionamento binários

Os tipos de relacionamento costumam ter certas restrições que limitam as combinações de entidades que podem participar no conjunto de relacionamentos correspondente. Essas restrições são determinadas com base na situação do minimundo que os relacionamentos representam. Por exemplo, na Figura 7.9, se a empresa tem uma regra de que cada funcionário precisa trabalhar para exatamente um departamento, então gostaríamos de descrever essa restrição no esquema. Podemos distinguir dois tipos principais de restrições de relacionamento binário: *razão de cardinalidade* e *participação*.

Razões de cardinalidade para relacionamentos binários. A **razão de cardinalidade** para um relacionamento binário especifica o número *máximo* de instâncias de relacionamento em que uma entidade pode participar. Por exemplo, no tipo de relacionamento binário TRABALHA_PARA, DEPARTAMENTO:FUNCIONARIO tem razão de cardinalidade 1:N, significando que cada departamento pode estar relacionado a (ou seja, emprega) qualquer número de funcionários,[9] mas um funcionário só pode estar relacionado a (trabalha para) um departamento. Isso significa que, para esse relacionamento TRABALHA_PARA em particular, uma entidade de departamento em particular pode estar relacionada a qualquer número de funcionários (N indica que não existe um número máximo). Por sua vez, um funcionário pode estar relacionado no máximo a um único departamento. As razões de cardinalidade possíveis para tipos de relacionamento binários são 1:1, 1:N, N:1 e M:N.

Um exemplo de relacionamento binário 1:1 é GERENCIA (Figura 7.12), o qual relaciona uma entidade de departamento ao funcionário que gerencia esse departamento. Isso representa as restrições do minimundo que — em qualquer ponto no tempo — um funcionário pode gerenciar apenas um departamento e um departamento pode ter apenas um gerente. O tipo de relacionamento TRABALHA_EM (Figura 7.13) é de razão de cardinalidade M:N, porque a regra do minimundo é a de que um funcionário pode trabalhar em vários projetos e um projeto pode ter vários funcionários.

As razões de cardinalidade para relacionamentos binários são representadas nos diagramas ER exibindo 1, M e N nos losangos, como mostra a Figura 7.2. Observe que, nessa notação, podemos especificar nenhum máximo (N) ou um máximo de um (1) na participação. Uma notação alternativa (ver Seção 7.7.4) permite que o projetista especifique um *número máximo* na participação, como 4 ou 5.

Restrições de participação e dependências de existência. A **restrição de participação** especifica se a existência de uma entidade depende dela estar relacionada a outra entidade por meio do tipo de relacionamento. Essa restrição especifica o número *mínimo* de instâncias de relacionamento em que cada entidade pode participar, e às vezes é chamada de **restrição de cardinalidade mínima**. Existem dois tipos de restrições de participação — total e parcial — que vamos ilustrar. Se a política de uma empresa afirma que *todo* funcionário precisa trabalhar para um departamento, então uma entidade de funcionário só pode existir se participar em, pelo menos, uma instância de relacionamento TRABALHA_PARA (Figura 7.9). Assim, a participação de FUNCIONARIO em TRABALHA_PARA é chamada de **participação total**, significando que cada entidade *no conjunto total* de entidades de funcionário deve estar relacionada a uma entidade de departamento por meio de TRABALHA_PARA. A participação total também é conhecida como **dependência de existência**. Na Figura 7.12, não esperamos que cada funcionário gerencie um departamento, de modo que a participação de FUNCIONARIO no tipo de relacionamento GERENCIA é **parcial**, significando que *uma parte do conjunto de* entidades de funcionário está relacionada a alguma entidade de departamento por meio de GERENCIA, mas não necessariamente todas. Vamos nos referir à razão de cardinalidade e restrições de participação, juntas, como as **restrições estruturais** de um tipo de relacionamento.

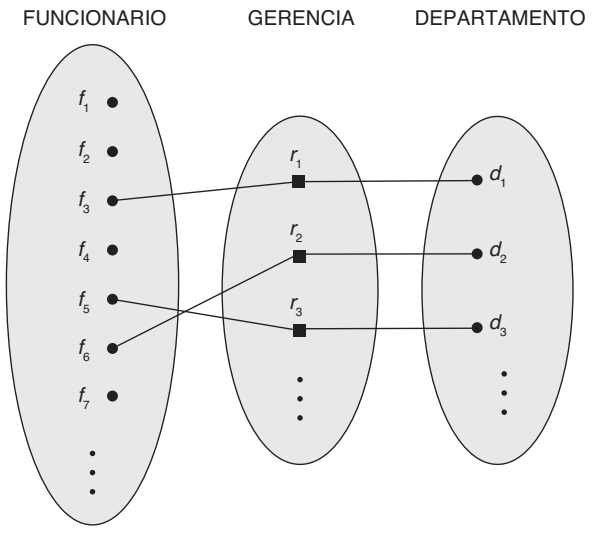

Figura 7.12
Um relacionamento 1:1, GERENCIA.

Em diagramas ER, a participação total (ou dependência de existência) é exibida como uma *linha dupla* que conecta o tipo de entidade participante ao relacionamento, enquanto a participação parcial é representada por uma *linha simples* (ver Figura 7.2). Observe que, nessa notação, podemos especificar nenhum mínimo (participação parcial) ou um mínimo de um (participação total). A notação alternativa (ver Seção 7.7.4) permite que o projetista indique um *número mínimo* específico da participação no relacionamento, como 4 ou 5.

Discutiremos as restrições sobre os relacionamentos de grau mais alto na Seção 7.9.

7.4.4 Atributos de tipos de relacionamento

Os tipos de relacionamento também podem ter atributos, semelhantes àqueles dos tipos de entidade. Por exemplo, para registrar o número de horas por semana que um funcionário trabalha em determinado projeto, podemos incluir um atributo Horas para o tipo de relacionamento TRABALHA_EM na Figura 7.13. Outro exemplo é incluir a data em que um gerente começou a chefiar um departamento por meio de um atributo Data_inicio para o tipo de relacionamento GERENCIA na Figura 7.12.

Observe que os atributos dos tipos de relacionamento 1:1 ou 1:N podem ser migrados para um dos tipos de entidade participantes. Por exemplo, o atributo Data_inicio para o relacionamento GERENCIA pode ser um atributo de FUNCIONARIO ou de DEPARTAMENTO, embora conceitualmente ele pertença a GERENCIA. Isso porque GERENCIA é um relacionamento 1:1, de modo que cada entidade de departamento ou funcionário participa de *no máximo uma* instância de relacionamento. Logo, o valor do atributo Data_inicio pode ser determinado separadamente,

pela entidade do departamento participante ou pela entidade do funcionário participante (gerente).

Para um tipo de relacionamento 1:N, um atributo de relacionamento pode ser migrado *somente* para o tipo de entidade no lado N do relacionamento. Por exemplo, na Figura 7.9, se o relacionamento TRABALHA_PARA também tiver um atributo Data_inicio que indica quando um funcionário começou a trabalhar para um departamento, esse atributo pode ser incluído como um atributo de FUNCIONARIO. Isso porque cada funcionário trabalha para somente um departamento, e por isso participa de, no máximo, uma instância de relacionamento em TRABALHA_PARA. Nos tipos de relacionamento 1:1 e 1:N, a decisão de onde colocar um atributo de relacionamento — como um atributo de tipo de relacionamento ou como um atributo de um tipo de entidade participante — é determinada de maneira subjetiva pelo projetista do esquema.

Para tipos de relacionamento M:N, alguns atributos podem ser determinados pela *combinação de entidades participantes* em uma instância de relacionamento, e não por qualquer entidade isolada. Esses atributos *precisam ser especificados como atributos de relacionamento*. Um exemplo é o atributo Horas do relacionamento M:N de TRABALHA_EM (Figura 7.13). O número de horas por semana que um funcionário trabalha atualmente em um projeto é determinado por uma combinação funcionário-projeto, e não de maneira separada por qualquer entidade.

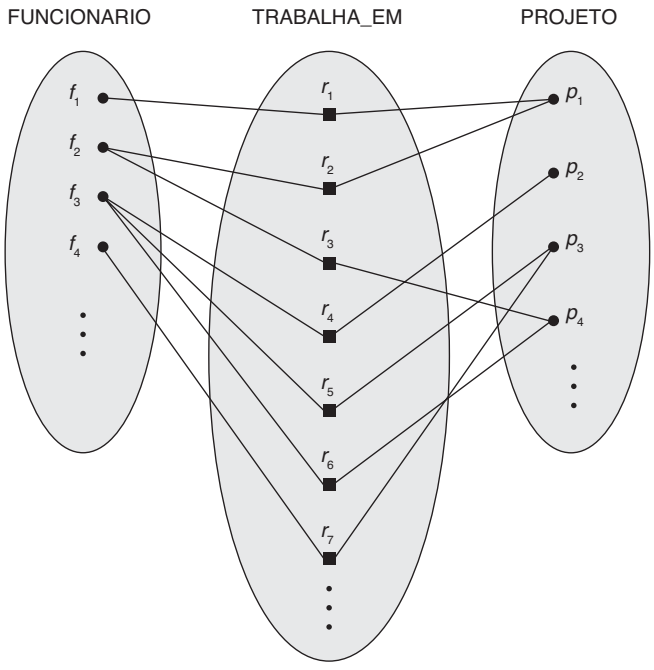

Figura 7.13
Um relacionamento M:N, TRABALHA_EM.

7.5 Tipos de entidade fraca

Tipos de entidade que não possuem atributos-chave próprios são chamados **tipos de entidade fraca.** Ao contrário, os **tipos de entidade regulares** que não têm um atributo-chave — o que inclui todos os exemplos discutidos até aqui — são chamados **tipos de entidade fortes**. As entidades pertencentes a um tipo de entidade fraca são identificadas por estarem relacionadas a entidades específicas de outro tipo em combinação com um de seus valores de atributo. Chamamos esse outro tipo de entidade de **tipo de entidade de identificação** ou **proprietário**,[10] e chamamos o tipo de relacionamento que relaciona um tipo de entidade fraca a seu proprietário de **relacionamento de identificação** do tipo de entidade fraca.[11] Um tipo de entidade fraca sempre tem uma *restrição de participação total* (dependência de existência) com relação a seu relacionamento de identificação, porque a entidade fraca não pode ser identificada sem uma entidade proprietária. Porém, nem toda dependência de existência resulta em um tipo de entidade fraca. Por exemplo, uma entidade CARTEIRA_MOTORISTA não pode existir a menos que esteja relacionada a uma entidade PESSOA, embora tenha a própria chave (Numero_habilitacao) e, portanto, não seja uma entidade fraca.

Considere o tipo de entidade DEPENDENTE, relacionado a FUNCIONARIO, que é usado para registrar os dependentes de cada funcionário por meio de um relacionamento 1:N (Figura 7.2). Em nosso exemplo, os atributos de DEPENDENTE são Nome (o primeiro nome do dependente), Data_nascimento, Sexo e Parentesco (com o funcionário). Dois dependentes de *dois funcionários distintos* podem, por coincidência, ter os mesmos valores para Nome, Data_nascimento, Sexo e Parentesco, mas, ainda assim, eles são entidades distintas. Eles são identificados como entidades distintas apenas depois de determinar a *entidade de funcionário em particular* à qual cada dependente está relacionado. Considera-se que cada entidade de funcionário *possui* as entidades dependentes que estão relacionadas a ele.

Um tipo de entidade fraca normalmente tem uma **chave parcial**, que é o atributo que pode iden-

[10] O tipo de entidade de identificação é também chamado de **tipo de entidade pai**, ou **tipo de entidade dominante**.

[11] O tipo de entidade fraca também é chamado de **tipo de entidade filho**, ou **tipo de entidade subordinado**.

tificar exclusivamente as entidades fracas que estão *relacionadas à mesma entidade proprietária*.[12] Em nosso exemplo, se considerarmos que dois dependentes do mesmo funcionário não poderão ter o mesmo nome, o atributo Nome de DEPENDENTE é a chave parcial. No pior dos casos, um atributo composto de *todos os atributos da entidade fraca* será a chave parcial.

Em diagramas ER, tanto um tipo de entidade fraca quanto seu relacionamento de identificação são distinguidos ao delimitar suas caixas e losangos com linhas duplas (ver Figura 7.2). O atributo de chave parcial é sublinhado com uma linha tracejada ou pontilhada.

Os tipos de entidade fraca às vezes podem ser representados como atributos complexos (compostos, multivalorados). No exemplo anterior, poderíamos especificar um atributo multivalorado Dependentes para FUNCIONARIO, que é um atributo composto com os atributos componentes Nome, Data_nascimento, Sexo e Parentesco. A escolha de qual representação usar é feita pelo projetista de banco de dados. Um critério que pode ser usado é escolher a representação do tipo de entidade fraca se houver muitos atributos. Se a entidade fraca participar independentemente nos tipos de relacionamento além de seu tipo de relacionamento de identificação, então ela não deverá ser modelada como um atributo complexo.

Em geral, podemos definir qualquer quantidade de níveis de tipos de entidade fraca; um tipo de entidade proprietário pode ele mesmo ser um tipo de entidade fraca. Além disso, um tipo de entidade fraca pode ter mais de um tipo de entidade de identificação e um tipo de relacionamento de identificação de grau maior que dois, conforme ilustraremos na Seção 7.9.

7.6 Refinando o projeto ER para o banco de dados EMPRESA

Agora, podemos refinar o projeto de banco de dados da Figura 7.8 alterando os atributos que representam relacionamentos para tipos de relacionamento. A razão de cardinalidade e a restrição de participação de cada tipo de relacionamento são determinadas com base nos requisitos listados na Seção 7.2. Se alguma delas não puder ser especificada dessa maneira, os usuários terão de ser questionados ainda mais para determinar essas restrições estruturais.

Em nosso exemplo, especificamos os seguintes tipos de relacionamento:

- GERENCIA, um tipo de relacionamento 1:1 entre FUNCIONARIO e DEPARTAMENTO. A participação de FUNCIONARIO é parcial. A participação de DEPARTAMENTO não é clara pelos requisitos. Questionamos os usuários, que dizem que um departamento precisa ter um gerente o tempo todo, o que implica participação total.[13] O atributo Data_inicio é atribuído a esse tipo de relacionamento.
- TRABALHA_PARA, um tipo de relacionamento 1:N entre DEPARTAMENTO e FUNCIONARIO. As duas participações são totais.
- CONTROLA, um tipo de relacionamento 1:N entre DEPARTAMENTO e PROJETO. A participação de PROJETO é total, enquanto a de DEPARTAMENTO é determinada para ser parcial, depois que os usuários indicaram que alguns departamentos podem não controlar projeto algum.
- SUPERVISAO, um tipo de relacionamento 1:N entre FUNCIONARIO (no papel de supervisor) e FUNCIONARIO (no papel de supervisionado). As duas participações são determinadas como sendo parciais, depois que os usuários indicaram que nem todo funcionário é um supervisor e nem todo funcionário tem um supervisor.
- TRABALHA_EM, determinado como sendo um tipo de relacionamento M:N com atributo Horas, depois que os usuários indicaram que um projeto pode ter vários funcionários trabalhando nele. As duas participações são determinadas como totais.
- DEPENDENTES_DE, um tipo de relacionamento 1:N entre FUNCIONARIO e DEPENDENTE, que também é o relacionamento de identificação para o tipo de entidade fraca DEPENDENTE. A participação de FUNCIONARIO é parcial, enquanto a de DEPENDENTE é total.

Depois de especificar os seis tipos de relacionamento citados, removemos dos tipos de entidade da Figura 7.8 todos os atributos que foram refinados para relacionamentos. Estes incluem Gerente e Data_inicio_gerente de DEPARTAMENTO; Departamento_gerenciador de PROJETO; Departamento, Supervisor e Trabalha_em de FUNCIONARIO; e Funcionario de DEPENDENTE. É importante ter o mínimo possível de redundância quando projetamos o esquema concei-

[12] A chave parcial às vezes é chamada de **discriminadora**.

[13] As regras no minimundo que determinam as restrições também são chamadas de ***regras de negócio***, pois elas são determinadas pelo ***negócio*** ou pela organização que utilizará o banco de dados.

tual de um banco de dados. Se alguma redundância for desejada no nível de armazenamento ou no nível de visão do usuário, ela pode ser introduzida mais tarde, conforme discutimos na Seção 1.6.1.

7.7 Diagramas ER, convenções de nomes e questões de projeto

7.7.1 Resumo da notação para diagramas ER

As figuras 7.9 a 7.13 ilustram exemplos da participação dos tipos de entidade nos tipos de relacionamento ao exibir seus conjuntos ou extensões — as instâncias de entidade individuais em um conjunto de entidades e as instâncias de relacionamento individuais em um conjunto de relacionamentos. Nos diagramas ER, a ênfase está em representar os esquemas em vez das instâncias. Isso é mais útil no projeto de banco de dados porque um esquema de banco de dados muda raramente, enquanto o conteúdo dos conjuntos de entidades muda com frequência. Além disso, o esquema é obviamente mais fácil de exibir, pois é muito menor.

A Figura 7.2 exibe o **esquema de banco de dados** EMPRESA como um **diagrama ER**. Agora, revisamos a notação completa do diagrama ER. Tipos de entidade como FUNCIONARIO, DEPARTAMENTO e PROJETO são mostrados nas caixas retangulares. Tipos de relacionamento como TRABALHA_PARA, GERENCIA, CONTROLA e TRABALHA_EM são mostrados em caixas em forma de losango, conectadas aos tipos de entidade participantes com linhas retas. Os atributos são mostrados em ovais, e cada atributo é conectado por uma linha reta a seu tipo de entidade ou tipo de relacionamento. Os atributos componentes de um atributo composto são conectados à oval que representa o atributo composto, conforme ilustrado pelo atributo Nome de FUNCIONARIO. Os atributos multivalorados aparecem em ovais duplas, conforme ilustrado pelo atributo Localizacoes de DEPARTAMENTO. Os atributos-chave têm seus nomes sublinhados. Os atributos derivados aparecem em ovais pontilhadas, conforme ilustrado pelo atributo Numero_funcionarios de DEPARTAMENTO.

Os tipos de entidade fraca são distinguidos ao serem colocados em retângulos duplos e terem seu relacionamento de identificação colocado em losangos duplos, conforme ilustrado pelo tipo de entidade DEPENDENTE e DEPENDENTES_DE identificando o tipo de relacionamento. A chave parcial do tipo de entidade fraca é sublinhada com uma linha tracejada.

Na Figura 7.2, a razão de cardinalidade de cada tipo de relacionamento *binário* é especificada pela conecção de um 1, M ou N em cada aresta participante. A razão de cardinalidade de DEPARTAMENTO:FUNCIONARIO em GERENCIA é 1:1, enquanto é 1:N para DEPARTAMENTO: FUNCIONARIO em TRABALHA_PARA, e M:N para TRABALHA_EM. A restrição de participação é especificada por uma linha simples para participação parcial e por linhas duplas para a participação total (dependência de existência).

Na Figura 7.2, mostramos os nomes de papel para o tipo de relacionamento SUPERVISAO, pois o mesmo tipo de entidade FUNCIONARIO desempenha dois papéis distintos nesse relacionamento. Observe que a razão de cardinalidade é 1:N de supervisor para supervisionado porque cada funcionário no papel de supervisionado tem, no máximo, um supervisor direto, ao passo que um funcionário no papel de supervisor pode controlar zero ou mais funcionários.

A Figura 7.14 resume as convenções para diagramas ER. É importante observar que existem muitas outras notações diagramáticas alternativas (ver Seção 7.7.4 e Apêndice A).

7.7.2 Nomeação apropriada de construções de esquema

Ao projetar um esquema de banco de dados, a escolha de nomes para tipos de entidade, atributos, tipos de relacionamento e (particularmente) funções nem sempre é simples. É preciso escolher nomes que transmitam, tanto quanto possível, os significados conectados às diferentes construções no esquema. Escolhemos usar *nomes no singular* para os tipos de entidade, em vez de nomes no plural, porque o nome se aplica a cada entidade individual pertencente a esse tipo de entidade. Em nossos diagramas ER, usaremos a convenção de que os nomes do tipo de entidade e tipo de relacionamento são escritos em letras maiúsculas, os nomes de atributo têm apenas a letra inicial em maiúscula e os nomes de papel são escritos em letras minúsculas. Usamos essa convenção na Figura 7.2.

Como uma prática geral, dada uma descrição narrativa dos requisitos do banco de dados, os *nomes* que aparecem na narrativa tendem a gerar nomes de tipo de entidade, e os *verbos* tendem a indicar nomes de tipos de relacionamento. Os nomes de atributo costumam surgir de nomes adicionais que descrevem os nomes correspondentes para tipos de entidade.

Outra consideração de nomeação envolve a escolha de nomes de relacionamento binário para tornar o diagrama ER do esquema legível da esquerda para a direita e de cima para baixo. Seguimos essa orientação de modo geral na Figura 7.2. Para explicar essa convenção de nomeação ainda mais, temos uma exceção para a Figura 7.2 — o tipo de relaciona-

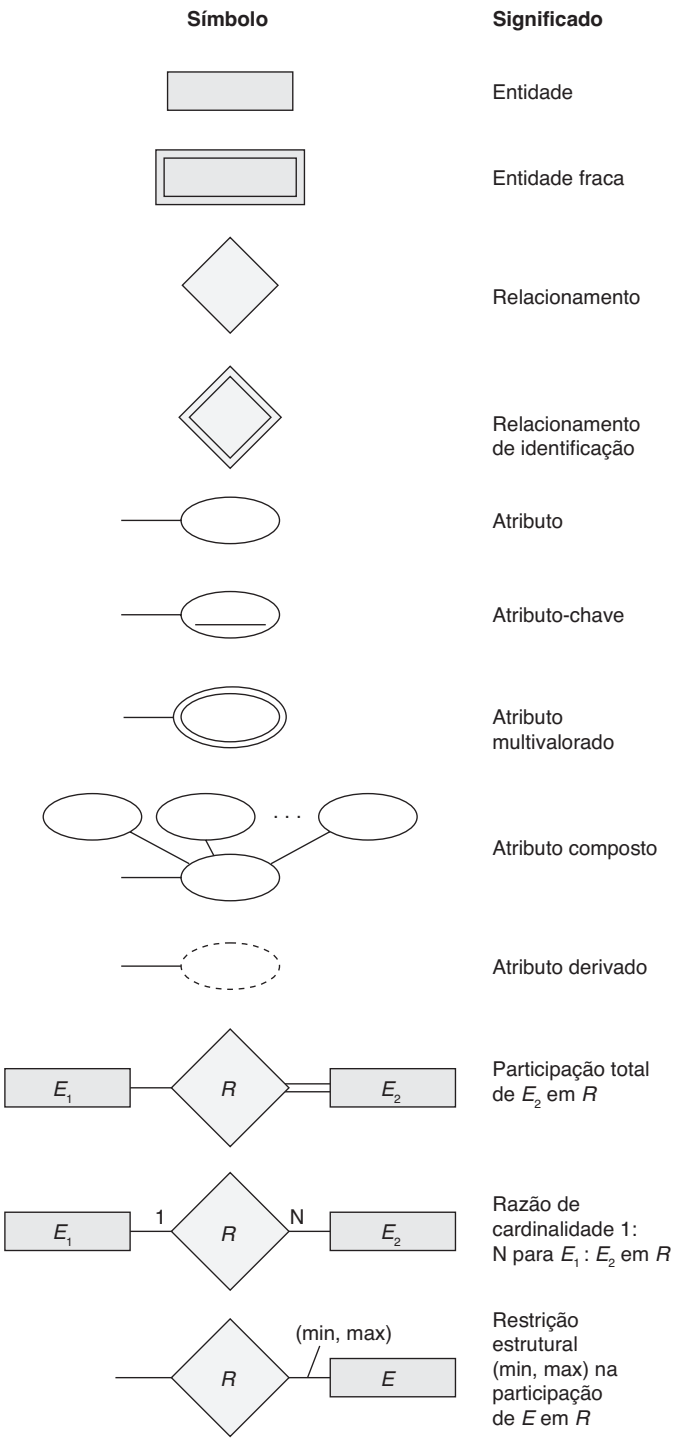

Figura 7.14
Resumo da notação para diagramas ER.

mento DEPENDENTES_DE, lido de baixo para cima. Quando descrevemos esse relacionamento, podemos dizer que as entidades DEPENDENTE (tipo de entidade inferior) são DEPENDENTES_DE (nome de relacionamento) um FUNCIONARIO (tipo de entidade superior). Para mudar isso e ler de cima para baixo, poderíamos renomear o tipo de relacionamento para POSSUI_DEPENDENTES, que então seria lido da seguinte forma: uma entidade FUNCIONARIO (tipo de entidade superior) POSSUI_DEPENDENTES (nome de relacionamento) do tipo DEPENDENTE (tipo de entidade inferior). Observe que esse problema surge porque cada relacionamento binário pode ser descrito começando de qualquer um dos dois tipos de entidades participantes, conforme discutido no início da Seção 7.4.

7.7.3 Escolhas de projeto para o projeto conceitual ER

Às vezes, é difícil decidir se um conceito em particular no minimundo deve ser modelado como um tipo de entidade, um atributo ou um tipo de relacionamento. Nesta seção, oferecemos algumas orientações rápidas sobre qual construção deve ser escolhida em situações específicas.

Em geral, o processo de projeto de esquema deve ser considerado um processo de refinamento iterativo, no qual um projeto inicial é criado e depois refinado iterativamente até que seja alcançado o mais adequado. Alguns dos refinamentos frequentemente utilizados incluem o seguinte:

- Um conceito pode ser modelado como um atributo e depois refinado para um relacionamento, pois é determinado que o atributo é uma referência para outro tipo de entidade. Com frequência acontece de um par desses atributos, que são inversos um do outro, ser refinado em um relacionamento binário. Discutimos esse tipo de refinamento com detalhes na Seção 7.6. É importante observar que, em nossa notação, quando um atributo é substituído por um relacionamento, o próprio atributo deve ser removido do tipo de entidade para evitar duplicação e redundância.
- De modo semelhante, um atributo que existe em vários tipos de entidade pode ser elevado ou promovido para um tipo de entidade independente. Por exemplo, suponha que cada tipo de entidade em um banco de dados UNIVERSIDADE, como ALUNO, PROFESSOR e DISCIPLINA, tenha um atributo Departamento no projeto inicial. O projetista pode então escolher criar um tipo de entidade DEPARTAMENTO com um único atributo Dept_nome e relacioná-lo aos três tipos de entidade (ALUNO, PROFESSOR e DISCIPLINA) por meio de relacionamentos apropriados. Outros atributos/relacionamentos de DEPARTAMENTO podem ser descobertos mais tarde.

- Um refinamento inverso para o caso anterior pode ser aplicado — por exemplo, se um tipo de entidade DEPARTAMENTO existir no projeto inicial com um atributo isolado Dept_nome e estiver relacionado a somente outro tipo de entidade, ALUNO. Nesse caso, DEPARTAMENTO pode ser reduzido ou rebaixado para um atributo de ALUNO.
- A Seção 7.9 vai discutir as escolhas referentes ao grau de um relacionamento. No Capítulo 8, discutiremos outros refinamentos referentes à especialização/generalização. O Capítulo 10 abordará refinamentos top-down e bottom-up adicionais que são comuns no projeto de esquema conceitual em grande escala.

7.7.4 Notações alternativas para diagramas ER

Existem muitas notações diagramáticas alternativas para exibir diagramas ER. O Apêndice A mostra algumas das mais populares. Na Seção 7.8, vamos introduzir a notação Unified Modeling Language (UML) para diagramas de classe, que foi proposta como um padrão para a modelagem conceitual de objeto.

Nesta seção, descrevemos uma notação ER alternativa para especificar restrições estruturais sobre os relacionamentos, que substitui a razão de cardinalidade (1:1, 1:N, M:N) e a notação de linha simples/dupla para as restrições de participação. Essa notação envolve associar um par de números inteiros (min, max) a cada *participação* de um tipo de entidade E em um tipo de relacionamento R, onde $0 \leq \text{min} \leq \text{max}$ e $\text{max} \geq 1$. Os números significam que, para cada entidade e em E, e precisa participar de pelo menos min e no máximo max instâncias de relacionamento em R *em qualquer ponto no tempo*. Nesse método, min = 0 implica participação parcial, enquanto $\text{min} > 0$ implica participação total.

A Figura 7.15 mostra o esquema de banco de dados EMPRESA usando a notação (min, max).[14] Em geral, usa-se ou a notação de razão de cardinalidade/linha, simples/linha dupla *ou* a notação (min, max). A notação (min, max) é mais precisa, e podemos usá-la para especificar algumas restrições estruturais para os tipos de relacionamento de *maior grau*. Porém, isso não é suficiente para especificar algumas restrições de chave nos relacionamentos de maior grau, conforme discutiremos na Seção 7.9.

A Figura 7.15 também apresenta todos os nomes de papel para o esquema de banco de dados EMPRESA.

7.8 Exemplo de outra notação: diagramas de classes UML

A metodologia UML está sendo bastante utilizada no projeto de software e tem muitos tipos de diagramas para diversas finalidades do projeto de software. Aqui, apresentamos rapidamente os fundamentos dos **diagramas de classes UML** e os comparamos com os diagramas ER. De algumas maneiras, os diagramas de classes podem ser considerados uma notação alternativa aos diagramas ER. A notação e os conceitos adicionais da UML serão apresentados na Seção 8.6 e no Capítulo 10. A Figura 7.16 mostra como o esquema de banco de dados ER EMPRESA da Figura 7.15 pode ser exibido usando a notação de diagrama de classes UML. Os *tipos de entidade* na Figura 7.15 são modelados como *classes* na Figura 7.16. Uma *entidade* em ER corresponde a um *objeto* em UML.

Nos diagramas de classes UML, uma **classe** (semelhante a um tipo de entidade em ER) é exibida como uma caixa (ver Figura 7.16) que inclui três seções: a seção superior mostra o **nome da classe** (semelhante ao nome do tipo de entidade); a seção do meio inclui os **atributos**; e a última seção inclui as **operações** que podem ser aplicadas aos objetos individuais (semelhante às entidades individuais em um conjunto de entidades) da classe. As operações *não são* especificadas em diagramas ER. Considere a classe FUNCIONARIO na Figura 7.16. Seus atributos são Nome, Cpf, Datanasc, Sexo, Endereco e Salario. O projetista pode, opcionalmente, especificar o **domínio** de um atributo, se desejar, colocando um sinal de dois-pontos (:) seguido pelo nome ou descrição do domínio, conforme ilustrado pelos atributos Nome, Sexo e Datanasc de FUNCIONARIO na Figura 7.16. Um atributo composto é modelado como um **domínio estruturado**, conforme ilustrado pelo atributo Nome de FUNCIONARIO. Um atributo multivalorado geralmente será modelado como uma classe separada, conforme ilustrado pela classe LOCALIZACAO na Figura 7.16.

Os tipos de relacionamento são chamados de **associações** em terminologia UML, e as instâncias de relacionamento são chamadas de **ligações**. Uma **associação binária** (tipo de relacionamento binário) é representada como uma linha que conecta as classes participantes (tipos de entidade) e pode, de maneira opcional, ter um nome. Um atributo de relacionamento, chamado **atributo de ligação**, é colocado em uma caixa que está conectada à linha da associa-

[14] Em algumas notações, particularmente aquelas usadas nas metodologias de modelagem de objeto, como UML, o (min, max) é colocado nos *lados opostos* aos que mostramos. Por exemplo, para o relacionamento TRABALHA_PARA da Figura 7.15, o (1,1) estaria no lado DEPARTAMENTO, e o (4,N) estaria no lado FUNCIONARIO. Aqui, usamos a notação original de Abrial (1974).

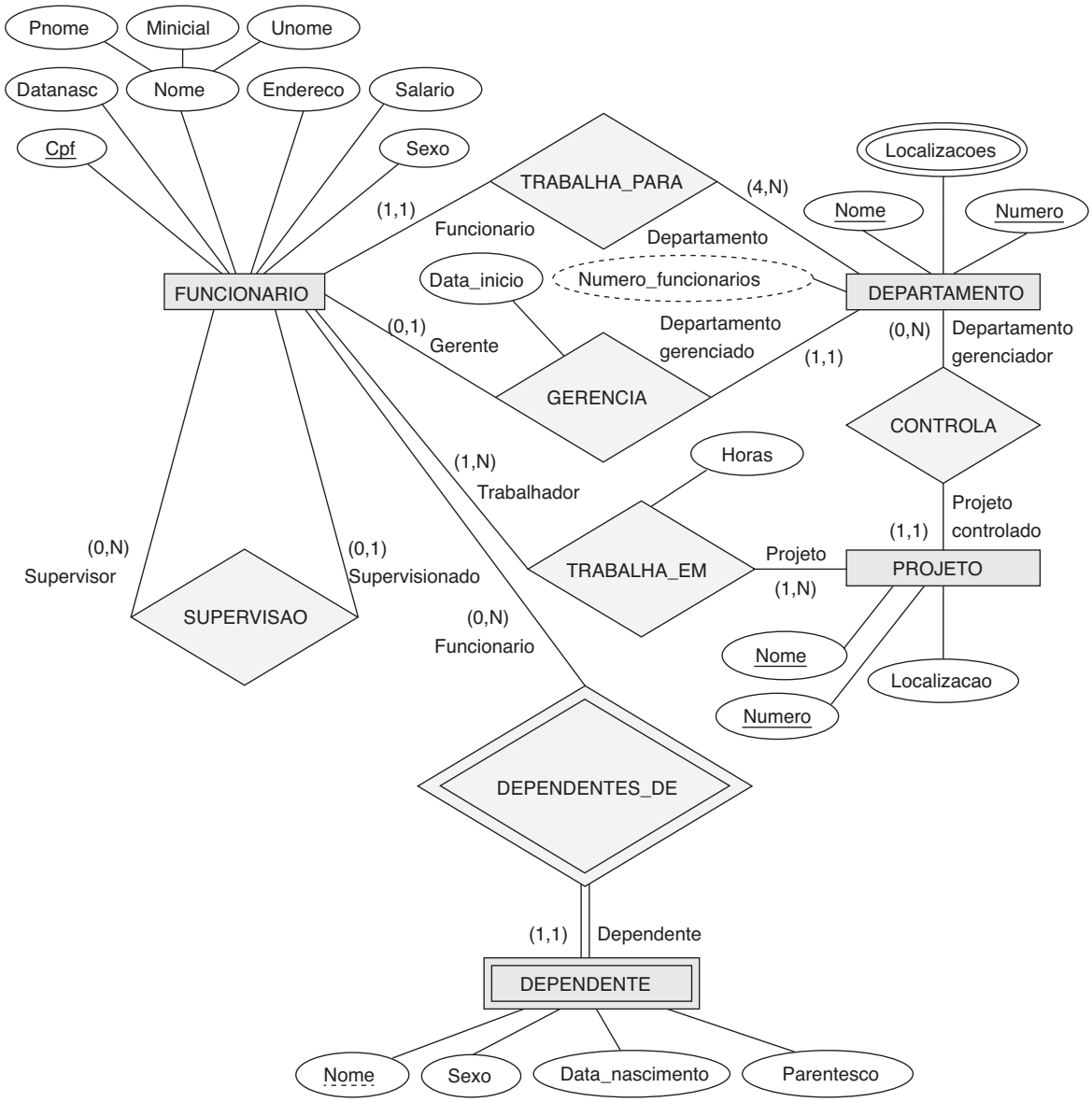

Figura 7.15
Diagramas ER para o esquema EMPRESA, com restrições estruturais especificadas usando a notação (min, max) e nomes de função.

ção por uma linha tracejada. A notação (min, max) descrita na Seção 7.7.4 é usada para especificar restrições de relacionamento, que são chamadas **multiplicidades** em terminologia UML. As multiplicidades são especificadas na forma *min..max*, e um asterisco (*) indica nenhum limite máximo na participação. Contudo, as multiplicidades são colocadas *nos lados opostos do relacionamento* quando comparadas com a notação discutida na Seção 7.7.4 (compare as figuras 7.15 e 7.16). Em UML, um único asterisco indica uma multiplicidade de 0..*, e um único 1 indica uma multiplicidade de 1..1. Um relacionamento recursivo (ver Seção 7.4.2) é chamado de **associação reflexiva** em UML, e os nomes de função — como as multiplicidades — são colocados nos cantos opostos de uma associação quando comparados com o posicionamento dos nomes de função na Figura 7.15.

Em UML, existem dois tipos de relacionamentos: associação e agregação. A **agregação** serve para representar um relacionamento entre um objeto inteiro e suas partes componentes, e possui uma notação diagramática distinta. Na Figura 7.16, modelamos os locais de um departamento e o local isolado de um projeto como agregações. Porém, agregação

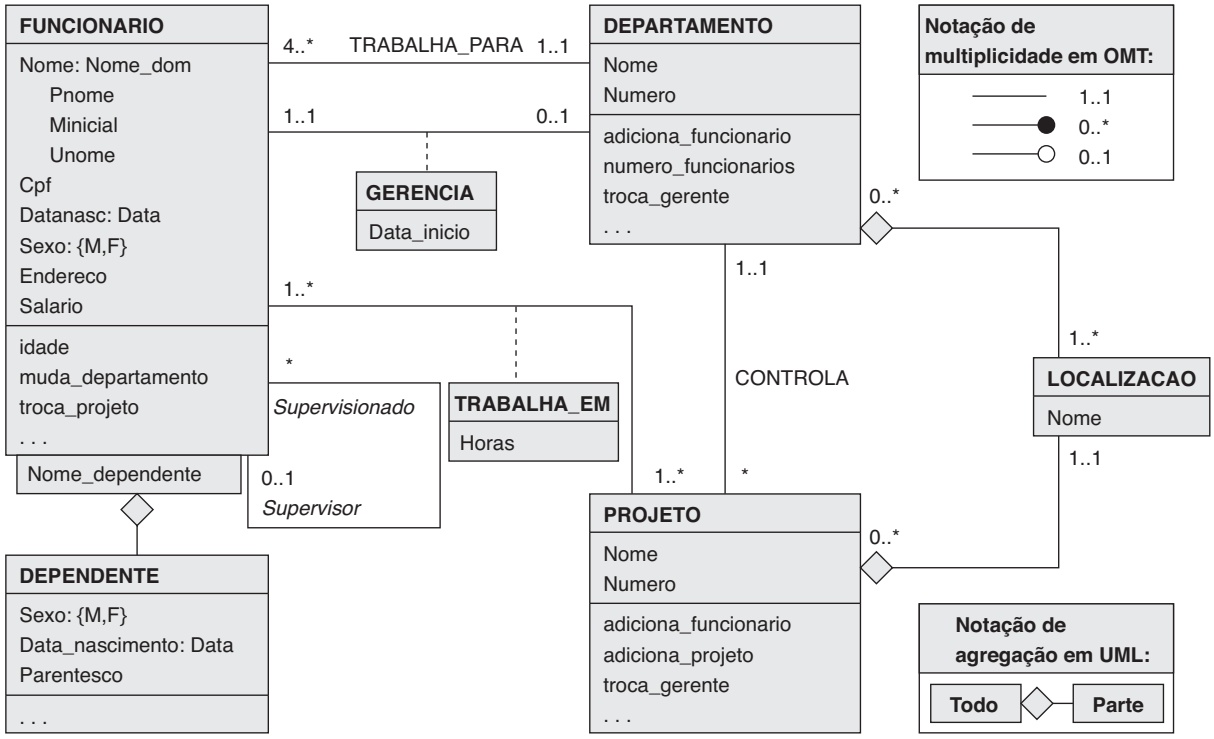

Figura 7.16
O esquema conceitual EMPRESA na notação do diagrama de classes UML.

e associação não possuem propriedades estruturais diferentes, e a escolha quanto a qual tipo de relacionamento usar é um tanto subjetiva. No modelo ER, ambas são representadas como relacionamentos.

A UML também distingue entre associações (ou agregações) **unidirecionais** e **bidirecionais**. No caso unidirecional, a linha que conecta as classes é exibida com uma seta para indicar que apenas uma direção para acessar objetos relacionados é necessária. Se nenhuma seta for exibida, o caso bidirecional é assumido, que é o padrão. Por exemplo, se sempre esperamos acessar o gerente de um departamento começando por um objeto DEPARTAMENTO, podemos desenhar a linha de associação representando a associação GERENCIA com uma seta de DEPARTAMENTO para FUNCIONARIO. Além disso, as instâncias de relacionamento podem ser especificadas para serem **ordenadas**. Por exemplo, poderíamos especificar que os objetos do funcionário relacionados a cada departamento por meio da associação (relacionamento) TRABALHA_PARA devem ser ordenados por seu valor de atributo Salario. Os nomes de associação (relacionamento) são *opcionais* em UML, e os atributos do relacionamento são exibidos em uma caixa conectada com uma linha tracejada à linha que representa a associação/agregação (ver Data_inicio e Horas na Figura 7.16).

As operações dadas em cada classe são derivadas dos requisitos funcionais da aplicação, conforme discutimos na Seção 7.1. Em geral, basta especificar os nomes de operação inicialmente para as operações lógicas que deverão ser aplicadas a objetos individuais de uma classe, conforme mostra a Figura 7.16. À medida que o projeto é refinado, mais detalhes são acrescentados, como os tipos de argumento (parâmetros) exatos para cada operação, mais uma descrição funcional de cada operação. A UML tem *descrições de função* e *diagramas de sequência* para especificar alguns dos detalhes da operação, mas estes estão fora do escopo de nossa discussão. O Capítulo 10 apresentará alguns desses diagramas.

Entidades fracas podem ser modeladas usando a construção chamada de **associação qualificada** (ou **agregação qualificada**) em UML. Esta pode representar tanto o relacionamento de identificação quanto a chave parcial, que é colocada em uma caixa ligada à classe proprietária. Isso é ilustrado pela classe DEPENDENTE e sua agregação qualificada a FUNCIONARIO na Figura 7.16. A chave parcial Nome_dependente é chamada de **discriminador** em terminologia UML, pois seu valor distingue os objetos associados (relacionados) ao mesmo FUNCIONARIO. As associações qualificadas não são restritas à modelagem de

entidades fracas e podem ser usadas para modelar outras situações em UML.

Esta seção não pretende oferecer uma descrição completa dos diagramas de classe UML, mas sim ilustrar um tipo popular de notação diagramática alternativa que pode ser utilizada para representar os conceitos de modelagem ER.

7.9 Tipos de relacionamento de grau maior que dois

Na Seção 7.4.2, definimos o **grau** de um tipo de relacionamento como o número de tipos de entidade participantes e chamamos um tipo de relacionamento de grau dois de *binário*, e de grau três de *ternário*. Nesta seção, explicamos melhor as diferenças entre os relacionamentos binário e de grau maior, quando escolher relacionamentos de grau maior *versus* binário, além de como especificar as restrições sobre relacionamentos de grau maior.

7.9.1 Escolhendo entre relacionamentos binário e ternário (ou de grau maior)

A notação de diagrama ER para um tipo de relacionamento ternário aparece na Figura 7.17(a), a qual mostra o esquema para o tipo de relacionamento FORNECE que foi mostrado no nível de conjunto de entidades/conjunto de relacionamentos ou de instância na Figura 7.10. Lembre-se de que o conjunto de relacionamentos de FORNECE é um conjunto de instâncias

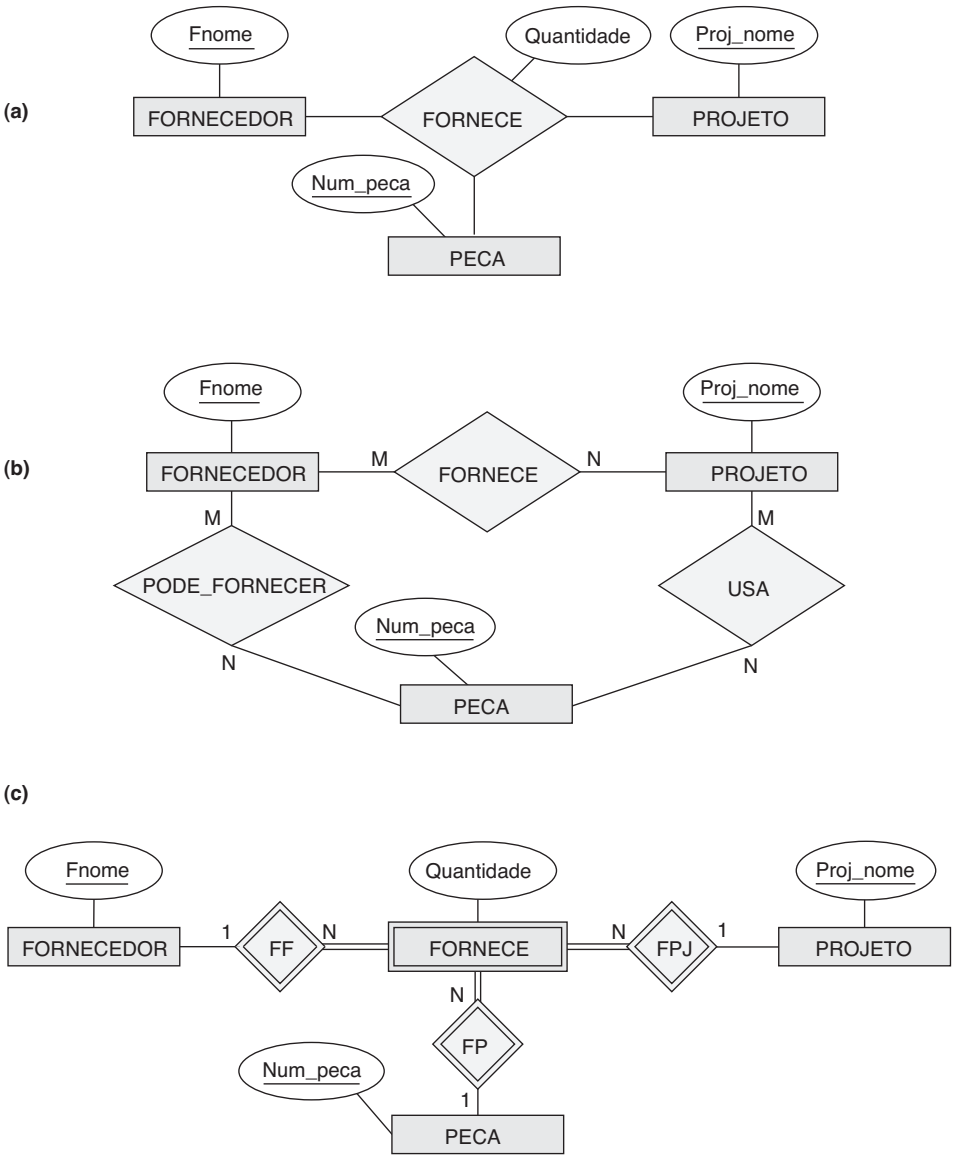

Figura 7.17
Tipos de relacionamento ternário. (a) O relacionamento FORNECE. (b) Três relacionamentos binários não equivalentes a FORNECE. (c) FORNECE representado como um tipo de entidade fraca.

de relacionamento (*f*, *j*, *p*), onde *f* é um FORNECEDOR que atualmente está abastecendo um PROJETO *j* com uma PECA *p*. Em geral, um tipo de relacionamento *R* de grau *n* terá *n* arestas em um diagrama ER, uma conectando *R* a cada tipo de entidade participante.

A Figura 7.17(b) mostra um diagrama ER para os três tipos de relacionamento binário PODE_FORNECER, USA e FORNECE. Em geral, um tipo de relacionamento ternário representa informações diferentes dos três tipos de relacionamento binário. Considere os três tipos de relacionamento binário PODE_FORNECER, USA e FORNECE. Suponha que PODE_FORNECER, entre FORNECEDOR e PECA, inclua uma instância (*f*, *p*) sempre que o fornecedor *f puder fornecer* a peça *p* (a qualquer projeto); USA, entre PROJETO e PECA, inclui uma instância (*j*, *p*) sempre que o projeto *j* usa a peça *p*; e FORNECE, entre FORNECEDOR e PROJETO, inclui uma instância (*f*, *j*) sempre que o fornecedor *f* fornece *alguma peça* ao projeto *j*. A existência de três instâncias de relacionamento (*f*, *p*), (*j*, *p*) e (*f*, *j*) em PODE_FORNECER, USA e FORNECE, respectivamente, não implica que existe uma instância (*f*, *j*, *p*) no relacionamento ternário FORNECE, pois o *significado é diferente*. Com frequência, é complicado decidir se um relacionamento em particular deve ser representado como um tipo de relacionamento de grau *n* ou se deve ser desmembrado em vários tipos de relacionamento de graus menores. O projetista deverá basear essa decisão na semântica ou significado da situação em particular que está sendo representada. A solução típica é incluir ao relacionamento ternário *com* um ou mais dos relacionamentos binários, se eles representarem significados diferentes e se todos forem necessários à aplicação.

Algumas ferramentas de projeto de banco de dados são baseadas em variações do modelo ER que permitem apenas relacionamentos binários. Nesse caso, um relacionamento ternário como FORNECE deve ser representado como um tipo de entidade fraca, sem chave parcial e com três relacionamentos de identificação. Os três tipos de entidade participantes FORNECEDOR, PECA e PROJETO são, juntos, os tipos de entidade proprietária (ver Figura 7.17(c)). Logo, uma entidade no tipo de entidade fraca FORNECE na Figura 7.17(c) é identificada pela combinação de suas três entidades proprietárias de FORNECEDOR, PECA e PROJETO.

Também é possível representar o relacionamento ternário como um tipo de entidade regular introduzindo uma chave artificial ou substituta. Neste exemplo, um atributo-chave Cod_fornecimento poderia ser usado para o tipo de entidade FORNECE, convertendo-o em um tipo de entidade regular. Três relacionamentos binários N:1 relacionam FORNECE aos três tipos de entidade participantes.

Outro exemplo é mostrado na Figura 7.18. O tipo de relacionamento ternário OFERECE representa informações sobre professores que oferecem cursos durante determinados semestres. Logo, ele inclui uma instância de relacionamento (*p*, *s*, *d*) sempre que o PROFESSOR *p* oferece a DISCIPLINA *d* durante o SEMESTRE *s*. Os três tipos de relacionamento binário mostrados na Figura 7.18 têm os seguintes significados: PODE_LECIONAR relaciona uma disciplina aos professores que *podem lecionar* esta disciplina, LECIONOU_DURANTE relaciona um semestre aos professores que *lecionaram alguma disciplina* durante esse semestre, e OFERECIDA_DURANTE relaciona um semestre às disciplinas oferecidas durante esse semestre *por qualquer professor*. Esses relacionamentos ternários e binários representam informações diferentes, mas certas restrições deverão ser mantidas entre os relacionamentos. Por exemplo, uma instância de relacionamento (*p*, *s*, *d*) não deve existir em OFERECE *a menos que* exista uma instância (*p*, *s*) em LECIONOU_DURANTE, uma instância (*s*, *d*) existe em OFERECIDA_DURANTE e uma instância (*p*, *d*) existe em PODE_LECIONAR. Contudo, a recíproca nem sempre é verdadeira: podemos ter instâncias (*p*, *s*), (*s*, *d*) e (*p*, *d*) nos três tipos de relacionamento binário sem a instância correspondente (*p*, *s*, *d*) em OFERECE. Observe que, neste exemplo, com base no significado dos relacionamentos, podemos deduzir as instâncias de LECIONOU_DURANTE e OFERECIDA_DURANTE com base nas instâncias em OFERECE, mas não podemos deduzir as instâncias de PODE_LECIONAR. Portanto, LECIONOU_DURANTE e OFERECIDA_DURANTE são redundantes e podem ser omitidas.

Embora em geral três relacionamentos binários *não possam* substituir um relacionamento ternário, eles podem fazer isso sob certas *restrições adicionais*. Em nosso exemplo, se o relacionamento PODE_LE-

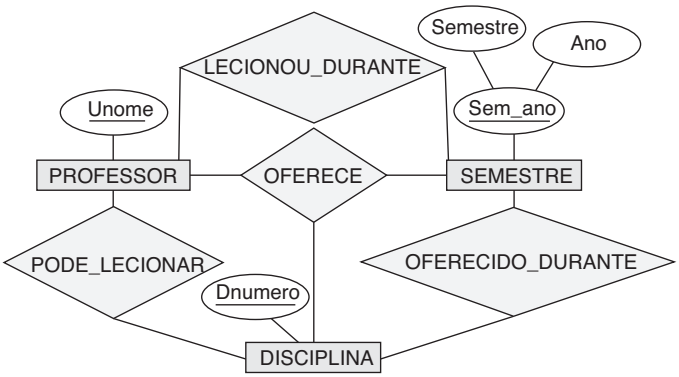

Figura 7.18
Outro exemplo de tipos de relacionamento ternário *versus* binário.

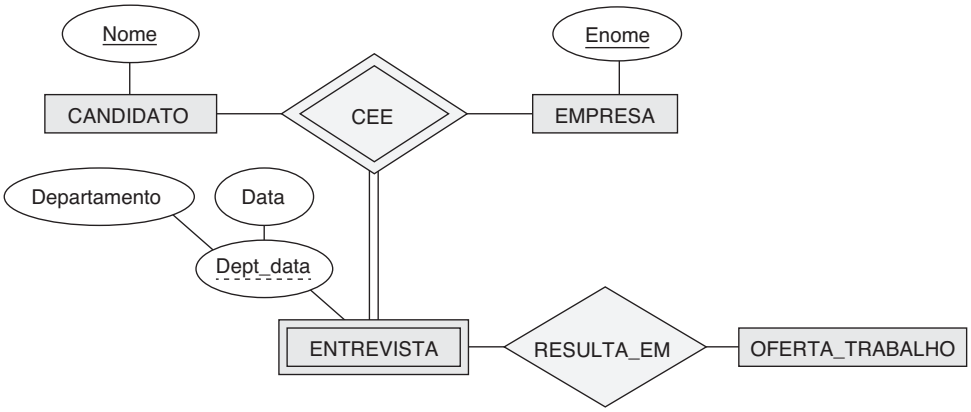

Figura 7.19
Um tipo de entidade fraca ENTREVISTA com um tipo de relacionamento de identificação ternário.

[15] Esta notação nos permite determinar a chave da ***relação do relacionamento***, conforme discutiremos no Capítulo 9.

[16] Isso também é verdadeiro para razões de cardinalidade dos relacionamentos binários.

CIONAR for 1:1 (um professor pode lecionar uma disciplina, e uma disciplina pode ser lecionada por apenas um professor), então o relacionamento ternário OFERECE pode ser omitido porque pode ser deduzido pelos três relacionamentos binários PODE_LECIONAR, LECIONOU_DURANTE e OFERECIDA_DURANTE. O projetista do esquema precisa analisar o significado de cada situação específica para decidir quais dos tipos de relacionamento binário e ternário são necessários.

Observe que é possível ter um tipo de entidade fraca com um ternário (*n*-ário) identificando o tipo de relacionamento. Nesse caso, o tipo de entidade fraca pode ter *vários* tipos de entidade proprietários. Um exemplo é mostrado na Figura 7.19. Ele mostra parte de um banco de dados que registra candidatos para entrevistas de emprego em diversas empresas, e pode fazer parte de um banco de dados de agência de emprego, por exemplo. Nos requisitos, um candidato pode ter várias entrevistas com a mesma empresa (por exemplo, com diferentes departamentos dela ou em datas separadas), mas uma oferta de emprego é feita com base em uma das entrevistas. Aqui, ENTREVISTA é representada como uma entidade fraca com dois proprietários CANDIDATO e EMPRESA, e com a chave parcial Dept_data. Uma entidade ENTREVISTA é identificada exclusivamente por um candidato, uma empresa e a combinação da data e departamento da entrevista.

7.9.2 Restrições sobre relacionamentos ternários (ou de grau mais alto)

Existem duas notações para especificar restrições estruturais sobre relacionamentos *n*-ários, e elas especificam restrições diferentes. Assim, *ambas devem ser usadas* se for importante determinar totalmente as restrições estruturais sobre um relacionamento ternário ou de grau maior. A primeira notação é baseada na notação de razão de cardinalidade dos relacionamentos binários exibidos na Figura 7.2. Aqui, um 1, M ou N é especificado em cada arco de participação (os símbolos M e N significam *muitos* ou *qualquer número*).[15] Vamos ilustrar essa restrição usando o relacionamento FORNECE da Figura 7.17.

Lembre-se de que o conjunto de relacionamento de FORNECE é um conjunto de instâncias de relacionamento (*f*, *j*, *p*), em que *f* é um FORNECEDOR, *j* é um PROJETO e *p* é uma PECA. Suponha que exista a restrição de que, para determinada combinação de projeto-peça, somente um fornecedor será usado (somente um fornecedor abastece determinado projeto com determinada peça). Nesse caso, colocamos 1 na participação de FORNECEDOR, e M, N nas participações de PROJETO, PECA na Figura 7.17. Isso especifica a restrição de que uma combinação em particular (*j*, *p*) pode aparecer no máximo uma vez no conjunto de relacionamento, pois cada combinação (PROJETO, PECA) desse tipo determina de maneira exclusiva um único fornecedor. Logo, qualquer instância de relacionamento (*f*, *j*, *p*) é identificada exclusivamente no conjunto de relacionamentos por sua combinação (*j*, *p*), que torna (*j*, *p*) uma chave para o conjunto de relacionamentos. Nessa notação, as participações que têm 1 especificado nelas não precisam fazer parte da chave de identificação para o conjunto de relacionamentos.[16] Se todas as três cardinalidades forem M ou N, então a chave será a combinação de todos os três participantes.

A segunda notação é baseada na notação (min, max) exibida na Figura 7.15 para relacionamentos

binários. Um (min, max) em uma participação aqui especifica que cada entidade está relacionada a pelo menos *min* e no máximo *max instâncias de relacionamento* no conjunto de relacionamentos. Essas restrições não têm influência na determinação da chave de um relacionamento *n*-ário, no qual $n > 2$,[17] mas especificam um tipo diferente de restrição, que faz restrições sobre o número de instâncias de relacionamento de que cada entidade participa.

Resumo

Neste capítulo, apresentamos os conceitos de modelagem de um modelo de dados conceitual de alto nível, o modelo Entidade-Relacionamento (ER). Começamos discutindo o papel que um modelo de dados de alto nível desempenha no processo de projeto de banco de dados, e depois apresentamos um exemplo de conjunto de requisitos para o banco de dados EMPRESA, que é um dos exemplos usados no decorrer deste livro. Definimos os conceitos básicos do modelo ER de entidades e seus atributos. Depois, discutimos os valores NULL e apresentamos os diversos tipos de atributos, que podem ser aninhados arbitrariamente para produzir atributos complexos:

- Simples ou atômicos.
- Compostos.
- Multivalorados.

Também discutimos rapidamente atributos armazenados *versus* derivados. Depois, abordamos os conceitos do modelo ER no nível de esquema ou 'conotação':

- Tipos de entidade e seus conjuntos de entidades correspondentes.
- Atributos-chave dos tipos de entidade.
- Conjuntos de valores (domínios) dos atributos.
- Tipos de relacionamento e seus conjuntos de relacionamentos correspondentes.
- Funções de participação dos tipos de entidade nos tipos de relacionamento.

Apresentamos dois métodos para especificar as restrições estruturais sobre tipos de relacionamento. O primeiro método distinguiu dois tipos de restrições estruturais:

- Razões de cardinalidade (1:1, 1:N, M:N para relacionamentos binários).
- Restrições de participação (total, parcial).

Observamos que, como alternativa, outro método de especificação de restrições estruturais é fazê-lo com números mínimo e máximo (min, max) sobre a participação de cada tipo de entidade em um tipo de relacionamento. Discutimos sobre tipos de entidade fraca e os conceitos relacionados de tipos de entidade proprietária, tipos de relacionamento de identificação e atributos-chave parciais.

Os esquemas Entidade-Relacionamento podem ser representados de maneira diagramática como diagramas ER. Mostramos como projetar um esquema ER para o banco de dados EMPRESA ao definir, primeiro, os tipos de entidade e seus atributos e depois refinando o projeto para incluir tipos de relacionamento. Apresentamos o diagrama ER para o esquema de banco de dados EMPRESA. Discutimos alguns dos conceitos básicos dos diagramas de classe UML e como eles se relacionam aos conceitos de modelagem ER. Também descrevemos os tipos de relacionamento ternário e de grau maior com mais detalhes, e discutimos as circunstâncias sob as quais eles são distinguidos dos relacionamentos binários.

Os conceitos de modelagem ER que apresentamos até aqui — tipos de entidade, tipos de relacionamento, atributos, chaves e restrições estruturais — podem modelar muitas aplicações de banco de dados. Contudo, aplicações mais complexas — como projeto de engenharia, sistemas de informações médicas e telecomunicações — exigem conceitos adicionais se quisermos modelá-las com maior precisão. Discutiremos alguns conceitos de modelagem avançados no Capítulo 8 e analisaremos técnicas de modelagem de dados ainda mais avançadas no Capítulo 26.

Perguntas de revisão

7.1. Discuta o papel de um modelo de dados de alto nível no processo de projeto de banco de dados.

7.2. Liste os diversos casos em que o uso de um valor NULL seria apropriado.

7.3. Defina os seguintes termos: *entidade*, *atributo*, *valor de atributo*, *instância de relacionamento*, *atributo composto*, *atributo multivalorado*, *atributo derivado*, *atributo complexo*, *atributo-chave* e *conjunto de valores* (*domínio*).

7.4. O que é um tipo de entidade? O que é um conjunto de entidades? Explique as diferenças entre uma entidade, um tipo de entidade e um conjunto de entidades.

7.5. Explique a diferença entre um atributo e um conjunto de valores.

7.6. O que é um tipo de relacionamento? Explique as diferenças entre uma instância de relacionamento, um tipo de relacionamento e um conjunto de relacionamentos.

7.7. O que é uma função de participação? Quando é necessário usar nomes de função na descrição dos tipos de relacionamento?

[17] No entanto, as restrições (min, max) podem determinar as chaves para relacionamentos binários.

7.8. Descreva as duas alternativas para especificar restrições estruturais sobre tipos de relacionamento. Quais são as vantagens e desvantagens de cada um?

7.9. Sob que condições um atributo de um tipo de relacionamento binário pode ser migrado para se tornar um atributo de um dos tipos de entidade participantes?

7.10. Quando pensamos nos relacionamentos como atributos, quais são os conjuntos de valores desses atributos? Que classe de modelos de dados é baseada nesse conceito?

7.11. O que queremos dizer com um tipo de relacionamento recursivo? Dê alguns exemplos.

7.12. Quando o conceito de uma entidade fraca é usado na modelagem de dados? Defina os termos *tipo de entidade proprietária*, *tipo de entidade fraca*, *tipo de relacionamento de identificação* e *chave parcial*.

7.13. Um relacionamento de identificação de um tipo de entidade fraca pode ser de um grau maior que dois? Dê exemplos para ilustrar sua resposta.

7.14. Discuta as convenções para exibir um esquema ER como um diagrama ER.

7.15. Discuta as convenções de nomeação usadas para os diagramas de esquema ER.

Exercícios

7.16. Considere o seguinte conjunto de requisitos para um banco de dados UNIVERSIDADE, que é usado para registrar os históricos dos alunos. Este é semelhante, mas não idêntico, ao banco de dados mostrado na Figura 1.2:

- **a.** A universidade registrar o nome, número de aluno, número do CPF, endereço atual e com seu número de telefone fixo, endereço permanente com seu número de telefone fixo, data de nascimento, sexo, turma (novato, segundo ano, ..., formado), departamento principal, departamento secundário (se houver) e programa de formação (graduação, mestrado, ..., doutorado) de cada aluno. Algumas aplicações do usuário precisam se referir à cidade, estado e CEP do endereço permanente do aluno e ao sobrenome do aluno. O número do CPF e o número de aluno possuem valores exclusivos para cada um deles.
- **b.** Cada departamento é descrito por um nome, código de departamento, número de escritório, número de telefone comercial e faculdade. Nome e código possuem valores exclusivos para cada departamento.
- **c.** Cada disciplina tem um nome, descrição, número de disciplina, número de horas por semestre, nível e departamento que oferece. O valor do número da disciplina é exclusivo para cada uma delas.
- **d.** Cada turma tem um professor, semestre, ano, disciplina e número de turma. O número de turma distingue as turmas da mesma disciplina que são lecionadas durante o mesmo semestre/ano; seus valores são 1, 2, 3, ..., até o número de turmas lecionadas durante cada semestre.
- **e.** Um relatório de notas tem um aluno, turma, nota com letra e nota numérica (0 A 10).

Projete um esquema ER para essa aplicação e desenhe um diagrama ER para o esquema. Especifique os atributos de chave de cada tipo de entidade, e as restrições estruturais sobre cada tipo de relacionamento. Observe quaisquer requisitos não especificados e faça suposições apropriadas para tornar a especificação completa.

7.17. Atributos compostos e multivalorados podem ser aninhados para qualquer número de níveis. Suponha que queiramos projetar um atributo para um tipo de entidade ALUNO a fim de registrar a formação acadêmica anterior. Esse atributo terá uma entrada para cada faculdade frequentada anteriormente, e cada entrada desse tipo será composta de um nome de faculdade, datas de início e término, entradas de título (títulos concedidos nessa faculdade, se houver) e entradas de histórico (disciplinas completadas nessa faculdade, se houver). Cada entrada de título contém o nome do título, o mês e o ano em que o título foi conferido, e cada entrada de histórico contém um nome de disciplina, semestre, ano e turma. Crie um atributo para manter essa informação. Use as convenções da Figura 7.5.

7.18. Mostre um projeto alternativo para o atributo descrito no Exercício 7.17 que use apenas tipos de entidade (incluindo tipos de entidade fraca, se for preciso) e tipos de relacionamento.

7.19. Considere o diagrama ER da Figura 7.20, que mostra um esquema simplificado para um sistema de reserva aérea. Extraia do diagrama ER os requisitos e restrições que produziram esse esquema. Tente ser o mais preciso possível em sua especificação de requisitos e restrições.

7.20. Nos capítulos 1 e 2, discutimos o ambiente e os usuários de banco de dados. Podemos considerar muitos tipos de entidade para descrever tal ambiente, como SGBD, banco de dados armazenado, DBA e catálogo/dicionário de dados. Tente especificar todos os tipos de entidade que podem descre-

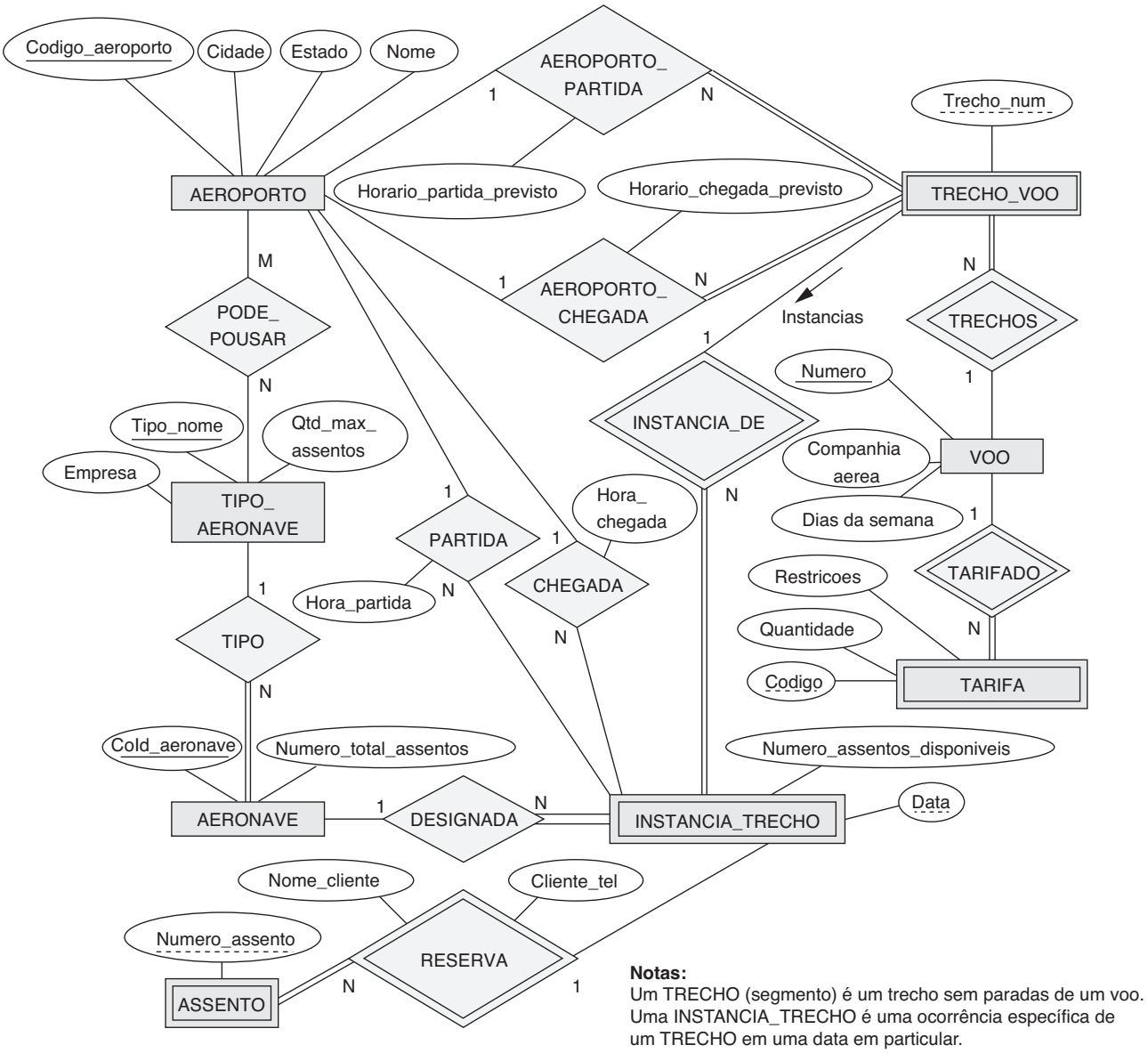

Figura 7.20
Um diagrama ER para um esquema de banco de dados COMPANHIA AEREA.

ver totalmente um sistema de banco de dados e seu ambiente; depois, especifique os tipos de relacionamento entre eles e desenhe um diagrama ER para descrever tal ambiente de banco de dados geral.

7.21. Projete um esquema ER para registrar informações sobre votos realizados no Congresso Nacional durante a sessão atual de dois anos no congresso. O banco de dados precisa registrar cada nome de ESTADO do Brasil (por exemplo, 'São Paulo', 'Rio de Janeiro', 'Porto Alegre') e incluir a Regiao do estado (cujo domínio é {'Nordeste', 'Centro-Oeste', 'Sudeste', 'Sul', 'Norte'}). Cada CONGRESSISTA no Congresso Nacional é descrito por seu Nome, mais o Estado representado, a Data_inicio em que o congressista foi eleito pela primeira vez e o Partido político ao qual ele ou ela pertence (cujo domínio é {'Oposição', 'Aliados', 'Independente', 'Outro'}). O banco de dados registra cada PROJETO_LEI (ou seja, lei proposta), incluindo a Nome_proj, a Data_votacao sobre a lei, se a lei passou_ou_falhou (cujo domínio é {'Sim', 'Não'}) e o Proponente (o congressista que patrocinou — ou seja, propôs — a lei). O banco de dados também registra como cada congressista votou em cada lei (domínio do atributo Voto é {'Sim', 'Não', 'Abstenção', 'Nulo'}). Desenhe um diagrama de esquema ER para essa aplicação. Indique claramente quaisquer suposições que você fizer.

7.22. Um banco de dados está sendo construído para registrar os times e jogos de uma liga esportiva. Um time tem uma série de jogadores, nem todos participando em todos os jogos. Deseja-se registrar os jogadores que participam em cada jogo para cada time, as posições em que eles jogaram e o resultado do jogo. Crie um diagrama de esquema ER para essa aplicação, indicando quaisquer suposições que você fizer. Escolha seu esporte favorito (por exemplo, futebol, basquete, voleibol).

7.23. Considere o diagrama ER mostrado na Figura 7.21 para parte de um banco de dados BANCO. Cada banco pode ter várias filiais, e cada filial pode ter várias contas e empréstimos.

- **a.** Liste os tipos de entidade forte (não fraca) no diagrama ER.
- **b.** Existe um tipo de entidade fraca? Se houver, diga seu nome, chave parcial e relacionamento de identificação.
- **c.** Quais restrições a chave parcial e o relacionamento de identificação do tipo de entidade fraca especificam nesse diagrama?
- **d.** Liste os nomes de todos os tipos de relacionamento e especifique a restrição (min, max) sobre cada participação de um tipo de entidade em um tipo de relacionamento. Justifique suas escolhas.
- **e.** Liste resumidamente os requisitos do usuário que levaram a esse projeto de esquema ER.
- **f.** Suponha que cada cliente deva ter pelo menos uma conta, mas esteja restrito a no máximo dois empréstimos de cada vez, e que uma filial de banco não pode ter mais de 1.000 empréstimos. Como isso é exposto nas restrições (min, max)?

7.24. Considere o diagrama ER da Figura 7.22. Suponha que um funcionário possa trabalhar em até dois departamentos ou não possa ser atribuído a qualquer departamento. Suponha que cada departamento deva ter um e possa ter até três números de telefone. Forneça restrições (min, max) sobre esse diagrama. *Indique claramente quaisquer suposições adicionais que estiver fazendo*. Sob que condições o relacionamento POSSUI_TELEFONE seria redundante neste exemplo?

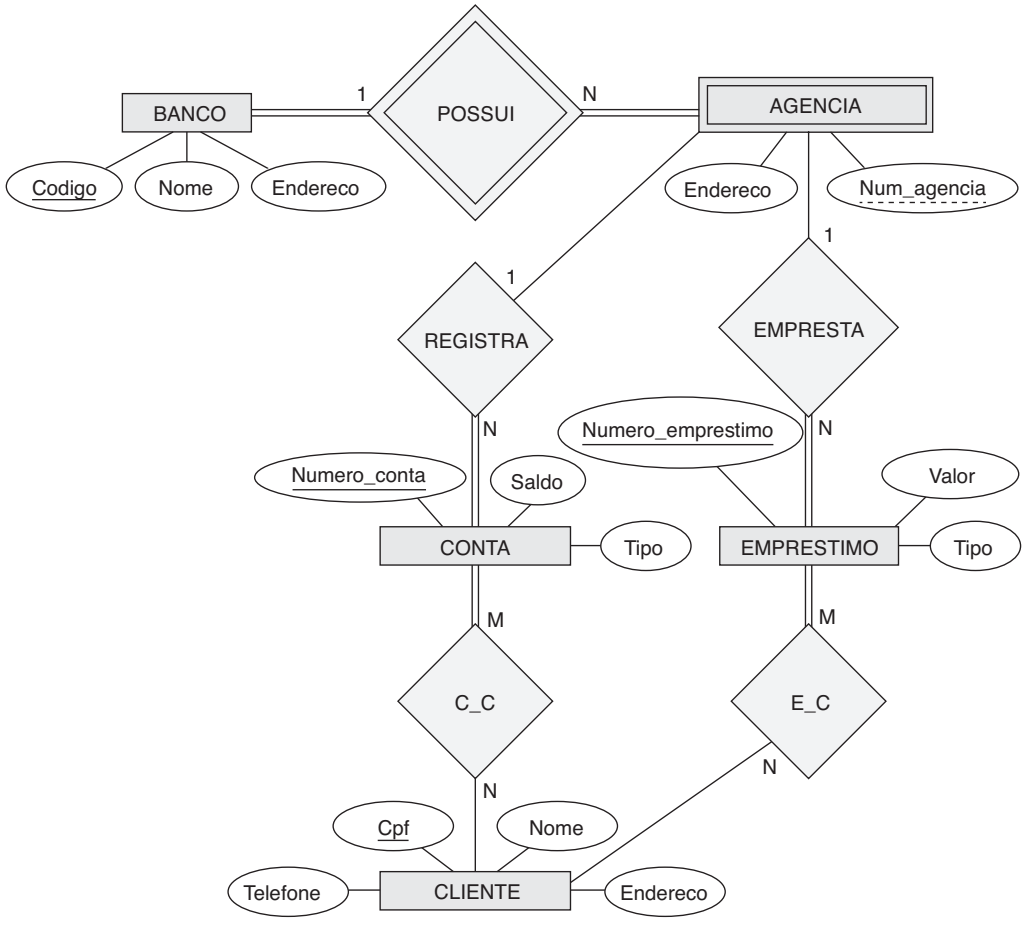

Figura 7.21
Um diagrama ER para um esquema de banco de dados BANCO.

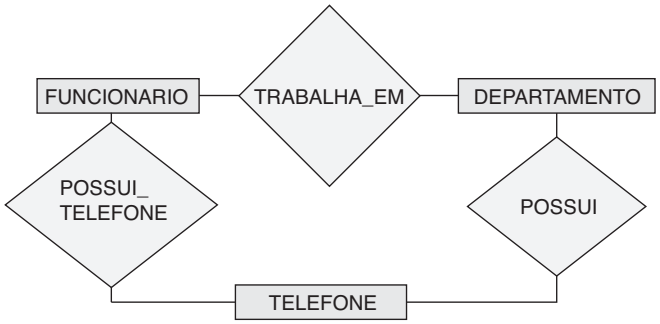

Figura 7.22
Parte de um diagrama ER para um banco de dados EMPRESA.

7.25. Considere o diagrama ER da Figura 7.23. Suponha que uma disciplina possa ou não usar um livro-texto, mas que um texto por definição é um livro que é usado em alguma disciplina. Uma disciplina não pode usar mais de cinco livros. Os professores lecionam de duas a quatro disciplinas. Forneça restrições (min, max) sobre esse diagrama. *Indique claramente quaisquer suposições adicionais que estiver fazendo*. Se acrescentarmos o relacionamento ADOTADO, para indicar o(s) livros(s)-texto que um professor utiliza para uma disciplina, ele deverá ser um relacionamento binário entre PROFESSOR e LIVRO_TEXTO, ou um relacionamento ternário entre todos os três tipos de entidade? Que restrições (min, max) você incluiria? Por quê?

7.26. Considere um tipo de entidade TURMA em um banco de dados UNIVERSIDADE, que descreve as ofertas de turmas das disciplinas. Os atributos da TURMA são Numero_turma, Semestre, Ano, Numero_disciplina, Professor, Num_sala (em que a turma é realizada), Predio (onde a turma é realizada), Dias_da_semana (domínio são as combinações possíveis de dias da semana em que a turma pode ser oferecida {'SQS', 'SQ', 'TQ' e assim por diante}) e Horas (domínio são todos os períodos possíveis durante os quais as turmas são oferecidas {'9–9:50', '10–10:50', ..., '15:30–16:50', '17:30–18:20', e assim por diante}). Suponha que Numero_turma seja exclusivo para cada disciplina em determinada combinação de semestre/ano (ou seja, se uma disciplina for oferecida várias vezes durante um semestre em particular, suas ofertas são numeradas com 1, 2, 3, e assim por diante). Existem várias chaves compostas por turma, e alguns atributos são componentes de mais de uma chave. Identifique três chaves compostas e mostre como elas podem ser representadas em um diagrama de esquema ER.

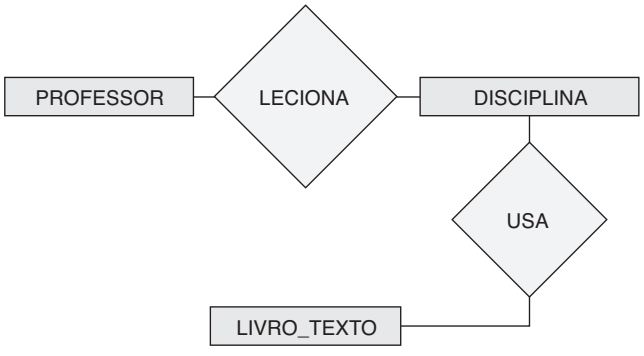

Figura 7.23
Parte de um diagrama ER para um banco de dados DISCIPLINAS.

7.27. Razões de cardinalidade normalmente ditam o projeto detalhado de um banco de dados. A razão de cardinalidade depende do significado no mundo real dos tipos de entidade envolvidos e é definida pela aplicação específica. Para os seguintes relacionamentos binários, sugira razões de cardinalidade com base no significado comum dos tipos de entidade. Indique claramente quaisquer suposições que você fizer.

	Entidade 1	Razão de cardinalidade	Entidade 2
1.	ALUNO	_____	CADASTRO_ PESSOA_FISICA
2.	ALUNO	_____	PROFESSOR
3.	SALA_AULA	_____	PAREDE
4.	PAIS	_____	PRESIDENTE_ATUAL
5.	DISCIPLINA	_____	LIVRO_TEXTO
6.	ITEM (que pode ser encontrado em um pedido)	_____	PEDIDO
7.	ALUNO	_____	AULA
8.	AULA	_____	PROFESSOR
9.	PROFESSOR	_____	ESCRITORIO
10.	ITEM_LEILOADO	_____	COD_LEILAO

7.28. Considere o esquema ER para o banco de dados FILMES mostrado na Figura 7.24.

Suponha que FILMES seja um banco de dados preenchido. ATOR é usado como um termo genérico e inclui atrizes. Dadas as restrições mostradas no esquema ER, responda às seguintes afirmações com *Verdadeira*, *Falsa* ou *Talvez*. Atribua uma resposta *Talvez* a declarações que, embora não mostradas explicitamente como sendo *Verdadeiras*, não se pode provar que sejam *Falsas* com base no esquema mostrado. Justifique cada resposta.

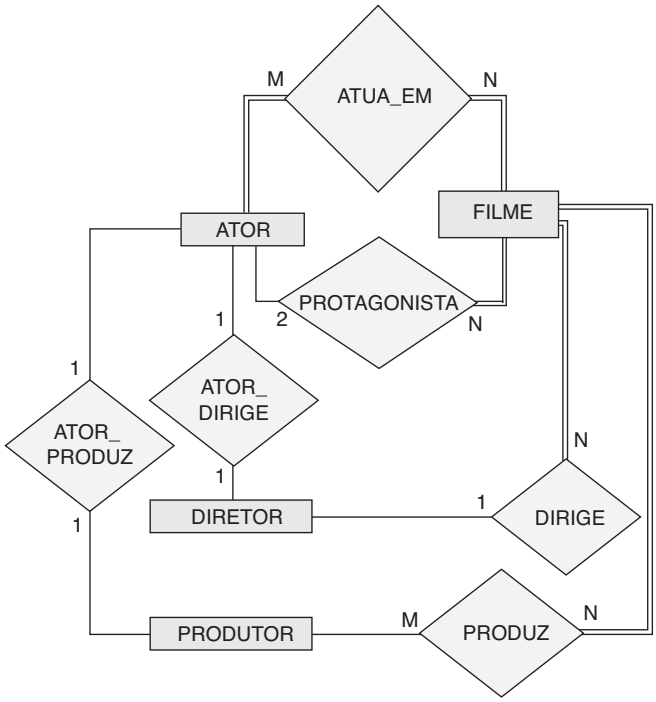

Figura 7.24
Um diagrama ER para um esquema de banco de dados FILMES.

a. Não existem atores neste banco de dados que não estiveram em nenhum filme.
b. Existem alguns atores que atuaram em mais de dez filmes.
c. Alguns atores foram protagonistas em vários filmes.
d. Um filme só pode ter um máximo de dois atores protagonistas.
e. Cada diretor foi ator em algum filme.
f. Nenhum produtor já foi um ator.
g. Um produtor não pode ser ator em outro filme.
h. Existem filmes com mais de doze atores.
i. Alguns produtores também já foram diretores.
j. A maioria dos filmes tem um diretor e um produtor.
k. Alguns filmes têm um diretor, mas vários produtores.
l. Existem alguns atores que foram protagonistas, dirigiram um filme e produziram algum filme.
m. Nenhum filme tem um diretor que também atuou nesse filme.

7.29. Dado o esquema ER para o banco de dados FILMES da Figura 7.24, desenhe um diagrama de instância usando três filmes que foram lançados recentemente. Desenhe instâncias de cada tipo de entidade: FILMES, ATORES, PRODUTORES, DIRETORES envolvidos; crie instâncias dos relacionamentos conforme existem na realidade para esses filmes.

7.30. Ilustre o Diagrama UML para o Exercício 7.16. Seu projeto UML deverá observar os seguintes requisitos:

a. Um aluno deverá ter a capacidade de calcular sua média e acrescentar ou retirar disciplinas obrigatórias e optativas.
b. Cada departamento deverá ser capaz de acrescentar ou retirar disciplinas e contratar ou demitir o corpo docente.
c. Cada professor deverá ser capaz de atribuir ou alterar a nota de um aluno para uma disciplina.

Nota: algumas dessas funções podem se espalhar por várias turmas.

Exercícios de laboratório

7.31. Considere o banco de dados UNIVERSIDADE descrito no Exercício 7.16. Crie o esquema ER para esse banco de dados usando uma ferramenta de modelagem de dados como ERwin ou Rational Rose.

7.32. Considere um banco de dados PEDIDO_CORREIO em que os funcionários fazem pedidos de peças dos clientes. Os requisitos de dados são resumidos da seguinte forma:

- A empresa que vende por catálogo tem funcionários, cada um identificado por um número de funcionário exclusivo, nome e sobrenome, e CEP.
- Cada cliente da empresa é identificado por um número de cliente exclusivo, nome e sobrenome, e CEP.
- Cada peça vendida pela empresa é identificada por um número de peça exclusivo, um nome de peça, preço e quantidade em estoque.
- Cada pedido feito por um cliente é recuperado por um funcionário e recebe um número de pedido exclusivo. Cada pedido contém quantidades especificadas de uma ou mais peças. Cada pedido tem uma data de recebimento bem como uma data de entrega esperada. A data de entrega real também é registrada.

Crie um diagrama Entidade-Relacionamento para o banco de dados de compras por catálogo e construa o projeto usando uma ferramenta de modelagem como ERwin ou Rational Rose.

7.33. Considere um banco de dados FILME em que os dados são registrados sobre a indústria do cinema. Os requisitos de dados são resumidos a seguir:

- Cada filme é identificado por um título e ano de lançamento. Cada filme tem uma duração em minutos. Cada um tem uma companhia produtora, e

é classificado sob um ou mais gêneros (como terror, ação, drama etc.). Cada filme tem um ou mais diretores e um ou mais atores participando dele. Também tem um resumo da trama. Finalmente, cada filme tem zero ou mais falas, cada uma delas dita por um ator em particular que aparece no filme.

- Os atores são identificados por nome e data de nascimento e aparecem em um ou mais filmes. Cada ator tem um papel no filme.
- Os diretores também são identificados por nome e data de nascimento e dirigem um ou mais filmes. É possível que um diretor atue em um filme (incluindo aquele que ele ou ela também pode dirigir).
- As empresas produtoras são identificadas por nome e cada uma tem um endereço. Uma produtora produz um ou mais filmes.

Crie um diagrama Entidade-Relacionamento para o banco de dados de filmes e construa o projeto usando uma ferramenta de modelagem de dados, como ERwin ou Rational Rose.

7.34. Considere um banco de dados REVISAO_CONFERENCIA em que os pesquisadores submetem seus artigos de pesquisa para avaliação. As análises dos revisores são registradas para uso no processo de seleção de artigo. O sistema de banco de dados atende principalmente a revisores que registram respostas a perguntas de avaliação para cada artigo que eles revisam e fazem recomendações com relação a se aceitar ou rejeitar o artigo. Os requisitos de dados são resumidos da seguinte forma:

- Autores de artigos são identificados exclusivamente pelo correio eletrônico. Os nomes e sobrenomes também são registrados.
- Cada artigo recebe um identificador exclusivo pelo sistema e é descrito por um título, resumo e o nome do arquivo eletrônico que contém o artigo.
- Um artigo pode ter vários autores, mas um deles é designado como o autor de contato.
- Os revisores dos artigos são identificados exclusivamente pelo endereço de correio eletrônico. Nome, sobrenome, número de telefone, afiliação e tópicos de interesse de cada revisor também são registrados.
- Cada artigo é atribuído a dois a quatro revisores. Um revisor avalia cada artigo atribuído a ele ou ela em uma escala de 1 a 10, em quatro categorias: mérito técnico, legibilidade, originalidade e relevância à conferência. Por fim, cada revisor oferece uma recomendação geral com relação a cada artigo.
- Cada revisão contém dois tipos de comentários escritos: um a ser visto pelo comitê de revisão apenas e o outro como retorno ao(s) autor(es).

Crie um diagrama Entidade-Relacionamento para o banco de dados REVISAO_CONFERENCIA e construa o projeto usando uma ferramenta de modelagem como ERwin ou Rational Rose.

7.35. Considere o diagrama ER para o banco de dados COMPANHIA AEREA mostrado na Figura 7.20. Construa esse projeto usando uma ferramenta de modelagem como ERwin ou Rational Rose.

Bibliografia selecionada

O modelo Entidade-Relacionamento foi introduzido por Chen (1976) e um trabalho relacionado aparece em Schmidt e Swenson (1975), Wiederhold e Elmasri (1979) e Senko (1975). Desde então, diversas modificações foram sugeridas no modelo ER. Incorporamos algumas delas em nossa apresentação. Restrições estruturais sobre os relacionamentos são discutidas em Abrial (1974), Elmasri e Wiederhold (1980), e Lenzerini e Santucci (1983). Atributos multivalorados e compostos são incorporados ao modelo ER em Elmasri et al. (1985). Embora não tenhamos discutido as linguagens para o modelo ER e suas extensões, há várias propostas para tais linguagens. Elmasri e Wiederhold (1981) propuseram a linguagem de consulta GORDAS para o modelo ER. Outra linguagem de consulta ER foi proposta por Markowitz e Raz (1983). Senko (1980) apresentou uma linguagem de consulta para o modelo DIAM de Senko. Um conjunto formal de operações, chamado álgebra ER, foi apresentado por Parent e Spaccapietra (1985). Gogolla e Hohenstein (1991) apresentaram outra linguagem formal para o modelo ER. Campbell et al. (1985) apresentaram um conjunto de operações ER e mostraram que elas são completas no sentido relacional. Uma reunião para a disseminação dos resultados de pesquisa relacionada ao modelo ER tem sido mantida regularmente desde 1979. A conferência, agora conhecida como International Conference on Conceptual Modeling, foi realizada em Los Angeles (ER 1979, ER 1983, ER 1997), Washington, D.C. (ER 1981), Chicago (ER 1985), Dijon, na França (ER 1986), Nova York (ER 1987), Roma (ER 1988), Toronto (ER 1989), Lausanne, na Suíça (ER 1990), San Mateo, na Califórnia (ER 1991), Karlsruhe, na Alemanha (ER 1992), Arlington, no Texas (ER 1993), Manchester, na Inglaterra (ER 1994), Brisbane, na Austrália (ER 1995), Cottbus, na Alemanha (ER 1996), Cingapura (ER 1998), Paris, na França (ER 1999), Salt Lake City, em Utah (ER 2000), Yokohama, no Japão (ER 2001), Tampere, na Finlândia (ER 2002), Chicago, em Illinois (ER 2003), Shanghai, na China (ER 2004), Klagenfurt, na Áustria (ER 2005), Tucson, no Arizona (ER 2006), Auckland, na Nova Zelândia (ER 2007), Barcelona, Catalunha, na Espanha (ER 2008) e Gramado, no Rio Grande do Sul, Brasil (ER 2009). A conferência de 2010 será realizada em Vancouver, BC, no Canadá.