

Aula 3 – O modelo de dados relacional

Objetivo

Analisar os principais conceitos do Modelo Relacional que representa um banco de dados em um conjunto de relações.

Na aula anterior conhecemos e vimos como construir diagramas utilizando o modelo ER. Nesta aula conheceremos o modelo de dados relacional. O Modelo de Dados Relacional foi introduzido por Ted Codd da IBM em 1970. Entre os modelos de dados existentes, o modelo relacional é o mais simples e difundido, com uma estrutura de dados uniforme, e também o mais formal.

3.1 Conceitos

O Modelo Relacional será a base para implementarmos nosso banco de dados. Ele representa os dados da base de dados como uma coleção de relações. Podemos pensar que, cada relação pode ser entendida como uma tabela ou um simples arquivo de registros (Citação). Por exemplo, a base de dados de arquivos que são mostradas pela Figura 3.1, é similar a representação do modelo relacional. No entanto, possui algumas importantes diferenças entre relações e arquivos como veremos a seguir.

ALUNO	Nome	Numero	Turma	Curso_Hab
	Smith	17	1	CC
	Brown	8	2	CC

NomedoCurso	Numerodocurso	Creditos	Departamento
Introdução à Ciência da Computação	CC1310	4	CC
Estrutura de Dados	CC3320	4	CC
Matemática Discreta	MAT2410	3	MATH
Banco de Dados	CC3380	3	CC

DISCIPLINA	IdentificadordeDisciplina	Numerodo-Curso	Semestre	Ano	Instrutor
	85	MAT2410	Segundo Semestre	98	Kihg
	92	CC1310	Segundo Semestre	98	Anderson
	102	CC3320	Primeiro Semestre	99	Knuth
	112	MAT2410	Segundo Semestre	99	Chang
	119	CC1310	Segundo Semestre	99	Anderson
	135	CC3380	Segundo Semestre	99	Stone

HISTORICO,,ESCOLAR	NumerodoAluno	Identificador/Disciplinas	Nota
	17	112	B
	17	119	C
	8	85	A
	8	92	A
	8	102	B
	8	135	A

PRE_REQUISITO	NumerodoCurso	NumerodoPre_requisito
	CC3380	CC3320
	CC3380	MAT2410
	CC3320	CC1310

Figura 3.1: Exemplo de um arquivo de dados baseado no modelo relacional

Fonte: ELMASRI & NAVATHE, 2005

Ao pensarmos em uma relação como uma tabela de valores, cada linha representa uma coleção de valores que estão relacionados. Esses valores podem ser interpretados como um fato que descreve uma entidade ou uma instância. Utilizamos o nome da tabela e os nomes das colunas para nos ajudar a interpretar o significado dos valores em cada linha da tabela, que são na verdade, dados a respeito dos dados, chamados **metadados**.

Observe a Figura 3.1: a primeira tabela é chamada ALUNO porque cada linha representa fatos sobre uma entidade aluno em particular. Os nomes das colunas - Nome, Número, Turma, Departamento - especificam como interpretar os valores em cada linha baseando-se nas colunas que cada valor se encontra.



Na linguagem do modelo relacional, cada linha é chamada de **tupla**, a coluna ou cabeçalho é chamado de **atributo** e a **tabela** de relação. Desta maneira, o conjunto de nomes das tabelas e suas colunas são chamados de **esquema da relação**. Assim, o esquema da relação ALUNO é:

ALUNO = {nome, número, turma, departamento}

Temos que conhecer também conceito de **grau de uma relação**, este é o número de atributos da relação. No exemplo acima, o grau de relação do esquema ALUNO é quatro, pois possui quatro colunas.

Uma coluna de dados possui um tipo de dado que descreve os valores que podem aparecer nela, por exemplo, na coluna **NÚMERO** de um aluno esperamos um valor numérico como 17, 18 e não caracteres, este tipo de dado que especifica os possíveis valores de uma coluna é chamada de **domínio**.

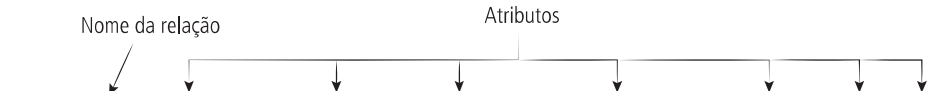
No esquema de relação ALUNO podemos especificar alguns domínios para atributos da relação ALUNO, vejamos alguns exemplos:

- **Nome:** Conjunto de cadeia de caracteres que representa nomes de pessoas;
- **Número:** Conjunto de dados numéricos com limite de cinco dígitos;
- **Turma:** Conjuntos de códigos das turmas da faculdade;
- **Departamento:** Conjunto de códigos dos departamentos acadêmicos, como CC, EP, etc.

Assim, de acordo com nosso exemplo de domínio para a tabela ALUNO, o que esperamos obter em uma linha (tupla), é o conjunto de valores dos atributos para um determinado estudante. Por exemplo, existe um aluno de nome Smith, seu número é 17, sua turma é 1 e seu departamento CC.



A Figura 3.2 mostra um exemplo da relação ALUNO. Cada tupla representa uma entidade aluno em particular, cada atributo corresponde a um cabeçalho de coluna, os valores apresentados como nulos (*null*) representam atributos em que os valores não existem para alguma tupla individual de ALUNO.



ALUNO	Nome	INSS	FoneResidencia	Endereco	FoneEscritorio	Idade	MPG
→	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	<i>null</i>	19	3.21
→	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	<i>null</i>	18	2.89
→	Dick Davidson	422-11-2320	<i>null</i>	3452 Elgin Road	749-1253	25	3.53
→	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
→	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	<i>null</i>	19	3.25

Figura 3.2: Conjunto de atributos e tuplas da relação ALUNO

Fonte: ELMASRI & NAVATHE, 2005

3.2 Atributos-chaves

Uma relação é definida como um conjunto de tuplas. Por padrão, todos os elementos de um conjunto devem ser distintos. Assim, todas as tuplas de uma relação também são distintas. Isto significa que nenhuma tupla pode ter a mesma combinação de valores para todos os seus atributos. Desta maneira, temos que ter um valor que chamamos de **atributo-chave** que é utilizado para identificar de modo unívoco uma tupla em uma relação.



Vamos voltar à Figura 3.1, nela podemos verificar que o Nr 18 da relação ALUNO serve para identificar o aluno Brown

Geralmente, um esquema de relação pode ter mais que uma chave. Nos casos em que isto ocorra, cada chave é chamada **chave-candidata**. Por exemplo, o esquema da relação ALUNO poderia ter um atributo adicional Código, para indicar o código interno de alunos na escola. Assim, o esquema teria duas chaves candidatas: Nr e Código.

Após identificarmos as chaves candidatas devemos definir uma delas como a **chave-primária** da relação. A indicação no modelo de qual chave-candidata é a chave-primária é realizada se sublinhado os atributos que formam a

chave-candidata escolhida como podemos ver na Figura 3.3.

ALUNO	ALUNO
•Nome	♦Codigo
♦Nr	•Nome
•Turma	•Nr
•Departamento	•Turma
	•Departamento

Figura 3.3: Tabela ALUNO com duas chaves candidatas: Nr e Codigo

Fonte: Elaborada pelo autor

Vejam os um exemplo, o atributo Nome da relação ALUNO não deve ser indicado como chave, uma vez que nada garante a que não haja ocorrência de nomes duplicados (homônimos).



Em certas relações pode ser necessário mais de um atributo para identificar cada tupla da relação de forma unívoca. Por exemplo, vejamos a relação PESSOA apresentada na Figura 3.4 a seguir.

PESSOA
•Nome
♦RG
♦OrgaoEmissor
•DataNasc

Figura 3.4: Tabela PESSOA utilizando chave composta

Fonte: Elaborada pelo autor

Conforme já vimos na aula dois, um determinado atributo pode não garantir a unicidade de uma tupla, desta maneira a identificação de cada pessoa deve ser feita pelo valor do atributo RG e do atributo OrgaoEmissor. A utilização de mais de um atributo para a composição da chave primária chamamos de **chave composta**. Alternativamente para não utilizarmos uma chave composta na relação PESSOA, poderíamos incluir um campo de identificação única como um código, por exemplo.

3.3 Chave estrangeira

O conceito de chave estrangeira é de grande importância na construção de banco de dados relacional. Vamos começar com exemplo:

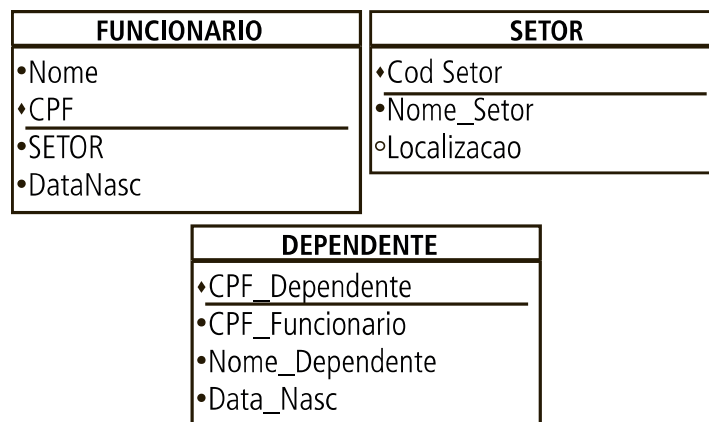


Figura 3.5: Tabelas para um esquema de um banco de dados relacional

Fonte: Elaborada pelo autor

No esquema de tabelas apresentada na Figura 3.5 temos três relações (tabelas). A relação FUNCIONARIO possui os dados de um funcionário de uma empresa, como o Nome, CPF, Setor onde trabalha e Data de Nascimento. A relação SETOR possui o Nome e Localização de um setor que é identificada por um código. A relação DEPENDENTE possui o CPF deste como chave primária, o CPF do funcionário o qual ele é dependente, o Nome e a Data de Nascimento do dependente.

Podemos verificar que alguns atributos estão presentes em mais de uma tabela. Através das relações apresentadas abaixo podemos verificar que o atributo Setor da tabela FUNCIONARIO representa o código do setor onde o funcionário está lotado, sendo o mesmo atributo Cod_Setor da tabela SETOR. O mesmo caso ocorre entre os atributos CPF e CPF_Funcionario das tabelas FUNCIONARIO e DEPENDENTE respectivamente. A Figura 3.6 apresentada abaixo ilustra esta relação.

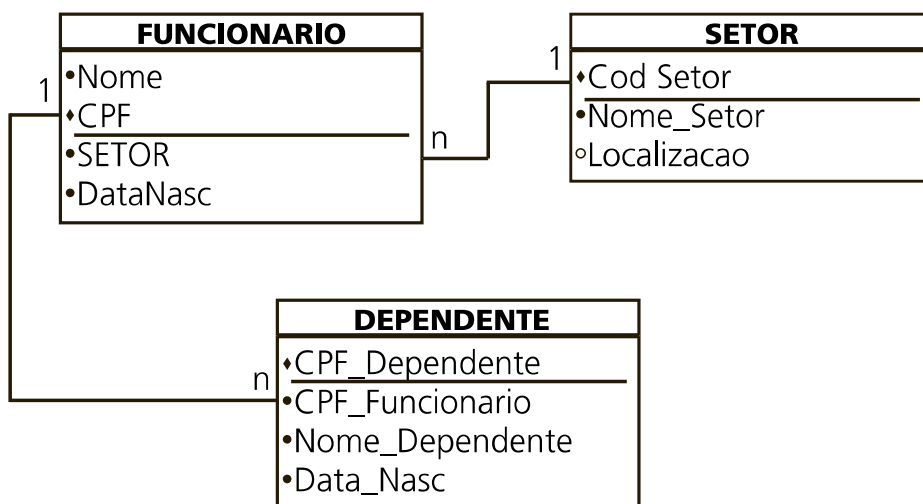


Figura 3.6: Relação entre tabelas

Fonte: Elaborada pelo autor

Como podemos verificar na Figura 3.6, apenas a chave primária de uma tabela deve ser repetida em outra tabela. É o que acontece no esquema relacional apresentado. A chave primária de FUNCIONARIO está representada na tabela DEPENDENTE e a chave primária da tabela SETOR está representada na tabela FUNCIONARIO, como **chave estrangeira**. Desta maneira, podemos saber, por exemplo, qual é o SETOR de um funcionário (através do atributo Setor) ou quais dependentes possui um FUNCIONARIO através do atributo CPF_Funcionario.

O valor do atributo Setor na tabela FUNCIONARIO poderia ser *null* (nulo) se o funcionário não estiver locado em nenhum setor no momento.



Desta forma, a **chave estrangeira** possibilita a implementação do conceito de relacionamento entre entidades. Com isto, podemos no Modelo Relacional, realizar a modelagem representada no Modelo Entidade Relacionamento que mostra um conjunto de entidades relacionado a outro conjunto de entidades, com determinada razão de cardinalidade. Então, o diagrama ER correspondente ao mapeamento realizada na Figura 3.6 poderia ser representado conforme a Figura 3.7 a seguir.

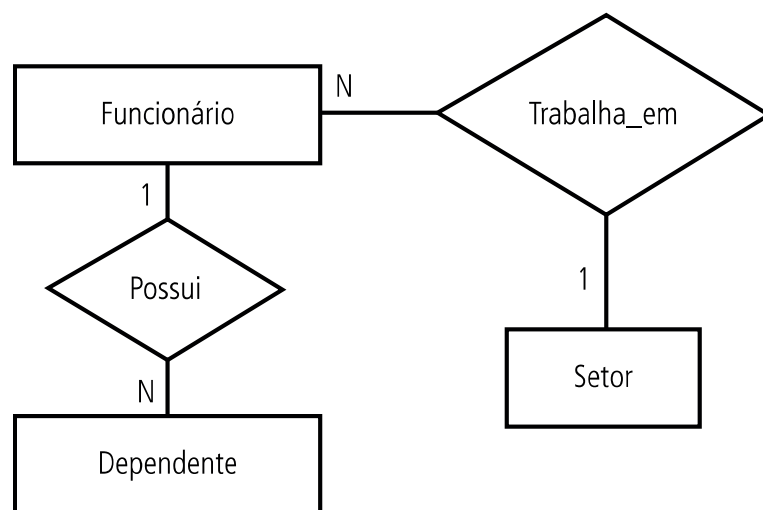


Figura 3.7: Diagrama ER entre entidades

Fonte: Elaborada pelo autor

3.4 Integridade referencial

A **restrição de integridade referencial** é responsável por manter a consistência entre tuplas (registros) de duas relações. Esta restrição de integridade referencial estabelece que uma tupla de uma relação que se refere à outra relação deve se referir a uma tupla existente naquela relação. Vamos exemplificar para ficar mais claro, vejamos a Figura 3.8:

FUNCIONARIO			
NOME	CPF	SETOR	DATA_NASC
Jose	33875419294	1	10/05/1980
Carlos	26218188892	2	17/05/1970

Setor		
Cod_Setor	Nome	Localizacao
1	Informatica	Andar 4
2	RH	Térrio

Figura 3.8: Relações FUNCIONARIO e SETOR

Fonte: Elaborada pelo autor

Na Figura 3.8 podemos observar que o atributo Setor de FUNCIONARIO indica o número do SETOR que cada funcionário trabalha. Assim, todos os valores do atributo Setor (chave estrangeira) nas tuplas da relação FUNCIONARIO devem pertencer ao conjunto de valores do atributo Cod_Setor (chave primária) da relação SETOR. Desta maneira podemos definir que o conceito de Integridade Referencial decorre da implementação de chaves estrangeiras em um esquema relacional. Assim temos duas regras para dizer que um conjunto de atributos será chave estrangeira de uma relação:

- Os atributos da **chave estrangeira** têm o mesmo **domínio** dos atributos da **chave-primária** a qual se relaciona. Podemos dizer então que os atributos chave estrangeira fazem referência à chave primária.
- O valor da **chave estrangeira** será algum valor que ocorre na **chave primária** a que ela faz referência ou terá o valor *null*.

Se as duas regras apresentadas acima não ocorrerem, não há INTEGRIDADE REFERENCIAL no banco de dados implementado.



As restrições de integridade devem ser especificadas no esquema da base de dados relacional se o projetista quiser manter essas restrições válidas para toda a base de dados. Desta maneira, em um sistema relacional, a linguagem de definição de dados (DDL) deve fornecer métodos para especificar os vários tipos de restrições tal que o SGDB possa garanti-las automaticamente.

3.5 Mapeamento

Como já vimos anteriormente, o Modelo Relacional (MRel) implementa as tabelas relacionadas, com muitos recursos para segurança dos dados, controle de acesso, consultas e manutenção dos dados. Porém, este modelo não é o modelo mais adequado para se fazer projetos de banco de dados. Então conhecemos o Modelo Entidade Relacionamento MER, que por meio de seus recursos visuais, se apresenta mais claro, simples e intuitivo.

Desta maneira, em projetos de banco de dados, normalmente a modelagem dos dados é realizada através de um modelo de dados de alto-nível. O modelo de dados de alto-nível geralmente adotado é o Modelo Entidade-Relacionamento (MER) e o esquema das visões e de toda a base de dados é especificado em diagramas entidade-relacionamento (DER).

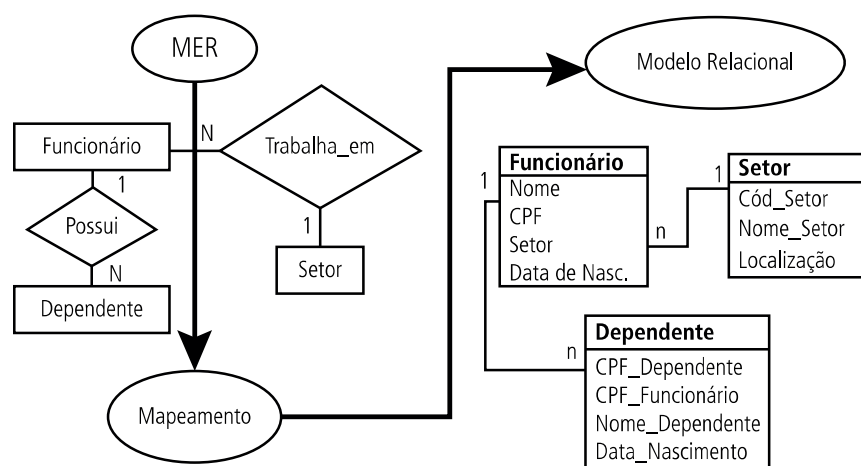


Figura 3.9: Mapeamento entre MER e MRel

Fonte: Elaborada pelo autor

O Modelo Entidade Relacionamento possui um maior número de abstrações do que as que foram vistas em nossas aulas. Existem ainda definições para: Conjunto de Entidades Fracas, Atributos Compostos, Atributos Multivalorados, por exemplo. Vimos em nossas aulas os principais conceitos do modelo, que permitem a construção da maior parte dos bancos de dados para as aplicações comerciais mais comuns.



O livro texto deste caderno - ELMASRI & NAVATHE, 2005 - descreve todas as extensões possíveis de um banco de dados, é muito útil para o estudante que queira se aprofundar no conteúdo.

Agora, veremos como traduzir um DER para um esquema MRel. Para isto utilizaremos um conjunto de passos para que possamos implementar nosso banco de dados em um SGBD Relacional.

Passo 1: Mapeamento dos tipos de entidades: Consiste na criação de uma relação que inclua todos os atributos de uma entidade. Assim, as entidades do MER são transformadas em tabelas no MRel. O atributo chave da entidade passa a ser a chave primária da relação (tabela). Vejamos um exemplo na Figura 3.10.

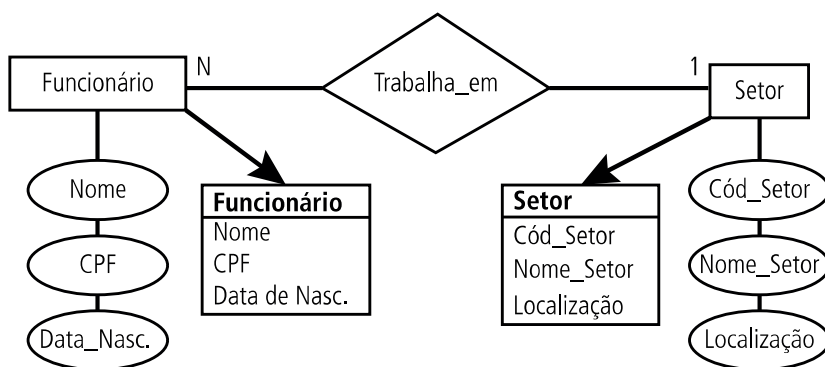


Figura 3.10: Primeiro passo no mapeamento entre MER e MRel

Fonte: Elaborada pelo autor

No exemplo da Figura 3.10, criamos as relações FUNCIONARIO e SETOR, que correspondem às entidades FUNCIONARIO e SETOR presentes no DER. Neste momento não nos preocupamos ainda com os relacionamentos. O que fizemos foi apenas definir as chaves primárias conforme dito anteriormente.

Nos próximos passos iremos realizar o mapeamento dos relacionamentos. Este mapeamento depende do grau do relacionamento (número de conjuntos de entidades nele envolvidos – binário, ternário, etc.) e da razão de cardinalidades (1:1, 1:N e N:M).

Passo 2: Mapeamento dos Conjuntos de Relacionamentos binários de razão de cardinalidade 1:1: Para este caso temos 3 possíveis opções de mapeamento: (1) criar chave estrangeira em uma das relações, (2) gerar uma única relação para as entidades e o relacionamento, (3) gerar uma relação exclusiva para o relacionamento, porém, esta opção caracteriza como veremos mais adiante o mapeamento N:M.

As opções mais utilizadas e que devem ser seguidas são a primeira e a terceira.



Vamos observar na Figura 3.11 apresentada abaixo uma opção de mapeamento utilizando a primeira opção.

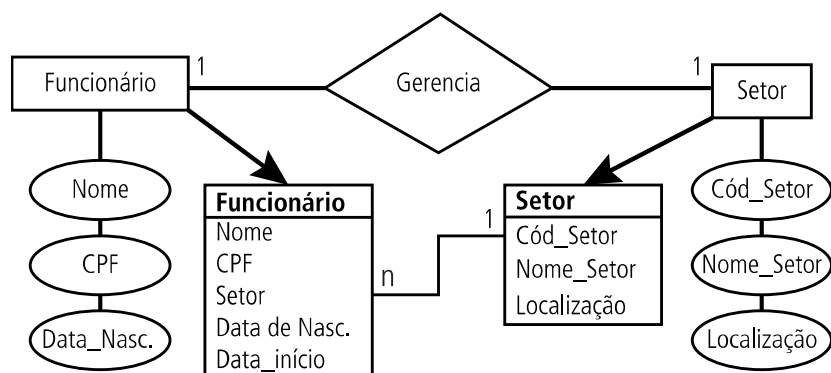


Figura 3.11: Primeira opção de mapeamento para relação 1:1

Fonte: Elaborada pelo autor

De acordo com a primeira opção de mapeamento apresentada na Figura 3.11, foi acrescentada à relação FUNCIONARIO os atributos do relacionamento GERENCIA (Data_Início) e a chave primária da relação SETOR (Cod_Setor) virou uma chave estrangeira da relação FUNCIONARIO (Setor), assim na prática apenas um percentual de tuplas na relação FUNCIONARIO terá um valor no atributo Setor, o restante dos valores dessa chave estrangeira assumirão o valor *null* (nulo), tendo em vista que somente alguns funcionários gerenciarão algum setor.

A segunda opção só seria aplicável se pensarmos no relacionamento de forma isolada, pois consistiria em gerar uma única tabela para as entidades e seu relacionamento como podemos observar na Figura 3.12.

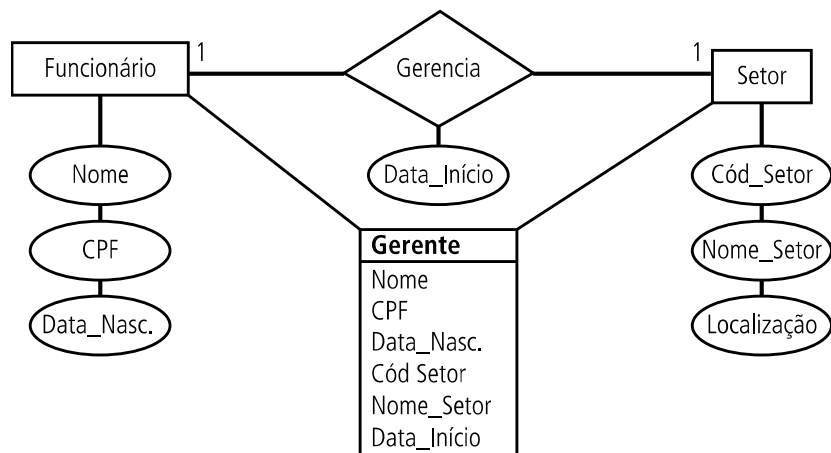


Figura 3.12: Segunda opção de mapeamento para relação 1:1

Fonte: Elaborada pelo autor

Quando pensamos no sistema de forma global, uma boa opção de mapeamento pode ser a terceira, que na verdade como dito anteriormente torna a relação N:M.

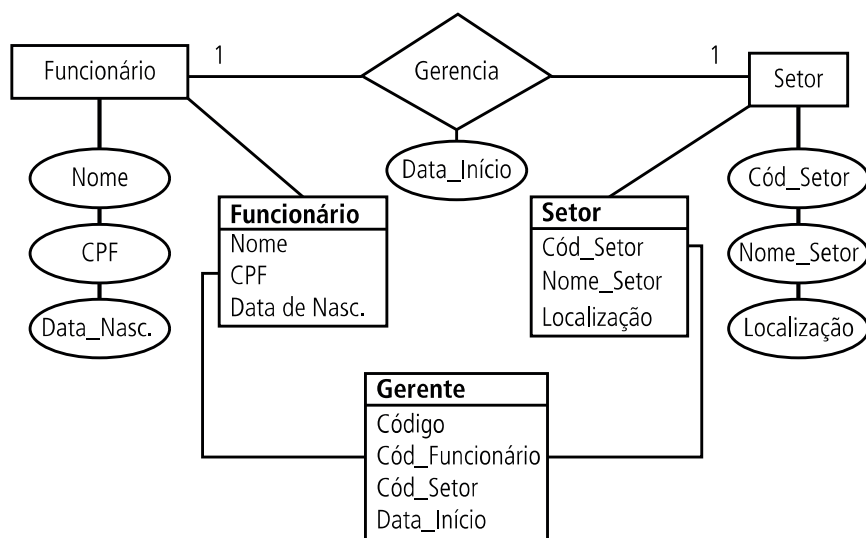


Figura 3.13: Terceira opção de mapeamento para relação 1:1

Fonte: Elaborada pelo autor

Como podemos observar na Figura 3.13, esta opção de mapeamento gerou uma tabela exclusiva para o relacionamento, onde as chaves primárias de FUNCIONARIO (CPF) e SETOR (Cod_Setor) são chaves estrangeiras em GERENTE (Cod_Funcionario e Cod_Setor). A relação GERENTE possui como atributos um código que identifica de forma única suas tuplas e o atributo Data_Inicio do relacionamento que originou a tabela.

Passo 3: Mapeamento dos Conjuntos de Relacionamentos binários de razão de cardinalidade 1:N: Em primeiro, devemos gerar uma tabela para cada um dos conjuntos de entidades conforme descrito em nosso primeiro passo. Para o mapeamento 1:N a relação que mapeia o Conjunto de entidades do lado **N** recebe a chave primária do outro Conjunto de Entidades (lado **1**) como chave estrangeira e os atributos do relacionamento. Vejamos o exemplo na Figura 3.14 a seguir:

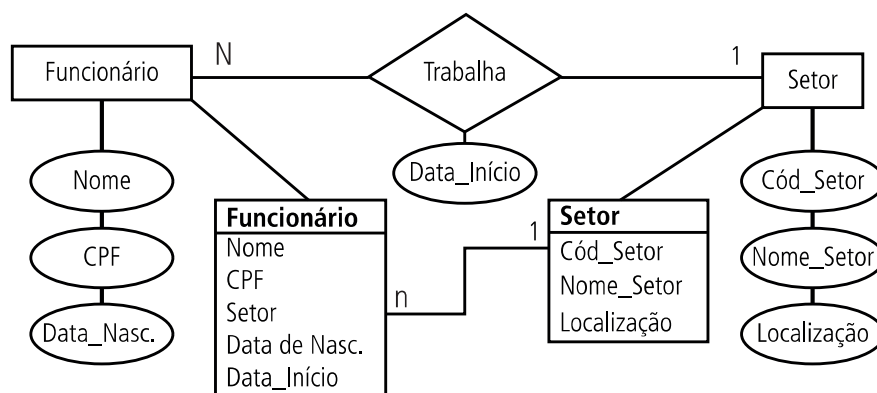


Figura 3.14: Mapeamento para relação 1:N

Fonte: Elaborada pelo autor



Como podemos observar na Figura 3.14, o diagrama ER apresenta as entidades FUNCIONARIO e SETOR com relacionamento 1:N, ou seja, um setor pode possuir vários funcionários. Para o mapeamento deste relacionamento, foi incluído na tabela gerada pela entidade do lado N (FUNCIONARIO) a chave primária da tabela gerado pela entidade do lado 1 (SETOR), desta forma a tabela FUNCIONARIO possui a chave primária de SETOR (Cod_Setor) como sua chave estrangeira (Setor), além de possuir também os atributos da relação TRABALHA (Data_Início).

Desta maneira, poderíamos ter as seguintes tabelas no banco de dados que foi mapeado a partir do diagrama apresentado na Figura 3.15:

FUNCIONARIO				
NOME	CPF	SETOR	DATA_NASC	Data_Inicio
Jose	33875419294	1	10/05/1980	01/01/2000
Carlos	26218188892	2	17/05/1970	01/05/2005

Setor		
Cod_Setor	Nome	Localizacao
1	Informatica	Andar 4
2	RH	Térrio

Figura 3.15: Tabelas geradas pelo mapeamento da relação 1:N

Fonte: Elaborada pelo autor

As tabelas acima mostram qual funcionário trabalha em cada setor. Sabemos então que José trabalha no setor de Informática desde 01/01/2000.

Passo 4: Mapeamento dos Conjuntos de Relacionamentos binários de razão de cardinalidade N:M: Devemos criar uma nova tabela para representar o relacionamento. Nesta nova tabela devemos incluir como chave-estrangeira as chaves-primárias das tabelas que representam os tipos de entidade participantes; sua combinação irá formar a chave-primária desta nova tabela. Vejamos um exemplo na Figura 3.16.

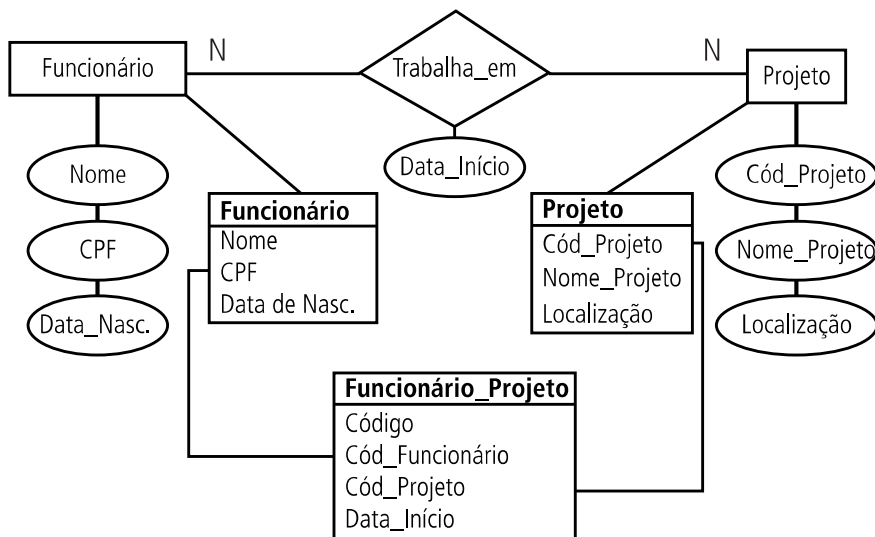


Figura 3.16: Exemplo mapeamento da relação N:M

Fonte: Elaborada pelo autor

Nosso exemplo apresentado na Figura 3.16 acima representa uma relação onde um PROJETO pode ter vários funcionários trabalhando e um funcionário pode trabalhar em vários projetos. No mapeamento deste relacionamento foi criada a relação FUNCIONARIO_PROJETO. Esta relação contém os atributos do relacionamento (no caso Data_Início) e as chaves primárias das relações FUNCIONARIO e PROJETO (Cod_Funcionario e Cod_Projeto) como chave estrangeira.

Vejamos um conjunto de tabelas de um banco de dados que foi mapeado a partir do diagrama apresentado na Figura 3.17 a seguir.

FUNCIONARIO		
NOME	CPF	DATA_NASC
Jose	33875419294	10/05/1980
Carlos	26218188892	17/05/1970

PROJETO		
Cod_Projeto	Nome_Projeto	Localizacao
1	Informatica para todos	Sala 2
2	Melhoria de trabalho	Sala 1

FUNCIONARIO_PROJETO			
Codigo	Cod_Projeto	Cod_Funcionario	Data_Inicio
1	1	1	01/01/2000
2	1	2	01/05/2005
3	2	2	01/05/2005
4	2	1	01/01/2000

Figura 3.17: Tabelas geradas pelo mapeamento da relação N:M

Fonte: Elaborada pelo autor

Na Figura 3.17 podemos observar que na tabela FUNCIONARIO_PROJETO podemos saber qual funcionário trabalha em qual projeto. Por exemplo, no projeto Informática para todos trabalha os funcionários José e Carlos. Desta maneira podemos ver que um projeto pode ter vários funcionários, assim como um funcionário pode fazer parte de vários projetos. Como chave primária da tabela FUNCIONARIO_PROJETO criamos um novo campo Codigo, poderíamos ainda ter identificado esta tabela através de uma chave composta pelas duas chaves estrangeiras da relação (Cod_Funcionario e Cod_Projeto).

Passo 5: Mapeamento de Relacionamentos N-ários: Para os relacionamentos de grau maior que dois devemos criar uma nova relação (tabela) para representar o relacionamento. Temos que incluir nesta tabela como chave-estrangeiras as chaves-primárias das relações que representam os tipos de entidades participantes. Temos que incluir também qualquer atributo simples do tipo de relacionamento n-ário. A chave-primária desta relação é normalmente uma combinação de todas as chaves-estrangeiras fazendo referência às relações que representam os tipos de entidades do relacionamento.

Resumo

Nesta aula nos aprofundamos no Modelo de Dados Relacional, vimos seus conceitos fundamentais e como um projeto de um esquema conceitual do modelo ER pode ser mapeado para um banco de dados relacional. Vimos também conceitos importantes como o de chave estrangeira, domínio e integridade referencial. Foi ilustrado por meio de exemplos os passos necessários para o mapeamento entre MER x MRel.