

Fundação Educacional de Fernandópolis

## Apostila de Banco de Dados

Prof. Evandro A. Jardini

1 de fevereiro de 2006

# Sumário

<b>1</b>	<b>Introdução</b>	<b>4</b>
1.1	Sistema de Processamento de Arquivos . . . . .	4
1.2	Independência de Dados . . . . .	5
1.3	Sistema Gerenciador de Banco de Dados (SGBD) . . . . .	6
1.3.1	Características de um SGBD . . . . .	6
1.4	Motivação para SGBDs . . . . .	7
1.4.1	Linguagens de acesso a um SGBD . . . . .	8
1.4.2	Arquitetura de um SGBD . . . . .	8
1.5	Exercícios . . . . .	9
1.6	Bibliografia . . . . .	9
<b>2</b>	<b>O Modelo Entidade-Relacionamento</b>	<b>10</b>
2.1	Entidades . . . . .	10
2.2	Atributos . . . . .	11
2.2.1	Chave Primária . . . . .	12
2.2.2	Atributos Multivalorados . . . . .	12
2.2.3	Atributos Compostos . . . . .	13
2.2.4	Atributo Derivado . . . . .	13
2.3	Entidades Fracas . . . . .	14
2.4	Relacionamentos . . . . .	14
2.4.1	Atributos de Relacionamento . . . . .	15
2.4.2	Cardinalidade dos Relacionamentos . . . . .	16
2.4.3	Grau dos Relacionamentos . . . . .	16
2.4.3.1	Relacionamentos Binários . . . . .	16
2.4.4	Relacionamentos Ternários . . . . .	16
2.4.5	Auto-Relacionamentos . . . . .	18
2.5	MER Estendido . . . . .	18
2.5.1	Generalização e Especialização . . . . .	18
2.5.1.1	Relacionamentos entre Entidades Especializadas . . . . .	20
2.5.1.2	Multiplas Especializações . . . . .	20
2.5.1.3	Restrições da Abstração de Generalização . . . . .	20
2.5.2	Agregação . . . . .	22
2.5.2.1	1 Caso do uso de Agregação (Opcionalidade) . . . . .	23
2.5.2.2	2 Caso do uso de Agregação (Identificador em Relacionamento) . . . . .	23
2.5.2.3	3 Caso do uso de Agregação (indica tempo) . . . . .	23
2.6	Exercícios . . . . .	23
2.6.1	Primeira Lista . . . . .	23
2.7	Bibliografia . . . . .	27

<b>3</b>	<b>O Modelo Relacional</b>	<b>28</b>
3.1	Introdução . . . . .	28
3.2	Domínios . . . . .	29
3.3	Relações . . . . .	29
3.4	Chave no Modelo Relacional . . . . .	30
3.4.1	Superchave . . . . .	30
3.4.2	Chave . . . . .	30
3.4.3	Chave Candidata . . . . .	31
3.4.4	Restrições de Integridade . . . . .	31
3.4.4.1	Chave Estrangeira (Integridade Referencial) . . . . .	31
3.4.4.2	Exemplos de Restrições de Integridade . . . . .	32
3.5	Outros tipos de Restrições . . . . .	33
3.6	Bibliografia . . . . .	33
<b>4</b>	<b>Mapeamento do Modelo ER para Modelo Relacional</b>	<b>34</b>
4.1	Introdução . . . . .	34
4.2	Mapeando Entidades . . . . .	34
4.3	Entidades Fracas . . . . .	35
4.4	Relacionamento Binário com Cardinalidade 1:1 . . . . .	35
4.5	Relacionamento Binário com Cardinalidade 1:N . . . . .	35
4.6	Relacionamento Binário com Cardinalidade M:N . . . . .	36
4.7	Relacionamentos Ternários . . . . .	36
4.8	Exemplo 1 . . . . .	37
4.9	Atributos Compostos e Multivalorados . . . . .	38
4.10	Generalização . . . . .	39
4.11	Agregação . . . . .	39
4.12	Exemplo 2 . . . . .	40
4.13	Resumo dos 6 passos . . . . .	40
4.14	Bibliografia . . . . .	41
<b>5</b>	<b>Normalização de Relações</b>	<b>42</b>
5.1	Introdução . . . . .	42
5.2	Normalização de Relações . . . . .	42
5.3	Anomalias de Modificações . . . . .	42
5.4	Primeira Forma Normal . . . . .	43
5.5	Dependência Funcional . . . . .	44
5.6	Segunda Forma Normal (2FN) . . . . .	44
5.7	Terceira Forma Normal (3FN) . . . . .	45
5.8	Exemplo Prático . . . . .	46
5.9	Exercícios . . . . .	48
<b>6</b>	<b>Álgebra Relacional</b>	<b>49</b>
6.1	Introdução . . . . .	49
6.2	Operação Selecionar ( $\sigma$ ) . . . . .	50
6.3	A Operação Projetar ( $\Pi$ ) . . . . .	51
6.4	A Operação União ( $\cup$ ) . . . . .	52
6.5	A Operação Intersecção de Conjuntos ( $\cap$ ) . . . . .	52
6.6	A Operação Diferença de Conjuntos ( $-$ ) . . . . .	53
6.7	A Operação Produto Cartesiano ( $\times$ ) . . . . .	53
6.8	A Operação Junção Natural ( $\bowtie$ ) . . . . .	54

6.9 A Operação Divisão ( $\div$ ) . . . . .	55
6.10 Exercícios . . . . .	56

# Capítulo 1

## Introdução

Um *Sistema Gerenciador de Banco de Dados* (SGBD) é constituído por um conjunto de dados associados a um conjunto de programas para acesso a esses dados. O conjunto de dados, comumente chamado de **banco de dados**, contém informações sobre uma empresa em particular. O principal objetivo de um SGBD é proporcionar um ambiente tanto conveniente quanto eficiente para a recuperação e armazenamento das informações do banco de dados.

SGBDs são projetados para gerir grandes volumes de informações. Devem possuir mecanismos para definição e manipulação de dados, além de prover compartilhamento e segurança dos mesmos.

### 1.1 Sistema de Processamento de Arquivos

Como exemplo, considere a área de um banco responsável por todas as informações de seus clientes e de suas contas-poupança. Um modo de guardar as informações no computador é armazená-las em sistemas de arquivos permanentes. Para o acesso aos dados foram desenvolvidos os seguintes módulos de programas:

- Programa para débito e crédito na contabilidade.
- Programa para incluir novos registros na contabilidade.
- Programa parabalanco da contabilidade.

A medida que surge na necessidade, outros módulos são desenvolvidos.

Considere que os programas foram desenvolvidos em Pascal (poderiam ter sido desenvolvidos em Clipper, Cobol, Basic, etc) e que a estrutura do registro de cliente é:

```
type Cliente = record
  nome:string;
  rua:string;
  cidade:string;
  cep;
end;
```

Neste *Sistema de Processamento de Arquivos* que acabamos de descrever, os registros são armazenados em vários arquivos e diversos programas são escritos para extrair e gravar estes registros.

Obter informações organizacionais em sistemas de processamento de arquivos apresenta numerosas desvantagens:

- **Redundância e inconsistência de dados:** Redundância é a característica onde um elemento de informação aparece duplicado em diversos lugares. Por exemplo, um cliente de uma empresa pode estar cadastrado tanto na base de dados do departamento de crédito assim como no departamento de contabilidade. Esta redundância pode causar a inconsistência, visto que as informações podem ser diferentes para o mesmo cliente.
- **Dificuldade no acesso aos dados:** Suponha que um diretor necessite encontrar os clientes que residem em uma área da cidade onde o CEP seja 12133-433. Se este módulo do sistema de informação não estiver implementado, o diretor terá duas opções: ele pega a lista de clientes e extrai a informação necessária, ou pede para o departamento de informática implementar a rotina necessária para obtenção do relatório.
- **Isolamento de dados:** Uma vez que os dados estão espalhados em diversos arquivos e podem ter formatos diferentes, é difícil escrever novos programas aplicativos para recuperar os dados adequados.
- **Anomalia de acesso concorrente:** Evitar problemas relacionados a acesso concorrentes em um dado. Por exemplo, se duas pessoas sacarem dinheiro de uma conta ao mesmo tempo, o resultado das operações concorrentes pode deixar um saldo inconsistente. Considere uma conta com a quantia de \$400 e dois saques simultâneos de \$100 e \$50. Se houver problema de concorrência, a conta pode ficar com valores de \$300 ou de \$350 ao invés de \$250.
- **Segurança:** Cada usuário do sistema poderá ter acesso aos dados relativos a sua função. Por exemplo, um caixa não pode ver dados pertencentes à diretoria.
- **Problemas de Integridade:** Os valores armazenados no BD devem satisfazer certos tipos de restrições de consistência. Por exemplo, um filho de um funcionário não pode ser cadastrado sem que o pai trabalhe na empresa.

## 1.2 Independência de Dados

O desenho a seguir representa um sistema de processamento de arquivo típico:

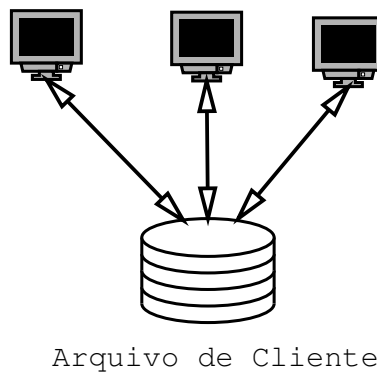


Figura 1.1: Sistema de Processamento de Arquivo

**De** acordo com a figura 1.1, nota-se que os programas gravam seus dados em disco, seguindo estruturas próprias. Para acessá-los é necessário conhecer sua estrutura.

**Se** vários programas compartilham seus dados, todos devem conhecer e manipular as mesmas estruturas.

**Se** algum programa precisar de alguma mudança de dados, todos os programas terão que ser alterados, mesmo que a alteração ocorra em dados que ele não utiliza.

Se entre os programas e os dados for colocado um sistema que converte o formato em que os dados estão gravados para o formato específico que cada programa precisa dos dados, então cada programa:

- Vê apenas os dados que lhe interessa;
- Não precisa entrar em detalhes de como seus dados estão fisicamente gravados;
- Não precisa ser modificado se a estrutura de dados que ele não utiliza for mudada.

É feita uma conversão específica dos dados gravados para a forma usada em cada programa. Alterar os dados obriga a alterar a forma de conversão para cada programa, mas não cada programa em si. As alterações ficam concentradas nesse sistema intermediário.

Esse sistema intermediário é chamado de Sistema Gerenciador de Banco de Dados (Figura 1.2).

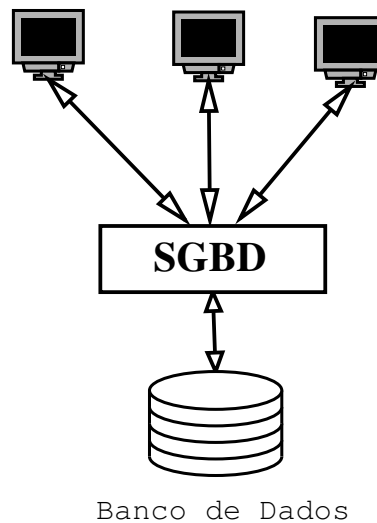


Figura 1.2: Independência de Dados provida pelo SGBD

## 1.3 Sistema Gerenciador de Banco de Dados (SGBD)

**Como** já foi dito, um SGBD e um conjunto de programas com a finalidade de manipular um conjunto de dados.

**Com** a introdução de um SGBD em uma organização demanda o surgimento de um novo profissional, o **Administrador de Banco de Dados (DBA)**.

**O** DBA é responsável pela manutenção do SGBD e da Base de Dados em si, centralizando todo o gerenciamento sobre a estrutura dos dados da empresa.

### 1.3.1 Características de um SGBD

**Segue** as características que um SGBD tem de ter:

- **CONTROLE DE REDUNDÂNCIA:** A redundância, ou seja, a repetição de dados, deve ser evitada para se minimizar possibilidade de inconsistências.
- **COMPARTILHAMENTO DE DADOS:** Em um ambiente multi-usuários deve-se possibilitar a manipulação simultânea de dados distintos ou dos mesmos dados conforme regras abaixo.

- **CONTROLE DE ACESSO:** Verificação automática do tipo de acesso pedido por cada usuário. Os níveis de segurança são estabelecidos para cada usuário independentemente, de acordo com suas necessidades. A identificação de cada usuário, por parte do SGBD, é feita pelo nome e senha cadastrados.
- **CONTROLE DE TRANSAÇÃO:** Transação é o conjunto de operações que devem ser executadas completamente. São normalmente usadas em situações críticas (atualizações ou inclusões) de longa duração que podem afetar a consistência do BD. Exemplo: bug milênio, corte dos 3 zeros, aumento geral dos produtos, etc. O SGBD deve utilizar mecanismos internos para que nenhuma falha ocorra durante a execução da transação.
- **ACESSO EM MÚLTIPLAS INTERFACES:** Possibilidade de usar diversas interfaces mesmo se o SGBD estiver sendo utilizado. Por exemplo: Se existe uma aplicação em Delphi com o SGBD Interbase, se trocar a linguagem para Visual Basic, não é necessário fazer alterações no SGBD.
- **RESTRICÇÕES DE INTEGRIDADE:** Estabelecimento de um formato para os dados inseridos de modo a garantir uma certa integridade e facilitar o armazenamento. Ex. Tamanho do nome emprega até 30 e em maiúsculo, armazenamento dos salários em dólar, etc. Algumas regras de integridade são estabelecidas pelo próprio SGBD para manter o BD consistente e outras são definidas pelo DBA por meio de sentenças condicionais que são verificadas toda vez que um dado é armazenado no BD.
- **BACKUP E RECUPERAÇÃO:** Estabelecer o backup automático do BD total ou parcial em momentos estabelecidos pelo DBA. Proporcionar proteção contra a perda de informações devido a falhas no dispositivo de armazenamento (discos).
- **INDEPENDÊNCIA DE DADOS:** A descrição física dos arquivos é mantida internamente pelo SGBD e é de sua inteira responsabilidade e exclusividade. Programas aplicativos não dispõem da descrição física e sim de uma descrição externa. Alterações nos arquivos podem não afetar os programas aplicativos.
- **INDEXAÇÃO AUTOMÁTICA:** Com a indicação explícita dos atributos que serão mais utilizados em consultas, o SGBD cria os arquivos de indexação que tornarão mais rápidas as pesquisas. A estrutura de indexação e de organização dos arquivos de dados é própria de cada SGBD e normalmente não é de domínio dos usuários comuns.

## 1.4 Motivação para SGBDs

A utilização dos SGBDs traz os seguintes problemas:

- É um produto caro;
- É um sistema a mais a ser aprendido e gerenciado;
- Ocupa espaço de armazenagem no computador.

**Porém** traz vantagens muito maiores:

### **CONSISTÊNCIA DE DADOS E INDEPENDÊNCIA DE DADOS**

**Além** disso,

- O SGBD é a ferramenta por excelência para promover a integração dos diversos componentes de um sistema de software;

- Concentram o maior potencial para promover acesso compartilhado de informação, sem bloquear desnecessariamente o acesso compartilhado;
- retiram dos programas aplicativos muita da complexidade de gerenciamento de estruturas de acesso aos dados;
- Facilitam a proteção contra a perda de dados, através de recursos de backup;
- Promovem a adoção de padrões para toda a empresa, facilitando seu emprego.

### 1.4.1 Linguagens de acesso a um SGBD

**Existe** uma linguagem específica para definir o esquema conceitual e físico de um SGBD. A linguagem utilizada é a Structured Query Language (SQL), que apesar de existir um padrão definido, varia de SGBD pra SGBD.

A SQL pode ser dividida em 3 grupos:

- Linguagem de Definição de Dados (DDL): Responsável por criar e manipular os “objetos” de armazenamento de dados no SGBD.
- Linguagem de Manipulação de Dados (DML): Responsável por manipular os dados do SGBD.
- Linguagem de Controle dos Dados (DCL): Responsável por permitir o acesso de usuários ao SGBD.

**Os** aplicativos são escritos em uma linguagem de programação, que pode ser: DELPHI, VISUAL BASIC, COBOL, C, PASCAL, ETC. **A SQL é utilizada de forma embutida nestas linguagens.**

A linguagem SQL engloba numa única linguagem todos os recursos necessários para a manipulação da Base de Dados.

### 1.4.2 Arquitetura de um SGBD

A arquitetura de um SGBD se divide em três níveis (Figura 1.3):

- **Nível Interno ou Físico:** é o mais próximo do meio de armazenamento físico, ou seja, é aquele que se ocupa do modo como os dados são fisicamente armazenados.
- **Nível Conceitual ou Lógico:** Descreve quais dados estão armazenados no banco de dados e quais os inter-relacionamentos entre eles. Este nível é utilizado pelos administradores.
- **Nível externo:** é o mais próximo dos usuários, ou seja, é aquele que se ocupa do modo como os dados são vistos por usuários individuais.

**Exemplos** para compreensão dos níveis:

- **No nível conceitual:** O banco de dados contém informações relativas a um tipo de entidade chamada EMPREGADO. Cada empregado contém um NÚMERO\_EMPREGADO, um NÚMERO\_DEPARTAMENTO e um SALÁRIO.
- **No nível interno:** Os empregados são representados por um tipo de registro armazenado, denominado EMP\_ARMAZENADO, com 20 bytes de comprimento. Contém 4 campos: um prefixo de 6 bytes mais 3 campos de informação de empregado. Além disso os registros são indexados sobre o campo EMP.
- **No nível externo:** O usuário SECRETÁRIA tem uma visão na qual cada empregado tem 2 campos: Nome e Salário. Já o usuário CONTADOR tem uma visão que cada empregado tem os campos Nome e Salário.

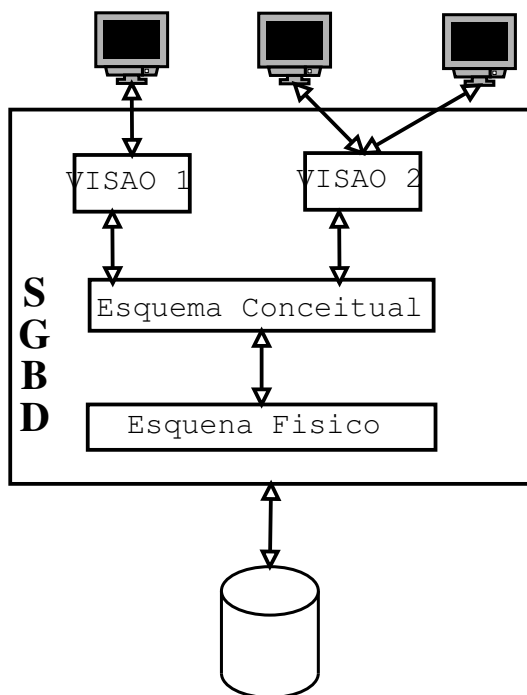


Figura 1.3: Arquitetura de SGBD

## 1.5 Exercícios

1. Resolver em grupo as questões 1.2, 1.4, 1.7, 1.8, 1.9, 1.10, 1.17 e 1.23 do capítulo 1 da bibliografia 2.  
*Atenção: Este capítulo encontra-se no Xerox. 14 páginas.*
2. Resolver em grupo as questões 2.3, 2.5, 2.7, 2.8 do capítulo 2 da bibliografia 3.

## 1.6 Bibliografia

- [1] Traina Jr, Caetano, *Modelagem de Dados*, Apostila, ICMC-USP.
- [2] Kroenke, David M., *Banco de Dados: Fundamentos, Projeto e Implementação*, Editora LTC, 6 ed, Capítulo 1, Rio de Janeiro, 1999
- [3] Date, C. J., *Introdução a Sistemas de Banco de Dados*, Editora Campus, Ed. 7, Capítulo 2, Rio de Janeiro, 2000.

# Capítulo 2

## O Modelo Entidade-Relacionamento

**Este** capítulo descreve e ilustra o uso do **Modelo Entidade-Relacionamento (MER)**, que foi apresentado por Peter Cher em 1976.

**Hoje** não existe um padrão único de modelo E-R aceito universalmente; em vez disto, há um conjunto de conceitos dos quais se origina a maioria das variantes do E-R.

**No** MER, os elementos que o compõe são representados graficamente através da ferramenta denominada **Diagrama Entidade - Relacionamento (DER)**.

A seguir são descritos os principais elementos que compõe o MER.

### 2.1 Entidades

**Define-se** entidade como aquele objeto que existe no mundo real com uma identificação distinta e com um significado próprio.

**São** as “coisas” que existem no negócio, ou ainda, descrevem o negócio.

**Se** alguma “coisa”, existente no negócio nos proporciona algum interesse em mantermos dados (informações armazenadas sobre ele), esta a caracteriza como uma Entidade do Negócio.

**Alguns** exemplos de entidades:

- O FUNCIONÁRIO João;
- O VEICULO Corsa;
- A ALUNA Maria;
- O CLIENTE Pedro;
- O PRODUTO A323....

**Entidades** de um mesmo tipo são agrupadas em **Classes de Entidade**. Assim, a classe de entidades FUNCIONÁRIOS é o conjunto de todas as instâncias de funcionários. Neste texto, classes de entidades estão impressas em letra maiúscula.

**Cada** ocorrência de um funcionário dentro da classe FUNCIONÁRIO é denominado **Instância de Entidade**.

A representação gráfica de uma entidade no MER se realiza através de um retângulo, com o nome desta entidade em seu interior, como mostra a figura 2.1.

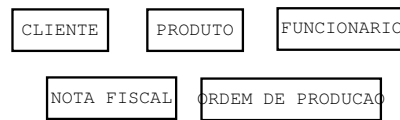


Figura 2.1: Exemplos de Entidades

**Importante:** As instâncias de uma entidade não são representadas no DER.

## 2.2 Atributos

**Toda** entidade possui **propriedade** que são descritas por **Atributos**. No MER supõe que todas as instâncias de uma dada classe de entidade possuem os mesmos atributos.

**Considere** a entidade **FUNCIONÁRIO** uma empresa. *O que descreve Funcionário?*

**Funcionário** é descrito por:

- número de matrícula
- nome
- data da admissão

Como é representado na tabela 2.1.

Tabela 2.1: Entidade Funcionário

Matrícula	Nome	Data Admissão
4455	João	24/04/1991
4456	Pedro	30/02/1992
4457	José	14/04/1992
4458	Manoel	01/01/1995

No DER os atributos *PODEM* ser representados por um **circulo** em torno de seu nome, como mostra a figura 2.2.

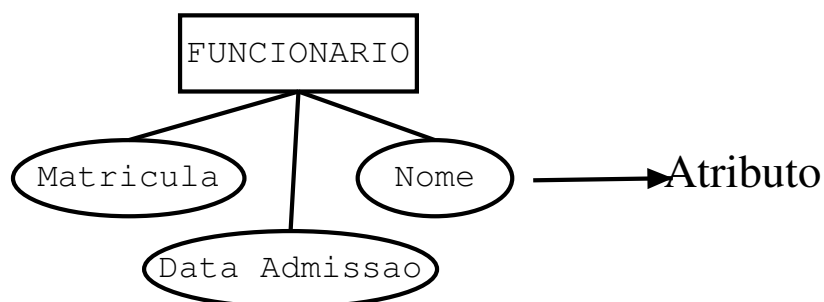


Figura 2.2: Representação de Atributos

### 2.2.1 Chave Primária

**Não** existe **DUAS INSTÂNCIAS DE ENTIDADES IGUAIS**.

**Sempre** haverá atributo (ou atributos) que nunca se repete.

**Este** atributo tem a função de atuar como identificador único das instâncias da entidade e é denominado de **CHAVE PRIMÁRIA**.

**Na** tabela 2.1, a chave primária é o atributo MATRICULA.

**Então**, como a chave primária identifica uma instância da entidade, ela tem duas restrições importantes:

- Não se repete;
- Não contém valor NULO.

**Um** VALOR NULO é um valor que não tem significado algum para o mundo real, somente para o conceitual.

No DER, um atributo chave primária é representado por um traço abaixo de seu nome, como mostra a figura 2.3.

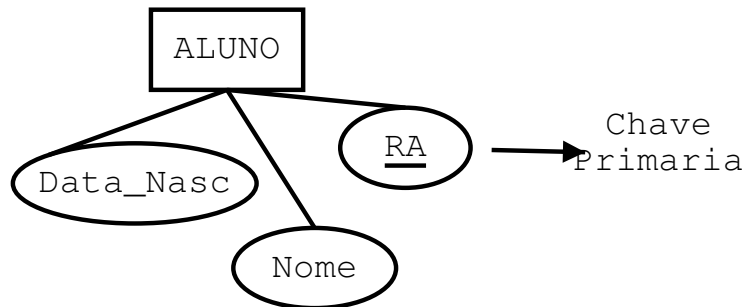


Figura 2.3: Atributo Chave Primária

Além da chave primária também temos:

- Superchave: é o conjunto de um ou mais atributos que, tomado coletivamente, permite-nos identificar unicamente uma entidade no conjunto de entidade.
- Chaves candidatas: chaves com unicidade em uma relação: Ex: CIC, RG, título eleitoral. Todos os atributos que conseguem identificar uma Relação.
- Chave secundária: chave sem unicidade em uma relação. Ex: idade, sexo, endereço.

### 2.2.2 Atributos Multivalorados

São atributos que para cada instância de um entidade, ele pode ocorrer várias vezes. No DER, é representado por duas elipses em torno do nome do atributo.

Ex.: Telefones para clientes ou alunos, Nomes de cidades à beira de uma rodovia, Nomes dos autores de um livro, etc.

A figura 2.4 representa o atributo telefone dos cliente de uma empresa.

A representação em um SGBD para atributos multivalorados é mostrado na tabela 2.2:

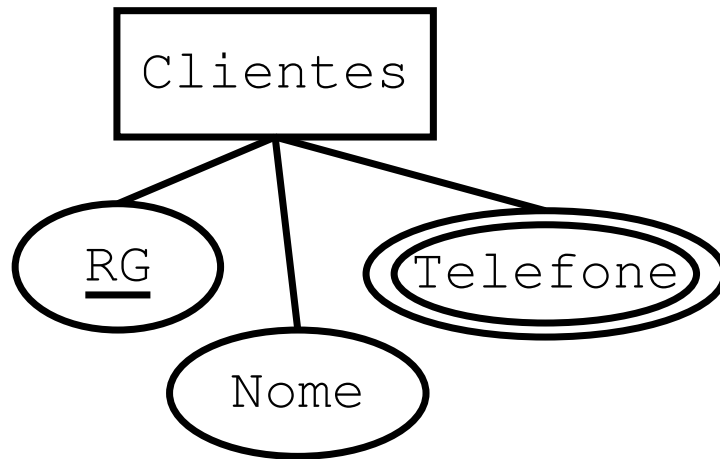


Figura 2.4: Atributos Multivalorados

Tabela 2.2: Representação em um SGBD de atributos multivalorados da entidade cliente

RG	Nome
11	Pedro
16	Afonso

RG_Cliente	Telefone
16	442
16	3244
11	1231

### 2.2.3 Atributos Compostos

São atributos formados por outros atributos.

Ex: Telefones, Enderecos, Nome, area de uma sala, etc

A figura 2.5, representa atributos compostos de uma entidade:

Em um SGBD, os atributos são representados da maneira como mostra a tabela 2.3

Tabela 2.3: Atributo Composto em um SGBD

numero	comprimento	largura	altura
s1	5	3	3
s2	5	2	3
s3	7	3	3

### 2.2.4 Atributo Derivado

São atributos derivados de outros atributos ou originados de algum calculo. Por exemplo, na figura 2.6, têm-se os campos *idade* e *qtidade\_de\_funcion.* O primeiro é calculado a partir da data de nascimento e o segundo é obtido através da contagem dos funcionários. Por esta razão, este atributo não precisa ser armazenado.

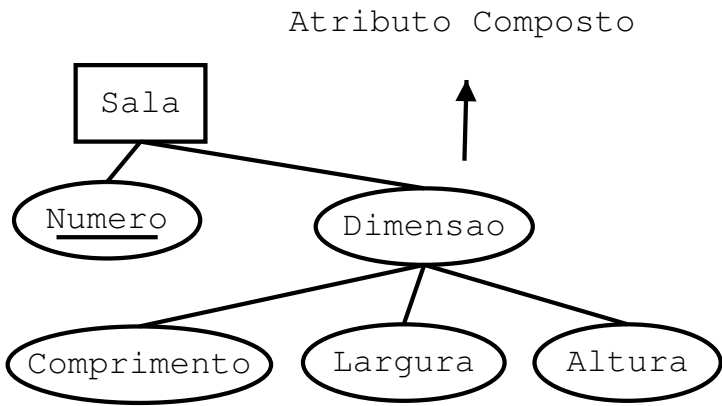


Figura 2.5: Atributos Compostos

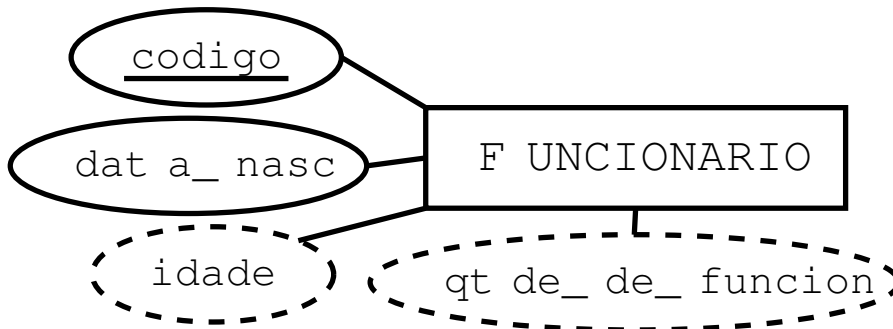


Figura 2.6: Atributo derivado (representado pela linha pontilhada).

### 2.3 Entidades Fracas

**Pode** ocorrer que alguma instância de entidade possua uma dependência existencial com outra instância de entidade e neste caso cada instância daquela Entidade existe somente porque está associada a outra instância de entidade diferente.

**Dizer** que uma Entidade é fraca, significa dizer: **QUE NÃO INTERESSA MANTER NA BASE OS DADOS DE UMA ENTIDADE SE ELA NÃO ESTIVER RELACIONADA COM OUTRA ENTIDADE.**

**Este** tipo de Entidade é denominada **Entidade Fraca**. Exemplo: Dependente é um tipo de Entidade Fraca pois existe somente se existir o funcionário.

A figura 2.7 representa o DER da entidade fraca:

Considerando a entidade Funcionario e seus Dependentes, a tabela 2.4 representa as instâncias de cada entidade:

### 2.4 Relacionamentos

Nenhuma informação armazenada no Banco de Dados existe isoladamente.

Todos os elementos pertencentes ao mundo real modelado de alguma forma está associado a outros elementos. Normalmente essas associações representam ações físicas ou alguma forma de dependência entre os elementos envolvidos.

**Relacionamento:** é a associação entre Entidades.

No DER, os relacionamentos são representados conforme mostra a figura 2.8.

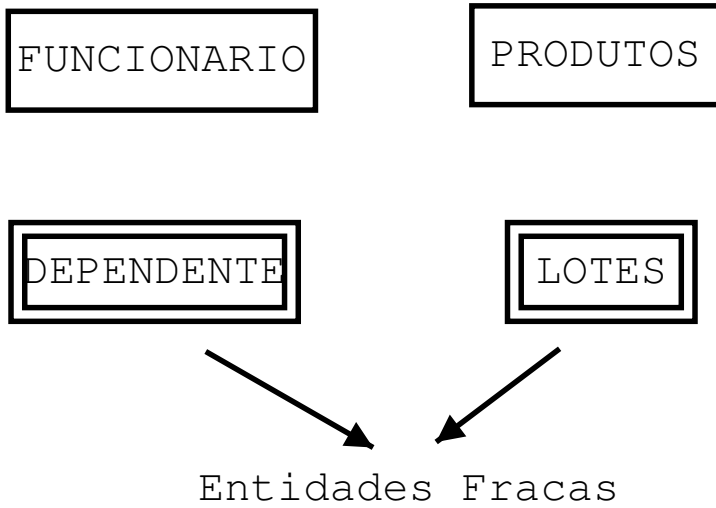


Figura 2.7: Entidade Fraca

Tabela 2.4: Representação das entidades Funcionários e Dependentes

Matrícula	Nome	Data Admissão
4455	João	24/04/1991
4456	Pedro	30/02/1992
4457	José	14/04/1992
4458	Manoel	01/01/1995

Matricula_Funcionario	Codigo_Depend	Nome
4456	01	Pedro Junior
4458	01	Marcelo
4456	02	Henrique

Agora que já temos as definições de Entidades e de Relacionamento, vamos aprender como encontrá-los em um problema:

*Cliente faz empréstimos*

**Dest**a frase, o que é Entidade e o que é relacionamento?

**Pode-se** dizer que os SUBSTANTIVOS são as Entidades e os VERBOS são os Relacioanamentos. Sendo assim tem-se:

- Entidades: Cliente e Empréstimo.
- Relacionamento: Faz

A figura 2.9 representa graficamente a modelagem deste enunciado.

### 2.4.1 Atributos de Relacionamento

Considere a figura 2.10.

Atributos de Relacionamentos são igualmente representados como elipses, ligadas aos conjuntos de Relacionamento



Figura 2.8: Representação gráfica de um relacionamento

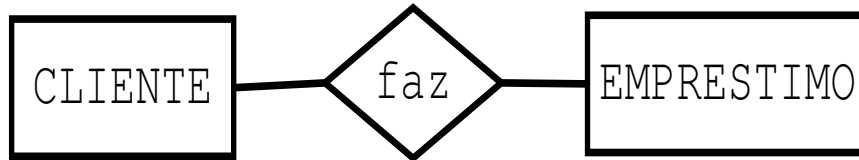


Figura 2.9: Identificação de Entidade e Relacionamento

**Perceba** que *Nota* é um atributo tipicamente do relacionamento *Cursa*.

**Se** fosse um atributo de Pessoa, cada pessoa teria apenas uma nota, não importa em qual disciplina.

**Se** fosse um atributo de Disciplina, todas as Pessoas matriculadas numa disciplina teriam a mesma nota.

**Outro** exemplo: *A quantidade de Material fornecidos por um Fornecedor.*

## 2.4.2 Cardinalidade dos Relacionamentos

A quantidade de Entidades envolvidas em um Relacionamento é determinado pela **Cardinalidade** do Tipo de Relacionamento, ou seja, pode-se estabelecer a quantidade mínima e máxima de Entidades envolvidas com cada Entidade relacionada.

A Cardinalidade Mínima que determina a quantidade mínima de Entidades relacionadas é determinada pelo número representativo, ou seja, 0 (zero), 1, 2,... N(muitos).

A Cardinalidade Máxima que determina a quantidade máxima de Entidades relacionadas é determinada pelo número representativo, ou seja, 0 (zero), 1, 2,... N (muitos).

A figura 2.11 demonstra os tipos de *Cardinalidades Máximas* que se tem para os relacionamentos Binários.

## 2.4.3 Grau dos Relacionamentos

**Um** Relacionamento pode envolver duas ou mais Entidades.

**O Grau** do Relacionamento é o número de Entidades envolvidas.

**Desta** forma pode-se categorizar os tipos de relacionamento em:

### 2.4.3.1 Relacionamentos Binários

Relacionamento que envolve duas Entidades. Figura 2.12.

### 2.4.4 Relacionamentos Ternários

Relacionamento que envolve duas Entidades. Figura 2.13.

Colocar quando usar o relacionamento ternário. Exemplo : fornecedor, peca, e projeto (tudo M:N).

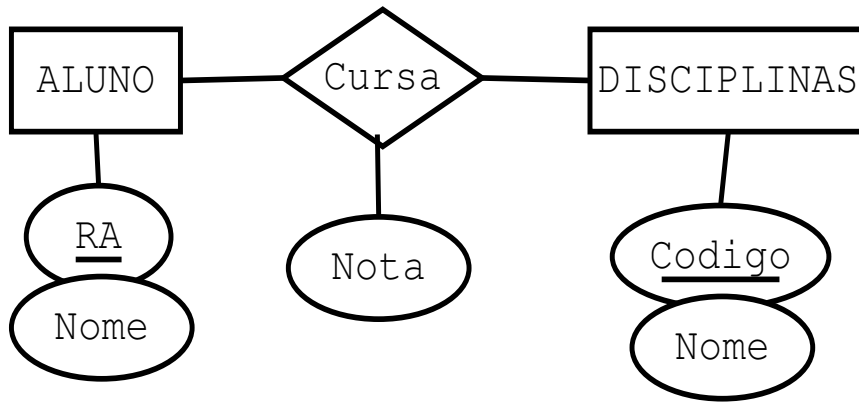


Figura 2.10: Atributo de Relacionamento

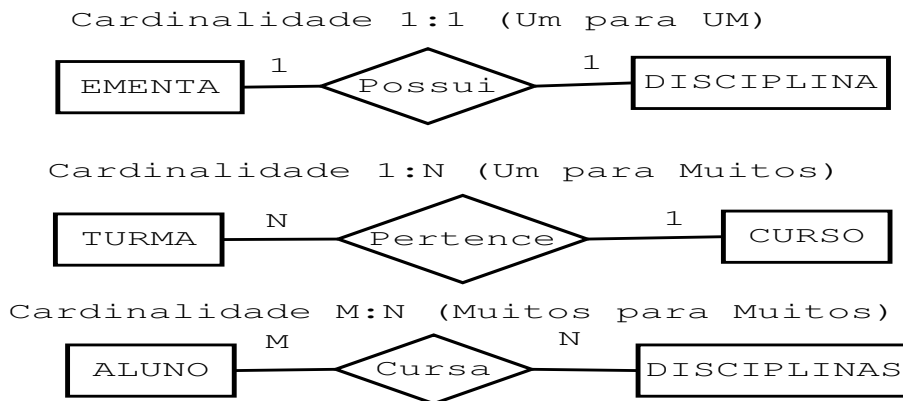


Figura 2.11: Tipos de Cardinalidade

### Como Determinar as Cardinalidades de um Relacionamento Ternário

Para determinar as Cardinalidades, por exemplo de ALUNO-PROFESSOR-DISCIPLINA, faça o seguinte:

1. Escolha uma Entidade, por exemplo ALUNO, e pergunte: Para cada Aluno, quantos pares Professor-Disciplina eu tenho.
2. Coloque a resposta na Entidade ALUNO. Neste caso N. Isto é, um Professor lecionando uma Disciplina pode ter vários Alunos.
3. Escolhendo a Entidade PROFESSOR. Pergunta-se: Para cada Professor, quantos pares Aluno-Disciplina eu tenho.
4. Coloque a resposta na Entidade PROFESSOR. Neste caso 1. Isto é, um Aluno não pode ter em uma certa Disciplina mais do que um Professor.
5. Escolhendo a Entidade DISCIPLINA. Pergunta-se: Para cada Disciplina, quantos pares Aluno-Professor eu tenho.
6. Coloque a resposta na Entidade Disciplina. Neste caso N. Isto é, um Professor pode dar a um certo Aluno mais do que uma Disciplina.

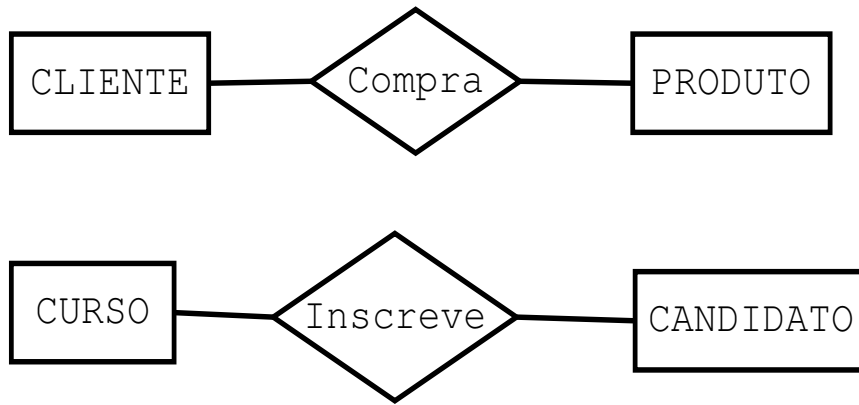


Figura 2.12: Exemplo de Relacionamento Binário

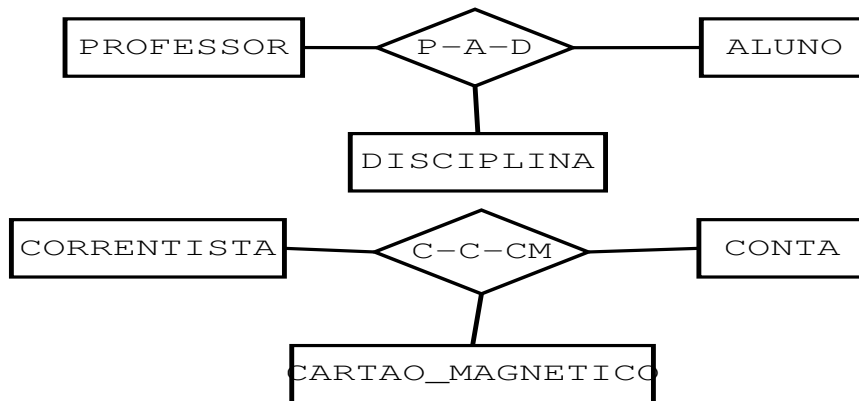


Figura 2.13: Exemplo de Relacionamento Ternário

### 2.4.5 Auto-Relacionamentos

Uma instância ou conjunto de instância de uma mesma Entidade, pode se relacionar com outra(s) instância(s) da mesma Entidade.

Na figura 2.14 temos os seguintes exemplo:

- *Disciplina é pré-requisito de Disciplina*
- *Funcionário gerencia Funcionário*

## 2.5 MER Estendido

Com o passar do tempo, percebeu-se que o MER original, criado por Peter Chen, não modelava alguns tipos de problemas. Surgiu então, uma extensão do MER denominada MER Estendido ou MER-RX.

### 2.5.1 Generalização e Especificação

Algumas Entidades contêm conjunto de atributos específicos. Quando ocorre a situação em que uma entidade possuir atributos que não fazem parte de todas as instâncias da entidade ou quando estas instâncias se relacionarem de maneira diferente com outras entidades, temos aí o conceito de GENERALIZAÇÃO/ ESPECIALIZAÇÃO.

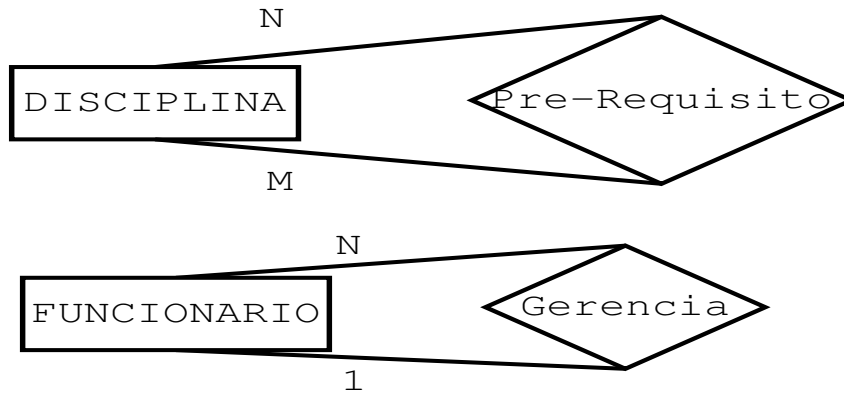


Figura 2.14: Exemplo de Auto-Relacionamentos

**Considere** a entidade *CLIENTE* com os atributos *númeroCliente*, *NomeCliente* e *QuantiaDevida*.

**Suponha** que um *CLIENTE* pode ser uma única pessoa individual, uma associação ou uma coporação e que serão armazenados dados adicionais dependendo do tipo.

**Suponha** ainda que estes dados adicionais são:

- *CLIENTE-INDIVIDUAL*: Endereco, NumeroPrevidenciaSocial
- *CLIENTE-ASSOCIACAO*: NomeAssociado, Endereco, Taxa
- *CLIENTE-CORPORACAO*: PessoaContato, Fone, NumeroIdentificacao

**Para** modelar esta situação demos 2 alternativas:

1. Alocar todos estes atributos na entidade *CLIENTE*. Neste caso, alguns dos atributos não são aplicáveis a todas as entidades.
2. Definir 3 entidades para cada um dos tipos. As quais seriam: *CLIENTE-INDIVIDUAL*, *CLIENTE-ASSOCIACAO* e *CLIENTE-CORPORACAO*.

**Outros** exemplos:

- *FUNCIONARIO*: CODIGO, NOME, ENDERECO
  - *SECRETARIA*: cursos, linguas
  - *ENGENHEIRO*: crea, especialidade
- *VEICULO*: chassis, marca
  - *UTILITARIO*: capacidade\_lugares
  - *TRANSPORTE*: Tara

A figura 2.15 demonstra esta situação do cliente. Uma vez que a entidade *CLIENTE* é uma entidade genérica, ela é denominada de **GENERALIZAÇÃO** (ou entidade de nível superior) e as entidades *INDIVIDUAL*, *ASSOCIADO* e *CORPORAÇÃO* são denominadas **ESPECIALIZAÇÃO** (ou entidade de nível inferior).

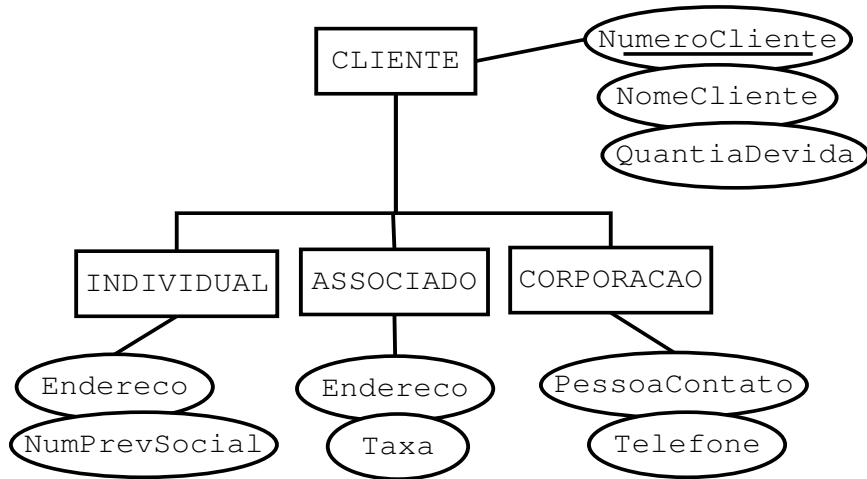


Figura 2.15: Generalização e Especialização

### 2.5.1.1 Relacionamentos entre Entidades Especializadas

Entre as especializações pode haver relacionamento. Considere a figura 2.16, ela demonstra o relacionamento que existe entre as especializações PROFESSOR e ALUNO.

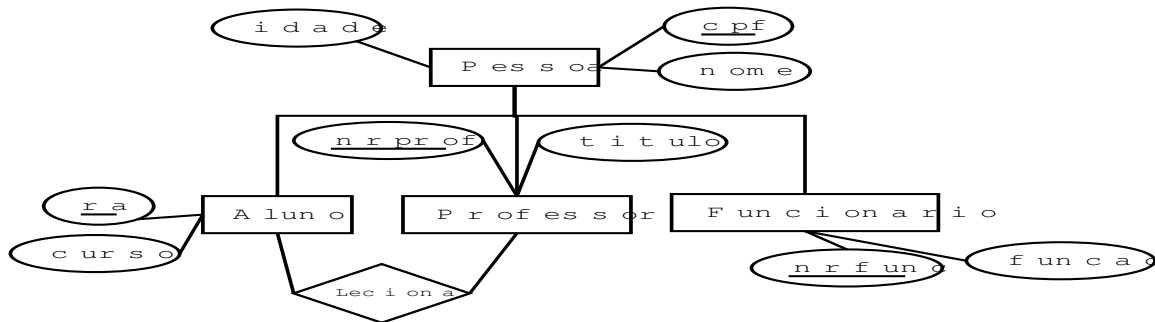


Figura 2.16: Relacionamento entre Especializações

### 2.5.1.2 Múltiplas Especializações

As especializações podem se especializar em outras especializações, isto ocorre na figura 2.17.

### 2.5.1.3 Restrições da Abstração de Generalização

Existem duas restrições que devem ser definidas para cada Ocorrência de Abstrações de Generalização:

- **Exclusão Mútua/Sobrepostos:** Onde **EXCLUSÃO MÚTUA** exige que uma entidade de nível superior pode pertencer a apenas um conjunto de nível entidades de nível inferior. No exemplo da figura 2.15 uma cliente só pode ser ou INDIVIDUAL ou ASSOCIADO ou CORPORATIVO. Já **SOBREPOSTOS**, uma entidade superior pode pertencer a mais de um conjunto de entidades de nível inferior. O que é o caso da figura 2.17, onde um ALUNO pode ser um FUNCIONARIO da escola.

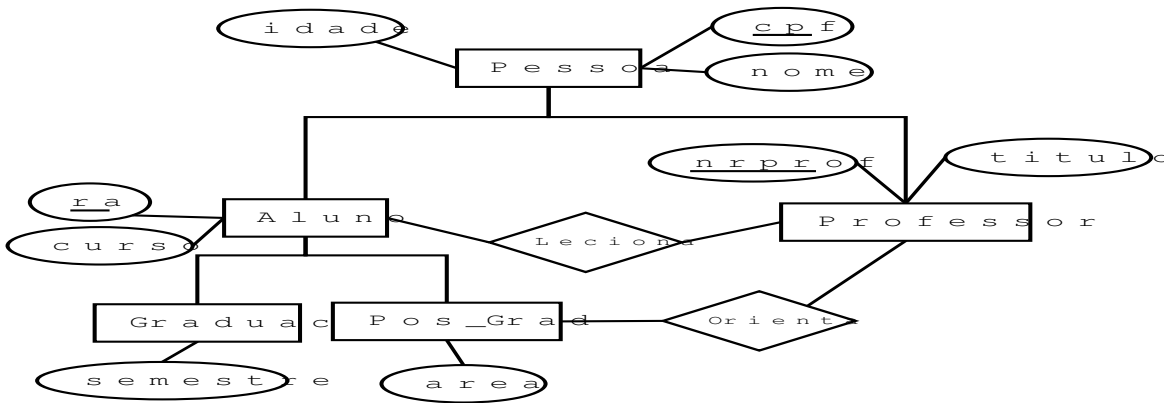


Figura 2.17: Múltiplas Especializações

- **Total/Parcial:** Onde **TOTAL**, significa que cada entidade superior deve pertencer a um conjunto de entidades de nível inferiores. Já **PARCIAL**, significa que uma entidade superior pode pertencer a um conjunto de entidades de nível inferior.

As possíveis combinações são:

- **Parcial Exclusiva:** Por exemplo, Um departamento ministra disciplinas para cursos de graduação e pós-graduação (deve haver estas entidades). Além disso pode ministrar disciplinas para treinamento sob solicitação de empresas (não precisa haver a entidade empresa).

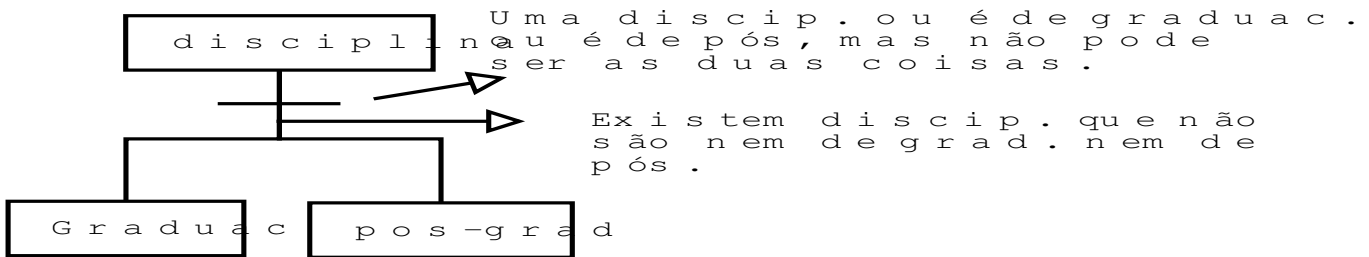


Figura 2.18: Parcial exclusiva

- **Parcial Sobreponível:** Um departamento contrata pessoal para desempenhar suas funções, tais como vigias, secretários, bibliotecários, etc.

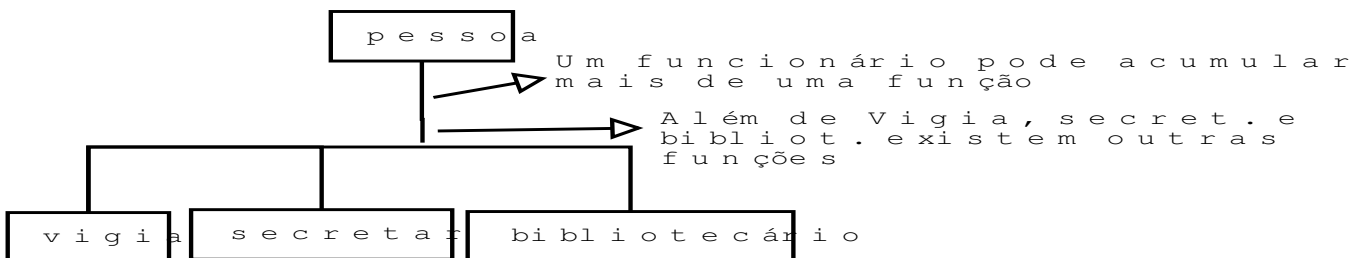


Figura 2.19: Parcial Sobreponível

- **Total Exclusiva:** Um departamento ministra disciplinas para cursos de grad.e pos-grad. Além disso pode ministrar disciplinas de especialização para treinamento sob solicitação de empresas.

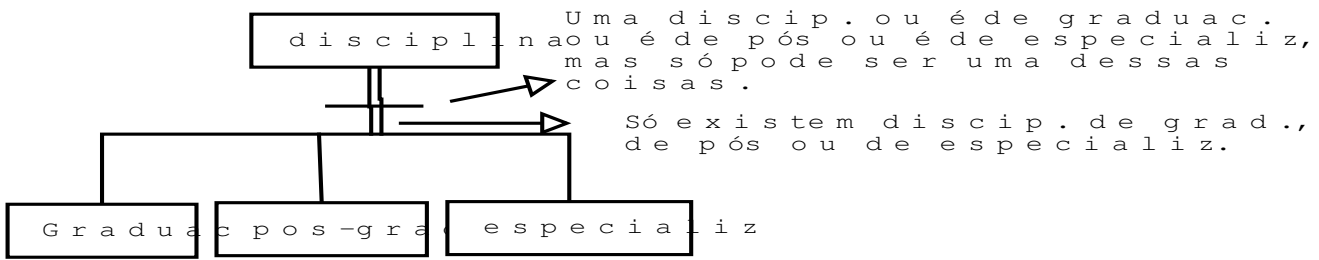


Figura 2.20: Total exclusiva

- **Total Sobreponível:** Os alunos de um departamento são de Graduação ou de Especialização, conforme os cursos que frequentam.

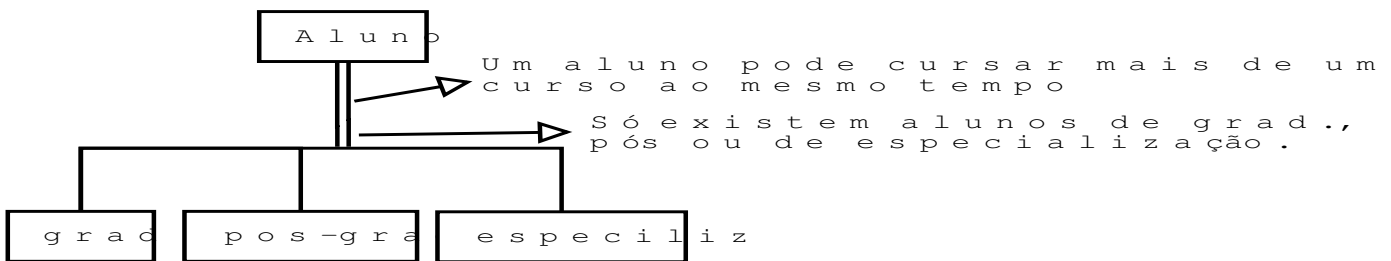


Figura 2.21: Total Sobreponível

### 2.5.2 Agregação

Uma limitação do modelo E-R é que não é possível expressar relacionamentos entre relacionamentos. Além

Para ilustrar esta necessidade, considere um banco de dados descrevendo informações sobre Funcionários que trabalham em um determinado Projeto e utilizam uma série de diferentes Máquinas em seus trabalhos.

Usando o modelo básico de construção E-R, obtemos o diagrama E-R da figura 2.22.

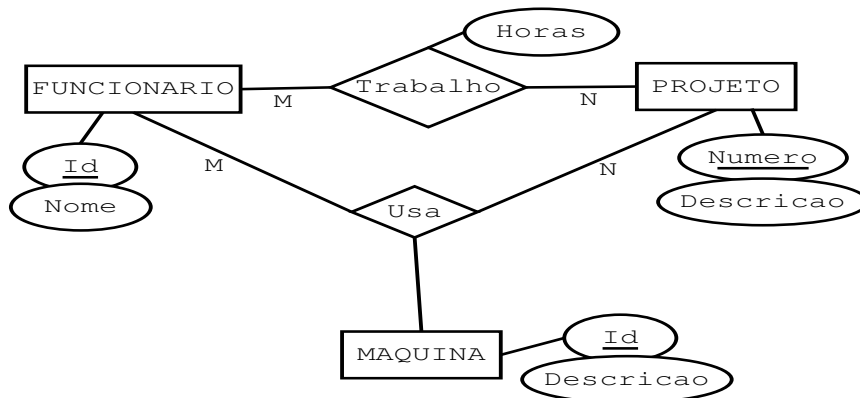


Figura 2.22: Diagrama E-R com relacionamentos redundantes

No exemplo da figura 2.22, o relacionamento *Usa* deve empregar informações já existentes previamente no conjunto de relacionamento *Trabalho*. Além disso, há pares Funcionário-Trabalho que não usam nenhuma máquina, isto é, não estão em *Usa*.

A solução é usar a **Agregação**. A agregação é uma abstração por meio da qual relacionamentos são tratados como entidades de nível superior. Assim, para nosso exemplo, o relacionamento *Trabalho* e as entidades FUNCIONÁRIO e PROJETO, torna-se-ão uma única entidade.

**Isto** permite que relacionar o conjunto FUNCIONARIO, TRABALHO e PROJETO com a entidade MÁQUINA.

A figura 2.23, mostra a solução de nosso problema em forma de agragação.

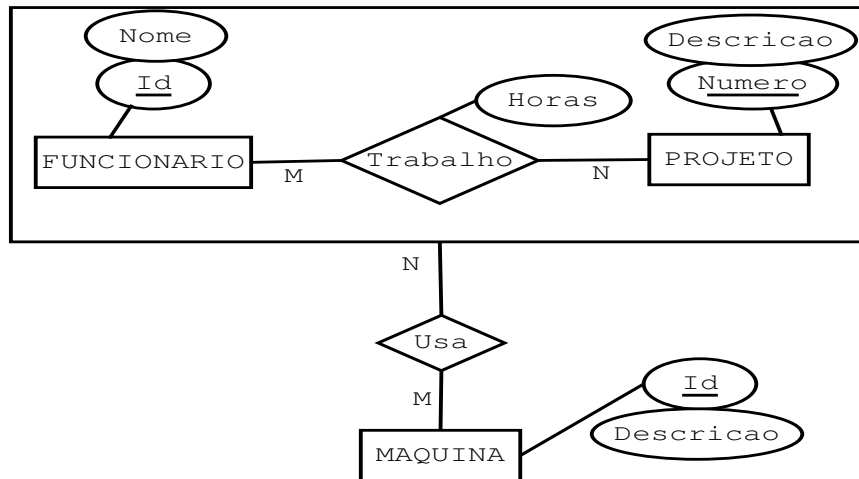


Figura 2.23: Diagrama E-R com Agregação

Agora com a gregação, significa dizer que o conjunto Funcionário-Projeto nem sempre utilizam máquinas.

**Outros** exemplos: Assassino-Vitima-Arma, Cliente-Conta-CartãoCrédito, Rua-Bairro-Linha.

**Agora** que já temos a noção do uso da Agregação vamos analisar os casos em que se deve usá-la:

- Quando é necessário identificar-se cada relacionamento.
- Quando é necessário mais de um relacionamento envolvendo as mesmas entidade (ACABAR)

### 2.5.2.1 1 Caso do uso de Agregação (Opcionalidade)

### 2.5.2.2 2 Caso do uso de Agregação (Identificador em Relacionamento)

### 2.5.2.3 3 Caso do uso de Agregação (indica tempo)

## 2.6 Exercícios

### 2.6.1 Primeira Lista

**1 - Obtenha o modelo Entidade-Relacionamento dos seguinte enunciados. Coloque os atributos necessários das entidades. Coloque pelo menos 3 atributos em cada entidade.**

1. Uma Empresa possui funcionários. Um funcionário trabalha em uma Empresa
2. Os Atletas participam de competições. Em uma compatição participam vários atletas.
3. Deseja-se fazer um banco de dados para uma rede de hotelaria. Um hotel possui quartos. Cada quarto pertence a apenas um hotel.

4. Um soldado, que possui as características nome, Registro Militar (RM), data de nascimento, possui armas. Uma arma, que possui as características de série, registro e calibre, é de um soldado. Uma arma é limpa por vários soldados. Um soldado limpa várias armas.
5. Um funcionário é supervisionado por um gerente (o gerente também é um funcionário). Um gerente (que também é funcionário) supervisiona vários funcionários. Do funcionário deseja-se saber nome, cpf e endereço.
6. Um médico trata de pacientes. Do médico deseja-se saber CRM, nome e suas especializações. Um paciente, no qual há a necessidade de sabermos seu nome, endereço e idade, é tratado por vários médico. Um paciente realiza vários tipos de exames. Um tipo de exame, destes há a necessidade de guardar seu número, data e descrição, é feito por vários paciente.
7. O aluno cursa disciplinas lecionadas por um professor cada uma. Para cada aluno deve-se manter as informações de ra, nome e seus telefones. Uma disciplina é cursada por vários alunos e é lecionada por um professor. Das disciplinas deseja-se saber código, número de créditos e descrição. Os professores lecionam diversas disciplinas cada um e em cada disciplina possui diversos alunos. Dos professores deseja-se saber seu código, nome e telefone.
8. Um médico trata de pacientes. Do médico deseja-se saber CRM, nome e suas especializações. O médico pede exames para vários pacientes. Um paciente, no qual há a necessidade de sabermos seu nome, endereço e idade, é tratado por vários médico. Um paciente realiza vários tipos de exames pedidos pelos médicos. Um tipo de exame, destes há a necessidade de guardar seu número, data e descrição, é feito por vários paciente a pedido dos médicos.
9. Uma empresa possui funcionários. Os funcionários podem ser engenheiros, secretárias ou técnicos.
10. Uma empresa fabrica automóveis. Os automóveis podem ser carros de passeio, caminhões ou ônibus.

### **Identifique as cardinalidade dos exercícios anteriores.**

### **2 - Obtenha o modelo Entidade-Relacionamento dos seguinte enunciados. Coloque os atributos necessários das entidades. Coloque pelo menos 3 atributos em cada entidade.**

1. Uma empresa deseja elaborar um cadastro completo de seus empregados e suas atividades. Para cada empregado é desejado armazenar seu nome, número funcional, telefone (ddd + número) e seus diversos dependentes (nem todo empregado possui dependente), o departamento no qual o empregado trabalha (todo empregado trabalha em um departamento), o departamento o qual o empregado gerencia (nem todo empregado gerencia departamento) e os diversos projetos no qual o empregado trabalha (nem todo empregado trabalha em projeto). Para cada departamento é necessário armazenar seu nome, número, os diversos empregados que o departamento possui (todo departamento possui empregado), o empregado que gerencia o departamento (todo departamento é gerenciado por um empregado) e os diversos projetos que o departamento controla (nem todo departamento controla projetos). Para cada projeto é necessário armazenar seu nome, seu número, as diversas cidades nas quais o projeto é desenvolvido, os diversos funcionários que trabalham no projeto (todo projeto possui funcionários) e o departamento que controla o projeto (todo projeto é controlado por um departamento). Para cada dependente é necessário armazenar seu nome, sexo, o relacionamento com o empregado e o empregado do qual o dependente depende (todo dependente depende de um empregado).
2. Um Hotel deseja montar um banco de dados de Hospedagem. Do cliente que hospeda-se no hotel, e desejado saber seu nome, RG, Endereço, telefone e o quarto que esta hospedado. Dos quartos, deseja-se saber seu numero, andar, a quantidade de leitos e os clientes que dormiram no quarto(um quarto pode abrigar mais de um cliente). Deseja-se saber também a data de hospedagem e o que o cliente consumiu.

3. Uma Transportadora quer automatizar seu controle de transporte. Ela deseja ter as seguintes informações de seus caminhões: Marca, modelo, ano, capacidade de transporte e a data em que um motorista viajou com o caminhão(mais de um motorista pode dirigir um caminhão). Do motorista deseja-se saber Nome, RG, Idade, Endereço e o caminhão com o qual já viajou. Um caminhão pode transportar diversos produtos, destes deseja-se saber nome, marca, fabricante e data de transporte( um tipo de produto pode viajar em mais de um caminhão).
4. A Federação Paulista de Futebol deseja elaborar um cadastro geral para o Campeonato Paulista de 1999. Para cada time é desejado armazenar seu nome, sua cidade, seu número de cadastro na FPF, a situação do time no campeonato, o estádio que o time possui (todo time possui um estádio), os jogos que o time participa (todos os times participam de jogos) bem como, o número de gols que o time marcou na partida, as diversas torcidas organizadas que o time possui (um time não é obrigado a possuir torcida organizada), os diversos jogadores que compõem o elenco do time (todo time possui jogadores no elenco) e os diversos jogadores cujos passes pertencem ao time (um time não é obrigado a possuir passes de jogadores). Para cada jogo é desejado armazenar seu número, a data, horário, os diversos membro da comissão de arbitragem, o estádio no qual o jogo é realizado (todo jogo é realizado em estádio), os times que participam do jogo (todo jogo é realizado por times), as diversas torcidas organizadas que frequentaram o jogo (nem todo jogo deve frequentado por torcidas organizadas) e os diversos jogadores que participaram ativamente do jogo, sendo desejado armazenar o número de gols, o número de cartões amarelos e o número de cartões vermelhos que o jogador recebeu no jogo (todo jogo é disputado por jogadores). Para cada estádio é desejado armazenar seu nome, localização, o número de torcedores, os diversos jogos que o estádio abriga (um estádio não é obrigado a abrigar jogos) e o time proprietário do estádio (um estádio pode ser público). Para cada torcida organizada é desejado armazenar seu número de cadastro na FPF, seu nome, o nome de seu presidente, sua sede, o número de associados, os diversos jogos que a torcida frequenta (uma torcida não é obrigada a frequentar jogos) e o time para o qual a torcida torce (toda torcida torce para um time). Para cada jogador é desejado armazenar o número de cadastro do jogador na FPF, seu nome, apelido, idade, o time ao qual o passe do jogador pertence (o jogador pode ter passe livre), o time para o qual o jogador atua (o jogador pode estar cadastrado na FPF e não atuar em um time) e os jogos dos quais o jogador participa ativamente (um jogador não é obrigado a participar de jogos ativamente).
5. Uma universidade deseja elaborar um cadastro acadêmico, envolvendo seus alunos, cursos e disciplinas. Para cada faculdade é desejado armazenar seu nome, seu número, os diversos departamentos que a faculdade possui (toda faculdade é dividida em departamentos) e os diversos professores que a faculdade possui (toda faculdade possui professores). Para cada departamento é necessário armazenar seu número, sendo que para cada faculdade, a numeração de seus departamentos vai de 1 até N, seu nome, o professor que chefia este departamento (todo departamento é chefiado por um professor), a faculdade à qual o departamento pertence (todo departamento pertence a uma faculdade), as diversas disciplinas que o departamento oferece (nem todo departamento oferece. Para cada curso é desejado armazenar seu código, seu nome, os diversos períodos nos quais o curso é oferecido, o departamento ao qual o curso pertence (todo curso pertence a um departamento), os diversos alunos que frequentam o curso (se não houver alunos frequentando o curso o mesmo não será cancelado) e as diversas disciplinas que são oferecidas pelo curso (todo curso oferece disciplinas). Para cada disciplina é desejado armazenar seu nome, código, carga horária, o departamento pelo qual a mesma é oferecida (toda disciplinas é oferecida por um departamento), os diversos cursos para os quais a disciplina é oferecida (mesmo que uma disciplina não seja oferecida por um curso ela não será cancelada), os diversos professores que lecionam a disciplina (toda disciplina é lecionada por professores) e os diversos alunos que frequentam a disciplina, sendo necessário armazenar para cada aluno a nota 1 N1, nota 2 N2, as faltas F, a média final do aluno  $((N1 + N2)/2)$  e a situação do aluno (média final > 7) e  $F \leq 25\%$  aluno é aprovado (se não houver alunos frequentando a disciplina, a mesma não será cancelada). Para cada aluno é desejado armazenar seu nome, ra, seus endereços completos (o da cidade onde reside e o da cidade da universidade) com rua, número, bairro, cep, cidade, seus telefones (o da cidade onde reside e o da cidade da universidade) com DDD + número, o curso que o aluno frequenta (todo aluno frequenta

um curso, não podendo frequentar mais que um), as diversas disciplinas que o aluno frequenta (todo aluno frequenta disciplina) e o professor que orienta o aluno (nem todo aluno é orientado por professor). Para cada professor é desejado armazenar seu número funcional, seu nome, a faculdade à qual ele pertence (todo professor pertence a uma faculdade), o departamento o qual o professor chefia (um professor não é obrigado a chefiar um departamento) e as diversas disciplinas que o professor leciona (um professor não é obrigado a lecionar disciplinas).

6. Um hospital deseja elaborar um cadastro para controlar as atividades de suas clínicas. Para cada clínica é desejado armazenar seu nome, seu código, sua especialidade e os diversos médicos que a clínica possui (toda clínica possui médicos). Para cada médico é desejado armazenar seu nome, número do CRM, sua especialidade, as diversas consultas que o médico faz (o médico não é obrigado a realizar consultas), as diversas cirurgias que o médico faz (um médico não é obrigado a realizar cirurgias) e a clínica ao qual o médico pertence. Para cada consulta é desejado armazenar seu número, sua data, horário, o médico que realizou a consulta (toda consulta é realizada por um médico) e o paciente que foi consultado (toda consulta é feita em um paciente). Para cada cirurgia é necessário armazenar seu número, a data, o horário, os diversos médicos que realizaram a cirurgia (toda cirurgia é realizada por médicos), as diversas enfermeiras que auxiliaram na cirurgia (toda cirurgia é auxiliada por enfermeiras) e o paciente no qual é realizada a cirurgia (toda cirurgia é realizada em um paciente). Para cada paciente é desejado armazenar seu cod, nome, rg, dt de nascimento, endereço completo (rua, número, bairro, cep, cidade), as diversas consultas que o paciente realizou (um paciente não é obrigado a realizar consultas), as cirurgias que o paciente realizou (um paciente não é obrigado a realizar cirurgias) e as diversas enfermeiras que atenderam o paciente assim como a data e o horário do atendimento (nem todo paciente é atendido por enfermeira). Caso o paciente seja menor de idade e não possua RG, então será necessário armazenar o RG do responsável pelo mesmo. Para cada enfermeira é desejado armazenar seu número, seu nome, sua especialidade, as diversas cirurgias que a enfermeira auxiliou (nem toda enfermeira auxilia em cirurgia) e os diversos pacientes que a enfermeira atendeu (toda enfermeira atende paciente).
7. Uma empresa deseja elaborar um sistema para controlar suas vendas. Para cada vendedor é necessário armazenar seu nome, seu número, endereço, os diversos clientes que o vendedor atende (todo vendedor atende clientes) e as diversas notas fiscais que o vendedor emite (todo vendedor emite notas fiscais). Para cada cliente é necessário armazenar seu código, nome fantasia, nome da empresa, cgc, os diversos vendedores que atenderam este cliente (todo cliente é atendido por vendedor) e todos os pedidos de compra que os clientes emitiram (nem todo cliente cadastrado emite pedido de compra). Para cada pedido de compra é desejado armazenar o seu número, a data de emissão, o valor total do pedido de compra, o cliente que emitiu o pedido de compra (todo pedido de compra é emitido por um cliente), as notas fiscais geradas para este pedido de compra (todo pedido de compra gera notas fiscais) e os diversos itens de pedido de compra (todo pedido de compra possui itens de pedido de compra). Cada item de pedido de compra é identificado por seu número que vai de 1..20 para cada pedido de compra, a quantidade, o valor do item, o pedido de compra ao qual o item pertence (todo item pertence a um pedido de compra) e o produto que o item descreve (todo item de pedido de compra descreve um produto). Para cada nota fiscal é necessário armazenar seu número, sua série (a numeração varia pela série), a data de emissão, o valor total da nota, o vendedor que emitiu a nota (toda nota é emitida por um vendedor), o pedido de compra que gerou a nota (toda nota é gerada por um pedido de compra) e os diversos itens de nota que a mesma contém (toda nota contém itens de nota). Para cada item de nota é necessário armazenar seu número que vai de 1..20 para cada nota fiscal, a quantidade de produto, o valor total do item, a nota fiscal ao qual o item pertence (todo item pertence a uma nota fiscal) e o produto que o item descreve (todo item de nota fiscal descreve um produto). Cada produto é descrito por seu código, descrição, valor unitário, os diversos itens de pedido de compra nos quais o mesmo é citado (um produto não é obrigado a estar citado em um item de pedido de compra) e todos os itens de nota fiscal nos quais o produto é descrito (nem todo produto deve ser descrito em item de nota fiscal).

**3 - Resolver os seguintes exercícios da Bibliografia 2:  
2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.8, 2.14, 2.15, 2.16 e 2.18**

## **2.7 Bibliografia**

[1] - Machado, F.; Mauricio, A. *Projeto de Banco de Dados*, Editora. Érica, 5 ed., Capítulo 4.

[2] - Korth, H.; Silberschatz, A. *Sistema de Banco de Dados*. Editora Makron Books, 3 ed., 1999

# Capítulo 3

## O Modelo Relacional

### 3.1 Introdução

Criado por Edgar Codd, nos anos 70, começou a ser realmente utilizado nas empresas a partir de 1987, através do SGBDs. A abordagem relacional está baseada no princípio de que as informações em uma base de dados podem ser consideradas como relações matemáticas e que estão representadas de maneira uniforme, através do uso de tabelas, ou falando de uma forma mais direta, um arquivo. Porém, um arquivo é mais restrito que uma tabela. Toda tabela pode ser considerada um arquivo, porém, nem todo arquivo pode ser considerado uma tabela. Este princípio coloca os dados (entidades e relacionamentos) dirigidos para estruturas mais simples de armazenar dados, que são as tabelas.

O conceito principal vem da teoria dos conjuntos (álgebra relacional) atrelado à idéia de que não é relevante ao usuário saber onde os dados estão nem como os dados estão (transparência). Os usuários manipulam objetos dispostos em linhas e colunas das tabelas, como mostra a figura a seguir. O usuário pode lidar com estes objetos, conta com um conjunto de operadores e funções de alto nível, constantes na álgebra relacional.

Terminologia do modelo relacional:

- **Tabela** é chamada de **RELAÇÃO**;
- **Linha** de uma tabela é chamada de **TUPLA**;
- **Coluna** é chamado de **ATRIBUTO**;
- O tipo de dado que descreve cada coluna é chamado de **DOMÍNIO**.

A figura 3.1 representa uma tabela de alunos:

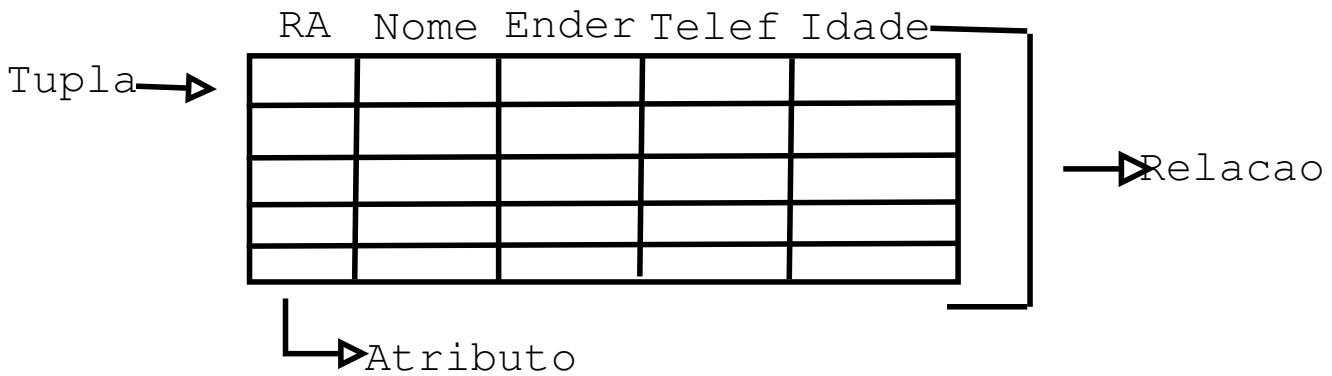


Figura 3.1: Representação de uma tabela no Modelo Relacional

Uma relação é representada da seguinte maneira:

$Aluno = \{Ra, Nome, Ender, Telef, Idade\}$

## 3.2 Domínios

O **Domínio** de um atributo é, em geral, um tipo de dado que especifica o que o atributo pode receber.

**Exemplo:**

- Número de salas de aula
  - Conjunto dos números de 1 a 150, inteiros no formato 999.
- Nomes de alunos
  - Conjunto de todos os nomes possíveis para pessoas no formato `String[60]`.
- Códigos de Disciplinas
  - Conjunto de três letras seguidas de um traço e de três dígitos: `AAA-999`.

É um conjunto de valores **atômicos**.

**Valor Atômico** significa um valor **indivisível** e **monovalorado**.

## 3.3 Relações

**Dado** o seguinte esquema de uma Relação de Alunos:

$Aluno = \{Nome, RG, Idade\}$

uma possível instanciação para esse esquema é a Relação:

$R(Aluno) = \{ \langle José, 12345, 21 \rangle, \langle Pedro, 54321, 18 \rangle, \langle Paulo, 32123, 22 \rangle \}$

**Como** um esquema de uma Relação **R** é definido como um conjunto, não existe a idéia de *ordem*. Assim, desde que se indique que cada valor **Vi** corresponde a um atributo **Ai**, a ordem dos atributos em esquemas de relações é apenas uma questão de disposição física.

**Importante:** A instância de uma relação em um determinado momento, é toda a relação no momento, ou seja, uma **instância de Alunos são todos os alunos cadastrados no momento**. Se amanhã acrescentar mais alunos, a instância será todos os **alunos antigos mais os novos**

## 3.4 Chave no Modelo Relacional

### 3.4.1 Superchave

**Um** conjunto de atributos de uma relação R que identifica univocamente cada tupla na relação R é chamada uma SUPERCHAVE.

**Considere** a seguinte relação Aluno:

Aluno = { Nome, Endereco, Telefone, RA, Idade }

Superchave(Aluno)={ Nome, Endereco, Telefone, RA, Idade }

Superchave(Aluno)={ Nome, Endereco } (Considerando que não existem duas pessoas com o nome em uma residência)

Superchave(Aluno)={ Nome, Telefone }

Superchave(Aluno)={ RA, Nome }

Superchave(Aluno)={ RA }

### 3.4.2 Chave

CHAVE é uma Superchave da qual não se pode retirar nenhum atributo e ainda preservar-se a propriedade de identificação unívoca.

Exemplo:

Aluno = { Nome, Endereco, Telefone, RA, Idade }

Superchave(Aluno)={ Nome, Endereco, Telefone, RA, Idade }: Não é chave, pois se retirarmos idade, ainda conseguiremos identificar um único aluno.

Superchave(Aluno)={ Nome, Endereco }: É chave, pois se retirarmos um atributo não conseguiremos identificar um aluno.

Superchave(Aluno)={ Nome, Telefone }: É chave.

Superchave(Aluno)={ RA, Nome }: Não é chave.

Superchave(Aluno)={ RA }: É chave.

**Em** geral, adota-se a convenção de que os atributos chaves são grifados. Se mais de um atributo participa de uma mesma chave, grifam-se esses atributos todos com o mesmo número de traços.

**Exemplo:**

Aluno = { Nome, Endereco, Telefone, RA, Idade }

### 3.4.3 Chave Candidata

É comum que exista mais de uma chave para uma mesma relação. Neste caso, cada uma das chaves é chamada de **Chave Candidata**.

**Quando** existe mais de uma, grifa-se cada chave candidata com um número diferente de traços.

**Exemplo**

Aluno = {Nome, Endereco, Telefone, RA, Idade}

### 3.4.4 Restrições de Integridade

**São** regras a respeito dos valores que podem ser armazenados nas relações que devem ser sempre satisfeitas.

**Existem** 3 que são consideradas necessárias a uma base de dados relacional:

- **Restrição de Integridade da Chave:** Uma chave candidata qualquer não pode ter o mesmo valor em duas tuplas distintas da mesma relação.
- **Restrição de Integridade da Entidade:** A chave primária de qualquer relação não pode ser nula em nenhuma tupla dessa relação.
- **Restrição de Integridade Referencial:** Informalmente, a restrição de integridade referencial declara que uma tupla em uma relação, que faz referência a outra relação, deve se referir a uma tupla existente nessa relação. O conceito de Integridade Referencial depende do conceito de *Chave Estrangeira*.

#### 3.4.4.1 Chave Estrangeira (Integridade Referencial)

Uma **Chave Estrangeira** ocorre quando um conjunto de atributos  $C \subseteq R1$  que não é chave em  $R1$ , é compatível com outro conjunto de domínio de atributos  $D \subseteq R2$  que é a Chave Primária da relação  $R2$ .

**Exemplo:**

Curso={Cod\_Cur, Nome, Qtidade\_Max\_Aluno}

Aluno={RA, Nome, Endereco, Idade, Num\_Curso}

Dom(Cod\_Cur) = Dom(Num\_Curso)

Se (Cod\_Cur) é Chave Primária de Curso, então (Num\_Curso) é Chave Estrangeira em Aluno.

**Importante:** A chave estrangeira pode ter o valor NULO.

Não existe uma representação formal para chave estrangeira. Normalmente, identifica-se com um arco direto de cada chave estrangeira à relação que ela faz referência. Para maior clareza, a seta deve apontar para a chave primária da relação referida (figura 3.2).

Uma outra maneira é colocar a sigla **FK (Foreign Key)** na frente de cada campo. Neste caso se os nomes dos atributos não forem os mesmos (chave primária com chave estrangeira) fica difícil de identificar de qual relação o atributo é chave estrangeira.

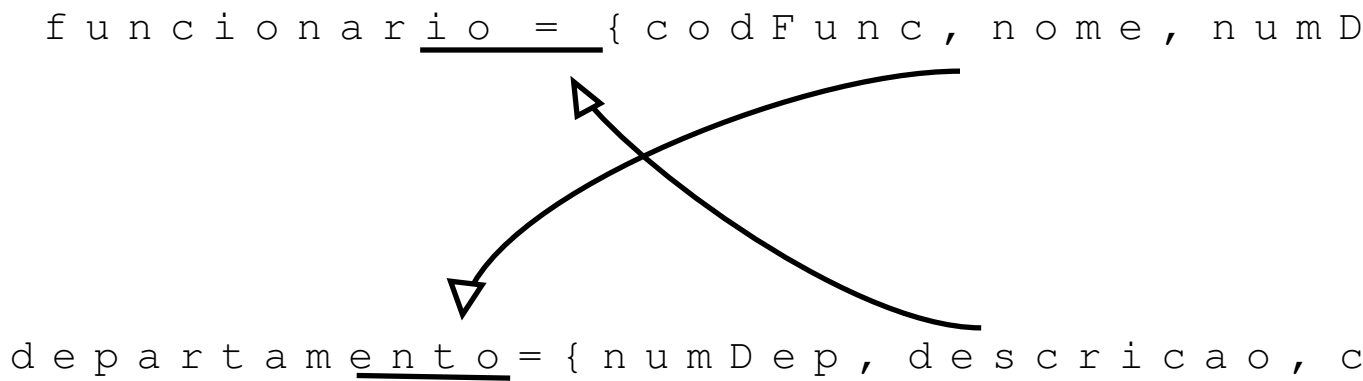


Figura 3.2: Representação de Chave Estrangeira

### 3.4.4.2 Exemplos de Restrições de Integridade

```
Aluno={ Nome, RA, Idade}
{ <Mario, 1234, 20>
  <Paulo, 4312 18>
  <Jose, 1212 21>
  <Marta 3322 19>}
```

```
Disciplina={ Sigla RA_Monitor}
{ D-104 1234
  D-123 1212
  D-149 1234
  D-189 NULL}
```

Dom(aluno.RA)=Dom(disciplina.RA\_Monitor)

**Relações Válidas:** Apesar de RA\_Monitor ser chave estrangeira, ela não é *Chave Candidata*.

```
Aluno={ Nome, RA, Idade}
{ <Mario, 1234, 20>
  <Paulo, 4312 18>
  <Jose, 1212 21>
  <Marta 3322 19>}
```

```
Disciplina={ Sigla RA_Monitor}
{ D-104 1234
  D-123 2222 (Errado)
  D-149 1234
  D-189 }
```

Dom(aluno.RA)=Dom(disciplina.RA\_Monitor)

**Relações Inválidas:** Valor 222 para RA\_Monitor inexistente na tabela Aluno

**Integridade Referencial Violada**

```

Aluno={  Nome,      RA,      Idade}
        { <Mario,    1234,    20>
          <Paulo,   NULL(Errado)  18>
          <Jose,    1212     21>
          <Marta   1212 (Errado)  19>}

```

```

Disciplina={  Sigla  RA_Monitor}
              { D-104    1234
                D-123    1212
                D-149    1234
                D-189    NULL}

```

Dom(aluno.RA)=Dom(disciplina.RA\_Monitor)

**Relações Inválidas:** Valor NULL para o atributo RA na relação Aluno e existem duas chaves primárias com o mesmo valor.

**Integridades de Entidade e de Chave Violadas Respectivamente**

### 3.5 Outros tipos de Restrições

As restrições apresentadas são restrições do próprio modelo relacional. Existem outros tipos, como por exemplo, as *restrições de integridade semânticas*.

Os exemplos destas restrições são “O salário de um empregado não deve exceder o do supervisor em um banco de dados relacional” e o número máximo de horas que um empregado pode trabalhar por semana em todos os projetos é 56”.

Essas restrições podem ser especificadas e impostas dentro dos programas de aplicação que atualizam o banco de dados ou pelo próprio SGBD através de **Stored Procedure** e **Triggers**. (Estes recursos serão vistos posteriormente no curso).

Outro tipo de restrição é a *dependência funcional*, que estabelece um relacionamento funcional entre dois conjuntos de atributos X e Y. Essa restrição especifica que o valor de X determina o valor de Y em todos os estados de uma relação.

### 3.6 Bibliografia

[1] - Machado, F.; Mauricio, A. *Projeto de Banco de Dados*, Editora. Érica, 5 ed., Capítulo 13.

[2] - Traina Jr, Caetano *Apostila de Modelagem de Dados*, USP-São Carlos.

# Capítulo 4

## Mapeamento do Modelo ER para Modelo Relacional

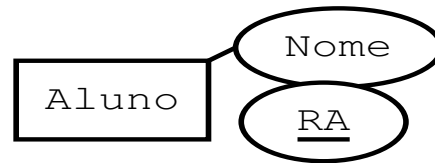
### 4.1 Introdução

- O MER é um Modelo Conceitual: Pode ser usado para especificar conceitualmente a estrutura de dados de uma aplicação.
- O Modelo Relacional é um Modelo Lógico (porém bem próximo ao Físico): É uma descrição de um banco de dados no nível de abstração visto pelo usuário do SGBD. Assim, o modelo lógico é dependente do tipo particular de SGBD que está sendo usado.
- O **Mapeamento** permite que se traduzam os esquemas concebidos com um modelo de conteúdo semântico mais alto para uma implementação utilizando um modelo lógico (ou físico).
- O Mapeamento do MER para o Mod. Relacional (MRel) é um procedimento executado em **6 passos** consecutivos apresentados a seguir.

### 4.2 Mapeando Entidades

Cada Conjunto de Entidades é mapeada como uma relação que envolve todos os atributos do conjunto de entidades.

Os atributos CHAVES comporão a chave da relação (figura 4.1).

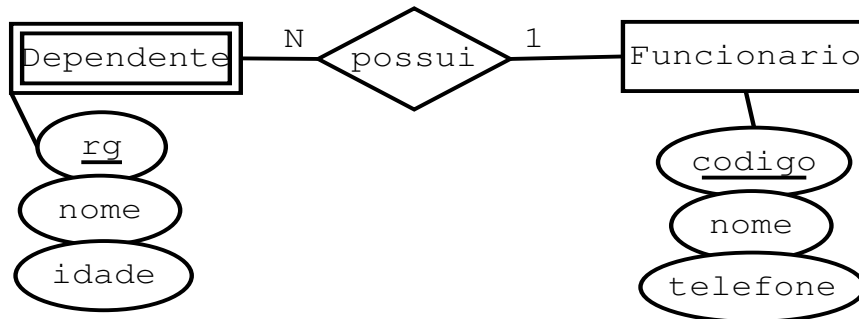


**Aluno**={Nome,RA}

Figura 4.1: Mapeamento Entidades

### 4.3 Entidades Fracas

**Conjunto de Entidades Fracas (CEF)** são mapeadas numa relação formada por todos os atributos do CEF mais os atributos que são chave das relação ao qual o CEF depende (figura 4.2).



**funcionario**={codigo, nome, telefone}  
**dependente**={rg, nome, idade, codigo\_func}

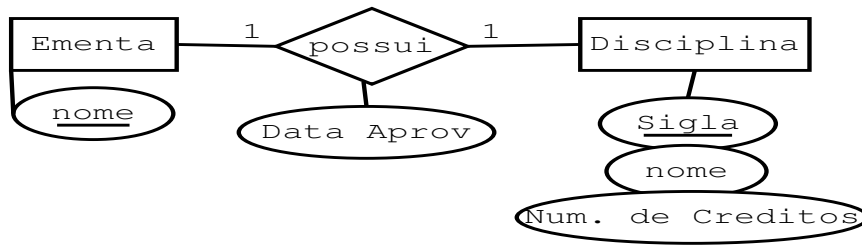
Figura 4.2: Mapeamento de Entidades Fracas

### 4.4 Relacionamento Binário com Cardinalidade 1:1

Conjunto de Relacionamentos Binários (CRB) de Cardinalidade 1:1 não são representados como novas relações. Seus atributos são acrescentados numa das relações que mapeiam os Conjuntos de Entidades (CE) envolvidos (qualquer uma). Nessa mesma relação inclui-se também os atributos chave da relação que mapeia o outro CE (Figura 4.3).

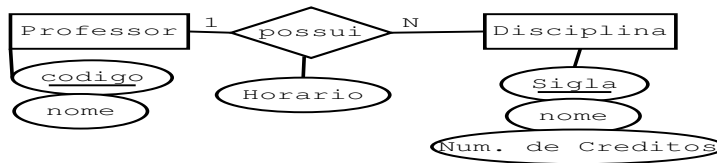
### 4.5 Relacionamento Binário com Cardinalidade 1:N

CRB de cardinalidade 1:N também não são representados como novas relações. Seus atributos são acrescentados na relação que mapeia o CE que ocupa o papel de cardinalidade N. Os atributos chave da relação que mapeia o CE que participa com cardinalidade 1 são também acrescentados na CE com papel de cardinalidade N (Figura 4.4).



**disciplina** = { sigla, nome, num\_credits }  
**ementa** = { nome, data\_aprov, sigla\_discip }

Figura 4.3: Mapeamento de Relacionamentos 1:1

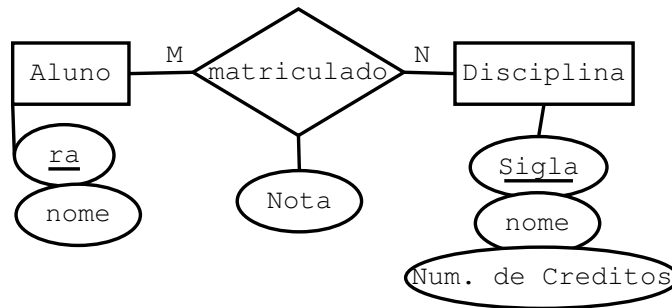


**professor** = { codigo, nome }  
**disciplina** = { sigla, nome, num\_credits, horario, codigo\_prof }

Figura 4.4: Relacionamento Binário com Cardinalidade 1:N

## 4.6 Relacionamento Binário com Cardinalidade M:N

Cada CR Binário de Cardinalidade M:N é representado como uma nova relação. Os atributos da relação são os do CR mais os atributos chave das relações que mapeiam os CEs envolvidos. A *Chave da Relação* é a concatenação dos atributos chaves das relações que mapeiam os CEs envolvidos (Figura 4.5).

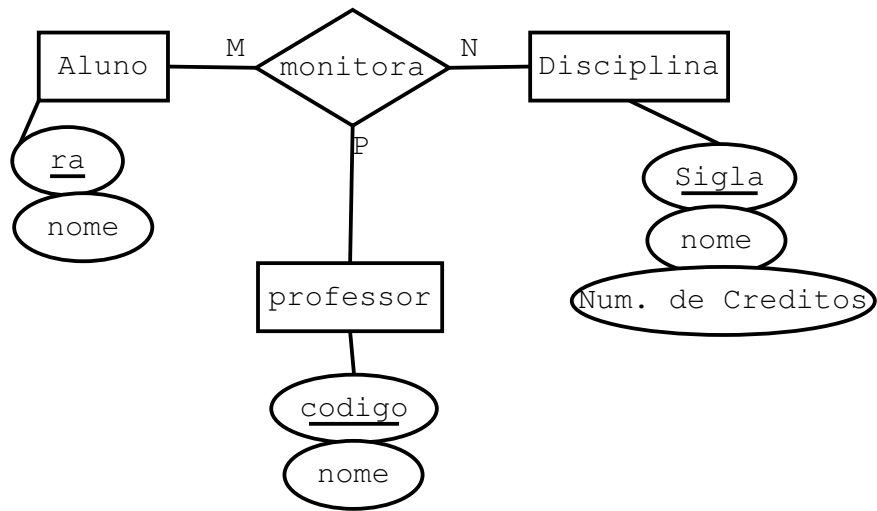


**aluno** = { ra, nome } | **disciplina** = { sigla, nome, num\_credits }  
**matriculado** = { ra, sigla, nota }

Figura 4.5: Relacionamento Binário com Cardinalidade M:N

## 4.7 Relacionamentos Ternários

Conjuntos de Relacionamentos de ordem maior do que dois com cardinalidade diferente de M:N:P têm um mapeamento complexo. Assim, usualmente se mapeiam os cjs. de Relac. ternários, quartenários, etc. como se todos fossem de cardinalidade vários para vários para vários... etc (Figura 4.6).

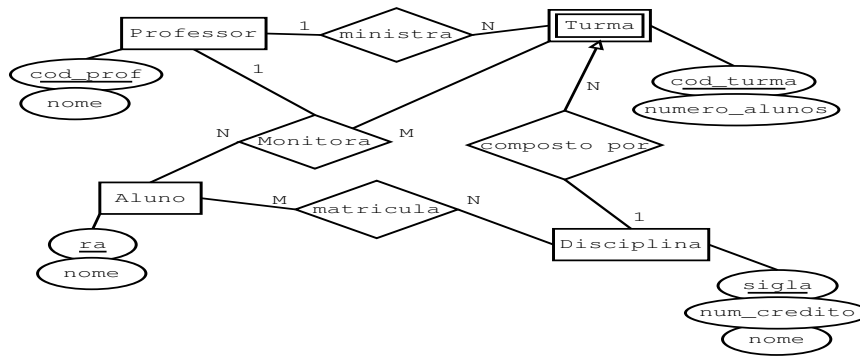


**aluno**={ra, nome} | **disciplina**={sigla, nome, num\_credits} | **professor**={codigo, nome}  
**monitora**={ra, sigla, codigo}

Figura 4.6: Relacionamento Ternário com Cardinalidade M:N:P

## 4.8 Exemplo 1

A figura 4.7 demonstra um exemplo de mapeamento de um DER completo para o modelo Relacional.

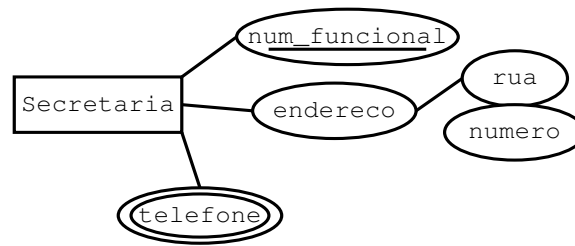


**professor** = {cod\_prof, nome} | **disciplina** = {sigla, nome, n\_credito} | **aluno** = {ra, nome}  
**turma** = {cod\_turma, sigla, n\_aluno, cod\_prof} | **monitora** = {cod\_prof, ra, cod\_turma, sigla}  
**matricula** = {sigla, ra}

Figura 4.7: Exemplo de DER completo mapeado para o Relacional

## 4.9 Atributos Compostos e Multivalorados

Os atributos compostos serão decompostos nas relações e os multivalorados torna-se-ão relações cuja chave primária será composta pela chave da entidade possuidora do atributo mais o atributo multivalorado (Figura 4.8).

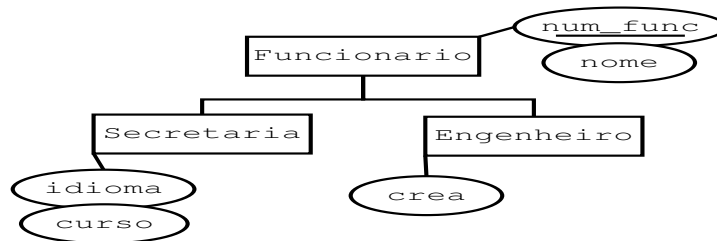


**secretaria**={num\_funcional, rua, numero} | **secretaria\_endereco**={num\_funcional, telefone}

Figura 4.8: Mapeamento de atributos composto e multivalorado

## 4.10 Generalização

A figura 4.9 mostra o mapeamento DER-Relacional de uma Generalização/Especialização.

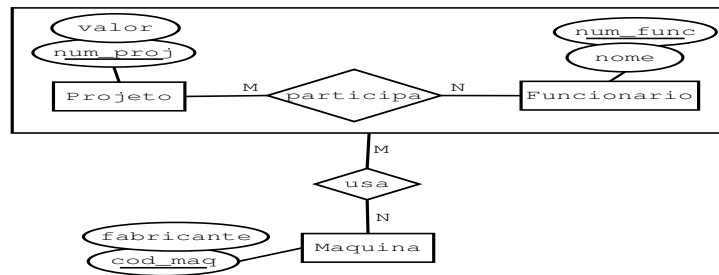


**funcionario**={num\_func, nome, tipo\_funcionario} | **secretaria** = {num\_func, idioma, curso}  
**engenheiro**={num\_func, crea}

Figura 4.9: Mapeamento Generalização/Especialização

## 4.11 Agregação

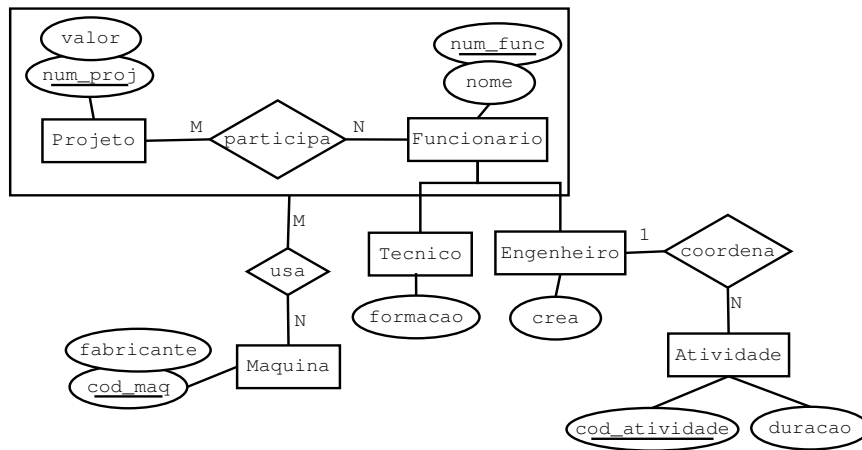
A figura 4.10 mostra o mapeamento DER-Relacional de uma Agregação.



**projeto**={num\_proj, valor} | **funcionario**={num\_func, nome} | **maquina**={cod\_maq, fabricante}  
**participa**={num\_proj, num\_func} | **usa**={num\_proj, num\_func, cod\_maq}

Figura 4.10: Mapeamento Agregação

## 4.12 Exemplo 2



**projeto**={num\_proj, valor} | **funcionario**={num\_func, nome} | **maquina**={cod\_maq, fabric}  
**participa**={num\_proj, num\_func} | **usa**={num\_proj, num\_func, cod\_maq}  
**tecnico**={num\_func, formacao} | **engenheiro**={num\_func, crea}  
**atividade**={cod\_atividade, duracao, num\_func}

Figura 4.11: Exemplo 2 de DER completo mapeado para o Relacional

## 4.13 Resumo dos 6 passos

1. Mapear todos os Cj. de Entidades Regulares do Diagrama ER.
2. Mapear todos os Cj. de Entidades Fracas do Diagrama ER.
3. Mapear todos os Cj. de Relacionamentos 1:1 do Diagrama ER.
4. Mapear todos os Cj. de Relacionamentos 1:N do Diagrama ER.
5. Mapear todos os Cj. de Relacionamentos M:N do Diagrama ER.
6. Mapear todos os Cj. de Relacionamentos Ordem > 2 do Diagrama ER.

## 4.14 Bibliografia

[1] - Traina Jr, Caetano Apostila de Modelagem de Dados, USP-São Carlos.

[2] - Setzer, Valdemar Banco de Dados: Conceitos, Modelos, Gerenciadores, Projeto Lógico e Físico, Editora Edgard Blucher Ltda, 3 ed.

[3] - Machado, F.; Mauricio, A. Projeto de Banco de Dados, Editora Érica, 5 ed., Capítulo 13.

# Capítulo 5

## Normalização de Relações

### 5.1 Introdução

O controle de Consistência pode ser feito:

- Pelo SGBD
- Pelo Aplicativo
- Pela própria construção do sistema

O controle obtido pela própria construção do sistema é , em geral, a melhor, pois não incorre em perda de eficiência durante a execução.

O controle através da própria construção do sistema é obtido no Modelo Relacional construindo-se as relações segundo regras que garantem a manutenção de certas propriedades.

As relações que atendem a um determinado conjunto de regras diz-se estarem em uma determinada **Forma Normal**.

### 5.2 Normalização de Relações

O processo de Normalização permite ao programador **controlar** quanto da consistência é garantida pela maneira de construção do sistema, e quanto deve ser responsabilidade dos aplicativos e/ou SGBD.

*Normalizar demais diminui a eficiência dos aplicativos e de menos abre flacos para inconsistência.*

**Antes** de entrarmos em detalhes sobre as formas normais, vamos olhar alguns problemas de relações mal normalizadas

### 5.3 Anomalias de Modificações

Considere a figura 5.1, ela demonstra uma tabela contendo nome de alunos de uma academia e suas respectivas atividades físicas com seus respectivos valores.

Esta tabela é representada pela seguinte relação:

*cliente = {nome, atividade, taxa}*

Tabela 5.1: Relação Aluno

Nome	Atividade	Taxa
José	Musculacao	30,00
Pedro	Judô	35,00
Manoel	Judo	35,00

**Suponha** a tupla do aluno José, bem neste caso, perdemos, além do nome do aluno, as informações referentes a atividade Musculação, bem como seu valor.

**Este** problema é denominado **Anomalia de Eliminação**.

**Outro** problema ocorre quando a academia implanta um novo curso e não podemos inseri-lo até que um aluno tenha a disposição de fazê-lo.

**Isto** é denominado **Anomalia de Inserção**.

**Agora**, note que Judô, está grafado de forma errada na tupla do aluno Manoel. Se uma busca for feita por Judô, só irá aparecer 1 aluno e não 2 alunos

**Denominamos** estes problema como **Anomalia de Modificação**.

## 5.4 Primeira Forma Normal

As Formas Normais têm nomes pelas quais são conhecidas.

**Para** o modelo relacional, a Forma Normal (FN) mais importante é a chamada **1 forma normal (1FN)**.

**Definição** da 1FN

*Uma relação está na 1FN quando todos os seus atributos são **Atômicos e Monovalorados**.*

**Um** atributo atômico é aquele que não é tratado em partes separadas.

**Um** atributo monovalorado é aquele que possui somente um valor (não uma lista).

**Considere** a seguinte relação:

*cliente={CPF, nome, endereço, (telefone)}*

A relação Cliente não está na 1FN, porque Nome e Endereço são atributos compostos e telefone é um atributo multivalorado.

A normalização desta relação resulta nas seguintes relações:

*cliente={CPF, nome, sobrenome, rua, numero\_casa, bairro}*  
*cliente\_telefone = {CPF, DDD, numero\_tel}*

**Note** que Telefone foi para uma nova relação composta pela chave primária de cliente mais o telefone (decomposto). Todos os atributos da relação Cliente\_Telefone fazem parte da chave primária.

**Outros** exemplos

*curso = {cod\_curso, descricao, (ra\_aluno, nome\_aluno)}*  
*obra = {cod\_obra, preco, (cidades)}*

**De** forma geral, para determinar a chave primária de uma tabela originada pela regra da 1FN deve-se proceder como segue:

1. Tomar como parte da chave primária da tabela na 1FN a chave primária da tabela Original.
2. Verificar se esta chave primária é suficiente para identificar as linhas da tabela na 1FN.
  - (a) Caso seja suficiente, a chave primária da tabela na 1FN é a mesma que a da tabela Original.
  - (b) Caso contrário, deve-se determinar quais as demais colunas que são necessárias para identificar as linhas da tabela na 1FN, compondo assim a chave primária na 1FN.

## 5.5 Dependência Funcional

**Para** entender as duas formas normais que serão apresentadas a seguir é necessário compreender o conceito *Dependência Funcional*.

**Dependência Funcional** é um relacionamento entre pelo menos dois atributos.

**Se** o valor de um conjunto de atributos A permite descobrir o valor de um outro conjunto B, dizemos que **A determina funcionalmente B** ou que **B depende de A**. Denotamos esta relação da seguinte forma:

$$A \rightarrow B$$

**Exemplo:**

RA  $\rightarrow$  Nome, Idade, Curso  
 (Sigla, Sala, Hora)  $\rightarrow$  Codigo\_Turma

**Importante:**

As dependências Funcionais devem ser identificadas pelo projetista do sistema sendo desenvolvido, **NÃO EXISTE MANEIRA DE INFERIR AS DEPENDÊNCIAS A PARTIR DA DESCRIÇÃO DA BASE.**

## 5.6 Segunda Forma Normal (2FN)

A 2 FN está relacionada com as Dependências Funcionais.

**Definição** da 2FN

*Uma tabela encontra-se na segunda forma normal, quando, além de estar na 1FN, não contém Dependências Funcionais Parciais, ou seja, todos atributos não chave devem depender funcionalmente da chave primária composta.*

**Importante:**

Deve-se verificar a violação da 2FN somente se a relação contiver **chaves compostas**.

**Considere** o seguinte exemplo:

aluno\_disciplina = {ra\_aluno, cod\_discip, nome\_aluno, carga\_horar\_discip, nota}

As dependências funcionais são:

ra\_aluno -> nome\_aluno  
 cod\_discip -> carga\_horar\_discip  
 (ra\_aluno, cod\_aluno) -> nota

**Pela** demonstração das dependências percebemos que existem atributos que contém dependência parcial da chave, neste caso, *nome\_aluno* depende *ra\_aluno* e *carga\_horar\_discip* depende *cod\_discip*.

**Isto** caracteriza a violação da 2FN, visto que nesta regra não deve-se existir dependência parcial da chave.

A normalização desta relação ficaria da seguinte forma.

aluno\_discipla = {ra\_aluno, cod\_discip, nota}  
 disciplina = {cod\_discip, carga\_horar\_discip}  
 aluno = {ra\_aluno, nome\_aluno}

A normalização foi feita da seguinte forma:

1. Mantém-se na tabela original as chaves e os atributos que dependem totalmente dela.
2. Para cada chave que possua atributos dependentes, cria-se uma nova relação, neste caso Aluno e Disciplina.

**Outros** exemplos:

projeto\_funcionario = {cod\_proj, cod\_func, nome, categoria, salario, data\_ini, temp\_proj}  
 onde:  
 (cod\_proj, cod\_func) -> data\_ini, temp\_proj  
 cod\_func -> nome, categoria, salario

cliente\_produto = {cpf\_cli, cod\_prod, nome\_cli, end\_cli, (telef\_cli), valor\_unit, qtdade}  
 onde:  
 (cpf\_cli, cod\_prod) -> qtdade  
 cpf\_cli -> nome\_cli, end\_cli, telef\_cli  
 cod\_prod -> valor\_init

## 5.7 Terceira Forma Normal (3FN)

A 3FN resolve problemas relacionados com *Dependências Transitivas*.

**Diz-se** que um atributo tem dependência transitiva quando depender de outro atributo que não é a chave primária da relação. Observe o exemplo:

compra = {cod\_compra, cod\_cliente, nome\_cliente, valor\_compra, tel\_cliente}  
 cod\_compra -> cod\_cliente, valor\_compra  
 cod\_cliente -> nome\_cliente, tel\_cliente

**Perceba** que o atributo chave primária *cod\_compra* identifica os atributos *cod\_cliente* e *valor\_compra* e que o atributo *cod\_cliente*, que não é chave primária, identifica os atributos *nome\_cliente* e *tel\_cliente*. Então, tanto *nome\_cliente* como *tel\_cliente* depende transitivamente de um atributo (neste caso *cod\_cliente*) que não é chave primária.

**Definição** da 3FN:

*Uma Relação está na 3 Forma Normal quando estiver na 1 FN e não existir dependência transitiva dos atributos.*

A normalização de uma relação para a 3FN dar-se pela seguinte maneira:

- Verifica-se um grupo de atributo que depende não diretamente da chave.
- Retira-se da relação esse grupo de atributos.
- Cria-se uma nova relação que contém esse grupo de atributos e inclui-se nela como chave os atributos dos quais esse grupo depende diretamente.
- Repetem-se esses passos até que todos os atributos restantes na relação original dependam diretamente de toda sua chave.

**Exemplos**

```
* compra = {cod_compra, cod_cliente, nome_cliente, valor_compra, tel_cliente}
compra = {cod_compra, cod_cliente, valor_compra}
cliente = {cod_cliente, nome_cliente, tel_cliente}
```

```
* chamada_funcionario = {rg_funcion, num_chamado, duracao_chamada, nome_funcion, cod_cidade_chamada,
nome_cidade_chamada}
chamada_funcionario = {rg_funcion, num_chamado, duracao_chamada, cod_cidade}
funcionario = {rg_funcion, nome_funcion}
cidade = {cod_cidade, nome_cidade}
```

**5.8 Exemplo Prático**

Vamos agora considerar um exemplo prático, baseado em uma nota fiscal de compra representada pela figura 5.1.

Numero Nota Fiscal:	Data da Nota:			
Codigo do Cliente:				
Nome do Cliente:	Telefone Cliente:			
Logradouro Cliente (rua, bairro, etc):				
Cod.Prod	Desc.Prod	Qtidade	Val.Prod	Val.Total

Figura 5.1: Nota Fiscal

Considere que um analista mapeou a nota para o seguinte esquema relacional:

$nota\_fiscal = \{ \underline{num\_nota}, cod\_cliente, nome\_cliente, logradouro\_cliente, telefone\_cliente, data\_nota\_fiscal, (cod\_produto, desc\_produto, qtidade, valor\_produto, valor\_total) \}$

Ao analisarmos esta relação, notamos que mesma causará problemas de inconsistência de dados. Bem, para evitarmos futuros problemas, vamos aplicar as formas normais e corrigir erros na relação.

1. Primeira forma normal:

A relação não está na 1FN pois existem atributos multivalorados e compostos. Vamos então passar para a 1FN:

$nota\_fiscal = \{ \underline{num\_nota}, cod\_cliente, rua\_cliente, bairro\_cliente, cep\_cliente, cidade\_cliente, telefone\_cliente, data\_nota\_fiscal \}$

$nota\_produto = \{ \underline{num\_nota}, cod\_produto, desc\_produto, qtidade, valor\_produto, valor\_total \}$

2. Segunda forma normal:

A relação não está na 2FN pois existem atributos com dependência parcial da chave primária:

$nota\_fiscal = \{ \underline{num\_nota}, cod\_cliente, rua\_cliente, bairro\_cliente, cep\_cliente, cidade\_cliente, telefone\_cliente, data\_nota\_fiscal \}$

$nota\_produto = \{ \underline{num\_nota}, cod\_produto, qtidade, valor\_total \}$

$produto = \{ \underline{cod\_produto}, desc\_produto, valor\_produto \}$

3. Terceira forma normal:

A relação não está na 3FN pois existem atributos com dependência transitiva:

$nota\_fiscal = \{ \underline{num\_nota}, cod\_cliente, data\_nota\_fiscal \}$

$cliente = \{ \underline{cod\_cliente}, rua\_cliente, bairro\_cliente, cep\_cliente, cidade\_cliente, telefone\_cliente \}$

$nota\_produto = \{ \underline{num\_nota}, cod\_produto, qtidade, valor\_total \}$

$produto = \{ \underline{cod\_produto}, desc\_produto, valor\_produto \}$

**Temos:**

$Dom(cliente.cod\_cliente) = Dom(nota\_fiscal.cod\_cliente)$

$Dom(nota\_fiscal.num\_nota) = Dom(nota\_produto.num\_nota)$

$Dom(produto.cod\_produto) = Dom(nota\_produto.cod\_produto)$

## 5.9 Exercícios

1. Identifique, nas relações abaixo, as dependências funcionais e se há violação de alguma forma normal (1, 2 e 3), se houver, normalize-as. Obs: Os campos que estão entre (parênteses) , indicam campos multivalorados.
  - (a) Funcionario={RG, nome, endereco, dada\_nasc.}
  - (b) Projeto={Codigo, descricao, (cidade),ano}
  - (c) Atleta={ID\_Atleta, Nome, numero\_entidade\_patricinadora, ender\_entidade\_patrocinadora, sexo, (numero\_documento, orgao\_expedidor\_documento, data\_expedicao\_documento)}
  - (d) Pedido= {Numero, data, item, descricao\_item, quantidade, valor\_do\_pedido}
  - (e) Aluno={ RA, nome\_aluno, Endereco, cod\_professor, nome\_professor, cod\_disciplina, descricao\_disciplina}
  - (f) Emprestimo={ Cod\_agencia, cidade\_agencia, cic\_cliente, nome\_cliente, (endereco\_cliente), valor\_empréstimo}
  - (g) Salario={RG\_Func, nome\_funcion, (data\_salario, valor)}
  - (h) Gerencia={Cod\_Depart, RG\_Gerente, descricao\_depart}
2. Dê o conjunto de dependências funcionais para o esquema relacional R(A,B,C,D) com chave primária nos atributos AB. Sabe-se que a relação está na 1FN mas não está na 2FN.
3. Defina a terceira forma normal. Dê um exemplo de uma relação em 2FN mas não em 3FN. Transforme a relação em relação em 3FN.

# Capítulo 6

## Álgebra Relacional

### 6.1 Introdução

A álgebra relacional é uma linguagem de consulta procedural. Consiste em um conjunto de operações que usam uma ou mais relações como entrada e produz uma nova relação.

#### Importante

Toda as operações da álgebra sobre uma relação produz uma outra relação. Mesmo que a relação resultante seja vazia.

As principais operações são:

- Seleção
- Projeção
- União
- Diferença
- Inteseccção
- Produto Cartesiano
- Junção
- Divisão

**Antes** de estudarmos cada uma destas operações, considere as instâncias das relações que compõe o banco de dados:

cod_cli	nome_cli	rua	cidade	cod_agenc	nome_agenc	gerente	cidade
c1	Joao	Rua TT	Rio Preto	ag1	Sao Jose	Pedro	Sao Paulo
c3	Pedro	Rua A4	Sao Paulo	ag4	Botafogo	Manoel	Rio de Janeiro
c2	Marcos	Rua XY	Sao Paulo	ag3	Praiana	Dario	São Paulo
c4	Maria	Rua A	Belo Horizonte	ag2	Bom Retiro	Pedro	Belo Horizonte

Relação Cliente

cod_agenc	cod_cli	num_conta	saldo
ag1	c3	101	500
ag2	c2	215	780
ag1	c1	102	400
ag3	c4	305	350
ag3	c3	105	230

Relação Conta

Relação Agência

cod_agenc	cod_cli	num_emprest	valor
ag3	c4	e100	1500
ag2	c2	e500	800
ag2	c2	e550	100
ag1	c1	e230	1200
ag3	c1	e150	150

Relação Empréstimo

## 6.2 Operação Selecionar ( $\sigma$ )

A operação Selecionar seleciona tuplas que satisfazem um dado predicado.

É representada pela letra grega Sigma ( $\sigma$ ).

**Considere** a consulta:

Encontre os empréstimos feito na agência “ag2”.

A resposta em álgebra relacional seria:

$\sigma_{cod\_agenc="ag2"}(emprestimo);$

O resultado desta consulta traria as seguintes tuplas:

cod_agenc	cod_cli	num_emprest	valor
<b>ag2</b>	<b>c2</b>	<b>e500</b>	<b>800</b>
<b>ag2</b>	<b>c2</b>	<b>e550</b>	<b>100</b>

**Podemos** querer saber todas as contas que possuem saldo maior do que 400 escrevendo:

$\sigma_{saldo>400}(conta);$

cod_agenc	cod_cli	num_conta	saldo
ag1	c3	101	500
ag2	c2	215	780

**Geralmente**, permitimos as comparações =,  $\neq$ , <,  $\leq$ , >,  $\geq$ .

**Além** disso, diversos predicados podem ser combinados em um predicado maior usando os conectivos **AND** ( $\wedge$ ) e **OR** ( $\vee$ ).

**Por** exemplo, para encontrar os empréstimos maiores do que 500 e realizados na agência “ag2”.

$\sigma_{cod\_agenc="ag2" \wedge valor > 500}(emprestimo);$

cod_agenc	cod_cli	num_emprest	valor
ag2	c2	e500	800

## 6.3 A Operação Projetar ( $\Pi$ )

**Representada** pela letra grega pi ( $\Pi$ ), a operação projetar seleciona (projeta) atributos específicos de uma relação. O resultado da projeção é uma nova relação com os atributos selecionados.

**Suponha** que desejemos saber o nome de todos os clientes de um banco:

$$\Pi_{\text{nome\_cli}}(\text{cliente});$$

**Teríamos** como resposta:

nome_cli
Joao
Pedro
Marcos
Maria

**Agora** queremos saber o nome dos gerentes e suas respectivas agências:

$$\Pi_{\text{gerente, nome\_agenc}}(\text{agencia});$$

gerente	nome_agenc
Pedro	Sao Jose
Manoel	Botafogo
Dario	Praiana
Pedro	Bom Retiro

**Suponha** que desejemos saber o nome de todos os gerentes, sem saber suas agências:

$$\Pi_{\text{gerente}}(\text{agencia});$$

gerente
Pedro
Manoel
Dario

**Note** que apareceu apenas um gerente de nome **Pedro**. Isto se dá ao fato de que para a álgebra, uma relação é um conjunto de elementos, e em um conjunto **não existem valores repetidos**.

**Podemos** combinar as operações de projeção com seleção.

**Considere** a consulta no qual desejemos os nomes dos clientes que residem em São Paulo:

$$\Pi_{\text{nome\_cli}}(\sigma_{\text{cidade}=\text{"Sao Paulo"}}(\text{cliente}));$$

nome_cli
Pedro
Marcos

**Ou** senão o código da agência e o código do cliente cujo saldo da conta seja inferior ou igual a 400:

$\Pi_{cod\_agenc, cod\_cli}(\sigma_{saldo \leq 400}(saldo));$

cod_agenc	cod_cli
ag1	c1
ag3	c4
ag3	c3

### IMPORTANTE

As operações de Projetar e Selecionar são consideradas *unárias*. Pois operam sobre apenas uma única relação.

As operações estudadas a seguir devem operar sobre relações **compatíveis em domínio**, ou seja, devem possuir o mesmo número de atributos e os atributos nas colunas correspondentes devem vir do mesmo domínio.

## 6.4 A Operação União ( $\cup$ )

A união de duas relações é formada pela adição das tuplas de uma relação às tuplas de uma segunda relação, para produzir uma terceira.

É representada, como na teoria dos conjuntos, pelo símbolo  $\cup$ .

**Como** exemplo, queremos saber todos os clientes (cod\_cli) que possuem contas ou que fizeram empréstimos na agência “ag3”. Existe duas maneiras de realizar esta consulta:

Maneira 1:

$R1 = \Pi_{cod\_cli}(\sigma_{cod\_agenc = "ag3"}(conta));$

$R2 = \Pi_{cod\_cli}(\sigma_{cod\_agenc = "ag3"}(emprestimo));$

$R3 = R1 \cup R2;$

Maneira 2:

$\Pi_{cod\_cli}(\sigma_{cod\_agenc = "ag3"}(conta)) \cup \Pi_{cod\_cli}(\sigma_{cod\_agenc = "ag3"}(emprestimo));$

Resultado

cod_cli
c4
c1
c3

## 6.5 A Operação Intersecção de Conjuntos ( $\cap$ )

Esta operação produz atua sobre duas relação compatíveis em domínio e produz uma terceira contendo as tuplas que aparecem simultaneamente nas duas relações. É representada pelo símbolo  $\cap$ .

**Consulta:** Encontre os codigos dos cliente que possuem contas e que fizeram empréstimos.

$$R1 = \Pi_{cod\_cli}(conta);$$

$$R2 = \Pi_{cod\_cli}(emprestimo);$$

$$R3 = R1 \cap R2;$$

cod_cli
c4
c2
c1

## 6.6 A Operação Diferença de Conjuntos (-)

A diferença de duas relações é uma terceira relação contendo as tuplas que ocorrem na primeira relação mas não ocorrem na segunda.

É representado pela símbolo -.

**Considere** que desejemos encontrar todos os clientes (cod\_cli) que possuem contas mas não fizeram empréstimos:

$$R1 = \Pi_{cod\_cli}(conta);$$

$$R2 = \Pi_{cod\_cli}(emprestimo);$$

$$R3 = R1 - R2;$$

cod_cli
c3

As operações a seguir não necessitam ser **compatíveis em domínio**.

## 6.7 A Operação Produto Cartesiano (X)

O produto de duas relações é a concatenação de todas as tuplas de uma relação com todas as tuplas de uma segunda relação.

O produto da relação A (tendo  $m$  tuplas) com a relação B (tendo  $n$  tuplas) é um relação contendo  $m$  vezes  $n$  tuplas.

A representação do produto cartesiano é feito pelo símbolo X.

**Consulta:** Encontre o nome dos clientes e o número de suas respectivas contas bancárias que possuem saldo maior do que 450:

$$R1 = \sigma_{saldo > 450}(conta);$$

$$R2 = cliente \times R1;$$

cliente. cod_cli	cliente. nome_cli	cliente. rua	cliente. cidade	conta. cod_agenc	conta. cod_cli	conta. num_conta	conta. saldo
c1	Joao	Rua TT	Rio Preto	ag1	c3	101	500
c1	Joao	Rua TT	Rio Preto	ag2	c2	215	780
c3	Pedro	Rua A4	Sao Paulo	ag1	c3	101	500
c3	Pedro	Rua A4	Sao Paulo	ag2	c2	215	780
c2	Marcos	Rua XY	Sao Paulo	ag1	c3	101	500
c2	Marcos	Rua XY	Sao Paulo	ag2	c2	215	780
c4	Maria	Rua A	Belo Horizonte	ag1	c3	101	500
c4	Maria	Rua A	Belo Horizonte	ag2	c2	215	780

$R3 = \Pi_{nome\_cli, num\_conta}(\sigma_{cliente.cod\_cli = conta.cod\_cli}(R2));$

nome_cli	num_conta
c3	101
c2	215

## 6.8 A Operação Junção Natural ( $\bowtie$ )

**Percebemos** na operação de produto cartesiano que foi necessário realizar uma seleção dos dados ( $R3 = \Pi_{nome\_cli, num\_conta}(\sigma_{cliente.cod\_cli = conta.cod\_cli}(R2));$ ) em cima do produto entre as relações *conta* e *R1*. Além do mais, um problema que ocorre no produto cartesiano é o consumo excessivo de memória. Imagine duas relação onde uma contém 3000 tuplas de 100 bytes cada e outra com 5000 tuplas de 200 bytes cada. Seria necessário 4.5 mb de memória para armazenar a relação resultante.

A operação **junção natural** as operações de seleção e produto cartesiano em uma única operação.

É representada pelo símbolo  $\bowtie$ .

A operação de junção natural forma um produto cartesiano de seus dois argumentos, faz uma seleção forçando uma igualdade sobre os atributos que aparecem em ambas relações e finalmente remove colunas duplicadas.

**Vamos** considerar novamente a consulta no qual desejemos saber o nome dos clientes e suas respectivas contas bancárias que possuem saldo maior que 450. Pode ser realizado de duas maneiras:

Maneira 1:

$R1 = \sigma_{saldo > 450}(conta);$

$R2 = \Pi_{nome\_cli, num\_conta}(cliente \bowtie_{num\_cli = num\_cli}(R1));$

Maneira 2:

$R1 = \sigma_{saldo > 450}(conta);$

$R2 = \Pi_{nome\_cli, num\_conta}(cliente \bowtie R1)$

//É importante frisar que desta maneira, os atributos de junção estão implícitos no símbolo de junção;

nome_cli	num_conta
c3	101
c2	215

## 6.9 A Operação Divisão ( $\div$ )

A operação divisão, representada por  $\div$ , retorna as tuplas da relação A que se relaciona com todas as tuplas da relação B ao mesmo tempo. Não é uma operação importante como as anteriores.

**Considere** a consulta de que quer encontrar todos os clientes (cod\_cli) que tem pelo menos uma conta em todas as agências de “São Paulo”:

$$R1 = \Pi_{\text{cod\_agenc}}(\sigma_{\text{cidade} = \text{''Sao Paulo''}}(\text{agencia}));$$

cod_agenc
ag1
ag3

$$R2 = \Pi_{\text{cod\_cli}, \text{cod\_agenc}}(\text{conta});$$

cod_cli	cod_agenc
c3	ag1
c2	ag2
c1	ag1
c4	ag3
c3	ag3

$$R3 = \Pi_{\text{cod\_cli}, \text{cod\_agenc}}(R2) \div \Pi_{\text{cod\_agenc}}(R1);$$

cod_cli	cod_agenc
c3	ag1
c3	ag3

## 6.10 Exercícios

Considere as seguintes relações para resolução dos exercícios:

Relação Empregado

Nome	RG	CIC	Depto	RG_Gerent	Salário
João Luiz	101010	111111	1	NULO	3.000,00
Fernando	202020	222222	2	101010	2.500,00
Ricardo	303030	333333	3	101010	2.300,00
Jorge	404040	444444	2	202020	4.200,00

Relação Departamento

DNome	DNum	RG_Gerent
Contabil	1	101010
Eng. Civil	2	303030
Eng. Mecân.	3	202020

Relação Projeto

PNum	PValor	PLocal
5	5.000,00	Sao Paulo
10	7.800,00	Rio Preto
20	9.500,00	Campinas

Relação Dependente

RGResp	DepenNome	Data_Nasc	Parent.	Sexo
101010	Jorge	27/12/94	Filho	M
101010	Luiz	18/11/93	Filho	M
202020	Fernanda	14/02/99	Filha	F
202020	Angelo	10/02/95	Filho	M
303030	Andreia	01/05/00	Filho	M

Relação Depart\_Projeto

Dept_Num	Num_Proj
2	5
3	10
2	20

Relação Emp\_Proj

RG_Emp	Num_Proj	Horas
202020	5	10
202020	10	25
303030	5	35
404040	20	30

1. Obtenha as seguintes consultas.

- consulta =  $\sigma$  dnum  $\geq$  1 (departamento);
- consulta =  $\sigma$  rg\_gerent = 101010 (empregado);
- consulta =  $\Pi$  nome, rg ( $\sigma$  salario  $\leq$  2500 (empregado));
- consulta =  $\Pi$  dept\_num (depart\_proj);
- consulta =  $\Pi$  rg\_emp, num\_proj ( $\sigma$  horas = 35 (emp\_proj));
- consulta =  $\Pi$  depennome ( $\sigma$  rgresp = 202020 ( $\sigma$  sexo = "m" (dependentes))));
- consulta1 =  $\Pi$  nome (empregado);  
consulta2 =  $\Pi$  depennome (dependentes);  
consulta3 = consulta1  $\cup$  consulta2;
- consulta = emp\_proj X projeto;
- consulta = empregado  $\bowtie$  rg = rgresp (dependentes)

2. Forme as expressões em álgebra relacional dos seguintes enunciados:

- Liste o nome de todos funcionários
- Liste o valor e a localização de todos os projetos
- Liste o nome de todos os departamentos
- Encontre o nome, rg do responsável e a data de nascimento de todos os dependentes
- Encontre o nome de todos os empregados que possuem dependentes

- (f) Encontre o nome e salário dos funcionários que recebem acima de 2000
  - (g) Consulte os nomes dos empregados que trabalham em todos os projetos controlados pelo departamento 2.
  - (h) Encontre os nomes dos funcionários que não possuem dependentes
  - (i) Para cada nome de empregado, obtenha o departamento que este gerencia.
  - (j) Com o nome do empregado Ricardo, monte uma pesquisa utilizando a álgebra relacional que mostre os projetos que o funcionário participa.
  - (k) Monte uma pesquisa em álgebra relacional utilizando o mesmo enunciado acima (enunciado L), porém não utilizando a relação empregado\_projeto.
  - (l) Liste o nome dos gerentes que possuem dependentes.
  - (m) Selecione todos os empregados que trabalham no departamento número 2 ou que supervisionam empregados que trabalham no departamento número 2.
3. Elabore a álgebra relacional para as consultas baseadas nas seguintes relações. O Aluno pode instanciar as relações e inserir alguns dados.
- Reside = {nome\_pessoa, rua, cidade}  
Trabalha = {nome\_pessoa, nome\_companhia, salário}  
Localizado = {nome\_companhia, cidade}  
Gerencia = {nome\_pessoa, nome\_gerente}
- (a) Dê todos os funcionários que moram na mesma cidade da companhia em que trabalham.
  - (b) Dê todos os funcionários que vivem na mesma cidade que seu gerente.
  - (c) Dê todos os empregados que não trabalham para a empresa "Kangle Corporation"