

Bases de Dados

Módulo 16a: Projeto de Banco de Dados (Lógico)

Prof. André Bruno de Oliveira

24/05/24 11:33

Projeto de Banco de dados

Neste módulo serão apresentadas as técnicas utilizadas para a elaboração de projetos de banco de dados relacionais juntamente com a teoria da normalização, ambas fazem parte do projeto de desenvolvimento de sistemas e aplicativos profissionais.

(continuação do módulo anterior)

3 Projeto Lógico

- O projeto lógico consiste na conversão do modelo conceitual para um esquema relacional de um determinado SGBD (SQLite, Oracle, MySQL, etc.). Este processo é chamado de Mapeamento ER-Relacional (ER-to-Relational Mapping).
- O objetivo final do projeto lógico é a obtenção de um **script DDL** para a geração de um conjunto de tabelas, campos e restrições de integridade em um SGBD relacional específico.

3 Projeto Lógico

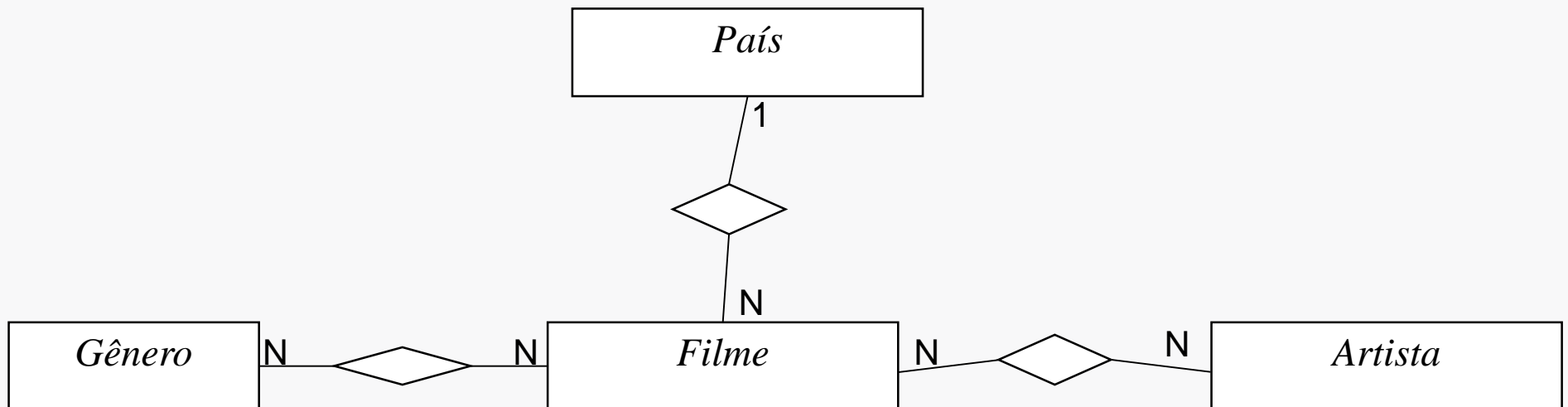
- O projeto lógico pode ser elaborado através de uma receita padrão, que consiste na utilização de um conjunto de regras de mapeamento. A próxima subseção descreve as regras mais importantes.

3.1 Mapeamento ER-Relacional

- A seguir apresentam-se as principais regras de mapeamento um passo-a-passo simplificado para converter um DER em um projeto lógico.

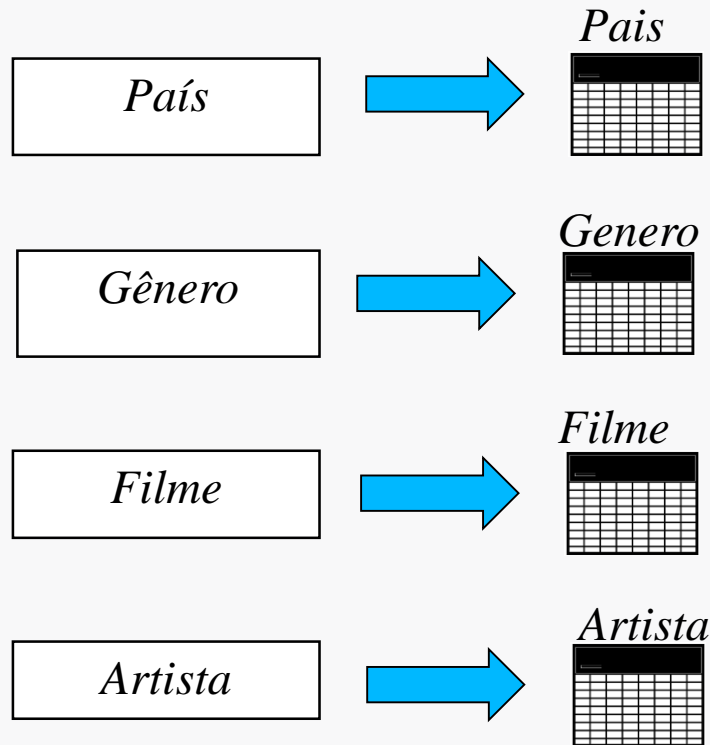
3.1 Mapeamento ER-Relacional

- **REGRA 1** –Mapeamento de Entidades
- O primeiro passo é realizar o mapeamento das entidades através da aplicação de uma regra muito simples: toda entidade transforma-se numa tabela na base de dados.
- Tome como exemplo o DER do site sobre cinema. Este diagrama possui quatro entidades: *País*, *Gênero*, *Filme* e *Artista*. Seguindo a regra recém-apresentada, cada uma deverá ser mapeada para uma tabela específica no BD



3.1 Mapeamento ER-Relacional

- **REGRA 1** –Mapeamento de Entidades
- Cada uma entidade é mapeada para uma tabela específica no BD.



3.1 Mapeamento ER-Relacional

- Com relação aos atributos das tabelas, é importante considerar que no nível conceitual não precisamos nos preocupar muito com os seus tipos ou dar um tratamento elementar para essa questão (ex.: utilizar apenas “alfanumérico”, “inteiro” e “real”).

3.1 Mapeamento ER-Relacional

- Entretanto, quando estamos no nível lógico já é preciso definir os tipos de uma forma mais adequada, de acordo com as opções oferecidas pelo SGBD.
- A tabela apresentada a seguir inclui exemplos de tipos de dados encontrados na maioria dos SGBDs relacionais, como Oracle, SQL Server, PostgreSQL, MySQL, etc.

3.1 Mapeamento ER-Relacional

Tipo de Dado	Descrição
CHAR(n), CHARACTER(n)	Texto de caracteres com tamanho fixo igual a “n”. Utilizado para colunas cujo tamanho é conhecido e fixo, como “sexo” (1 caractere), “cep” (8 caracteres), “cnpj” (14 caracteres), etc.
VARCHAR(n)	Texto com tamanho variável. O espaço que não for utilizado não ocupa espaço no banco de dados. Este campo é utilizado para colunas cujo tamanho é muito variável, como “endereço”, “nome” e “razão social”.
INT, INTEGER	Tipos numéricos utilizados para representar números inteiros.
NUM, REAL, FLOAT	Tipos numéricos utilizados para representar números reais.
NUMBER	No SGBD Oracle, pode ser utilizado para armazenar números inteiros ou reais.
BOOLEAN	Tipo booleano, onde um valor 0 é considerado FALSE e 1 é considerado TRUE.
DATE	Para armazenar uma data (dia/mês/ano).
DATETIME	Para armazenar data e hora com precisão de segundos (ou milissegundos, dependendo do SGBD).
TIMESTAMP, TIME	Para armazenar data e hora com precisão de milissegundos (ou superior, dependendo do SGBD).
CLOB, TEXT	Pode armazenar textos volumosos (ex.: com até 4GB).
BLOB	Pode armazenar arquivos diversos (JPG, AVI, XLS, DOC, etc).

3.1 Mapeamento ER-Relacional

- Como exemplo, veja um possível mapeamento da entidade *País* do modelo conceitual para a tabela *Pais* do modelo lógico.

Tabela <i>Pais</i>			
Nome da Coluna	Tipo	PK?	NOT NULL
sigla	CHAR(2)	X	X
nome	VARCHAR(30)		X

3.1 Mapeamento ER-Relacional

- O mapeamento inclui a especificação do nome da tabela e de todos os seus atributos. Para cada atributo, especifica-se o **nome**, **tipo**, se **corresponde à chave primária** e se é um campo **NOT NULL**. Sendo assim, veja que a tabela *Pais* possui dois campos: “sigla”, do tipo CHAR(2), chave primária e NOT NULL; e “nome” do tipo VARCHAR(30), NOT NULL.

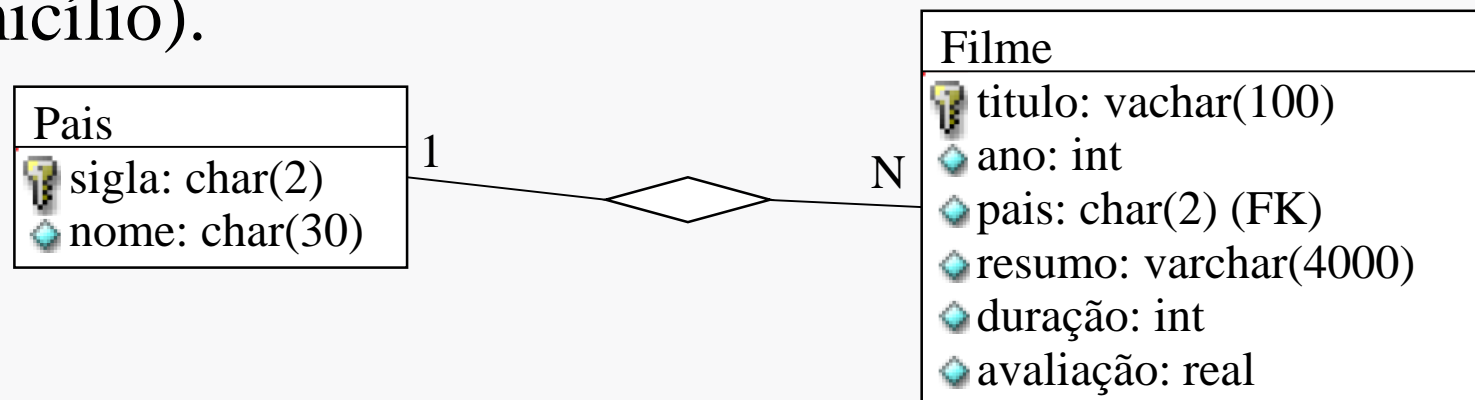
Tabela <i>Pais</i>			
Nome da Coluna	Tipo	PK?	NOT NULL
sigla	CHAR(2)	X	X
nome	VARCHAR(30)		X

3.1 Mapeamento ER-Relacional

- **Regra 2 –Mapeamento de Relacionamentos 1 x N**
- Esta classe de relacionamento é a mais comum no mundo real.
- Vimos anteriormente, que em uma relação 1 x N entre as entidades A e B , um membro de A pode relacionar-se com muitos membros da entidade B , mas cada membro de B somente pode estar relacionado a um membro da entidade A .

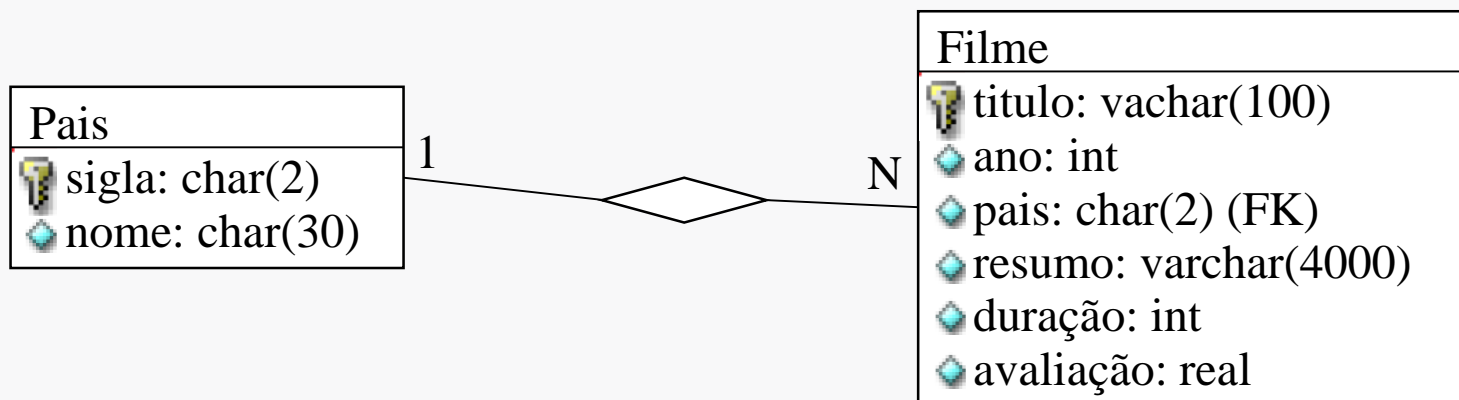
3.1 Mapeamento ER-Relacional

- **Regra 2 –Mapeamento de Relacionamentos 1 x N**
- No DER de filmes, um exemplo deste tipo de relacionamento é o que ocorre entre *País* e *Filme*: um país pode produzir muitos filmes e um filme é produzido em um único país.
- Outro exemplo é o relacionamento entre *Domicílio* e *Pessoa*, apresentado na última aula (um domicílio pode ter muitas pessoas e uma pessoa mora em um único domicílio).



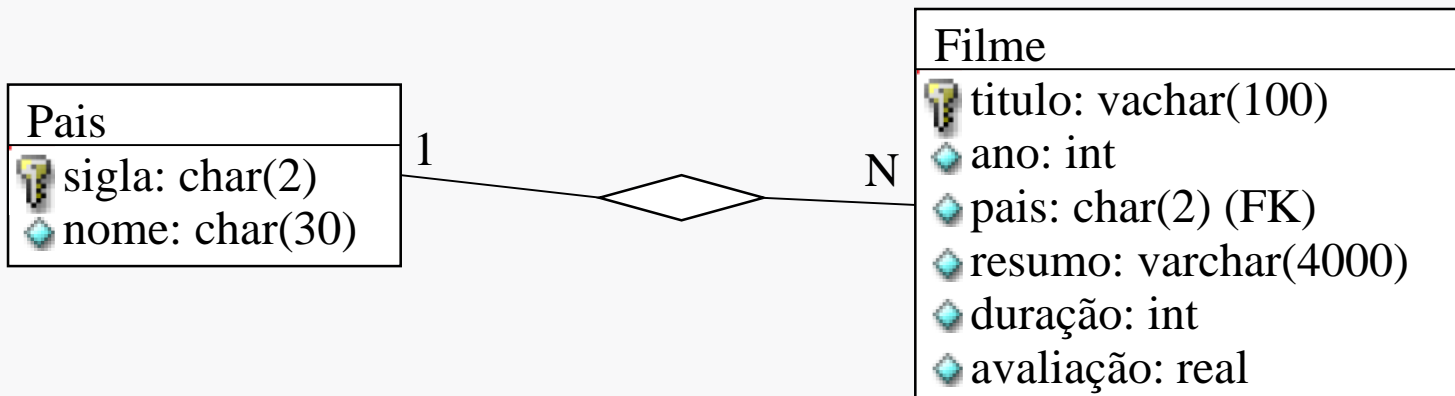
3.1 Mapeamento ER-Relacional

- **Regra 2 –Mapeamento de Relacionamentos 1 x N**
 - A regra de mapeamento conceitual-lógico para relacionamentos 1 x N é a seguinte:
 1. Identificar a relação *A* cujos membros podem se relacionar com muitos membros de *B*.
 2. Incluir como chave estrangeira em *B*, a chave primária de *A*.



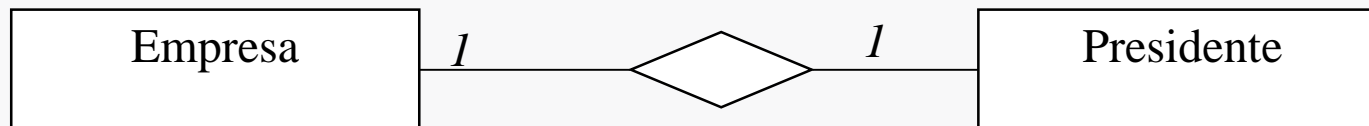
3.1 Mapeamento ER-Relacional

- **Regra 2 –Mapeamento de Relacionamentos 1 x N**
- Veja que a chave primária da tabela *Pais* teve que ser transportada como chave estrangeira para a tabela *Filme*, para que o relacionamento entre estas duas tabelas pudesse ser implementado.
- Observe que na tabela *Pais*, o nome do campo que armazena a sigla do país é “*sigla*”, enquanto na tabela *Filme* o nome deste campo é “*pais*”. O que importa que é as duas variáveis possuem as mesmas características.



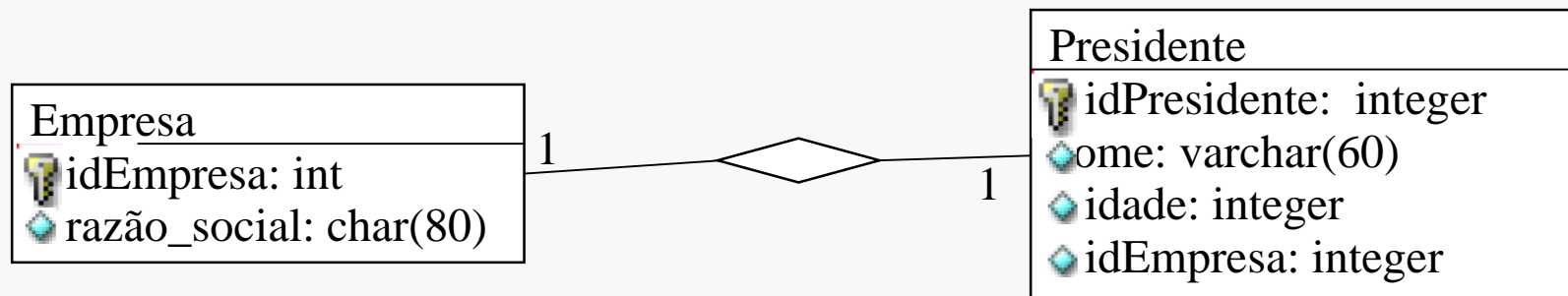
3.1 Mapeamento ER-Relacional

- **Regra 3 – Realizar o Mapeamento de Relacionamentos 1 x 1**
- Esta classe de relacionamento ocorre com muito menos frequência na prática. Em uma relação 1 x 1 entre as entidades *A* e *B*, cada membro de *A* pode relacionar-se com no máximo um membro de *B*, e vice-versa.



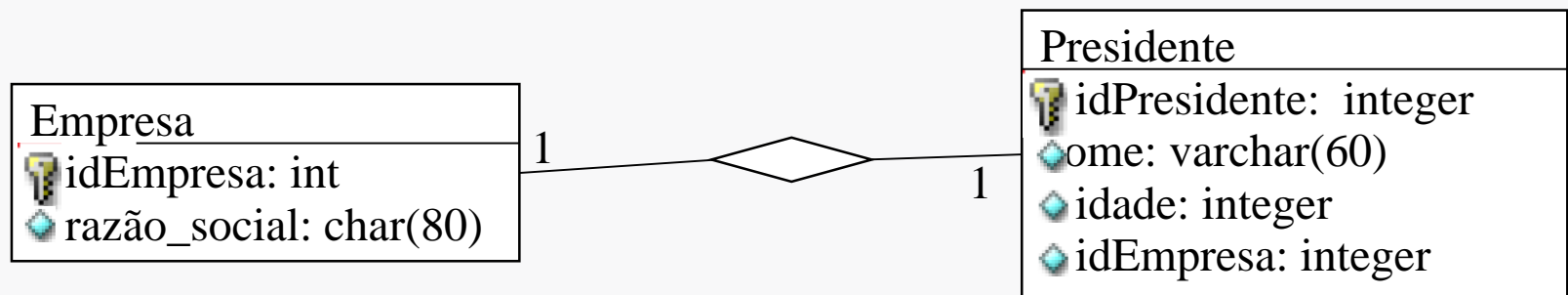
3.1 Mapeamento ER-Relacional

- **Regra 3 – Realizar o Mapeamento de Relacionamentos 1 x 1**
- A abordagem mais comum para realizar este tipo de mapeamento é implementar uma chave estrangeira. Mas neste caso, diferentemente do que ocorre no mapeamento 1 x N, qualquer uma das tabelas poderá receber a chave estrangeira.



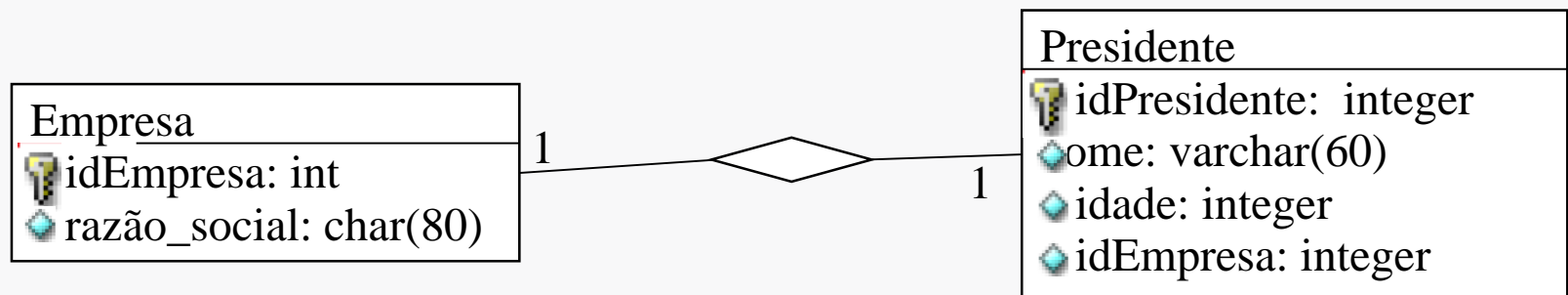
3.1 Mapeamento ER-Relacional

- **Regra 3 – Realizar o Mapeamento de Relacionamentos 1 x 1**
- Veja este exemplo onde a PK de *Empresa* foi exportada como FK para *Presidente*. Como cada linha de *Presidente* corresponde a um presidente de uma empresa específica e o código da empresa (idEmpresa) é suficiente para identificá-la.



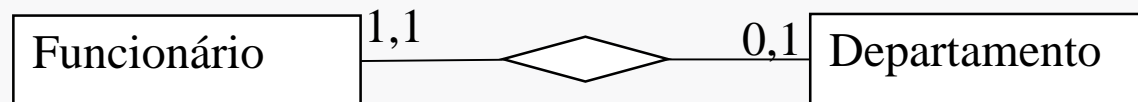
3.1 Mapeamento ER-Relacional

- **Regra 3 – Realizar o Mapeamento de Relacionamentos 1 x 1**
- Pode-se notar que no mapeamento de relacionamentos 1 x 1, a FK pode ser implementada em qualquer uma das tabelas, já que isso não ocasionará qualquer problema relacionado ao armazenamento de dados no BD (diferentemente do que ocorre nos relacionamentos 1 x N, onde existe uma tabela certa para receber a FK).



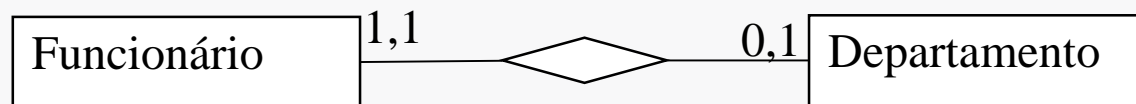
3.1 Mapeamento ER-Relacional

- **Regra 3 – Realizar o Mapeamento de Relacionamentos 1 x 1**
- No entanto, pode-se ser interessante observar a **cardinalidade mínima** para determinar a “melhor tabela” para receber a FK.
- Por exemplo, o relacionamento que representa a associação entre *Departamento* e *Funcionário*. Cada departamento pode ser chefiado por apenas um funcionário:
 - Funcionário chefia Departamento.



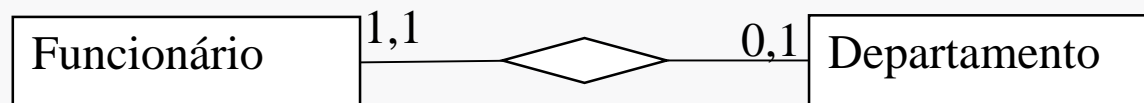
3.1 Mapeamento ER-Relacional

- **Regra 3 – Realizar o Mapeamento de Relacionamentos 1 x 1**
- Neste relacionamento, um departamento é chefiado por no mínimo 1 e no máximo 1 funcionário; e um funcionário pode chefiar no mínimo 0 e no máximo 1 departamento.



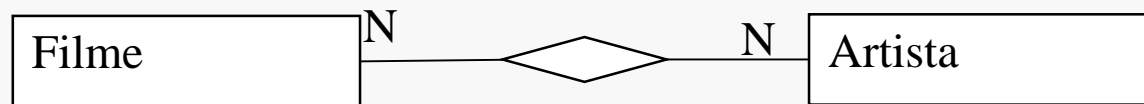
3.1 Mapeamento ER-Relacional

- **Regra 3 – Realizar o Mapeamento de Relacionamentos 1 x 1**
- Neste caso, a melhor solução é a FK ficar em departamento.
- Imagine que esta empresa tenha 100 funcionários e 5 departamentos. Se a FK ficar em funcionário (id do departamento como FK em funcionário), haverá 95% de registros com valor nulo (NULL) na FK, pois somente 5 funcionários serão chefes de departamento.



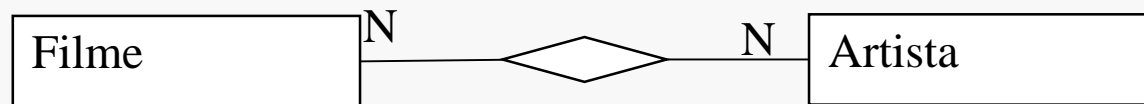
3.1 Mapeamento ER-Relacional

- **Regra 4 – Realizar o Mapeamento de Relacionamentos N x N**
- Neste grau de relacionamento, cada membro da entidade *A* pode relacionar-se com muitos membros da entidade *B* e vice-versa. Um exemplo é o relacionamento entre *Filme* e *Artista*: um filme pode ter muitos artistas. E um artista pode atuar em vários filmes.



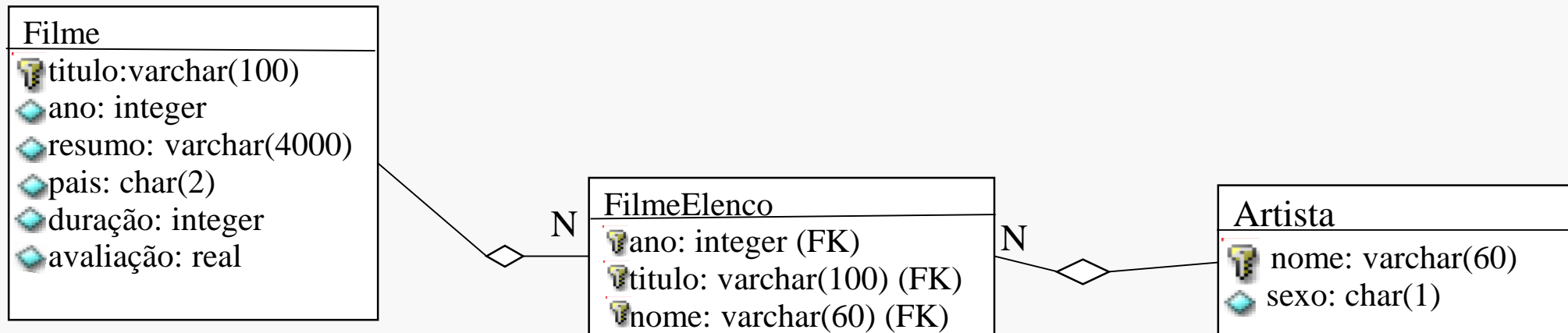
3.1 Mapeamento ER-Relacional

- **Regra 4 – Realizar o Mapeamento de Relacionamentos N x N**
- Para este tipo de relacionamento, a seguinte regra deve ser aplicada: **todo relacionamento N x N transforma-se em uma tabela no projeto lógico.** A tabela gerada pelo relacionamento N x N entre as entidades *A* e *B*, deve conter a chave primária da entidade *A* e a chave primária da entidade *B*.



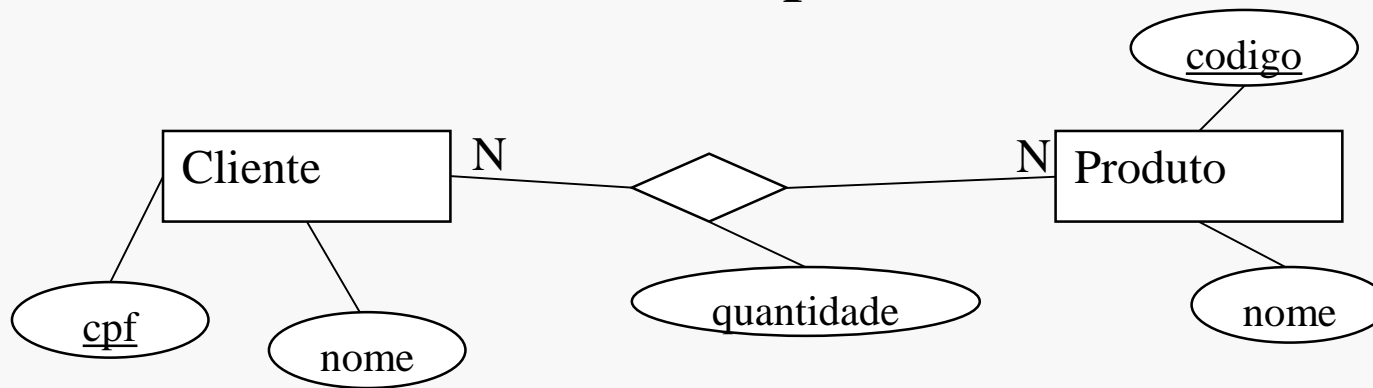
3.1 Mapeamento ER-Relacional

- **Regra 4 – Realizar o Mapeamento de Relacionamentos N x N**
- A tabela *FilmeElenco* do BD cinema surge do relacionamento de N x N entre *Filme* e *Artista*.
- A mesma regra vale para o relacionamento N x N entre *Filme* e *Gênero* do BD cinema.



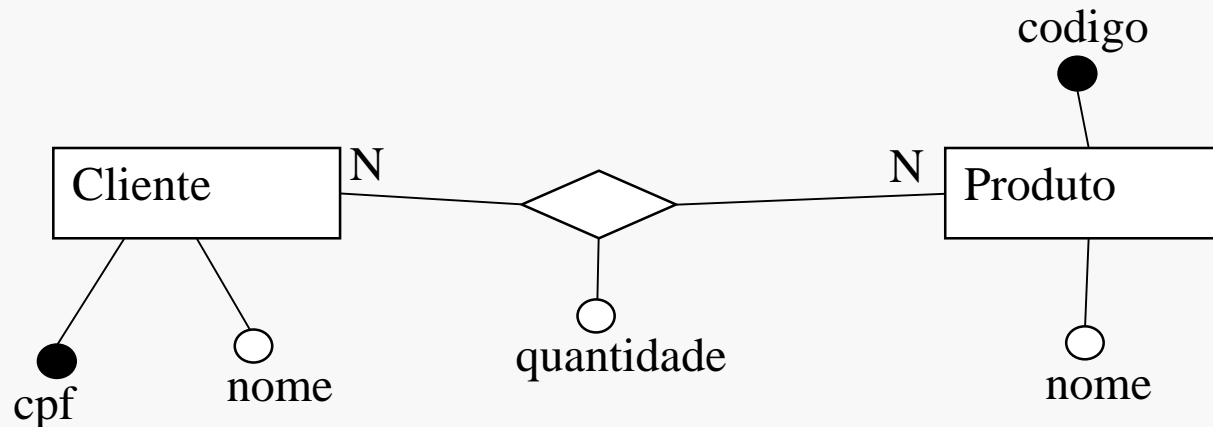
3.1 Mapeamento ER-Relacional

- **Relacionamentos N x N Podem Possuir Atributos**
- Relacionamentos N x N são muito especiais já que podem possuir atributos. Para ver isso na prática, veja esta figura a seguir onde o relacionamento “Cliente compra Produto” em um diagrama E-R. Trata-se de um relacionamento N x N, pois um cliente compra muitos produtos e um produto pode ser comprado por vários clientes. Neste relacionamento, o atributo “quantidade” pertence ao relacionamento compra.



3.1 Mapeamento ER-Relacional

- **Relacionamentos N x N Podem Possuir Atributos**



- Esquema relacional correspondente
 - *Cliente* (cpf, nome)
 - *Produto* (codigo, nome)
 - Compra (cpf, codigo, quantidade)

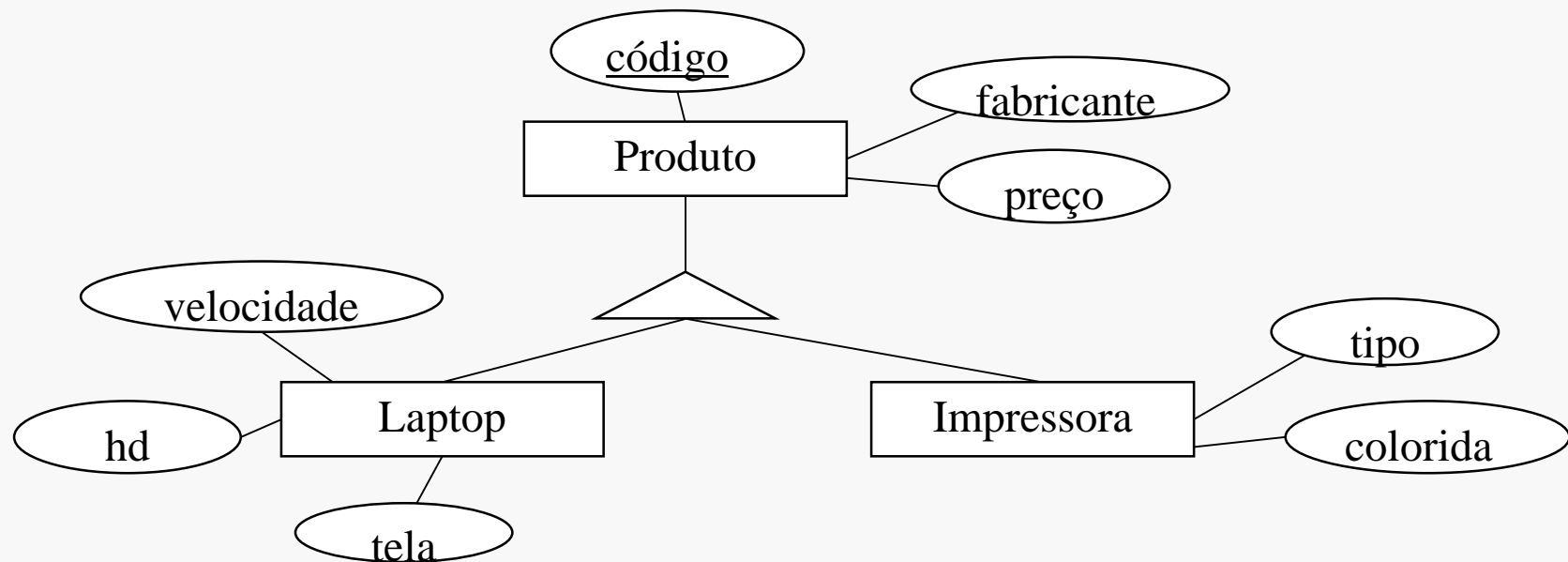
Parada para técnica para ir ao banheiro

3.2 Outras regras - Mapeamentos Especiais

- As Regras 1 a 4 tratam dos casos de mapeamento ER-Relacional mais comumente empregadas na prática. No entanto, relacionamentos especiais como **auto-relacionamentos** e **generalizações** também podem estar presentes em determinados modelos conceituais. Esta seção descreve a forma de mapear estes relacionamentos especiais.

3.2.1 Generalização e Especialização

- Considere o DER a seguir da entidade genérica *Produto* com especialização para *Laptop* e *Impressora*. Modelos como estes são conhecidos como *Enhanced Entity-Relationship Mode* (EER), relacionamento de entidade aprimorado .

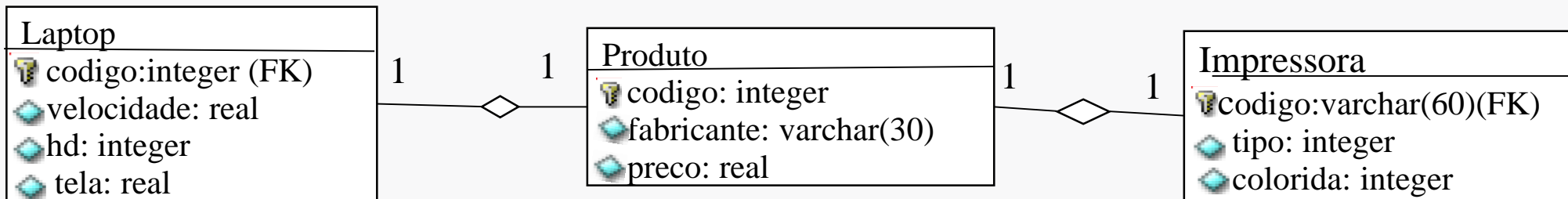
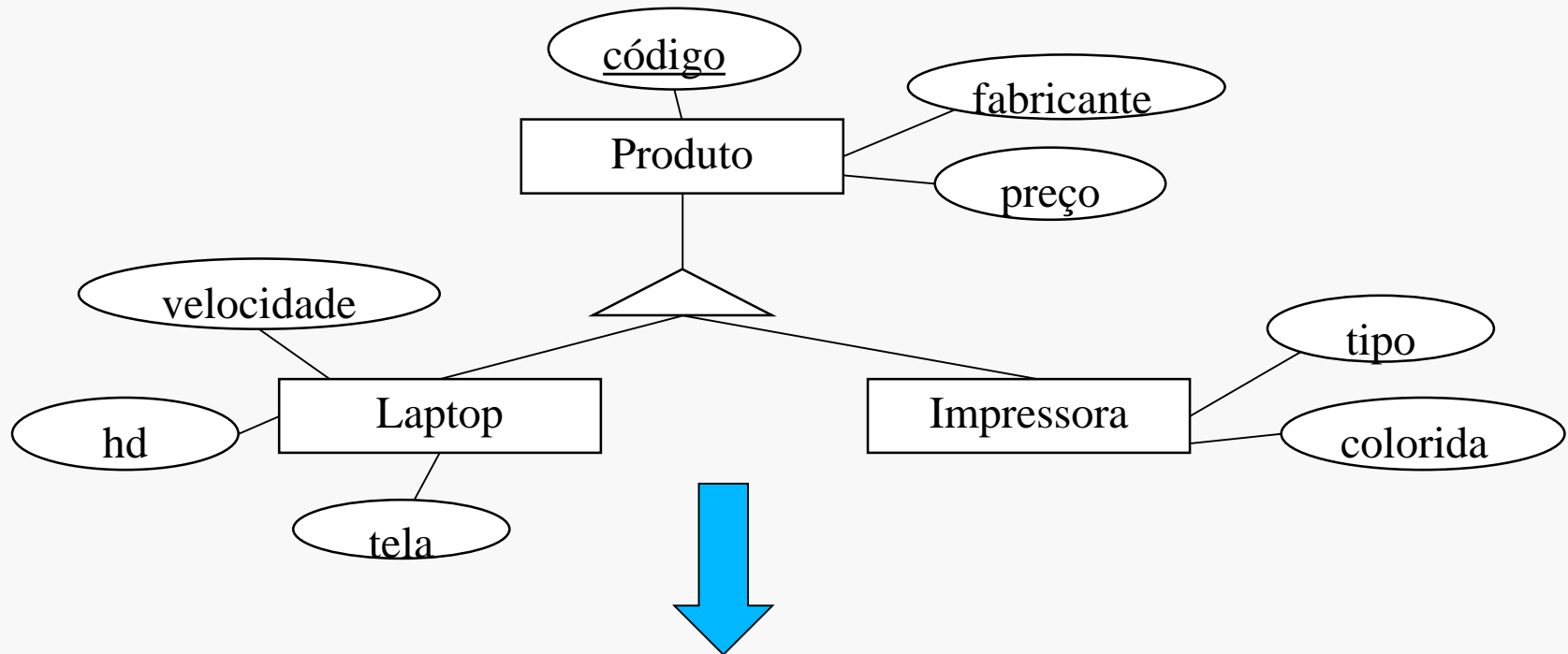


3.2.1 Generalização e Especialização

- Existem algumas técnicas distintas de realizar o mapeamento, sendo mais comum a utilização da seguinte abordagem em dois passos:
 - (i) criar uma tabela distinta para cada entidade (respeitando assim a regra 1);
 - (ii) implementar um relacionamento do tipo 1 x 1 entre cada entidade especializada e a entidade genérica. A PK da entidade genérica deverá então ser transportada como FK para cada entidade especializada.

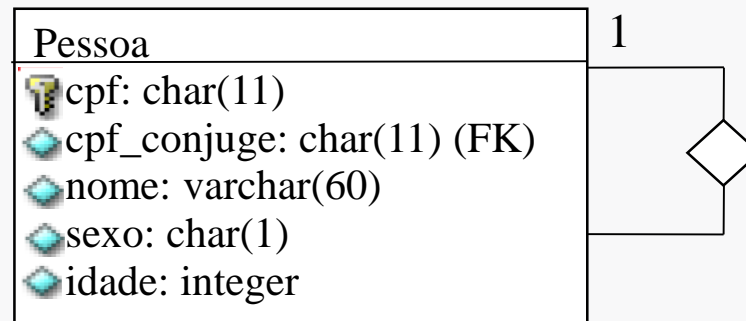
3.2.1 Generalização e Especialização

- Exemplo:



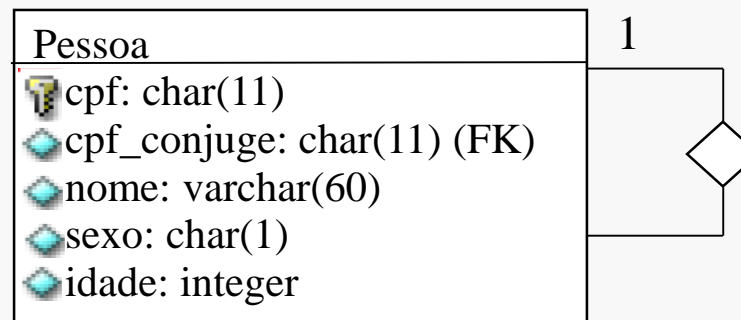
3.2.2 Auto-Relacionamento

- Uma entidade auto-relacionada é aquela cujos membros se relacionam com outros membros da própria unidade. Um exemplo deste tipo de relacionamento é apresentado a seguir.



3.2.2 Auto-Relacionamento

- Supondo que a entidade *Pessoa* seja definida por (cpf, nome, sexo, idade), a implementação deste auto-relacionamento pode ser realizada através da inclusão de um segundo campo de “cpf” na tabela *Pessoa*. Este campo será utilizado para armazenar o CPF do cônjuge de cada membro de *Pessoa*. Observe que “cpf_conjuge” é uma FK referenciando o campo “cpf” (PK) da própria tabela *Pessoa*.



3.2.3 Chave Substituta

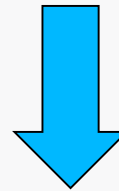
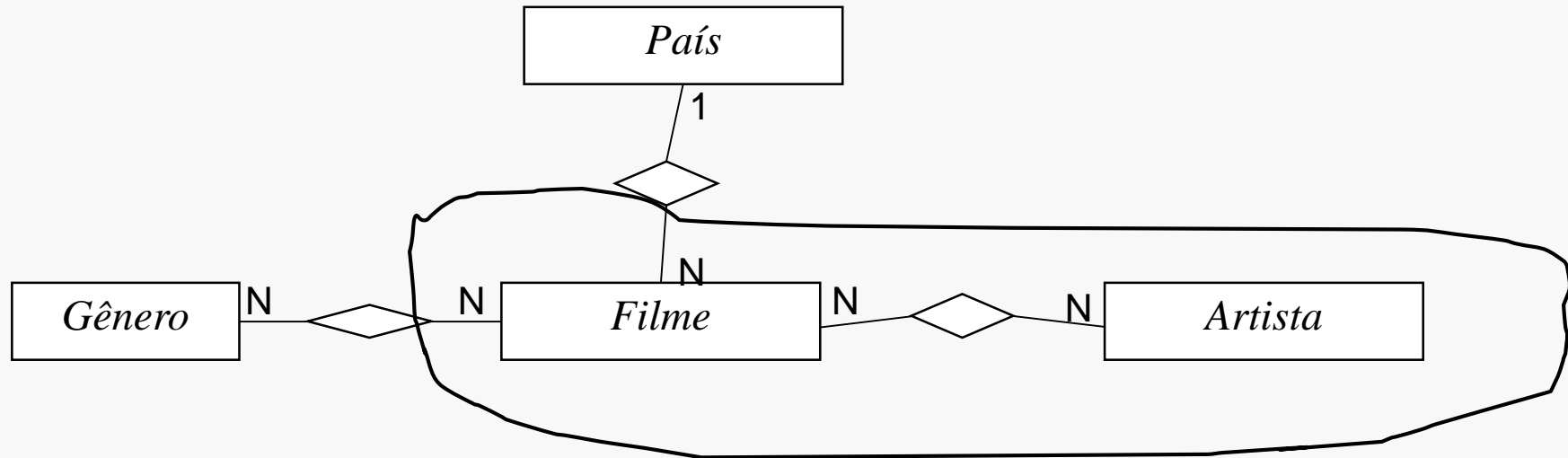
- Uma chave substituta (*surrogate key*) é um campo que não tem significado para o minimundo (não representa uma propriedade real de uma entidade) e que é introduzida artificialmente em uma tabela do modelo lógico para servir como uma chave primária.
- Mas por que isso é feito? Isso ocorre porque em muitas situações práticas, a chave primária “natural” de uma tabela é muito “complicada” ou consiste em um campo ou combinação de campos com tamanho muito grande.







3.2.3 Chave Substituta

- Por exemplo: em nosso BD de filmes, a chave primária de *Filme* é o “título” (VARCHAR(100)) e o “ano” (INTEGER). Já em *Artista*, a chave primária é “nome” (VARCHAR(60)).
- Não é bom ter uma PK onde um dos campos é um VARCHAR(100) ou VARCHAR(60), pois uma PK pode sempre ser transportada como uma FK para várias outras tabelas e isto acarretaria numa degradação da performance do BD, além de um aumento desnecessário no espaço ocupado.




3.2.3 Chave Substituta



- Exemplo: chave composta



Filme
 título: varchar(100)
 ano: integer
 resumo: varchar(4000)
 país: char(2)
 duração: integer
 avaliação: real

N

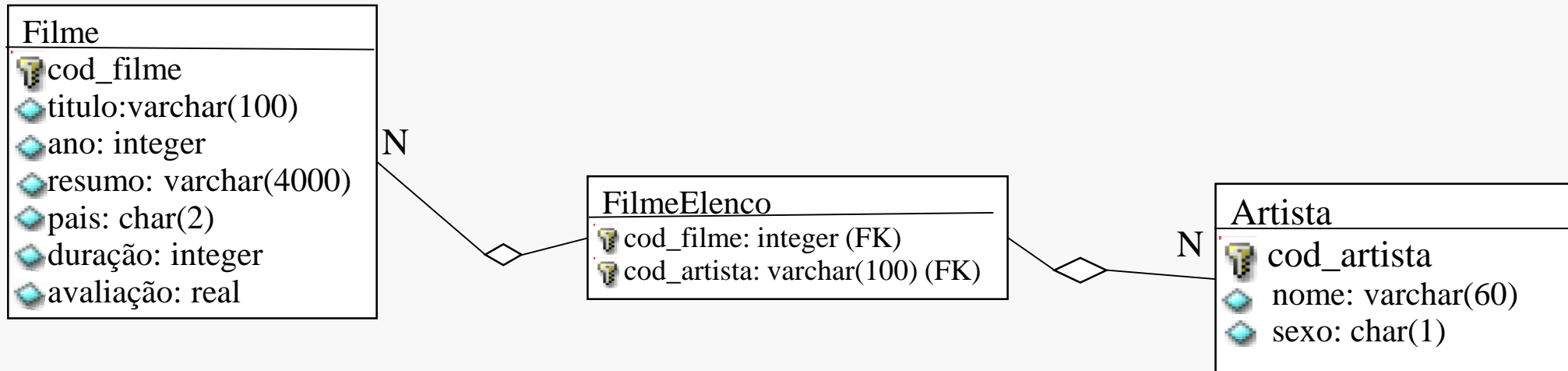
FilmeElenco
 ano: integer (FK)
 título: varchar(100) (FK)
 nome: varchar(60) (FK)

Artista
 nome: varchar(60)
 sexo: char(1)

N

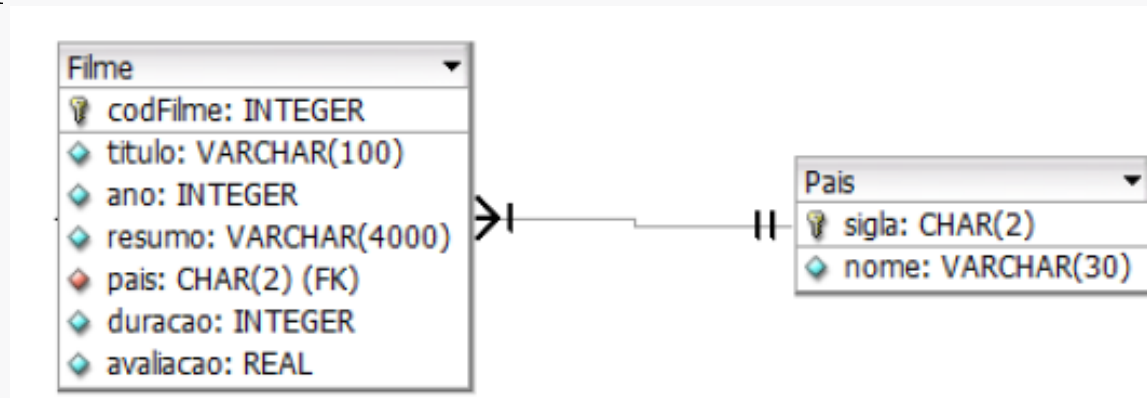
3.2.3 Chave Substituta

- Para resolver este problema, cria-se uma *surrogate key* chamada “CodFilme”, um simples número inteiro sequencial que passará a ser utilizado como PK de filme.
- A mesma coisa pode ser feita na tabela Artista.
- Exemplo: chave substituta.



3.2.4 Notação Pé-de-Galinha em um Modelo Lógico

- **Notação Pé-de-Galinha em um Modelo Lógico**
- Conforme apresentado anteriormente, muitos projetistas preferem utilizar a Notação Pé-de-Galinha.
- O exemplo na figura usa um relacionamento 1 X N entre *Filme* e *País*. No modelo lógico, um símbolo do **pé-de-galinha** ao lado de *Filme* indica que um membro de *País* pode estar associado a um ou muitos membros de *Filme*.
- O Símbolo dos dois traços ao lado de país indica que *Filme* pode estar associado a um membro de *País*.



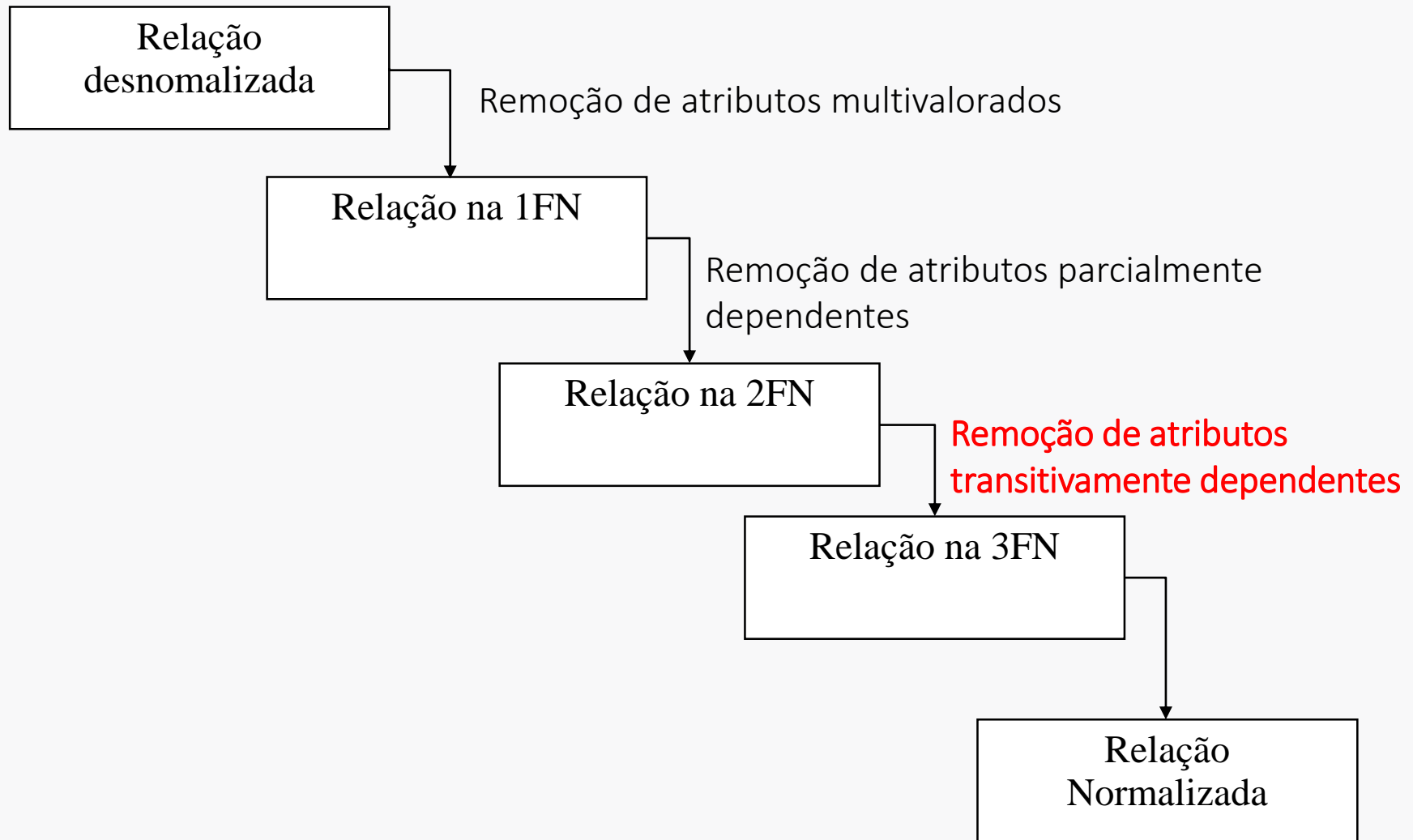
4 Desnormalização

- Objetivo: Otimizar a performance se consultas.
- Considere a tabela Cliente(idcliente, nome, bairro, cidade), da qual uma possível instância é apresentada com "idcliente" sendo a PK.

idcliente	nome	bairro	cidade
1	Carlos	Bangu	Rio de Janeiro
2	Rosana	Marinas	Angra dos Reis
3	Max	Areão	Taubaté
4	Marcos	Bauxita	Ouro Preto

- **Assumindo que não existam nomes de bairros iguais em cidades diferentes**, pode-se assumir que:

Normalização – (Lembrando)



4 Desnormalização

- $\text{idcliente} \rightarrow \{\text{nome, bairro, cidade}\}$, pois idcliente é PK.
- Só que bairro define cidade , assim pela regra não respeita a 3FN e viola a BCNF (Uma relação R está na FNBC se todos os seus determinantes são chaves candidatas).

idcliente	nome	bairro	cidade
1	Carlos	Bangu	Rio de Janeiro
2	Rosana	Marinas	Angra dos Reis
3	Max	Areão	Taubaté
4	Marcos	Bauxita	Ouro Preto

4 Desnormalização

- Pela teoria de normalização deve ser desmembrada em duas tabelas:
 - i) *Cliente* (idcliente, nome, bairro);
 - ii) *BairroCidade* (bairro, cidade).

Cliente

idcliente	nome	bairro	cidade
1	Carlos	Bangu	Rio de Janeiro
2	Rosana	Marinas	Angra dos Reis
3	Max	Areão	Taubaté
4	Marcos	Bauxita	Ouro Preto

4 Desnormalização

- No entanto, em situações como esta, muitas vezes os projetistas de BD optam por deliberadamente violar a 3FN ou a FNBC para deixar o projeto mais prático, mesmo que isto acarrete definição de uma tabela “desnormalizada”.
- Um única tabela não tem operação de JOIN e desmembrando haverá. O uso do JOIN num universo de milhões de linhas contribui para uma queda de performance.

Cliente

idcliente	nome	bairro	cidade
1	Carlos	Bangu	Rio de Janeiro
2	Rosana	Marinas	Angra dos Reis
3	Max	Areão	Taubaté
4	Marcos	Bauxita	Ouro Preto

4 Desnormalização

- Esta decisão de manter desnormalizado deve considerar que bairro tenha uma importância pouco relevante para o projeto, pois a grafia do nome pode variar. Imagine se alguém digitar “C. Nova” e outra pessoa digitar “Cidade Nova” no momento do cadastro de cliente. Isso ira prejudicar por exemplo, uma query de frequência para contar quantos clientes há no bairro em questão.

Cliente

idcliente	nome	bairro	cidade
....
8	Piter	Cidade Nova	Rio de Janeiro
9	Walesca	C. Nova	Rio de Janeiro
...

5 Um Exemplo Passo a Passo

- Considere uma loja que vende madeiras e outros materiais para fabricantes de móveis. Cada venda da loja é atualmente registrada em uma nota de fornecimento em papel.
- Suponha que a loja queira criar um BD para manter todas as informações referentes às notas de fornecimento de mercadorias.

Nota de Fornecimento de Mercadoria					
Data: 09/10/2017			Nº da Nota: 1019		
Dados do Cliente					
Id. Cliente	Nome	Telefone	Endereço		
31	Retrô Móveis	21-2142-0000	Av. Chile, 500		
Item	Cod. Produto	Produto	Quantidade	Preço (R\$)	Total
1	P87	Chapa de MDF	10	14,90	149,00
2	P23	Rodapé em Poliestierno	17	12,90	219,30
3	P44	Estrado	5	8,50	42,50

5 Um Exemplo Passo a Passo

- Exemplo 1 de nota de mercadoria

Nota de Fornecimento de Mercadoria					
Data: 09/10/2017			Nº da Nota: 1019		
Dados do Cliente					
Id. Cliente	Nome	Telefone	Endereço		
31	Retrô Móveis	21-2142-0000	Av. Chile, 500		
Item	Cod. Produto	Produto	Quantidade	Preço (R\$)	Total
1	P87	Chapa de MDF	10	14,90	149,00
2	P23	Rodapé em Poliestierno	17	12,90	219,30
3	P44	Estrado	5	8,50	42,50

5 Um Exemplo Passo a Passo

- Exemplo 2 de nota de mercadoria

Nota de Fornecimento de Mercadoria					
Data: 11/10/2017			Nº da Nota: 1035		
Dados do Cliente					
Id. Cliente	Nome	Telefone	Endereço		
48	Móveis Alfa	21-3333-0000	Rua Frei Caneca, 1350		
Item	Cod. Produto	Produto	Quantidade	Preço (R\$)	Total
1	P20	Chapa de MDP	20	11,90	238,00
2	P87	Chapa de MDF	20	13,90	278,00

5 Um Exemplo Passo a Passo

- **Passo 1:**
 - Resolver o problema partindo da identificação de todas as entidades e relacionamentos.
 - Também é possível identificar os atributos de cada entidade (em problemas mais complexos isso pode ser mais difícil em um primeiro momento – nestes casos você irá ao menos tentar identificar os principais atributos das entidades).

5 Um Exemplo Passo a Passo

- **Passo 1:**
 - A partir dos exemplos 1 e 2 de notas de mercadorias, é possível compreender que:
 - Uma nota é relativa a um único cliente. Cada nota possui um número único e uma data de emissão.
 - Uma nota contém diversos itens. Cada item representa um produto, comprado em uma determinada quantidade, por um determinado preço unitário.

Nota de Fornecimento de Mercadoria					
Data: 09/10/2017			Nº da Nota: 1019		
Dados do Cliente					
Id. Cliente	Nome	Telefone	Endereço		
31	Retrô Móveis	21-2142-0000	Av. Chile, 500		
Item	Cod. Produto	Produto	Quantidade	Preço (R\$)	Total
1	P87	Chapa de MDF	10	14,90	149,00
2	P23	Rodapé em Poliestierno	17	12,90	219,30
3	P44	Estrado	5	8,50	42,50

5 Um Exemplo Passo a Passo

- **Passo 1:**
 - A partir dos exemplos 1 e 2 de notas de mercadorias é possível compreender que:
 - Um cliente pode estar associado a diversas notas (compras realizadas em dias diferentes). Cada cliente possui um código (id) único, nome, telefone e endereço.
 - Cada produto tem um código e nome.

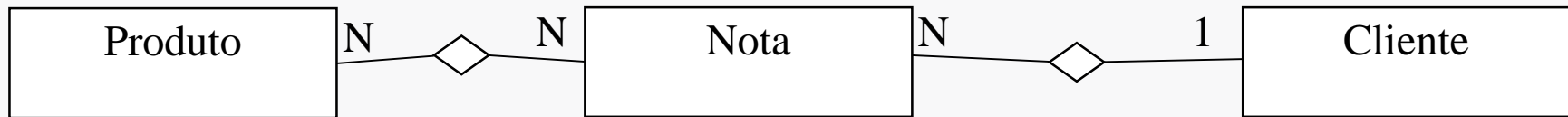
Nota de Fornecimento de Mercadoria					
Data: 11/10/2017			Nº da Nota: 1035		
Dados do Cliente					
Id. Cliente	Nome	Telefone	Endereço		
48	Móveis Alfa	21-3333-0000	Rua Frei Caneca, 1350		
Item	Cod. Produto	Produto	Quantidade	Preço (R\$)	Total
1	P20	Chapa de MDP	20	11,90	238,00
2	P87	Chapa de MDF	20	13,90	278,00

5 Um Exemplo Passo a Passo

- **Passo 2:**
 - De acordo com as anotações feitas no passo 1, é possível elaborar o modelo conceitual para o BD, na forma de um DER.
 - As entidades são os elementos que possuem dados, informações ou características que se deseja guardar. Neste caso, os mais evidentes são: *Nota*, *Produto* e *Cliente*.

5 Um Exemplo Passo a Passo

- **Passo 2:**
 - Este modelo é mostrado a seguir. Veja que o modelo possui três entidades e dois relacionamentos. Um dos relacionamentos é do tipo 1 x N (*Nota* e *Cliente*) e o outro do tipo N x N (*Nota* e *Produto*).



5 Um Exemplo Passo a Passo

- **Passo 2:**
- Neste momento, é importante registrar, mesmo que em um papel ou documento a parte, os atributos associados a cada entidade e relacionamento. Isto vai ajudar a não ficar perdido no momento de realizar o projeto lógico.

Rascunho

Produto (codproduto: alfanumerico(2), nomProduto alfanumerico(50))

Itemnota (numnota: inteiro, codproduto: alfanumerico(2), quantidade: inteiro, precounit: real) codproduto é FK

Nota (numnota: inteiro, datanota: data, idcliente: inteiro) idcliente é FK

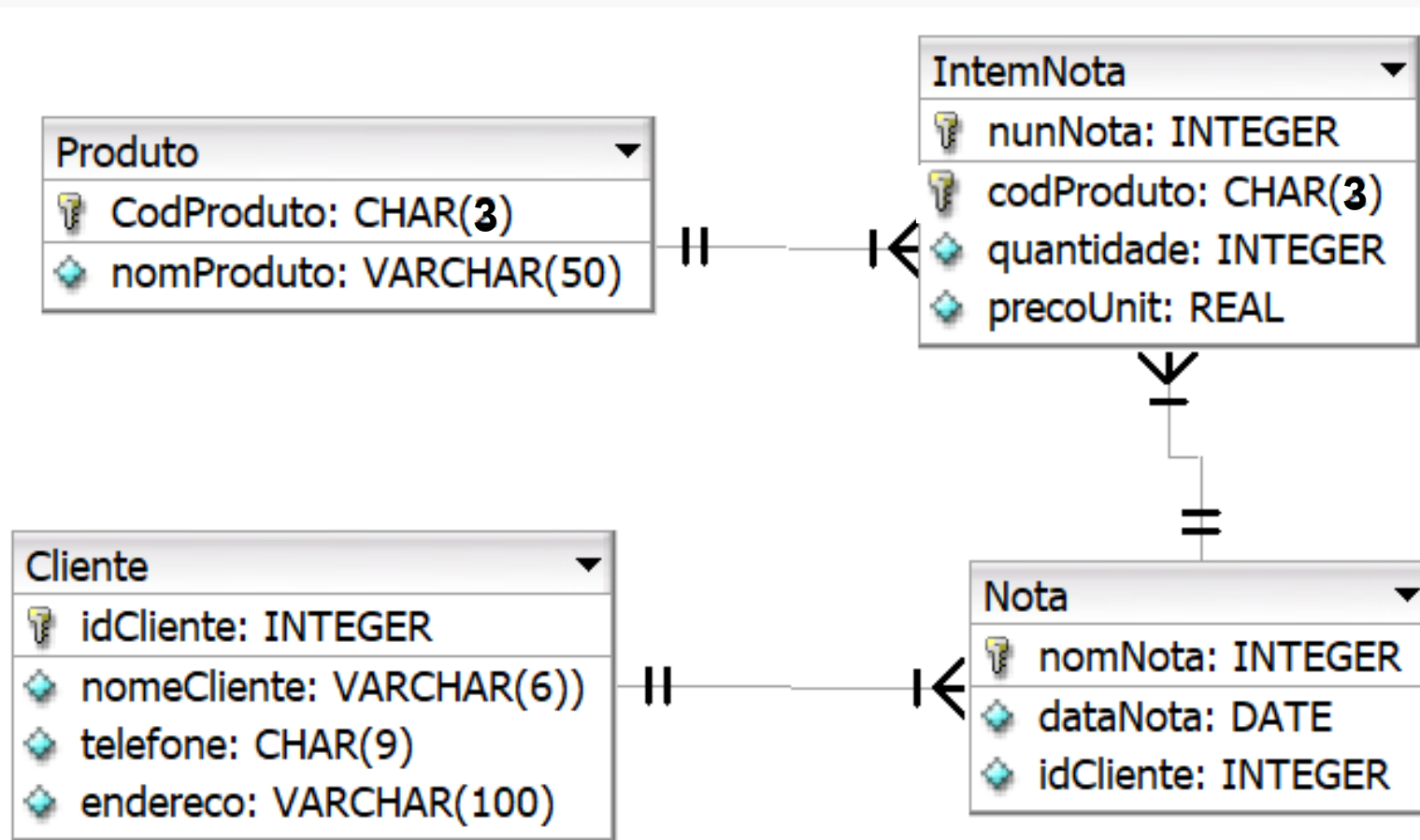
Cliente (idcliente: inteiro, nomecliente: alfanumerico(60), telefone: alfanumerico (9), endereço: alfanumerico (100))

5 Um Exemplo Passo a Passo

- **Passo 3:**
 - Elaborar o projeto lógico. Para tal é preciso:..
 - Aplicar as regras de mapeamento ER-Relacional sobre o modelo conceitual elaborado no Passo 2;
 - Determinar os atributos, tipos e chaves de cada tabela, considerando o levantamento feito no Passo 1.

5 Um Exemplo Passo a Passo

- **Passo 3:**
 - Projeto conceitual e lógico de BD Relacional da loja de madeiras feito no DBdesigner.



5 Um Exemplo Passo a Passo

Passo 3: Projeto Lógico Relacional (implementa-se um formato de tabela com linhas e colunas)

Define-se um conjunto mínimo de atributos para atender os requisitos propostos.

NN = NÃO NULO

PK = chave primária

FK = chave estrangeira

Produto

codProduto: char(3) PK FK
nomProduto: varchar(50) NN

Nota

numNota: integer PK
codProduto: char(3) FK NN
idCliente: integer

ItemNota

numNota: INTEGER PK NN
codProduto: CHAR(3) PK FK NN
Quantidade: integer NN
precoUnit: real NN

Cliente

idCliente: INTEGER PK NN
nomeCliente: varchar(60) NN
Telefone: char(9) NN
endereco: varchar(100) NN

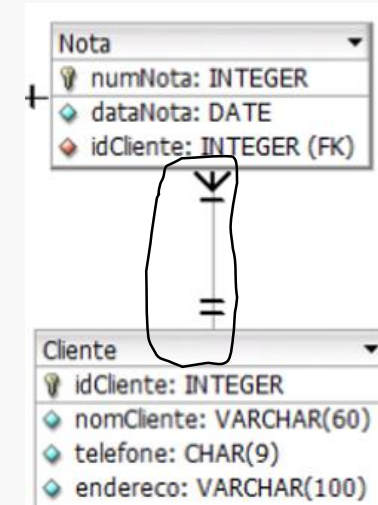
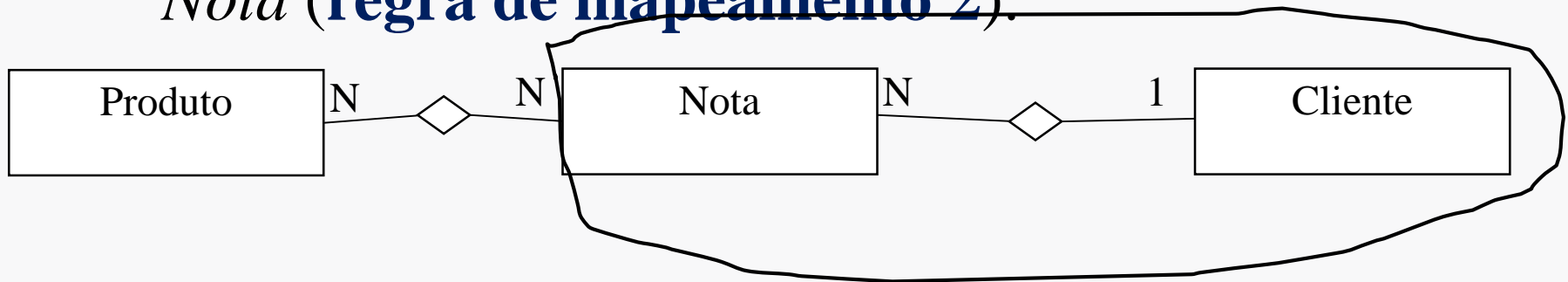
5 Um Exemplo Passo a Passo

- **Raciocínio utilizado para passar do conceitual para o Lógico:**
- As três entidades do modelo conceitual (*Nota*, *Produto* e *Cliente*) se transformaram em três tabelas no modelo lógico (aplicação da regra de mapeamento 1).



5 Um Exemplo Passo a Passo

- Raciocínio utilizado para passar do conceitual para o Lógico:
- O campo “idCliente”, que é PK de *Cliente*, foi incluído como FK na tabela *Nota*, para possibilitar a implementação do relacionamento 1 x N entre *Cliente* e *Nota* (**regra de mapeamento 2**).



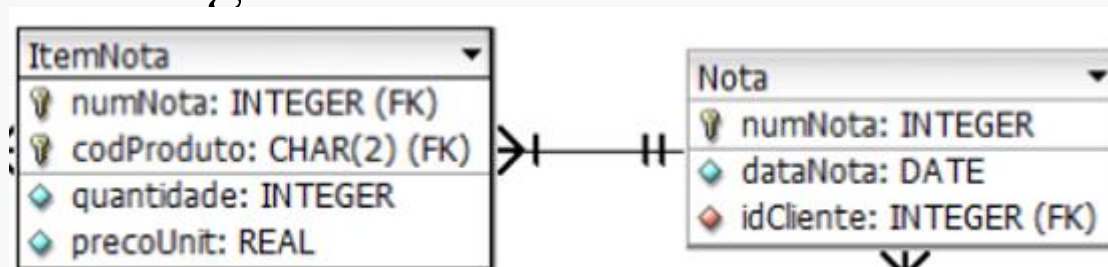
5 Um Exemplo Passo a Passo

- **Raciocínio utilizado para passar do conceitual para o Lógico:**
 - Não é necessário armazenar o **preço total no BD**, pois este é um atributo derivado do preço unitário do produto e da quantidade comprada (**precoUnit x quantidade**). Desta forma, seria redundante armazenar o preço total (assim diminui a chance de ocorrência de anomalias).

Nota de Fornecimento de Mercadoria					
Data: 11/10/2017			Nº da Nota: 1035		
Dados do Cliente					
Id. Cliente	Nome	Telefone	Endereço		
48	Móveis Alfa	21-3333-0000	Rua Frei Caneca, 1350		
Item	Cod. Produto	Produto	Quantidade	Preço (R\$)	Total
1	P20	Chapa de MDP	20	11,90	238,00
2	P87	Chapa de MDF	20	13,90	278,00

5 Um Exemplo Passo a Passo

- **Raciocínio utilizado para passar do conceitual para o Lógico:**
 - Com relação ao preço, em muitos sistemas reais é necessário criar uma tabela auxiliar de histórico de preços para armazenar todo o histórico de preços de todos os produtos.
 - Em nosso projeto utilizamos uma estratégia mais simples (*que nem sempre pode ser utilizada em projetos reais*): como temos o preço unitário na tabela *ItemNota*, pode-se identificar o preço do produto na data em que aquela nota foi gerada.



5 Um Exemplo Passo a Passo

- **Raciocínio utilizado para passar do conceitual para o Lógico:**
- Também desconsideramos a informação da coluna “Item”, pois ela refere-se apenas à posição do item na nota de papel (se o item está na 1ª linha, 2ª linha, etc.).

Nota de Fornecimento de Mercadoria					
Data: 11/10/2017			Nº da Nota: 1035		
Dados do Cliente					
Id. Cliente	Nome	Telefone	Endereço		
48	Móveis Alfa	21-3333-0000	Rua Frei Caneca, 1350		
Item	Cod. Produto	Produto	Quantidade	Preço (R\$)	Total
1	P20	Chapa de MDP	20	11,90	238,00
2	P87	Chapa de MDF	20	13,90	278,00

5 Um Exemplo Passo a Passo

- **Raciocínio utilizado para passar do conceitual para o Lógico:**
- Com o projeto lógico construído, é interessante verificar se há anomalias que possam causar problemas ao sistema. Assim, pode-se usar a 3FN.
- 1FN - Remoção de atributos multivalorados: Todas as tabelas possuem atributos atômicos?
 - Duas dúvidas podem surgir: uma para o atributo telefone do cliente, mas para nosso projeto a intenção é manter somente um telefone de contato.
 - Outra dúvida é para o atributo endereço, pois ele é um atributo composto que pode ser desmembrado para logradouro, bairro, CEP, município, estado.

5 Um Exemplo Passo a Passo

- **Raciocínio utilizado para passar do conceitual para o Lógico:**
 - 1FN - Remoção de atributos multivalorados: Todas as tabelas possuem atributos atômicos?
 - Terceira dúvida, um cliente pode ter mais de um endereço? Baseado na ficha não há uma informação que confirme isso. É possível que o endereço de entrega seja diferente da entrega anterior. Neste caso, uma possibilidade é mover endereço para ItemNota ou criar uma tabela de endereços de entrega do cliente associado a ItemNota para evitar redundâncias. Uma terceira saída é criar redundância, adicionando endereço de entrega na nota. Vamos considerar que o endereço seja um dado que pouco muda e não seja preciso fazer mudanças.

5 Um Exemplo Passo a Passo

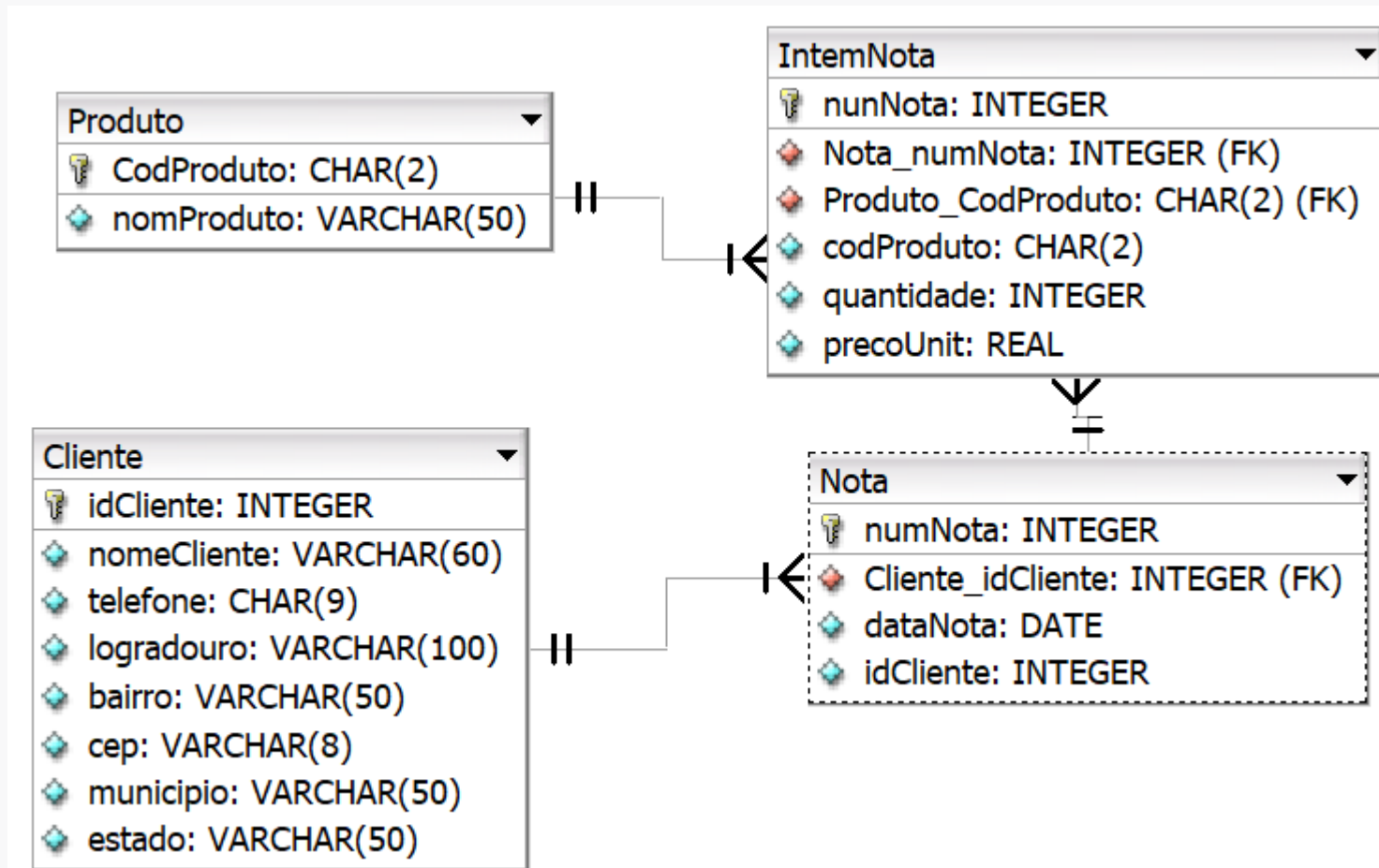
- **Raciocínio utilizado para passar do conceitual para o Lógico:**
- Com o projeto lógico construído, é interessante verificar se há anomalias que possam causar problemas ao sistema. Assim, pode-se usar a 3FN.
- 2FN - Remoção de atributos parcialmente dependentes:
A única tabela com chave composta, *Itemnota* (numnota, cod_produto, quantidade, precounit), seus atributos não-chaves, “quantidade e precounit”, não podem ser determinados por somente um dos dois atributos da chave composta.

5 Um Exemplo Passo a Passo

- **Raciocínio utilizado para passar do conceitual para o Lógico:**
- Com o projeto lógico construído, é interessante verificar se há anomalias que possam causar problemas ao sistema. Assim, pode-se usar a 3FN.
- 3FN - Remoção de atributos transitivamente dependentes: Cada tabela tem atributos que são funcionalmente dependentes da PK e nenhum outro atributo não-chave define um outro atributo não-chave, o que elimina a possibilidade de existir transitividade. Por exemplo em *Cliente*(idcliente, nome, telefone, endereco) nome até poderia em muitos casos definir endereço, o que sugeriria a divisão da tabela *Cliente* em mais partes, só que podem haver homônimos e mesmo assim, nome é um atributo complexo que pode ser digitado com uma grafia diferente, assim idcliente é uma chave substituta.

5 Um Exemplo Passo a Passo

- **Modelo atualizado** após aplicação desmembrando atributo o composto endereço.



6 Exercícios

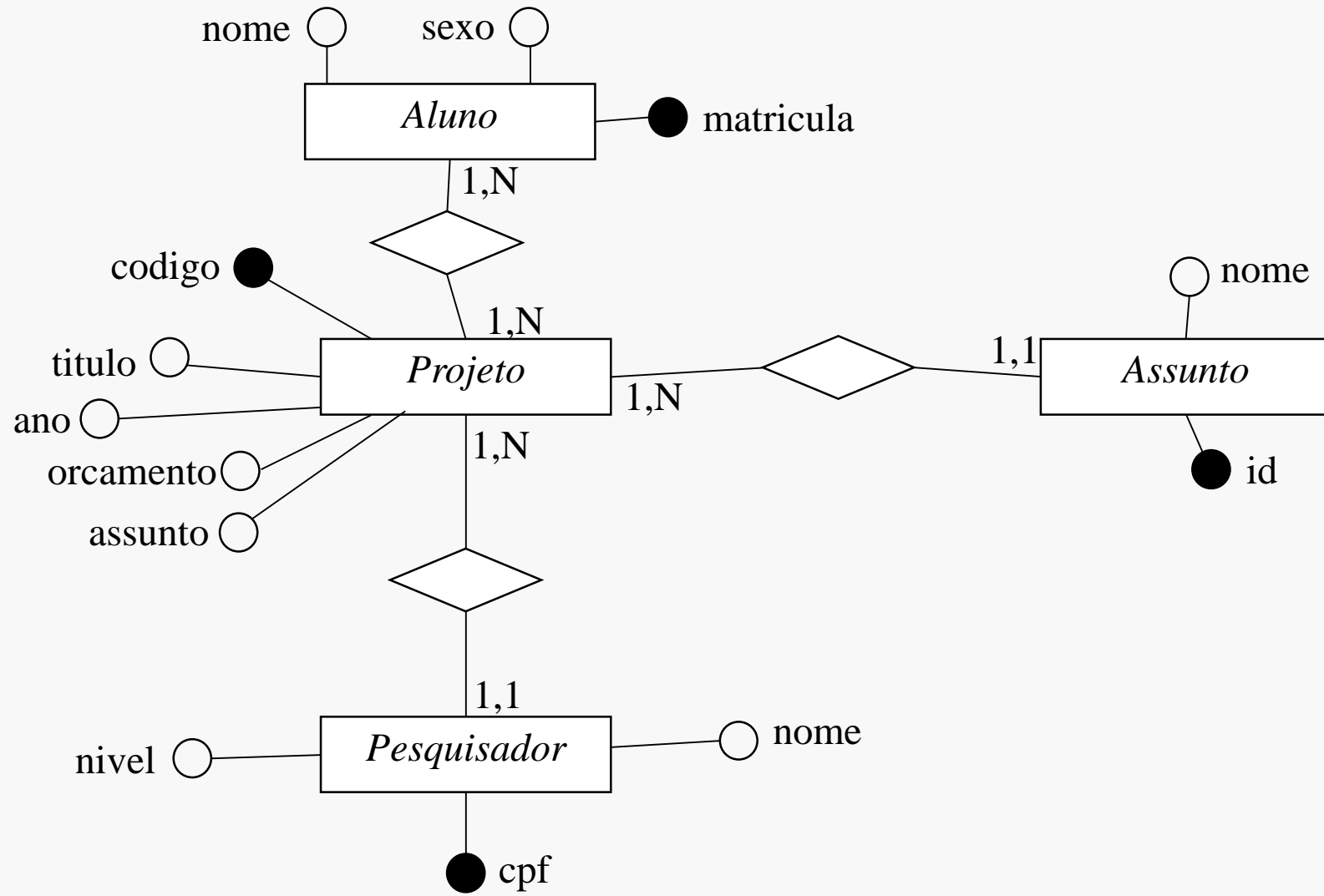
Exercício 2. Um sistema que gerencia os projetos de iniciação científica de uma universidade possui as seguintes características:

- Cada projeto é identificado unicamente por um código, possui um título, ano, um orçamento previsto em R\$ e está associado a um único assunto (ex: Estatística, Física, Banco de Dados, etc.)
- Um assunto possui um nome e nenhum outro atributo.
- Um projeto possui um único pesquisador responsável e diversos alunos participantes.
- Pesquisadores possuem CPF, nome e nível (J=Júnior, P=pleno ou S=senior). Atenção professores estrangeiros não possuem CPF, neste caso deve CPF receber o valor 000000000000.
- Alunos possuem uma matrícula, nome, sexo e idade e podem estar alocados em mais de um projeto.

A partir deste minimundo descrito, **faça o projeto lógico de um BD** para ser utilizado por este sistema.

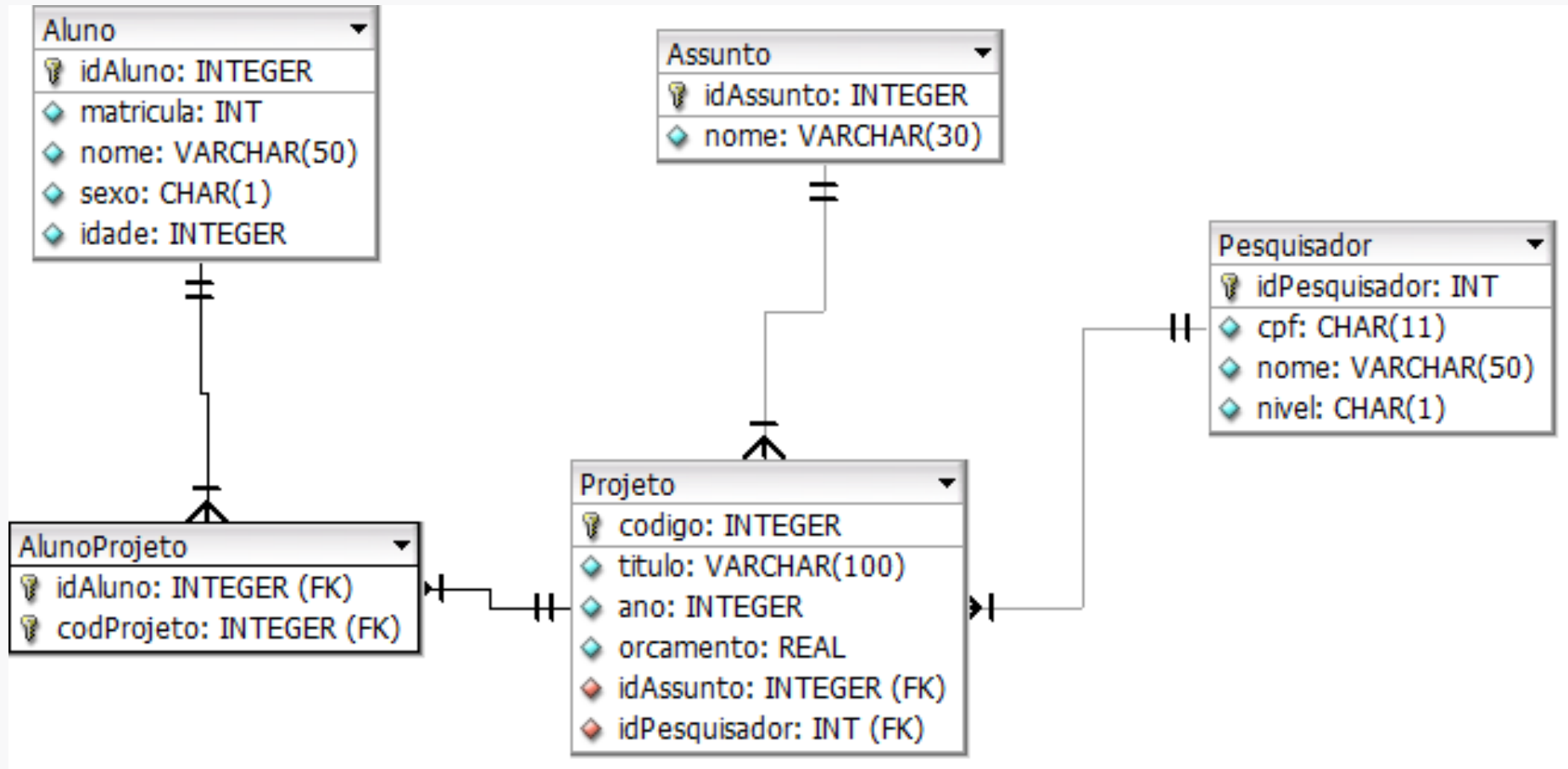
6 Exercícios

Projeto conceitual:



6 Exercícios

Projeto lógico de BD relacional:



6 Exercícios

- Conversão de conceitual para lógico:
- **Primeiro passo:** cada entidade é traduzida para uma tabela. Neste processo, cada atributo da entidade define uma coluna desta tabela. Os atributos identificadores da entidade definem as colunas que compõem a chave primária da tabela
 - Sobre os nomes das colunas, não é aconselhável simplesmente transcrever os nomes de atributos para nomes de colunas. Nomes de colunas serão referenciados frequentemente em programas e outras formas de texto em computador. Assim, para diminuir o trabalho dos programadores é conveniente manter os nomes de colunas curtos. Além disso, em um SGBD relacional, o nome de uma coluna não deve conter espaço em branco e caracteres especiais (acentos, arroba, etc). Assim, nomes de atributos compostos de diversas palavras devem ser abreviados.

6 Exercícios

- Conversão de conceitual para lógico:
- **Segundo passo:** Implementação de relacionamentos e respectivos atributos. Os relacionamentos são implementados seguindo a regra de que a menor cardinalidade deve conter a FK.
- Relacionamentos N X N são transformados em tabela, neste caso *AlunoProjeto* contem a PK de Aluno e a PK de Projeto para formar sua PK.
- **Terceiro passo:** Aplica-se a 3FN. Veja que CPF em *Pesquisador* é não-chave e define as colunas nome e nível, contudo para manter preservado o documento de identificação usa-se uma chave substituta. Outro motivo é que podem haver pesquisadores estrangeiros sem CPF.

6 Exercícios

Projeto lógico relacional:

Aluno

Idaluno : integer PK
matricula: int NN
nome : varchar(50) NN
Sexo: char(1)
Idade: integer

AlunoProjeto

Idaluno : integer PK FK (Aluno)
codprojeto : integer PK FK (Projeto)

Assunto

Idassunto : integer PK
nome : varchar(30) NN

Projeto

codigo : integer PK FK
titulo : varchar(100) NN
Ano: integer
orcamento: real
Idassunto: integer FK NN (Assunto)
Idpesquisador: integer FK NN (Pesquisador)

Pesquisador

idpesquisador : integer PK
cpf : char(11) NN
Ano: integer
nome: varchar(50) NN
Nivel : char(1)

Obs.: Note que os atributos FK e PK possuem mesmo nome para facilitar a identificação da referência e possui o nome da tabela que contém a PK. **Para efeito de prova, usem esta representação para facilitar.**

Obrigado