

Centro de Ciências Exatas e da Tecnologia
Cursos de Informática

Modelagem de Banco de Dados (Tradução para o modelo lógico)

Daniel L. Notari

Agosto - 2022

Projeto de um Banco de Dados

Modelo conceitual

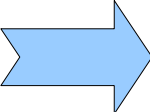
==

modelo de dados abstrato, que descreve a estrutura de um banco de dados de forma independente de um SGBD particular

Ex: modelo ER, modelo UML

Modelo lógico

==



modelo de dados que representa a estrutura de dados de um banco de dados conforme vista pelo usuário do SGBD

Ex: modelo relacional, modelo a objetos, modelo hierárquico

Modelo físico

==

contém detalhes sobre a representação interna das informações:

Ex: estruturas de índices, estruturas de arquivos, níveis de isolamento (otimização de performance)

Modelo lógico

- Independente do SGBD
- Descrição do BD no nível dos usuários do BD (programadores, usuários que tem acesso direto aos dados do banco)
- Não apresenta detalhes do armazenamento interno das informações (estruturas de arquivo índices de acesso)

Modelo lógico

- Tabela
 - linhas (tuplas), colunas (atributos)
- Chaves
 - Primária, estrangeira
- Domínios
 - Faixa de valores que um atributo pode conter
- Valores Nulos
- Restrições de integridade:
 - Integridade de domínio
 - Integridade de vazio
 - Integridade de chave (chave primária única)
 - Integridade referencial (chave estrangeira)

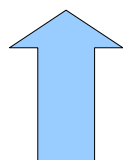
Modelo lógico

Contexto	Terminologia		
Modelo Relacional (Formal)	Relação	Tupla	Atributo
Modelo Relacional (Informal)	Tabela	Linha	Coluna
Teoria dos Conjuntos	Conjunto	Lista	Nodo
Sistema de Arquivos	Arquivo	Registro	Campo
Orientação a Objetos	Classe	Objeto	Atributo

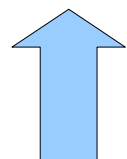
Modelo lógico

Notação resumida:

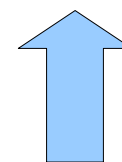
Aluno (CodAluno, Nome, #CodCurso)
Curso (CodCurso, Nome)



**Nome da
Tabela**



**Chave
Primária**



**Chave
Estrangeira**

Aluno

CodAluno	Nome	CodCurso
2034	Joaquim	124
3028	Ana	124
1212	Paula	138

Curso

CodCurso	Nome
124	Ciência da Computação
138	Sistemas de Informação

Modelo físico

- Descrição detalhada de como a base de dados está armazenada internamente
- Linguagens e notações para o modelo físico variam de produto a produto (SGBD)
- Produtos escondem o modelo físico

Modelo ER (Entidade-Relacionamento)

Parte 1 (Entidade, Atributo, chave, auto-relacionamento, entidade fraca)

Transformação DER para modelo lógico

I. Tradução inicial de entidades e respectivos atributos;

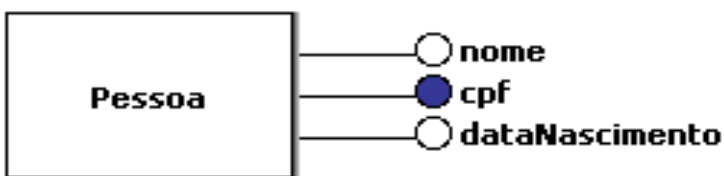
II. Tradução de relacionamentos (auto**, binário, ternário, **entidade fraca**, agregação) e respectivos atributos;**

III. Tradução de generalizações/especializações e respectivos atributos;

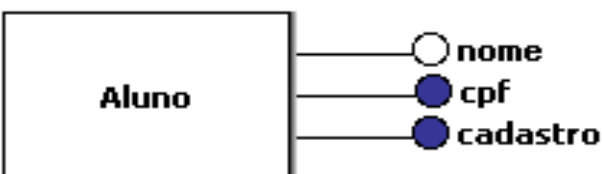
Tradução de entidades e atributos

- Cada entidade é traduzida para uma tabela
- Cada atributo define uma coluna desta tabela
 - Nomes de colunas devem ser curtos
- Atributos identificadores compõem a chave primária da tabela

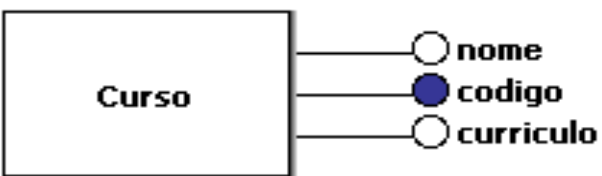
Tradução de entidades e atributos



Pessoa (cpf, nome, dataNascimento)



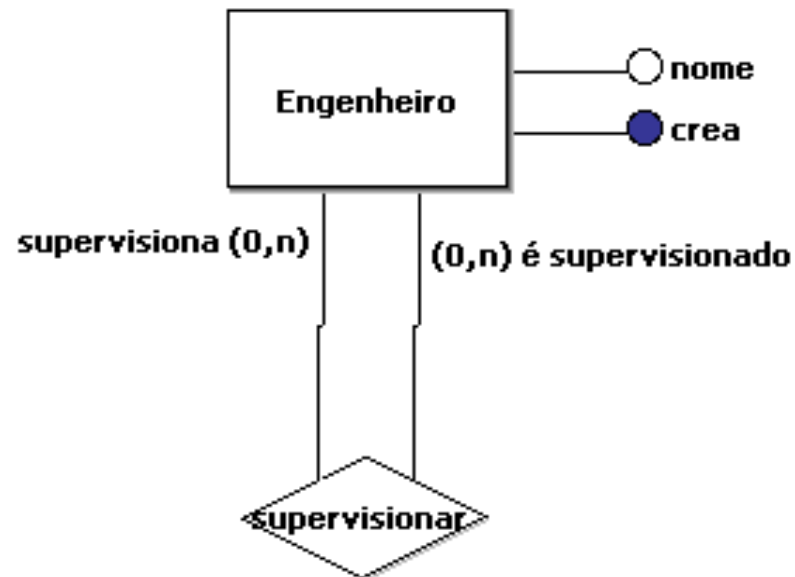
Aluno (cpf, cadastro, nome)



curso (Codigo, nome, curriculo)

Tradução Auto-relacionamento

- Criar uma nova tabela com a adição da coluna identificador duas vezes, uma para cada papel exercido no relacionamento. A chave primária será composta pelas colunas com cardinalidade n.



Engenheiro (crea, nome)

Supervisionar(#creaSupervisiona, #creaEsupervisionado)

Tradução Auto-relacionamento

Engenheiro (crea, nome)

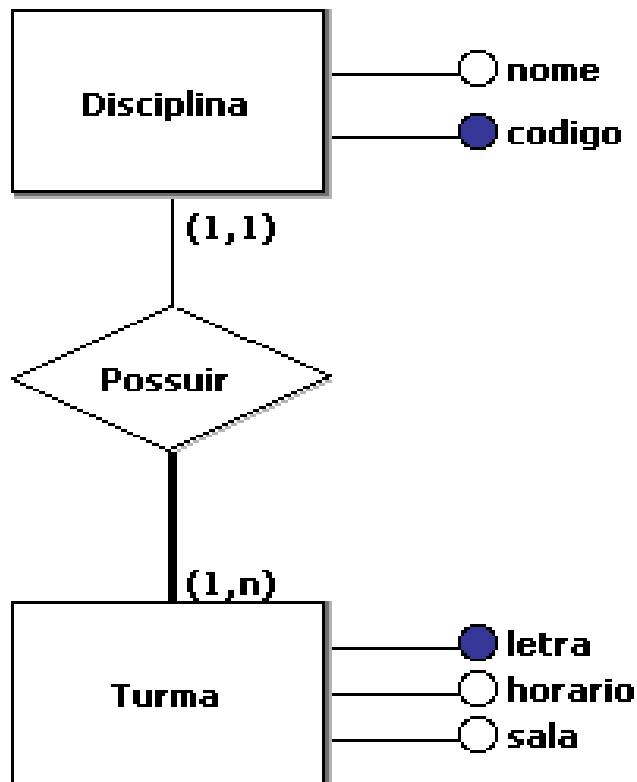
Supervisionar(#creaSupervisiona, #creaEsupervisionado)

```
create table engenheiro(  
    crea int not null primary key,  
    nome varchar(40) not null);
```

```
create table supervisionar(  
    crea_supervisiona    int not null,  
    crea_e_supervisionado int not null,  
    primary key (crea_supervisiona, crea_e_supervisionado ),  
    foreign key (crea_supervisiona) references engenheiro(crea),  
    foreign key (crea_e_supervisionado) references engenheiro(crea));
```

Tradução relacionamento entidade fraca

- Entidades fracas devem ter, como parte de sua chave primária, a chave primária da entidade forte.



Disciplina (codigo, nome)

Turma(#coddisc, letra, horario, sala)

Tradução relacionamento entidade fraca

Disciplina (codigo, nome)

Turma(#coddisc, letra, horario, sala)

```
create table disciplina(  
    codigo int not null primary key,  
    nome varchar(40) not null);
```

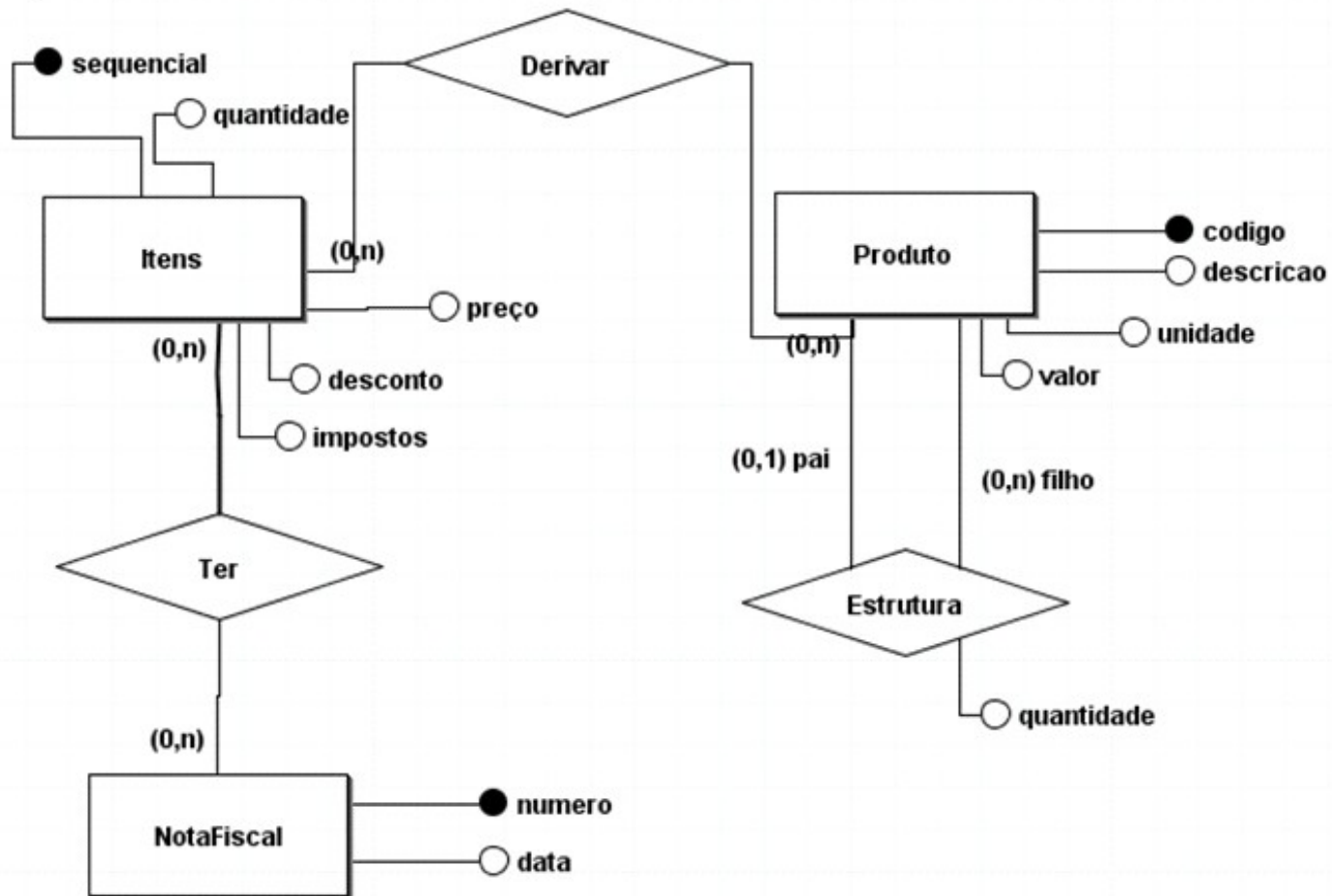
```
create table turma(  
    coddisc int not null,  
    letra char not null,  
    horario char(5) not null,  
    sala int not null,  
    primary key (coddisc, letra),  
    foreign key (coddisc) references disciplina(codigo));
```

Exercício de tradução

- Para o modelo de itens de uma nota fiscal usando entidade fraca, a partir do DER, crie o modelo lógico e o físico.
- Para o modelo de componentes de uma estrutura de produto usando auto-relacionamento, a partir do DER, crie o modelo lógico e o físico.

Exercício de Modelagem

Modele os itens de uma nota fiscal usando entidade fraca. Crie o DER.
Modele os componentes de uma estrutura de produto usando auto-relacionamento. Crie o DER.

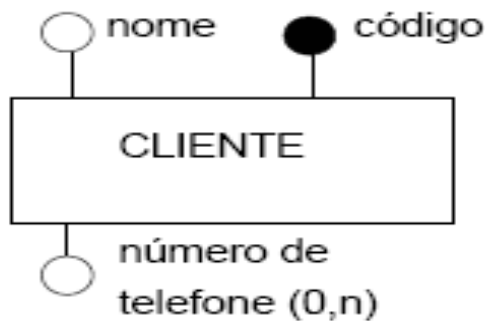


Exercício de Modelagem

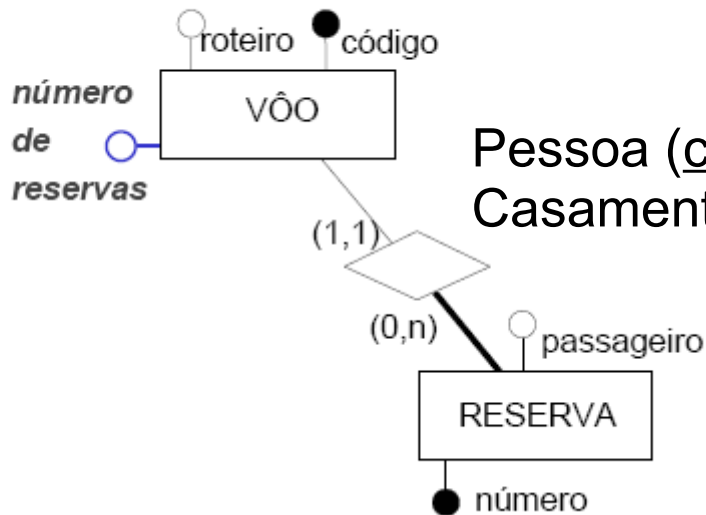
- Produto (codigo, descricao, preco, unidade)
- Item (#codprod, sequencia, quantidade, desconto, valor)
- Estruturar (#codprodpai, #codprodfilho)
- NotaFiscal (numero, data)
- NotaFiscalItens (#numeroNF, #codprod, #sequencialitem)

Exercício de tradução

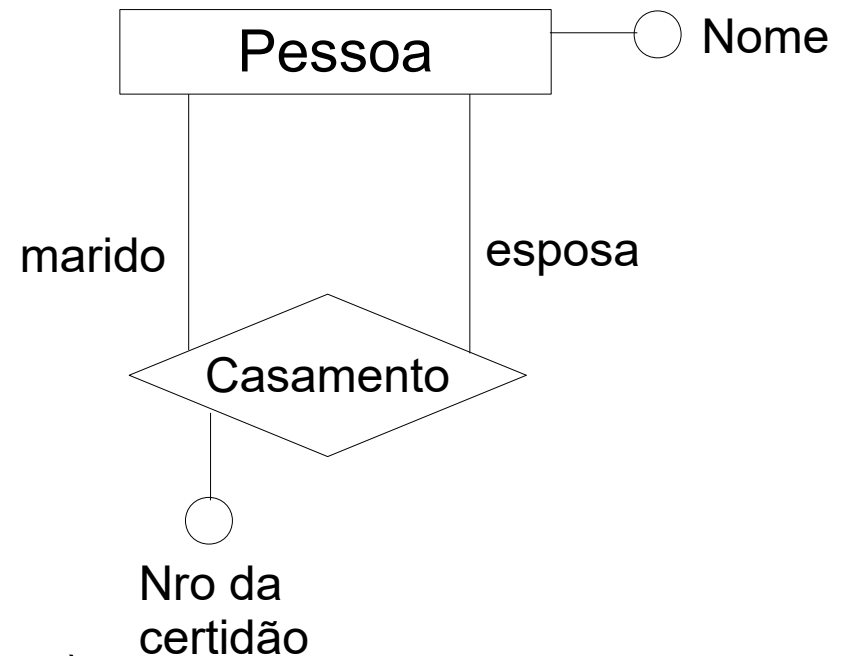
- Traduza o DER para o modelo lógico e, posteriormente para o modelo físico.



Cliente (codigo, nome),
Telefone (#codcli, numero)



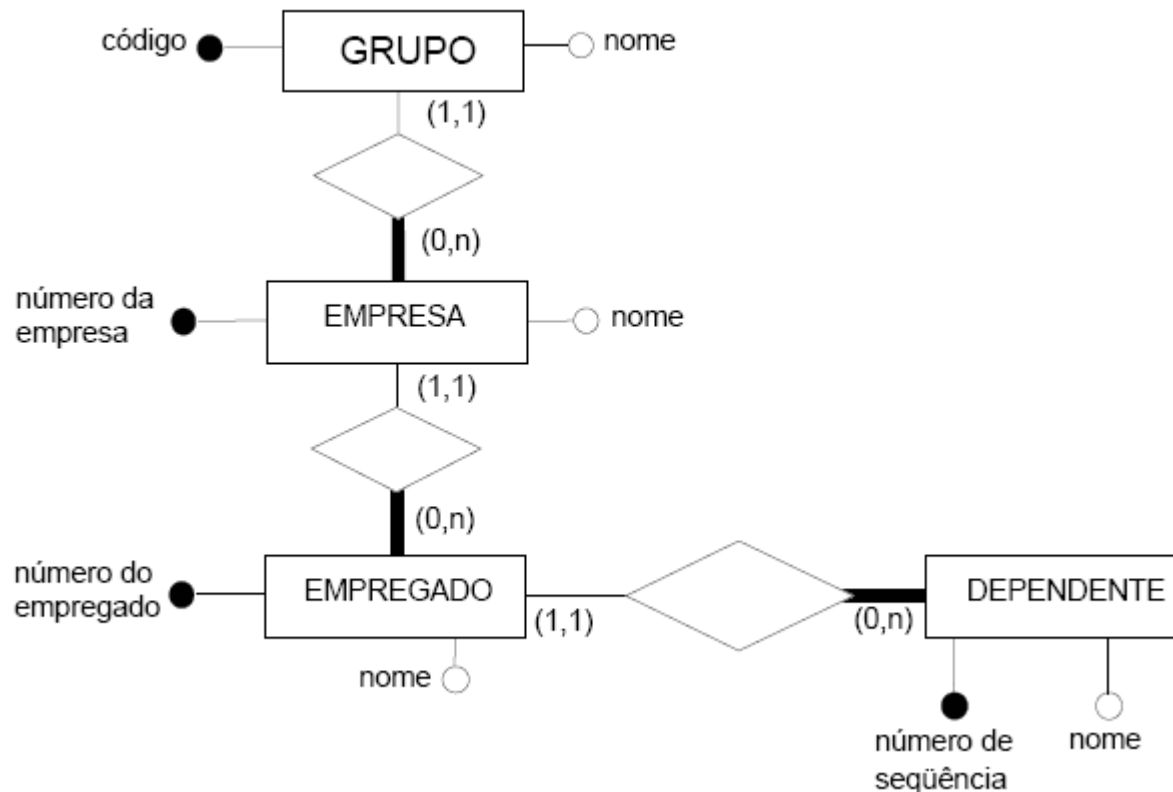
Pessoa (codigo, nome),
Casamento (#codpessoamarido, #codigopessoaesposa, nro)



atributo totalizador
Voo(codigo, nome, num_reservas)
Reserva(#codigovoo, numero, passageiro)

Exercício de tradução

- Traduza o DER para o modelo lógico e, posteriormente para o modelo físico.



Grupo (codgrupo, nome)

Empresa (#codgrupo, numempresa, nome)

Empregado (#codgrupo, #numempresa, codemp, nome)

Dependente(#codgrupo, #numempresa, #codemp, numseq, nome)

Modelo ER (Entidade-Relacionamento)

Parte 2 (cardinalidade, relacionamento binário e ternário)

Transformação DER para modelo lógico

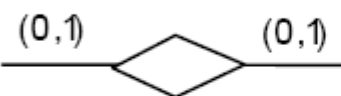





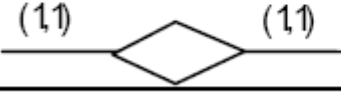



I. Tradução inicial de entidades e respectivos atributos;




II. Tradução de relacionamentos (auto, binário, ternário, entidade fraca, agregação) e respectivos atributos;

III. Tradução de generalizações/especializações e respectivos atributos;

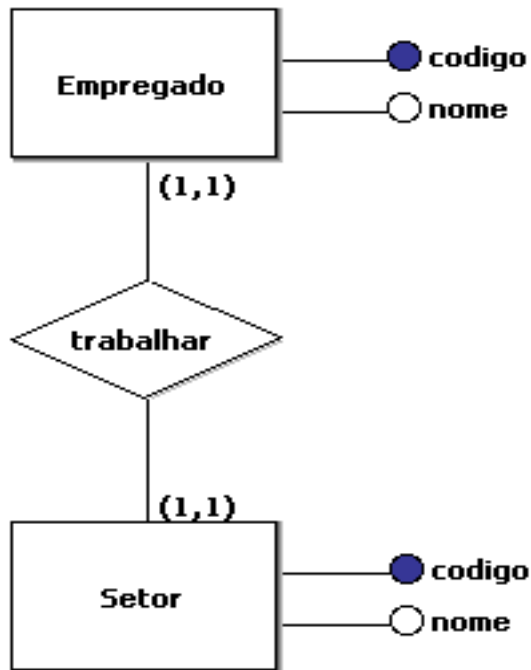
Tradução de relacionamento binário (1:1)

- Determinada pelas cardinalidades mínima e máxima dos relacionamentos:
 - Relacionamentos 1:1 – adição de colunas ou fusão de tabelas das entidades;

Tipo de relacionamento		Regra de implementação		
		Tabela própria	Adição coluna	Fusão tabelas
(0,1)		±		
(0,1)			±	
(1,1)				

 Alternativa preferida
  Pode ser usada
  Não usar

Tradução de relacionamento binário (1:1)



Empregado (codigo, nome, setor)

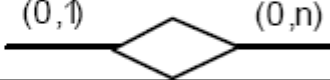





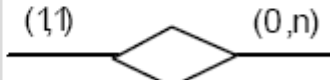



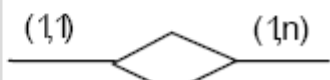



FUSÃO DE TABELAS

```
create table empregado(  
  codigo int not null primary key,  
  nome varchar(40) not null,  
  setor varchar(40) not null);
```


Tradução de relacionamento binário (1:n)

- Relacionamentos 1:n – adição de coluna na tabela originada da entidade associada à cardinalidade n;

Relacionamentos 1:n

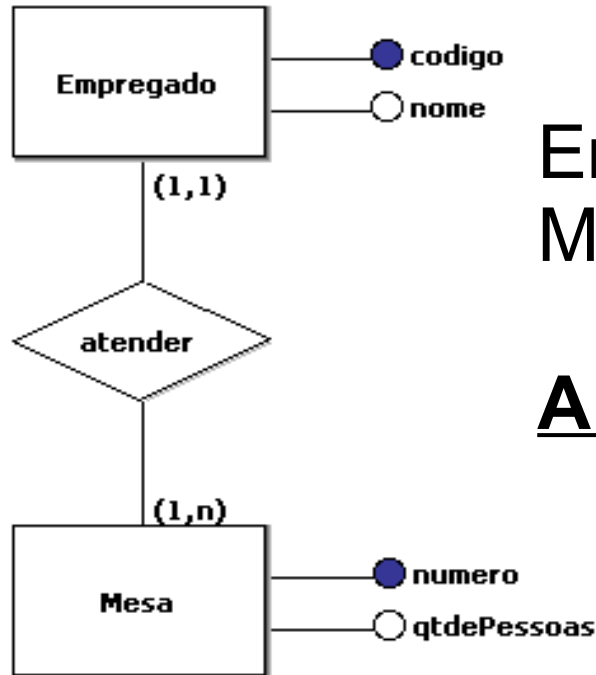
Tipo de relacionamento			Regra de implementação		
			Tabela própria	Adição coluna	Fusão tabelas
(0,1)		(0,n)	±		
(0,1)		(1n)	±		
(11)		(0,n)			
(11)		(1n)			

 Alternativa preferida

± Pode ser usada

 Não usar

Tradução de relacionamento binário (1:n)



Empregado (codigo, nome)

Mesa (numero, qtdePessoas, #codemp)

ADIÇÃO DE COLUNAS NA RELAÇÃO 1:1

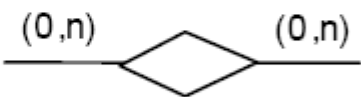



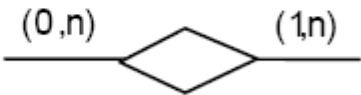



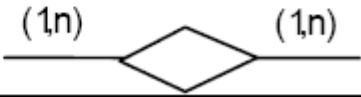



```
create table empregado(  
    codigo int not null primary key,  
    nome varchar(40) not null);
```



```
create table mesa(  
    numero      int not null,  
    qtde_pessoas int not null,  
    codemp      int not null ,  
    primary key (numero),  
    foreign key (codemp) references empregado(codigo));
```

Tradução de relacionamento binário (n:n)

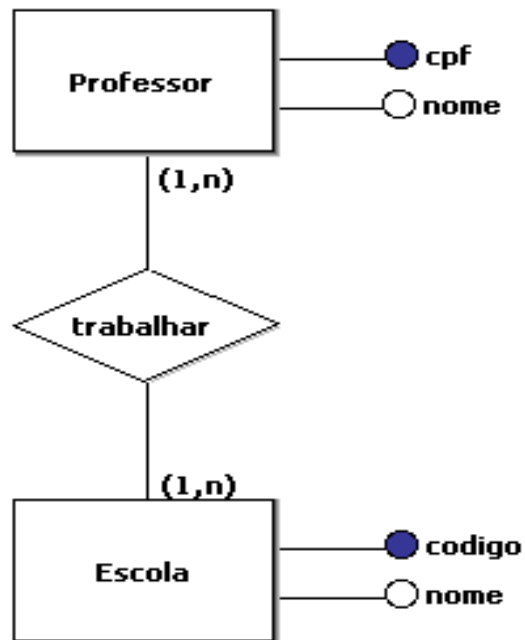
- Relacionamentos n:n – definição de tabela própria contendo as chaves primárias das entidades relacionadas.

Relacionamentos n:n

Tipo de relacionamento		Regra de implementação		
		Tabela própria	Adição coluna	Fusão tabelas
				
				
				

 Alternativa preferida  Não usar

Tradução de relacionamento binário (n:n)



Professor (cpf, nome)

Escola (codigo, nome)

Trabalhar(#cpf,#codescola)

CRIAÇÃO DE NOVA TABELA (**SEMPRE**)

```
create table professor(
    cpf int not null primary key,
    nome varchar(40) not null);
```

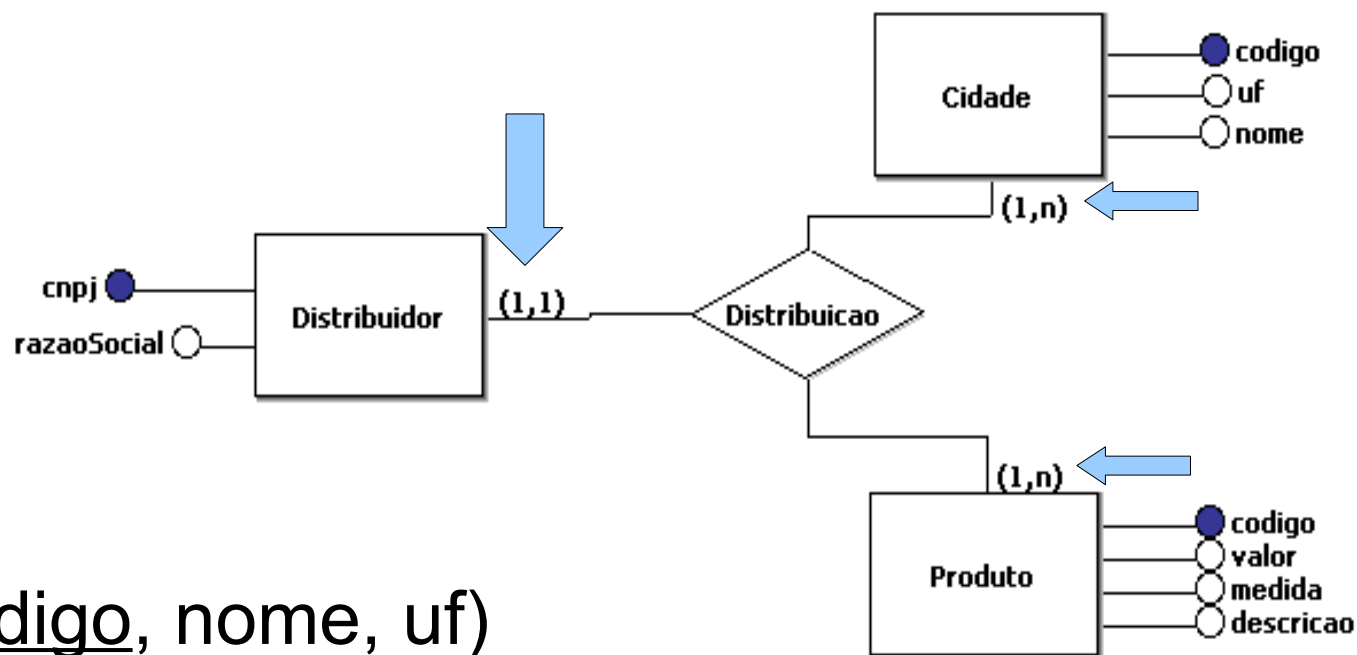
```
create table escola(
    codigo int not null primary key,
    nome varchar(40) not null);
```

```
create table trabalhar(
    cpf int not null ,
    codescola int not null,
    primary key (cpf, codescola),
    foreign key (cpf) references professor(cpf),
    foreign key (codescola) references escola(codigo));
```

Tradução de relacionamento ternário

- Criar uma nova tabela com a adição das colunas identificadoras das três tabelas que fazem parte do relacionamento.
- A chave primária será composta pelas colunas identificadoras com cardinalidade n.

Tradução de relacionamento ternário



Cidade (codigo, nome, uf)

Produto (codigo, valor, medida, descricao)

Distribuidor(cnpj, razaoSocial)

Distribuir(#codcid, #codprod, #cnpj)

**Verificar cardinalidade do relacionamento
para definir se é chave primária**

Tradução de relacionamento ternário

```
create table cidade(  
  codigo int not null primary key,  
  nome varchar(40) not null,  
  uf char(2) not null);
```

```
create table produto(  
  codigo int not null primary key,  
  descricao varchar(40) not null,  
  medida char(2) not null,  
  valor float not null);
```

```
create table distribuidor(  
  cnpj int not null primary key,  
  razao_social varchar(40) not null);
```

```
create table distribuir(  
  codcid int not null ,  
  codprod int not null,  
  cnpj int not null,  
  primary key (codcid, codprod),  
  foreign key (codcid) references cidade(codigo),  
  foreign key (codprod) references produto(codigo),  
  foreign key (cnpj) references distribuidor(cnpj));
```

Tradução de atributos no relacionamento

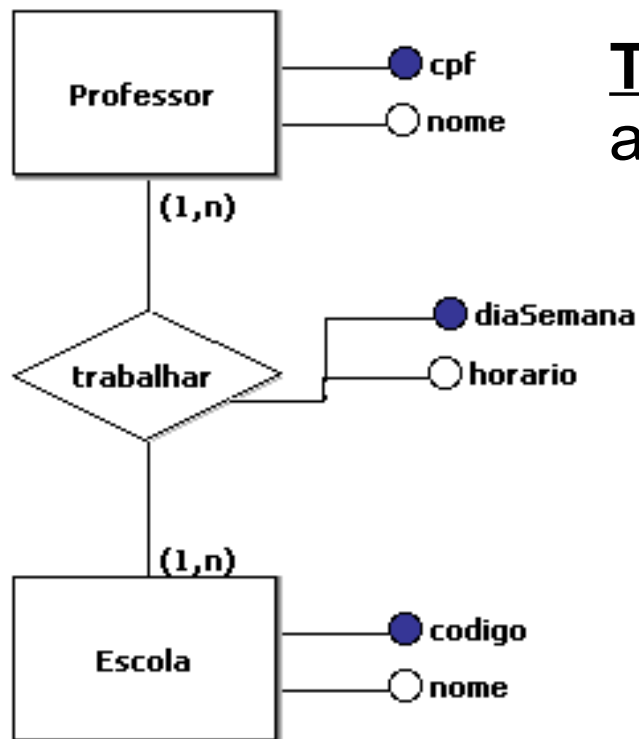


Tabela já existe!!!

alter table trabalhar

drop constraint trabalhar_pkey,

add dia_semana int not null,

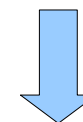
add horario char(5) not null,

add primary key (cpf, codescola, dia_semana);

Professor (cpf, nome)

Escola (codigo, nome)

Trabalhar(#cpf,#codescola, diaSemana, horario)



Tradução de atributos no relacionamento

Cidade (codigo, nome, uf)

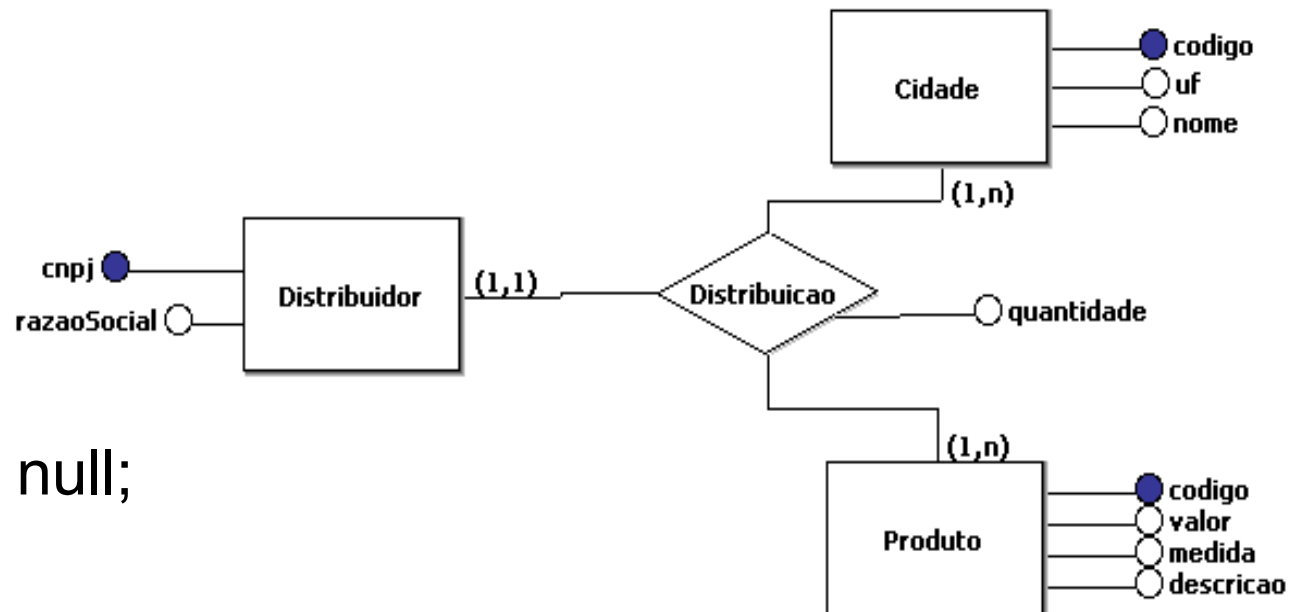
Produto (codigo, valor, medida, descricao)

Distribuidor(cnpj, razaoSocial)

Distribuir(#codcid, #codprod, #cnpj, quantidade)



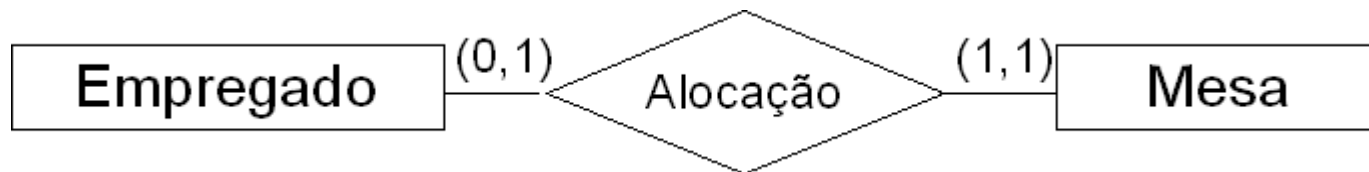
Acrescentar coluna quantidade na tabela Distribuir



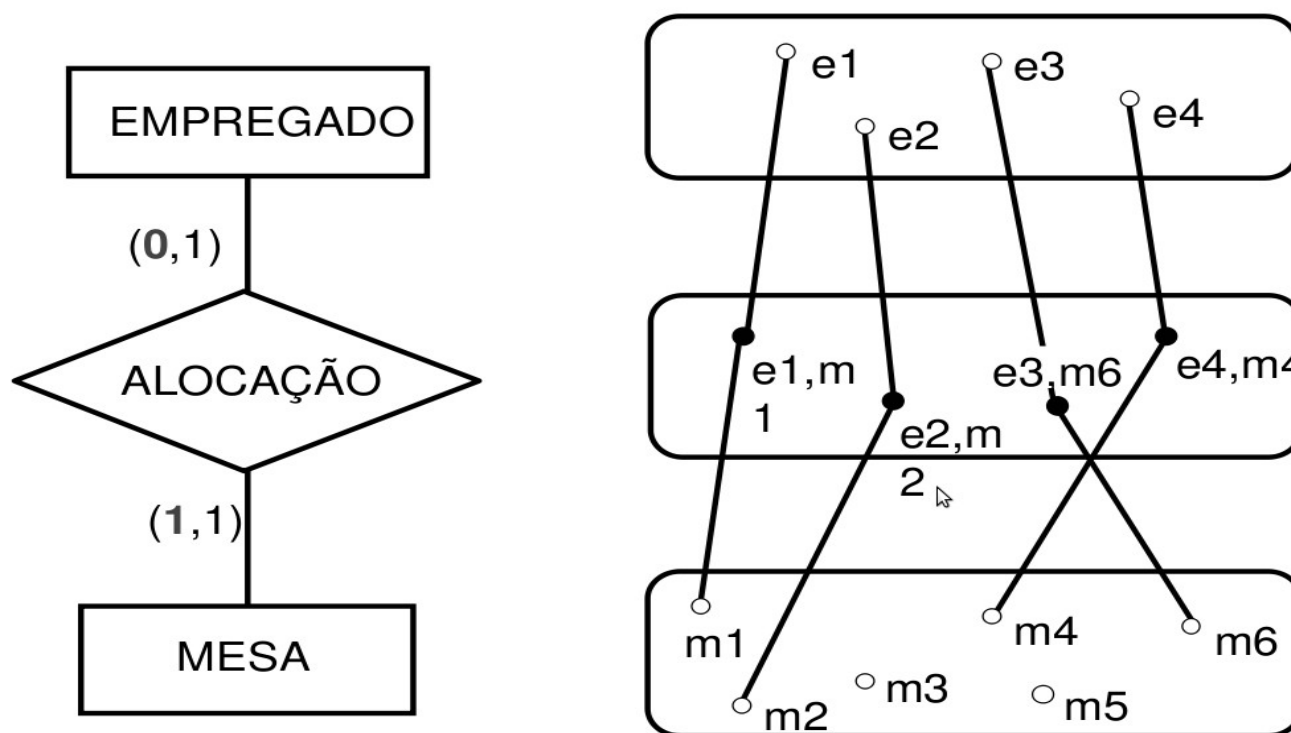
alter table distribuir
add quantidade int not null;

Exercício de tradução

Considere o DER abaixo. Para que a restrição de cardinalidade mínima seja obedecida, que ocorrências de entidade devem existir no banco de dados quando for incluída uma ocorrência de EMPREGADO? E quando for incluída uma ocorrência de MESA? Se ajudar, faça o diagrama de ocorrências.



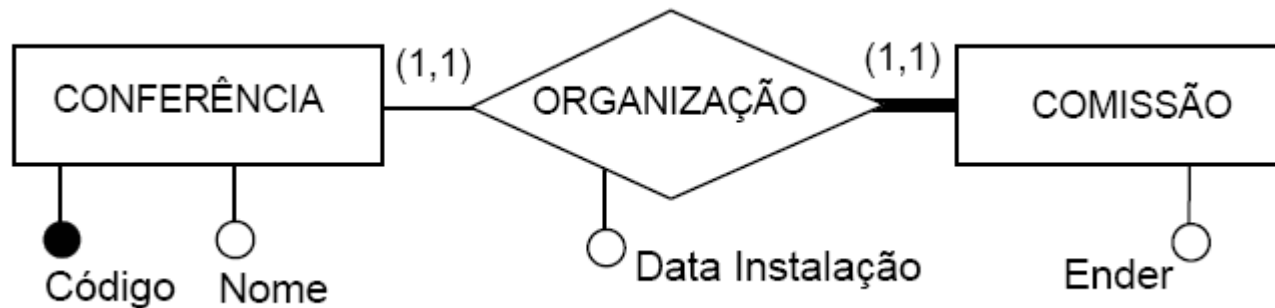
Exercício de tradução



ER apresentado está correto. Ao incluir um empregado, este tem que ser associado a um empregado. Ao incluir uma mesa, esta não precisa estar relacionada a um empregado.

Exercício de tradução

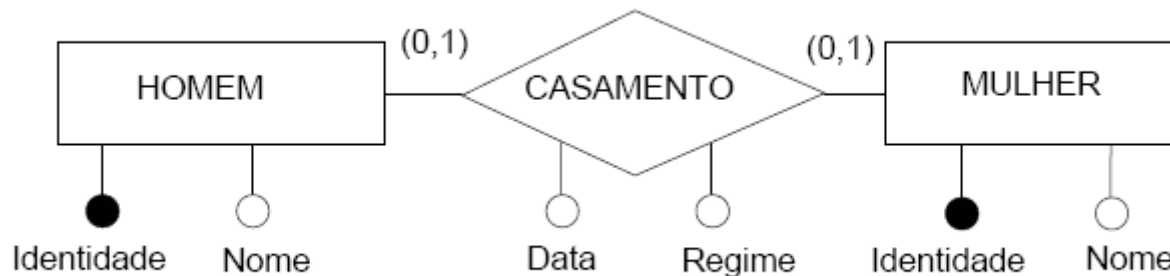
- Traduza o DER para o modelo lógico e, posteriormente para o modelo físico.



Conferencia(codconferencia, nome, data_instalacao, endereco)

Exercício de tradução

- Traduza o DER para o modelo lógico e, posteriormente para o modelo físico.



Adição de colunas

Homem(identidadeH, nome)

Mulher(identidadeM, nome, #identidadeH, data, regime)

Tabela Própria

Homem(identidadeH, nome)

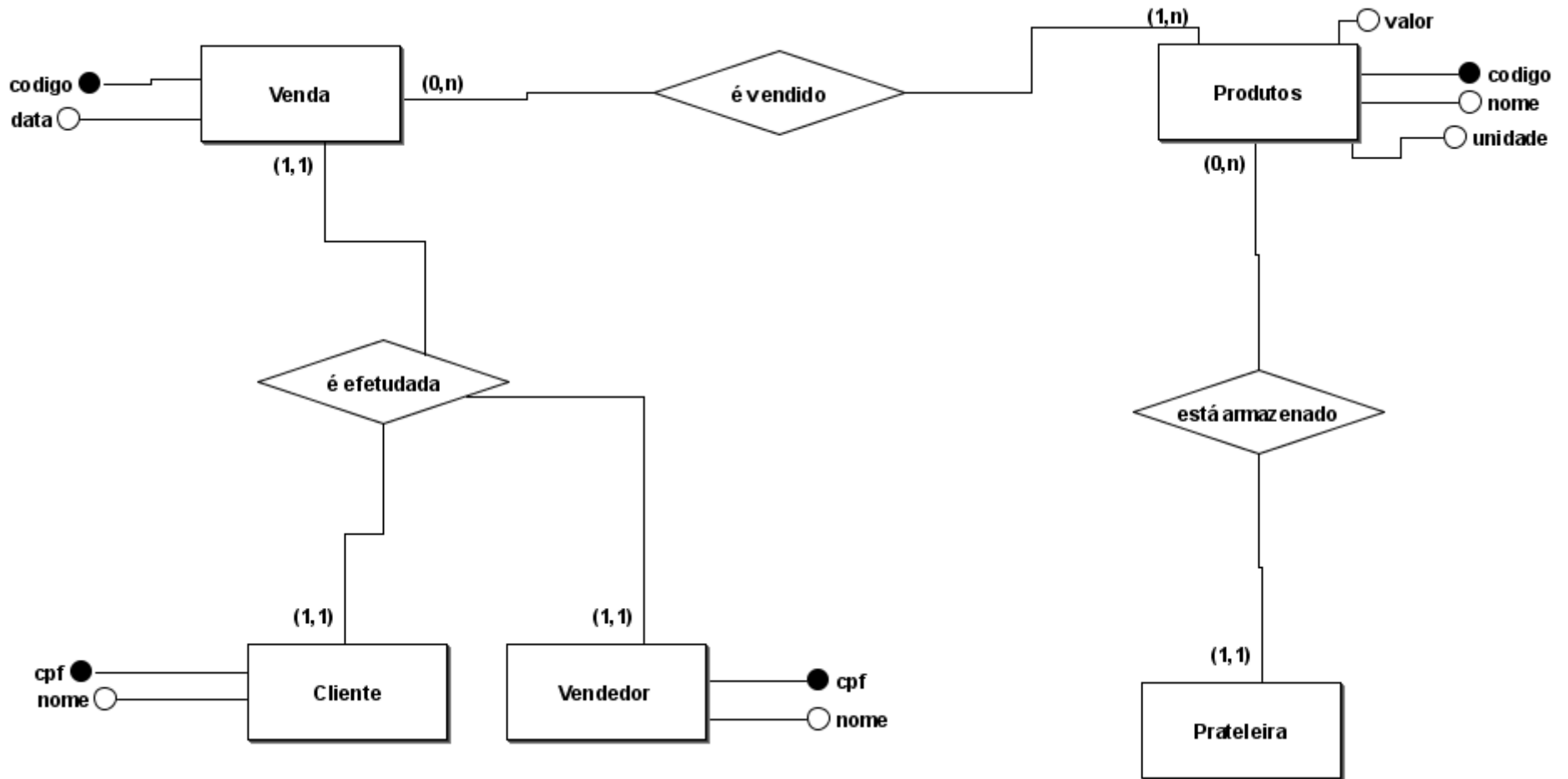
Mulher(identidadeM, nome)

Casamento(#identidadeH, #identidadeH, data, regime)

Exercício de Modelagem

Deseja-se construir um banco de dados para um sistema de vendas. Em cada venda são vendidos vários produtos e um determinado produto pode aparecer em diferentes vendas. Cada venda é efetuada por um vendedor para um determinado cliente. Um produto está armazenado em uma prateleira. Identifique as possíveis entidades com seus atributos, bem como, os relacionamentos. Faça um desenho do ER e um diagrama de ocorrências. Procure usar relacionamentos binários e ternários, pelo menos um de cada.

Exercício de Modelagem



Exercício de tradução

- Para o DER criado para construir um banco de dados para um sistema de vendas, crie o modelo lógico e o modelo físico.
- Venda (codigo, data, hora, #cpfvendedor, #cpfcli)
- Produtos (codigo, descricao, unidade, preco, #codprat)
- Vendedor (cpf, nome)
- Cliente (cpf, nome)
- Prateleira (codigo, fila, altura)
- VendaProdutos (#codvenda, #codprod, quantidade)

Modelo ER (Entidade-Relacionamento)

Parte 3 (Extensões e variações do modelo ER: agregação e generalização/especialização)

Transformação DER para modelo lógico

I. Tradução inicial de entidades e respectivos atributos;

II. Tradução de relacionamentos (auto, binário, ternário, entidade fraca, **agregação) e respectivos atributos;**

III. Tradução de generalizações/especializações e respectivos atributos;

Tradução de relacionamentos com agregação

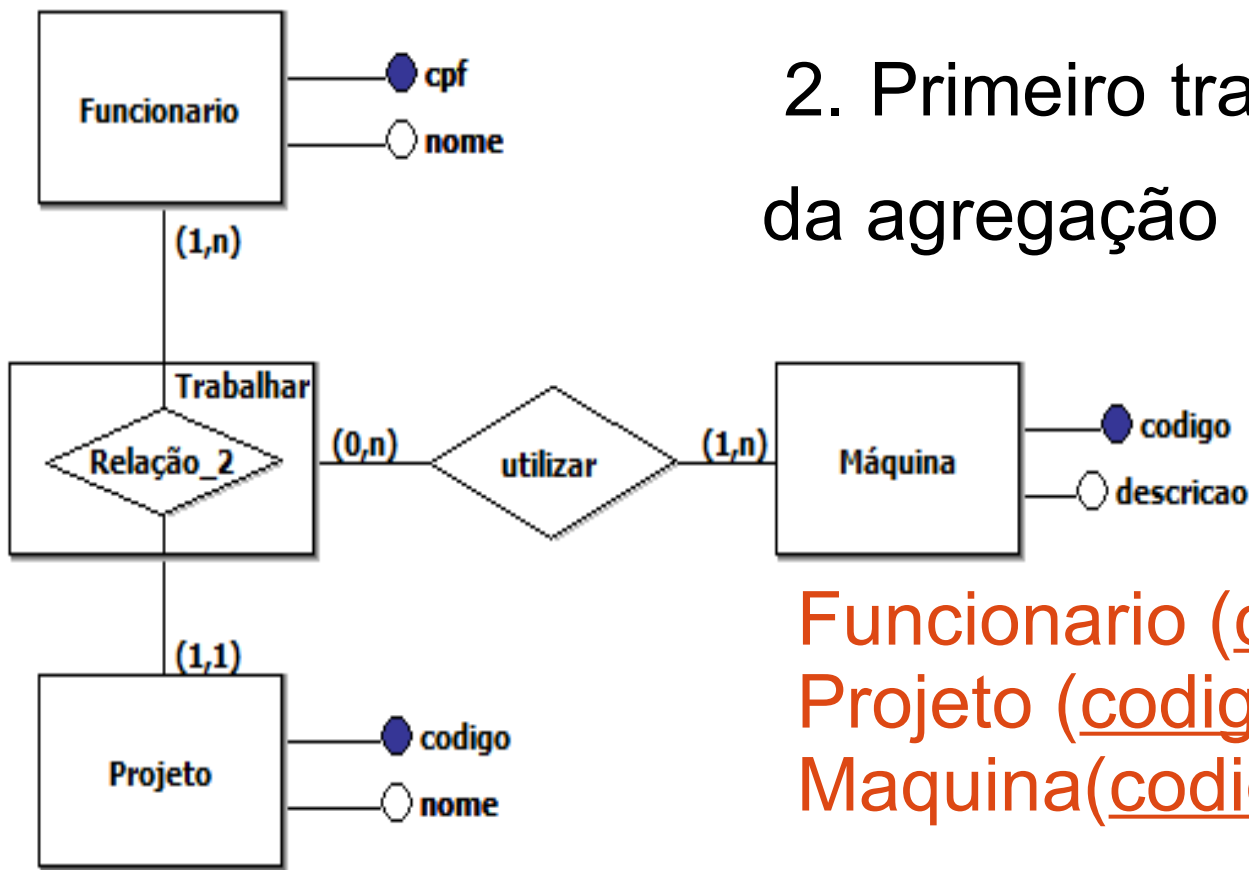
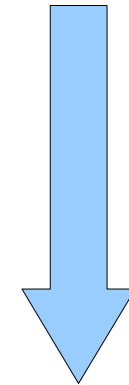
1. Traduzir inicialmente as entidades que participam da agregação

Funcionario (cpf, nome)

Projeto (codigo, nome)

Maquina(codigo, descricao)

2. Primeiro traduzir a relação interna da agregação



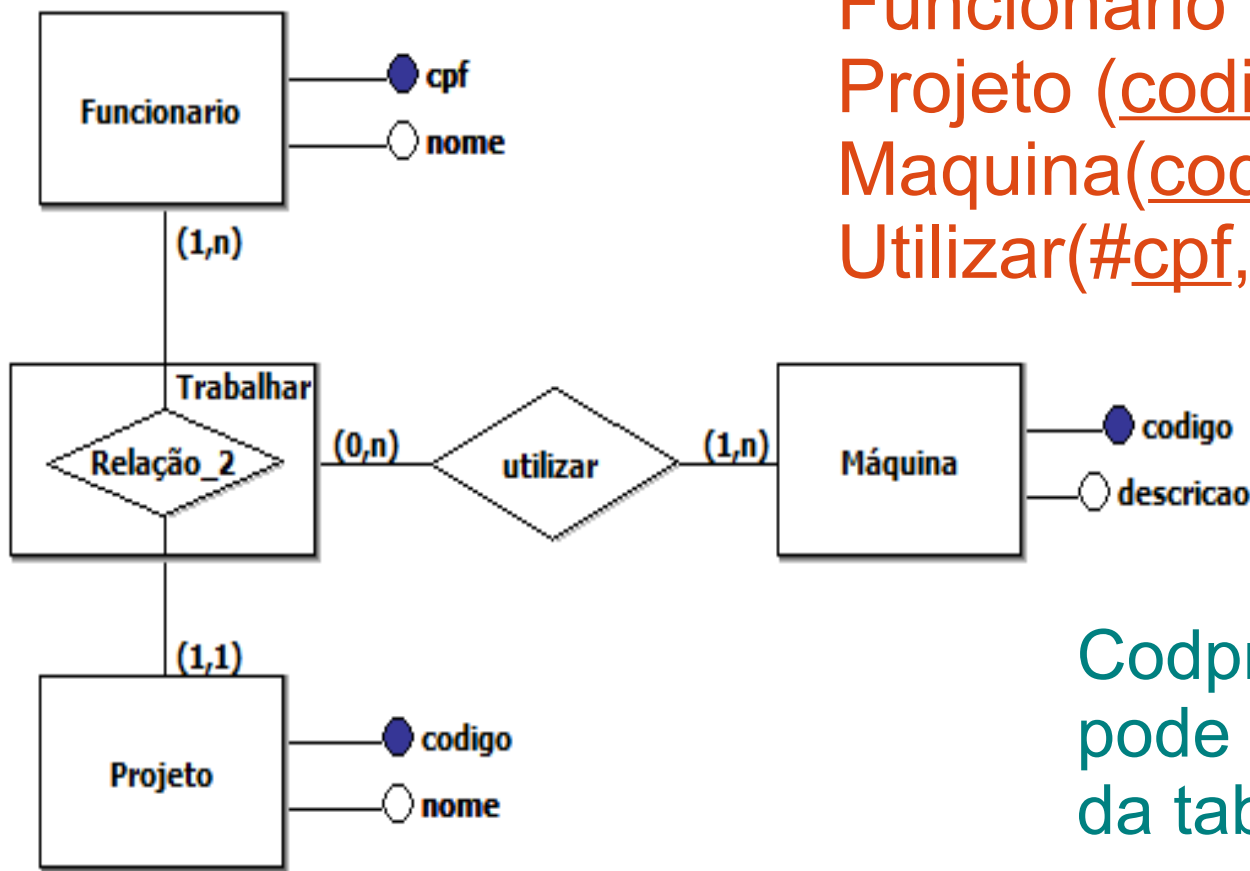
Funcionario (cpf, nome, #codproj)

Projeto (codigo, nome)

Maquina(codigo, descricao)

Tradução de relacionamentos com agregação

3. Criar uma nova tabela com a adição das colunas identificadoras das tabelas participantes do relacionamento. A chave primária será composta por estas colunas.



Funcionario (cpf, nome, #codproj)
Projeto (codigo, nome)
Maquina(codigo, descricao)
Utilizar(#cpf, #codproj, #codmaq)

Codproj da tabela Funcionário
pode ser removido por causa
da tabela Utilizar.

Tradução de relacionamentos com agregação

Funcionario (cpf, nome)

Projeto (codigo, nome)

Maquina(codigo, descricao)

Utilizar(#cpf, #codproj, #codmaq)

```
create table projeto(  
  codigo int not null primary key,  
  nome varchar(40) not null);
```

```
create table funcionario(  
  cpf int not null primary key,  
  nome varchar(40) not null);
```

```
create table maquina(  
  codigo int not null primary key,  
  descricao varchar(40) not null);
```

```
create table utilizar(  
  cpf int not null ,  
  codproj int not null,  
  codmaq int not null,  
  primary key (cpf, codproj, codmaq),  
  foreign key (cpf) references funcionario(cpf),  
  foreign key (codproj) references projeto(codigo),  
  foreign key (codmaq) references maquina(codigo));
```

Tradução de generalização/especialização

- Duas abordagens:
 - Uso de uma tabela para cada entidade da hierarquia
 - Uso de uma única tabela para toda hierarquia

Tradução de generalização/especialização

- Uso de uma **tabela única** para toda hierarquia:
 1. chave primária correspondente ao identificador da entidade mais genérica (para entidade genérica e especializada);
 2. uma coluna **tipo** (caso não exista), para identificar que tipo de entidade especializada está sendo representada por cada linha da tabela;

Tradução de generalização/especialização

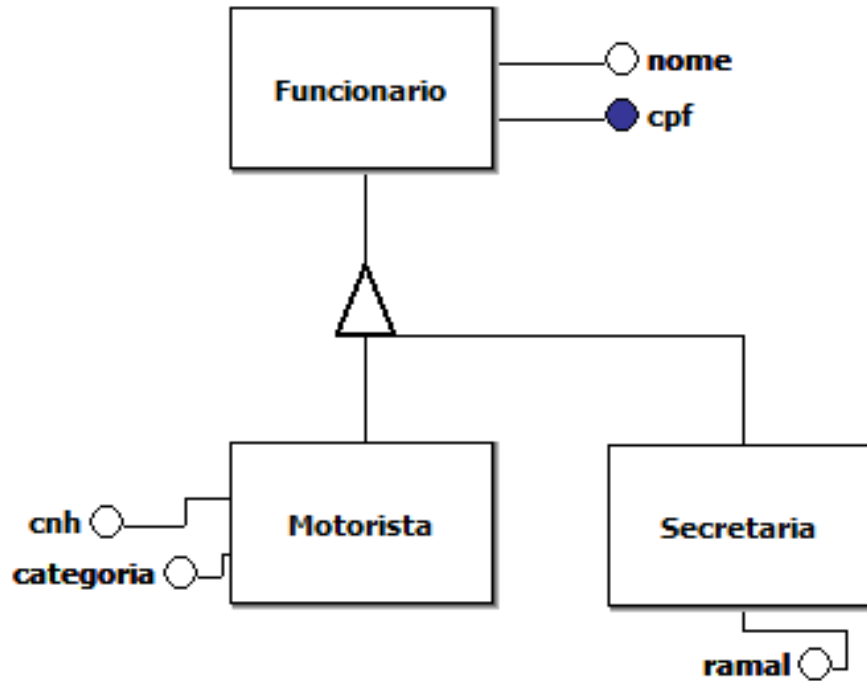
- 3. uma coluna para cada atributo da entidade genérica;
- 4. colunas referentes aos relacionamentos dos quais participa a entidade genérica (para relacionamentos implementados pela alternativa de adicionar colunas à tabela da entidade genérica);

Tradução de generalização/especialização

5. uma coluna para cada atributo da entidade especializada (como opcionais, pois só terão valores quando a linha for referente à entidade especializada em questão;

6. colunas referentes aos relacionamentos dos quais participa cada entidade especializada e que sejam implementados através da alternativa de adicionar colunas à tabela de entidade (colunas definidas como opcionais, pois terão valores somente na linha referente à entidade especializada em questão)

Tradução de generalização/especialização



```
create table funcionario(
    cpf int not null primary key,
    nome varchar(40) not null,
    cnh int not null,
    categoria char not null,
    ramal int not null,
    tipo char not null);
```

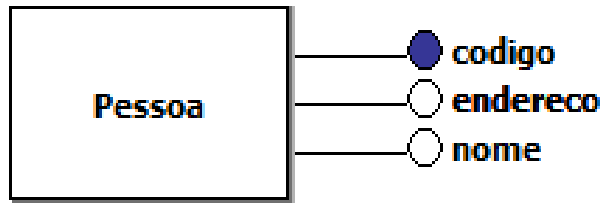
Funcionario (cpf, nome, cnh, categoria, ramal, tipo)

Tradução de generalização/especialização

- Uma tabela para cada entidade especializada:

1. Tabela referente à entidade generalizada e tabelas referentes a cada uma de suas especializações possuem a mesma chave primária;
2. Todas as colunas são agrupadas em uma única tabela
3. uma coluna **tipo** (opcional), para identificar que tipo de entidade especializada está sendo representada por cada linha da tabela;

Tradução de generalização/especialização



Pessoa (codigo, nome, endereco)

Fisica (#codigo, cpf, rg)

Juridica(#codigo, cnpj, iss)

```
create table pessoa(  
    codigo int not null primary key,  
    nome varchar(40) not null,  
    endereco varchar(40) not null,  
    tipo char not null);
```

```
create table fisica(  
    codigo int not null primary key,  
    cpf int not null,  
    rg varchar(10) not null,  
    foreign key (codigo) references  
    pessoa(codigo));
```

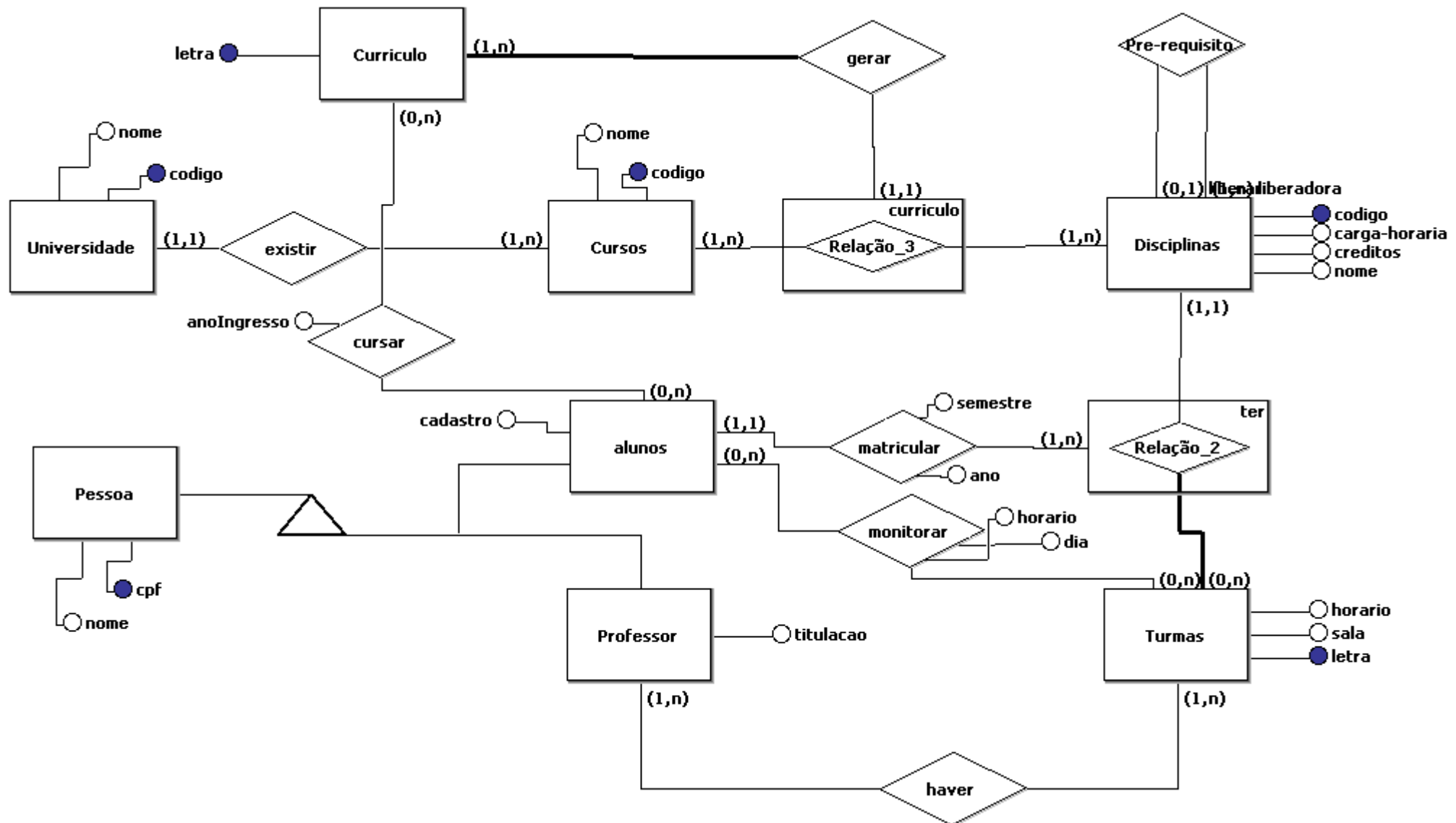
```
create table juridica(  
    codigo int not null primary key,  
    cnpj int not null,  
    iss int not null,  
    foreign key (codigo) references pessoa(codigo));
```

Exercício de Modelagem

- Em uma universidade, existem diversos cursos de graduação. Cada curso possui um conjunto de disciplinas. Cada disciplina possui um conjunto de turmas. Para cada turma, há um ou mais professores. Os alunos realizam matrículas nas turmas. Para algumas turmas onde há mais de 30 alunos matriculados, existe um monitor alocado a esta turma. Crie um DER completo. Use a relação de agregação para modelar os monitores. Use a relação de generalização/especialização para modelar uma entidade em comum para professores e alunos.

Exercício de Modelagem

- Exercício Universidade



Exercício de Tradução

Universidade(codigo, nome)

Cursos(codigo, nome, #coduniversidade)

Disciplinas(codigo, nome, carga-horaria, créditos)

CursoDisciplina(#codcurso, #coddisc)

Curriculo(#codcurso, letra)

CurriculoDisciplinas(#codcurso, #letra, #coddisc)

PreRequisito(#coddisc, #coddisclibera)

Turmas(#codisc, letra, horario, sala)

Pessoa(cpf, nome, tipo)

Alunos(cpf, cadastro)

AlunoCurso(cpf, codcurso, letra, anoingresso)

Matricular(#cpf, #coddisc, #letra, semestre, ano)

MonitorarDisciplina(#cpf, #coddisc, #letra, horario, dia)

Professor(cpf, titulacao)

ProfessorTurma(cpf, #coddisc, #letra)