

# [MAC0439] Laboratório de Bancos de Dados

## Aula 6

### Uma Revisão sobre Projeto Conceitual de Bancos de Dados usando o Modelo Entidade-Relacionamento (Estendido)

Kelly Rosa Braghetto

DCC-IME-USP

19 de agosto de 2016

# Projeto de bancos dados

Envolve as seguintes etapas:

1. Levantamento e análise dos requisitos
2. Projeto conceitual
3. Projeto lógico
4. Projeto físico

# Fase 1: Levantamento e análise dos requisitos

Nessa fase, o projetista:

- ▶ Registra concisamente os requisitos dos usuários com relação aos dados
- ▶ Define requisitos funcionais (operações/transações) conhecidos das aplicações

Sobre os requisitos:

- ▶ São levantados por meio de entrevistas com os usuários
- ▶ Incluem os dados exigidos para processamento, os seus relacionamentos e as informações relevantes para a escolha da plataforma de software para o BD

## Fase 2: Projeto conceitual

Fase de criação de um esquema conceitual para o BD, utilizando um modelo de dados conceitual de alto nível.

### Esquema conceitual (definição):

Descrição concisa de requisitos de dados dos usuários, contendo descrições detalhadas sobre os tipos de entidades, relacionamentos e restrições

### Modelos conceituais mais usados:

- ▶ Modelo ER (Entidade-Relacionamento)
- ▶ Diagramas de classe da UML

## Fase 3: Projeto lógico

Essa fase corresponde ao mapeamento do esquema conceitual para um modelo de dados de implementação.

⇒ Passo necessário para a implementação do BD utilizando um SGBD comercial.

Exemplos de modelos de implementação bastante usados:

- ▶ modelo relacional
- ▶ modelo objeto-relacional

Refinamento do esquema (etapa opcional):

- ▶ Ele identifica problemas em potencial no modelo lógico criado e aplica técnicas para melhorar o modelo

## Fase 4: Projeto físico

Fase na qual são definidas as estruturas de armazenamento interno, índices, caminhos de acesso, organizações de arquivos para os arquivos do BD e outros ajustes finos.

- ▶ Finalidade: otimizar o desempenho das operações de consulta e manipulação dos dados
- ▶ Pode até mesmo modificar o projeto de BD resultante das fases anteriores, a fim de satisfazer critérios de desempenho desejados
- ▶ Exemplo: “denormalização” das relações em *Data Warehouses* (bancos de dados analíticos, em que o desempenho para os operações de consulta é um requisito importante)

# Modelo Entidade-Relacionamento (ER)

- ▶ Criado por Peter Chen em 1976; ainda hoje é o mais usado para a modelagem conceitual de BDs
- ▶ É simples e de interpretação intuitiva (mesmo para usuários não-especialistas)
- ▶ Descreve os dados com base em três conceitos principais:
  - ▶ **entidades** – “algo” do mundo real, com uma existência independente (ex.: pessoa, carro, disciplina, etc.)
  - ▶ **relacionamentos** – associam entidades (ex.: pessoa tem carro)
  - ▶ **atributos** – propriedades particulares que descrevem uma entidade (ex.: nome da pessoa, cor do carro, etc.)

# Tipos de atributos no modelo ER

## Simple (atômicos) × Compostos

- ▶ **Atributos compostos** – podem ser divididos em partes menores. Ex.: o atributo endereço pode ser dividido em Rua, Cidade, Estado e CEP.
- ▶ **Atributos simples (ou atômicos)** – atributos que são indivisíveis.

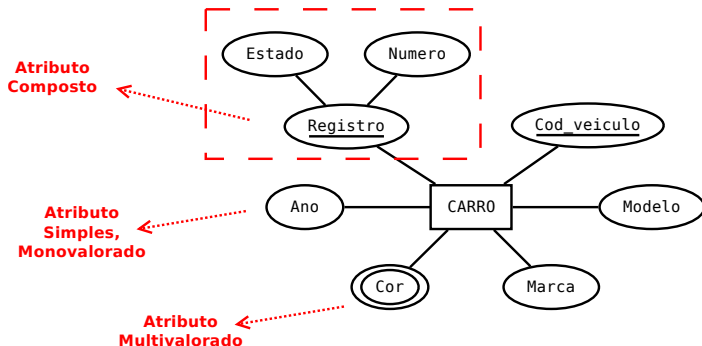
## Monovalorados × Multivalorados

- ▶ **Monovalorado** – atributo que tem um único valor para uma dada entidade. Ex.: o atributo idade para uma pessoa.
- ▶ **Multivalorado** – atributo para o qual diferentes entidades podem ter diferentes quantidades de valores. Ex.: o atributo titulação para uma pessoa.



# Notação para tipos de entidades e atributos no modelo ER

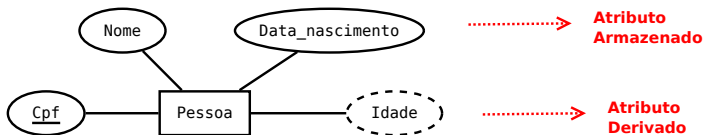
Exemplo de atributos simples, compostos, monovalorados, multivalorados



# Tipos de atributos no modelo ER

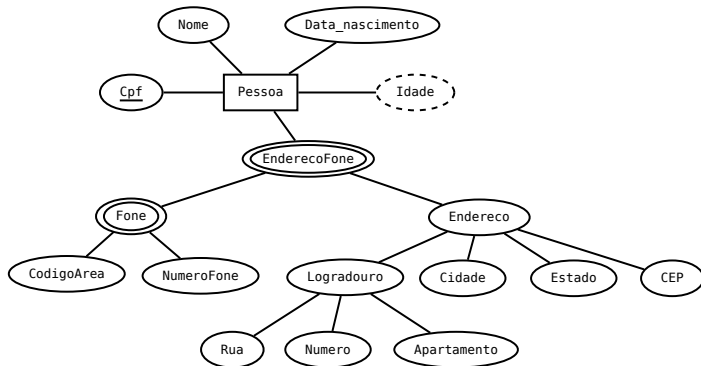
## Armazenados × Derivados

- ▶ **Atributo derivado** – é derivado a partir de outro(s) atributo(s) ou entidade(s) relacionado(s)
- ▶ **Atributo armazenado** – é um atributo que não é derivado.



# Notação para tipos de entidades e atributos no modelo ER

Exemplo de atributo complexo (composto e multivalorado, com aninhamentos)



# Tipo de entidade e chaves

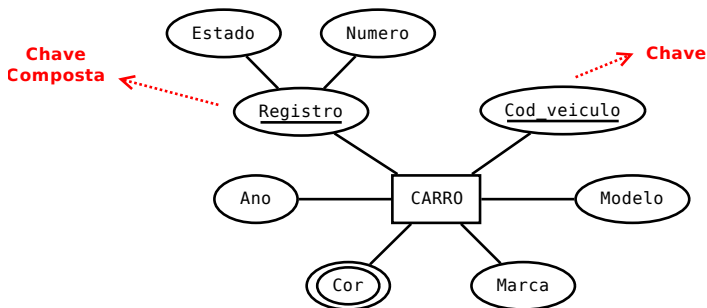
Um **tipo de entidade** define um conjunto de entidades que possuem os mesmos atributos.

## Conceitos importantes:

- ▶ **Atributo-chave (restrição de exclusividade)** – atributo cujo valor é distinto para toda entidade pertencente ao conjunto de entidades do tipo (ou seja, que identifica cada entidade de forma unívoca).
- ▶ **Chave composta** – é formada por diversos atributos, cuja a combinação dos valores é distinta para cada entidade.
- ▶ Alguns tipos de entidade têm mais de um atributo chave ou chave composta (ex.: NUSP e CPF para aluno). Outros, nem têm uma chave – são os chamados **tipos de entidade fraca**.

# Notação para tipos de entidades e atributos no modelo ER

## Exemplo de chave e de chave composta



# valores dos atributos

## Valor NULL (nulo)

- ▶ É um valor especial, usado quando uma entidade **não possuiu um valor** para um atributo.
- ▶ O NULL serve tanto para indicar que um atributo **não se aplica** a uma dada entidade, quanto para indicar que o valor para um atributo de uma dada entidade é **desconhecido**.
- ▶ “Desconhecido” se aplica a dois casos distintos:
  - ▶ quando é sabido que existe um valor para o atributo, mas ele está faltando (**ex.:** Altura – todo mundo tem!)
  - ▶ quando não é sabido se o valor existe ou não (**ex.:** FoneResidencial – uma pessoa pode ou não ter)

## Tipo de relacionamento

Um **tipo de relacionamento**  $R$  entre  $n$  tipos de entidades  $E_1, E_2, \dots, E_n$  define um conjunto de associações (= relacionamentos) entre as entidades desses tipos.

Exemplo: relacionamento binário TRABALHA\_PARA entre os tipos de entidade EMPREGADO e DEPARTAMENTO

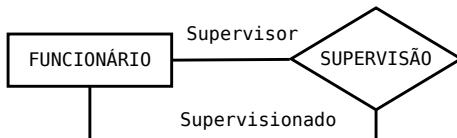


# Propriedades de um tipo de relacionamento

## Nomes dos papéis

- ▶ Só são estritamente necessários quando um mesmo tipo de entidade pode participar mais de uma vez em um mesmo tipo de relacionamento (= **relacionamento recursivo**).
  - ▶ Nesse caso, eles são fundamentais para definir o sentido de cada participação.

Ex.: tipo de relacionamento SUPERVISÃO, em que o tipo de entidade EMPREGADO participa duas vezes – uma no papel de *supervisor*, outra no papel de *supervisionado*.





# Restrições sobre tipos de relacionamento binários

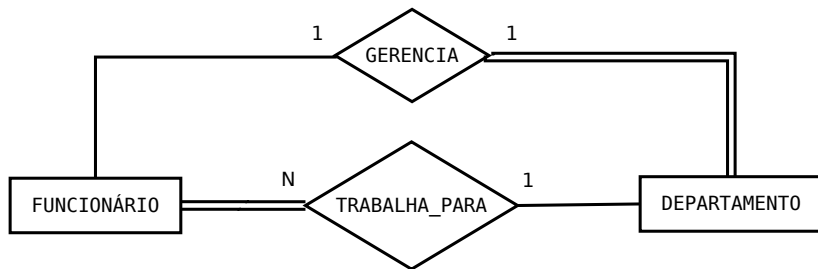
Existem restrições (determinadas por situações do minimundo) que limitam as combinações de entidades que podem participar de um relacionamento binário.

## Restrições possíveis para relacionamentos binários:

- ▶ **Razão de cardinalidade:** especifica o **número máximo** de instâncias do relacionamento em que uma entidade pode participar. As razões de cardinalidade possíveis são **1:1**, **1:N**, **N:1** e **M:N**.
- ▶ **Restrição de participação:** determina o **número mínimo** de instâncias de relacionamento em que uma entidade deve participar. A participação pode ser **total** ou **parcial**.

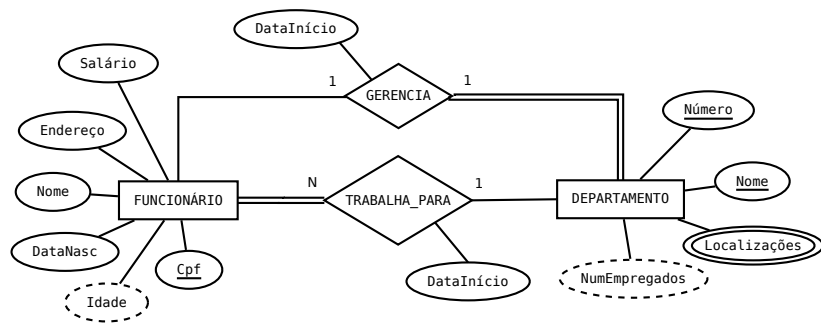
# Notação das restrições de cardinalidade e participação

Exemplos de tipos de relacionamento envolvendo diferentes restrições



## Atributos de tipos de relacionamento

Tipos de relacionamento podem ter atributos, de forma similar aos tipos de entidade.

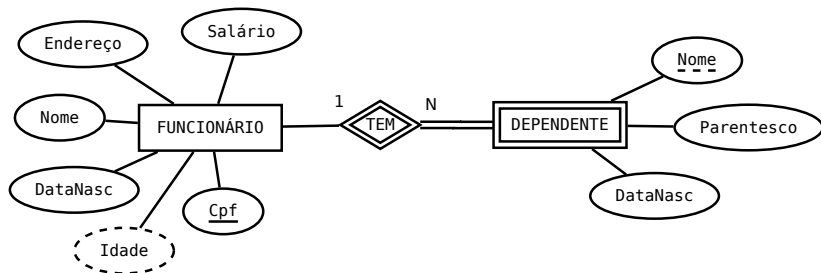


## Tipo de entidade fraca

- ▶ **Tipo de entidade forte** – tipo de entidade que possui um atributo-chave.
- ▶ **Tipo de entidade fraca** – tipo de entidade que não possui um atributo-chave.
  - ▶ Entidades de tipos de entidade fraca são identificadas por estarem relacionadas (associadas) a entidades de um outro tipo de entidade (chamado de **tipo de entidade identificador** ou **tipo de entidade proprietária**).
  - ▶ Esse tipo de relacionamento é chamado de **relacionamento identificador** do tipo de entidade fraca.

## Notação dos tipos de entidade fraca

Exemplo: Tipo de entidade fraca DEPENDENTE em um relacionamento identificador com FUNCIONÁRIO



## Tipo de entidade fraca

- ▶ Um tipo de entidade fraca sempre tem uma restrição de participação total em relação ao seu relacionamento identificador.
- ▶ Um tipo de entidade fraca normalmente tem uma **chave parcial**, que é um conjunto de atributos que identifica univocamente as entidades fracas que estão relacionadas a uma mesma entidade proprietária.
  - ▶ No pior caso, a chave parcial será a composição de todos atributos do tipo de entidade fraca.
- ▶ Quando um tipo de entidade fraca não é participante em tipos de relacionamento, então ele pode ser definido como um atributo complexo (composto, multivalorado) em seu tipo de entidade proprietária.

# Diagrama ER para o esquema EMPRESA

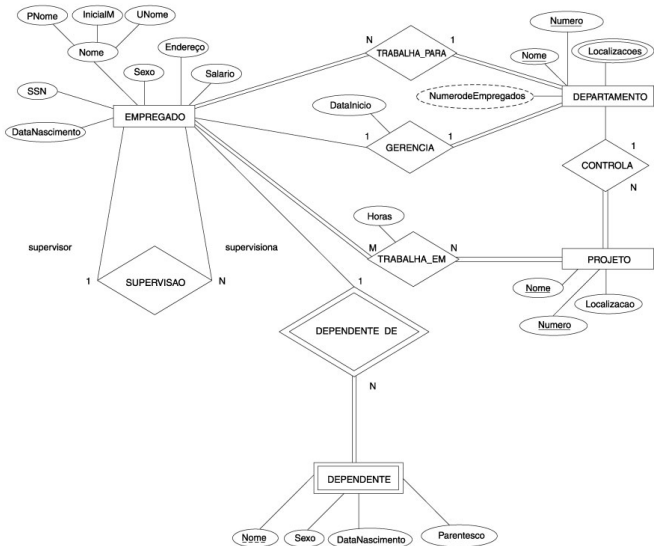


FIGURA 3.2 Um diagrama do esquema ER para o banco de dados EMPRESA.

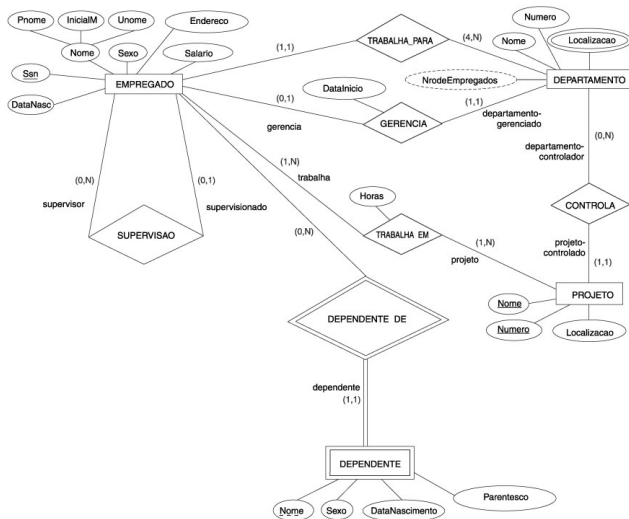
# Notações alternativas para o modelo ER

## Notação (min,max) para a razão de cardinalidade

- ▶ Ideia: associar um par (min,max) a cada participação de um tipo de entidade no relacionamento
- ▶ Cada entidade do tipo de entidade deve participar de pelo menos *min* e no máximo *max* entidades do relacionamento
- ▶  $min = 0$  implica em participação parcial
- ▶  $min > 0$  implica em participação total

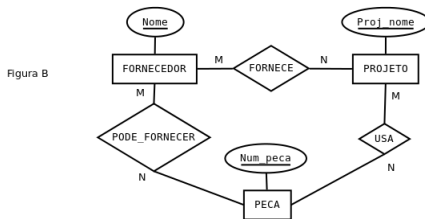
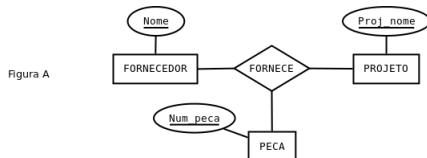


# Diagrama ER para o esquema EMPRESA – com notação (min,max) e o nomes de papéis



# Tipos de relacionamento de grau maior que dois

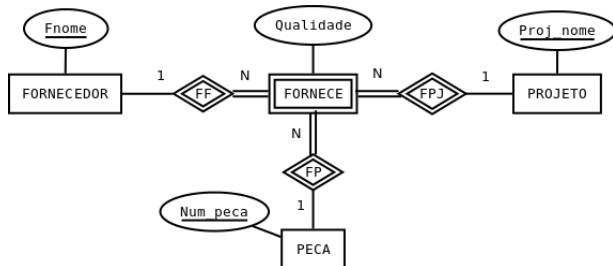
## Exemplo: tipo de relacionamento FORNECE



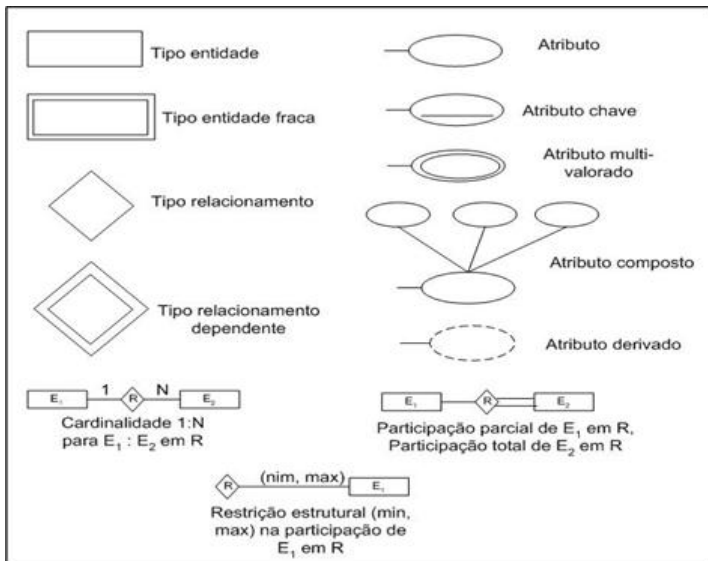
Os 3 relacionamentos binários (Figura B) **não** são equivalentes ao ternário da Figura A.

## Tipos de relacionamento de grau maior que dois

Exemplo: FORNECE como um tipo de entidade fraca



# Resumo da notação



# Modelo Entidade-Relacionamento Estendido

## Contexto (década de 1980)

- ▶ Desejo: projeto de bancos de dados que refletisse mais precisamente as restrições de dados
- ▶ Necessidade de conceitos adicionais (+ abstrações!) para a *modelagem semântica de dados*
- ▶ Novos modelos criados em áreas como as de *representação do conhecimento* (IA) e *modelagem de objetos* (ES)
- ▶ Resultado: **modelo ER estendido** (ou **EER**)

# Conceitos do modelo EER

- ▶ Todos os conceitos do modelo ER
- ▶ **Subclasse / Superclasse**
  - ▶ **Especialização / Generalização**
  - ▶ Herança de atributo e relacionamento
- ▶ **Categoria (ou Tipo de União)**

## EER – Subclasses e superclasses

- ▶ **Classe** – conjunto ou coleção de entidades; isso inclui qualquer construção do EER que agrupe as entidades, como os *tipos de entidade*, *subclasses*, *superclasses* e *categorias*
- ▶ Uma **subclasse**  $S$  é uma classe cujas entidades devem sempre ser um subconjunto das entidades de outra classe, chamada de **superclasse** do relacionamento classe/subclasse, indicado como  $C/S$ . Portanto, em  $C/S$ , temos que  $S \subseteq C$
- ▶ Uma entidade pode ser membro de várias subclasses
- ▶ Uma entidade membro de uma subclasse herda todos os atributos e relacionamentos da sua superclasse
- ▶ Uma subclasse por si só também é considerada um tipo de entidade

## EER – especialização / generalização

- ▶ **Especialização** – é o processo de definir um **conjunto de subclasses** de um tipo de entidade, chamado de **superclasse** da especialização.

Especialização = refinamento conceitual

- ▶ **Generalização** – é o processo inverso, de suprimir as diferenças entre vários tipos de entidade (subclasses) e identificar suas características comuns, generalizando-as em uma única superclasse.

Generalização = síntese conceitual

O relacionamento entre uma subclasse e sua superclasse é chamado de “**É-UM**”.



## EER – especialização/generalização

- ▶ Uma **especialização**  $Z = \{S_1, S_2, \dots, S_n\}$  é um conjunto de subclasses que têm a mesma superclasse  $Z$ , isto é,  $Z/S_i$ , para todo  $i$  em  $\{1, 2, \dots, n\}$ .
- ▶  $Z$  é chamada da **superclasse da especialização**, ou de **generalização das subclasses**  $\{S_1, S_2, \dots, S_n\}$ .
- ▶ Uma especialização  $Z$  é **total** sempre que:  $\bigcup_{i=1}^n S_i = Z$ .  
Do contrário,  $Z$  é **parcial**.
- ▶  $Z$  é **disjunta** (símbolo **d** no círculo) sempre que tivermos  $S_i \cap S_j = \emptyset$  para  $i \neq j$ .  
Do contrário,  $Z$  é **sobreposta** (símbolo **o** no círculo).

## Especialização – exemplo

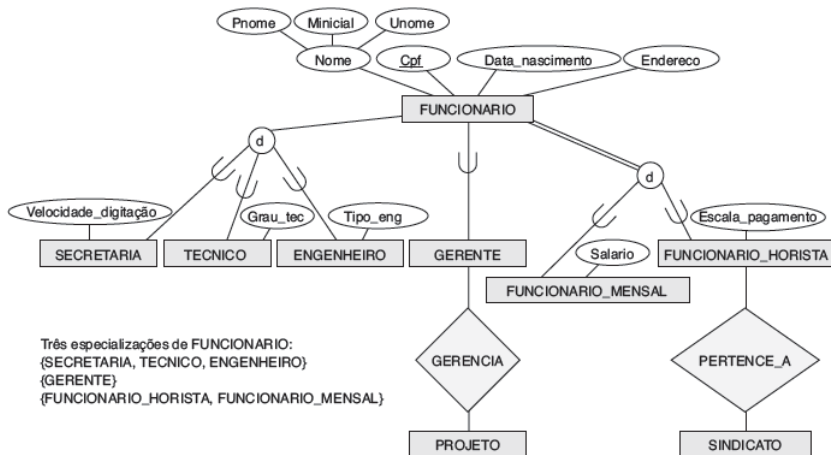
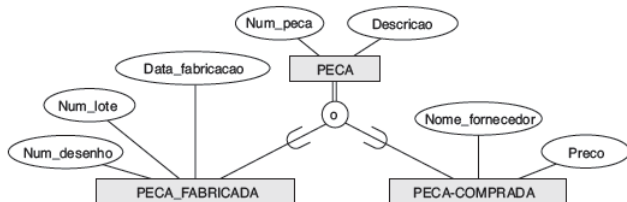


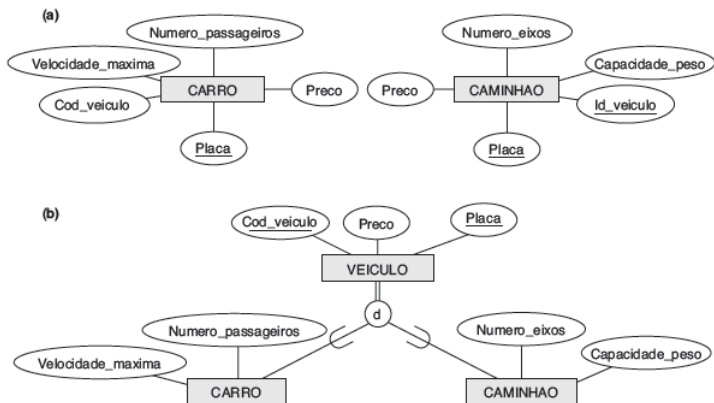
Diagram EER para representar subclasses e especialização

## Especialização – exemplo



## Especialização sobreposta

## Generalização – exemplo



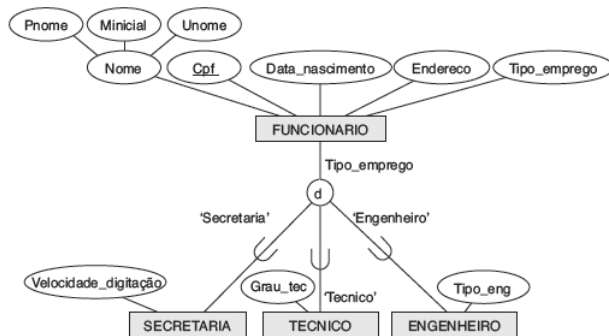
(a) Tipos de entidade CARRO e CAMINHAO.

(b) Generalizando esses tipos em VEICULO.

## EER – especialização/generalização

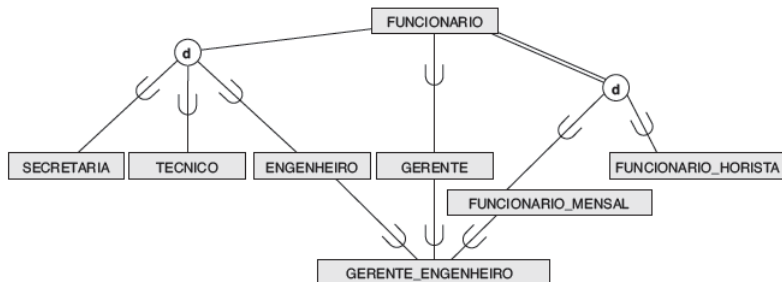
- ▶ O conjunto de subclasses que forma uma especialização é definido com base em algumas características de **distinção das entidades** da superclasse.
- ▶ Uma subclasse  $S$  de  $C$  é **definida por predicado** se um predicado  $p$  nos atributos de  $C$  for usado para especificar quais entidades de  $C$  são membros de  $S$ , isto é,  $S = C[p]$ , em que  $C[p]$  é o conjunto de entidades de  $C$  que satisfazem  $p$ .  
Um subclasse que não é definida por predicado é dita **definida por usuário**.
- ▶ Uma especialização  $Z$  (ou generalização  $G$ ) é **definida por atributo** se um predicado ( $A = c_i$ ), no qual  $A$  é um atributo de  $Z$  e  $c_i$  é um valor constante do domínio de  $A$ , for usado para especificar os membros de cada subclasse  $S_i$  em  $Z$ .  
Se  $c_i \neq c_j$ , para  $i \neq j$  e  $A$  é um atributo monovalorado, então a especialização será disjunta.

## Especialização – exemplo



Especialização definida por atributo sobre Tipo\_emprego

## Especialização – exemplo de reticulado



--

Um reticulado de especialização com a subclasse compartilhada GERENTE\_ENGENHEIRO

# Abstração de especialização/generalização

## Razões para se usar especializações no modelo EER

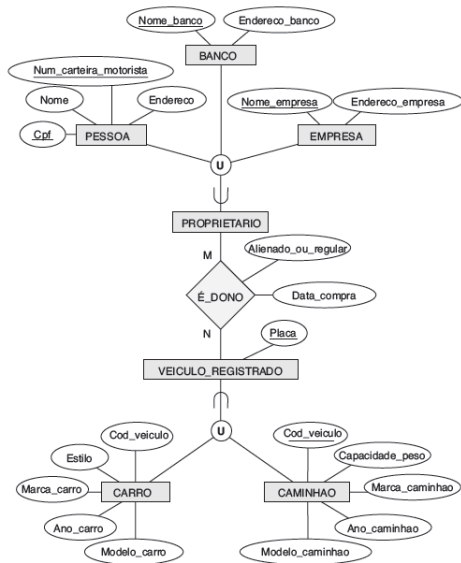
1. quando certos atributos podem ser usados somente em algumas das entidades da superclasse. Uma subclasse é definida de modo a agrupar as entidades para as quais esses atributos se aplicam;
2. quando apenas as entidades que são membros de uma subclasse podem participar de algum tipo de relacionamento.



## EER – categoria ou tipo de união

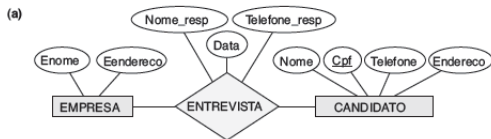
- ▶ Uma **categoria**  $T$  é formalmente definida como uma classe que é um subconjunto da união de  $n$  superclasses definidas  $D_1, D_2, \dots, D_n$ ,  $n > 1$ , e é especificada como:  
$$T \subseteq (D_1 \cup D_2 \cup \dots \cup D_n)$$
- ▶ Um predicado  $p_i$  nos atributos de  $D_i$  pode ser usado para especificar os membros de cada  $D_i$  que são membros de  $T$ . Se um predicado for especificado em todo  $D_i$ , temos:  
$$T = (D_1[p_1] \cup D_2[p_2] \cup \dots \cup D_n[p_n])$$
- ▶ Em uma **categoria total**,  $T = (D_1 \cup D_2 \cup \dots \cup D_n)$ , ou seja,  $T$  controla a **união de todas as entidades** em suas superclasses.
- ▶ Em uma **categoria parcial**,  $T$  pode controlar um **subconjunto da união**.

# EER – categorização



Duas categorias:  
PROPRIETARIO e  
VEICULO\_REGISTRADO

# Abstração de agregação - Exemplos



O tipo de relacionamento ENTREVISTA

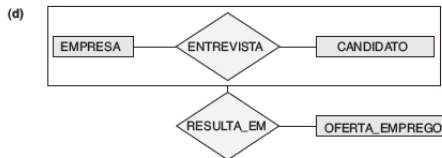


Inclusão de OFERTA\_EMPREGO em relacionamento ternário (**incorreto!**)

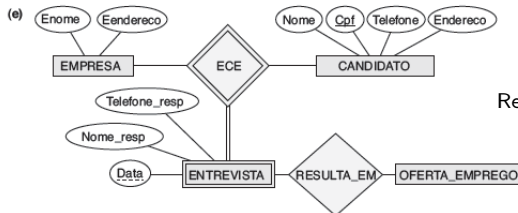


Relacionamento participando de outro relacionamento (**incorreto!**)

# Abstração de agregação - Exemplos



Usando agregação e um objeto composto (geralmente não permitido em ER)



Representação correta em ER

## Abstração de agregação

Um objeto agregado de alto nível faz-se especialmente necessário quando:

- ▶ quando o relacionamento que o deu origem deve ter o seu próprio identificador
- ▶ quando pode haver mais de uma instância de seu tipo de relacionamento gerador envolvendo as mesmas entidades
- ▶ ele mesmo está associado a um outro objeto por meio de um relacionamento
- ▶ ele possui atributos que não são comuns a todas as instâncias do seu relacionamento gerador

O relacionamento entre os objetos primitivos e seu objeto agregado é chamado de **É-UMA-PARTE-DE**.

# Ferramentas de apoio à modelagem conceitual (gratuitas)

Geram automaticamente código em SQL a partir dos modelos

- ▶ EERCASE (UFPE/CIn) –  
<https://sites.google.com/a/cin.ufpe.br/eercase/>
  - ▶ Modelos EE e EER com a notação usada no livro do Elmasri/Navathe
- ▶ PostgreSQL Database Modeler (PgModeler) –  
<http://www.pgmodeler.com.br/>
  - ▶ Usa um tipo de modelo que é uma mistura de modelagem conceitual e lógica
  - ▶ Específico para BDs mantidos no PostgreSQL
- ▶ MySQL Workbench –  
<http://www.mysql.com/products/workbench/>
  - ▶ Ampara a modelagem e a administração dos BDs
  - ▶ Específica para BDs mantidos no MySQL

# Ferramentas de apoio à modelagem conceitual (gratuitas)

Geram automaticamente código em SQL a partir dos modelos

- ▶ Open ModelSphere – <http://www.modelsphere.com/>
  - ▶ Não é específica para BDs (também permite a criação de modelos UML e modelos para processos de negócio)

Puramente desenho

- ▶ Diagram Editor (DIA) – <http://dia-installer.de/>
- ▶ draw.io – <https://www.draw.io/>
  - ▶ Ferramenta online

## Referências Bibliográficas

- ▶ *Sistemas de Bancos de Dados* (6ª edição), Elmasri e Navathe. Pearson, 2010.  
Capítulos 7 (Modelos ER) e 8 (Modelo EER)
- ▶ *Sistemas de Gerenciamento de Banco de Dados* (3ª edição), Ramakrishnan e Gehrke, 2008.  
Capítulo 2