

**Instituto Federal Minas Gerais - Campus Bambuí**  
**Departamento de Engenharia e Computação - DEC**  
**Curso de Engenharia de Computação - ENGCAMP**

# **Banco de Dados I**

## **03 - Construção de DER**

Marcos Roberto Ribeiro

2024



Um DER é um modelo formal, preciso, não ambíguo:

- Se várias pessoas lerem um mesmo DER, estas devem interpretá-lo da mesma maneira
- Um DER pode servir de entrada para um ferramenta *computer aided software engineering* (CASE)
- Apesar de ser uma representação de banco de dados em alto nível, todas as pessoas envolvidas com o projeto conceitual de banco de dados devem entender a semântica dos DER para evitar erros de comunicação

# Diferentes DER podem ser equivalentes

- Dois DER são equivalentes quando ambos geram o mesmo esquema lógico de banco de dados<sup>1</sup>
- Exemplo de equivalência:



<sup>1</sup>Veremos como fazer essa geração nas próximas aulas

# Atributo x entidade

---

Em uma indústria de automóveis, como registrar a cor dos automóveis?

- Como um atributo de automóvel?
- Ou como uma entidade associada a automóvel?

# Atributo x entidade

Em uma indústria de automóveis, como registrar a cor dos automóveis?

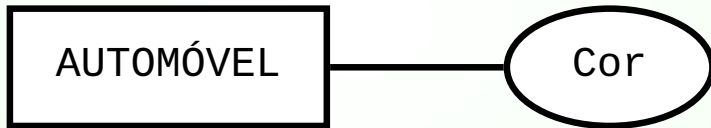
- Como um atributo de automóvel?
- Ou como uma entidade associada a automóvel?
- Caso o objeto a ser modelado esteja vinculado a outros objetos ou tenha propriedades próprias, deve ser modelado como entidade
- No caso da indústria de automóveis, isso poderia acontecer se fosse necessário armazenar dados do fabricante da tinta aplicada ao veículo:



# Atributo x entidade

Em uma indústria de automóveis, como registrar a cor dos automóveis?

- Como um atributo de automóvel?
- Ou como uma entidade associada a automóvel?
- Quando o objeto apresenta valores fixos, não possui propriedades próprias e não está relacionado com outros objetos, pode ser modelado como atributo



# Atributo x especialização

---

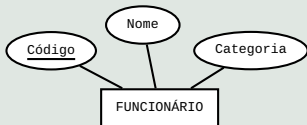
- Outro conflito que pode ocorrer na modelagem de um banco de dados é saber se um objeto deve ser modelado como atributo ou como especialização de uma entidade
- Uma especialização deve ser usada quando um objeto é um refinamento de outro e possui propriedades particulares
- Por exemplo, a categoria de funcionários de uma empresa

# Atributo x especialização

- Outro conflito que pode ocorrer na modelagem de um banco de dados é saber se um objeto deve ser modelado como atributo ou como especialização de uma entidade
- Uma especialização deve ser usada quando um objeto é um refinamento de outro e possui propriedades particulares
- Por exemplo, a categoria de funcionários de uma empresa

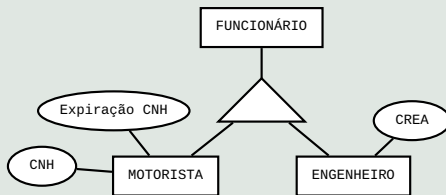
## Atributo

Se for preciso guardar apenas qual categoria pertence um funcionário, pode-se usar um atributo



## Especialização

Por outro lado, se a categoria de funcionário possuir propriedades próprias, o correto é usar a especialização

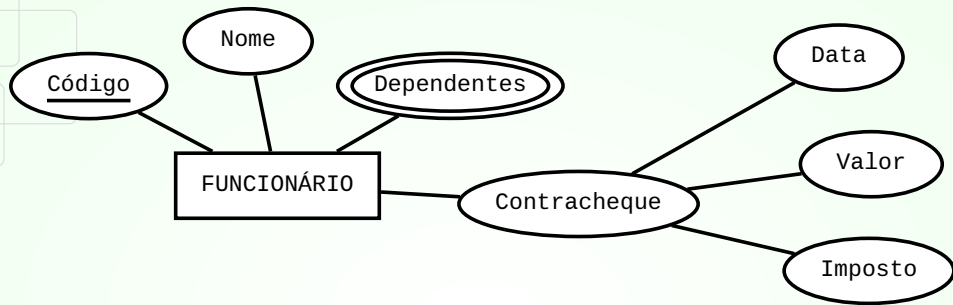






- Atributos multi-valorados e compostos são indesejáveis pelas seguintes razões:
  - A manipulação de atributos desse tipo é mais dispendiosa tanto para o SGBD quanto para o desenvolvimento de aplicações
  - Atributos multi-valorados e compostos podem induzir a erros de modelagem, tais como ocultar entidades e relacionamentos

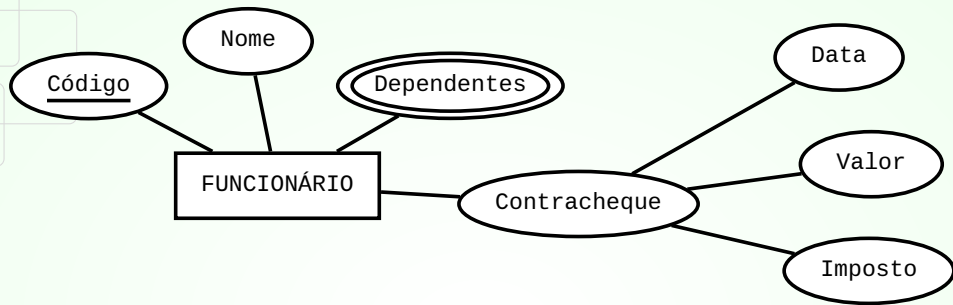
# Atributos multi-valorados e compostos - exemplo com problemas



## ■ Quais problemas podem ser observados?

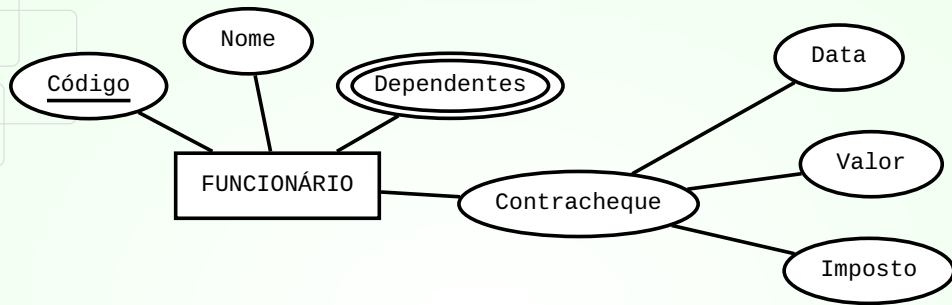
- Cada dependente possui características próprias como nome e nascimento
- Algumas dessas características podem impactar no salário do funcionário, como a idade do dependente
- O contracheque deve ser uma entidade, pois é composto por atributos muito diferentes
- Como exercício, crie esse DER

# Atributos multi-valorados e compostos - exemplo com problemas



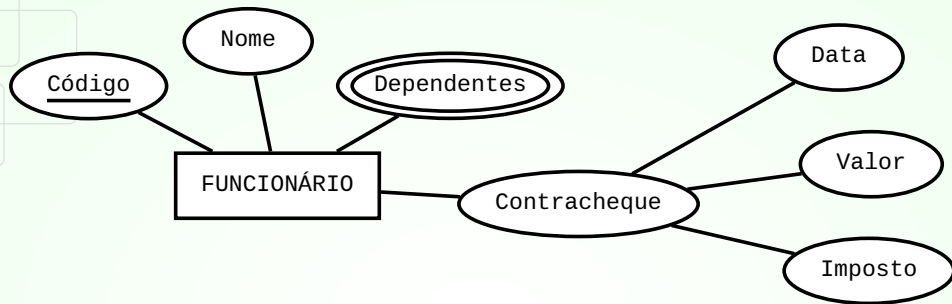
- Quais problemas podem ser observados?
  - Cada dependente possui características próprias como nome e nascimento
  - Algumas dessas características podem impactar no salário do funcionário, como a idade do dependente
  - O contracheque deve ser uma entidade, pois é composto por atributos muito diferentes
  - Como exercício, corrija esse DER

# Atributos multi-valorados e compostos - exemplo com problemas



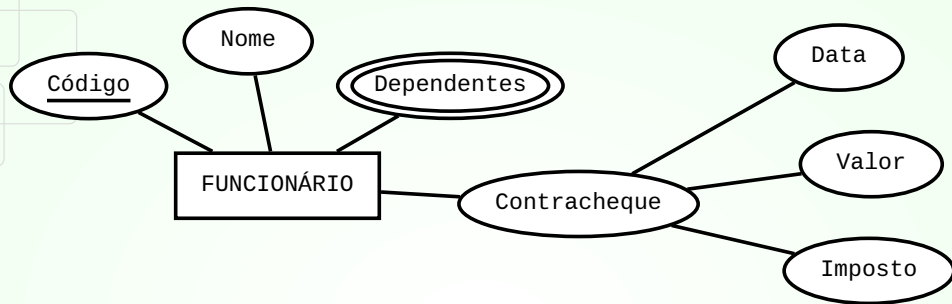
- Quais problemas podem ser observados?
  - Cada dependente possui características próprias como nome e nascimento
  - Algumas dessas características podem impactar no salário do funcionário, como a idade do dependente
  - O contracheque deve ser uma entidade, pois é composto por atributos muito diferentes
  - Como exercício, corrija esse DER

# Atributos multi-valorados e compostos - exemplo com problemas



- Quais problemas podem ser observados?
  - Cada dependente possui características próprias como nome e nascimento
  - Algumas dessas características podem impactar no salário do funcionário, como a idade do dependente
  - O contracheque deve ser uma entidade, pois é composto por atributos muito diferentes
  - Como exercício, corrija esse DER

# Atributos multi-valorados e compostos - exemplo com problemas



- Quais problemas podem ser observados?
  - Cada dependente possui características próprias como nome e nascimento
  - Algumas dessas características podem impactar no salário do funcionário, como a idade do dependente
  - O contracheque deve ser uma entidade, pois é composto por atributos muito diferentes
  - Como exercício, corrija esse DER



- Uma vez construído, o DER deve ser validado e verificado para garantir a construção de um “bom banco de dados”
- Para isso, o DER deve ser correto, completo e livre de redundância



- Um DER está correto quando não contém erros de modelagem

Erros sintáticos: quando o DER não respeita as regras de construção do modelo ER, por exemplo, ligação entre entidades diretamente, especialização de relacionamentos, dentre outros

Erros semânticos: ocorrem quando, mesmo sintaticamente correto, o DER reflete uma realidade inconsistente





- Um DER está correto quando não contém erros de modelagem

**Erros sintáticos:** quando o DER não respeita as regras de construção do modelo ER, por exemplo, ligação entre entidades diretamente, especialização de relacionamentos, dentre outros

**Erros semânticos:** ocorrem quando, mesmo sintaticamente correto, o DER reflete uma realidade inconsistente



- Um DER está correto quando não contém erros de modelagem

**Erros sintáticos:** quando o DER não respeita as regras de construção do modelo ER, por exemplo, ligação entre entidades diretamente, especialização de relacionamentos, dentre outros

**Erros semânticos:** ocorrem quando, mesmo sintaticamente correto, o DER reflete uma realidade inconsistente

# Exemplos de erros semânticos

## Estabelecer associações incorretamente

- Um exemplo é associar, a uma entidade, um atributo que pertence a outra entidade
- Por exemplo, em um DER com as entidade **CLIENTE** e **VENDEDOR**, associar o nome do vendedor ao cliente para indicar o vendedor de cada cliente. Como corrigir esse erro?

## Usar uma entidade como atributo de outra entidade

- Um exemplo seria um DER com uma entidade **BANCO** e, no mesmo DER, uma **ENTIDADE** cliente com um atributo banco
- Cada objeto modelado deve ser representado uma única vez no DER

## Usar um número incorreto de entidade no relacionamento

- Como, por exemplo, fundir em um relacionamento ternário dois relacionamentos binários

# Exemplos de erros semânticos

## Estabelecer associações incorretamente

- Um exemplo é associar, a uma entidade, um atributo que pertence a outra entidade
- Por exemplo, em um DER com as entidade **CLIENTE** e **VENDEDOR**, associar o nome do vendedor ao cliente para indicar o vendedor de cada cliente. Como corrigir esse erro?

## Usar uma entidade como atributo de outra entidade

- Um exemplo seria um DER com uma entidade **BANCO** e, no mesmo DER, uma **ENTIDADE** cliente com um atributo banco
- Cada objeto modelado deve ser representado uma única vez no DER

## Usar um número incorreto de entidade no relacionamento

- Como, por exemplo, fundir em um relacionamento ternário dois relacionamentos binários

# Exemplos de erros semânticos

## Estabelecer associações incorretamente

- Um exemplo é associar, a uma entidade, um atributo que pertence a outra entidade
- Por exemplo, em um DER com as entidade **CLIENTE** e **VENDEDOR**, associar o nome do vendedor ao cliente para indicar o vendedor de cada cliente. Como corrigir esse erro?

## Usar uma entidade como atributo de outra entidade

- Um exemplo seria um DER com uma entidade **BANCO** e, no mesmo DER, uma **ENTIDADE** cliente com um atributo banco
- Cada objeto modelado deve ser representado uma única vez no DER

## Usar um número incorreto de entidade no relacionamento

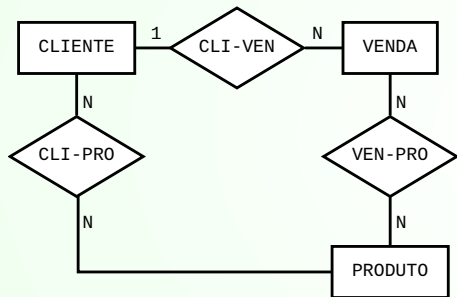
- Como, por exemplo, fundir em um relacionamento ternário dois relacionamentos binários



- Um DER completo deve fixar todas as propriedades desejáveis do banco de dados
- Para verificar se o modelo está completo, podem ser feitos dois testes:
  - Ver se todos os dados que devem ser obtidos do banco de dados estão presentes
  - Ver se todas as transações de modificação do banco de dados podem ser executadas

# DER livre de redundâncias

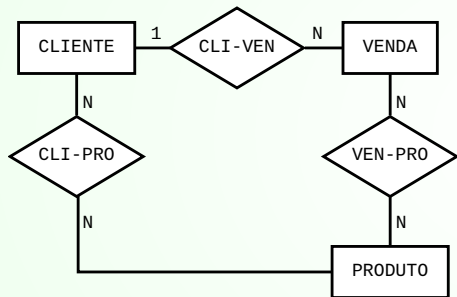
- O DER deve ser mínimo, ou seja, não deve conter conceitos redundantes
- Um tipo de redundância que pode aparecer são relacionamentos redundantes que são resultado da combinação de outros relacionamentos com as mesmas entidades



- O relacionamento **CLI-PRO** é redundante, pois pode ser obtido por meio dos relacionamentos **CLI-VEN** e **VEN-PRO**
- Como é possível obter a venda para cada cliente e os produtos de cada venda, por transição, é possível obter os produtos vendidos para cada cliente
- Nesse caso, o relacionamento pode ser eliminado sem perda de informação

# DER livre de redundâncias

- O DER deve ser mínimo, ou seja, não deve conter conceitos redundantes
- Um tipo de redundância que pode aparecer são relacionamentos redundantes que são resultado da combinação de outros relacionamentos com as mesmas entidades



- O relacionamento **CLI-PRO** é redundante, pois pode ser obtido por meio dos relacionamentos **CLI-VEN** e **VEN-PRO**
- Como é possível obter a venda para cada cliente e os produtos de cada venda, por transição, é possível obter os produtos vendidos para cada cliente
- Nesse caso, o relacionamento pode ser eliminado sem perda de informação

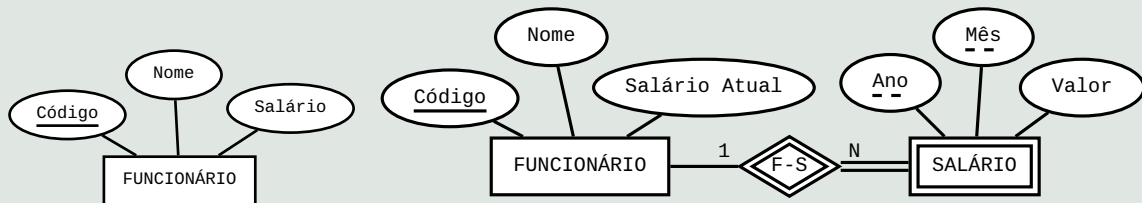




- Certos bancos de dados, por necessidades de informações no futuro, ou até mesmo por questões legais, precisam guardar históricos de alterações das informações
- Para esses bancos, devemos considerar o aspecto temporal para que a modelagem seja feita corretamente

## Atributos cujos valores modificam com o tempo

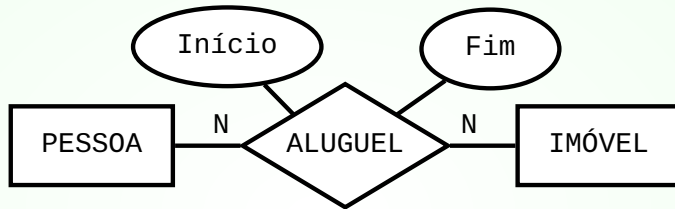
- Como exemplo, vamos considerar um sistema de pagamento, é preciso guardar todas as mudanças sobre o salário do funcionário e não somente o salário atual
- Nessa situação, o salário deixa de ser um atributo e passa a ser uma entidade



- Por que o atributo *Salário Atual*?
- Por que a entidade salário é fraca? E suas chaves?

# Relacionamentos cujos valores modificam com o tempo

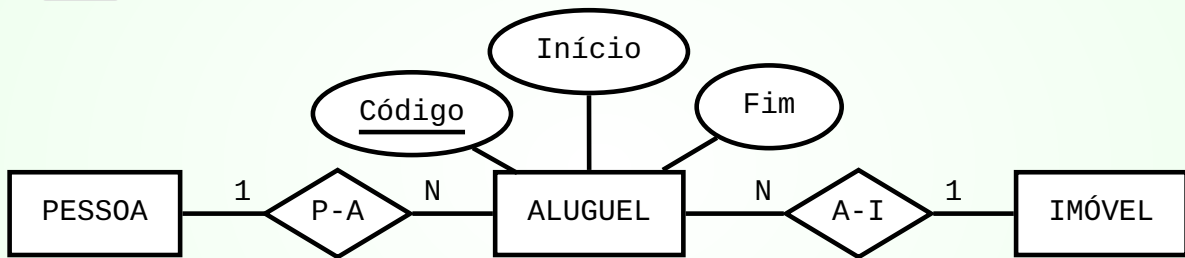
- Outra situação que pode acontecer é a necessidade de guardar informações históricas de um relacionamento



- Uma pessoa pode alugar mais de um imóvel e um imóvel pode ser alugado por mais de uma pessoa
- O problema é alugar o mesmo imóvel para a mesma pessoa mais de uma vez

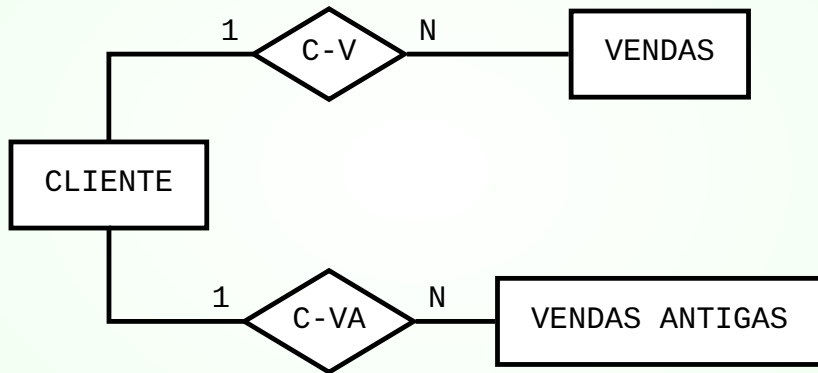
# Relacionamentos cujos valores modificam com o tempo (solução)

- Para resolver o problema do aluguel temos que transformar o relacionamento **ALUGUEL** em entidade

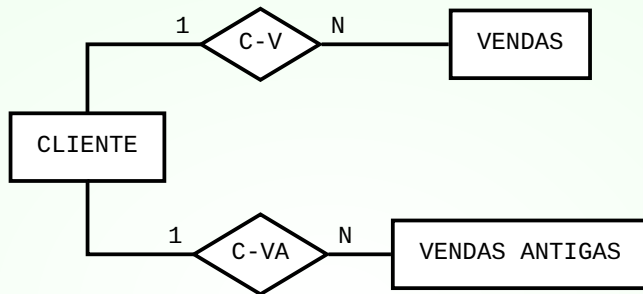


# Separação de dados recentes e antigos

- Em bancos de dados com enormes quantidades de informação em uma única entidade, é possível duplicar tal entidade para obter um melhor desempenho
- Dessa forma, separando os dados recentes de dados antigos



# Separação de dados recentes e antigos (observações)



- Essa prática deve ser feita somente em situações extremas, já que acarreta uma alta complexidade ao banco e ao sistema que o utiliza
- Todas as entidades ligadas a **VENDAS** também devem ser ligadas a **VENDAS ANTIGAS**
- Além disso, os sistemas que trabalham com o banco de dados devem amenizar o máximo possível essa separação de dados para os usuários

- O processo de modelagem é um processo incremental, ou seja, um modelo de banco de dados não é construído em um único passo, mas em muitos pequenos passos
- Gradativamente, o modelo vai sendo enriquecido com novos conceitos que vão sendo ligados aos existentes ou os existentes vão sendo aperfeiçoados
- Uma estratégia de modelagem ER é uma sequência de passos de transformação de modelos
- Normalmente, é aplicada uma combinação das diversas estratégias de modelagem
- Quando construímos um modelo de um banco de dados, estamos aprendendo fatos sobre a realidade, a sequência de ideias que se tem durante um processo de aprendizagem, dificilmente, é controlada por uma estratégia



- Basicamente, existem duas fontes de informação a serem usadas durante o processo de modelagem:

**Descrições de dados existentes:** Essa situação ocorre quando deseja-se obter um modelo de dados para um sistema computacional existente ou de documentos como notas fiscais e outros

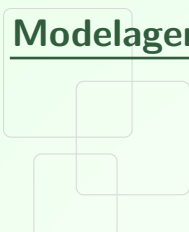
**Conhecimento de pessoas:** Quando um novo sistema está sendo proposto, as informações são obtidas com as pessoas que possuem o conhecimento para o a construção do novo sistema





- A modelagem partindo de descrições de dados existentes utiliza a *engenharia reversa* para obter um modelo a partir de um sistema já existente
- Nessa situação, é usada a estratégia **ascendente** (*botton-up*)
- Essa estratégia parte de conceitos mais detalhados e abstrai gradativamente
- A modelagem inicia com a identificação dos atributos que, por sua vez, são agregados em entidades
- As entidades são relacionadas e generalizadas



- 
- Para a modelagem de novos bancos de dados utilizando o conhecimento de pessoas sobre o sistema, existem duas estratégias:
    - **Descendente (*top-down*)**
    - ***Inside-out***

# Estratégia top-down

---



- A estratégia *top-down* parte de conceitos mais abstratos e, gradativamente, refina estes acrescentando mais detalhes
- O processo de modelagem inicia com a identificação de entidades genéricas
- A partir daí, são definidos seus relacionamentos e especializações
- Por último, são definidos os atributos das entidades e dos relacionamentos

# Estratégia Top-Down

- 1 Enumeração das entidades
- 2 Identificação dos relacionamentos e hierarquias
- 3 Atribuição das cardinalidades máximas aos relacionamentos
- 4 Determinação dos atributos das entidades e relacionamentos
- 5 Determinação dos atributos identificadores
- 6 Verificação do banco de dados quanto ao aspecto temporal
- 7 Identificação de entidades fracas
- 8 Atribuição das restrições de participação
- 9 Procura-se construções redundantes
- 10 Validação do com modelo com pessoas envolvidas

- Em qualquer desses passos, é possível retornar aos passos anteriores
- Por exemplo, durante a identificação de atributos é possível que sejam identificadas novas entidades, fazendo com que o processo retorne ao primeiro passo

- A estratégia *inside-out* consiste em partir de conceitos considerados mais importantes (centrais) e ir gradativamente adicionando conceitos periféricos relacionados
- O processo inicia com a identificação de uma entidade particularmente importante que, supostamente, está relacionada a muitas outras entidades
- Em seguida, são procurados atributos, entidades relacionadas, generalizações e especializações da entidade em foco, e assim, recursivamente, até obter-se o modelo completo
- A denominação da estratégia provém da ideia de que entidades mais importantes em um modelo e com mais relacionamentos são desenhadas no centro do DER, a fim de evitar o cruzamento de linhas

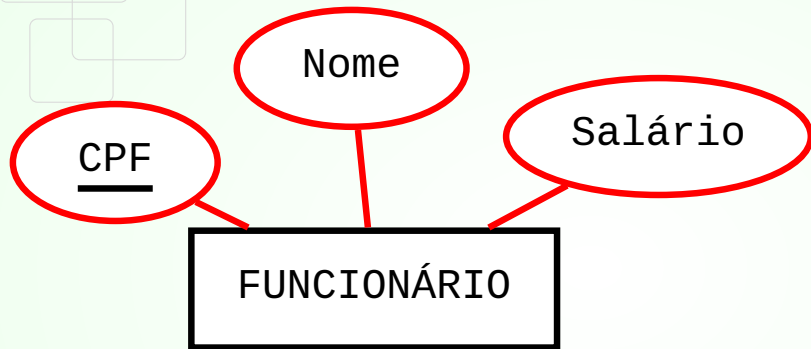
# Estratégia inside-out - exemplo de folha de pagamento



**FUNCIONÁRIO**

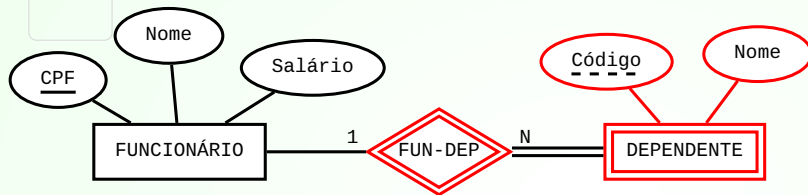
Como se trata de uma folha de pagamento de funcionários, podemos considerar que a entidade mais importante é **FUNCIONÁRIO**

## Estratégia inside-out - exemplo de folha de pagamento



Podemos colocar os atributos de funcionários

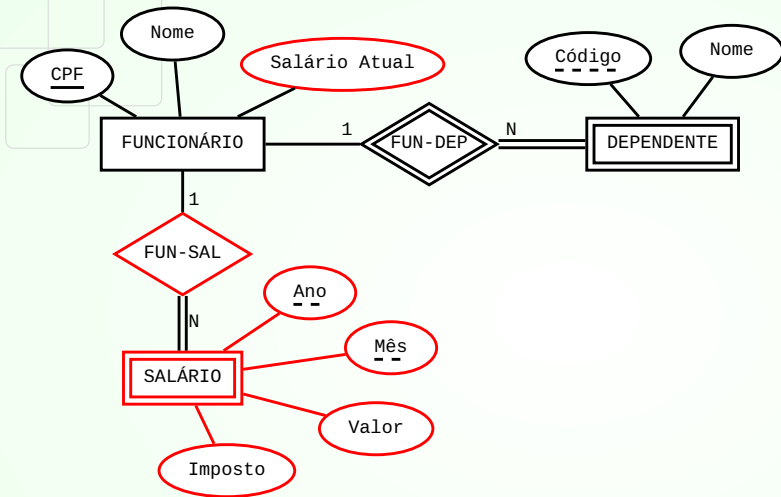
# Estratégia inside-out - exemplo de folha de pagamento



Como o cálculo do salário dos funcionários envolve seus dependentes, devemos criar essa entidade e relacioná-la com **FUNCIONÁRIO**



# Estratégia inside-out - exemplo de folha de pagamento



Em seguida, observamos que é preciso guardar o histórico de salários dos funcionários, então criamos uma entidade para isso e ligamos a funcionário



ELMASRI, R.; NAVATHE, S. B. **Sistemas de banco de dados**. 6. ed. São Paulo: Pearson Addison Wesley, 2011.

HEUSER, C. A. **Projeto de banco de dados**. 6. ed. Porto Alegre: Bookman, 2009.

RAMAKRISHNAN, R.; GEHRKE, J. **Sistemas de gerenciamento de banco de dados**. 3. ed. São Paulo: McGrawHill, 2008.