

UFSC-CTC-INE

Curso de Sistemas de Informação

INE 5600 – Bancos de Dados III

BDOO:

**Modelo de Dados e
Manipulação de Dados**

BD00: Modelo de Dados

BD Orientado a Objetos

- **BDOO**: paradigma OO + SGBD
 - orientação a objetos
 - encapsulamento: objeto “encapsula” uma *estrutura* (atributos) e um *comportamento* (métodos)
 - vantagem: cada aplicação com sua interface (“visão”) particular dos dados
 - reusabilidade: novos objetos podem aproveitar propriedades já definidas em outros objetos
 - vantagem: redefinições de dados são evitadas
 - SGBD
 - gerenciamento eficiente de dados persistentes
 - acesso otimizado e concorrente; segurança; integridade; ...
- **SGBDOO**: gerenciamento de objetos persistentes

Modelo de Dados OO

- BDR
 - modelo formalmente definido e com um conjunto fixo de conceitos
- BDOO
 - falta de consenso sobre um padrão (conjunto de conceitos)
- SGBDOOs com modelos heterogêneos
 - carência de uma base formal
 - início das pesquisas em BDOO
 - muita atividade experimental, voltada às necessidades das aplicações
 - tentativa de padronização: ODMG

Modelo de Dados OO - Conceitos

1. Identidade de objeto (OID)
2. Métodos
3. Classes
4. Estruturas complexas
5. Herança

Modelo de Dados OO - Conceitos

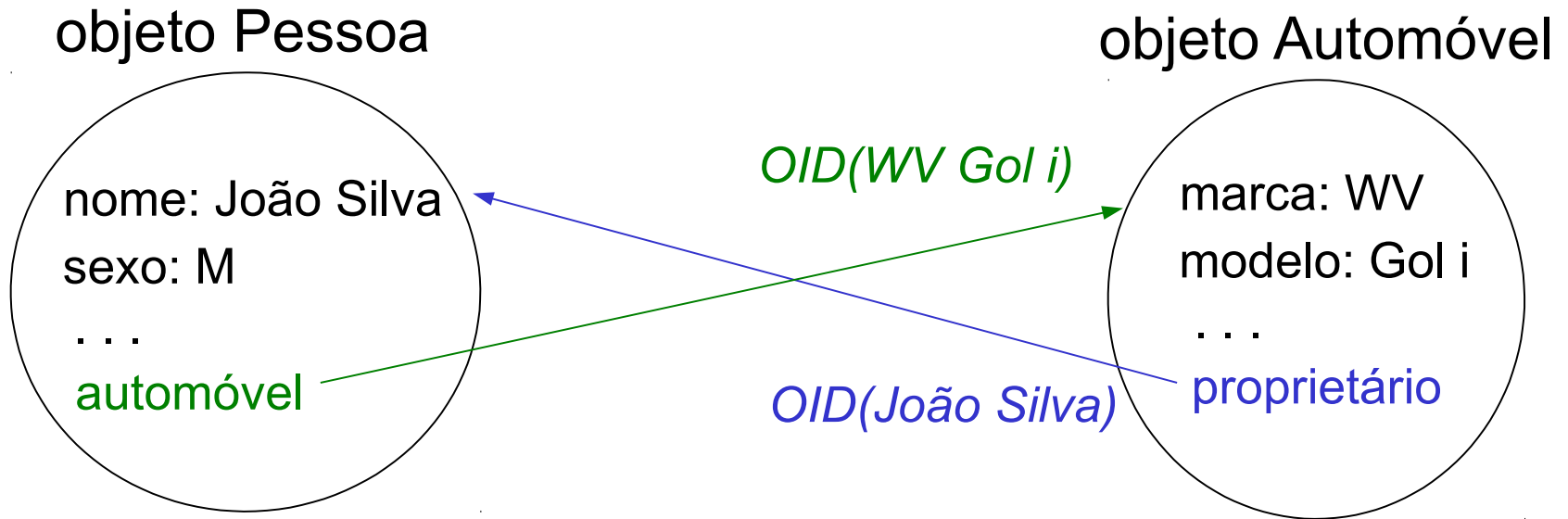
1. Identidade de objeto (OID)
2. Métodos
3. Classes
4. Estruturas complexas
5. Herança

Identidade do Objeto (OID)

- Identificador **único** de cada objeto
 - gerado pelo SGBDOO e imutável
 - visível ou não para o usuário
- Diferenças com relação a BDR
 - chave primária é passível de alteração
 - consistência de unicidade
 - consistência de integridade referencial
 - chave primária em muitos casos é um atributo artificial e visível ao usuário
 - atributo adicional sem muita semântica

Relacionamentos entre Objetos

- Referências a OIDs



- Vantagem

- tipo do dado para referências é uniforme (OID)
 - evita consistência de tipo de dado

Igualdade de Identidade

- Introduz dois tipos de comparação
 - igualdade de identidade ($=$) (para *OIDs*)
 - igualdade de valor ($= =$) (para *atributos*)



- $o1 = = o2$ (verdadeiro!)
- $o1 = o2$ (?)

- Observação
 - *OID* não dispensa (não substitui) a definição de um identificador visível para o usuário

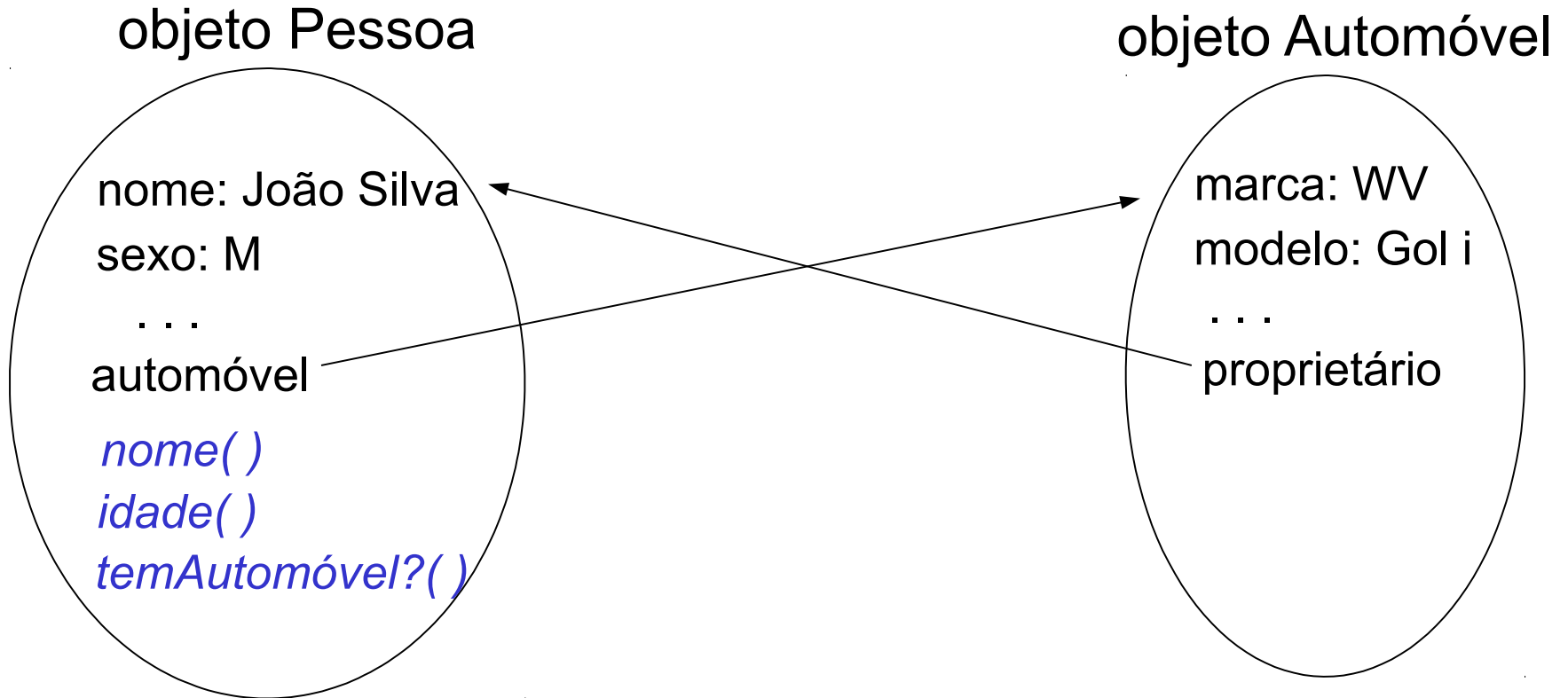
Modelo de Dados OO - Conceitos

1. Identidade de objeto (OID)
2. Métodos
3. Classes
4. Estruturas complexas
5. Herança

Métodos

- Operações associadas a um objeto
 - comportamento de um objeto é mantido no BD
 - BDR mantém apenas a estrutura dos dados
- Composição
 - assinatura (*interface pública do objeto*)
 - implementação (*LPOO utilizada pelo BDOO*)
- Vantagem: encapsulamento de comportamento
 - simplifica o código das aplicações
 - cada aplicação acessa uma interface particular
 - autorizações de acesso e/ou visões podem ser aplicadas em nível de métodos
 - métodos podem servir para programação de RIs
- BDOOs não possuem, em geral, recursos para definição de RIs, como *checks* e *triggers*

Métodos

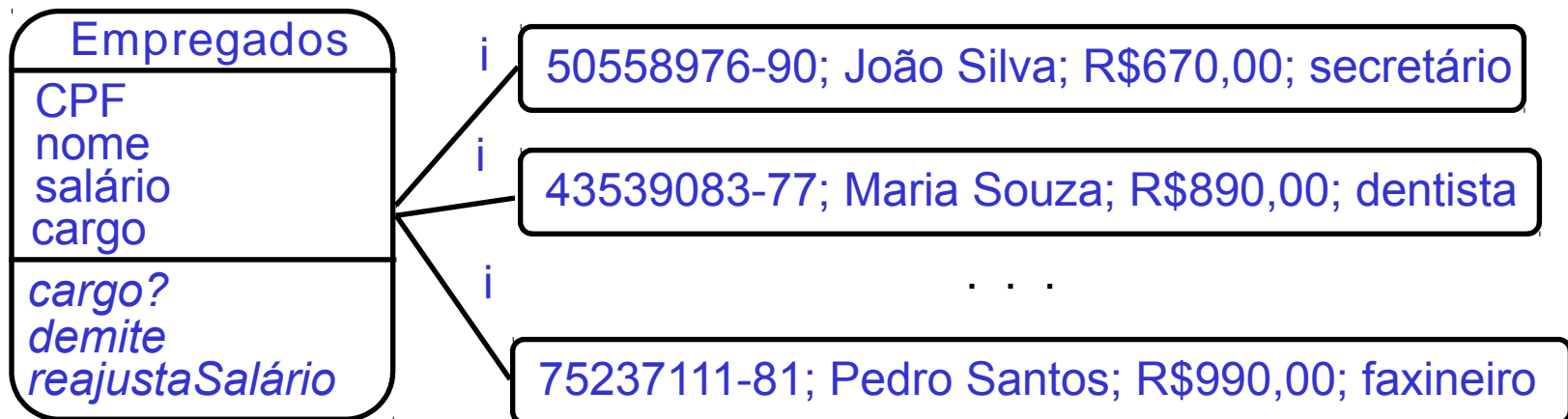


Modelo de Dados OO - Conceitos

1. Identidade de objeto (OID)
2. Métodos
3. Classes
4. Estruturas complexas
5. Herança

Classe

- Conjunto de objetos (instâncias) com a mesma estrutura e comportamento
- Base para a formulação de operações DML
 - função idêntica a de uma tabela em um BDR
- Vantagem: reusabilidade
 - instâncias de uma classe compartilham a mesma estrutura e implementação de métodos



Tipo X Classe

- Ambos os conceitos podem estar presentes em SGBDOOs
- Principal consenso
 - Tipo
 - definição de uma estrutura e/ou assinatura de métodos
 - não possui uma extensão (instâncias)
 - utilizado na definição de uma ou mais classes
 - Classe
 - definição de um conjunto de instâncias (extensão)
 - base para consultas ao BD
 - pode ou não ser definida a partir de um tipo
 - implementa os métodos do tipo, caso ela tenha sido definida a partir de um tipo

Tipo X Classe - Exemplos

```
tipo Pessoa (  
    nome string; sexo char; ...  
    método idade() retorna inteiro;  
    ...)
```

```
classe Empregado tipo Pessoa (  
    método idade() retorna inteiro  
    begin . . . end;  
    ...)
```

```
classe Estudante tipo Pessoa (. . .)
```

```
classe Automóvel (marca: string; modelo: string; ...)
```


Modelo de Dados OO - Conceitos

1. Identidade de objeto (OID)
2. Métodos
3. Classes
4. Estruturas "completas"
5. Herança

Estruturas Complexas

- Atributos com domínios não-atômicos
 - característica não suportada por BDRs
- Tipos de domínios
 - primitivos (atômicos)
 - inteiros, cadeias de caracteres, datas, ...
 - referência (OIDs)
 - nomes de classes (determinam relacionamentos)
 - construídos a partir de construtores de tipos
 - definição de domínios complexos pelo usuário
- Vantagem
 - flexibilidade na definição de objetos complexos

Construtores de Tipos

- **Tupla** (*tuple*)
 - domínio é um registro
- **Conjunto/Coleção** (*set / bag*)
 - domínio é um agrupamento de dados
- **Lista** (*list*)
 - domínio é um agrupamento ordenado de dados
- Exemplos de domínios complexos
 - conjunto de inteiros
 - tuplas de listas de *strings*
 - listas de conjuntos de tuplas
 - ...

Exemplo de Classe

Classe Empregados (

CPF: *integer*,

nome: *string*,

endereço: **TUPLE** (rua: *string*,
número: *integer*,
cidade: *Cidades*),

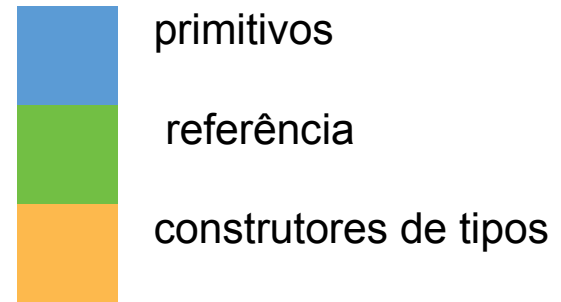
especializações: **LIST**(*string*), (*por ordem de experiência*)

cargo: *string*;

departamento: *Departamentos*,

salário: *real*,

atividades: **SET** (**TUPLE** (projeto: *Projetos*,
tarefa: *string*)));



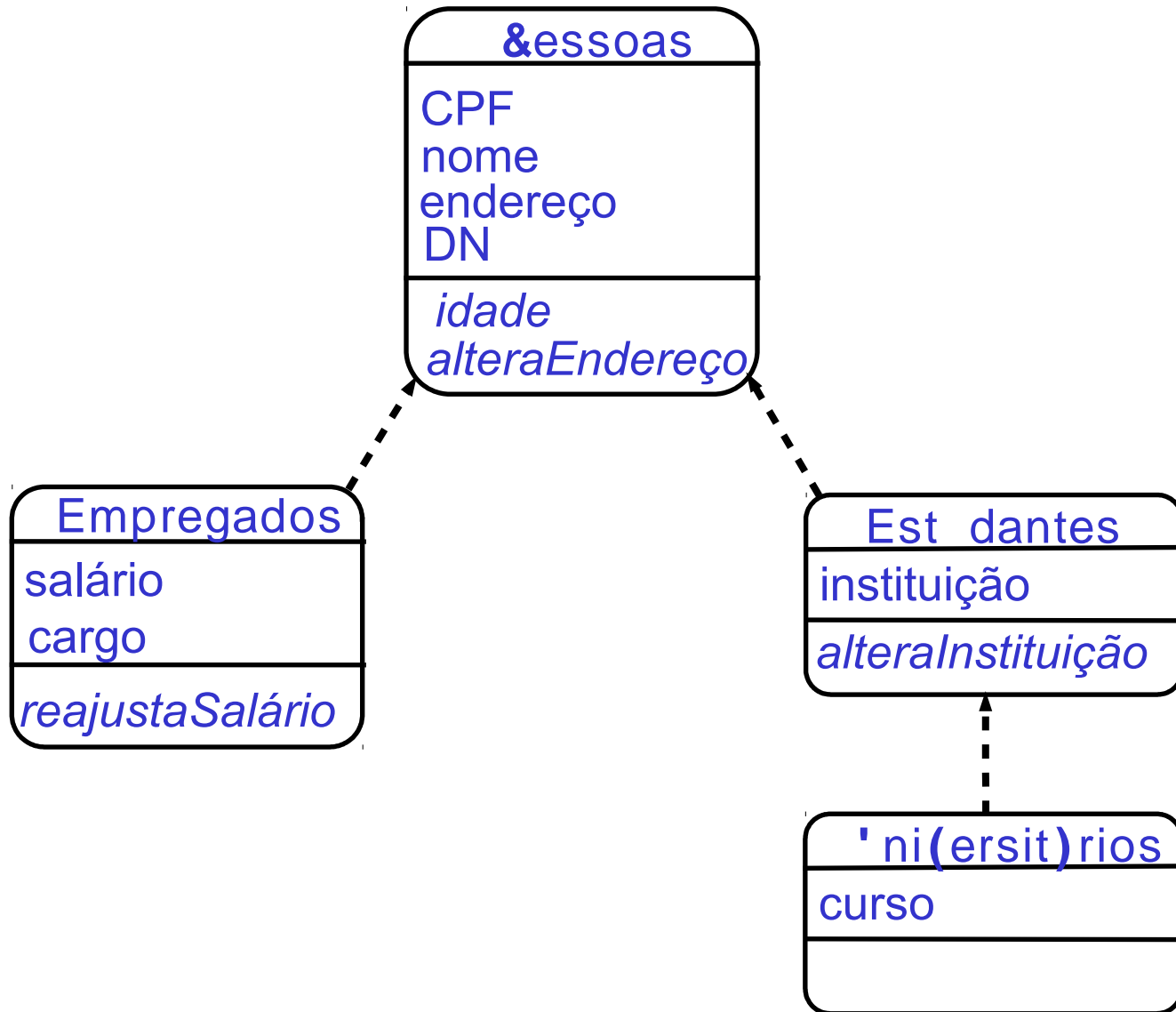
Modelo de Dados OO - Conceitos

1. Identidade de objeto (OID)
2. Métodos
3. Classes
4. Estruturas complexas
- #. \$eran%a

Herança

- Suporte à representação de relacionamentos com semântica de generalização e especialização
 - especialização
 - classe (subclasse) herda propriedades de outra classe (superclasse) e define novas propriedades
 - subclasse: categorização da superclasse
 - generalização (*É-UM*)
 - propriedades comuns de classes (subclasses) podem ser definidas uma única vez em uma superclasse
- Vantagem: reusabilidade

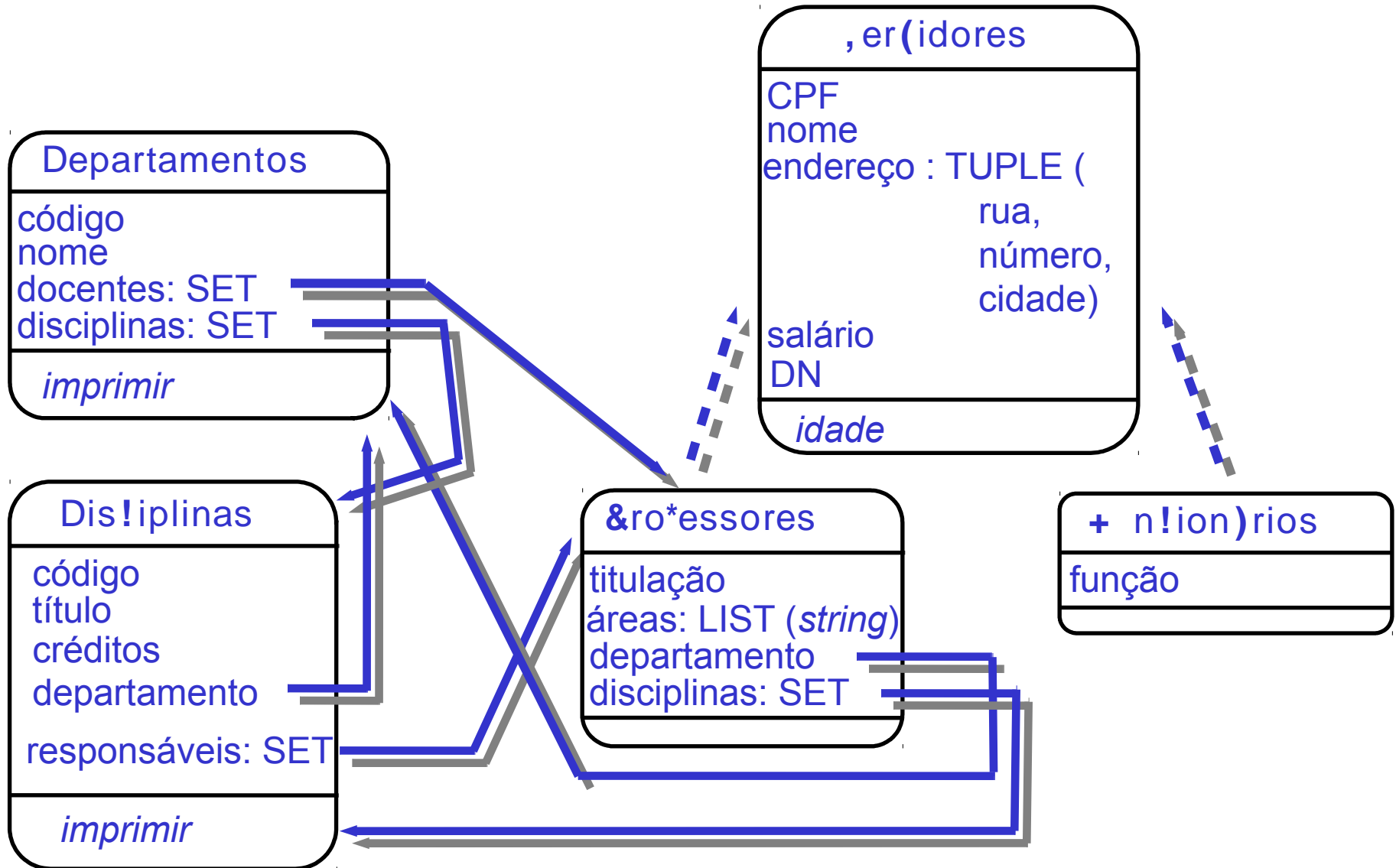
Hierarquia de Herança



Questões sobre Herança

- Redefinição de propriedades (*Overriding*)
 - preocupação: consultas válidas na superclasse
 - alternativas
 - (i) redefinição não é permitida (*herança estrita*); ou
 - (ii) atributos: domínios mais restritos
 - exemplo: A: real (superclasse) e A: inteiro (subclasse)
 - métodos: domínios mais restritos para os parâmetros e para o tipo do resultado

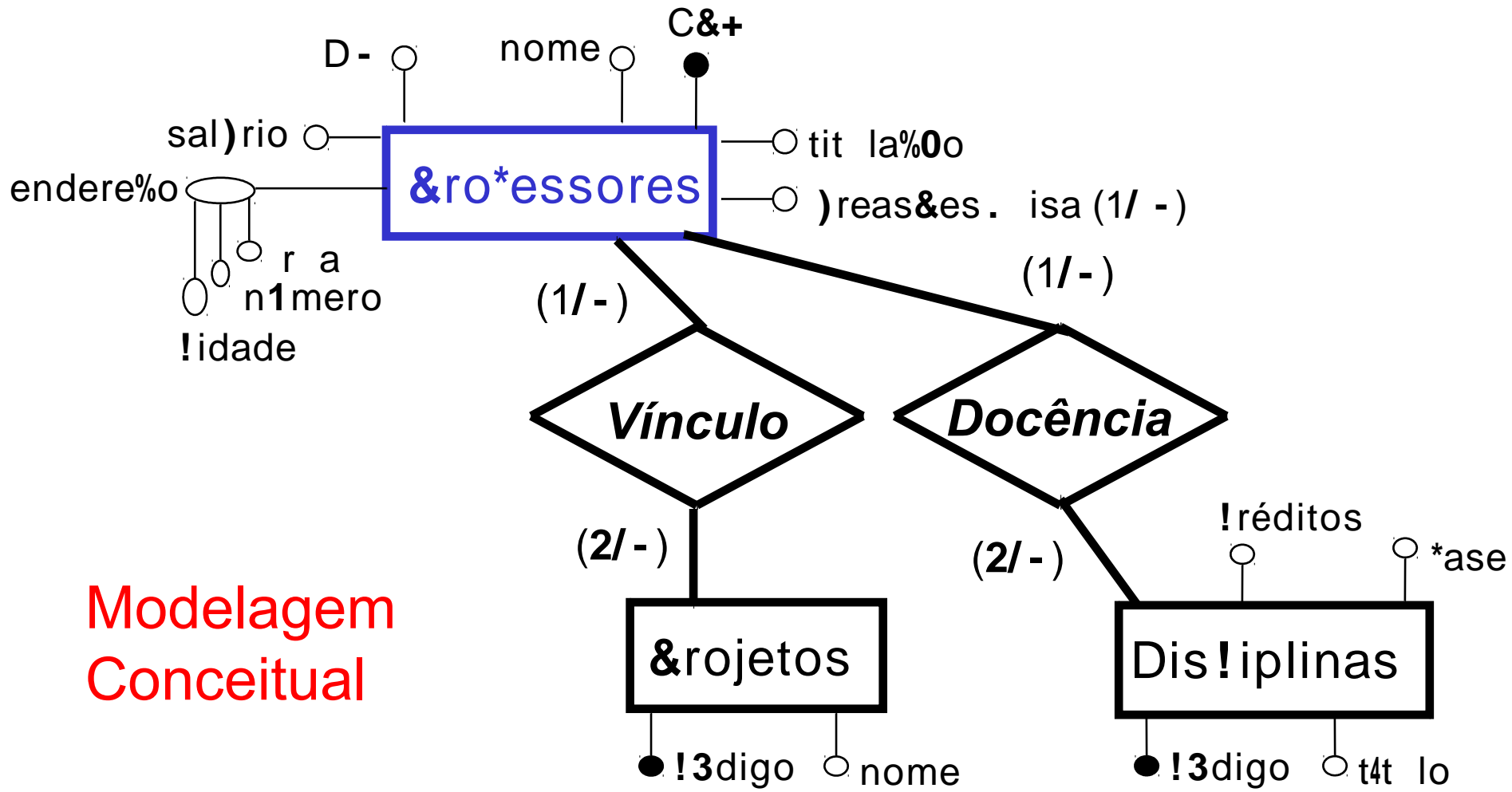
Exemplo de Esquema OO



Modelo de Dados OO

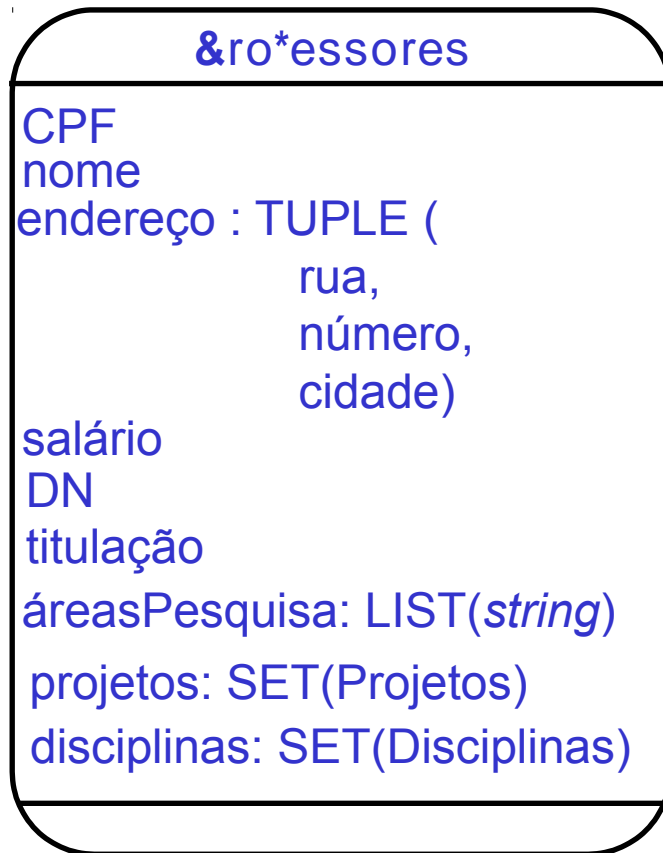
- Comparação com o modelo relacional
 - maior nível de abstração
 - modelo de objetos tem maior afinidade semântica com modelos conceituais de dados
 - representação mais natural de uma entidade do mundo real
 - mais adequado a representação de dados complexos de aplicações não-convencionais
 - aplicações geográficas, das áreas de arquitetura e engenharia
 - modelo mais complexo
 - maior número de conceitos

Exemplo



Modelagem
Conceitual

Exemplo



Modelagem
Lógica OO

Exemplo

Professores

<u>C&+</u>	nome	r a	n1mero	!idade	sal)rio	tit la%0o	D -
----------------	------	-----	--------	--------	---------	-----------	-----

ÁreasPesquisa

<u>C&+</u>	<u>5rea</u>	ordem
----------------	-------------	-------

Docência

<u>C&+</u>	<u>Codd</u>
----------------	-------------

Pesquisa

<u>C&+</u>	<u>Codp</u>
----------------	-------------

Modelagem
Lógica Relacional

BDOO: Padrão ODMG e Manipulação de Dados

ODMG - *Object Database Management Group*

- (Tentativa de) Padronização para modelo de dados e acesso para SGBDOO
- Consórcio de pesquisadores e fabricantes
- Componentes principais do padrão
 - modelo de objetos
 - linguagem de definição de dados (ODL)
 - linguagem de consulta (OQL)

Definição de Classe em ODL

```
class Departamento {  
  attribute string nome;  
  attribute short código;  
  attribute struct Endereço{  
    string rua, short número,  
    string cidade} localização;  
  attribute struct atendimento{  
    time horaInício, time hora !rmino}  
  hor"rio;  
  attribute Empregado che#e;  
  relationship set$Empregado% empregados  
    in&erse Empregado'' depto;  
  &oid adicionaEmp(short )*+ raises  
  (,ah rabalha, )*Ine-istente+;  
  . };
```



atributo atômico



atributo estruturado

atributo de referência a objeto

relacionamento

Herança em ODL

```
/lasse Empregado {  
    attribute short );*;  
    attribute string nome;  
    attribute enum g0nero{1,2} se-o;  
    attribute Date D3;  
    attribute #loat sal"rio;  
    relationship Departamento depto  
        in&erse Departamento' ' empregados;  
    444  
};
```

```
/lasse 5ro#essor  
    e-tends Empregado {  
        attribute string titulaçãõ;  
        attribute string area6tuaçãõ;  
        444  
};
```

OID e Chave

- **OID**

- identificador do objeto

- **Chave**

- uma ou mais propriedades cujos valores devem ser únicos

```
class Departamentos
(7e8 código+
{
attribute string nome;
attribute short código;
444
};
```

```
class /idades
(7e8 (estado,nome++ {
attribute string estado;
attribute string nome;
444
}
```

OQL

- Linguagem de consulta declarativa
 - violação de encapsulamento
 - maior flexibilidade para formulação de consultas
- Não há suporte para operações de atualização de dados (I,A,E)
 - métodos devem ser implementados
- Extensão da linguagem SQL com suporte ao tratamento de
 - objetos complexos
 - junções por valor ou por OID
 - invocação de métodos
 - buscas em hierarquias de herança

Consultas e Resultados

- Ponto de partida de uma consulta
 - extensão de uma classe (*extent*)

```
select e49 ← variável de iteração  
#from e in Empregados
```

- Resultados de consultas
 - atributos de objetos e/ou novas estruturas

```
select struct (  
    nome'          d4nome  
    emps)icos'    (select e49  
                    #from e in d4empregados  
                    :here e4sal"rio % ;<<<++  
#from d in Departamentos
```

Expressões de Caminho

- Permitem a navegação em estruturas complexas e objetos associados
 - objetos associados
 - atributos de referência e relacionamentos
 - utiliza-se a **notação de ponto** (“.”)
- Exemplo

```
select p4nome, p4titulação  
#rom p in 5ro#essores  
:here p4depto4código = >I3E?
```

Expressões de Caminho

- Variáveis de iteração são definidas para a navegação em coleções de objetos referenciados (referências 1:N)
 - a variável de iteração associa-se com cada elemento da coleção referenciada
- Exemplo

```
select e4nome  
#from d in Departamentos, e in d4empregados  
:here d4código = >I3E?  
and e4sal"rio % ;<<<
```

Junções

- Junções entre conjuntos de objetos são permitidas, como em BDRs
 - junções podem ser por valor ou por OID
- Exemplo

```
select e@4nome
#rom eA in Empregados, e@ in Empregados
:here eA4nome = >5edro /ampos Bouza?
and e@4sal"rio = eA4sal"rio ← junção por valor
and e@4depto = eA4depto ← junção por OID
```

Invocação de Métodos

- Métodos podem ser declarados em consultas da mesma forma que atributos
- Exemplos

```
select e4nome  
#rom e in Empregados  
:here e4idade % ;<
```

```
select d4código, d4nroCoras6tendimento  
#rom d in Departamentos
```


Consultas em Hierarquias de Classes

- Consultas aplicadas a uma classe processam automaticamente objetos da classe e de suas subclasses
- Restrições sobre subclasses alvo podem ser especificadas
- Exemplo

```
select (5ro#essores, 5esDuisadores+e4nome  
#rom e in Empregados  
:here e4sal"rio % E<<<
```