BD Objeto-Relacional - Motivação

- SGBDs Relacionais (SGBDRs)
  - sistemas já consolidados no mercado
  - boa performance
    - muitos anos de pesquisa e aprimoramento
    - eficiência: otimização de consultas, gerenciamento de transações
  - não atendem adequadamente os requisitos de dados de novas categorias de aplicações

BD Objeto-Relacional - Motivação

- SGBDs Orientado a Objetos (SGBDOO)
  - modelo de dados mais rico
    - adequado ao mercado de aplicações não-convencionais
  - pior desempenho, se comparado com SGBDR
  - heterogeneidade a nível de modelo e de capacidades de consulta e atualização
- SGBDs Objeto-Relacional (SGBDOR)
  - combina as melhores características do modelo de objetos no modelo relacional
    - modelo rico + eficiência no gerenciamento de dados
  - tecnologia presente em alguns SGBDRs
    - exemplos: Oracle, Informix, DB2, Postgres

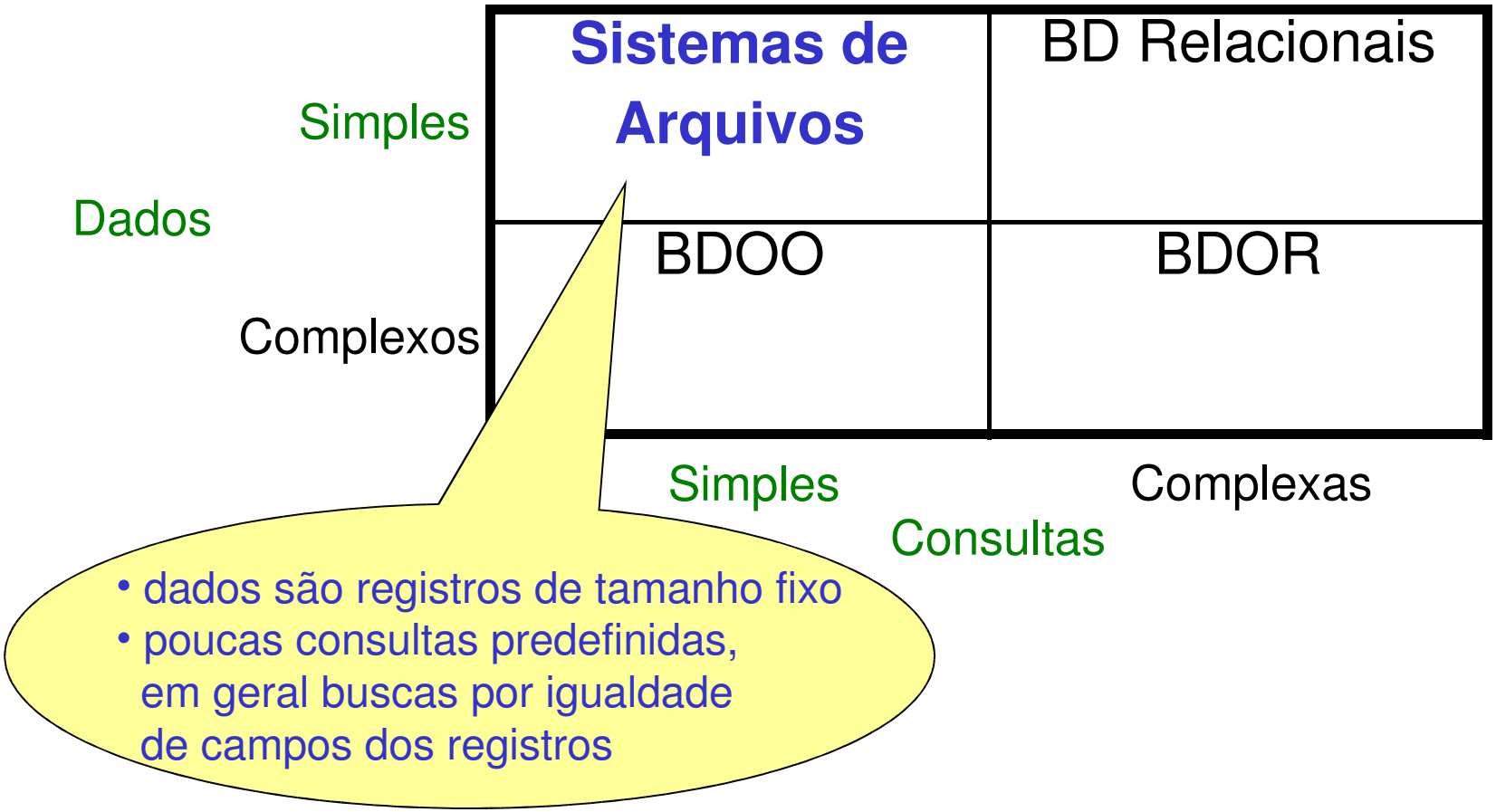
Classificação de *Stonebreaker*

- Pai da tecnologia OR (1997)
- Classifica os principais sistemas gerenciadores de dados em 4 quadrantes

Dados \ Consultas	Simples	Complexas
Simples	Sistemas de Arquivos	BD Relacionais
Complexos	BDOO	BDOR

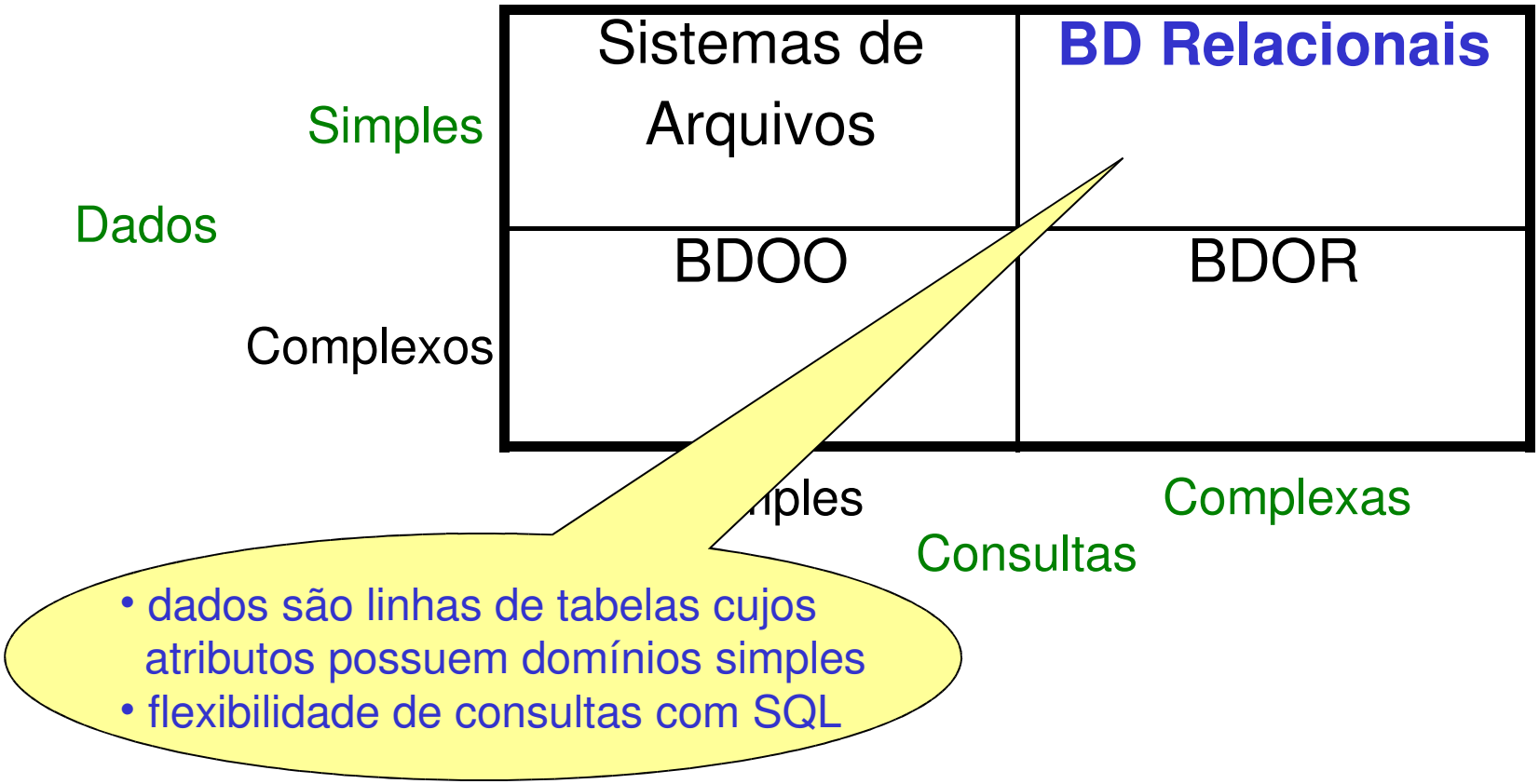
Classificação de *Stonebreaker*

- Quadrantes de tipos de gerenciadores de dados



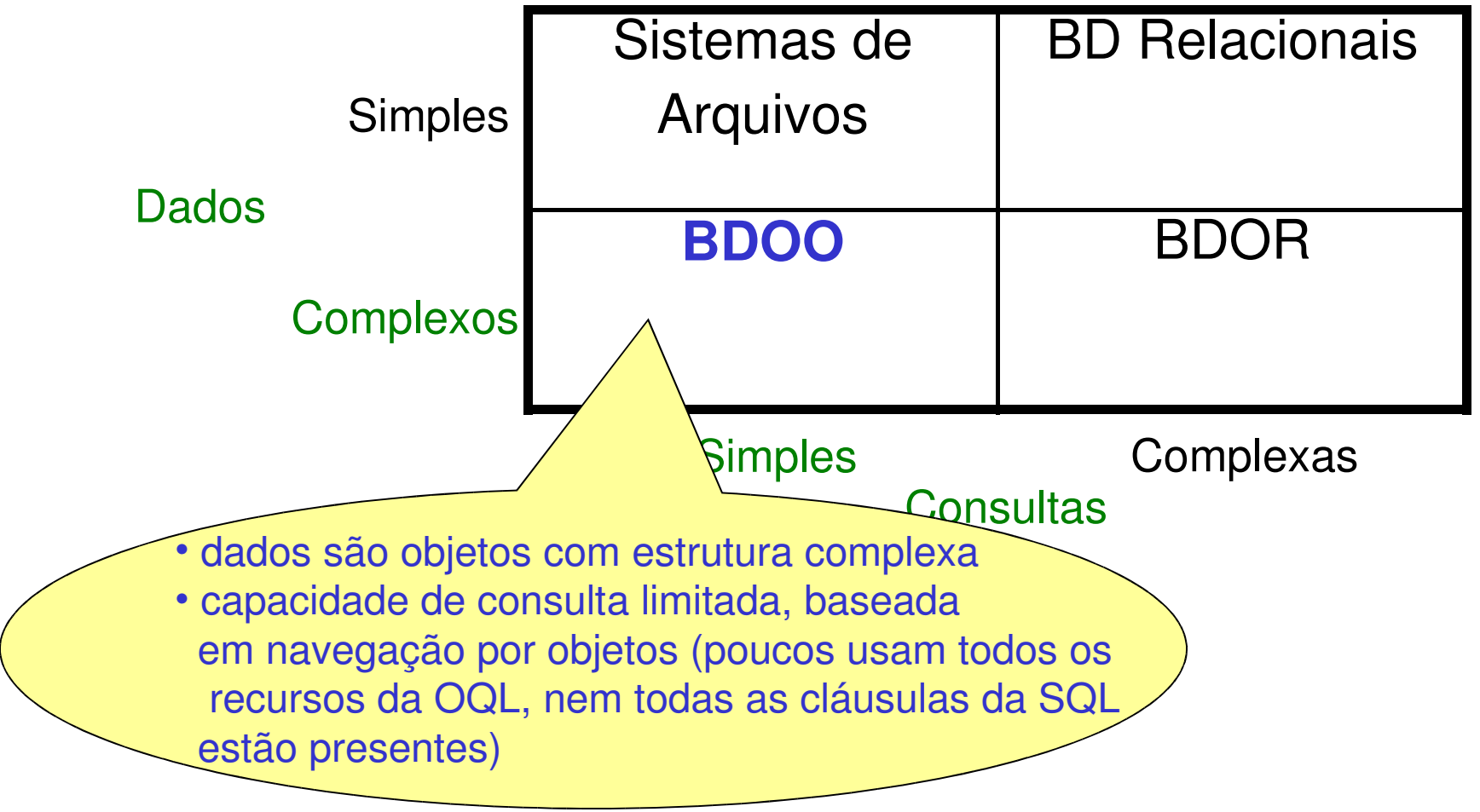
Classificação de *Stonebreaker*

- Quadrantes de tipos de gerenciadores de dados



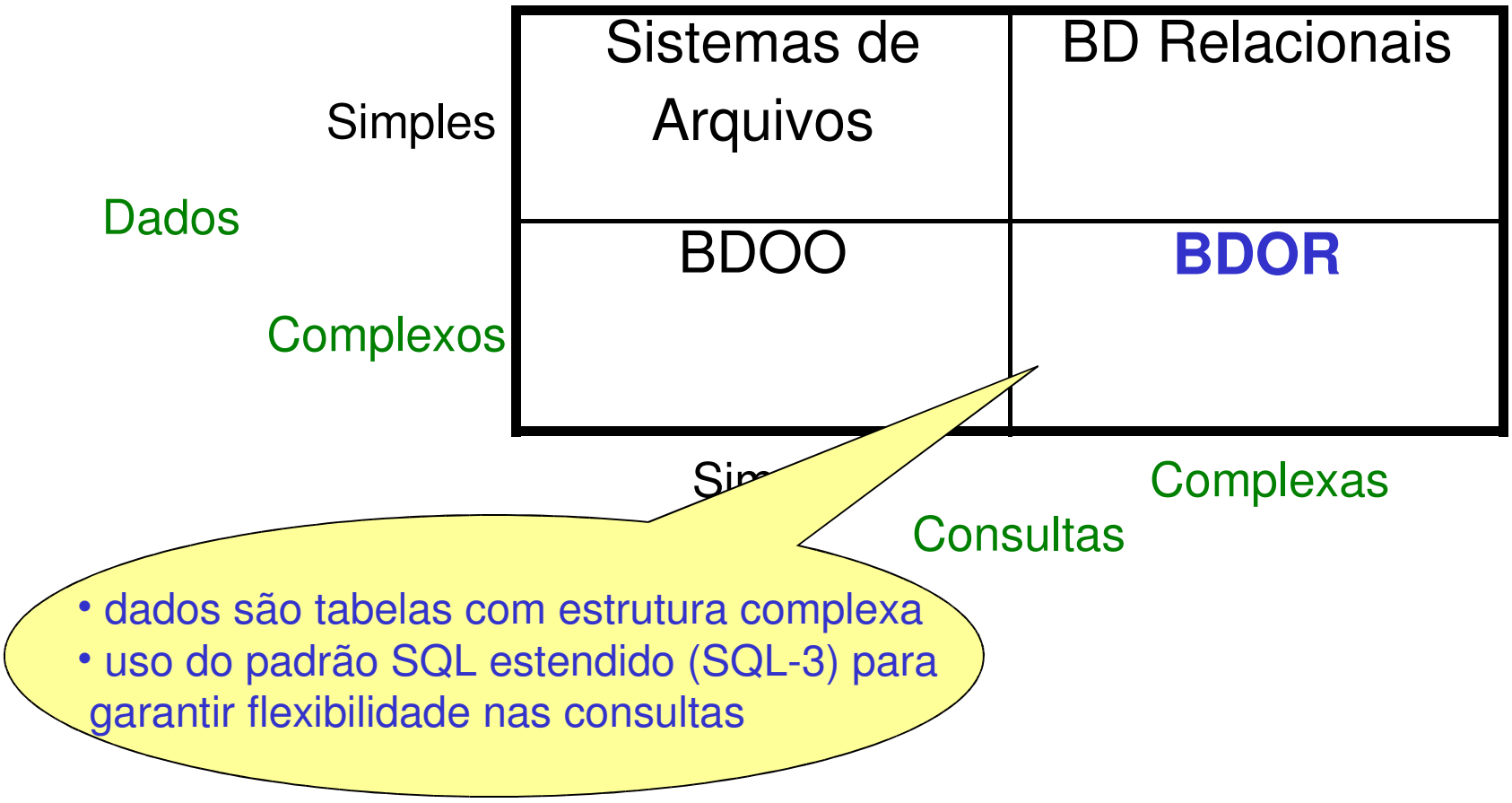
Classificação de *Stonebreaker*

- Quadrantes de tipos de gerenciadores de dados



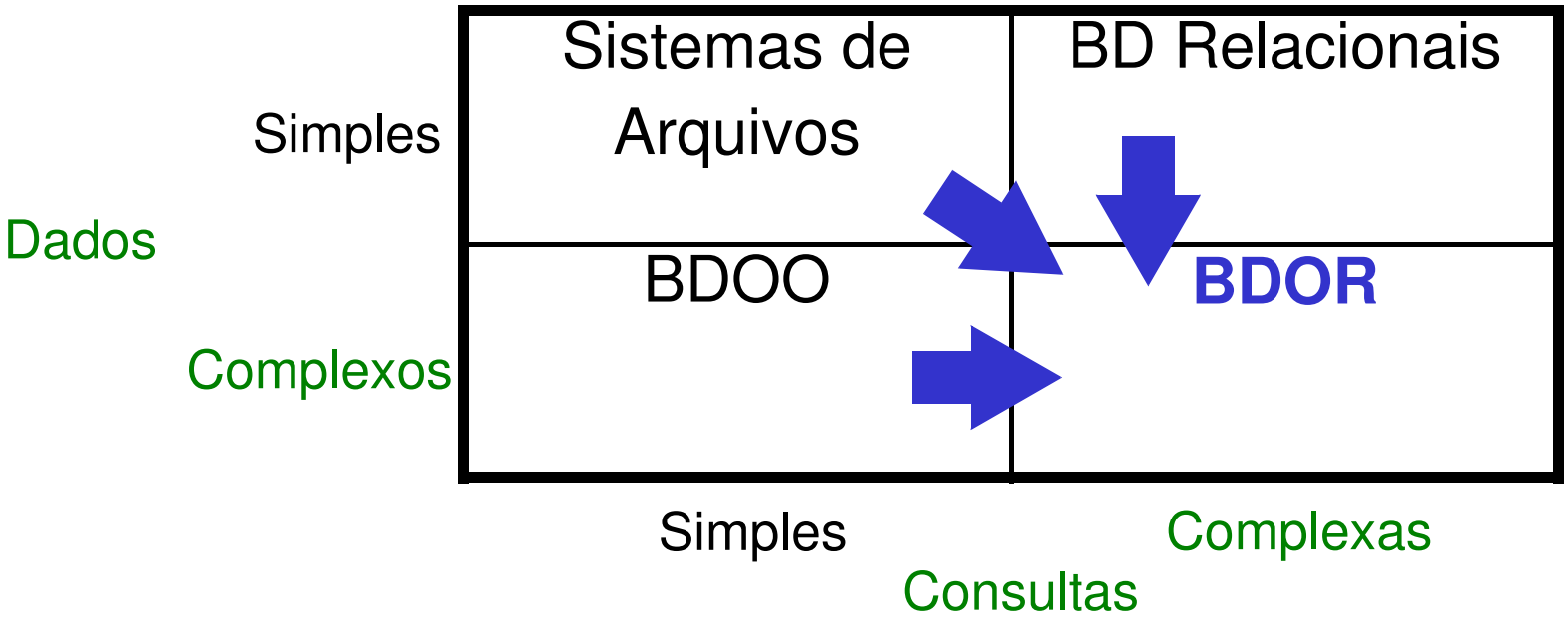
Classificação de *Stonebreaker*

- Quadrantes de tipos de gerenciadores de dados



Classificação de *Stonebreaker*

- Tendência
  - migração para tecnologia OR



BDR x BDOO x BDOR

Critério	BDR	BDOO	BDOR
padrão	SQL-2	ODMG 3.0	SQL-3
suporte a dados complexos	não	sim	sim
performance	alta	baixa	espera-se que seja alta
maturidade	maduro	razoavelmente maduro	razoavelmente novo
uso de SQL	SQL *full*	OQL	SQL estendido para objetos
vantagem	eficiência de acesso	modelo de dados rico	modelo rico + eficiência de acesso
uso comercial	larga escala	pequena escala	tendência: alcançar larga escala

SQL-3 (SQL 99)

- Versão mais atual da SQL
  - por enquanto (SQL-4 em andamento) ...
- Extensão da SQL-2 (SQL 92)
  - tratamento de objetos
  - consultas recursivas
  - instruções de programação
  - ...

SQL-3

- Suporte ao tratamento de objetos
  - tabelas aninhadas (objetos linha)
  - tipos abstratos de dados (TADs)
  - referências e OIDs
  - objetos complexos
  - definição de comportamento
  - herança

Definição de Objetos

- Duas formas
  - tipo objeto linha (*row object*)
    - define uma estrutura de tupla (registro)
    - atributos podem conter outras tuplas
      - permite a definição de uma estrutura aninhada
  - tipo abstrato de dado (TAD)
    - define uma estrutura complexa
    - define comportamento e herança

Objeto Linha

- Definição

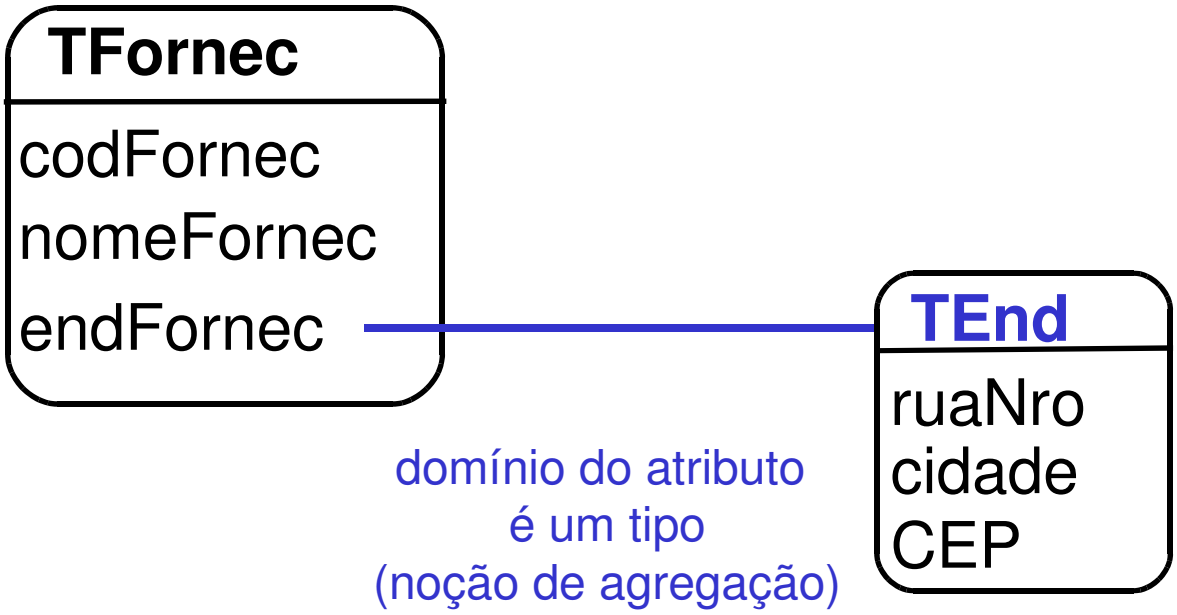
```
CREATE ROW TYPE (<declaração_componentes>)
```

- Exemplos

```
CREATE ROW TYPE TFornec(
   codFornec       CHAR(4),
   nomeFornec           VARCHAR(40),
   endFornec       TEnd );

CREATE ROW TYPE TEnd(
   ruaNro          VARCHAR(60),
   cidade          VARCHAR(40),
   CEP             INTEGER );
```

Modelagem OR – Tipo Objeto Linha

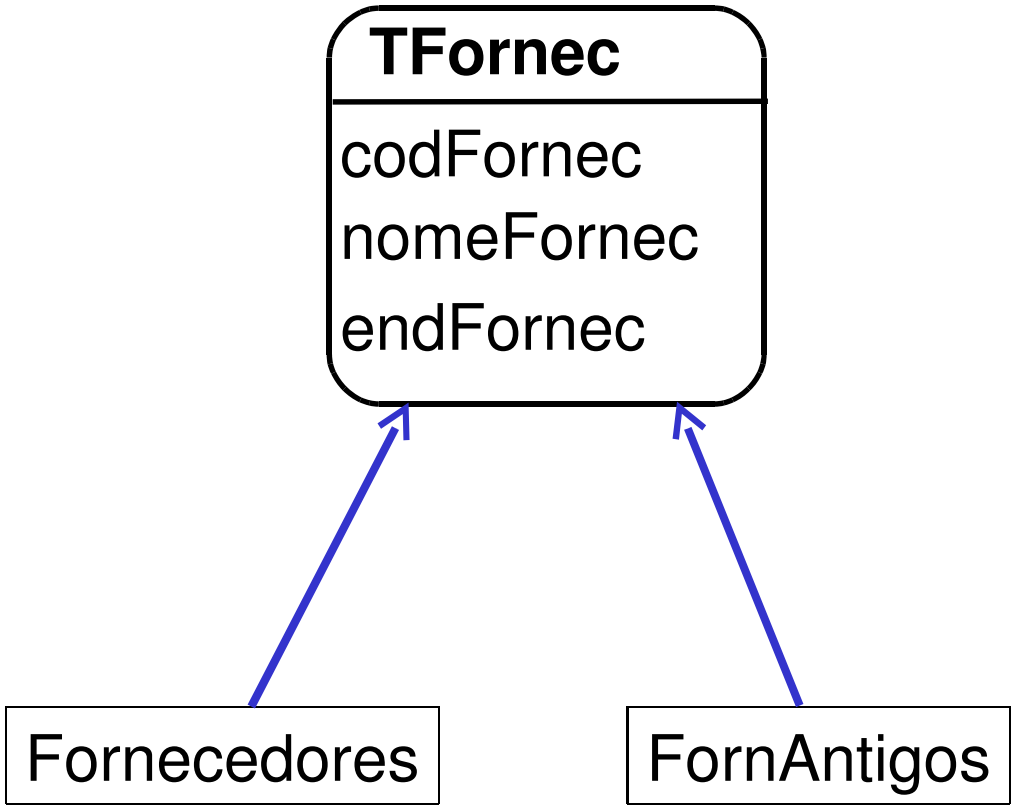


Criação de Tabelas

- Indicação do tipo a que pertence
- Várias tabelas podem ser de um mesmo tipo
- Exemplos

```
CREATE TABLE Fornecedores OF TYPE TFornec;
CREATE TABLE FornAntigos OF TYPE TFornec;
```

Modelagem OR – Tabela Baseada em Tipo



Acesso a Atributos Aninhados

- Notação de ponto (“dois pontos”) para navegação em atributos que fazem parte de uma estrutura aninhada
- Exemplo

```
SELECT codFornec, endFornec..ruaNro
FROM Fornecedores
WHERE endFornec..cidade = 'Florianopolis'
```

Criação de Objetos Linha

- Indicação de valores para todos os níveis de aninhamento
- Exemplo

```
INSERT INTO Fornecedores
VALUES ('F102', 'João Silva', TEnd('rua A,
120', 'Florianópolis', 88000));
```

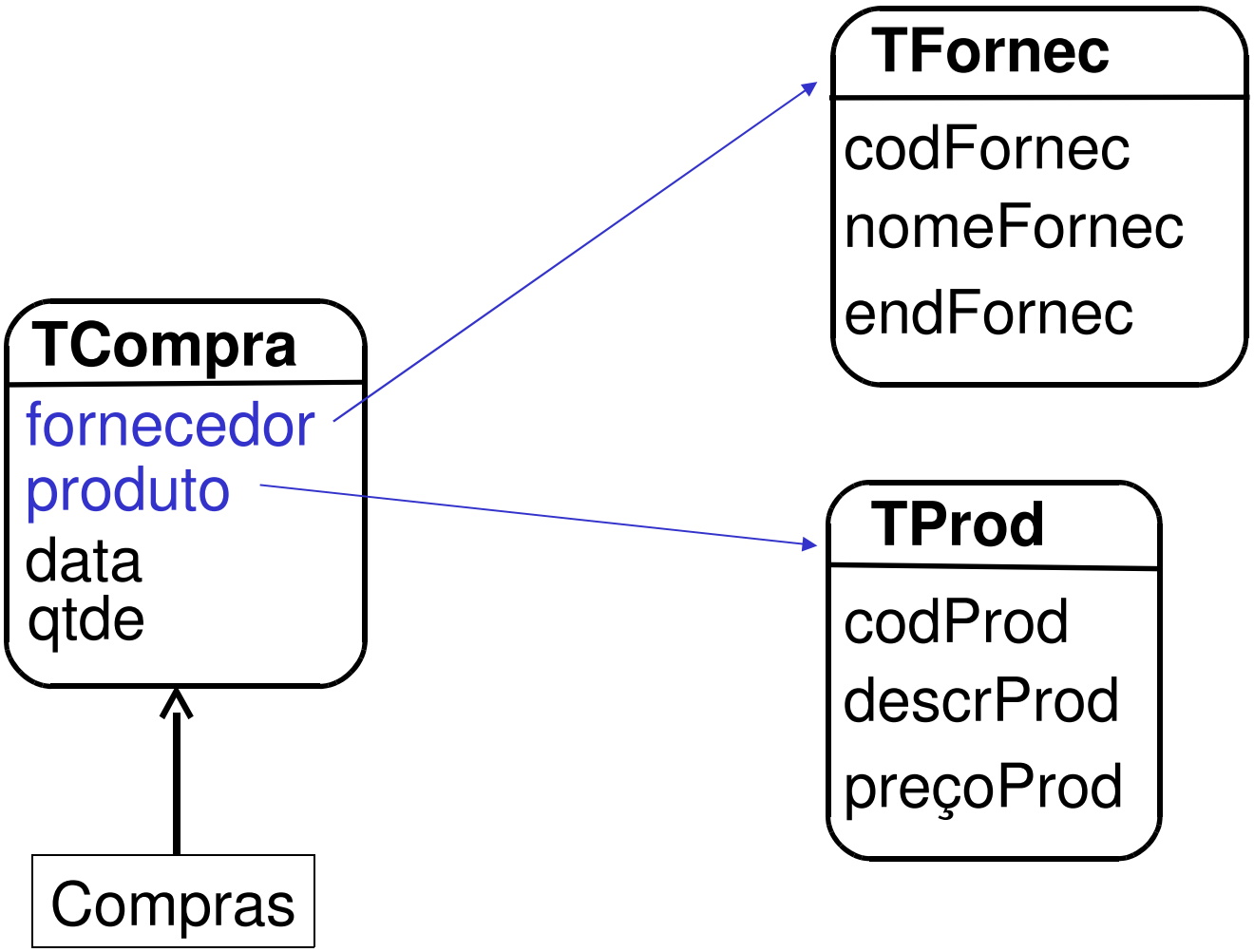
Referência

- Definição de relacionamento entre objetos
- Não é semelhante a uma chave estrangeira
  - chave estrangeira pode ser composta
  - só referencia uma tabela que tenha definido um OID (tabela baseada em um tipo)
- Exemplo

```
CREATE ROW TYPE TCompra(
   fornecedor      REF(TFornec),
   produto         REF(TProd),
   data            DATE,
   qtde            INTEGER);

CREATE TABLE Compras OF TYPE TCompra;
```

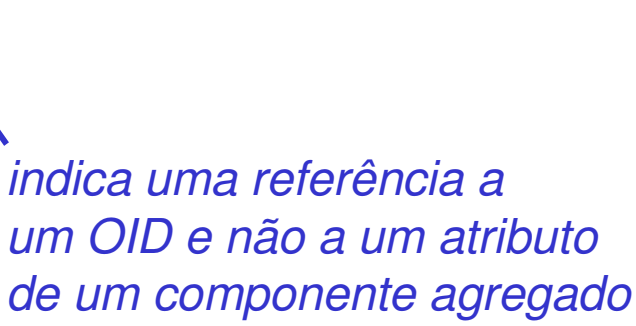
Modelagem OR – Referências



Acesso a Objetos Relacionados

- Exemplo

```
SELECT fornecedor->nomeFornec
FROM Compras
WHERE qtde > 1000
AND produto->codProd = 45;
```

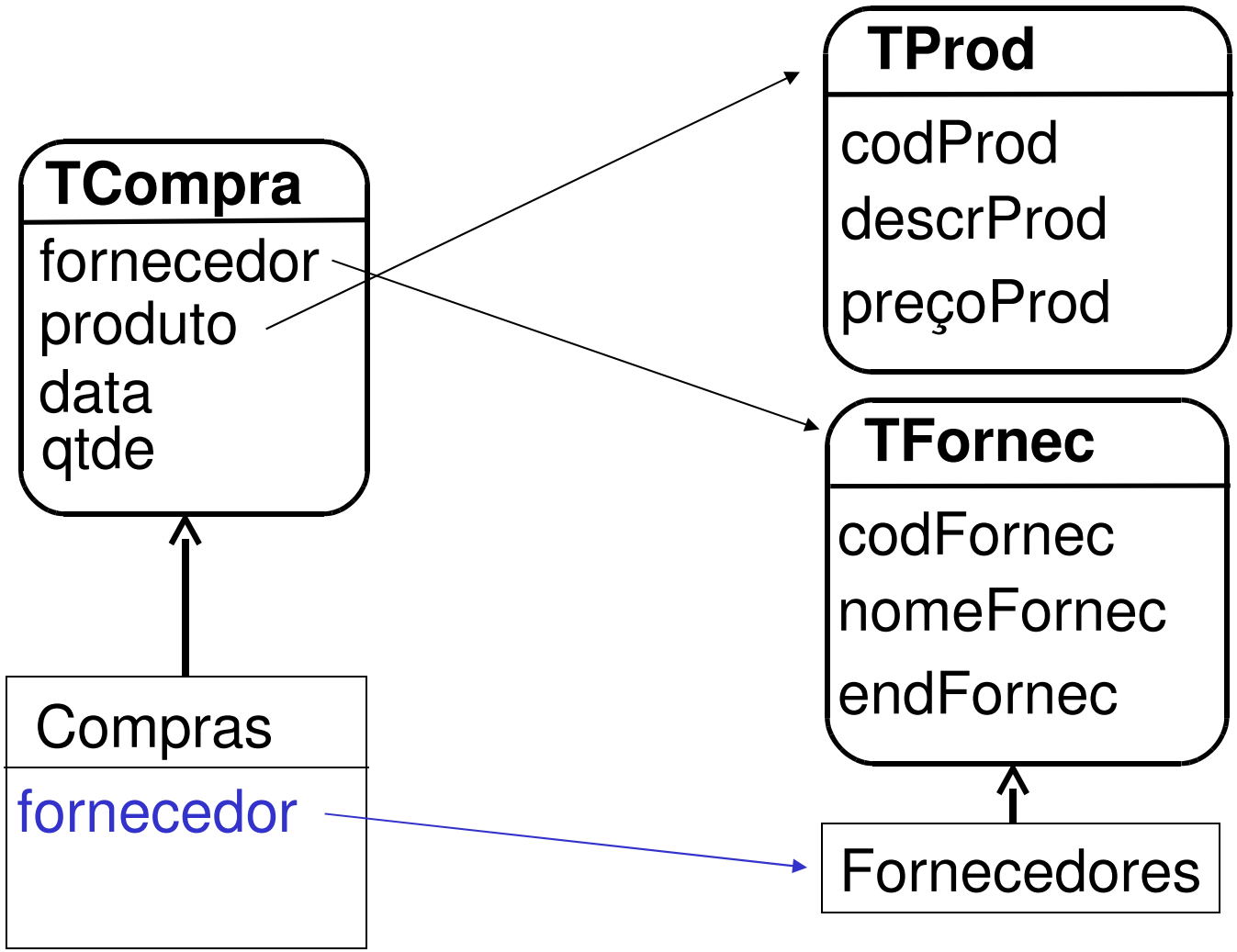


Escopo de Referência

- Uma referência indica um tipo
- Deve-se definir o escopo da referência quando mais de uma tabela pertence ao tipo
  - `SCOPE FOR <nome_atributo> IS <nome_tabela>`
- Exemplo

```
CREATE TABLE Compras OF TYPE TCompra
SCOPE FOR fornecedor IS Fornecedores;
```

Modelagem OR – Escopo de Referência



OID

- Em BDOR, um OID
  - é o valor indicado por atributos de referência
  - pode ser uma chave primária
  - pode ser definido pelo usuário ou pelo sistema
- Exemplos

```
CREATE TABLE Fornecedores OF TYPE TFornec
REF IS codFornec PRIMARY KEY;

CREATE TABLE Produtos OF TYPE TProd
REF IS codProd USER GENERATED;
```

atributo definido pelo usuário, mas seu valor é controlado pelo sistema (OID)

OID gerado e controlado pelo sistema

```
... REF IS SYSTEM GENERATED;
```

Modelagem OR – Identificadores

Fornecedores
codFornec

Produtos
codProd (OID)

X
(OID)

Comparações de OIDs

- Comparações idênticas às comparações de valores de outros tipos de atributos
- Exemplo

```
SELECT Fornecedores.nomeFornec
FROM Fornecedores, Compras, Produtos
WHERE Produtos.tipo = 'Parafuso'
AND Compras.qtde > 1000
AND Produtos.codProd = Compras.produto
AND Fornecedores.codFornec = Compras.fornecedor;
```

Criação de Objetos com Referência

- Indicação dos valores de OIDs
- Exemplo

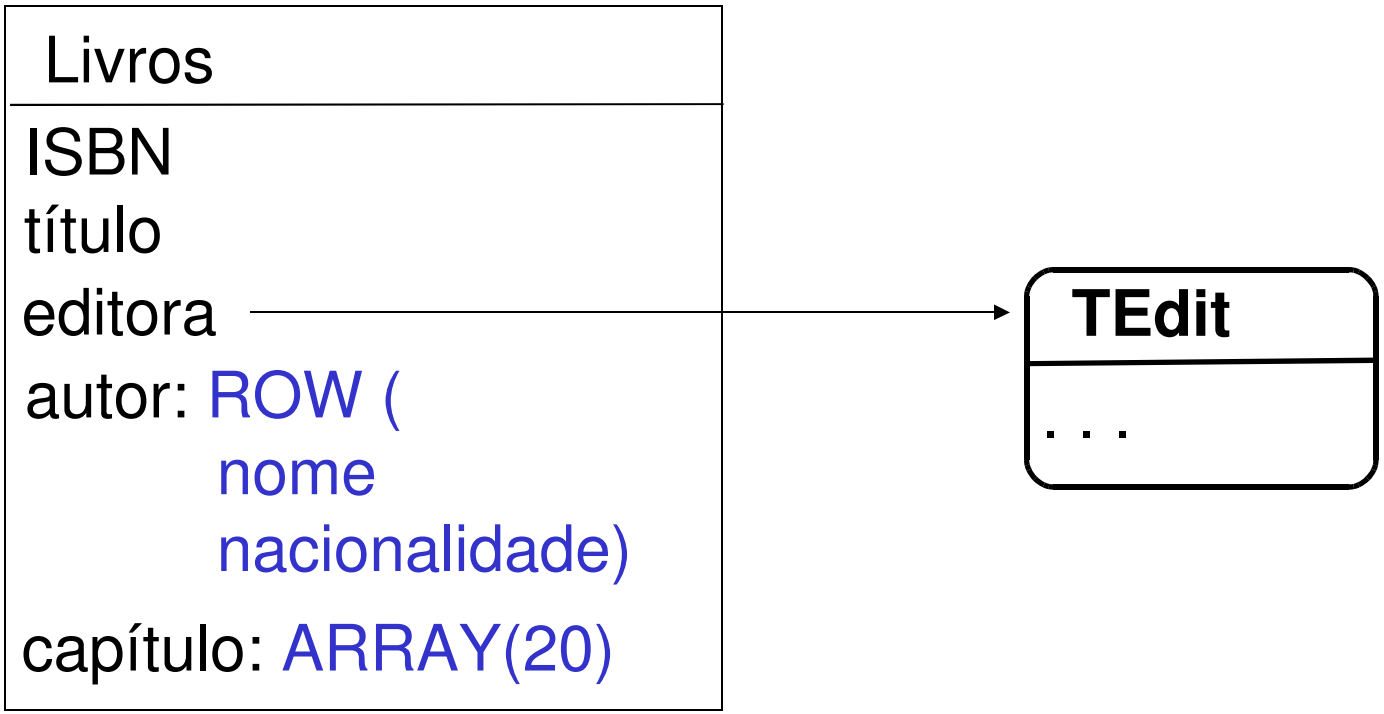
```
INSERT INTO Compras
VALUES (REF('F102'), REF(1002), '10/12/03',
1300);
```

Definição de Objetos Complexos

- Novos tipos de dados
  - *Row* (tupla)
  - *Array* (coleção ordenada)
    - arrays n-dimensionais não são permitidos
- Exemplo

```
CREATE TABLE Livros (
    ISBN        CHAR(10),
    título      VARCHAR(60) NOT NULL,
    editora     REF(TEdit),
    autor       ROW (nome            VARCHAR(50),
                     nacionalidade   VARCHAR(15)),
    capítulo    VARCHAR(20) ARRAY[20]
    );
```

Modelagem OR – Objetos Complexos



Acesso a Objetos Complexos

```
insert into Livros
values ('65893/186-9', 'Banco de Dados Objeto-
  Relacional', REF('Campus'), ('João Souza',
  'brasileira'), ARRAY['Introdução', 'OO', 'BD
  Objeto-Relacional', 'Conclusão']);

select capitulos[1]
from Livros
where autor..nome = 'João Souza'
```

Objetos Complexos

- Alguns SGBDORs suportam outros tipos coleção
  - Informix
    - *List*, *Set* e *Multiset* (coleção)
  - Oracle
    - VARRAY (*array* variável cujos elementos podem ser objetos)
    - NESTED TABLE (atributo cujo domínio é uma tabela aninhada)

Objetos Complexos - LOBs

- LOB - *Large OBject*
  - objeto de tamanho grande
  - podem ser armazenados diretamente no BDOR como atributos de tabelas
    - são agora considerados parte do esquema do BD e não precisam ser mantidos e tratados em arquivos separados
- Tipos de LOBs
  - CLOB (*Character LOB* – texto longo)
  - BLOB (Binary LOB – imagem)
- Definidos em termos de KB, MB e GB

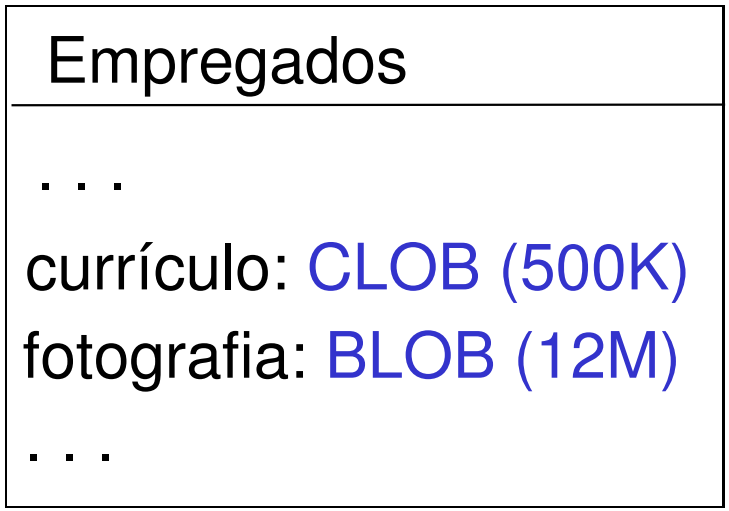
```
CREATE TABLE empregados (...
      currículo          CLOB(500K),
      fotografia         BLOB(12M),
...)
```

LOBs

- Operações
  - CONCATENATION, SUBSTRING, POSITION, OVERLAY, predicados LIKE, ..., funções definidas pelo usuário
- Exemplo

```
SELECT nome
FROM empregados
WHERE currículo LIKE '*UFSC*'
```

Modelagem OR – LOBs



Tipo Abstrato de Dados (TAD)

- Define comportamento para os objetos
  - encapsulamento de atributos e métodos
- Permite herança de um tipo para um subtipo
- Definição

```
CREATE TYPE <nomeTAD> (
   <listaAtributos>
   [<declaraçãoAssinMétodos>] )
[INSTANTIABLE]
[[NOT] FINAL]
```

pode gerar tabelas

pode ou não ser especializado

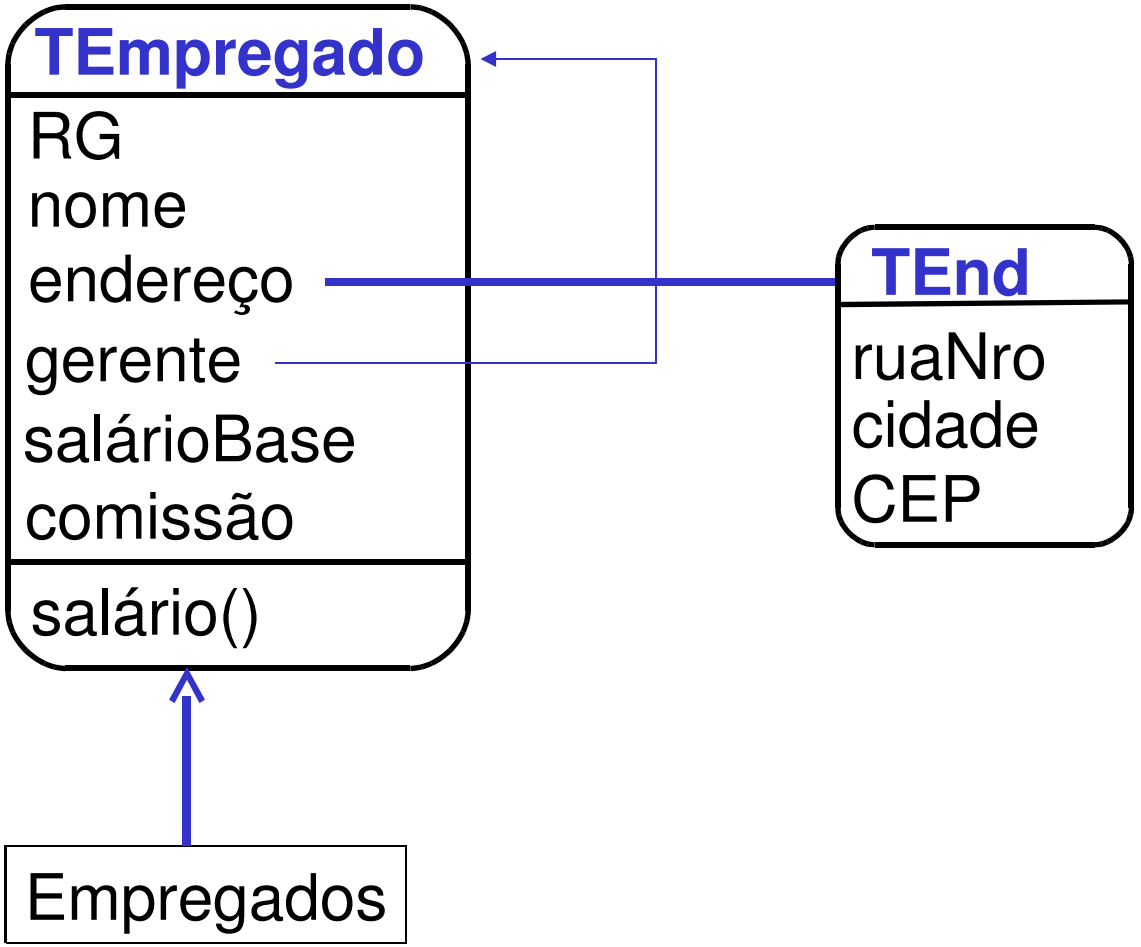
TAD - Exemplo

```
CREATE TYPE TEmpregado (
       RG                INTEGER,
       nome              VARCHAR(40),
       endereço          Tend,
       gerente           REF(TEmpregado),
       salárioBase       DECIMAL (7,2),
       comissão          DECIMAL (7,2),

      METHOD salário() RETURNS DECIMAL (7,2);
    ... )
  INSTANTIABLE
  NOT FINAL;

 CREATE TABLE Empregados OF TYPE TEmpregado;
```

Modelagem OR – TAD



TAD - Comportamento

- SQL-3 permite a definição de métodos, funções e procedimentos
- Implementação de método

```
CREATE METHOD salário()
FOR Tempregado
RETURN REAL
BEGIN
      RETURN salarioBase + comissão*0.8;
END
```

Métodos x Funções/Proc

- Métodos
  - só podem ser definidos dentro de um TAD
  - identificação do método a ser executado é determinado em tempo de execução (*late binding*)
    - depende do objeto que o invoca ou de parâmetros
- Funções/Procedimentos
  - podem ser definidos fora de um TAD
    - `CREATE FUNCTION / CREATE PROCEDURE`
  - identificação da função/procedimento a ser executado é determinado em tempo de compilação (*early binding*)

Métodos

- Consultas SQL podem invocar métodos ou funções

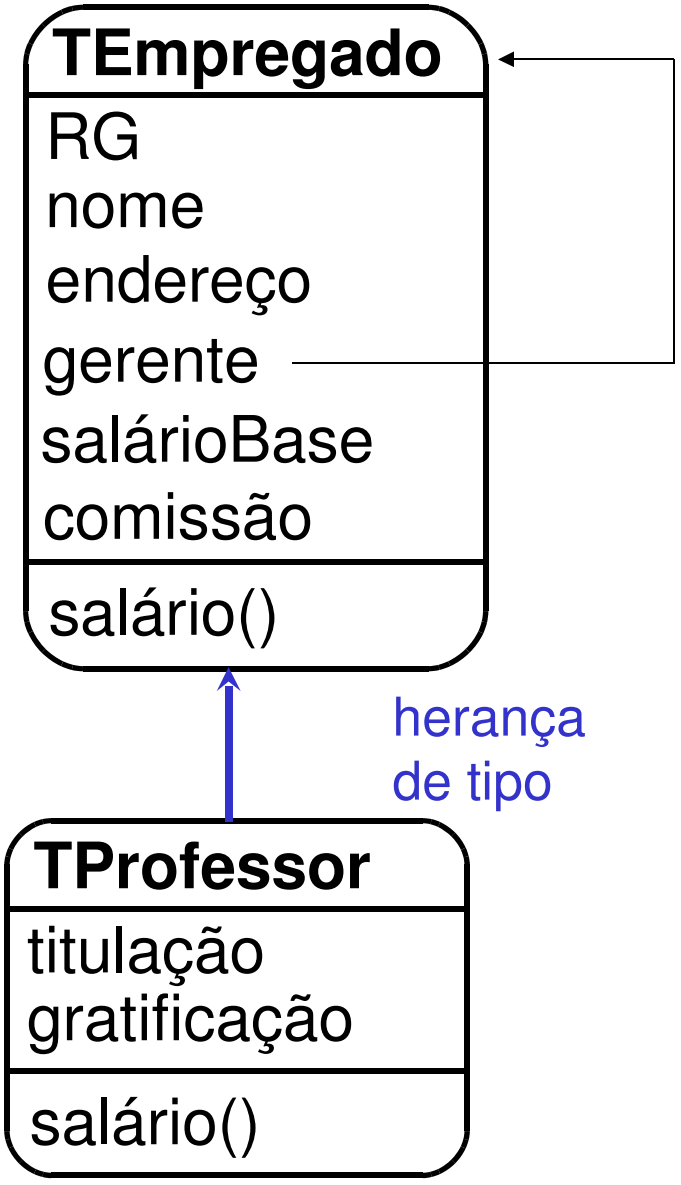
```
select RG,nome
from empregados
where salário() > 1000.00;
```

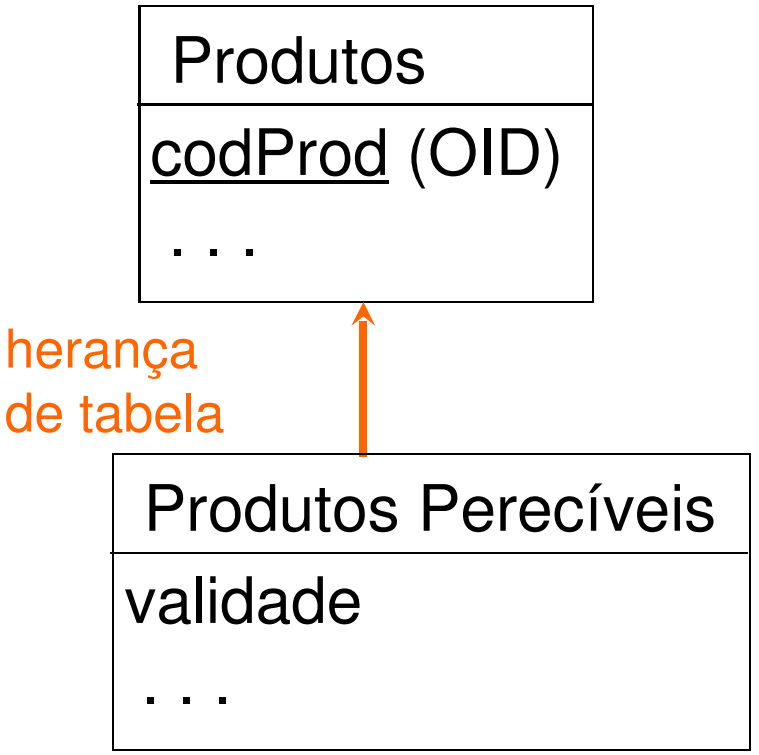
Herança

- Definição
  - `CREATE TYPE <nomeTAD> UNDER <nomeTAD>(...)`
  - `CREATE TABLE <nomeTab> UNDER <nomeTab> (...)`
- Herança múltipla não é permitida
- Exemplo

```
CREATE TYPE Tprofessor UNDER Tempregado (
    titulação           VARCHAR(15),
    gratificação        DECIMAL (7,2),
    OVERRIDING METHOD salário() RETURNS DECIMAL (7,2);
    ... )
 INSTANTIABLE
 NOT FINAL
```

Modelagem OR – Herança





Herança de tabelas implica redundância de dados:

- I na sub-tabela ⇒ I na super-tabela
- E na sub-tabela ⇒ E na super-tabela
- $A(a_i)$ na sub-tabela ⇒ $A(a_i)$ na super-tabela
- $A(a_i)$ na super-tabela ⇒ $A(a_i)$ na sub-tabela

Projeto Lógico de BDOR

- Combina recomendações de projeto de BDR e BDOO

Esquema ER	**Esquema OR**
entidade	tabela (pode-se definir adicionalmente um TAD ou um objeto linha para uma entidade, caso haja necessidade ou não de comportamento e/ou reuso de definição)
entidade fraca	• atributo com domínio tupla (*ROW*) OU • atributo de referência fraca -> forte
relacionamento 1:1	• fusão de entidades em uma tabela OU • referências entre tabelas
relacionamento 1:N	atributo de referência na tabela correspondente à entidade do lado N

Projeto Lógico de BDOR

Esquema ER	Esquema OR
relacionamento M:N	• tabela de relacionamento OU • atributo(s) com domínio(s) *ARRAY*
atributo monovalorado	atributo atômico
atributo composto	atributo com domínio tupla (*ROW*)
atributo multivalorado	atributo com domínio *ARRAY*
especialização	hierarquia de herança entre tipos ou tabelas
entidade associativa	mesmas recomendações para mapeamento de relacionamentos binários

Para resolver

- Apresentar a modelagem lógica objeto-relacional para o domínio da clínica