Gerenciamento de Dados e Informação

Sistemas Objeto-Relacionais

Fernando Fonseca
Ana Carolina
Robson Fidalgo

Conceitos Básicos

Introdução

- A tecnologia de BD tem evoluído para atender à crescente demanda de manipulação de aplicações e dados complexos
- SGBD convencionais (ex: relacional, de rede e hierárquico) são adequados para muitas aplicações comerciais
  - Contudo, aplicações mais recentes têm requisitos e características não triviais que não são bem resolvidas pelos SGBD convencionais

Introdução

- Exemplos de limitações dos SGBD convencionais
  - Não oferecem suporte para implementar diretamente
    - Atributo composto
    - Atributo multivalorado
    - Especialização/Generalização
    - Tipos Complexos
    - Comportamento de objeto

Introdução

- Os SGBDOO surgiram para suprir estas limitações
  - The Object-Oriented Database System Manifesto (1989)
- Porém, os SGBDOO não foram bem aceitos pelo mercado* e pela academia**
  - * Grande esforço tecnológico e financeiro para migrar de SGBDR (dominante do mercado) para SGBDOO
  - ** Falta de padronização e base formal
    - Tentativa de padronização: ODMG

Introdução

- Para contornar a fraca aceitação dos SGBDOO surgiram os SGBDOR
  - Third Generation Database System Manifesto (1990)
  - SGBDOR → mantêm as vantagens do modelo relacional* e acrescentam características do modelo OO**
    - * Modelo eficiente
    - **Modelo mais rico semanticamente
  - A tecnologia OR é uma camada de abstração construída sobre a tecnologia relacional
    - Permite incrementar o legado relacional com tecnologia OO

Introdução

- Sistemas de Banco de Dados Objeto-Relacionais podem ser vistos como uma tentativa de estender sistemas de banco de dados relacionais com a funcionalidade necessária para dar suporte a uma classe mais ampla de aplicações e, de certa forma, prover uma ponte entre os paradigmas relacional e orientado a objetos

Uma Classificação de Aplicações

	Dados Simples	Dados Complexos
Necessidade de Linguagem de Consulta	2	4
Sem necessidade de Linguagem de Consulta	1	3

Uma Classificação de Aplicações

- Quadrante 1: Aplicações com dados simples, sem necessidade de linguagem de consulta
  - Operadores: get file
    put file
  - Bom desempenho
  - Exemplo: um editor de texto tradicional

Gerenciador de Arquivos (sistema operacional)

Uma Classificação de Aplicações

	Dados Simples	Dados Complexos
Necessidade de Linguagem de Consulta	2	4
Sem necessidade de Linguagem de Consulta	Gerenciadores de Arquivos	3

Uma Classificação de Aplicações

- Quadrante 2: Aplicações com dados simples e necessidade de linguagem de consulta
  - Linguagem de consulta
  - Ferramentas de interfaces
  - Desempenho (gerenciamento de transações consistente)
  - Segurança

SGBD relacionais

Uma Classificação de Aplicações

	Dados Simples	Dados Complexos
Necessidade de Linguagem de Consulta	SGBD Relacionais	4
Sem necessidade de Linguagem de Consulta	Gerenciadores de Arquivos	3

Uma Classificação de Aplicações

- Quadrante 3: Aplicações com dados complexos, sem necessidade de linguagem de consulta
  - Necessidade de rotinas específicas para manipulação dos dados complexos
  - Grande integração com uma linguagem de programação
  - Desempenho na atualização de variáveis persistentes

SGBD orientados a Objetos

Uma Classificação de Aplicações

	Dados Simples	Dados Complexos
Necessidade de Linguagem de Consulta	SGBD Relacionais	4
Sem necessidade de Linguagem de Consulta	Gerenciadores de Arquivos	SGBD Orientados a Objetos

Uma Classificação de Aplicações

- Quadrante 4: Aplicações com dados complexos e necessidade de linguagem de consulta
  - Linguagem de Consulta estendida a objetos complexos (SQL3)
  - Ferramentas de visualização não convencionais
  - Otimizador de consultas

SGBD Objeto-relacionais

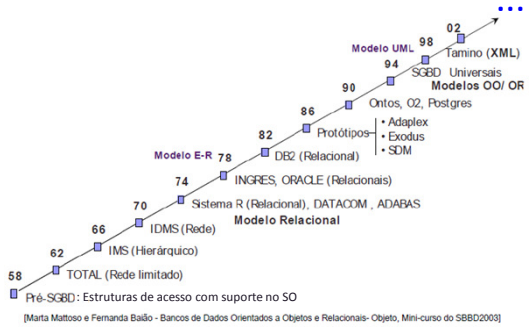
OBJETO: objetos complexos | **RELACIONAL: ling. consulta**

Uma Classificação de Aplicações

	Dados Simples	Dados Complexos
Necessidade de Linguagem de Consulta	SGBD Relacionais 100	SGBD Objeto-relacionais 150
Sem necessidade de Linguagem de Consulta	Gerenciadores de Arquivos	SGBD Orientados a Objetos 1

Previsão para anos 2000

História dos SGBD



[Marta Mattoso e Fernanda Baião - Bancos de Dados Orientados a Objetos e Relacionais- Objeto, Mini-curso do SBBD2003]

Abordagem e Linguagem OR

- A abordagem OR é uma extensão do modelo de dados relacional
  - A extensão permite que usuários estendam o BD a partir da criação de novos tipos e operações
- A linguagem OR é uma extensão de SQL
  - A linguagem SQL estendida oferece suporte para a definição de tipos de dados complexos e métodos, além da instanciação, manipulação e referência de objetos

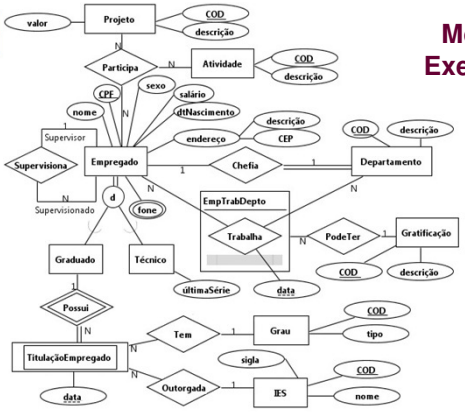
SQL3 → ORACLE 11g

Aspectos OR no Oracle 11g

Oracle OR

- Conceitos básicos
  - Tipo de objetos
  - Métodos
  - Evolução de tipos
  - Herança de tipos
  - Tabela de objetos
  - Tabela de objetos com herança
  - Objetos de linha e objetos de coluna
  - Referência de objetos
  - Coleção de objetos

Modelo Exemplo



Oracle OR – Tipo de Objeto

- É um tipo abstrato de dados (TAD)
  - É um tipo de dado definido pelo usuário que encapsula propriedades (atributos) e comportamento (métodos)
  - Corresponde ao molde de um objeto
    - Não aloca espaço de armazenamento
    - Não pode armazenar dados

Oracle OR – Tipo de Objeto

- Permite capturar inter-relacionamento estrutural de objetos, estendendo a estrutura bidimensional relacional

Clientes

CPF	NOME	ENDEREÇO			FONES
		DESCRIÇÃO	CIDADE	ESTADO	
444.444.444-44	Rita S. Lima	R. Sta. Ana, 10	Olinda	PE	2222-2222
					3333-3333
888.888.888-88	José R. Silva	Av. Recife, 20	Recife	PE	4444-4444
...	...	...	...	...	...

O exemplo acima pode ser feito diretamente em estrutura OR, mas não em estrutura Relacional

Oracle OR – Tipo de Objeto

- São especificados a partir de
  - Atributos → propriedades do objeto (opcional)
  - Métodos → procedimentos ou funções (opcional)

Oracle OR – Tipo de Objeto

- Especificação da interface pública de um objeto
  - **Sintaxe resumida:**

```
CREATE [OR REPLACE] TYPE nome_tipo AS OBJECT (

  [lista de atributos]

  [lista de métodos]

);

/
```


Oracle OR – Tipo de Objeto

- A entidade Projeto do modelo exemplo

Especificação

```
CREATE OR REPLACE TYPE tp_projeto AS OBJECT (
 Cod            INTEGER,
 descricao      VARCHAR2(20),
 valor    NUMBER(12,2),
MEMBER PROCEDURE exibir_detalhes ( SELF tp_projeto),
MAP MEMBER FUNCTION projetoTOInt RETURN INTEGER
) NOT FINAL;
/
```

Para permitir criar um subtipo

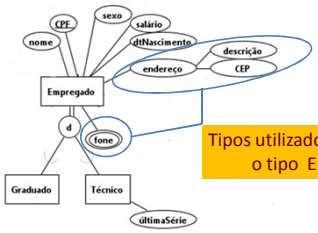
Atributos

Métodos

Não é possível inserir dados em tp_projeto → um tipo de objeto é um molde, não podendo armazenar dados.

Oracle OR – Tipo de Objeto

- Tipos compostos
  - A entidade Empregado do modelo exemplo



Tipos utilizados para compor o tipo Empregado

Oracle OR – Tipo de Objeto

- Inicialmente devem ser definidos os tipos que serão utilizados para compor o tipo mais complexo
  - Ex.: O tipo Endereço

```
CREATE OR REPLACE TYPE tp_endereco AS OBJECT(

  descricao varchar2(30),

  CEP  varchar2(9)

);

/
```


Oracle OR – Tipo de Objeto

- Supondo tendo sido definido o tipo tp_fones, na sequencia deve ser definido o tipo Empregado

```
CREATE OR REPLACE TYPE tp_empregado AS OBJECT(
  CPF varchar2(12),
  nome  varchar2(25),
  sexo char,
  salario number(8,2),
  dtNascimento date,
  endereco tp_endereco,
  fones tp_fones,
  MEMBER FUNCTION salarioAnual RETURN NUMBER,
  ORDER MEMBER FUNCTION comparaSalario (X
          tp_empregado) RETURN INTEGER
) NOT FINAL;
/
```


Oracle OR – Tipo de Objeto

- Pode ser usado da mesma forma que é usado um tipo primitivo
  - O tipo tp_endereco poderia ser utilizado

Para definir o tipo de um atributo de uma tabela

```
CREATE TABLE tb_lojas (
 nome       varchar2(12 ),
 endereco    tp_endereco );

CREATE TABLE tb_fornecedor (
 razao_social   varchar2(20),
 endereco tp_endereco);
```

Para definir o tipo de um atributo de um TAD

```
CREATE TYPE tp_loja AS OBJECT (
  nome  varchar2(12 ),
  endereco tp_ endereco );
/
CREATE TYPE tp_fornecedor AS OBJECT (
 razao_social varchar2 (20),
 endereco tp_endereco);
/
```

Oracle OR – Tipo de Objeto

- Tipos abstratos
  - São tipos que não podem ter instâncias de objetos criadas em tabelas de objetos
  - Podem ser utilizados na definição de outros tipos, os quais podem ser instanciados

```
CREATE OR REPLACE TYPE tp_penalidade AS OBJECT(
  id integer,
  descricao varchar2(30),
) NOT INSTANTIABLE;
/
```

Não instanciável

Oracle OR – Tipo de Objeto

- Métodos
  - São funções ou procedimentos que são declarados na definição de um tipo de objeto
  - Exigem o uso de parênteses (mesmo sem parâmetros)
    - O uso de ( ) é para diferenciar o método de um procedimento ou função comum
  - Podem ser
    - MEMBER
    - MAP ou ORDER
    - Construtor

Oracle OR – Tipo de Objeto

- Um tipo de objeto sempre possui um construtor, pode possuir zero ou mais métodos membro e pode possuir um método map ou um método order, porém não os dois

realiza comparações objeto-a-objeto

fornece a base para comparar objetos, mapeando as Instâncias dos objetos em um dos tipos escalares DATE, NUMBER, VARCHAR2

permitem acesso aos dados da instância do objeto

Método que cria uma nova instância para o objeto, atribuindo valores aos seus atributos

Oracle OR – Tipo de Objeto

- Métodos (Cont.)
  - Construtor
    - Criado implicitamente (pelo Oracle) ao criar um tipo de objeto ou explicitamente pelo programador
    - Deve ser exatamente igual ao nome do tipo
    - Pode haver mais de um construtor para um tipo de objeto, sendo diferenciados pelos parâmetros
    - Utilizado para inserir um novo objeto no BD
  - Inserir dados de um fornecedor

```
INSERT INTO tb_fornecedor VALUES (tp_fornecedor('Casas Araújo',
 tp_endereco ('Rua da Regeneração, 80, Beberibe', '51035-100'));
```

Invocando o método construtor padrão

Oracle OR – Tipo de Objeto

- Métodos (Cont.)
  - MEMBER
    - São os métodos mais comuns
      - Implementam as operações das instâncias do tipo
    - São invocados pela qualificação de objeto
      - Objeto.método()
  - MAP ou ORDER
    - São funções para comparar objetos
      - São mutuamente exclusivos!
    - Métodos ORDER não podem ser definidos em subtipos e são menos eficientes do que métodos MAP

Oracle OR – Tipo de Objeto

- Métodos (Cont.)
  - ORDER
    - Exige como parâmetro um objeto do mesmo tipo
    - Compara o objeto corrente (SELF) com o objeto do parâmetro (x)
    - Usa a lógica interna do objeto para efetuar a comparação entre dois objetos diferentes (mas do mesmo tipo), devolvendo um inteiro correspondente ao tipo de ordem

Inteiro retornado	Interpretação
Positivo	SELF > X
Negativo	SELF < X
Zero	SELF = X

Oracle OR – Tipo de Objeto

- Métodos (Cont.)
  - MAP
    - Não exige parâmetro
    - Compara vários objetos (ex: ORDER BY)
    - Faz uma comparação de tipos padrão, usando atributos do objeto como fatores da comparação
      - Permite comparar objetos mapeando suas instâncias em um dos tipos escalares (ex: DATE, NUMBER, VARCHAR2) ou tipo SQL (ex: CHARACTER ou REAL)

Oracle OR – Tipo de Objeto

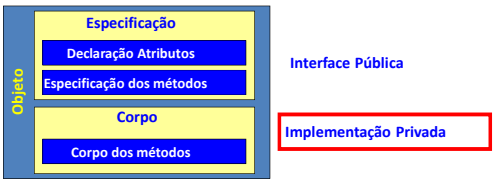
- Métodos
  - MAP (Cont.)
    - Retorna um dos atributos do objeto
    - É chamado implicitamente quando há comparação de objetos, como por uso de
      - DISTINCT, GROUP BY, UNION e ORDER BY
    - Só podem ser declarados em um subtipo se houver um método MAP declarado no supertipo

Oracle OR – Tipo de Objeto

Caso não seja especificado um método MAP ou ORDER, o Oracle só pode concluir se dois objetos são iguais, nada podendo dizer sobre a ordem de relacionamento entre estes objetos

Oracle OR – Tipo de Objeto

- Métodos (Cont.)



Oracle OR – Tipo de Objeto

- Métodos (Cont.)
  - Implementação privada do corpo de métodos de um objeto
    - **Sintaxe resumida:**

```
CREATE [OR REPLACE] TYPE BODY nome_tipo AS
   [lista de subprogramas - procedimento, função ou
construtor-]
   [lista de funções MAP ou ORDER]
END ;
/
```


Oracle OR – Tipo de Objeto

- Métodos (Cont.)
  - Corpo dos métodos para o tipo tp_projeto

```
CREATE OR REPLACE TYPE BODY tp_projeto AS
MEMBER PROCEDURE exibir_detalhes ( SELF tp_projeto) IS
BEGIN
   DBMS_OUTPUT.PUT_LINE('Detalhes de um Projeto');
   DBMS_OUTPUT.PUT_LINE('CÓDIGO: ' ||TO_CHAR(cod));
   DBMS_OUTPUT.PUT_LINE('DESCRIÇÃO: ' || descricao);
   DBMS_OUTPUT.PUT_LINE('VALOR: ' || 'R$. ' || TO_CHAR(valor));
 END;
MAP MEMBER FUNCTION projetoTOInt RETURN INTEGER IS
 p INTEGER := cod;
BEGIN
   RETURN p;
 END;
END;
/
```

Oracle OR – Tipo de Objeto

- Métodos (Cont.)
  - Corpo dos métodos para tp_empregado

```
CREATE OR REPLACE TYPE BODY tp_empregado IS
MEMBER FUNCTION salarioAnual RETURN NUMBER IS
BEGIN
RETURN salario*12;
END;
ORDER MEMBER FUNCTION comparaSalario (X tp_empregado)
RETURN INTEGER IS
BEGIN
RETURN SELF.salario - X.salario;
END;
END;
/
```


Oracle OR – Tipo de Objeto

- Evolução de tipos
  - A partir do uso de ALTER TYPE é possível
    - Adicionar e excluir atributos
    - Adicionar e excluir métodos
    - Modificar as propriedades de um atributo
      - Tamanho, precisão e tipo
    - Modificar o status FINAL e INSTANTIABLE de um tipo
    - . . .

Oracle OR – Tipo de Objeto

- Evolução de tipos (Cont.)
  - Acrescentar o atributo ag_financiadora ao tipo tp_projeto

```
ALTER TYPE tp_projeto
  ADD ATTRIBUTE (ag_financiadora VARCHAR2(5)) CASCADE ;
```

Vai propagar a mudança para todos os tipos dependentes

  - Remover o método MAP projetoToInt do tipo tp_projeto

```
ALTER TYPE tp_projeto
  DROP MAP MEMBER FUNCTION projetoToInt RETURN NUMBER
INVALIDATE;
```

Vai invalidar todos os tipos dependentes

Oracle OR – Tipo de Objeto

- Evolução de tipos (Cont.)
  - Alterar a definição do tipo tp_projeto para não permitir especialização

```
ALTER TYPE tp_projeto FINAL INVALIDATE;
```

Vai invalidar todos os tipos dependentes

  - Modificar a definição do atributo ag_financiadora do tipo tp_projeto para permitir registrar FACEPE como agência financiadora

```
ALTER TYPE tp_projeto
  MODIFY ATTRIBUTE ag_financiadora VARCHAR2(6) CASCADE ;
```

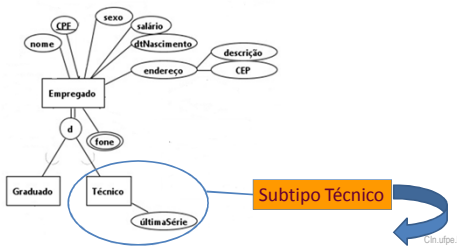
Modificando o tamanho do atributo

Oracle OR – Tipo de Objeto

- Herança de tipos
  - Permite criar uma hierarquia de subtipos especializados
  - Os tipos derivados (subtipos) herdam os atributos e métodos dos tipos ancestrais (supertipos)
  - Os subtipos podem acrescentar novos atributos ou métodos e/ou redefinir os métodos herdados dos supertipos

Oracle OR – Tipo de Objeto

- Herança de tipos (Cont.)
  - Considerando a hierarquia de tipos de Empregado



Oracle OR – Tipo de Objeto

- Herança de tipos (Cont.)
  - Criar o subtipo tp_tecnico do tipo tp_empregado

```
CREATE TYPE tp_tecnico UNDER tp_empregado(
  ultimaSerie  VARCHAR2(30)
) NOT FINAL;
/
```

Para permitir definição de subtipos

Por default um tipo de objeto é FINAL!

Oracle OR – Tipo de Objeto

- Herança de tipos (Cont.)
  - Os métodos também podem ser declarados como FINAL
    - Os subtipos não podem redefinir sua implementação
  - Diferentemente dos tipos de objetos, os métodos são definidos por padrão como NOT FINAL

Oracle OR – Tipo de Objeto

- Herança de tipos (Cont.)
  - Definir o tipo para a entidade Atividade, considerando que seu método MAP atividadeToCadeia não deve ser redefinido por seus subtipos

```
CREATE OR REPLACE TYPE tp_atividade AS object (
Cod integer,
descricao varchar2(20),
FINAL MAP MEMBER FUNCTION atividadeToCadeia RETURN VARCHAR2
) NOT FINAL;
/
```

Oracle OR – Tipo de Objeto

- Herança de tipos (Cont.)
  - Corpo do método atividadeToCadeia

```
CREATE OR REPLACE TYPE BODY tp_atividade AS
FINAL MAP MEMBER FUNCTION atividadeToCadeia RETURN
VARCHAR2 IS
 p VARCHAR2(20) := descricao;
 BEGIN
   RETURN p;
 END;
END;
/
```


Oracle OR – Tipo de Objeto

- Herança de tipos (Cont.)
  - Construir o tipo tp_graduado para a especialização Graduado, considerando:
    - Criar método construtor
    - O método salarioAnual deve obrigatoriamente ser redefinido para acrescentar 10% de gratificação do total calculado para os demais empregados

Subtipo Graduado

Oracle OR – Tipo de Objeto

- Herança de tipos (Cont.)

Método salarioAnual deve ser redefinido

```
CREATE OR REPLACE TYPE tp_graduado UNDER tp_empregado(
CONSTRUCTOR FUNCTION tp_graduado (x1 tp_empregado)
        RETURN SELF AS RESULT,
OVERRIDING MEMBER FUNCTION salarioAnual RETURN
NUMBER
);
/
```

Herda os atributos de Empregado
(CPF, nome, sexo, salario, dtNascimento, endereco, fones)

Oracle OR – Tipo de Objeto

- Herança de tipos (Cont.)
  - Corpo do tipo tp_graduado

```
CREATE OR REPLACE TYPE BODY tp_graduado AS
CONSTRUCTOR FUNCTION tp_graduado (x1 tp_empregado)          RETURN
SELF AS RESULT IS
  BEGIN
     cpf := x1.cpf; nome := x1.nome; sexo := x1.sexo;  salario := x1.salario;
dtNascimento := x1. dtNascimento; endereco := x1. endereco; fones := x1.
fones; supervisor := x1.supervisor; RETURN;
  END;
OVERRIDING MEMBER FUNCTION salarioAnual RETURN NUMBER IS
BEGIN
    RETURN salario*12*1.1;
  END;
END;
/
```


Oracle OR – Tabela de Objetos

- São tabelas especiais onde cada linha armazena um objeto
  - Provê uma **visão relacional** desses objetos
  - As linhas de uma tabela de objetos possuem um OID (object identifier) implícito (definido pelo ORACLE)
  - Os objetos de uma tabela de objetos podem ser referenciados (REF) por outros objetos
  - Nos comandos de manipulação de objetos utilizar aliases para as tabelas
- OID pode ser definido pelo programador (só recomendado para objetos interoperáveis entre diferentes BD)

Oracle OR – Tabela de Objetos

- Uma tabela OR pode ser definida com uma única coluna ou múltiplas colunas – Ex.: A tabela tb_atividade
  - Única coluna (tabela de objetos)
    - Cada linha sendo um objeto do tipo tp_atividade

```
CREATE TABLE tb_atividade of tp_atividade
(cod     PRIMARY KEY );
```

Fazer o mesmo para outras restrições
EX: UNIQUE, NOT NULL, FOREIGN KEY, CHECK

  - Múltiplas colunas
    - Uma coluna para cada atributo do tipo tp_atividade
      - EX: Todos já vistos até agora

Oracle OR – Tabela de Objetos

- Inserção de objetos em tabelas de objeto – Ex.: tb_atividade

```
INSERT INTO tb_atividade VALUES (tp_atividade(1, 'Analista'));
```

1 row created.

```
INSERT INTO tb_atividade VALUES (tp_atividade(2, 'Administrador'));
```

1 row created.

```
INSERT INTO tb_atividade VALUES (tp_atividade(3, 'Programador');
```

1 row created.

  - Consulta objetos em tb_atividade

```
SELECT * FROM tb_atividade;
```

COD	DESCRICAO
1	Analista
2	Administrador
3	Programador

Oracle OR – Tabela de Objetos

- Consulta método MAP nos objetos de tb_atividade – Ordenação
  - Listar as atividades em ordem alfabética

Alias

```
SELECT a.cod, a.descricao FROM tb_atividade  a
 ORDER BY a.atividadeToCadeia( );
```

Método deve ser chamado sempre com uso de parênteses

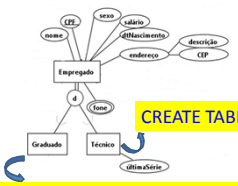
COD	DESCRICAO
2	Administrador
1	Analista
3	Programador

Oracle OR – Tabela de Objetos

- Tabelas de objetos (object tables) para herança
  - Não há estrutura de armazenamento associada com os tipos que pertencem a uma hierarquia
    - Deve-se criar tabelas de objetos para manipular as hierarquias dos tipos, formando uma hierarquia de tabelas
    - Por razões de eficiência, deve-se armazenar os objetos em uma tabela definida com o último tipo na hierarquia

Oracle OR – Tabela de Objetos

- Tabelas de objetos (object tables) para herança (Cont.)
  - A hierarquia de Empregado




```
CREATE TABLE tb_tecnico OF tp_tecnico;
```

```
CREATE TABLE tb_graduado of tp_graduado;
```


Oracle OR – Tabela de Objetos

- Tabelas de objetos para herança (Cont.)
  - Descrição das tabelas especializadas – Ex.: tb_tecnico

```
SQL> desc tb_tecnico;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CPF                                                VARCHAR2(12)
 NOME                                               VARCHAR2(25)
 SEXO                                               CHAR(1)
 SALARIO                                            NUMBER(8,2)
 DTNASCIMENTO                                       DATE
 ENDERECO                                           TP_ENDERECO
 FONES                                              TP_FONES
 ULTIMASERIE                                        VARCHAR2(30)
```

Tp_empregado

Oracle OR – Tabela de Objetos

- Tabela de Objetos para herança (Cont.)
  - Exemplo de uso – Objeto graduado
    - Criação de objeto

```
INSERT INTO tb_graduado VALUES
        (tp_graduado(tp_empregado('132516702-16',
          'Ana Paula', 'F', 12345.00, to_date('10/04/1970',
          'dd/mm/yyyy'), tp_endereco('R. Janaína, 15',
'52020-200'), null)));
1 row created.
```


Oracle OR – Tabela de Objetos

- Exemplo de uso – Objeto graduado (Cont.)
  - Consulta objetos

```
select * from tb_graduado;
```

CPF	NOME	S	SALARIO	DTNASCIME	ENDERECO (DESCRICAO, CEP)	FONES(DESCRICAO)
132516702-16	Ana Paula	F	12345	10-APR-70	TP_ENDERECO('R. Janaína, 15', '52010-200')	

Empregado (CPF,NOME, SEXO, SALARIO, DTNASCIME, ENDERECO, FONES)

Oracle OR – Tabela de Objetos

- Exemplo de uso – Objeto graduado (Cont.)
  - Consulta método membro

```
select g.salarioAnual( ) from tb_graduado g;
```

G.SALARIOANUAL()
162954

  - Consulta método ORDER
    - Comparar os salários dos empregados cujos CPF são 132516702-16 e 420316123-45

Oracle OR – Tabela de Objetos

```
DECLARE
mb tp_graduado;
m number;
BEGIN
SELECT VALUE(p) INTO mb FROM tb_graduado p
WHERE p.cpf = '132516702-16';
SELECT d.comparasalario(mb) into m FROM tb_graduado d WHERE
d.cpf = '420316123-45';
IF m > 0 THEN DBMS_OUTPUT.PUT_LINE('EMPREGADO DE CPF: '
||'420316123-45' || ' TEM SALARIO MAIOR QUE O DO EMPREGADO DE CPF: '
||TO_CHAR(mb.cpf) ); END IF;
IF m = 0 THEN DBMS_OUTPUT.PUT_LINE('EMPREGADO DE CPF: '
||'420316123-45' || ' TEM SALARIO IGUAL AO DO EMPREGADO DE CPF: '
||TO_CHAR(mb.cpf) ); END IF;
IF m < 0 THEN DBMS_OUTPUT.PUT_LINE('EMPREGADO DE CPF: '
||'420316123-45' || ' TEM SALARIO MENOR QUE O DO EMPREGADO DE CPF: '
||TO_CHAR(mb.cpf) );END IF;
END;
/
```

Determina empregado A ser compararado

Determina empregado corrente

Realizar comparações

Oracle OR – Tabela de Objetos

- Consulta método ORDER (Cont.)

Empregado de CPF: 420316123-45 TEM SALARIO MENOR QUE O DO EMPREGADO DE CPF: 132516702-16

Procedimento PL/SQL concluído com sucesso.

CPF	SALARIO
132516702-16	12345
215439210-15	10115
420316123-45	8500

Oracle OR – Tabela de Objetos

- Outro exemplo
  - Criar a tabela tb_projeto

```
CREATE TABLE tb_projeto OF tp_projeto;
```

  - Inserção de objetos em tb_projeto

```
INSERT INTO tb_projeto VALUES (tp_projeto(1, 'Requisitos', 100000.00));
```

1 row created.

```
INSERT INTO tb_projeto VALUES (tp_projeto(2, 'Diagramas', 50900.00));
```

1 row created.

```
INSERT INTO tb_projeto VALUES (tp_projeto(3, 'Codificação', 50900.00));
```

1 row created.

Oracle OR – Tabela de Objetos

- Consulta objetos em Projeto

```
SELECT * FROM tb_projeto;
```

COD	DESCRICAO	VALOR
1	Requisitos	100000
2	Diagramas	50900
3	Codificação	50900

Oracle OR – Tabela de Objetos

- Consultar detalhes de objeto corrente Projeto – Método exibirdetalhes

```
DECLARE
mb tp_projeto;
BEGIN
SELECT VALUE(p) INTO mb FROM tb_projeto p
WHERE p.cod = 2;
mb.exibir_detalhes();
END;
/
```

Construir bloco para chamar método procedure

Oracle OR – Tabela de Objetos

- Consultar detalhes de objeto corrente Projeto – Método exibirdetalhes (Cont.)

Detalhes de um Projeto
CÓDIGO: 2
DESCRIÇÃO: Diagramas
VALOR: R$. 50900

Procedimento PL/SQL concluído com sucesso.

Oracle OR – Tabela de Objetos

- Objeto de linha e objeto de coluna
  - Além dos objetos armazenados em tabelas (Row Objects), pode haver objetos armazenados em colunas (Column Objects)
    - Column Objects: são objetos armazenados em **colunas** de tabelas relacionais ou como **atributos** de tipos objetos

```
CREATE TABLE tb_contatos (
 contato      tp_tecnico,
 dt_contato   DATE );
```

```
CREATE TYPE tp_contatos AS OBJECT (
 contato      tp_ tecnico,
 dt_contato   DATE );
```

ORACLE OR – Tabela de Objetos

- OID (OBJECT IDENTIFIER)
  - Cada objeto possui um identificador único ou manipulador
    - É automaticamente atribuído quando um objeto é armazenado em uma object table
    - É armazenado em uma coluna oculta de 16 bytes do tipo RAW
    - Pode ser referenciado por colunas em outras tabelas, analogamente à chave estrangeira referenciando uma chave primária

Oracle OR – Tabela de Objetos

- Referência de objetos
  - É um ponteiro lógico para um Row Object
    - Usado para fazer referência
    - É definido a partir do OID do objeto
    - Oferece acesso rápido/direto
    - Não garante integridade referencial

ORACLE OR - Tabela de Objetos

- Referência de objetos (Cont.)
  - REF

    Referências para objetos são do tipo REF
    - Um atributo pode ser declarado como um REF (uma referência) para um tipo de objeto
    - Referências para objetos são úteis para identificar unicamente e localizar um objeto
    - Somente é possível obter referências para objetos que possuam OID, ou seja, só é possível referenciar objetos armazenados em object tables

ORACLE OR - Tabela de Objetos

- Referência de objetos (Cont.)
  - REF em Colunas
    - Uma coluna de uma tabela (ou um atributo de um object type) pode ser declarado como sendo do tipo REF

```
...
<atributo> REF <tipo de objeto>;
...
```


ORACLE OR - Tabela de Objetos

- Referência de objetos (Cont.)
  - REF como operador
    - Quando é necessário obter o identificador de um objeto de uma tabela, utiliza-se o operador REF(), tendo como argumento o aliás de uma object table

```
SELECT REF(P) FROM <tabela> P WHERE ...;
```

Alias

Condição deve retornar só um objeto

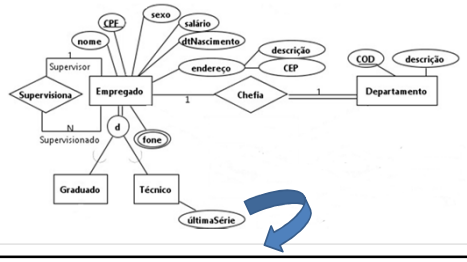
ORACLE OR - Tabela de Objetos

- Referência de objetos (Cont.)
  - Uma coluna do tipo REF pode referenciar objetos do tipo indicado que estejam em qualquer tabela
  - Para restringir o escopo de referências para uma única tabela usar

SCOPE IS

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Considerando que um Empregado só pode chefiar um Departamento ou supervisionar outro, caso ele seja Graduado



Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Alterando a definição de Empregado

```
ALTER TYPE tp_empregado
ADD ATTRIBUTE (supervisor REF tp_graduado)
CASCADE ;
```

  - Criando tipo Departamento

```
CREATE OR REPLACE TYPE tp_departamento as OBJECT(
cod            INTEGER(3),
descricao      VARCHAR (30),
Chefe          REF  tp_graduado);
/
```


Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Criando a tabela Departamento

```
CREATE TABLE tb_departamento OF tp_departamento(
cod            PRIMARY KEY,
descricao      NOT NULL,
chefe   SCOPE IS tb_graduado);
```

Propriedades

Só aceita objetos da tabela tb_graduado

Indispensável quando houver mais de uma tabela definida para o mesmo tipo de objeto

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Inserindo dados nas tabelas

```
INSERT INTO tb_graduado VALUES
        (tp_graduado(tp_empregado('215439210-15',
          'Jonas Mota', 'M', 10115.00, to_date('12/03/1975',
          'dd/mm/yyyy'), tp_endereco('R. Sanharó, 32',
'51020-710'), null, (SELECT REF(G) FROM tb_graduado G
WHERE cpf ='132516702-16') )));
```

1 row created.

```
INSERT INTO tb_graduado VALUES
        (tp_graduado(tp_empregado('420316123-45',
          'Helena Ramos', 'F', 8500.00, to_date('21/11/1982',
          'dd/mm/yyyy'), tp_endereco('R. Roriz, 100',
'50135-316'), null, (SELECT REF(G) FROM tb_graduado G
WHERE cpf ='132516702-16') )));
```

1 row created.

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)

```
INSERT INTO tb_departamento
SELECT 1, 'Finanças', REF (G)
FROM tb_graduado G WHERE cpf = '215439210-15';
```

1 row created.

```
INSERT INTO tb_departamento
SELECT 2, 'Projetos', REF (G)
FROM tb_graduado G WHERE cpf = '420316123-45';
```

1 row created.

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Consultar dados de Departamento

```
SELECT * FROM tb_departamento D;
```

Referência do objeto Graduado

COD	DESCRICAO	CHEFE
1	Finanças	0000220208F9383E69D8A40222E04015AC0702244EF9383E69D87A0222E04015AC0702244E
2	Projetos	0000220208F9383E69D8A50222E04015AC0702244EF9383E69D87A0222E04015AC0702244E

```
SELECT REF(D) FROM tb_departamento D WHERE D.descricao
= 'Projetos';
```

Referência do próprio objeto

REF(D)
0000280209F9383E69D8A70222E04015AC0702244EF9383E69D8A30222E04015AC0702244E0100656C0001

Oracle OR – Tabela de Objetos

- Consultar dados de Departamento (Cont.)

```
SELECT descricao, chefe FROM tb_departamento;
```

DESCRICAO	CHEFE
Finanças	0000220208F9383E69D8A40222E04015AC0702244EF9383E69D87A0222E04015AC0702244E
Projetos	0000220208F9383E69D8A50222E04015AC0702244EF9383E69D87A0222E04015AC0702244E

OID gerado pelo Oracle

Oracle OR – Tabela de Objetos

- Consultar dados de Departamento (Cont.)

```
SELECT D.chefe.cpf as CPF_Chefe,
       D.chefe.nome as Nome_Chefe,
       D.Descricao as Departamento
FROM tb_departamento D;
```

Propriedades dos objetos tp_graduado

CPF_CHEFE	NOME_CHEFE	DEPARTAMENTO
215439210-15	Jonas Mota	Finanças
420316123-45	Helena Ramos	Projetos

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Verificando a validade das referências (Dangling)
    - Remover o empregado graduado Jonas Mota (CPF

```
DELETE FROM tb_graduado WHERE cpf = '215439210-15';
```

Remove o objeto Jonas Mota

```
SELECT D.chefe.cpf as CPF_Chefe,
       D.chefe.nome as Nome_Chefe,
       D.descricao as Departamento
FROM tb_departamento D;
```

CPF_CHEFE	NOME_CHEFE	DEPARTAMENTO
		Finanças
420316123-45	Helena Ramos	Projetos

O objeto Jonas Melo não é listado, mas Helena Ramos continua aparecendo

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Verificando a validade das referências (Dangling)

```
SELECT D.chefe.cpf as CPF_Chefe,
       D.chefe.nome as Nome_Chefe,
       D.descricao as Departamento
FROM tb_departamento D
WHERE D.chefe IS DANGLING;
```

CPF_CHEFE	NOME_CHEFE	DEPARTAMENTO
		Finanças

Só aparecem os objetos sem referências válidas para Chefe

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Verificando a validade das referências (Dangling)

```
SELECT D.chefe.cpf as CPF_Chefe,
       D.chefe.nome as Nome_Chefe,
       D.descricao as Departamento
FROM tb_departamento D
WHERE D.chefe IS NOT DANGLING;
```

CPF_CHEFE	NOME_CHEFE	DEPARTAMENTO
420316123-45	Helena Ramos	Projetos

Só aparecem os objetos com referências válidas para Chefe

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Verificando a validade das referências (Dangling)

```
SELECT D.chefe.cpf as CPF_Chefe,
       D.chefe.nome as Nome_Chefe,
       D.descricao as Departamento
FROM tb_departamento D
WHERE D.chefe IS NOT NULL;
```

CPF_CHEFE	NOME_CHEFE	DEPARTAMENTO
		Finanças
420316123-45	Helena Ramos	Projetos

DANGLING != NULL

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Garantindo a integridade referencial
    - Cláusula WITH ROWID
      - Importa o OID e a identificação física da linha onde o objeto está armazenado
      - Mantém o acesso direto ao objeto (Bom desempenho)
    - Tabela tb_departamento
      - Remover a definição anterior da tabela Departamento para redefinir considerando a garantia de integridade referencial

```
DROP TABLE tb_departamento;
```


Oracle OR – Tabela de Objetos

- Garantindo a integridade referencial (Cont.)

```
CREATE TABLE tb_departamento OF tp_departamento(
cod             PRIMARY KEY,
descricao               NOT NULL,
chefe   WITH ROWID REFERENCES tb_graduado);
```

Garante a integridade referencial

Faz a REF para o objeto da tabela Graduado

Oracle OR – Tabela de Objetos

- Referência de objetos
  - Garantindo a integridade referencial (Cont.)
    - Inserir o Departamento 3, RH, com Jonas Melo (CPF 215439210-15) como chefe

```
INSERT INTO tb_departamento
SELECT 3, 'RH', REF (G)
FROM tb_graduado G WHERE cpf = '215439210-15';
```

0 rows created.

Não deve permitir inserir, pois o empregado Jonas Melo foi excluído

- Inserir o Departamento 2, Projetos, com Helena Ramos (CPF 420316123-45) como chefe

```
INSERT INTO tb_departamento
SELECT 2, 'Projetos', REF (G)
FROM tb_graduado G WHERE cpf = '420316123-45';
```

1 row created.

Oracle OR – Tabela de Objetos

- Referência de objetos
  - Garantindo a integridade referencial (Cont.)

```
DELETE FROM tb_graduado WHERE cpf= '420316123-45';
```

```
DELETE FROM tb_graduado
*
ERROR at line 1:
ORA-02292: integrity constraint (U_FDFD.SYS_C00100072) violated - child record found
```

Deve lançar um erro, pois o graduado 420316123-45 tem dependente

- Inserir o Departamento 2, Finanças, com Ana Paula (CPF 132516702-16) como chefe

```
INSERT INTO tb_departamento
SELECT 1, 'Finanças', REF (G)
FROM tb_graduado G WHERE cpf = '132516702-16';
```

1 row created.

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - DEREF
    - Retorna um objeto referenciado por uma coluna do tipo REF
    - Aplicar DEREF a um objeto dangling retorna um objeto null

Oracle OR – Tabela de Objetos

- Referência de objetos (Cont.)
  - Usando o DEREF
    - Consulta Departamento

```
SELECT DEREF(D.chefe) as CHEFE, D.descricao as Departamento
FROM tb_departamento D;
```

CHEFE(CPF, NOME, SEXO, SALARIO, DTNASCIMENTO, ENDERECO(DESCRICAO, CEP), FONES(DESCRICAO), SUPERVISOR)	DEPARTAMENTO
TP_GRADUADO('420316123-45', 'Helena Ramos', 'F', 8500, '21-NOV-82', TP_ENDERECO('R. Roriz, 100', '50135-316'), NULL, NULL)	Projetos
TP_GRADUADO('132516702-16', 'Ana Paula', 'F', 12345, '10-APR-70', TP_ENDERECO('R . Janaína, 15', '52010-200'), NULL, NULL)	Finanças

Retorna os objetos do tipo tp_graduado

Oracle OR – Tabela de Objetos

- Referência de Objetos (Cont.)
  - Consulta Departamento sem usar DEREF

```
SELECT D.Chefe as Chefe, D.descricao as Departamento
FROM tb_departamento D;
```

CHEFE	DEPARTAMENTO
0000220208F9383E69D8A50222E04015AC0702244EF9383E69D87A0222E04015AC0702244E	Projetos
0000220208F9383E69D87B0222E04015AC0702244EF9383E69D87A0222E04015AC0702244E	Finanças

Retorna o OID dos objetos

Oracle OR – Tabela de Objetos

- Referência de Objetos
  - Usando o DEREF (Cont.)
    - Acessando diretamente um atributo de um objeto referenciado

Ambos válidos

```
SELECT DEREF(D.chefe).nome as Nome_Chefe,
D.descricao as Departamento
FROM tb_departamento D;
```

Retorna o nome do objeto Chefe

- No exemplo abaixo, DEREF não é necessário

```
SELECT D.chefe.nome as Nome_Chefe,
D.descricao as Departamento
FROM tb_departamento D;
```

NOME_CHEFE	DEPARTAMENTO
Helena Ramos	Projetos
Ana Paula	Finanças

Oracle OR – Tabela de Objetos

- Acessando diretamente um atributo de um objeto referenciado
  - Informar para cada empregado graduado que tenha supervisor, seu nome e o nome do supervisor

```
SELECT g.nome as Empregado, g.supervisor.nome as Supervisor
FROM tb_graduado g WHERE g.supervisor IS NOT NULL;
```

EMPREGADO	SUPERVISOR
Jonas Mota	Ana Paula
Helena Ramos	Ana Paula

Oracle OR – Tabela de Objetos

- Referência de Objetos (Cont.)
  - Operador VALUE
    - Exibe os dados das instâncias dos objetos
    - Usa o mesmo formato que DEREF
      - Consultar dados dos chefes dos Departamentos

```
SELECT VALUE(D) Value_Depto FROM tb_departamento D;
```

VALUE_DEPTO (COD, DESCRICAO, CHEFE)
TP_DEPARTAMENTO(2, 'Projetos', 0000220208F9383E69D8A50222E04015AC0702244EF9383E69D87A0222E04015AC0702244E)
TP_DEPARTAMENTO(1, 'Finanças', 0000220208F9383E69D87B0222E04015AC0702244EF9383E69D87A0222E04015AC0702244E)

Retorna os objetos do tipo tp_departamento

Oracle OR – Coleções de Objetos

- Podem ser usadas para representar
  - Atributos multivalorados
  - Relacionamentos 1:n, n:1 ou n:m
  - São de dois tipos
    - VARRAY
    - NESTED TABLE

Oracle OR – Coleções de Objetos

- VARRAY X NESTED TABLE
  - Varray: coleção **ordenada** de uma quantidade **fixa** de elementos (índice inicia a partir de 1)
    - São armazenados como objetos contínuos
    - Um varray é armazenado "in line", ou seja, na mesma estrutura da tabela
  - Nested table: coleção não ordenada de uma quantidade **arbitrária** de elementos
    - É uma tabela aninhada (tabela de uma tabela)
    - É armazenada "out line", ou seja, em uma outra estrutura (tabela)

Oracle OR – Coleções de Objetos

- VARRAY X NESTED TABLE (Cont.)
  - Varray: Indicada quando é necessário acessar elementos pelo índice ou manipular a coleção inteira como um valor
  - Nested table: indicada quando é necessário eficiência na execução de consultas sobre coleções

Oracle OR – Coleções de Objetos

- Uso de VARRAY
  - Definir tipo tp_fone

```
CREATE OR REPLACE TYPE tp_fone AS OBJECT (
  cod_area    VARCHAR2(2),
  numero      VARCHAR2(8));
/
```

  - Redefinir tipo tp_fones
    - Remover tabelas que utilizem qualquer tipo dependente de tp_fones
    - Remover todos os tipos dependentes de tp_fones
    - Remover tp-fones

Oracle OR – Coleções de Objetos

- Redefinir tipo tp_fones como uma coleção de telefones

```
CREATE OR REPLACE TYPE tp_fones AS VARRAY(5) OF tp_fone;
/
```

- Definir tabela relacional com atributo VARRAY

```
CREATE TABLE tb_fones_departamento(
 cod_depto NUMBER(5),
 lista_fones tp_fones);
```

  - Inserir tupla na tabela tb_fones_departamento

```
INSERT INTO tb_fones_departamento VALUES (100,
 tp_fones (tp_fone('81', '22222222'),
           tp_fone('81', '33333333'),
           tp_fone('81', '44444444')));
```

1 row created.

Oracle OR – Coleções de Objetos

- Consulta tabela com atributo do tipo VARRAY

```
SELECT * FROM tb_fones_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
100	TP_FONES(TP_FONE('81', '22222222'), TP_FONE('81', '33333333'), TP_FONE('81', '44444444'))

Retorna objeto do tipo tp_fones

Oracle OR – Coleções de Objetos

- Coleções como NESTED TABLE
  - Criar tipo nested table de telefones

```
CREATE TYPE tp_nt_fone AS TABLE OF tp_fone;
/
```

  - Criar tabela com atributo nested table de telefones

```
CREATE TABLE tb_lista_fone_departamento(
  cod_depto NUMBER(5),
  lista_fone tp_nt_fone)
NESTED TABLE lista_fone STORE AS tb_lista_fone;
```

  - Inserir objeto na tabela com atributo do tipo nested table de telefones

```
INSERT INTO tb_lista_fone_departamento VALUES (1,
 tp_nt_fone (tp_fone('81', '55555555'),
             tp_fone('81', '66666666')));
```

1 row created.

Oracle OR – Coleções de Objetos

- Consulta tabela com atributo do tipo nested table

```
SELECT * FROM tb_lista_fone_departamento;
```

COD_DEPTO	LISTA_FONE(COD_AREA, NUMERO)
1	TP_NT_FONE(TP_FONE('81', '55555555'), TP_FONE('81', '66666666'))

Retorna objetos do tipo tp_nt_fones

Oracle OR – Coleções de Objetos

- Uso de NESTED TABLE – definições ligeiramente modificadas
  - Definir tipo de objeto contendo um atributo nested table

```
CREATE OR REPLACE TYPE tp_lista_fones_Departamento AS OBJECT(
 cod_depto NUMBER(5),
 lista_fones tp_nt_fone);
/
```

  - Criar tabela de objetos do tipo tp_lista_fones_departamento

```
CREATE TABLE tb_lista_fones_Departamento  OF tp_lista_fones_departamento
NESTED TABLE lista_fones STORE AS tb_lista_fones;
```


Oracle OR – Coleções de Objetos

- Inserir objeto na tabela tb_lista_fones_departamento

```
INSERT INTO tb_lista_fones_departamento VALUES
(tp_lista_fones_Departamento (1, tp_nt_fone (tp_fone('81',
'55555555'),tp_fone('81', '66666666')));
```

1 row created.

- Consulta tabela tb_lista_fones_departamento

```
SELECT * FROM tb_lista_fones_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
1	TP_NT_FONE(TP_FONE('81', '55555555'), TP_FONE('81', '66666666'))

Retorna objeto do tipo tp_nt_fones

Oracle OR – Coleções de Objetos

- Uso do operador TABLE

A função TABLE pode ser usada tanto para consultar uma NESTED TABLE quanto um VARRAY

```
SELECT *
FROM TABLE(SELECT d.lista_fones
           FROM tb_lista_fones_departamento d
           WHERE d.cod_depto = 1);
```

COD_AREA	NUMERO
81	55555555
81	66666666

```
SELECT d.cod_depto, T.*
FROM tb_lista_fones_departamento d, TABLE(d.lista_fones) T;
```

Alias

COD_DEPTO	COD_AREA	NUMERO
1	81	55555555
1	81	66666666

Oracle OR – Coleções de Objetos

- Excluir dados de atributo NESTED TABLE

```
UPDATE tb_lista_fones_departamento D
SET D.lista_fones = NULL
WHERE D.cod_depto = 1 ;
```

Para excluir uma NESTED TABLE atribui-se NULL

1 row updated.

- Consulta tabela tb_lista_fones_departamento

```
SELECT * FROM tb_lista_fones_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
1	

Oracle OR – Coleções de Objetos

- Inserir nova coleção no atributo do tipo nested table

Para inserir novamente valores na NESTED TABLE, esta tem que ser recriada.

```
UPDATE tb_lista_fones_departamento C
SET C.lista_fones = tp_nt_fone (
                    tp_fone('81', '55555555'),
                    tp_fone('81', '66666666'))
WHERE cod_depto = 1 ;
```

1 row updated.

- Consulta tabela tb_lista_fones_departamento

```
SELECT * FROM tb_lista_fones_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
1	TP_NT_FONE(TP_FONE('81', '55555555'), TP_FONE('81', '66666666'))

Retorna objeto do tipo tp_nt_fones

Oracle OR – Coleções de Objetos

- Atualizar alguns valores de um atributo do tipo NESTED TABLE
  - Alterar o código de área do telefone 6666666 do departamento de código 1

Para atualizar apenas alguns valores da NESTED TABLE

Determina a tabela aninhada

```
UPDATE TABLE (
SELECT lista_fones
FROM tb_lista_fones_departamento
WHERE cod_depto = 1) F
SET F.cod_area = '21'
WHERE F.numero = '66666666';
```

1 row updated.

Oracle OR – Coleções de Objetos

- Consulta tabela tb_lista_fones_departamento

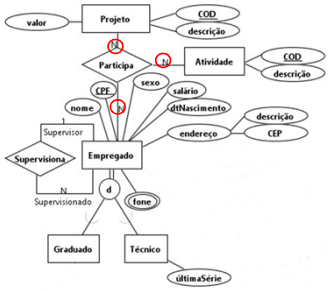
```
SELECT * FROM tb_lista_fones_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
1	TP_NT_FONE(TP_FONE('81', '55555555'), TP_FONE('21', '66666666'))

Retorna objeto do tipo tp_nt_fones

Oracle OR – Coleções de Objetos

- NESTED TABLE DE REFERÊNCIAS – Ideal para relacionamentos múltiplos



Oracle OR – Coleções de Objetos

- NESTED TABLE DE REFERÊNCIAS
  - Redefinir ATIVIDADE

```
CREATE OR REPLACE TYPE tp_atividade AS object (
Cod integer,
descricao varchar2(20)
) NOT FINAL;
/
```

  - Redefinir PROJETO

```
CREATE OR REPLACE TYPE tp_projeto AS OBJECT (
 Cod          INTEGER,
 descricao    VARCHAR2(20),
 valor    NUMBER(12,2)
) NOT FINAL;
/
```


Oracle OR – Coleções de Objetos

- Definir Endereco

```
CREATE OR REPLACE TYPE tp_endereco AS
OBJECT(
  descricao varchar2(30),
  CEP  varchar2(9)
);
/
```


Oracle OR – Coleções de Objetos

- Redefinir Telefone

```
CREATE OR REPLACE TYPE tp_fone AS
OBJECT (
   cod_area     VARCHAR2(2),
   numero       VARCHAR2(8));
/
```

```
CREATE OR REPLACE TYPE tp_fones AS VARRAY(5) OF tp_fone;
/
```


Oracle OR – Coleções de Objetos

- Redefinir EMPREGADO

```
CREATE OR REPLACE TYPE tp_empregado AS
OBJECT(
  CPF varchar2(12),
  nome  varchar2(25),
  sexo char,
  salario number(8,2),
  dtNascimento date,
  endereco tp_endereco,
  fones tp_fones
) NOT FINAL;
/
```

Oracle OR – Coleções de Objetos

- Redefinir GRADUADO e Tecnico

```
CREATE OR REPLACE TYPE tp_graduado UNDER
tp_empregado(
);
/
```

```
CREATE TYPE tp_tecnico UNDER tp_empregado(
  ultimaSerie VARCHAR2(30)
) NOT FINAL;
/
```


Oracle OR – Coleções de Objetos

- Alterar EMPREGADO para incluir o auto relacionamento Supervisor

```
ALTER TYPE tp_empregado
ADD ATTRIBUTE (supervisor REF tp_graduado) CASCADE ;
```

Só pode ser criado após a criação de Graduado

Oracle OR – Coleções de Objetos

- Definição de Tabelas de Objetos

```
CREATE TABLE tb_atividade of tp_atividade
(cod PRIMARY KEY );
```

```
CREATE TABLE tb_projeto OF tp_projeto;
```

```
CREATE TABLE tb_GRADUADO of tp_graduado;
```

```
CREATE TABLE tb_tecnico OF tp_tecnico;
```


Oracle OR – Coleções de Objetos

- Definição do relacionamento ternário Projeto, Atividade e Empregado - Tipo e Tabela

```
CREATE OR REPLACE TYPE tp_relac AS OBJECT(
  projeto REF tp_projeto,
  atividade REF tp_atividade,
  empregado REF tp_empregado
) NOT FINAL;
/
```

```
CREATE TABLE tb_relac OF tp_relac;
```


Oracle OR – Coleções de Objetos

- Inserção de objetos em tb_graduado

```
INSERT INTO tb_graduado VALUES
    (tp_graduado(tp_empregado('132516702-16', 'Ana Paula',
    'F', 12345.00, to_date('10/04/1970', 'dd/mm/yyyy'),
     tp_endereco('R. Janaína, 15','52020-200'), tp_fones( ),
    null)) ));
```

1 row created.

Considerar os dados anteriores de projetos e atividades reinseridos nas respectivas tabelas redefinidas

Oracle OR – Coleções de Objetos

- Inserção de objetos em tb_graduado (Cont.)

```
INSERT INTO tb_graduado VALUES
    (tp_graduado(tp_empregado('215439210-15',
      'Jonas Mota', 'M', 10115.00, to_date('12/03/1975',
      'dd/mm/yyyy'), tp_endereco('R. Sanharó, 32',
     '51020-710'), tp_fones(tp_fone('81', '32712044'),
     tp_fone('21', '32295968')),
    (SELECT REF(G) FROM tb_graduado G WHERE
      cpf ='132516702-16') )));
```

1 row created.

Supervisor Ana Paula

Oracle OR – Coleções de Objetos

- Inserção de objetos em tb_graduado (Cont.)

```
INSERT INTO tb_graduado VALUES
        (tp_graduado(tp_empregado('420316123-45',
          'Helena Ramos', 'F', 8500.00, to_date('21/11/1982',
           'dd/mm/yyyy'), tp_endereco('R. Roriz, 100',
        '50135-316'), null, (SELECT REF(G) FROM tb_graduado G
WHERE cpf ='132516702-16') )));
```

1 row created.

Supervisor Ana Paula

- Consulta objetos em tb_graduado

```
SELECT * FROM tb_graduado;
```


CPF	NOME	Sexo	SALARIO	DTNASCIM	ENDERECO(DESCRICAO, CEP)	FONES(COD_AREA, NUMERO)	SUPERVISOR
132516702-16	Ana Paula	F	12345	10/04/70	TP_ENDERECO('R. Janaína, 15', '52020-200')	TP_FONES()	
215439210-15	Jonas Mota	M	10115	12/03/75	TP_ENDERECO('R. Sanharó, 32', '51020-710')	TP_FONES(TP_FONE('81', '32712044'), TP_FONE('21', '32295968'))	000022020877E4CF200969468294EEB9D9639C7A7C004A59C6F6AA40EFBFC9951C7F06A3B8
420316123-45	Helena Ramos	F	8500	21/11/82	TP_ENDERECO('R. Roriz, 100', '50135-316')		000022020877E4CF200969468294EEB9D9639C7A7C004A59C6F6AA40EFBFC9951C7F06A3B8

Oracle OR – Coleções de Objetos

- Inserção de objetos em tb_tecnico (Cont.)

```
INSERT INTO tb_tecnico VALUES
        (tp_tecnico(tp_empregado('813509123-35',
          'Márcia Rocha', 'F', 9200.00, to_date('13/08/1976',
           'dd/mm/yyyy'), tp_endereco('R. Andaluzia, 1245',
'51005-356'), null, (SELECT REF(G) FROM tb_graduado G
WHERE cpf ='420316123-45') , 'Terceiro') ));
```

1 row created.

Supervisor Helena Ramos

Oracle OR – Coleções de Objetos

- Inserção de objetos em tb_tecnico (Cont.)

```
INSERT INTO tb_tecnico VALUES
        (tp_tecnico(tp_empregado('515422936-18',
          'Marcos Lessa', 'M', 7934.00, to_date('19/10/1986',
           'dd/mm/yyyy'), tp_endereco('R. Baronesa Léa, 15',
         '50540-930'), tp_fones(tp_fone('83', '63502143'),
         tp_fone('11', '98764592')),
        (SELECT REF(G) FROM tb_graduado G  WHERE
          cpf ='420316123-45')) , 'Quinto')));
```

1 row created.

Supervisor Helena Ramos

- Consulta objetos em tb_tecnico

```
SELECT * FROM tb_tecnico;
```


Oracle OR – Coleções de Objetos

CPF	NOME	Sexo	SALARIO	DTNASCIM	ENDERECO (DESCRICAO, CEP)	FONES(COD_AREA, NUMERO)	SUPERVISOR	ULTIMASERIE
813509123-35	Márcia Rocha	F	9200	13/08/76	TP_ENDERECO('R. Andaluzia, 1245', '51005-356')		000022020899E4CF200969468294EEB9D9639C7A7C004A59C6F6AA40EFBFC9951C7EA1A3B8	Terceiro
515422936-18	Marcos Lessa	M	7934	19/10/86	TP_ENDERECO('R. Baronesa Léa, 15', '50540-930')	TP_FONES(TP_FONE('83', '63502143'), TP_FONE('11', '98764592'))	000022020899E4CF200969468294EEB9D9639C7A7C004A59C6F6AA40EFBFC9951C7EA1A3B8	Quinto

Oracle OR – Coleções de Objetos

- Inserção de objetos no relacionamento ternário Projeto, Atividade e Empregado

```
INSERT INTO tb_relac VALUES
        (tp_relac((SELECT REF(P) FROM tb_projeto P  WHERE
          cod =1), (SELECT REF(A) FROM tb_atividade A
          WHERE cod =1) , (SELECT REF(G) FROM tb_graduado
          G  WHERE cpf ='420316123-45') ));
```

1 row created.

No projeto Requisitos, a atividade Analista é exercida pelo empregado Helena Ramos

Oracle OR – Coleções de Objetos

- Inserção de objetos no relacionamento ternário Projeto, Atividade e Empregado (Cont.)

```
INSERT INTO tb_relac VALUES
    (tp_relac((SELECT REF(P) FROM tb_projeto P
      WHERE cod =2), (SELECT REF(A) FROM tb_atividade A
      WHERE cod =3) , (SELECT REF(G) FROM tb_graduado
      G  WHERE cpf ='420316123-45') ));
```

1 row created.

No projeto Diagramas, a atividade Programador é exercida pelo empregado Helena Ramos

Oracle OR – Coleções de Objetos

- Inserção de objetos no relacionamento ternário Projeto, Atividade e Empregado (Cont.)

```
INSERT INTO tb_relac VALUES
    (tp_relac((SELECT REF(P) FROM tb_projeto P  WHERE
      cod =3), (SELECT REF(A) FROM tb_atividade A
      WHERE cod =2), (SELECT REF(G) FROM tb_tecnico G
      WHERE cpf ='813509123-35') ));
```

1 row created.

No projeto Codificação, a atividade Administrador é exercida pelo empregado Márcia Rocha

- Consulta objetos no relacionamento ternário tb_relac

```
SELECT * FROM tb_relac;
```


PROJETO	ATIVIDADE	EMPREGADO
0000220208B3DBBF5627684645AB5E920AA79B19BE2C9FC909B3FA45B988F2CAB9371487CB	0000220208EF627EEDB9FC4ED58F63BE8EE3191C3E30055CEA4B704400BA9C8C85FB8321E4	0000220208993FA804AE4E4A33B57E6A8CCF323C2E004A59C6F6AA40EFBFC9951C7F06A3B8
0000220208565CEA598E0F4B01B01DA700B255B46A2C9FC909B3FA45B988F2CAB9371487CB	0000220208E9B61A54AC464E6DBBF32DF7964DF22C30055CEA4B704400BA9C8C85FB8321E4	0000220208993FA804AE4E4A33B57E6A8CCF323C2E004A59C6F6AA40EFBFC9951C7F06A3B8
0000220208533E144D974C4EC088E8029DF7BE2DFA2C9FC909B3FA45B988F2CAB9371487CB	00002202085CAFAC6F8FED44CAB9E1F913F046DE3F30055CEA4B704400BA9C8C85FB8321E4	0000280209209C051BD77246B38231736B2D438A2940A46FC71F594CCDB6F9FCDEA83E31B7010002470000

Oracle OR – Coleções de Objetos

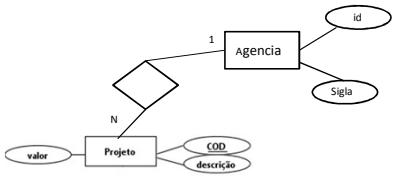
- Informar a descrição do projeto, descrição da atividade realizada em cada projeto e o nome do empregado que a realiza

```
SELECT  r.projeto.descricao, r.atividade.descricao,
r.empregado.nome FROM tb_relac r;
```

PROJETO.DESCRICAO	ATIVIDADE.DESCRICAO	EMPREGADO.NOME
Requisitos	Analista	Helena Ramos
Diagramas	Programador	Helena Ramos
Codificação	Administrador	Márcia Rocha

Oracle OR – Coleções de Objetos

- Considerando que projetos podem ser financiados por agências de fomento (CNPq, CAPES, FACEPE, FINEPE,...), incluir AGENCIA no modelo exemplo, conforme o diagrama



Oracle OR – Coleções de Objetos

- Definição do tipo de objeto do relacionamento entre AGENCIA e PROJETO

```
CREATE OR REPLACE TYPE tp_ref_relac AS OBJECT(
  projeto REF tp_projeto) NOT FINAL;
/
```

- Definição do tipo de objeto do relacionamento do lado N entre AGENCIA e PROJETO

```
CREATE TYPE tp_nt_ref_relac  AS TABLE OF tp_ref_relac;
/
```

Oracle OR – Coleções de Objetos

- Definição do tipo de objeto AGENCIA

```
CREATE OR REPLACE TYPE tp_agencia AS OBJECT (
  id          INTEGER,
  sigla       VARCHAR2(20),
  Projetos    tp_nt_ref_relac
) NOT FINAL;
/
```

- Definição da tabela de objetos tb_agencia

```
CREATE TABLE tb_agencia  OF tp_agencia
NESTED TABLE Projetos STORE AS lista_projetos;
```


Oracle OR – Coleções de Objetos

- Inserção de objetos em tb_agencia

Inicializa a Tabela Aninhada

```
INSERT INTO tb_agencia VALUES
        (tp_agencia(1, 'CAPES', tp_nt_ref_relac( )));
```

1 row created.

```
INSERT INTO tb_agencia VALUES
        (tp_agencia(2, 'CNPq', tp_nt_ref_relac( )));
```

1 row created.

```
INSERT INTO tb_agencia VALUES
        (tp_agencia(3, 'FACEPE', tp_nt_ref_relac( )));
```

1 row created.

```
INSERT INTO tb_agencia VALUES
        (tp_agencia(4, 'FINEPE', tp_nt_ref_relac( )));
```

1 row created.

Oracle OR – Coleções de Objetos

- Consulta objetos de tb_agencia

```
SELECT * FROM tb_agencia;
```

ID	SIGLA	PROJETOS(PROJETO)
1	CAPES	TP_NT_REF_RELAC()
2	CNPq	TP_NT_REF_RELAC()
3	FACEPE	TP_NT_REF_RELAC()
4	FINEPE	TP_NT_REF_RELAC()

Oracle OR – Coleções de Objetos

- Inserir os projetos Codificação (3) e Requisitos (1) para a agência FACEPE (3)

```
UPDATE tb_agencia A
SET A.projetos  = tp_nt_ref_relac (
                  tp_ref_relac((SELECT REF(P) FROM
tb_projeto P  WHERE
          P.cod =3)), tp_ref_relac((SELECT REF(P)
FROM tb_projeto P  WHERE
          P.cod =1)))
WHERE id = 3 ;
```

1 row updated.

Oracle OR – Coleções de Objetos

- Inserir o projeto Diagramas (2) para a agência CAPES (1)

```
UPDATE tb_agencia A
SET A.projetos  = tp_nt_ref_relac (
                  tp_ref_relac((SELECT REF(P)
FROM tb_projeto P  WHERE
          P.cod =2)))
WHERE id = 1 ;
```

1 row updated.

Oracle OR – Coleções de Objetos

- Consulta objetos em tb_agencia

```
SELECT * FROM tb_agencia;
```

ID	SIGLA	PROJETOS(PROJETO)
1	CAPES	TP_NT_REF_RELAC(TP_REF_RELAC(0000220208F9603B1CCEEED1C9E04015AC0702177BF9603B1CCEECD1C9E04015AC0702177B))
2	CNPq	TP_NT_REF_RELAC()
3	FACEPE	TP_NT_REF_RELAC(TP_REF_RELAC(0000220208F9603B1CCEEFD1C9E04015AC0702177BF9603B1CCEECD1C9E04015AC0702177B), TP_REF_RELAC(0000220208F9603B1CCEEDD1C9E04015AC0702177BF9603B1CCEECD1C9E04015AC0702177B))
4	FINEPE	TP_NT_REF_RELAC()

Oracle OR – Coleções de Objetos

- Informar os nomes e valores dos projetos financiados pela CAPES

```
SELECT T.projeto.descricao, T.projeto.valor
FROM TABLE(SELECT a.projetos
           FROM tb_agencia a
           WHERE a.id = 1)T;
```

Agência CAPES

PROJETO.DESCRICAO	PROJETO.VALOR
Diagramas	50900

Oracle OR – Coleções de Objetos

- Operações em elementos específicos de coleções
  - Tabelas aninhadas - tb_lista_fone_departamento

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
1	TP_NT_FONE(TP_FONE('81', '55555555'), TP_FONE('21', '66666666'))

- Inserção de novo elemento na tabela aninhada

```
INSERT INTO TABLE (<select para determinar a tabela aninhada>)
VALUES (<valores componentes do elemento>);
```


Oracle OR – Coleções de Objetos

- Inserção de novo elemento na tabela aninhada lista_fone
  - Inserir o telefone com código 83 e número 99999999 do departamento de código 1

Determina a tabela aninhada corrente

```
INSERT INTO TABLE(SELECT l.lista_fone
           FROM tb_lista_fone_departamento l
           WHERE l.cod_depto = 1)
  VALUES ('83', '99999999');
```


Oracle OR – Coleções de Objetos

- Consultando a tabela

```
SELECT * FROM tb_lista_fone_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
1	TP_NT_FONE(TP_FONE('81', '55555555'), TP_FONE('21', '66666666') , tp_fone('83', '99999999'))

Novo telefone inserido

Oracle OR – Coleções de Objetos

- Atualização de elemento
  - Atualizar o número do telefone 55555555 para 44444444 no departamento 1 – Utilização da função VALUE( )

Atualiza o telefone indicado

Determina a tabela aninhada corrente

```
UPDATE TABLE(SELECT l. lista_fone
           FROM tb_lista_fone_departamento l
           WHERE l.cod_depto = 1) e
  SET VALUE(e) = tp_fone('81', '44444444')
  WHERE e.numero = '55555555';
```


Oracle OR – Coleções de Objetos

- Consultando a tabela

```
SELECT * FROM tb_lista_fone_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
1	TP_NT_FONE(TP_FONE('81', '44444444'), TP_FONE('21', '66666666') , tp_fone('83', '99999999'))

Telefone 55555555 atualizado para 444444444

Oracle OR – Coleções de Objetos

- Remoção de elemento
  - Remover o telefone de número 66666666

Indica o telefone a remover

Determina a tabela aninhada corrente

```
DELETE FROM TABLE(SELECT l. lista_fone
                  FROM tb_lista_fone_departamento l
                  WHERE l.cod_depto = 1) e
  WHERE e.numero =  '66666666';
```


Oracle OR – Coleções de Objetos

- Consultando a tabela

```
SELECT * FROM tb_lista_fone_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
1	TP_NT_FONE(TP_FONE('81', '44444444'), tp_fone('83', '99999999'))

Telefone 66666666 removido

Oracle OR – Coleções de Objetos

- Varrays - tb_fones_departamento

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
100	TP_FONES(TP_FONE('81', '22222222'), TP_FONE('81', '33333333'), TP_FONE('81', '44444444'))

- Embora concebidos para serem manipulados como um todo (coleção densa), é possível manipular elementos específicos, exceto realizar remoções
  - Uso de PL – Blocos, Procedures ou Functions
    - Necessário trabalhar com variável auxiliar do tipo do Varray

Oracle OR – Coleções de Objetos

- Consulta a elementos específicos
  - Qual o segundo telefone do Departamento 100?

Variável do tipo do Varray

```
Declare
n tb_fones_departamento.lista_fones%type;
Begin
Select d.lista_fones into n from tb_fones_departamento d
    where d.cod_depto = 100;
Dbms_output.put_line('Segundo Telefone = '|| n(2).cod_area
    || '-' || n(2).numero);
End;
/
```

Consulta o segundo elemento

Copia os dados do Varray da tabela para a variável

Oracle OR – Coleções de Objetos

- Resposta obtida

```
Segundo Telefone = 81-33333333

PL/SQL procedure successfully completed.
```

- Acrescentar o telefone 21-22222222 ao conjunto de telefones do Departamento 100

Oracle OR – Coleções de Objetos

```
Declare
n tb_fones_departamento.lista_fones%type;
i integer;
Begin
Select d.lista_fones into n from tb_fones_departamento d
  where d.cod_depto = 100;
n.extend;
i := n.count;
n(i) := tp_fone('21', '22222222');
Update tb_fones_departamento d set  d.lista_fones = n
  where d.cod_depto = 100;
End;
/
```

Copia varray para variável auxiliar

Aumenta tamanho do varray

Conta elementos do varray

Insere novo telefone no novo elemento

Atualiza varray na tabela

Oracle OR – Coleções de Objetos

- Consulta tabela tb_fones_departamento

```
SELECT * FROM tb_fones_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
100	TP_FONES(TP_FONE('81', '22222222'), TP_FONE('81', '33333333'), TP_FONE('81', '44 444444'), TP_FONE('84', '99999999'), TP_FONE('21', '22222222'))

Novo telefone inserido

Oracle OR – Coleções de Objetos

- Quantos telefones tem o Departamento 100?

```
Declare
n tb_fones_departamento.lista_fones%type;
Begin
Select d.lista_fones into n from tb_fones_departamento d
   where d.cod_depto = 100;
Dbms_output.put_line('QTDE = '|| n.count);
End;
/
```

Copia varray para variável auxiliar

Conta elementos do varray

Oracle OR – Coleções de Objetos

- Resposta obtida

```
QTDE = 5

PL/SQL procedure successfully completed.
```

- Alterar o número do quarto telefone do departamento 100 para 32712042

Oracle OR – Coleções de Objetos

```
Declare
n tb_fones_departamento.lista_fones%type;
i integer;
Begin
Select d.lista_fones into n from tb_fones_departamento d
   where d.cod_depto = 100;
n(4).numero := '32712042';
Update tb_fones_departamento d set  d.lista_fones = n
  where d.cod_depto = 100;
End;
/
```

Copia varray para variável auxiliar

Altera número do quarto telefone

Atualiza varray na tabela

Oracle OR – Coleções de Objetos

- Consulta tabela tb_fones_departamento

```
SELECT * FROM tb_fones_departamento;
```

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
100	TP_FONES(TP_FONE('81', '22222222'), TP_FONE('81', '33333333'), TP_FONE('81', '44 444444'), TP_FONE('84', '32712042'), TP_FONE('21', '22222222'))

Quarto telefone atualizado para 32712042

Oracle OR – Coleções de Objetos

- Alterar o número do telefone 33333333 para 63333333

```
Declare
n tb_fones_departamento. lista_fones%type;
i integer := 0;
Begin
Select d.lista_fones into n from tb_fones_departamento d where d.cod_depto = 100;
For m in 1..n.count loop
If n(m).numero = '33333333' then
i := m;
Exit;
End if;
End loop;
If  i <> 0 then
n(i).numero := '63333333';
Update tb_fones_departamento d set  d.lista_fones = n where d.cod_depto = 100;
End if;
End;
/
```

Copia varray para variável auxiliar

Determinar a posição do telefone 33333333 no VARRAY

Se existir, altera número achado

Atualiza varray na tabela

Oracle OR – Coleções de Objetos

- Consulta tabela tb_fones_departamento

SELECT * FROM tb_fones_departamento;

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
100	TP_FONES(TP_FONE('81', '22222222'), TP_FONE('81', '63333333'), TP_FONE('81', '44444444'), TP_FONE('84', '32712042'), TP_FONE('21', '22222222'))

Telefone 33333333 atualizado para 63333333

Oracle OR – Coleções de Objetos

- Remover o terceiro telefone do Departamento 100
  - Varrays são densos, portanto, não se pode remover elementos individualmente, exceto o último

```
Declare
n tb_fones_departamento.lista_fones%type;
Begin
Select d.lista_fones into n from tb_fones_departamento d
   where d.cod_depto = 100;
For i in 3.. n.count-1 loop
n(i) := n(i+1);
End loop;
n.trim;
Update tb_fones_departamento d set d.lista_fones = n
      where d.cod_depto = 100;
End;
/
```

Copia varray para variável auxiliar

Copia para a posição 3 até a penúltima, o elemento que está na posição da frente

Remove o último elemento

Atualiza varray na tabela

Oracle OR – Coleções de Objetos

- Consulta tabela tb_fones_departamento

SELECT * FROM tb_fones_departamento;

COD_DEPTO	LISTA_FONES(COD_AREA, NUMERO)
100	TP_FONES(TP_FONE('81', '22222222'), TP_FONE('81', '63333333'), TP_FONE('84', '32712042'), TP_FONE('21', '22222222'))

Telefone 44444444 removido

Coleções Multinível

Oracle OR – Coleções Multinível

- São tipos de coleção cujos elementos são eles próprios, direta ou indiretamente, outro tipo de coleção
  - Tabela aninhada com elementos de tipo tabela aninhada
  - Tabela aninhada com elementos de tipo varray
  - Varray com elementos de tipo tabela aninhada
  - Varray com elementos de tipo varray
  - Tabela aninhada ou varray de um tipo definido pelo usuário que tem um atributo que é uma tabela aninhada ou varray

Oracle OR – Coleções Multinível

- Tabela aninhada com um atributo do tipo tabela aninhada
  - Além de ter os telefones de cada Departamento armazenados em tabelas aninhadas, considerar que para cada telefone, a companhia precisa também armazenar os números dos seus ramais e os responsáveis por eles em tabelas aninhadas
    - Remover tabela e tipos do sistema

```
DROP TABLE tb_fones_departamento;

DROP TYPE tp_nt_fones;

DROP TYPE tp_fone;
```

Oracle OR – Coleções Multinível

- Criar novos tipos e tabela
  - Criação do tipo Ramais

```
CREATE OR REPLACE TYPE tp_ramais AS OBJECT (
   ramal          VARCHAR2(4),
   responsavel          VARCHAR2(15));
/
```

  - Criação do tipo coleção de ramais como tabela aninhada

```
CREATE TYPE tp_nt_ramais AS TABLE OF tp_ramais;
/
```


Oracle OR – Coleções Multinível

- Criar novos tipos e tabela (Cont.)
  - Criação do tipo Telefone

```
CREATE OR REPLACE TYPE tp_fones AS OBJECT (
Cod_area VARCHAR2(2),
Numero VARCHAR2(8),
Ramais tp_nt_ramais);
/
```

  - Criação do tipo coleção de telefones como tabela aninhada

```
CREATE TYPE tp_nt_fones AS TABLE OF tp_fones;
/
```


Oracle OR – Coleções Multinível

- Criar novos tipos e tabela (Cont.)
  - Criação do tipo Departamentos

Tabela aninhada em outra tabela aninhada

```
CREATE OR REPLACE TYPE tp_departamentos AS OBJECT (
Cod_depto VARCHAR2(5),
Telefones  tp_nt_fones);
/
```

  - Criação da tabela Fones_Departamento

```
CREATE TABLE tb_fones_departamento OF tp_departamentos (
Cod_depto PRIMARY KEY)
NESTED TABLE telefones STORE AS tb_telefones
(NESTED TABLE ramais STORE AS tb_ramais);
```


Oracle OR – Coleções Multinível

- Inserção de Dados

Departamento	Fones	Ramal	Responsável
100	(81) 22222222	3333	Luiz Carlos
		1111	Ana Paula
		7777	Maria Carla
	(81) 33333333		
	(81) 44444444		

```
INSERT INTO tb_fones_departamento VALUES ('100',
 tp_nt_fones (tp_fones('81', '22222222', tp_nt_ramais(tp_ramais('3333', 'Luiz Carlos'),
 tp_ramais('1111', 'Ana Paula'), tp_ramais('7777', 'Maria Carla'))),
              tp_fones('81', '33333333', tp_nt_ramais( ) ),
             tp_fones('81', '44444444', tp_nt_ramais( ))));
```

1 row inserted.

Oracle OR – Coleções Multinível

- Consultado a tabela

```
SELECT * FROM tb_fones_departamento;
```

COD_D	TELEFONES(COD_AREA, NUMERO, RAMAIS(RAMAL, RESPONSAVEL))
100	TP_NT_FONES(TP_FONES('81', '22222222', TP_NT_RAMAIS(TP_RAMAIS('3333', 'Luiz Carlos'), TP_RAMAIS('1111', 'Ana Paula'), TP_RAMAIS('7777', 'Maria Carla'))), TP_FON ES('81', '33333333', TP_NT_RAMAIS()), TP_FONES('81', '44444444', TP_NT_RAMAIS()))

Manipulação de Coleções Multinível

Oracle OR – Coleções Multinível

- Informe os ramais e nomes dos responsáveis para o telefone 22222222 do Departamento 100.

```
SELECT l.ramal AS Ramal, l.responsavel AS Contato
FROM tb_fones_departamento r, TABLE(r.telefones) c,
TABLE(c.ramais) l WHERE c.numero = '22222222';
```

RAMA	CONTATO
3333	Luiz Carlos
1111	Ana Paula
7777	Maria Carla

Junção entre a tabela e suas tabelas aninhadas

Oracle OR – Coleções Multinível

- Informe os ramais e nomes dos responsáveis para o telefone 63333333 do Departamento 100.

```
SELECT l.ramal AS Ramal, l.responsavel AS Contato
FROM tb_fones_departamento r, TABLE(r.telefones) c,
TABLE(c.ramais) l WHERE c.numero = '63333333';
```

no rows selected

Oracle OR – Coleções Multinível

- Incluir o ramal 4142 com responsável Sophia Loren para o telefone (81) 22222222 do Departamento 100

```
INSERT INTO TABLE( SELECT r.ramais
FROM TABLE( SELECT l.telefones FROM tb_fones_departamento l
WHERE l.cod_depto = '100') r
WHERE r.cod_area = '81' AND r.numero = '22222222')
VALUES ('4142', ' Sophia Loren');
```

1 row inserted.

Definir a tabela aninhada para inserção

Oracle OR – Coleções Multinível

- Consultado a tabela

```
SELECT * FROM tb_fones_departamento;
```

COD_D	TELEFONES(COD_AREA, NUMERO, RAMAIS(RAMAL, RESPONSAVEL))
100	TP_NT_FONES(TP_FONES('81', '22222222', TP_NT_RAMAIS(TP_RAMAIS('3333', 'Luiz Carlos'), TP_RAMAIS('1111', 'Ana Paula'), TP_RAMAIS('7777', 'Maria Carla'), TP_RAMAIS('4142', ' Sophia Loren'))), TP_FONES('81', '33333333', TP_NT_RAMAIS()), TP_FONES('81', '44444444', TP_NT_RAMAIS())

Ramal incluído

Oracle OR – Coleções Multinível

- Incluir o ramal 5555 com responsável Brigitte Bardot para o telefone (81) 22222222 do Departamento 100

```
INSERT INTO TABLE( SELECT r.ramais
FROM TABLE( SELECT l.telefones FROM tb_fones_departamento l
WHERE l.cod_depto = '100') r
WHERE r.cod_area = '81' AND r.numero = '22222222')
VALUES ('5555', 'Brigitte Bardot');
```

1 row inserted.

Oracle OR – Coleções Multinível

- Consultado a tabela

```
SELECT * FROM tb_fones_departamento;
```

COD_D	TELEFONES(COD_AREA, NUMERO, RAMAIS(RAMAL, RESPONSAVEL))
100	TP_NT_FONES(TP_FONES('81', '22222222', TP_NT_RAMAIS(TP_RAMAIS('3333', 'Luiz Carlos'), TP_RAMAIS('1111', 'Ana Paula'), TP_RAMAIS('7777', 'Maria Carla'), TP_RAMAIS('4142', ' Sophia Loren'), TP_RAMAIS('5555', 'Brigitte Bardot'))), TP_FONES('81', '33333333', TP_NT_RAMAIS()), TP_FONES('81', '44444444', TP_NT_RAMAIS())

Ramal incluído

Oracle OR – Coleções Multinível

- Qual o responsável pelo Ramal 3333 do telefone (81) 22222222 do Departamento 100?

```
SELECT r.responsavel FROM TABLE (SELECT m.ramais
FROM TABLE( SELECT l.telefones FROM tb_fones_departamento l
WHERE l.cod_depto = '100') m
WHERE m.cod_area = '81' AND m.numero = '22222222') r
WHERE r.ramal = '3333';
```

RESPONSAVEL
Luiz Carlos

Oracle OR – Coleções Multinível

- Remova o ramal 1111 do telefone (81) 2222222 do Departamento 100

```
DELETE FROM TABLE (SELECT r.ramais FROM
TABLE (SELECT l.telefones FROM tb_fones_departamento l
WHERE l.cod_depto = '100') r WHERE r.cod_area = '81' AND
r.numero = '22222222') m WHERE m.ramal = '1111';
```

1 row deleted.

Oracle OR – Coleções Multinível

- Consultando a tabela

```
SELECT * FROM tb_fones_departamento;
```

COD_D	TELEFONES(COD_AREA, NUMERO, RAMAIS(RAMAL, RESPONSAVEL))
100	TP_NT_FONES(TP_FONES('81', '22222222', TP_NT_RAMAIS(TP_RAMAIS('3333', 'Luiz Carlos'), TP_RAMAIS('7777', 'Maria Carla'), TP_RAMAIS('4142', ' Sophia Loren'), TP_RAMAIS('5555', 'Brigitte Bardot'))), TP_FONES('81', '33333333', TP_NT_RAMAIS()), TP_FONES('81', '44444444', TP_NT_RAMAIS()))

Ramal 1111 foi removido.

Oracle OR – Coleções Multinível

- Atualizar a responsável pelo Ramal 7777 do telefone (81) 22222222 do Departamento 100 para Carla Bruni

```
UPDATE TABLE (SELECT r.ramais FROM
TABLE (SELECT l.telefones FROM tb_fones_departamento l
WHERE l.cod_depto = '100') r WHERE r.cod_area = '81' AND
r.numero = '22222222') m
SET m.responsavel = 'Carla Bruni' WHERE m.ramal = '7777';
```

1 row updated.

Oracle OR – Coleções Multinível

- Consultado a tabela

```
SELECT * FROM tb_fones_departamento;
```

COD_D	TELEFONES(COD_AREA, NUMERO, RAMAIS(RAMAL, RESPONSAVEL))
100	TP_NT_FONES(TP_FONES('81', '22222222', TP_NT_RAMAIS(TP_RAMAIS('3333', 'Luiz Carlos'), TP_RAMAIS('7777', 'Carla Bruni'), TP_RAMAIS('4142', ' Sophia Loren'), TP_RAMAIS('5555', 'Brigitte Bardot'))), TP_FONES('81', '33333333', TP_NT_RAMAIS()), TP_FONES('81', '44444444', TP_NT_RAMAIS()))

Responsável do Ramal 7777 alterada.

Oracle OR – Coleções Multinível

- Informar os ramais e respectivos responsáveis do telefone 81 22222222 do Departamento de Código 100

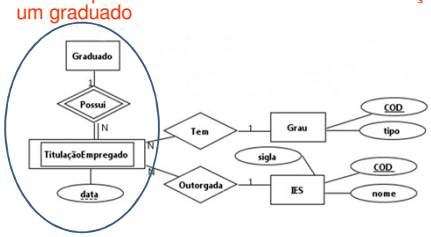
```
SELECT p.ramal AS RAMAL, p.responsavel AS RESPONSAVEL
FROM tb_fones_departamento l, TABLE (l.telefones) r,
TABLE(r.ramais) p WHERE l.cod_depto = '100' AND r.cod_area = '81'
AND r.numero = '22222222';
```

RAMA	RESPONSAVEL
3333	Luiz Carlos
7777	Carla Bruni
4142	Sophia Loren
5555	Brigitte Bardot

Exercícios Extras

Oracle OR

- Modelar os elementos do diagrama abaixo, definindo
  - Método que retorna titulação mais recente de um graduado
  - Método para determinar o número de titulações de um graduado



Oracle OR

- Criar tipos que serão utilizados na definição de outros tipos
  - Entidade GRAU

```
CREATE OR REPLACE TYPE tp_grau AS OBJECT(
  Cod integer,
  tipo  varchar2(20)
) ;
/
```


Oracle OR

- Criar tipos que serão utilizados na definição de outros tipos (Cont.)
  - Entidade IES

```
CREATE OR REPLACE TYPE tp_ies AS OBJECT(
  cod integer,
  nome  varchar2(40),
  sigla varchar2(10)
) ;
/
```


Oracle OR

- Criar tipos que serão utilizados na definição de outros tipos (Cont.)
  - Entidade TitulacaoEmpregado

```
CREATE OR REPLACE TYPE tp_titulacaoempregado AS OBJECT(
  tem REF tp_grau,
  outorgada REF tp_ies,
  data date
);
/
```


Oracle OR

- Criar tipos que serão utilizados na definição de outros tipos (Cont.)
  - Relacionamento possui

```
CREATE OR REPLACE TYPE tp_possui  AS TABLE OF
tp_titulacaoempregado;
/
```

Oracle OR

- Redefinir a entidade Graduado
  - Remover a tabela tb_graduado

```
DROP TABLE tb_GRADUADO;
```

  - Remover o tipo tp_graduado que faz parte da definição do Tipo tp_empregado

```
DROP TYPE tp_GRADUADO force;
```

  - Remover o tipo tp_empregado que ficou inválido devido à remoção de tp_graduado

```
DROP TYPE tb_empregado;
```


Oracle OR

- Redefinir tp_empregado

```
CREATE OR REPLACE TYPE tp_empregado AS OBJECT(
  CPF varchar2(12),
  nome  varchar2(25),
  sexo char,
  salario number(8,2),
  dtNascimento date,
  endereco tp_endereco,
  fones tp_fones,
  MEMBER FUNCTION salarioAnual RETURN NUMBER,
  ORDER MEMBER FUNCTION comparaSalario (X
          tp_empregado) RETURN INTEGER
) NOT FINAL;
/
```

Corpo a ser criado do tipo é o definido no slide 43

Oracle OR

- Redefinir tp_graduado

```
CREATE OR REPLACE TYPE tp_graduado UNDER tp_empregado(
possui_tit  tp_possui,
MEMBER FUNCTION titulacaoRecente RETURN REF tp_grau,
MEMBER FUNCTION numeroTitulos  RETURN INTEGER
);
/
```


Oracle OR

- Definir corpo do tipo tp_graduado

```
CREATE OR REPLACE TYPE BODY tp_graduado AS
MEMBER FUNCTION titulacaoRecente  RETURN  REF tp_grau IS
j tp_titulacaoempregado;
i binary_integer;
l binary_integer;
BEGIN
i := 1;
j := self.possui_tit(i);
IF j IS NOT NULL
THEN
WHILE self.possui_tit(i+1) IS NOT  NULL
LOOP
   IF self.possui_tit(i+1).data  > j.data THEN
   j := self.possui_tit(i+1);
   i := i+1;
   l := i;
   END IF;
```


Oracle OR

- Definir corpo do tipo tp_graduado (Cont.)

```
END LOOP;
ELSE
l := 0;
END IF;
IF l = 0 THEN RETURN null;
ELSE
RETURN  j.tem;
END IF;
END;
MEMBER FUNCTION numeroTitulos RETURN INTEGER IS
 BEGIN
    IF  self.possui_tit  IS NOT NULL
    THEN
     RETURN self.possui_tit.count(  );
     ELSE RETURN 0;
     END IF;
  END;
  END;
/
```


Oracle OR

- Alterar definição de tp_empregado para incluir atributo Supervisor

```
ALTER TYPE tp_empregado
ADD ATTRIBUTE (supervisor REF tp_graduado) CASCADE ;
```

- Criar tabelas para objetos dos tipos tp_grau, tp_ies e tp_graduado

```
CREATE TABLE tb_grau of tp_grau;
```

```
CREATE TABLE tb_ies of tp_ies;
```

```
CREATE TABLE tb_graduado of tp_graduado
NESTED TABLE possui_tit  STORE AS lista_titulos;
```

Oracle OR

- Inserir objetos do tipo tp_grau

```
INSERT INTO tb_grau VALUES (tp_grau(1, 'Engenharia Civil'));
```

1 row created.

```
INSERT INTO tb_grau VALUES (tp_grau(2, 'Administração'));
```

1 row created.

```
INSERT INTO tb_grau VALUES (tp_grau(3, 'Economia '));
```

1 row created.

```
INSERT INTO tb_grau VALUES (tp_grau(4, 'Computação'));
```

1 row created.

Oracle OR

- Inserir objetos do tipo tp_ies

```
INSERT INTO tb_ies VALUES (tp_ies(1, 'Universidade Católica de Pernambuco', 'UNICAP'));
```

1 row created.

```
INSERT INTO tb_ies VALUES (tp_ies(2, 'Universidade Federal de Pernambuco', 'UFPE'));
```

1 row created.

```
INSERT INTO tb_ies VALUES (tp_ies(3, 'Universidade Federal Rural de Pernambuco', 'UFRPE'));
```

1 row created.

```
INSERT INTO tb_ies VALUES (tp_ies(4, 'Universidade Federal de Tocantins', 'UFTO'));
```

1 row created.

Oracle OR

- Inserir os graduados com os dados abaixo

Graduado	Fones	Supervisor	Titulação
420316123-45, Helena Ramos, F, 8500.00, 21/11/1982, R. Roriz, 100, 50135-316			Engenharia Civil, UFPE, 12/11/2000 Administração, UNICAP, 05/03/2003 Computação, UFTO, 10/01/2008
215439210-15, Jonas Mota, M, 7115.00, 12/03/1975, R. Sanharó, 32, 51020-710		Helena Ramos	Administração, UFTO, 12/11/1999

Oracle OR

- Inserir os graduados com os dados abaixo (Cont.)

Graduado	Fones	Supervisor	Titulação
132516702-16, Ana Paula, F, 6345.00, 10/04/1970, R. Janaína, 15, 52020-200		Jonas Mota	Economia, UNICAP, 20/12/2000 Engenharia, UFTO, 10/06/2004

Oracle OR

- Inserir objetos do tipo tp_graduado

```
INSERT INTO tb_graduado VALUES
        (tp_graduado('420316123-45',
          'Helena Ramos', 'F', 8500.00, to_date('21/11/1982',
          'dd/mm/yyyy'), tp_endereco('R. Roriz, 100',
        '50135-316'), null, null, tp_possui(tp_titulacaoempregado(
         (SELECT REF(G) FROM tb_grau G WHERE g.cod = 1), (SELECT
         REF(I) FROM tb_ies I WHERE i.cod =2), to_date('12/11/2000'
          , 'dd/mm/yyyy')), tp_titulacaoempregado((SELECT REF(G)
        FROM tb_grau G WHERE g.cod = 2), (SELECT REF(I) FROM
        tb_ies I WHERE i.cod = 1), to_date('05/03/2003',
          'dd/mm/yyyy') ), tp_titulacaoempregado((SELECT REF(G)
        FROM tb_grau G WHERE g.cod =4), (SELECT REF(I) FROM
        tb_ies I WHERE i.cod =4), to_date('10/01/2008',
          'dd/mm/yyyy') )));
```


Oracle OR

- Inserir objetos do tipo tp_graduado (Cont.)

```
INSERT INTO tb_graduado VALUES
        (tp_graduado('215439210-15',
          'Jonas Mota', 'M', 7115.00, to_date('12/03/1975',
          'dd/mm/yyyy'), tp_endereco('R. Sanharó, 32',
        '51020-710'), null, (SELECT REF(G) FROM tb_graduado G
        WHERE g.cpf = '420316123-45'), tp_possui(
        tp_titulacaoempregado(
        (SELECT REF(G) FROM tb_grau G WHERE g.cod = 2), (SELECT
        REF(I) FROM tb_ies I WHERE i.cod = 4), to_date('12/11/1999'
         , 'dd/mm/yyyy')) )));
```

Oracle OR

- Inserir objetos do tipo tp_graduado (Cont.)

```
INSERT INTO tb_graduado VALUES
        (tp_graduado('132516702-16',
           'Ana Paula', 'F', 6345.00, to_date('10/04/1970',
            'dd/mm/yyyy'), tp_endereco('R. Janaína, 15','52020-200'),
          null, (SELECT REF(G) FROM tb_graduado G WHERE g.cpf =
          '215439210-15'), tp_possui(tp_titulacaoempregado(
          (SELECT REF(G) FROM tb_grau G WHERE g.cod = 3), (SELECT
          REF(I) FROM tb_ies I WHERE i.cod =1), to_date('20/12/2000'
           , 'dd/mm/yyyy')), tp_titulacaoempregado((SELECT REF(G)
          FROM tb_grau G WHERE g.cod = 1), (SELECT REF(I) FROM
          tb_ies I WHERE i.cod = 4), to_date('10/06/2004',
           'dd/mm/yyyy') )))); 
```


Oracle OR

- Informar nomes dos graduados

```
select g.nome from tb_graduado g;
```

NOME
Helena Ramos
Jonas Mota
Ana Paula

Oracle OR

- Informar o nome do supervisor da graduada Ana Paula

```
select g.supervisor.nome from tb_graduado g where
g.nome = 'Ana Paula';
```

SUPERVISOR.NOME
Jonas Mota