# Lista de Exercício 4 (*ListEx4*)
# Versão 1

## CE-240 Projeto de Sistema de BD

Prof. Dr. Adilson Marques da Cunha
Aluna Simone Cunha Léo

# 1. Introdução

## 1.1 Título

Implementação de um BD Modelo de Dados Relacional e sua Conversão para os Modelos de Dados Hierárquico, Rede e Orientado a Objetos.

## 1.2 Motivação

Com este trabalho a autora desta listex var ter a oportunidade de implementar e testar a 3FN do Protótipo de Aplicativo de Banco de Folha de Pagamento. Complementando convertendo a 3FN do aplicativo em epigrafe para os modelos: Hierárquico, Rede e Orientado a Objetos..

## 1.3 Objetivo

1.3.1) Implementar a Terceira Forma Normal (3ªFN) do Protótipo de Aplicativo de Banco de Dados (BD), da autora desta listex, utilizando um Modelo de Dados Relacional em um Sistema Gerenciador de Banco de Dados (SGBD) previamente escolhido e testar a sua funcionalidade, visando reduzir o desperdício de recursos nas futuras fases de integração e melhorar a eficiência operacional dos futuros Bancos de Dados Setoriais (BDS), Bancos de Dados Corporativo (BDC) e do Banco de Dados Holding (BDH); e

1.3.2) Pesquisar os Modelos de Dados Hierárquico, Rede e Orientado a Objetos, e Converter a 3ªFN do Protótipo de Aplicativo de BD, da autora desta lixtex, no Modelo de Dados Relacional para os Modelos de Dados Hierárquico, Rede e Orientado a Objetos, visando identificar algumas das suas principais diferenças e características.

# 2. Conteúdo

## 2.1 Terceira Forma Normal – 3 FN

VER ANEXO 1

## 2.2 Modelo Fisico

VER ANEXO 2

## 2.3 Criação das Tabelas e Inserção dos Registros (sql).

VER ANEXO 3

**EMPRESA**

**Criação da tabela EMPRESA**

```
CREATE TABLE EMPRESA (
 emp_id VARCHAR(10) NOT NULL,
 emp_nome VARCHAR(45) NOT NULL,
 emp_cnpj VARCHAR(20) NOT NULL,
 emp_vl_salario_base DOUBLE NOT NULL
);

 ALTER TABLE  EMPRESA ADD ( PRIMARY KEY (emp_id) );
```

**Inserção de dados na tabela EMPRESA**

INSERT INTO EMPRESA(emp_id, emp_nome, emp_cnpj, emp_vl_salario_base)
VALUES (emp1, MonitorAma, 34.725.001/1887-01, 350,00}

INSERT INTO EMPRESA(emp_id, emp_nome, emp_cnpj, emp_vl_salario_base)
VALUES (emp2, Solange Confecção, 25.865.001/1998-01, 350,00}

INSERT INTO EMPRESA(emp_id, emp_nome, emp_cnpj, emp_vl_salario_base)
VALUES (emp3, ProjeIn, 28.755.001/1785-01, 350,00}

**PONTO**

**Criação da tabela PONTO**

```
CREATE TABLE PONTO (
  pon_id VARCHAR(10) NOT NULL,
  pon_geom VARCHAR(20) NOT NULL,
  pon_altitude INT NOT NULL
);

ALTER TABLE  PONTO ADD ( PRIMARY KEY (pon_id) ) ;
```

**Inserção de dados na tabela PONTO**

INSERT INTO PONTO (pon_id, pon_geom, pon_altitude)
VALUES (pon1, SDO_GEOMETRY (2001, 8307,null, sdo_elem_info_array (1,1,1),
Sdo_ordinate_array (-23.28, -46.23)), 715)

INSERT INTO PONTO (pon_id, pon_geom, pon_altitude)
VALUES (pon2, SDO_GEOMETRY (2001, 8307,null, sdo_elem_info_array (1,1,1),
sdo_ordinate_array (-23.31, -46.26)), 790)

INSERT INTO PONTO (pon_id, pon_geom, pon_altitude)

VALUES (pon3, SDO_GEOMETRY (2001, 8307,null, sdo_elem_info_array (1,1,1), sdo_ordinate_array (-23.26, -46.47)), 884)

**DEPARTAMENTO**

**Criação da tabela DEPARTAMENTO**

```
CREATE TABLE DEPARTAMENTO (
  dep_id VARCHAR(10) NOT NULL,
  pon_id VARCHAR(10) NOT NULL,
  emp_id VARCHAR(10) NOT NULL,
  dep_nome VARCHAR(45) NOT NULL,
  dep_end VARCHAR(45) NOT NULL,
  dep_cidade VARCHAR(20) NOT NULL,
  dep_cep VARCHAR(20) NOT NULL
);

ALTER TABLE DEPARTAMENTO ADD (PRIMARY KEY (dep_id));
ALTER TABLE DEPARTAMENTO ADD (FOREIGN KEY (emp_id) REFERENCES
EMPRESA);
ALTER TABLE DEPARTAMENTO ADD (FOREIGN KEY (pon_id) REFERENCES
PONTO);
```

**Inserção de dados na tabela DEPARTAMENTO**

```
INSERT INTO DEPARTAMENTO (dep_id, pon_id, emp_id, dep_nome, dep_end,
dep_cidade, dep_cep)
VALUES(dep1, pon1, emp1, Rios, Av. Águas Vermelhas 15, São Paulo SP, 07134-340)

INSERT INTO DEPARTAMENTO (dep_id, pon_id, emp_id, dep_nome, dep_end, dep_cidade,
dep_cep)
VALUES(dep2, pon2, emp1, Bacias, Rua Almino Afonso 200, São Paulo SP, 08240-260)

INSERT INTO DEPARTAMENTO (dep_id, pon_id, emp_id, dep_nome, dep_end,
dep_cidade, dep_cep)
VALUES(dep3, pon3, emp1, Controle dos PCDs, Praça Henrique Dumond Villares 15, São
Paulo SP, 05335-040)
```

**FUNCIONARIO**

**Criação da tabela FUNCIONARIO**

```
CREATE TABLE FUCIONARIO (
  fun_id VARCHAR(10) NOT NULL,
  dep_id VARCHAR(10) NOT NULL,
```

```
  fun_nome VARCHAR(45) NOT NULL,
  fun_cargo VARCHAR(20) NOT NULL,
  fun_dt_admissao DATE NOT NULL,
  fun_vl_salario DOUBLE NOT NULL
);

ALTER TABLE FUNCIONARIO ADD (PRIMARY KEY (fun_id)) ;
ALTER TABLE FUNCIONARIO ADD (FOREIGN KEY (dep_id) REFERENCES
DEPARTAMENTO);
```

**Inserção de dados na tabela PONTO**

INSERT INTO FUCIONARIO (fun_id, dep_id, fun_nome, fun_cargo, fun_dt_admissao, fun_vl_salario)
VALUES(fun1, dep2, Francisco de Assis Junior, Diretor Financeiro, 04/03/1990, 4800,00}

INSERT INTO FUCIONARIO (fun_id, dep_id, fun_nome, fun_cargo, fun_dt_admissao, fun_vl_salario)
VALUES({fun2, dep3, Carlos Augusto Matias, Analista de Sistemas, 10/08/200, 2000,00}

INSERT INTO FUCIONARIO (fun_id, dep_id, fun_nome, fun_cargo, fun_dt_admissao, fun_vl_salario)
VALUES(fun3, dep1, Letícia Costa Maia, Avenida Ver. Joaquim P. Barbosa 133 São Paulo SP, Auxiliar Administrativo, 22/09/2003, 1200,00)

**EVENTO**

**Criação da tabela EVENTO**

```
CREATE TABLE EVENTO (
  eve_id VARCHAR(10) NOT NULL,
  eve_nome VARCHAR(45) NOT NULL,
  eve_il_adic BOOL NOT NULL,
  eve_il_vl BOOL NOT NULL,
  eve_il_hora BOOL NOT NULL,
  eve_il_pc BOOL NOT NULL
);

ALTER TABLE EVENTO ADD (PRIMARY KEY (eve_id)) ;
```

**Inserção de dados na tabela EVENTO**

INSERT INTO EVENTO (eve_id, eve_nome, eve_il_adic, eve_il_vl, eve_il_hora, eve_il_pc)
vALUES(eve1, Hora Extra, 1, 0, 1, 0)

INSERT INTO EVENTO (eve_id, eve_nome, eve_il_adic, eve_il_vl, eve_il_hora, eve_il_pc)
VALUES(eve2, INSS, 0, 0, 0, 1)

INSERT INTO EVENTO (eve_id, eve_nome, eve_il_adic, eve_il_vl, eve_il_hora, eve_il_pc)
VALUES(eve3, IRFF, 0, 1, 0, 1)

**PAGAMENTO**

**Criação da tabela PAGAMENTO**

```
CREATE TABLE PAGAMENTO (
  pag_id VARCHAR(10) NOT NULL,
  fun_id VARCHAR(10) NOT NULL,
  pag_dt DATE NOT NULL,
  pag_vl_liquido DOUBLE NOT NULL
);

ALTER TABLE PAGAMENTO ADD (PRIMARY KEY (pag_id)) ;
ALTER TABLE PAGAMENTO ADD (FOREIGN KEY (fun_id) FUNCIONARIO);
```

**Inserção de dados na tabela PAGAMENTO**

INSERT INTO PAGAMENTO (pag_id, fun_id, pag_dt, pag_vl_liquido)
VALUES (pag1, fun1, 05/01/2007, 2619,30)

**ITEM_PAGAMENTO**

**Criação da tabela ITEM_PAGAMENTO**

```
CREATE TABLE ITEM_PAGAMENTO (
  eve_id VARCHAR(10) NOT NULL,
  pag_id VARCHAR(10) NOT NULL,
  ite_tipo DOUBLE NULL,
  ite_vl_adic DOUBLE NULL,
  Ite_vl_desc DOUBLE NULL
);

ALTER TABLE ITEM_PAMENTO ADD (FOREIGN KEY (pag_id) REFERENCES
PAGAMENTO);
ALTER TABLE ITEM_PAMENTO ADD (FOREIGN KEY (eve_id) REFERENCES
EVENTO);
```

**Inserção de dados na tabela ITEM_PAGAMENTO**

INSERT INTO ITEM_PAGAMENTO (eve_id, pag_id, ite_tipo, ite_vl_adic, Ite_vl_desc)
VALUES (eve1, pag1, 10.00, 127.30, NULL)

INSERT INTO ITEM_PAGAMENTO (eve_id, pag_id, ite_tipo, ite_vl_adic, Ite_vl_desc)
VALUES (eve2, pag1, 11.00, NULL, 308.00)

## 2.4 Consultas

**Lista de Consultas:**

**A) Uma Relação**

Linguagem Natural: Obter o salario do fucionario que possui o cargo de Diretor Financeiro.

SELECT FUNCIONARIO WHERE fun_cargo = ‘Diretor Finaceiro’ GIVING TEMP;
PROJECT TEMP OVER fun_vl_salario GIVING RESULTADO

SELECT fun_vl_salario
FROM FUNCIONARIO
WHERE fun_cargo= ‘Diretor Finaceiro’;

VER ANEXO 4. A)

**B) Duas Relações**

Linguagem Natural: Listar os nomes dos Departamentos pertencentes a empresa MonitorAma.

SELECT EMPRESA WHERE emp_nome = ‘MonitorAma’ GIVING TEMP1;
JOIN TEMP1 AND EMPRESA OVER emp_id TEMP2;
PROJECT TEMP OVER dep_nome GIVING RESULTADO

SELECT dep_nome
FROM DEPARTAMENTO, EMPRESA
WHERE EMPRESA.emp_nome='MonitorAma' and
EMPRESA.emp_id=DEPARTAMENTO.emp_id;

VER ANEXO 4. B)

**C) Três Relações**

Linguagem Natural: Encontrar o nome do Funcionario 1, o nome do Departamento o nome da Empresa que este funcionario trabalha.

SELECT FUNCIONARIO fuc_id = 1 GIVING TEMP1;
JOIN TEMP1 AND DEPARTAMENTO over dep_id GIVING TEMP2;
JOIN TEMP2 AND EMPRESA over emp_id GIVING TEMP3;
PROJEC TEMP3 OVER (fun_nome, dep_nome, emp_nome) GIVING RESULTADO;

SELECT FUNCIONARIO.fun_nome, DEPARTAMENTO.dep_nome, EMPRESA.emp_nome
FROM FUNCIONARIO, DEPARTAMENTO, EMPRESA
WHERE FUNCIONARIO.fun_id='fun1'
And FUNCIONARIO.dep_id=DEPARTAMENTO.dep_id

And EMPRESA.emp_id=DEPARTAMENTO.emp_id;

VER ANEXO 4. C)

**D) Georeferenciada**

Linguagem Natural: Encontrar a localização do ponto do Departamento 1.

```
SELECT DEPARTAMENTO dep_id = 1 GIVING TEMP1
JOIN TEMP1 AND PONTO over (pon_id) GIVING TEMP2;
PROJECT TEMP2 OVER (pon_altitude, pon_geom) GIVING RESULTADO;
```

```
Select PONTO.pon_geom, PONTO.pon_altitude
From DEPARTAMENTO, PONTO
Where DEPARTAMENTO.dep_id=1
And DEPARTAMENTO.pon_id = PONTO. pon_id;
```

VER ANEXO 4. D)

## 2.4 Converter do Modelo de Dados Relacional para os Modelos de Dados Hierárquico, Rede e Orientado a Objetos.

VER ANEXO 5

**A) Modelo Hierárquico**

Um BD hierárquico é uma coleção de árvores de registros. Os registros são usados para representar os dados e ponteiros são usados para representar o relacionamento entre os dados, numa ligação do tipo pai-filho.

As restrições são:

- Existe redundância de dados (um registro filho tem mais do que um pai é duplicado), para evitar isso um determinado registro somente pode possuir um registro pai;
- Dificuldades de representação de relacionamentos do tipo Muitos para Muitos;
- Ao eliminar um registro eliminam-se os seus filhos, podendo-se eliminar informação relevante;
- A programação nos SGBD hierárquicos é muito complexa.
-

De acordo com a restrição que um filho só pode ter um pai, este modelo de Dados terá duplicação de registro. Portanto para o dado problema esta Modelagem (Modelo Hierárquico) não é apropriada.

VER ANEXO 5. A)

**B) Modelo de Redes**

O BD em rede é um grafo, onde os nós representam os registros e os arcos representam os relacionamentos entre os registros, através de ligações pai-filho. Diferente do modelo hierárquico, um registro pode possuir diversos registros pai, (eliminando o conceito de hierarquia, permitindo que um mesmo registro possua várias associações);

VER ANEXO 5. B)

**C) Modelo Orientado a Objetos**

No modelo de dados orientado a objetos todas as entidades conceituais são modeladas com objetos (Worboys, 1994). Um objeto representa uma única entidade, e descreve tanto seus atributos quanto o seu comportamento. Cada objeto pertence a uma classe. A classe define uma estrutura e um conjunto de operações que são comuns a um grupo de objetos. Instancia são objetos individuais de uma determinada classe. Um objeto funciona como uma estrutura de dados complexa, que é capaz de armazenar todos os seus dados, juntamente com informações sobre os procedimentos necessários pra sua própria criação, destruição e manipulação.
Este modelo de dados é mais adequado para o tratamento de objetos complexos como textos, gráficos e imagens, e dinâmicos, como programas e simulações.

VER ANEXO 5. C)

# 3. Conclusão

Com esta Lista de Exercício foi possível melhorar o modelo Entidade Relacionamento inclusive mudar o banco de dados para atender os tipos de dados georeferenciados de acordo com o Oracle 10g.
A autora desta listex observou que o modelo hierárquico não é apropriado devido ao fato da duplicidade de registros, já o modelo de redes suporta um filho possuir mais de um pai. Entre os três: Hierárquico, Redes e Orientado a Objetos, o Orientado a Objetos seria o mais indicado.