Universidade de São Paulo
Escola de Artes, Ciências e Humanidades
Disciplina: Laboratório de Banco de Dados
Profª Dra. Fátima Nunes.

Administração de Condomínio

Parte III - Artefato A

Fernando K. G. de Amorim – 10387644
João Guilherme da Costa Seike – 9784634
Lucas Pereira Castelo Branco – 10258772
Victor Gomes de O. M. Nicola – 9844881

São Paulo
2019

Objetivo do Artefato A:

Inserir pelo menos uma característica de cada categoria abaixo do modelo objeto-relacional e enunciar uma regra de negócio ou um requisito de sistema, que motive o seu uso. As características que devem ser trabalhadas são:

- "**objetos complexos**" (atributo composto e atributo multivalorado): inserir estes atributos no modelo (considere apenas a parte do modelo em que os atributos serão inseridos), implementar essa modelagem no SGBD (se possível) ou apresentar o código SQL padrão para a implementação desta modelagem (se a tecnologia não permitir sua implementação). Seja uma implementação de SGBD ou um código SQL, comentar o código destacando tomadas de decisões que são úteis no contexto do sistema modelado;
- "**tipos referência**": fazer uso deste recurso no SGBD (se possível) ou apresentar o código SQL padrão para a implementação desta modelagem (se a tecnologia não permitir sua implementação). Seja uma implementação de SGBD ou um código SQL, comentar o código destacando tomadas de decisões que são úteis no contexto do sistema modelado;
- "**herança**": inserir este recurso no modelo relacional (considere apenas a parte do modelo em que os atributos serão inseridos), implementar essa modelagem no SGBD (se possível) ou apresentar o código SQL padrão para a implementação desta modelagem (se a tecnologia não permitir sua implementação). Seja uma implementação de SGBD ou um código SQL, comentar o código destacando tomadas de decisões que são úteis no contexto do sistema modelado;

Regras de negócio:

Todo condomínio possui funcionários, para saber quem trabalha em cada condomínio, é necessário saber quem é a pessoa, qual sua função, seu horário de trabalho, e a referência para o condomínio que ela trabalha, garantindo que não trabalhe em outro condomínio.

Códigos:

Implementação em PostgreSQL:

```
CREATE TYPE adm_condominio.TPessoa AS (

        cpf varchar(11),
        nome varchar(40),
        data_nascimento date,
        sexo char(1)
);
```

O tipo Pessoa, é um tipo de objeto composto, ele possui um CPF de uma pessoa, seu nome, a data de nascimento dela e o seu sexo. Pois queremos saber que Pessoa é essa, é o tipo mais básico para compor um Funcionário.

```
CREATE TYPE adm_condominio.TCondominio AS (

        nome_condominio varchar(40),
        tipo_condominio char(1)
);
```

O tipo Condominio é também um objeto composto, ele possui o nome do condomínio e qual o tipo de condomínio corresponde o mesmo, associação ou edifício.

```
CREATE TYPE adm_condominio.TFuncionario AS (

        pessoa TPessoa,
        funcao varchar(40),
        horario_entrada time,
        horario_saida time
);
```

Criamos então o tipo Funcionario para a nossa regra de negócio, inserimos o tipo Pessoa dentro de Funcionario para simular uma herança, pois o PostgreSQL não possui uma implementação de herança para TYPE, somente para tabelas normais, com isso, inseri-lo em outro tipo é uma forma de simular a herança de TYPEs. O tipo Funcionario possui uma função, ele descreve o que o Funcionario faz no condomínio. E claro, um horário de entrada e saída do seu trabalho.

```
CREATE TABLE adm_condominio.TipoCondominio (

        id_condominio serial not null primary key,
        condominio TCondominio,
        fk_id_endereco int not null references
        adm_condominio.Endereco(id_endereco)
);
```

Para fazermos a referência com tipos, é necessário a criação de uma tabela com um identificador de chave primária para ser feita a referência desse condomínio em Funcionario, dado que não existe implementação no PostgreSQL, é necessário simular a referência utilizando os recursos do modelo relacional, com a utilização de chaves primárias e estrangeiras.

```
CREATE TABLE adm_condominio.TipoFuncionario (

        id_funcionario serial not null primary key,
        funcarionario TFuncionario,
        fk_id_condominio int not null references
adm_condominio.TipoCondominio(id_condominio)
);
```

Com a criação da tabela de tipo de Condomínio com a chave primária, precisamos criar uma tabela com tipo Funcionario com uma chave estrangeira que faça a relação com a tabela de tipo Condomínio, simulando assim a referência de tipos.

Implementação em SQL Padrão:

```
CREATE TYPE Pessoa (

        cpf varchar(11),
        nome varchar(40),
        data_nascimento date,
        sexo char(1)
) not final


CREATE TYPE Condominio (
        id_condominio int,
        nome_condominio varchar(40),
        tipo_condominio char(1)
```

```
) final

CREATE TYPE Funcionario under Pessoa
(
        funcao varchar(40),
        horario_entrada time,
        horario_saida time,
        id_condominio ref(Condominio) scope id_condominio
) final
```