Exercício 3 - Criação das Tabelas SQL

1.1 Contextualização do problema

Posteriormente às modelagens realizadas nas etapas anteriores (MER e MRel) para o serviço online de anúncio e reserva de propriedades online, precisamos implementar o resultado das relações em tabelas, adotando o PostgreSQL. Ainda que, toda transformação do modelo relacional para comandos em SQL seja trivial e direta, alguns detalhes precisam ficar claros, como por exemplo, os tipos de dados, garantindo assim que no final obtenhamos o desenvolvimento de uma base de dados mais fiel à proposta pelo problema. Estas considerações, mais relevantes, serão abordadas aqui abaixo.

1.2 Estrutura geral do comando CREATE TABLE para criação de tabelas

Comando de criação de tabela: CREATE TABLE

```
CREATE TABLE Tabela (
        Att1            DataType [NOT NULL],
        Att2            DataType [NOT NULL],
        Attn            DataType [NOT NULL],
            [constraints]
);
```

1.2.1 Funcionamento:

1. CREATE TABLE Tabela: Cria a tabela na base de dados com respectivo nome ‘Tabela’
2. Att1, Att2, Attn: Define os atributos da tabela.
3. DataType: Tipo de dado dos atributos das colunas.
4. [NOT NULL]: Define, opcionalmente, que um atributo não pode ter valor nulo. Usado, muita das vezes, para definir chaves candidatas que se deve garantir unicidade.
5. [constrainsts]: Define as restrições de integridade, as quais são opcionais. Apenas as seguintes restrições foram utilizadas nesta implementação:
    - CONSTRAINT ConstraintName PRIMARY KEY(AttPK): Cria-se uma restrição, determinando a chave primária ‘AttPK’ e cuja restrição é nomeada por ‘ConstraintName’
    - CONSTRAINT ConstraintName FOREIGN KEY(AttFK) REFERENCES TableNameFK(AttRef): Define uma restrição com nome ‘ConstraintName’, determinando uma chave estrangeira ‘AttFK’, a qual referencia ‘AttRef’ da tabela ‘TableNameFK’.

1.3 Criação das tabela a partir do Modelo Relacional

As tabelas para este problema foram definidas segundo a seguinte sequência:

a) Tabelas independentes: CE's e CR's que não possuem chaves estrangeiras.
   Exemplo: **Figura 1.3.1 - Regra, Comodidade e Localizacao**
b) Tabelas dependentes apenas das criadas em *a)*, ou seja, tabelas que possuam chaves estrangeiras contidas nas tabelas criadas na etapa *a)*
   Exemplo: **Figura 1.3.2 - Ponto_Interesse**
c) Tabelas dependentes das criadas em a) e/ou b)
   Exemplo: **Figura 1.3.3 - Usuario**

```
CREATE TABLE Regra (
    ID_Regra            SERIAL NOT NULL,
    Nome                VARCHAR(50),
    Descricao           TEXT,
        CONSTRAINT PK_REGRA PRIMARY KEY(ID_Regra)
);

CREATE TABLE Comodidade (
    ID_Comodidade       SERIAL NOT NULL,
    Nome                VARCHAR(50),
    Descricao           TEXT,
        CONSTRAINT PK_COMODIDADE PRIMARY KEY(ID_Comodidade)
);

CREATE TABLE Localizacao (
    Cidade              VARCHAR(40) NOT NULL,
    Estado              VARCHAR(40) NOT NULL,
    Pais                VARCHAR(40) NOT NULL,
    Bairro              VARCHAR(40),
        CONSTRAINT PK_LOCALIZACAO PRIMARY KEY(Cidade, Estado, Pais)
);
```

Figura 1.3.1 - Regra, Comodidade e Localizacao

```sql
CREATE TABLE Ponto_Interesse (
    ID_PontoInteresse   SERIAL NOT NULL,

    Cidade              VARCHAR(40) NOT NULL,
    Estado              VARCHAR(40) NOT NULL,
    Pais                VARCHAR(40) NOT NULL,

    Nome                VARCHAR(100),
    Descricao           TEXT,

        CONSTRAINT PK_PONTO_INTERESSE PRIMARY KEY(ID_PontoInteresse, Cidade, Estado, Pais),
        CONSTRAINT FK_PONTO_INTERESSE FOREIGN KEY (Cidade, Estado, Pais)
            REFERENCES Localizacao(Cidade, Estado, Pais)
);
```

Figura 1.3.2 - Ponto_Interesse

```sql
CREATE TABLE Usuario (
    Nome                VARCHAR(50) NOT NULL,
    Sobrenome           VARCHAR(50) NOT NULL,
    Telefone            VARCHAR(30) NOT NULL,
    Tipo                VARCHAR(15),
    Data_Nasc           DATE,
    Endereco            VARCHAR(70),
    Sexo                CHAR(1),
    Email               VARCHAR(40),
    Senha               VARCHAR(256),

    Cidade              VARCHAR(40),
    Estado              VARCHAR(40),
    Pais                VARCHAR(40),

        CONSTRAINT PK_USUARIO PRIMARY KEY(Nome, Sobrenome, Telefone),
        CONSTRAINT FK_USUARIO FOREIGN KEY (Cidade, Estado, Pais)
            REFERENCES Localizacao(Cidade, Estado, Pais)
);
```

Figura 1.3.3 - Usuario

2.1 Aspectos importantes sobre as Tabelas e Atributos criados

1. As tabelas que possuem ID como chaves primária possuem o tipo de dado ***SERIAL***, permitindo que durante a inserção o próprio SGBD crie valores inteiros consecutivos.
   Exemplo: Figura 1.3.1
2. Variáveis que armazenam valores monetários são declaradas como do tipo ***MONEY***, dado que o PostgreSQL consegue lidar com operações monetárias nesse tipo de dado. Exemplo: Figura 2.1.1 - Preco_Estadia do tipo Money
3. Na tabela Usuário o atributo *Sexo* foi mapeado como um **CHAR** de tamanho 1 ('M', 'F'). Exemplo: Figura 2.1.2 - Tabela Usuário
4. Em *Usuário*, o atributo *Tipo* representa o tipo de usuário na plataforma, ou seja, "Locatário", "Anfitrião" ou "Ambos".
   Exemplo: Imagem 2.1.2 - Tabela Usuário
5. Em *Fotos*, no atributo *Conteudo*, utilizou-se o atributo **VARCHAR(200)** para representar um link para a imagem que se deseja salvar em conteúdo. Caso deseje salvar o conteúdo da imagem em si, basta trocar este atributo para **BYTEA**.
   Exemplo: Figura 2.1.3 - Tabela Fotos

```
CREATE TABLE Locacao (
    ID_Locacao          SERIAL NOT NULL,
    Preco_Estadia       MONEY,
    Numero_Hospedes     DECIMAL(2),
    Codigo_Promocional  VARCHAR(15),
    Imposto_Pago        MONEY,
    Preco_Total         MONEY,
    Desconto            MONEY,

        CONSTRAINT PK_LOCACAO PRIMARY KEY(ID_Locacao)
);
```

Figura 2.1.1 - Preco_Estadia do tipo MONEY

```sql
CREATE TABLE Usuario (
    Nome                VARCHAR(50) NOT NULL,
    Sobrenome           VARCHAR(50) NOT NULL,
    Telefone            VARCHAR(30) NOT NULL,
    Tipo                VARCHAR(15),
    Data_Nasc           DATE,
    Endereco            VARCHAR(70),
    Sexo                CHAR(1),
    Email               VARCHAR(40),
    Senha               VARCHAR(256),

    Cidade              VARCHAR(40),
    Estado              VARCHAR(40),
    Pais                VARCHAR(40),

        CONSTRAINT PK_USUARIO PRIMARY KEY(Nome, Sobrenome, Telefone),
        CONSTRAINT FK_USUARIO FOREIGN KEY (Cidade, Estado, Pais)
            REFERENCES Localizacao(Cidade, Estado, Pais)
);
```

Figura 2.1.2 - Tabela Usuario

```sql
CREATE TABLE Fotos (
    Nome                    VARCHAR(50) NOT NULL,

    Avaliacao               SERIAL NOT NULL,

    Conteudo                VARCHAR(200),

        CONSTRAINT PK_FOTOS PRIMARY KEY(Nome, Avaliacao),
        CONSTRAINT FK_FOTOS FOREIGN KEY(Avaliacao)
            REFERENCES Avaliacao(ID_Avaliacao)
);
```

Imagem 2.1.3 - Tabela Fotos