UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

Departamento de Matemática Aplicada

MAE015 - Top. Engenharia de Dados A: Banco de Dados para a Ciência

Imagem retirada da web

Avaliação de Aplicação de Banco de Dados Relacional

Aluno: Paulo Victor Innocencio

DRE: 116213599

Aplicação de Banco de Dados p/ Locadora de Veículos

Introdução

Este relatório detalha o desenvolvimento de um sistema de gerenciamento para uma locadora de veículos. O sistema foi desenvolvido em Python e utiliza o MySQL como Sistema de Gerenciamento de Banco de Dados (SGBD). O objetivo é fornecer funcionalidades para registrar, consultar e deletar informações de clientes, veículos, locações, reservas e outros elementos relacionados.

Estrutura do Banco de Dados

A estrutura do banco de dados foi projetada para atender às necessidades da locadora, com as seguintes tabelas principais:

- **Grupo_Veiculos**: Armazena informações sobre os diferentes grupos de veículos disponíveis para locação.
- **Veiculo**: Armazena informações detalhadas sobre cada veículo, incluindo características e grupo ao qual pertence.
- **Acessorio**: Contém a descrição dos acessórios disponíveis.
- **Veiculo_Acessorio**: Relaciona veículos com seus acessórios.
- **Cliente**: Armazena informações dos clientes.
- **Pessoa_Fisica** e **Pessoa_Juridica**: Subclasses da tabela Cliente para diferenciar pessoas físicas de jurídicas.
- **Funcionario**: Armazena informações dos funcionários da locadora.
- **CNH**: Registra as carteiras de habilitação dos funcionários.
- **Reserva**: Armazena informações sobre as reservas de veículos.

- **Locacao**: Contém os dados das locações efetuadas.
- **Foto**: Armazena URLs de fotos dos veículos.
- **Prontuario**: Registra históricos e ocorrências dos veículos.
- **Protecao_Adicional**: Armazena tipos de proteções adicionais disponíveis.
- **Locacao_Protecao**: Relaciona locações com as proteções adicionais contratadas.

Criação das Tabelas (SQL)

Foram elaborados scripts SQL para a criação das tabelas no banco de dados MySQL, incluindo chaves primárias e estrangeiras, além de tabelas intermediárias para relacionamentos muitos-para-muitos (N).

```sql
CREATE TABLE Veiculo (
    Placa VARCHAR(10) PRIMARY KEY,
    Chassis VARCHAR(50) NOT NULL,
    Marca VARCHAR(50) NOT NULL,
    Modelo VARCHAR(50) NOT NULL,
    Cor VARCHAR(20) NOT NULL,
    Tipo_Mecanizacao ENUM('Manual', 'Automática') NOT NULL,
    Ar_Condicionado BOOLEAN NOT NULL,
    Cadeirinha BOOLEAN NOT NULL,
    Dimensoes VARCHAR(50),
    ID_Grupo INT,
    FOREIGN KEY (ID_Grupo) REFERENCES Grupo_Veiculos(ID_Grupo)
);
```

Exemplo da criação da Tabela 'Veículo'

Desenvolvimento do Código

O código foi desenvolvido em Python utilizando a biblioteca *mysql.connector* para interagir com o banco de dados MySQL. A seguir estão alguma das principais funções implementadas:

1. Conexão ao Banco de Dados

A função *create_connection* estabelece a conexão com o banco de dados MySQL utilizando os parâmetros de host, usuário, senha e nome do banco de dados.

```python
import mysql.connector
from mysql.connector import Error

def create_connection(host_name, user_name, user_password, db_name):
    connection = None
    try:
        connection = mysql.connector.connect(
            host=host_name,
            user=user_name,
            passwd=user_password,
            database=db_name
        )
        print("Conexão ao MySQL DB bem sucedida")
    except Error as e:
        print(f"O erro '{e}' ocorreu")
    return connection
```

Exemplo da Conexão ao Banco de Dados

2. Registro de Clientes

A função registrar_cliente insere um novo cliente na tabela 'Cliente'.

```python
def registrar_cliente(connection, cliente):
    query = """
    INSERT INTO Cliente (Nome, Endereco, Telefone, Email) VALUES (%s, %s, %s, %s)
    """
    cursor = connection.cursor()
    try:
        cursor.execute(query, cliente)
        connection.commit()
        print("Cliente registrado com sucesso")
    except Error as e:
        print(f"O erro '{e}' ocorreu")
```

Exemplo do Registro de Clientes no Banco de dados

3. Consulta de Locações Atuais

A função *consultar_locacoes_atuais* consulta e exibe todas as locações cuja data de devolução seja maior ou igual à data atual.

```python
def consultar_locacoes_atuais(connection):
    query = """
    SELECT * FROM Locacao WHERE Data_Hora_Devolucao >= CURDATE()
    """
    cursor = connection.cursor()
    try:
        cursor.execute(query)
        locacoes = cursor.fetchall()
        for locacao in locacoes:
            print(locacao)
    except Error as e:
        print(f"O erro '{e}' ocorreu")
```

Exemplo da Consulta de Locações o ao Banco de Dados

4. Registro de Locações

A função *registrar_locacao* insere uma nova locação na tabela 'Locacao'.

```python
def registrar_locacao(connection, locacao):
    query = """
    INSERT INTO Locacao (Data_Hora_Retirada, Data_Hora_Devolucao, Patio_Saida, Patio_Chegada, ID_Cliente, Placa, Estado_Entrega, Estado_Devolucao) VALUES (%s, %s
    """
    cursor = connection.cursor()
    try:
        cursor.execute(query, locacao)
        connection.commit()
        print("Locação registrada com sucesso")
    except Error as e:
        print(f"O erro '{e}' ocorreu")
```

Exemplo do Registro de Locações o ao Banco de Dados

5. Deletar Registros

Foram implementadas funções para deletar registros das tabelas '*Cliente*', '*Veiculo*' e '*Locacao*'.

```python
def deletar_cliente(connection, id_cliente):
    query = """
    DELETE FROM Cliente WHERE ID_Cliente = %s
    """
    cursor = connection.cursor()
    try:
        cursor.execute(query, (id_cliente,))
        connection.commit()
        print("Cliente deletado com sucesso")
    except Error as e:
        print(f"O erro '{e}' ocorreu")

def deletar_veiculo(connection, placa):
    query = """
    DELETE FROM Veiculo WHERE Placa = %s
    """
    cursor = connection.cursor()
    try:
        cursor.execute(query, (placa,))
        connection.commit()
        print("Veículo deletado com sucesso")
    except Error as e:
        print(f"O erro '{e}' ocorreu")

def deletar_locacao(connection, id_locacao):
    query = """
    DELETE FROM Locacao WHERE ID_Locacao = %s
    """
    cursor = connection.cursor()
    try:
        cursor.execute(query, (id_locacao,))
        connection.commit()
        print("Locação deletada com sucesso")
    except Error as e:
        print(f"O erro '{e}' ocorreu")
```

Exemplo das Funções para Deletar Registros ao Banco de Dados

Testes e Validação

O sistema foi testado para garantir que todas as operações de inserção, consulta e deleção funcionem corretamente. Exemplos de registros foram criados e deletados com sucesso, conforme mostrado nos testes abaixo:

```python
# Configurações de conexão
connection = create_connection("127.0.0.1", "root", "password", "locadora_veiculos")

# Exemplos de uso
novo_cliente = ("Maria Silva", "Rua A, 123", "555-1234", "maria@example.com")
registrar_cliente(connection, novo_cliente)

novo_veiculo = ("ABC-1234", "1HGCM82633A123456", "Honda", "Civic", "Preto", "Automática", True, False, "4.5x1.8x1.4m", 1)
registrar_veiculo(connection, novo_veiculo)

nova_locacao = ("2023-07-01 10:00:00", "2023-07-10 10:00:00", "Patio A", "Patio B", 1, "ABC-1234", "Bom", "Bom")
registrar_locacao(connection, nova_locacao)

novo_pagamento = (1, "2023-07-01", 1500.00)
registrar_pagamento(connection, novo_pagamento)

consultar_locacoes_atuais(connection)

consultar_historico_locacoes(connection, 1)

# Exemplos de deletar dados
deletar_cliente(connection, 1)
deletar_veiculo(connection, "ABC-1234")
deletar_locacao(connection, 1)
deletar_pagamento(connection, 1)
```

Exemplo dos Testes e Validações de Registros realizados no Banco de Dados

Conclusão

O sistema de gerenciamento para a locadora de veículos foi desenvolvido com sucesso, proporcionando funcionalidades essenciais para o registro, consulta e deleção de dados. O uso de Python em conjunto com MySQL garantiu uma integração eficiente e fácil de implementar. O código foi testado e validado, assegurando a funcionalidade esperada.

Referências

- Materiais de apoio da disciplina: ***Top. Engenharia de Dados A: Banco de Dados para a Ciência***
- Dicionário de dados: *https://www.luis.blog.br/dicionario-de-dados.html*

- Diferences between ER Diagram and ERR diagram: *https://nulab.com/learn/software-development/er-diagrams-vs-eer-diagrams-whats-the-difference/*
- Todos os códigos fonte e arquivos adicionais disponíveis em: *https://github.com/xxpaulo-victorxx/MAE015-Aplicacao-SGBDR*