Departamento de Eletrónica, Telecomunicações e Informática
Mestrado Integrado em Engenharia de Computadores e Telemática

Gestão de Stands e Oficinas

Turma P1G10
João Coelho, 80335
Marco Silva, 84770

Base de dados - 2018/2019

Descrição do Projeto

- Interface de gestão de uma empresa com Stands e Oficinas destinada aos funcionários da mesma
- Tem como objetivo simplificar o processo de vendas de veículos, gestão de peças em stock e da reparação de veículos dos clientes

REQUISITOS

- Gestão de peças em stock
- Gestão de veículos para venda
- Gestão de veículos para reparar
- Gestão de mecânicos
- Gestão de clientes
- Gestão de oficinas
- Gestão de stands

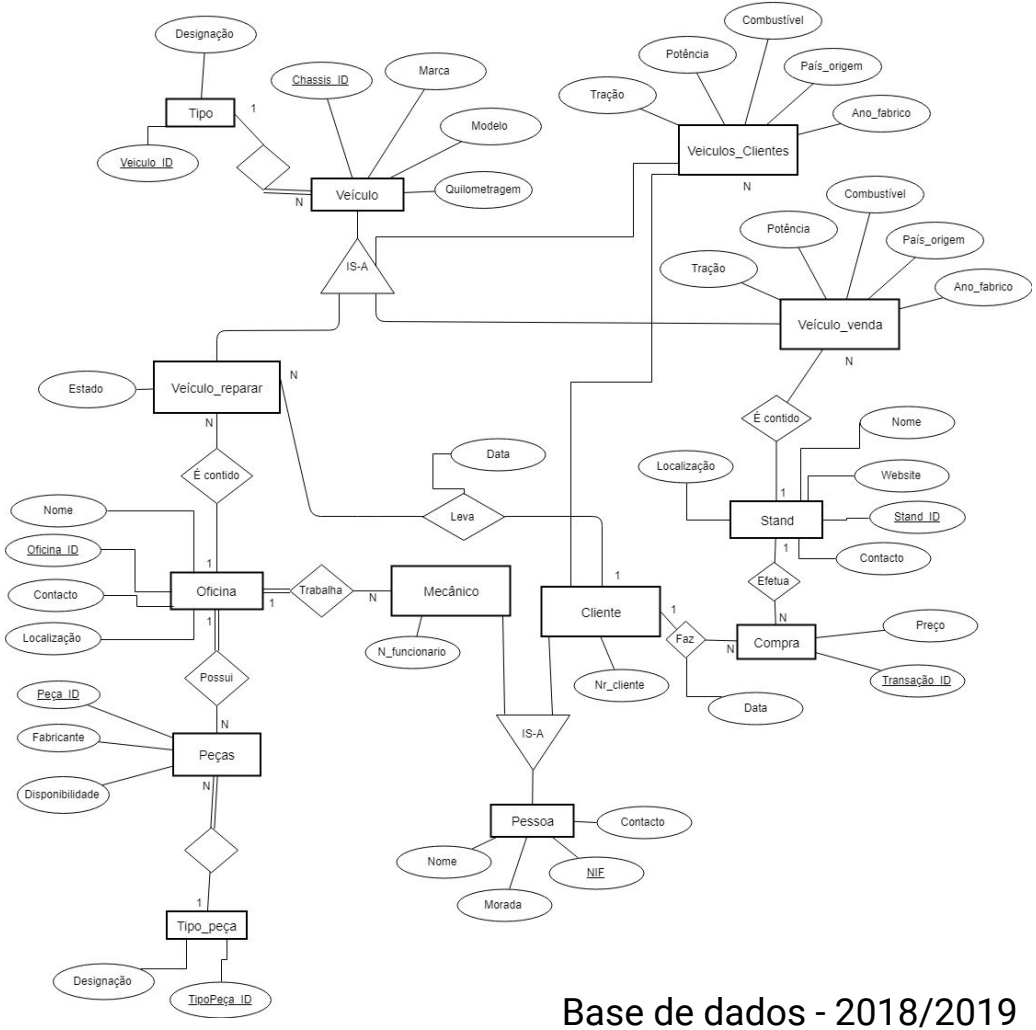


Diagrama de Entidade-Relação

UDFs

```
-- Retorna todos os veiculos do cliente correspondente ao ID dado
CREATE FUNCTION STAND.ClientVehicles(@Cliente INT) RETURNS TABLE AS
    RETURN(
        SELECT Chassis_ID, Marca, Modelo, Tipo_Veiculo
        FROM STAND.VEICULOS_CLIENTES WHERE Cliente_ID = @Cliente
    );
GO

-- Retorna todos os veiculos de todos os clientes
CREATE FUNCTION STAND.AllClientVehicles() RETURNS TABLE AS
    RETURN(
        SELECT Chassis_ID, Marca, Modelo, Tipo_Veiculo, Cliente_ID
        FROM STAND.VEICULOS_CLIENTES
    );
GO

-- Retorna todos os veiculos de um certo tipo (Recebe o ID do tipo como argumento)
CREATE FUNCTION STAND.VehicleByType(@Veiculo INT) RETURNS TABLE AS
    RETURN(
        SELECT 'Para Reparar' AS Type, Chassis_ID,Marca,Modelo FROM STAND.VEICULO_REPARAR WHERE Tipo_Veiculo = @Veiculo
        UNION
        SELECT 'Para Venda' AS Type, Chassis_ID,Marca,Modelo FROM STAND.VEICULO_VENDA WHERE Tipo_Veiculo = @Veiculo
        UNION
        SELECT 'Dos Clientes' AS Type, Chassis_ID,Marca,Modelo FROM STAND.VEICULOS_CLIENTES WHERE Tipo_Veiculo = @Veiculo
        UNION
        SELECT 'Repararados' AS Type, Chassis_ID,Marca,Modelo FROM STAND.VEICULOS_REPARADOS WHERE Tipo_Veiculo = @Veiculo
    );
GO
```

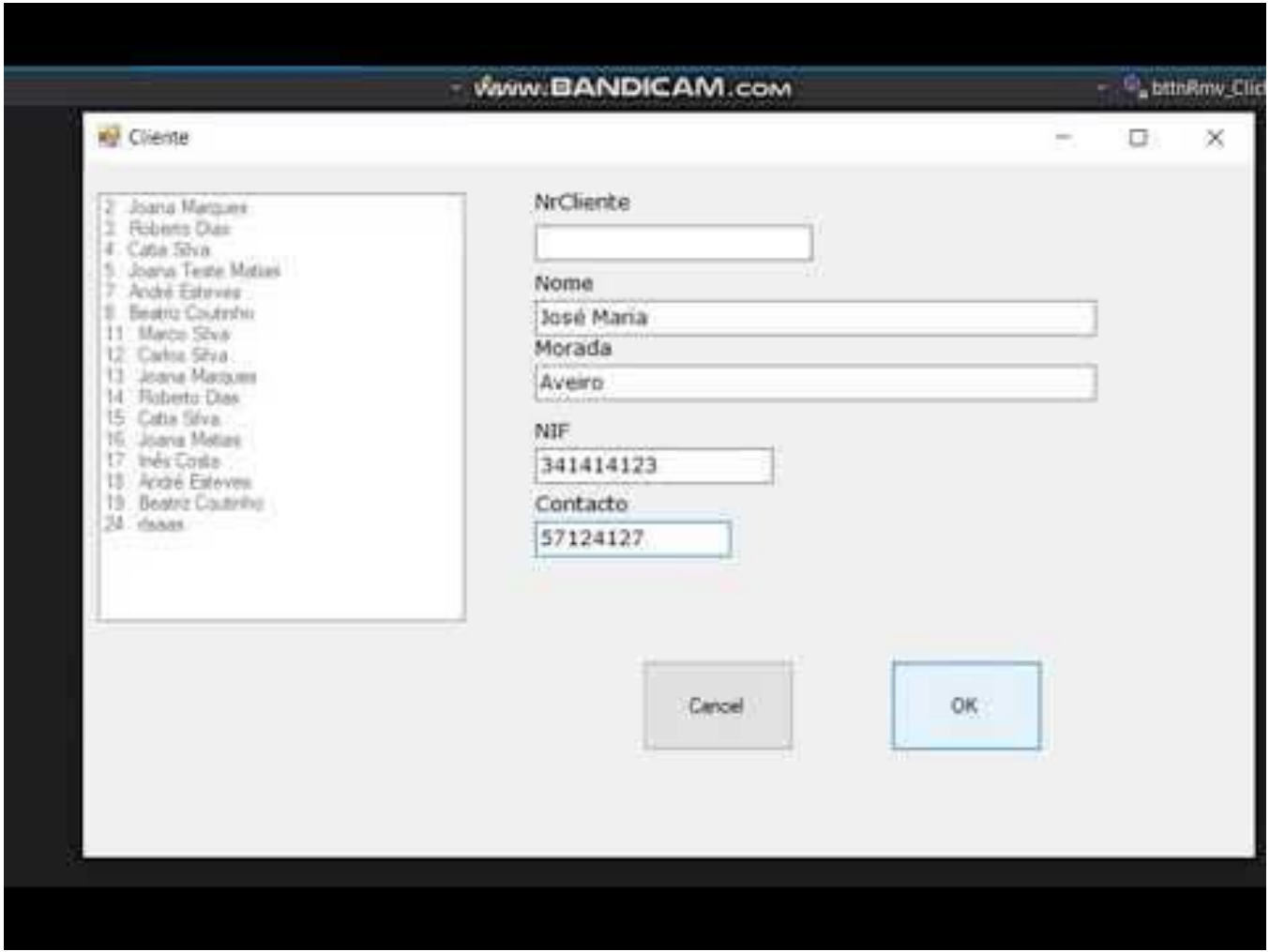
Triggers

```
CREATE TRIGGER trg_updateTransaction on STAND.FAZ AFTER INSERT AS
BEGIN
    UPDATE STAND.FAZ SET Data_Compra = GETDATE() FROM STAND.FAZ f JOIN Inserted i ON f.Transacao = i.Transacao;
END;
GO

CREATE TRIGGER trg_updateVehicleToMechanic on STAND.LEVA AFTER INSERT AS
BEGIN
    UPDATE STAND.LEVA SET Data_Entrega = GETDATE() FROM STAND.LEVA l JOIN Inserted i ON l.Veiculo_ID = i.Veiculo_ID;
END;
```

Stored Procedures

```
--Adicionar cliente
CREATE PROCEDURE sp_addClient @Nome VARCHAR(40), @NIF CHAR(9), @Morada VARCHAR(30), @Contacto CHAR(9) AS
    IF @NIF is null
    BEGIN
        PRINT 'O NIF não pode ser vazio.'
        RETURN
    END

    DECLARE @nifClient AS TINYINT
    SET @nifClient = (SELECT COUNT(STAND.CLIENTE.NIF) AS nifClient FROM STAND.CLIENTE WHERE STAND.CLIENTE.NIF=@NIF)

    DECLARE @nifMechanic AS TINYINT
    SET @nifMechanic = (SELECT COUNT(STAND.MECANICO.NIF) AS nifMechanic FROM STAND.MECANICO WHERE STAND.MECANICO.NIF=@NIF)

    IF (@nifMechanic != 0)
    BEGIN
        PRINT 'Este NIF esta associado a um funcionário'
        RETURN
    END

    IF (@nifClient = 0)
    BEGIN
        IF @Nome is null
        BEGIN
            PRINT 'O Nome não pode estar vazio'
            RETURN
        END

        IF @Contacto is null
        BEGIN
            PRINT 'O Contacto não pode estar vazio'
            RETURN
        END

        INSERT INTO STAND.CLIENTE(Nome, NIF, Morada, Contacto)
        VALUES (@Nome, @NIF, @Morada, @Contacto);
    END
    ELSE
        PRINT 'O cliente associado a este NIF ja existe'
GO
```

DEMO


www.BANDICAM.com
Cliente
2 Joana Marques
3 Roberto Dias
4 Catia Silva
5 Joana Teste Matias
7 André Esteves
8 Beatriz Coutinho
11 Marco Silva
12 Carlos Silva
13 Joana Marques
14 Roberto Dias
15 Catia Silva
16 Joana Matias
17 Inês Costa
18 André Esteves
19 Beatriz Coutinho
24 dsaas
NrCliente
Nome
José Maria
Morada
Aveiro
NIF
341414123
Contacto
57124127
Cancel
OK