Perfumaria Orquídea

Apresentação Final

Base de Dados

P6G1

Hugo Paiva de Almeida 93915 LEI
Pedro Bastos 93150 LEI

Introdução

O porquê?

Interesse Pessoal

Funcionalidades

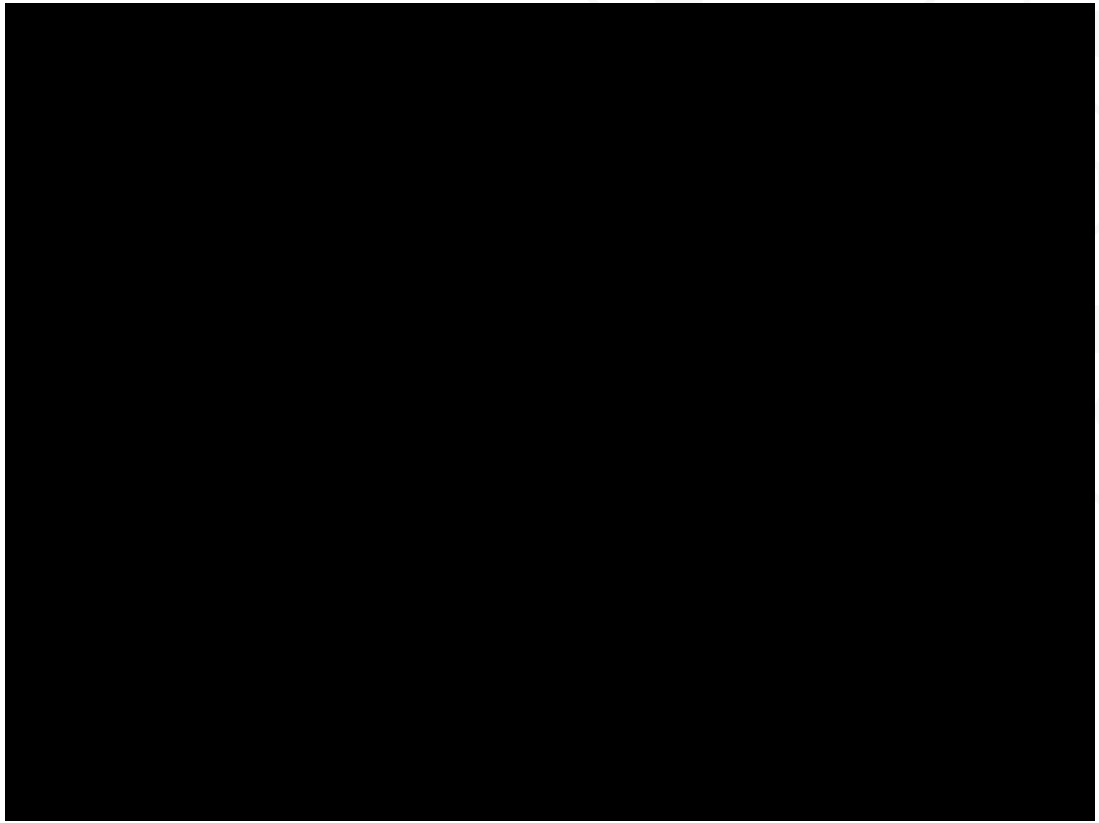
Cliente

Compras e Serviços

Funcionário

E outras...

Demo



Stored Procedures

```sql
DROP PROCEDURE perf.getProductFilters;
GO

CREATE PROCEDURE perf.getProductFilters
    @deleted BIT = NULL,
    @nome VARCHAR(30) = NULL,
    @marca VARCHAR(30) = NULL,
    @categoria VARCHAR(30) = NULL,
    @destinatario VARCHAR(10) = NULL,
    @orderby VARCHAR(50) = NULL,
    @ordem VARCHAR(30) = NULL
AS
BEGIN
        SET NOCOUNT ON
        IF (@ordem = 'Ascendente')
        BEGIN
            SELECT id, preco, familiaolfativa, categoria, nome, marca, linha, tamanho, descricao, imagem, stock, destinatario, deleted
            FROM Perfumaria.perf.produto
            WHERE stock > 0 AND
                  deleted = 0 AND
                  nome LIKE ('%'+ISNULL(@nome, nome)+'%') AND
                  marca = ISNULL(@marca,marca) AND
                  categoria = ISNULL(@categoria,categoria)
            ORDER BY CASE WHEN @orderby='Nome' THEN nome END,
                     CASE WHEN @orderby='Marca' THEN marca END,
                     CASE WHEN @orderby='Categoria' THEN categoria END,
                     CASE WHEN @orderby='Preço' THEN preco END
        END

        ELSE IF (@ordem = 'Descendente')
        BEGIN
            SELECT id, preco, familiaolfativa, categoria, nome, marca, linha, tamanho, descricao, imagem, stock, destinatario, deleted
            FROM Perfumaria.perf.produto
            WHERE stock > 0 AND
                  deleted = 0 AND
                  nome LIKE ('%'+ISNULL(@nome, nome)+'%') AND
                  marca = ISNULL(@marca,marca) AND
                  categoria = ISNULL(@categoria,categoria)
            ORDER BY CASE WHEN @orderby='Nome' THEN nome END DESC,
                     CASE WHEN @orderby='Marca' THEN marca END DESC,
                     CASE WHEN @orderby='Categoria' THEN categoria END DESC,
                     CASE WHEN @orderby='Preço' THEN preco END DESC
        END

END
GO
```

```sql
DROP PROCEDURE perf.Login;
GO


CREATE PROCEDURE perf.Login
    @email VARCHAR(255),
    @password VARCHAR(25),
    @responseMessage VARCHAR(250)='' OUTPUT,
    @type BIT=0 OUTPUT
AS
BEGIN
    SET NOCOUNT ON

    IF EXISTS (SELECT TOP 1 email FROM Perfumaria.perf.utilizador WHERE email = @email)
    BEGIN
        SET @email=(SELECT email FROM Perfumaria.perf.utilizador
        WHERE email=@email AND pw=HASHBYTES('SHA2_512', @password))

        IF(@email IS NULL)
        BEGIN
            SET @type=0
            SET @responseMessage='Incorrect password'
        END
        ELSE
        BEGIN
           SET @responseMessage='User successfully logged in'
           IF EXISTS (SELECT TOP 1 email FROM Perfumaria.perf.funcionario WHERE email = @email)
            SET @type = 1
           ELSE
            SET @type = 0
        END
    END
    ELSE
        BEGIN
            SET @type=0
            SET @responseMessage='Invalid login'
        END

END
GO
```

Stored Procedures

```
DROP PROCEDURE perf.addMarc;
GO


CREATE PROCEDURE perf.addMarc
    @cliente_email VARCHAR(255),
    @servico_id INT,
    @funcionario_email VARCHAR(255),
    @dataMarc SMALLDATETIME,
    @responseMessage VARCHAR(250) = 'Erro! Tente noutra hora.' OUTPUT
AS
BEGIN
    SET NOCOUNT ON
    BEGIN TRY
        DECLARE @duracao INT
        SELECT @duracao = duracao_media FROM Perfumaria.perf.funcionario_faz_servico WHERE funcionario_email=@funcionario_email AND deleted = 0 AND  servico_id = @servico_id
            IF (@duracao IS NOT NULL AND @dataMarc > GETDATE())
            BEGIN
                IF (EXISTS(SELECT 1 FROM Perfumaria.perf.marcacao WHERE (dataMarc BETWEEN @dataMarc AND DATEADD(mi, @duracao, @dataMarc)) AND funcionario_email=@funcionario_email AND deleted=0)
                OR EXISTS(SELECT 1 FROM Perfumaria.perf.marcacao WHERE dataMarc BETWEEN @dataMarc AND DATEADD(mi, @duracao, @dataMarc) AND cliente_email=@cliente_email AND deleted=0))
                SET @responseMessage = 'Hora não disponível!'
                ELSE
                BEGIN
                INSERT INTO Perfumaria.perf.marcacao
                (cliente_email, servico_id, funcionario_email, dataMarc)
                VALUES(@cliente_email, @servico_id, @funcionario_email, @dataMarc)

                SET @responseMessage='Marcação efetuado com sucesso!'
                END
            END
    END TRY
    BEGIN CATCH
        SET @responseMessage='Failure'
    END CATCH


END
GO
```

UDF

```
DROP FUNCTION perf.clientFutureMarc;
GO

CREATE FUNCTION perf.clientFutureMarc (@email VARCHAR(255)) RETURNS TABLE
AS
    RETURN (SELECT dataMarc, tipo, preco, fname AS NomeFuncionario
                FROM ((Perfumaria.perf.marcacao JOIN Perfumaria.perf.servico ON servico_id=servico.id) JOIN Perfumaria.perf.utilizador ON funcionario_email = email)
                WHERE (cliente_email=@email AND DATEDIFF(mi, GETDATE(), dataMarc) > 0)
                ORDER BY dataMarc ASC OFFSET 0 ROWS)
GO
```

```
DROP FUNCTION perf.funcSellHistory;
GO

CREATE FUNCTION perf.funcSellHistory (@email VARCHAR(255)) RETURNS TABLE
AS
    RETURN (SELECT datacompra, clienteemail, SUM(unidades * preco) as total, compranumero
                FROM (((Perfumaria.perf.compra_tem_produto JOIN Perfumaria.perf.compra ON compranumero=numero)JOIN Perfumaria.perf.produto ON produtoid=id)
                JOIN Perfumaria.perf.compra_presencial ON compra_presencial.numero=compra.numero)
                WHERE (funcemail=@email)
                GROUP BY datacompra, clienteemail, compranumero
                ORDER BY datacompra DESC OFFSET 0 ROWS)
GO
```

Triggers

```sql
DROP TRIGGER perf.useCuponTrigger;
GO

CREATE TRIGGER perf.useCuponTrigger ON perf.[cliente_usa_cupao]
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @cupao as CHAR(10);
    DECLARE @email as VARCHAR(255);
    DECLARE @pontos as INT;
    SELECT @cupao = cupao_id, @email = cliente_email FROM inserted;
    SELECT @pontos = pontos_atribuidos FROM Perfumaria.perf.cupao WHERE id = @cupao;
    BEGIN TRY
        UPDATE perf.cliente
        SET pontos += @pontos
        WHERE email = @email
    END TRY
    BEGIN CATCH
        raiserror ('Não foi possível atribuir os pontos', 16, 1);
        ROLLBACK TRAN
    END CATCH

END
GO
```

```sql
DROP TRIGGER perf.changeProductTrigger;
GO

CREATE TRIGGER perf.changeProductTrigger ON perf.[produto]
AFTER INSERT, UPDATE
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @stock as INT;
    DECLARE @produtoid as INT;
    SELECT @stock = stock, @produtoid = id FROM inserted;
    IF (@stock < 0)
    BEGIN
        RAISERROR('Stock inválido.', 16, 1);
        ROLLBACK TRAN;
    END
    IF (@stock= 0)
    BEGIN
    BEGIN TRY
        UPDATE Perfumaria.perf.produto
        SET deleted = 1
        WHERE  id = @produtoid
    END TRY
    BEGIN CATCH
        raiserror ('Não foi possível mudar o produto.', 16, 1);
        ROLLBACK TRAN
    END CATCH
    END
END
GO
```

Transações

```
DROP PROCEDURE perf.addNewFunc;
GO

CREATE PROCEDURE perf.addNewFunc
    @email VARCHAR(255),
    @contribuinte CHAR(9),
    @fname VARCHAR(20),
    @lname VARCHAR(20),
    @pw VARCHAR(25),
    @sexo BIT,
    @dataNasc DATETIME,
    @foto VARCHAR(100),
    @contacto_default_id INT = NULL,
    @administrator TINYINT,
    @salario INT,
    @emailFunc VARCHAR(255),
    @responseMessage NVARCHAR(250) OUTPUT
AS
BEGIN
    BEGIN TRANSACTION
    SET NOCOUNT ON
    BEGIN TRY
        IF EXISTS(SELECT email FROM Perfumaria.perf.funcionario WHERE email=@emailFunc AND administrator=2)
            BEGIN
                INSERT INTO Perfumaria.perf.utilizador
                (email, contribuinte, fname, lname, pw, sexo, dataNasc, foto, contacto_default_id)
                VALUES(@email, @contribuinte, @fname, @lname, HASHBYTES('SHA2_512', @pw), @sexo, @dataNasc, @foto, @contacto_default_id)

                INSERT INTO Perfumaria.perf.funcionario
                (email, administrator, salario)
                VALUES(@email, @administrator, @salario)

                SET @responseMessage='Success'
            END
        ELSE
            SET @responseMessage='Permition denied'
        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        SET @responseMessage='Failed'
        ROLLBACK
    END CATCH

END
GO
```

Índices

```
1  DROP INDEX idx_produto on Perfumaria.perf.produto;
2  GO
3
4  CREATE INDEX idx_produto
5  ON Perfumaria.perf.produto (preco, marca, nome, categoria);
6  GO
```