Documentação do Banco de Dados "docesdagio"

Geovanna Moura SP3137465
Miqueias Nascimento SP3131785

08 de fevereiro de 2025

1. Visão Geral

O banco de dados **docesdagio** foi criado para gerenciar informações de uma doceria, incluindo clientes, produtos, pedidos e notas fiscais. Ele possui tabelas inter-relacionadas para armazenar os dados de maneira organizada e eficiente.

2. Estrutura do Banco de Dados

2.1. Tabela `tb_doceria`

Armazena informações da doceria.

- `pk_cnpj` (VARCHAR 20) - Identificação única da doceria (chave primária).
- `nome` (VARCHAR 60) - Nome da doceria.

2.2. Tabela `tb_endereco_doceria`

Guarda informações de endereço da doceria.

- `pk_cep` (VARCHAR 10) - CEP (chave primária).
- `rua` (VARCHAR 100) - Nome da rua.
- `bairro` (VARCHAR 60) - Bairro.
- `numero` (NUMERIC) - Número da residência.
- `complemento` (VARCHAR 100) - Complemento.
- `cidade` (VARCHAR 60) - Cidade.
- `fk_cnpj` (VARCHAR 20) - Chave estrangeira referenciando `tb_doceria(pk_cnpj)`.

2.3. Tabela `users`

Armazena informações dos clientes.

- `id` (INT AUTO_INCREMENT) - Identificação única (chave primária).
- `nome` (VARCHAR 100) - Nome do usuário.
- `email` (VARCHAR 100) - Email (valor único).
- `senha` (VARCHAR 255) - Senha (recomenda-se armazenamento com hash).
- `cpf` (CHAR 14) - CPF do cliente, armazenado de forma informativa.
- `telefone` (VARCHAR 20) - Telefone do cliente.
- `endereco` (VARCHAR 255) - Endereço do cliente.
- `tipo` (ENUM) - Tipo de usuário (admin ou comum).

2.4. Tabela `tb_funcionarios`

Armazena informações dos funcionários da doceria.

- `pk_id_funcionario` (INT AUTO_INCREMENT) - Identificação única (chave primária).
- `nome` (VARCHAR 60) - Nome do funcionário.
- `funcao` (VARCHAR 60) - Função do funcionário.
- `fk_cnpj` (VARCHAR 20) - Chave estrangeira referenciando `tb_doceria(pk_cnpj)`.

2.5. Tabela `tb_produto`

Armazena informações sobre os produtos vendidos na doceria.

- `pk_id_produto` (INT AUTO_INCREMENT) - Identificação única (chave primária).
- `nome_produto` (VARCHAR 60) - Nome do produto.
- `valor_produto` (FLOAT) - Preço do produto.
- `qntd_estoque_produto` (NUMERIC) - Quantidade em estoque.
- `descricao_produto` (VARCHAR 200) - Descrição do produto.

2.6. Tabela `tb_pedido`

Armazena informações dos pedidos realizados pelos clientes.

- `pk_id_pedido` (INT AUTO_INCREMENT) - Identificação única (chave primária).
- `fk_id_cliente` (INT) - Chave estrangeira referenciando `users(id)`.
- `fk_cnpj` (VARCHAR 20) - Chave estrangeira referenciando `tb_doceria(pk_cnpj)`.
- `data_hora` (DATETIME) - Data e hora do pedido (padrão: `GETDATE()`).

2.7. Tabela `tb_itens_pedidos`

Armazena os itens dentro dos pedidos.

- `pk_itens_pedidos` (INT AUTO_INCREMENT) - Identificação única (chave primária).
- `fk_id_cliente` (INT) - Chave estrangeira referenciando `users(id)`.
- `fk_id_produto` (INT) - Chave estrangeira referenciando `tb_produto(pk_id_produto)`.
- `fk_id_pedido` (INT) - Chave estrangeira referenciando `tb_pedido(pk_id_pedido)`.
- `subtotal_itens_pedidos` (FLOAT) - Subtotal do item.
- `data_hora` (DATETIME) - Data e hora do pedido.

2.8. Tabela `tb_nota_fiscal`

Armazena os dados da nota fiscal gerada.

- `pk_nota_fiscal` (INT AUTO_INCREMENT) - Identificação única (chave primária).
- `fk_itens_pedidos` (INT) - Chave estrangeira referenciando `tb_itens_pedidos(pk_itens_pedidos)`.
- `fk_cnpj` (VARCHAR 20) - Chave estrangeira referenciando `tb_doceria(pk_cnpj)`.
- `fk_id_cliente` (INT) - Chave estrangeira referenciando `users(id)`.

3. Procedures e Funções

3.1. Procedure `PreencherContato`

Preenche a tabela `tb_contato` automaticamente com base nos dados da tabela `users`.

```
CREATE PROCEDURE PreencherContato
AS
BEGIN
   SET NOCOUNT ON;
   INSERT INTO tb_contato (telefone, email, fk_id_cliente)
   SELECT telefone, email, id
   FROM users
   WHERE id NOT IN (
      SELECT fk_id_cliente FROM tb_contato
   );
END;
```

3.2. Procedure `PreencherEndereco`

Preenche a tabela `tb_endereco` com base nos dados da tabela `users`.

```
CREATE PROCEDURE PreencherEndereco
AS
BEGIN
   SET NOCOUNT ON;
   INSERT INTO tb_endereco (endereco, fk_id_cliente)
   SELECT endereco, id
   FROM users
   WHERE id NOT IN (
      SELECT fk_id_cliente FROM tb_endereco
   );
END;
```

3.3. Função `ObterContato`

Retorna os contatos de um cliente específico.

```
CREATE FUNCTION ObterContato(@id_cliente INT)
RETURNS TABLE
AS
RETURN (
   SELECT * FROM tb_contato WHERE fk_id_cliente = @id_cliente
);
```

3.4. Função `ObterEndereco`

Retorna o endereço de um cliente específico.

```
CREATE FUNCTION ObterEndereco(@id_cliente INT)
RETURNS TABLE
AS
RETURN (
   SELECT * FROM tb_endereco WHERE fk_id_cliente = @id_cliente
);
```

4. Código completo

```
CREATE DATABASE docesdagio;
USE docesdagio;


CREATE TABLE tb_doceria (
   pk_cnpj VARCHAR(20) PRIMARY KEY,
   nome VARCHAR(60)
);


CREATE TABLE tb_endereco_doceria (
   pk_cep VARCHAR(10) PRIMARY KEY,
   rua VARCHAR(100),
   bairro VARCHAR(60),
   numero NUMERIC,
   complemento VARCHAR(100),
   cidade VARCHAR(60),
   fk_cnpj VARCHAR(20),
   FOREIGN KEY (fk_cnpj) REFERENCES tb_doceria(pk_cnpj)
);


CREATE TABLE users (
   id INT AUTO_INCREMENT PRIMARY KEY,
   nome VARCHAR(100) NOT NULL,
```

```
    email VARCHAR(100) UNIQUE NOT NULL,
    senha VARCHAR(255) NOT NULL,
    cpf CHAR(14) UNIQUE NOT NULL,
    telefone VARCHAR(20) NOT NULL,
    endereco VARCHAR(255) NOT NULL,
    tipo ENUM('admin', 'comum') DEFAULT 'comum' -- Adicionando nível de acesso
);


CREATE TABLE tb_contato (
    telefone VARCHAR(20),
    email VARCHAR(100),
    fk_id_user INT,
    FOREIGN KEY (fk_id_user) REFERENCES users(id)
);


CREATE TABLE tb_endereco (
    endereco VARCHAR(255),
    fk_id_user INT,
    FOREIGN KEY (fk_id_user) REFERENCES users(id)
);


CREATE TABLE tb_funcionarios (
    pk_id_funcionario INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(60),
    funcao VARCHAR(60),
    fk_cnpj VARCHAR(20),
    FOREIGN KEY (fk_cnpj) REFERENCES tb_doceria(pk_cnpj)
);


CREATE TABLE tb_produto (
    pk_id_produto INT AUTO_INCREMENT PRIMARY KEY,
    nome_produto VARCHAR(60),
    valor_produto FLOAT,
    qntd_estoque_produto NUMERIC,
    descricao_produto VARCHAR(200)
);


CREATE TABLE tb_pedido (
    pk_id_pedido INT AUTO_INCREMENT PRIMARY KEY,
    fk_id_user INT,
    fk_cnpj VARCHAR(20),
    FOREIGN KEY (fk_id_user) REFERENCES users(id),
    FOREIGN KEY (fk_cnpj) REFERENCES tb_doceria(pk_cnpj)
```

```
);


CREATE TABLE tb_itens_pedidos (
    pk_itens_pedidos INT AUTO_INCREMENT PRIMARY KEY,
    fk_id_user INT,
    fk_id_produto INT,
    fk_id_pedido INT,
    subtotal_itens_pedidos FLOAT,
    data_hora DATETIME,
    FOREIGN KEY (fk_id_user) REFERENCES users(id),
    FOREIGN KEY (fk_id_produto) REFERENCES tb_produto(pk_id_produto),
    FOREIGN KEY (fk_id_pedido) REFERENCES tb_pedido(pk_id_pedido)
);


CREATE TABLE tb_nota_fiscal (
    pk_nota_fiscal INT AUTO_INCREMENT PRIMARY KEY,
    fk_itens_pedidos INT,
    fk_cnpj VARCHAR(20),
    fk_id_user INT,
    FOREIGN KEY (fk_itens_pedidos) REFERENCES tb_itens_pedidos(pk_itens_pedidos),
    FOREIGN KEY (fk_cnpj) REFERENCES tb_doceria(pk_cnpj),
    FOREIGN KEY (fk_id_user) REFERENCES users(id)
);


CREATE PROCEDURE PreencherContato()
BEGIN
    INSERT INTO tb_contato (telefone, email, fk_id_user)
    SELECT telefone, email, id
    FROM users
    WHERE id NOT IN (
        SELECT fk_id_user FROM tb_contato
    );
END;


CREATE PROCEDURE PreencherEndereco()
BEGIN
    INSERT INTO tb_endereco (endereco, fk_id_user)
    SELECT endereco, id
    FROM users
    WHERE id NOT IN (
        SELECT fk_id_user FROM tb_endereco
    );
END;
```

```
CREATE FUNCTION ObterContato(id_user INT)
RETURNS TABLE
RETURN (
   SELECT *
   FROM tb_contato
   WHERE fk_id_user = id_user
);


CREATE FUNCTION ObterEndereco(id_user INT)
RETURNS TABLE
RETURN (
   SELECT *
   FROM tb_endereco
   WHERE fk_id_user = id_user
);

CALL PreencherContato();
CALL PreencherEndereco();
```

5. Considerações Finais

Esta documentação fornece uma visão detalhada da estrutura do banco de dados **docesdagio**, suas tabelas, chaves, relacionamentos, procedures e funções utilizadas. O banco foi projetado para ser eficiente, garantindo a integridade dos dados e a manutenção simplificada das informações da doceria.