CESMAC
CENTRO UNIVERSITÁRIO

SI
SISTEMAS DE INFORMAÇÃO

DOCUMENTAÇÃO DE BANCO DE DADOS

Documentação elaborada para composição da nota da Disciplina de Projeto de Banco de Dados, orientado pelo profª Tacyana Batista.

Sumário

Minimundo

O EasyOrder é um sistema de cardápio digital integrado para restaurantes, com foco em otimizar o atendimento e proporcionar uma experiência moderna aos clientes.

No restaurante, cada mesa possui um QR Code exclusivo. Quando o cliente escaneia esse código com seu smartphone, é automaticamente direcionado ao sistema, que reconhece de qual mesa o pedido será feito.

Antes de realizar o pedido, o cliente precisa se cadastrar com informações básicas (nome, e-mail ou telefone, senha e data de nascimento). Esse cadastro permite que o sistema registre seu histórico de pedidos, atribua pontos de fidelidade e personalize a experiência.

O cardápio digital é organizado em categorias (entradas, pratos principais, sobremesas, bebidas, etc.), e cada item do cardápio possui descrição, preço, custo, tempo de preparo estimado, imagem e possibilidade de personalização (como adicionar/remover ingredientes).

Durante a realização do pedido, o cliente pode:

- Selecionar múltiplos itens;
- Adicionar observações ou personalizações;
- Revisar e confirmar o pedido;
- Realizar o pagamento pelo próprio aplicativo (cartão, Pix, carteiras digitais) ou deixar para pagar na mesa.

Assim que o pedido é confirmado, ele é enviado automaticamente para o sistema da cozinha, que organiza e prioriza os pedidos. O cliente recebe notificações em tempo real sobre o andamento (pedido recebido, em preparo, pronto e entregue).

O sistema também coleta feedbacks e avaliações após a refeição, registrando comentários e notas de 1 a 5, para que o restaurante possa melhorar seu atendimento e cardápio.

No lado administrativo, os gestores do restaurante têm acesso a:

- Gerenciamento completo do cardápio;
- Controle de estoque, com movimentações de entrada, saída e ajustes;
- Registro de promoções e programas de fidelidade;
- Relatórios de vendas, consumo por categoria e preferências de clientes;

Benefícios para o restaurante:

- Redução de erros e tempo de atendimento;
- Melhor controle de estoque e pedidos;
- Relatórios estratégicos para tomada de decisão;
- Sistema de fidelidade que aumenta o engajamento dos clientes.

Benefícios para o cliente:

- Autonomia e agilidade no pedido;
- Facilidade no pagamento digital;
- Transparência com notificações de status;
- Participação em promoções e programas de fidelidade.

MER (Modelo de Entidade e Relacionamento)

O Modelo de Entidade e Relacionamento (MER) representa a estrutura lógica do banco de dados, definindo as entidades, seus atributos e como elas se conectam.

Entidades e Atributos

usersAdmin (Equipe)

Representa os usuários administrativos do sistema (Dono, Gerente, etc.).

- **idUser**: SERIAL PRIMARY KEY
- **name**: VARCHAR(255) NOT NULL
- **email**: VARCHAR(255) UNIQUE NOT NULL
- **password**: VARCHAR(255) NOT NULL
- **type**: ENUM('dono', 'gerente', 'garcom', 'cozinha') NOT NULL
- **active**: BOOLEAN DEFAULT TRUE
- **dateRegister**: TIMESTAMP DEFAULT CURRENT_TIMESTAMP
- **lastLogin**: TIMESTAMP

clients (Clientes)

Representa os consumidores finais que realizam os pedidos.

- **idClient**: SERIAL PRIMARY KEY
- **name**: VARCHAR(255) NOT NULL
- **email**: VARCHAR(255) UNIQUE (Opcional se tiver telefone)
- **phone**: VARCHAR(20) UNIQUE (Opcional se tiver email)
- **password**: VARCHAR(255) NOT NULL
- **dateRegister**: TIMESTAMP DEFAULT CURRENT_TIMESTAMP
- **loyaltyPoints**: INT DEFAULT 0

desks (Mesas)

Representa as mesas físicas do estabelecimento.

- **idDesk**: SERIAL PRIMARY KEY
- **deskNumber**: INT UNIQUE NOT NULL
- **qrCodeUid**: VARCHAR(50) UNIQUE NOT NULL
- **capacity**: INT NOT NULL
- **condition**: ENUM('livre', 'ocupada', 'reservada', 'manutencao') DEFAULT 'livre'

menuItems (Cardápio)

Itens disponíveis para venda.

- **idItem**: SERIAL PRIMARY KEY
- **name**: VARCHAR(100) NOT NULL
- **description**: TEXT
- **price**: DECIMAL(10,2) NOT NULL
- **cost**: DECIMAL(10,2) (Custo de produção)
- **timePreparation**: INT (Minutos estimados)
- **emphasis**: BOOLEAN DEFAULT FALSE (Destaque no cardápio)
- **active**: BOOLEAN DEFAULT TRUE
- **imagePath**: VARCHAR(255) (URL da imagem)
- **options**: JSON (Opções de personalização)
- **category**: VARCHAR(50) (Categoria do item: Lanches, Bebidas, etc.)

orders (Pedidos)

Cabeçalho do pedido, vinculando cliente e mesa.

- **idOrder**: SERIAL PRIMARY KEY
- **idClient**: INT (FK -> clients)
- **idDesk**: INT (FK -> desks)
- **timeDate**: TIMESTAMP DEFAULT CURRENT_TIMESTAMP
- **status**: ENUM('recebido', 'preparando', 'pronto', 'entregue', 'cancelado') DEFAULT 'recebido'
- **total**: DECIMAL(10,2) NOT NULL
- **observation**: TEXT
- **origin**: ENUM('app', 'balcao', 'telefone') DEFAULT 'app'

orderItems (Itens do Pedido)

Tabela associativa que detalha os produtos de cada pedido.

- **idOrderItem**: SERIAL PRIMARY KEY
- **idOrder**: INT NOT NULL (FK -> orders)
- **idItem**: INT NOT NULL (FK -> menuItems)
- **amount**: INT DEFAULT 1 NOT NULL
- **unitPrice**: DECIMAL(10,2) NOT NULL (Preço no momento da venda)
- **custom**: JSON (Escolhas do cliente)
- **observation**: TEXT (Obs específica do item)

DER (Diagrama de Entidade e Relacionamento)

Esta seção descreve textualmente o diagrama, detalhando o significado de cada relacionamento e como ele é implementado através das chaves primárias (PK) e estrangeiras (FK).

Relacionamento Cliente-Pedido:

- **Regra:** Um clients realiza N orders.
- **Descrição:** Associa um pedido a quem o fez, permitindo histórico de pedidos e, futuramente, programas de fidelidade.
- **Implementação:** A chave primária clients.idClient (PK) é usada como chave estrangeira orders.idClient(FK).

Relacionamento Mesa-Pedido:

- **Regra:** Uma desks sedia N orders.
- **Descrição:** Identifica a localização física de um pedido dentro do restaurante, essencial para a entrega pela equipe.
- **Implementação:** A chave primária desks.idDesk (PK) é usada como chave estrangeira orders.idDesk (FK).

Relacionamento Pedido-Item (N:M):

- **Regra:** Um orders contém M menuItems.
- **Descrição:** É o relacionamento central que forma o pedido. Como um pedido pode ter vários pratos e um prato pode estar em vários pedidos, utiliza-se uma entidade associativa.
- **Implementação:** A tabela associativa orderItems conecta as duas pontas. Ela possui duas chaves estrangeiras: idOrder (referenciando orders) e idItem (referenciando menuItems). Além disso, armazena atributos específicos daquela relação, como amount (quantidade), unitPrice (preço no momento da venda) e observation(customizações).

Scripts DDL (Data Definition Language)

A seguir estão os scripts SQL para criar todas as tabelas do banco de dados EasyOrder.

```
CREATE TYPE userType AS ENUM ('dono', 'gerente', 'garcom', 'cozinha');

CREATE TYPE deskCondition AS ENUM ('livre', 'ocupada', 'reservada',
'manutencao');

CREATE TYPE orderStatus AS ENUM ('recebido', 'preparando', 'pronto',
'entregue', 'cancelado');

CREATE TYPE orderOrigin AS ENUM ('app', 'balcao', `telefone');


CREATE TABLE clientes (
    idCliente INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) UNIQUE,
    telefone VARCHAR(20) UNIQUE NOT NULL,
    password VARCHAR(255),
    birthData DATE,
    dateRegister DATETIME DEFAULT CURRENT_TIMESTAMP,
    loyaltyPoints INT DEFAULT 0
);

CREATE TABLE desks (
    idDesk INT AUTO_INCREMENT PRIMARY KEY,
    deskNumber VARCHAR(10) UNIQUE NOT NULL,
    qrcodeUld VARCHAR(50) UNIQUE NOT NULL,
    capacity INT NOT NULL,
    `condition` ENUM('livre', 'ocupada', 'reservada', 'manutencao') DEFAULT
            'livre'
);

CREATE TABLE menuItens (
    idItem INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    descricao TEXT,
    price DECIMAL(10,2) NOT NULL,
    cost DECIMAL(10,2),
    timePreparo INT,
    emphasis BOOLEAN DEFAULT FALSE,
    activo BOOLEAN DEFAULT TRUE,
    imagePath VARCHAR(255),
    options JSON,
    category VARCHAR(50) DEFAULT 'GERAL'
);

CREATE TABLE orders (
    idOrder INT AUTO_INCREMENT PRIMARY KEY,
    idCliente INT,
    idDesk INT NOT NULL,
    timeData DATETIME DEFAULT CURRENT_TIMESTAMP,
    status ENUM('recebido', 'preparando', 'pronto', 'entregue', 'cancelado')
            DEFAULT 'recebido',
    total DECIMAL(10,2) NOT NULL,
    observacao TEXT,
    FOREIGN KEY (idCliente) REFERENCES clientes(idCliente),
    FOREIGN KEY (idDesk) REFERENCES desks(idDesk)
);
```

```
CREATE TABLE itemOrder (
    idItemOrder INT AUTO_INCREMENT PRIMARY KEY,
    idOrder INT NOT NULL,
    idItem INT NOT NULL,
    amount INT NOT NULL DEFAULT 1,
    unityPrice DECIMAL(10,2) NOT NULL,
    custom JSON,
    observacao TEXT,
    FOREIGN KEY (idOrder) REFERENCES orders(idOrder) ON DELETE CASCADE,
    FOREIGN KEY (idItem) REFERENCES menuItens(idItem)
);
```

Scripts DML (Data Manipulation Language)

Esta seção apresenta os comandos utilizados para inserir, atualizar e manipular os dados no sistema EasyOrder.

1. Carga Inicial de Dados (Inserts)

Cadastrando a Equipe (Admin):

```
INSERT INTO usersAdmin (name, email, password, type, active)
VALUES
    ('Luiz Dono', 'dono@easyorder.com', '$2b$12$ExampleHash...', 'dono',
TRUE),
    ('Carlos Gerente', 'gerente@easyorder.com', '$2b$12$ExampleHash...',
'gerente', TRUE),
    ('Equipe Cozinha', 'cozinha@easyorder.com', '$2b$12$ExampleHash...',
'cozinha', TRUE);
```

Cadastrando Clientes (Exemplos de Email e Telefone):

```
INSERT INTO clients (name, email, phone, password)
VALUES
    ('Maria Silva', 'maria@gmail.com', '82999991111',
'$2b$12$ExampleHash...'),
    ('João Souza', NULL, '82988882222', '$2b$12$ExampleHash...'); -- Cliente
só com telefone
```

Cadastrando Mesas:

```
INSERT INTO desks (deskNumber, qrCodeUid, capacity, condition)
VALUES
    (1, 'uuid-mesa-1', 4, 'livre'),
    (2, 'uuid-mesa-2', 2, 'ocupada'),
    (3, 'uuid-mesa-3', 6, 'reservada');
```

Cadastrando o Cardápio (Com Categorias e Opções JSON):

```
INSERT INTO menuItems (name, description, price, cost, timePreparation,
category, imagePath, options, active)
VALUES
    -- Lanche
    ('X-Burger da Casa', 'Pão brioche, carne 180g, queijo e salada.', 28.90,
12.00, 15, 'Lanches',
     'https://firebasestorage.googleapis.com/.../xburger.jpg',
     '{"opcoes_obrigatorias": [{"titulo": "Ponto", "itens": ["Ao Ponto", "Bem
Passado"]}], "adicionais": [{"titulo": "Bacon", "preco_extra": 3.00}]}',
     TRUE),

    -- Bebida
    ('Suco de Laranja', 'Natural, 500ml.', 10.00, 3.50, 5, 'Bebidas',
     'https://firebasestorage.googleapis.com/.../suco.jpg',
     '{"adicionais": [{"titulo": "Com Gelo", "preco_extra": 0}, {"titulo":
"Sem Açúcar", "preco_extra": 0}]}',
     TRUE);
```

2. Simulação de um Pedido (Transação)

O fluxo de criar um pedido envolve inserir na tabela orders e depois na orderItems.

1. Criar o Pedido (Cabeçalho):

```
INSERT INTO orders (idClient, idDesk, total, observation, status, origin)
VALUES (1, 2, 38.90, 'Entregar tudo junto', 'recebido', 'app');
-- Suponha que gerou o ID: 10
```

2. Adicionar os Itens do Pedido:

```
INSERT INTO orderItems (idOrder, idItem, amount, unitPrice, custom,
observation)
VALUES
    (10, 1, 1, 28.90, '{"Ponto": "Ao Ponto", "Adicionais": ["Bacon"]}', 'Sem
cebola'), -- 1x X-Burger
    (10, 2, 1, 10.00, '{"Com Gelo": true}', NULL); -- 1x Suco
```

3. Atualizações de Rotina (Updates)

Cozinha atualizando o status do pedido:

```
UPDATE orders
SET status = 'preparando'
WHERE idOrder = 10;
```

Garçom liberando uma mesa:

```
UPDATE desks
SET condition = 'livre'
WHERE deskNumber = 2;
```

Gerente alterando o preço de um prato:

```
UPDATE menuItems
SET price = 32.90
WHERE idItem = 1;
```

4. Exclusão Lógica (Soft Delete)

Para manter a integridade dos dados e o histórico financeiro, o sistema não remove registros fisicamente.

Desativar um item do cardápio (em vez de deletar):

```
UPDATE menuItems
SET active = FALSE
WHERE idItem = 1;
```

Desativar um funcionário:

```
UPDATE usersAdmin
SET active = FALSE
WHERE idUser = 5;
```

Scripts DQL (Data Query Language)

Abaixo estão as principais consultas utilizadas pelo sistema EasyOrder para autenticação, operação da cozinha e relatórios gerenciais.

1. Autenticação e Usuários

Buscar usuário administrativo para Login (Dono/Gerente/Cozinha):

```
SELECT idUser, name, password, type
FROM usersAdmin
WHERE email = 'email@exemplo.com' AND active = TRUE;
```

Buscar cliente por Email ou Telefone (Login unificado):

```
SELECT idClient, name, password, loyaltyPoints
FROM clients
WHERE (email = 'cliente@email.com' OR phone = '82999998888')
AND active = TRUE;
```

2. Operação do Cardápio (Cliente)

Listar cardápio ativo organizado por categoria e nome:

```
SELECT idItem, name, description, price, imagePath, category
FROM menuItems
WHERE active = TRUE
ORDER BY category ASC, name ASC;
```

Buscar detalhes de um item específico (incluindo opções de personalização):

```
SELECT name, description, price, timePreparation, options
FROM menuItems
WHERE idItem = 1;
```

3. Gestão de Pedidos (Cozinha e Admin)

Listar pedidos ativos para a Cozinha (Painel Kanban): *Retorna os pedidos que ainda não foram entregues ou cancelados, com o nome da mesa e do cliente.*

```
SELECT
    o.idOrder,
    o.status,
    o.total,
    o.timeDate,
    o.observation,
    d.deskNumber,
    c.name AS clientName
FROM orders o
LEFT JOIN desks d ON o.idDesk = d.idDesk
LEFT JOIN clients c ON o.idClient = c.idClient
WHERE o.status NOT IN ('entregue', 'cancelado')
ORDER BY o.timeDate DESC;
```

Detalhar itens de um pedido específico:

```
SELECT
    mi.name AS Item,
    oi.amount AS Qtd,
    oi.unitPrice AS PrecoUnitario,
    oi.observation AS Obs,
    oi.custom AS Personalizacao
FROM orderItems oi
JOIN menuItems mi ON oi.idItem = mi.idItem
WHERE oi.idOrder = 10; -- Exemplo: Pedido #10
```

4. Relatórios Gerenciais (Dashboard)

Faturamento total e contagem de pedidos (Dashboard do Dono):

```
SELECT
    COUNT(*) AS totalOrders,
    COALESCE(SUM(total), 0) AS totalRevenue
FROM orders
WHERE status != 'cancelado';
```

Top 5 itens mais vendidos:

```
SELECT
    mi.name,
    SUM(oi.amount) AS totalVendido
FROM orderItems oi
JOIN menuItems mi ON oi.idItem = mi.idItem
GROUP BY mi.name
ORDER BY totalVendido DESC
LIMIT 5;
```

Histórico de pedidos de um cliente:

```
SELECT
    mi.name,
    SUM(oi.amount) AS totalVendido
FROM orderItems oi
JOIN menuItems mi ON oi.idItem = mi.idItem
GROUP BY mi.name
ORDER BY totalVendido DESC
LIMIT 5;
```

Conclusão

O presente projeto teve como objetivo a concepção e a estruturação de um banco de dados relacional para o sistema **"EasyOrder"**, uma solução de cardápio digital que visa modernizar e otimizar o atendimento em restaurantes. A finalidade foi criar uma base de dados robusta, íntegra e escalável, capaz de suportar todas as funcionalidades principais do sistema, desde o cadastro de clientes e a navegação no cardápio até a realização de pedidos e o acompanhamento de status em tempo real.

A construção do modelo seguiu as etapas fundamentais do projeto de banco de dados. Iniciou-se com a definição do Minimundo, que estabeleceu o escopo e as regras de negócio. A partir disso, foi elaborado o **Modelo de Entidade e Relacionamento (MER)**, que definiu conceitualmente as entidades, seus atributos e as relações entre elas. A representação gráfica deste modelo, o **Diagrama de Entidade e Relacionamento (DER)**, permitiu a visualização clara da estrutura lógica do banco de dados, que foi posteriormente traduzida para a linguagem de definição de dados (DDL) com a criação dos scripts SQL para o PostgreSQL.

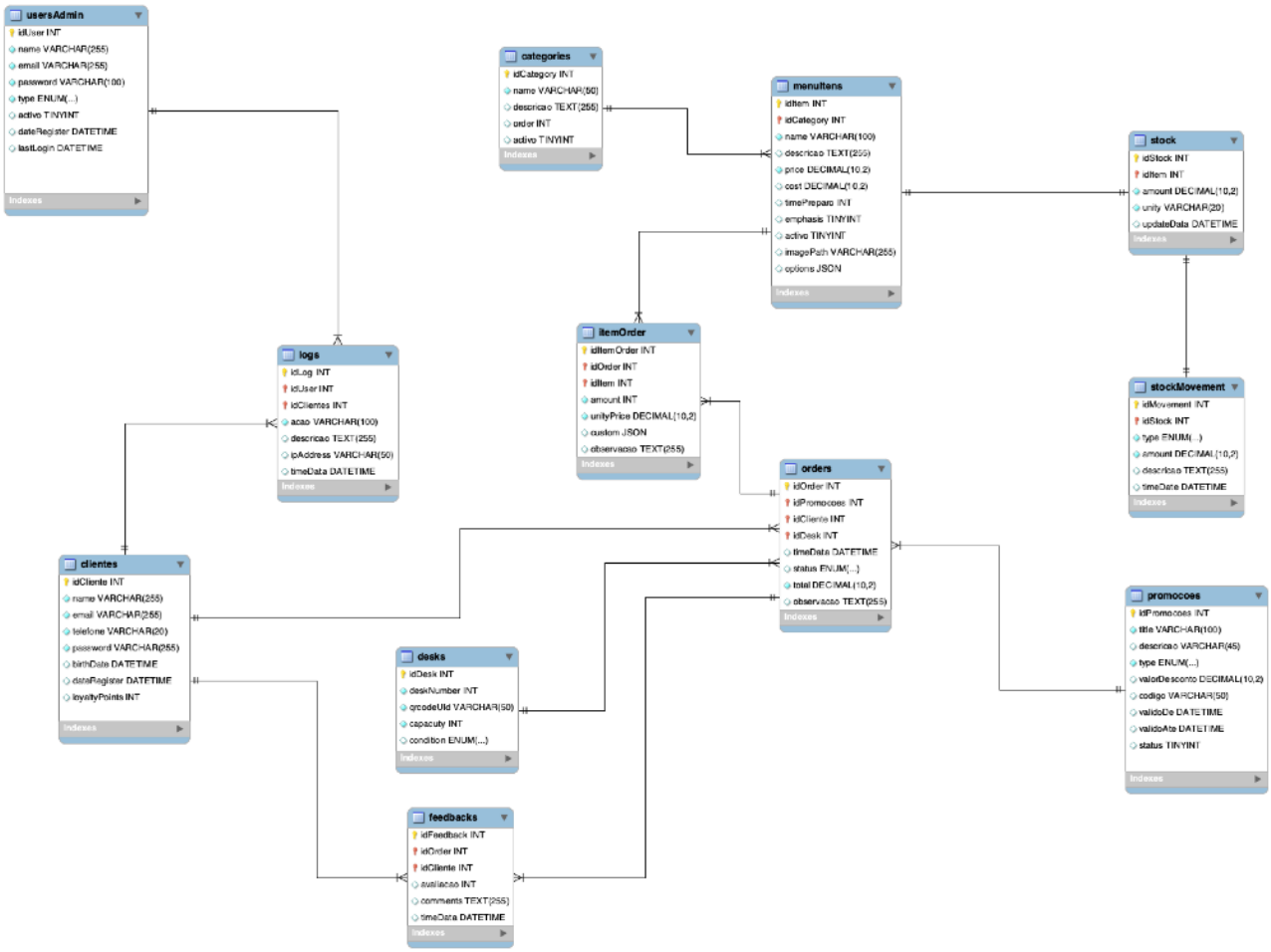
O modelo resultante preza pela integridade e consistência dos dados, utilizando chaves primárias para identificar unicamente cada registro e chaves estrangeiras para garantir a coesão entre as tabelas relacionadas, como a ligação entre clients e orders. A estrutura foi projetada seguindo os princípios de normalização para evitar redundância e anomalias de dados. A resolução de relacionamentos muitos-para-muitos (N:M), como o que existe entre orders e menuItems, foi implementada de forma eficaz através da entidade associativa orderItems.

Um dos desafios e decisões importantes durante a fase de modelagem foi a estratégia de preservação de dados. Optou-se por implementar a **Exclusão Lógica** (*Soft Delete*) através do campo active nas tabelas principais. Essa decisão de design garante que, mesmo que um prato saia do cardápio ou um funcionário seja desligado, o histórico financeiro e os relatórios de pedidos anteriores permaneçam intactos e consistentes, prevenindo a criação de registros órfãos.

A execução deste trabalho permitiu a aplicação prática de conceitos teóricos essenciais da disciplina de Projeto de Banco de Dados, reforçando a importância de uma modelagem cuidadosa antes da implementação física. Como próximos passos ou trabalhos futuros, o modelo atual serve como uma base sólida para a expansão do sistema, podendo incorporar facilmente as tabelas de estoque, avaliações (feedbacks) e o módulo de promoções originalmente planejado.

Em suma, o banco de dados do "EasyOrder" constitui uma fundação sólida, eficiente e escalável, pronta para suportar as operações do sistema e agregar valor ao negócio, melhorando tanto a eficiência operacional do restaurante quanto a experiência do cliente.

ANEXOS


usersAdmin
idUser INT
name VARCHAR(255)
email VARCHAR(255)
password VARCHAR(100)
type ENUM(...)
ativo TINYINT
dateRegister DATETIME
lastLogin DATETIME
Indexes
categories
idCategory INT
name VARCHAR(50)
descricao TEXT(255)
order INT
ativo TINYINT
Indexes
menuItens
idItem INT
idCategory INT
name VARCHAR(100)
descricao TEXT(255)
price DECIMAL(10,2)
cost DECIMAL(10,2)
timePreparo INT
emphasis TINYINT
ativo TINYINT
imagePath VARCHAR(255)
options JSON
Indexes
stock
idStock INT
idItem INT
amount DECIMAL(10,2)
unity VARCHAR(20)
updateData DATETIME
Indexes
logs
idLog INT
idUser INT
idClientes INT
acao VARCHAR(100)
descricao TEXT(255)
ipAddress VARCHAR(50)
timeData DATETIME
Indexes
itemOrder
idItemOrder INT
idOrder INT
idItem INT
amount INT
unityPrice DECIMAL(10,2)
custom JSON
observacao TEXT(255)
Indexes
stockMovement
idMovement INT
idStock INT
type ENUM(...)
amount DECIMAL(10,2)
descricao TEXT(255)
timeData DATETIME
Indexes
orders
idOrder INT
idPromocoes INT
idCliente INT
idDesk INT
timeData DATETIME
status ENUM(...)
total DECIMAL(10,2)
observacao TEXT(255)
Indexes
clientes
idCliente INT
name VARCHAR(255)
email VARCHAR(255)
telefone VARCHAR(20)
password VARCHAR(255)
birthDate DATETIME
dateRegister DATETIME
loyaltyPoints INT
Indexes
desks
idDesk INT
deskNumber INT
qrcodeUid VARCHAR(50)
capacuty INT
condition ENUM(...)
Indexes
promocoes
idPromocoes INT
title VARCHAR(100)
descricao VARCHAR(45)
type ENUM(...)
valorDesconto DECIMAL(10,2)
codigo VARCHAR(50)
validoDe DATETIME
validoAte DATETIME
status TINYINT
Indexes
feedbacks
idFeedback INT
idOrder INT
idCliente INT
avaliacao INT
comments TEXT(255)
timeData DATETIME
Indexes