**FACULDADE SENAI “GASPAR RICARDO JÚNIOR”**

**CFP 402**

**PROJETO FINAL: BANCO DE DADOS RELACIONAL**

**CAUÃ RODRIGUES MESQUITA**

**CAIQUE LEANDRO TESSAROTO**

**JOÃO PEDRO AMERICO MATIAS**

**LUCAS IANOVSKI**

**SOROCABA**

**2025**

**FACULDADE SENAI “GASPAR RICARDO JÚNIOR”**

**CFP 402**

**PROJETO FINAL: BANCO DE DADOS RELACIONAL**

**CAUÃ RODRIGUES MESQUITA**

**CAIQUE LEANDRO TESSAROTO**

**JOÃO PEDRO AMERICO MATIAS**

**LUCAS IANOVSKI**

Tecnólogo em Análise e Desenvolvimento de Sistemas, 1º semestre.

Trabalho solicitado no componente curricular de banco de dados.

**Sorocaba, 20 de fevereiro de 2025.**

**SUMÁRIO**

[1. INTRODUÇÃO 3](#_Toc710487518)

[2. MODELAGEM CONCEITUAL 4](#_Toc1942567907)

[3. MODELAGEM LÓGICA 5](#_Toc1193329309)

[4. ESTRUTURA DO BANCO DE DADOS 6](#_Toc1168704273)

[5. MANIPULAÇÃO DE DADOS 7](#_Toc400287904)

[6. CONTROLE DE ACESSO 8](#_Toc1499610188)

[7. CONTROLE DE TRANSAÇÕES 9](#_Toc216056967)

[8. CONCLUSÃO 10](#_Toc77031322)

[9. REFERÊNCIAS 11](#_Toc1068700223)

## **1. INTRODUÇÃO**

Este trabalho tem como objetivo apresentar o desenvolvimento de um sistema de banco de dados relacional para uma farmácia, contemplando todas as etapas do processo de modelagem, normalização e implementação. O projeto foi desenvolvido com base em um cenário realista, no qual uma farmácia necessita de uma estrutura eficiente para armazenar e gerenciar informações sobre clientes, produtos, pedidos, vendas e controle de estoque.  
  
Para atender a essas necessidades, foi realizado inicialmente o levantamento dos requisitos essenciais do sistema, seguido da modelagem conceitual utilizando o Diagrama Entidade-Relacionamento (DER), que representou graficamente todas as entidades, atributos e relacionamentos identificados. O modelo resultante foi posteriormente transformado em um modelo lógico, com a definição das tabelas, chaves primárias, estrangeiras e demais restrições necessárias.  
O banco de dados foi implementado no SGBD PostgreSQL, com a criação de scripts em SQL divididos entre DDL (Data Definition Language), DML (Data Manipulation Language) e DQL (Data Query Language).  
  
Todas as decisões de projeto foram tomadas com base em prãticas recomendadas de modelagem relacional e organização de dados, permitindo a construção de um sistema coeso, funcional e preparado para ser utilizado em um ambiente de negócio real.

## **2. MODELAGEM CONCEITUAL**

A modelagem conceitual tem como objetivo representar de forma abstrata os principais dados que compõem o sistema de gerenciamento de uma farmácia, bem como os relacionamentos existentes entre eles. Esta modelagem foi realizada através do Diagrama Entidade-Relacionamento (DER), apresentado na Figura 1.

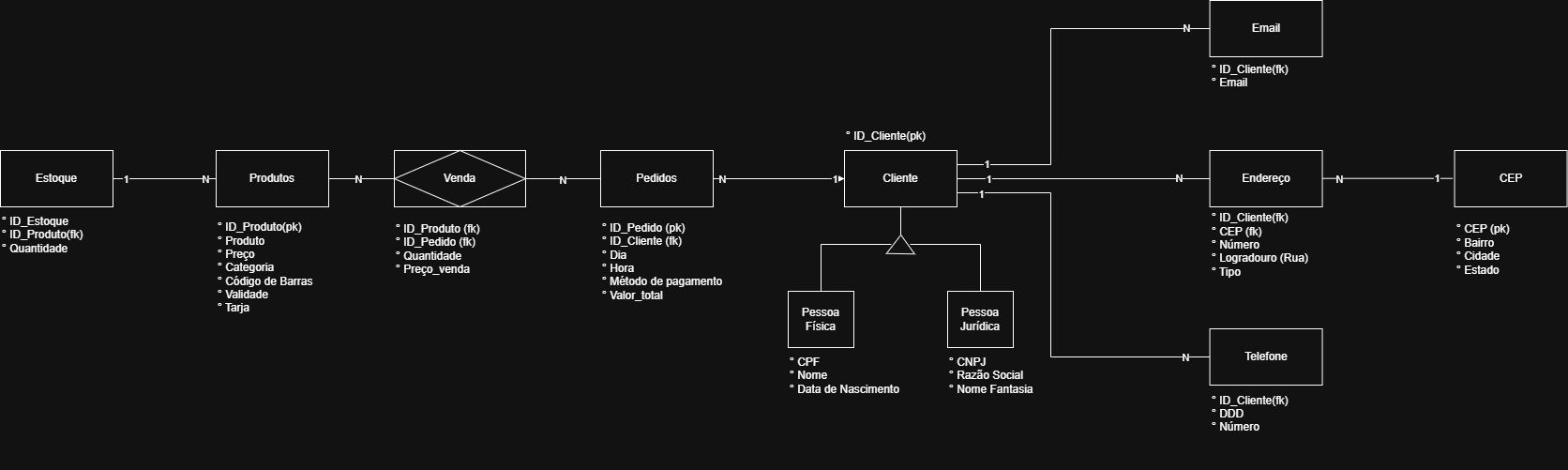
O sistema contempla as seguintes entidades:

* **Produto:** Armazena informações dos produtos comercializados, incluindo nome, preço, categoria, código de barras, validade e tarja, além de possuir uma identificação única (*ID\_Produto*).
* **Estoque:** Controla a quantidade disponível de cada produto, associando-se diretamente à entidade *Produto* por meio de um relacionamento de cardinalidade 1:N, permitindo que cada produto possua seu controle de quantidade.
* **Pedido:** Representa uma transação de compra realizada por um cliente. Esta entidade registra informações como data, hora, método de pagamento e valor total da venda. É identificada por um *ID\_Pedido*.
* **Venda:** Atua como uma entidade associativa entre *Pedido* e *Produto*, permitindo registrar quais produtos foram vendidos em determinado pedido, incluindo a quantidade e o preço de venda de cada item.
* **Cliente:** Armazena os dados gerais dos clientes, possuindo uma chave primária (*ID\_Cliente*). Esta entidade se especializa em *Pessoa Física* e *Pessoa Jurídica*, diferenciando os dados específicos de cada tipo.
* **Pessoa Física:** Contém atributos exclusivos de clientes pessoas físicas, como *CPF*, *nome* e *data de nascimento*.
* **Pessoa Jurídica:** Contém informações específicas de clientes empresas, como *CNPJ*, *razão social* e *nome fantasia*.
* **Endereço:** Guarda os dados de endereço dos clientes, incluindo logradouro, número e tipo (residencial, comercial, etc.). Está diretamente relacionado à entidade *Cliente* e, por meio dela, se conecta à entidade *CEP*.
* **CEP:** Mantém os dados de bairro, cidade e estado, centralizando as informações geográficas e evitando redundâncias nos cadastros de endereço.
* **Telefone:** Permite o cadastro de múltiplos telefones para cada cliente, armazenando DDD e número.
* **Email:** Armazena os endereços de e-mail dos clientes, permitindo também múltiplos registros por cliente.

Os relacionamentos foram definidos da seguinte forma:

* Um *Produto* pode estar associado a um ou mais registros de *Estoque*, enquanto cada registro de estoque refere-se a um único produto.
* Cada *Pedido* pode incluir diversos *Produtos*, através da entidade associativa *Venda*, que também permite registrar a quantidade de cada item e seu preço de venda.
* Um *Cliente* pode realizar vários *Pedidos*, bem como possuir múltiplos *Telefones*, *Emails* e *Endereços*.
* Os *Endereços* estão vinculados diretamente ao *Cliente* e referenciam a entidade *CEP*, que centraliza as informações territoriais.
* A entidade *Cliente* possui uma generalização, sendo especializada em *Pessoa Física* e *Pessoa Jurídica*, de modo que cada cliente é obrigatoriamente um ou outro.

Na Figura 1, a seguir, apresenta-se o Diagrama Entidade-Relacionamento (DER) que ilustra de maneira gráfica toda a modelagem conceitual descrita.

**Figura 1 – Diagrama Entidade-Relacionamento do sistema de Farmácia**  
*Fonte: Elaboração própria.*

**3. MODELAGEM LÓGICA**

A modelagem lógica foi realizada com base no DER apresentado, visando a transformação das entidades em tabelas relacionais normalizadas. Foram definidos os nomes das tabelas, atributos, tipos de dados, chaves primárias, chaves estrangeiras e demais restrições de integridade.

Todas as tabelas foram projetadas de forma a manter a integridade referencial e a consistência dos dados. As entidades Pessoa Física e Pessoa Jurídica foram tratadas como especializações da entidade Cliente. As entidades Endereço, Telefone e Email foram relacionadas com Cliente em relações de um-para-muitos.

O modelo foi normalizado até a Terceira Forma Normal (3FN), garantindo a eliminação de dados redundantes e a independência dos atributos.

## **4. ESTRUTURA DO BANCO DE DADOS**

Esta seção apresenta os scripts desenvolvidos em SQL para implementação do banco de dados relacional da farmácia. Os scripts estão organizados em DDL (criação da estrutura), DML (manipulação dos dados) e DQL (consultas).

### **4.1 Estrutura do Banco de Dados**

A seguir, são apresentadas as tabelas que compõem o banco de dados do sistema de gerenciamento de farmácia, com a respectiva descrição dos atributos e suas funções:

Cliente: Armazena os dados básicos de cada cliente, sejam pessoas físicas ou jurídicas.

ID\_Cliente: Identificador único (chave primária).

Tipo\_Cliente: Indica se o cliente é do tipo “Física” ou “Jurídica”.

**Pessoa\_Fisica:** Armazena informações específicas de clientes pessoa física.

ID\_Cliente: Chave estrangeira que referencia a tabela Cliente.

CPF: Cadastro de Pessoa Física (único).

Nome: Nome completo.

Data\_Nascimento: Data de nascimento.

**Pessoa\_Juridica:** Contém dados específicos de clientes do tipo empresa.

ID\_Cliente: Chave estrangeira que referência a tabela Cliente.

CNPJ: Cadastro Nacional da Pessoa Jurídica (único).

Razao\_Social: Nome legal da empresa.

Nome\_Fantasia: Nome comercial da empresa.

**Email:** Permite o registro de múltiplos endereços de e-mail por cliente.

ID\_Email: Identificador único.

ID\_Cliente: Chave estrangeira para Cliente.

Email: Endereço eletrônico único.

**Telefone:** Permite o cadastro de múltiplos números de telefone por cliente.

ID\_Telefone: Identificador único.

ID\_Cliente: Chave estrangeira para Cliente.

DDD: Código de área.

Número: Número de telefone.

**CEP:** Tabela de apoio que armazena informações geográficas centralizadas.

CEP: Código de Endereçamento Postal (chave primária).

Bairro, Cidade, Estado: Localização geográfica.

**Endereço:** Relaciona um cliente a um endereço específico.

ID\_Endereco: Identificador único.

ID\_Cliente: Chave estrangeira.

CEP: Chave estrangeira para CEP.

Número, Logradouro, Tipo: Detalhes do endereço.

**Produtos:** Contém informações dos produtos comercializados.

ID\_Produto: Identificador único.

Produto: Nome comercial.

Preco: Valor unitário.

Categoria: Tipo do produto (ex: analgésico, antibiótico).

Codigo\_Barras: Código único.

Validade: Data de validade.

Tarja: Indicação de controle (ex: Isento, Vermelha).

**Estoque:** Controla a quantidade de produtos disponíveis.

ID\_Estoque: Identificador único.

ID\_Produto: Chave estrangeira para Produtos.

Quantidade: Quantidade disponível.

**Pedidos:** Registra as compras realizadas por clientes.

ID\_Pedido: Identificador único.

ID\_Cliente: Chave estrangeira.

Dia, Hora: Data e hora do pedido.

Metodo\_Pagamento: Forma de pagamento.

Valor\_Total: Valor total da compra.

**Vendas:** Representa os itens contidos em um pedido.

ID\_Pedido, ID\_Produto: Chaves compostas (PK e FK).

Quantidade: Número de unidades vendidas.

Preco\_Venda: Preço por unidade na venda.

### **4.2 Relacionamento entre Tabela**

O banco de dados está estruturado de forma relacional, com a utilização de chaves primárias e estrangeiras para garantir a integridade referencial. Os relacionamentos estão descritos a seguir:

A tabela Cliente se relaciona com:

Pessoa\_Fisica e Pessoa\_Juridica (relação 1:1 por especialização).

Email, Telefone e Endereço (relação 1:N – um cliente pode ter vários).

A tabela Endereço se relaciona com CEP em uma relação N:1 (muitos endereços podem compartilhar o mesmo CEP).

A tabela Pedidos se relaciona com Cliente (1:N), permitindo muitos pedidos por cliente.

A tabela Vendas representa uma relação N:N entre Pedidos e Produtos, onde:

Um pedido pode conter vários produtos.

Um produto pode aparecer em vários pedidos.

Essa relação é mediada pela tabela associativa Vendas.

A tabela Estoque se relaciona com Produtos (1:1 ou 1:N), dependendo da estrutura. Neste projeto, consideramos um estoque único por produto.

**4.3 Scripts DDL – Data Definition Language**

Os scripts a seguir definem a criação das tabelas, chaves primárias e estrangeiras, bem como as restrições necessárias para a integridade do banco de dados.

-- SCRIPT DDL – CRIAÇÃO DAS TABELAS

-- Projeto: Sistema de Vendas (E-commerce)

-- Banco: PostgreSQL

CREATE TABLE Cliente (

ID\_Cliente SERIAL PRIMARY KEY,

Tipo\_Cliente VARCHAR(10) NOT NULL -- Física ou Jurídica

);

CREATE TABLE Pessoa\_Fisica (

ID\_Cliente INT PRIMARY KEY,

CPF CHAR(11) UNIQUE NOT NULL,

Nome VARCHAR(100) NOT NULL,

Data\_Nascimento DATE NOT NULL,

FOREIGN KEY (ID\_Cliente) REFERENCES Cliente(ID\_Cliente) ON DELETE CASCADE

);

CREATE TABLE Pessoa\_Juridica (

ID\_Cliente INT PRIMARY KEY,

CNPJ CHAR(14) UNIQUE NOT NULL,

Razao\_Social VARCHAR(100) NOT NULL,

Nome\_Fantasia VARCHAR(100),

FOREIGN KEY (ID\_Cliente) REFERENCES Cliente(ID\_Cliente) ON DELETE CASCADE

);

CREATE TABLE Email (

ID\_Email SERIAL PRIMARY KEY,

ID\_Cliente INT NOT NULL,

Email VARCHAR(150) UNIQUE NOT NULL,

FOREIGN KEY (ID\_Cliente) REFERENCES Cliente(ID\_Cliente) ON DELETE CASCADE

);

CREATE TABLE Telefone (

ID\_Telefone SERIAL PRIMARY KEY,

ID\_Cliente INT NOT NULL,

DDD CHAR(2) NOT NULL,

Numero VARCHAR(9) NOT NULL,

FOREIGN KEY (ID\_Cliente) REFERENCES Cliente(ID\_Cliente) ON DELETE CASCADE

);

CREATE TABLE CEP (

CEP CHAR(8) PRIMARY KEY,

Bairro VARCHAR(100) NOT NULL,

Cidade VARCHAR(100) NOT NULL,

Estado CHAR(2) NOT NULL

);

CREATE TABLE Endereco (

ID\_Endereco SERIAL PRIMARY KEY,

ID\_Cliente INT NOT NULL,

CEP CHAR(8) NOT NULL,

Numero VARCHAR(10) NOT NULL,

Logradouro VARCHAR(150) NOT NULL,

Tipo VARCHAR(20),

FOREIGN KEY (ID\_Cliente) REFERENCES Cliente(ID\_Cliente) ON DELETE CASCADE,

FOREIGN KEY (CEP) REFERENCES CEP(CEP)

);

CREATE TABLE Produtos (

ID\_Produto SERIAL PRIMARY KEY,

Produto VARCHAR(100) NOT NULL,

Preco DECIMAL(10,2) NOT NULL,

Categoria VARCHAR(50),

Codigo\_Barras VARCHAR(20) UNIQUE,

Validade DATE,

Tarja VARCHAR(50)

);

CREATE TABLE Estoque (

ID\_Estoque SERIAL PRIMARY KEY,

ID\_Produto INT NOT NULL,

Quantidade INT NOT NULL,

FOREIGN KEY (ID\_Produto) REFERENCES Produtos(ID\_Produto)

);

CREATE TABLE Pedidos (

ID\_Pedido SERIAL PRIMARY KEY,

ID\_Cliente INT NOT NULL,

Dia DATE NOT NULL,

Hora TIME NOT NULL,

Metodo\_Pagamento VARCHAR(50),

Valor\_Total DECIMAL(10,2),

FOREIGN KEY (ID\_Cliente) REFERENCES Cliente(ID\_Cliente)

);

CREATE TABLE Vendas (

ID\_Pedido INT NOT NULL,

ID\_Produto INT NOT NULL,

Quantidade INT NOT NULL,

Preco\_Venda DECIMAL(10,2) NOT NULL,

PRIMARY KEY (ID\_Pedido, ID\_Produto),

FOREIGN KEY (ID\_Pedido) REFERENCES Pedidos(ID\_Pedido),

FOREIGN KEY (ID\_Produto) REFERENCES Produtos(ID\_Produto)

);

## **5. MANIPULAÇÃO DE DADOS**

**5.1 Scripts DML – Data Manipulation Language**

Os comandos abaixo foram utilizados para preencher o banco com dados de exemplo.

INSERT INTO Cliente (Tipo\_Cliente) VALUES ('Física'); -- ID 1

INSERT INTO Cliente (Tipo\_Cliente) VALUES ('Jurídica'); -- ID 2

INSERT INTO Cliente (Tipo\_Cliente) VALUES ('Física'); -- ID 3

INSERT INTO Cliente (Tipo\_Cliente) VALUES ('Jurídica'); -- ID 4

INSERT INTO Pessoa\_Fisica (ID\_Cliente, CPF, Nome, Data\_Nascimento)

VALUES

(1, '12345678901', 'Ana Souza', '1995-03-20'),

(3, '98765432100', 'Carlos Mendes', '1988-10-15');

INSERT INTO Pessoa\_Juridica (ID\_Cliente, CNPJ, Razao\_Social, Nome\_Fantasia)

VALUES

(2, '11222333000199', 'Drogaria Vida LTDA', 'Farmácia Vida'),

(4, '55667788000122', 'Clínica Bem Estar', 'Clínica Bem Estar');

INSERT INTO Email (ID\_Cliente, Email)

VALUES

(1, '[ana.souza@email.com](mailto:ana.souza@email.com)'),

(2, '[contato@farmaciavida.com.br](mailto:contato@farmaciavida.com.br)'),

(3, '[carlosmendes@email.com](mailto:carlosmendes@email.com)'),

(4, '[contato@clinicabemestar.com](mailto:contato@clinicabemestar.com)');

INSERT INTO Telefone (ID\_Cliente, DDD, Numero)

VALUES

(1, '11', '912345678'),

(2, '21', '30021234'),

(3, '19', '998877665'),

(4, '31', '32211234');

-- CEPs

INSERT INTO CEP (CEP, Bairro, Cidade, Estado)

VALUES

('01310930', 'Bela Vista', 'São Paulo', 'SP'),

('20040002', 'Centro', 'Rio de Janeiro', 'RJ'),

('13050002', 'Jardim Proença', 'Campinas', 'SP');

INSERT INTO Endereco (ID\_Cliente, CEP, Numero, Logradouro, Tipo)

VALUES

(1, '01310930', '50', 'Av. Paulista', 'Residencial'),

(2, '20040002', '100', 'Rua da Farmácia', 'Comercial'),

(3, '13050002', '456', 'Rua das Rosas', 'Residencial'),

(4, '20040002', '300', 'Av. Central', 'Comercial');

INSERT INTO Produtos (ID\_Produto, Produto, Preco, Categoria, Codigo\_Barras, Validade, Tarja)

VALUES

(1, 'Dipirona 500mg – 20 comprimidos', 9.90, 'Analgésico', '7896015512345', '2026-12-31', 'Isento'),

(2, 'Paracetamol 750mg – 20 comprimidos', 11.50, 'Antitérmico', '7891058009876', '2026-08-30', 'Isento'),

(3, 'Amoxicilina 500mg – 21 cápsulas', 28.90, 'Antibiótico', '7896422512367', '2025-12-01', 'Vermelha'),

(4, 'Protetor Solar FPS 50 – 200ml', 49.90, 'Dermocosmético', '7891910001133', '2026-10-15', 'Isento');

INSERT INTO Estoque (ID\_Produto, Quantidade)

VALUES

(1, 100),

(2, 120),

(3, 50),

(4, 30);

INSERT INTO Pedidos (ID\_Cliente, Dia, Hora, Metodo\_Pagamento, Valor\_Total)

VALUES (1, '2025-06-10', '14:30', 'Pix', 21.40);

INSERT INTO Vendas (ID\_Pedido, ID\_Produto, Quantidade, Preco\_Venda)

VALUES (1, 1, 1, 9.90),

(1, 2, 1, 11.50);

INSERT INTO Pedidos (ID\_Pedido, ID\_Cliente, Dia, Hora, Metodo\_Pagamento, Valor\_Total)

VALUES (2, 3, '2025-06-12', '10:15', 'Cartão de Crédito', 28.90);

INSERT INTO Vendas (ID\_Pedido, ID\_Produto, Quantidade, Preco\_Venda)

VALUES (2, 3, 1, 28.90);

INSERT INTO Pedidos (ID\_Pedido, ID\_Cliente, Dia, Hora, Metodo\_Pagamento, Valor\_Total)

VALUES (3, 4, '2025-06-13', '09:00', 'Boleto', 99.80);

INSERT INTO Vendas (ID\_Pedido, ID\_Produto, Quantidade, Preco\_Venda)

VALUES (3, 4, 2, 49.90);

**5.2 Scripts DQL – Data Query Language**

As consultas abaixo são utilizadas para recuperar e analisar dados do sistema.

-- CONSULTAS DQL – FARMÁCIA - PROJETO (PostgreSQL)

-- 1. Produtos mais vendidos

SELECT

p.Produto,

SUM(v.Quantidade) AS Total\_Vendido

FROM Vendas v

JOIN Produtos p ON v.ID\_Produto = p.ID\_Produto

GROUP BY p.Produto

ORDER BY Total\_Vendido DESC;

-- 2. Total gasto por cliente (PF e PJ)

SELECT

c.ID\_Cliente,

COALESCE(pf.Nome, pj.Razao\_Social) AS Nome\_Cliente,

SUM(pe.Valor\_Total) AS Total\_Gasto

FROM Cliente c

LEFT JOIN Pessoa\_Fisica pf ON c.ID\_Cliente = pf.ID\_Cliente

LEFT JOIN Pessoa\_Juridica pj ON c.ID\_Cliente = pj.ID\_Cliente

JOIN Pedidos pe ON pe.ID\_Cliente = c.ID\_Cliente

GROUP BY c.ID\_Cliente, pf.Nome, pj.Razao\_Social;

-- 3. Estoque

SELECT

p.Produto,

e.Quantidade

FROM Estoque e

JOIN Produtos p ON e.ID\_Produto = p.ID\_Produto

-- 4. Email dos clientes

SELECT

c.ID\_Cliente,

COALESCE(pf.Nome, pj.Razao\_Social) AS Nome\_Cliente,

e.Email

FROM Cliente c

LEFT JOIN Pessoa\_Fisica pf ON c.ID\_Cliente = pf.ID\_Cliente

LEFT JOIN Pessoa\_Juridica pj ON c.ID\_Cliente = pj.ID\_Cliente

JOIN Email e ON c.ID\_Cliente = e.ID\_Cliente

ORDER BY Nome\_Cliente;

-- 5. Consulta detalhada de um pedido

SELECT

pe.ID\_Pedido,

COALESCE(pf.Nome, pj.Razao\_Social) AS Cliente,

p.Produto,

v.Quantidade,

v.Preco\_Venda,

pe.Metodo\_Pagamento

FROM Pedidos pe

JOIN Cliente c ON pe.ID\_Cliente = c.ID\_Cliente

LEFT JOIN Pessoa\_Fisica pf ON c.ID\_Cliente = pf.ID\_Cliente

LEFT JOIN Pessoa\_Juridica pj ON c.ID\_Cliente = pj.ID\_Cliente

JOIN Vendas v ON v.ID\_Pedido = pe.ID\_Pedido

JOIN Produtos p ON p.ID\_Produto = v.ID\_Produto

WHERE pe.ID\_Pedido = 1;

-- 6. Produtos comprados por um cliente

SELECT

COALESCE(pf.Nome, pj.Razao\_Social) AS Cliente,

pr.Produto,

v.Quantidade,

v.Preco\_Venda,

pe.Dia

FROM Vendas v

JOIN Pedidos pe ON v.ID\_Pedido = pe.ID\_Pedido

JOIN Cliente c ON pe.ID\_Cliente = c.ID\_Cliente

LEFT JOIN Pessoa\_Fisica pf ON c.ID\_Cliente = pf.ID\_Cliente

LEFT JOIN Pessoa\_Juridica pj ON c.ID\_Cliente = pj.ID\_Cliente

JOIN Produtos pr ON v.ID\_Produto = pr.ID\_Produto

WHERE c.ID\_Cliente = 1;

-- 7. Vendas realizadas em uma data

SELECT

pe.ID\_Pedido,

COALESCE(pf.Nome, pj.Razao\_Social) AS Cliente,

p.Produto,

v.Quantidade,

v.Preco\_Venda

FROM Pedidos pe

JOIN Cliente c ON pe.ID\_Cliente = c.ID\_Cliente

LEFT JOIN Pessoa\_Fisica pf ON c.ID\_Cliente = pf.ID\_Cliente

LEFT JOIN Pessoa\_Juridica pj ON c.ID\_Cliente = pj.ID\_Cliente

JOIN Vendas v ON v.ID\_Pedido = pe.ID\_Pedido

JOIN Produtos p ON v.ID\_Produto = p.ID\_Produto

WHERE pe.Dia = '2025-06-10';

-- 8. Vendas agrupadas de acordo com a categoria de produto

SELECT

p.Categoria,

SUM(v.Quantidade) AS Total\_Unidades,

SUM(v.Quantidade \* v.Preco\_Venda) AS Receita\_Total

FROM Vendas v

JOIN Produtos p ON v.ID\_Produto = p.ID\_Produto

GROUP BY p.Categoria

ORDER BY Receita\_Total DESC;

-- 9. Total de pedidos por cliente

SELECT

COALESCE(pf.Nome, pj.Razao\_Social) AS Cliente,

COUNT(pe.ID\_Pedido) AS Total\_Pedidos

FROM Pedidos pe

JOIN Cliente c ON pe.ID\_Cliente = c.ID\_Cliente

LEFT JOIN Pessoa\_Fisica pf ON c.ID\_Cliente = pf.ID\_Cliente

LEFT JOIN Pessoa\_Juridica pj ON c.ID\_Cliente = pj.ID\_Cliente

GROUP BY pf.Nome, pj.Razao\_Social

ORDER BY Total\_Pedidos DESC;

## **6. CONTROLE DE ACESSO**

Neste projeto, optou-se por não implementar comandos de controle de acesso (DCL), como GRANT e REVOKE, por se tratar de um sistema acadêmico de uso individual e sem múltiplos perfis de acesso. Em um cenário real, a utilização desses comandos seria fundamental para garantir que apenas usuários autorizados pudessem acessar ou modificar determinados dados do banco de dados.  
  
Como o foco foi a estruturação e manipulação dos dados em um ambiente controlado, não houve necessidade de configurar permissões diferenciadas entre usuários.

## **7. CONTROLE DE TRANSAÇÕES** Comandos de controle de transações (DTL) não foram utilizados neste projeto, pois o sistema simulado não envolveu operações críticas simultâneas ou cenários complexos de rollback. Em um ambiente de produção, o uso de transações seria essencial para garantir a integridade em operações como atualizações financeiras ou registros encadeados. Neste caso, as manipulações foram feitas de forma isolada e sequencial, não exigindo controle transacional avançado.

## **8. CONCLUSÃO** O desenvolvimento deste projeto permitiu consolidar os conhecimentos adquiridos em Banco de Dados Relacional por meio da criação de um sistema de gerenciamento para uma farmácia. Todas as etapas fundamentais de um projeto de banco de dados foram contempladas: levantamento de requisitos, modelagem conceitual, modelagem lógica, normalização até a Terceira Forma Normal (3FN) e implementação dos scripts no PostgreSQL.

As entidades e seus relacionamentos foram definidos com clareza, respeitando as boas práticas de integridade referencial. Os dados foram organizados em estruturas bem projetadas, e os testes com inserções, atualizações e consultas demonstraram que o sistema se comporta de forma consistente e eficiente. O uso de comandos SQL (DDL, DML e DQL) validou a funcionalidade e a coerência do modelo lógico desenvolvido.

Apesar de os comandos de controle de acesso (DCL) e de controle de transações (DTL) não terem sido aplicados neste projeto, essa escolha se justifica por se tratar de um trabalho acadêmico com uso individual e controlado, onde não havia necessidade de criar níveis de permissão ou lidar com transações simultâneas. Mesmo assim, estudamos e entendemos a importância desses recursos, que são essenciais em sistemas maiores, com múltiplos usuários ou operações críticas. Por isso, reconhecemos que, em projetos futuros, o uso dessas técnicas será fundamental para garantir mais segurança e confiabilidade ao banco de dados

Dessa forma, o grupo considera que os objetivos propostos foram plenamente alcançados, demonstrando a capacidade de projetar, estruturar e validar um banco de dados relacional completo, funcional e bem documentado — pronto para servir de base para aplicações reais.

## **9. REFERÊNCIAS**

GITHUB - André Souza. Material da disciplina Banco de Dados. Disponível em: <https://github.com/profAndreSouza/Material/tree/main/Banco%20de%20Dados>. Acesso em: 19 jun. 2025.

REPOSITÓRIO OFICIAL DO GRUPO. PROJ-DADOS. Disponível em: <https://github.com/Joao-Pedro-Matias/PROJ-DADOS>. Acesso em: 19 jun. 2025.