

***Sistema de gerenciamento de supermercado***

Banco de dados



2024/2025 – LEIC Grupo 107:

1. Arthur Teixeira – up202300368@up.pt
2. Guilherme Cardozo - up202401169@up.pt
3. Pedro Gouveia - up202200045@up.pt

1. **Descrição do Contexto:**

O ponto central do projeto é o desenvolvimento de um sistema para gerir um estoque de um supermercado. Ele permitirá otimizar o controle e monitorar os diferentes produtos do estoque. O foco do sistema é gerenciar vários aspectos do fluxo de produtos, desde a compra dos fornecedores até a venda para os clientes, atualizando eficientemente a quantidade de produtos estocados.

2. **Domínio do Projeto:**

O domínio deste projeto é a gestão de estoque em um supermercado. A gestão mais eficiente do estoque é muito importante devido a quantidade de transações, mudanças corriqueiras no número de itens e a diversidade de produtos. O sistema deve gerir um banco de dados de produtos que se diferenciam em atributos: preço, data de validade e detalhes do fornecedor, por exemplo, rastreando as compras e vendas, com atualizações no estoque.

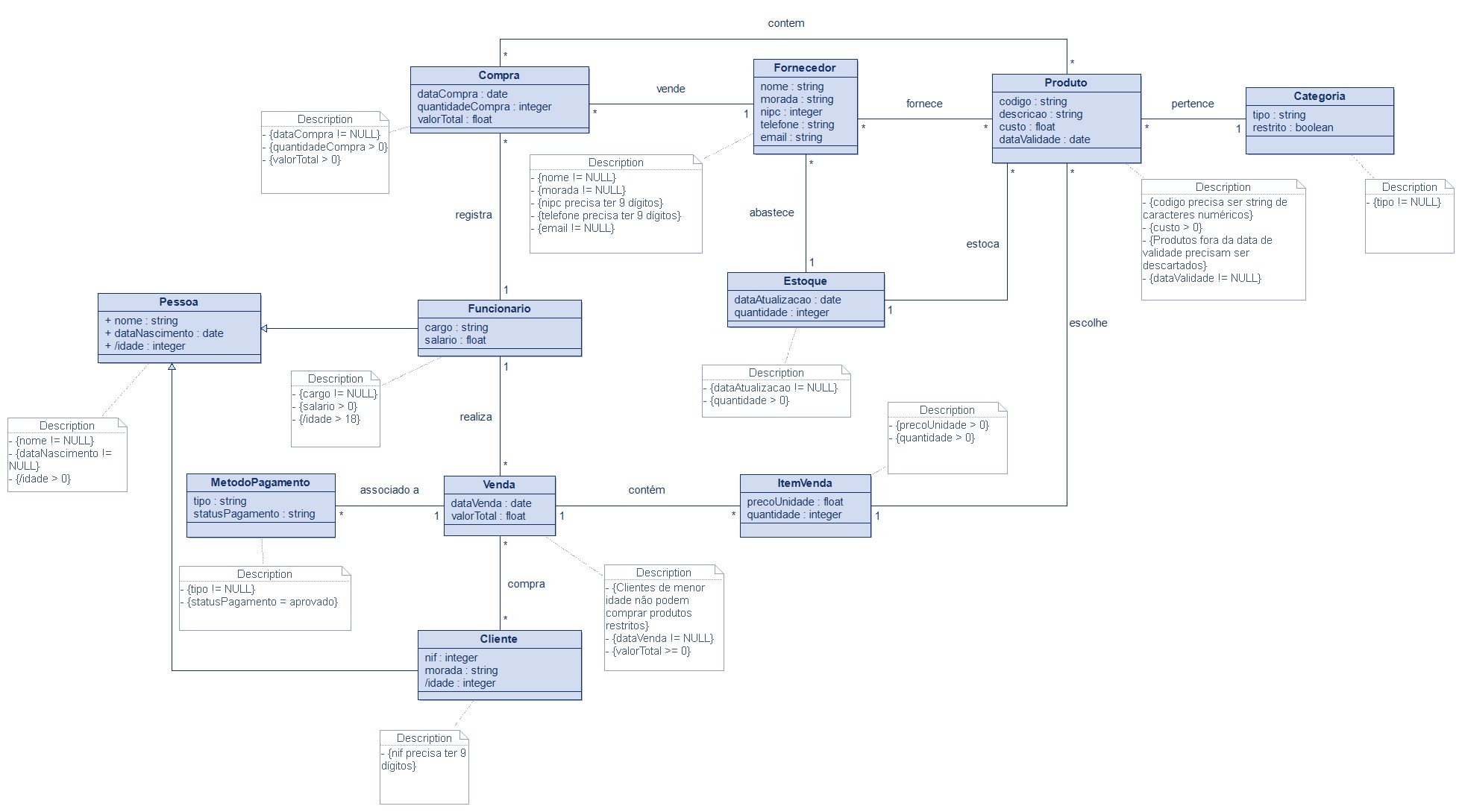
O modelo cria um sistema que pode registrar produtos, gerenciar fornecedores e acompanhar as compras e vendas. Portanto, será possível saber o que está disponível, o que falta e o que precisa ser reposto de forma clara e assertiva. Esse sistema ajuda na operação diária do supermercado e garante que os produtos estejam sempre disponíveis para os clientes.

1. **Requisitos de Dados:**

Para que o sistema funcione, ele precisará gerir informações essenciais. As principais são: “Produto”, “Fornecedor”, “Compra” e “Venda”, e todas elas estão interligadas para mostrar o ciclo de um produto, desde o momento em que ele chega ao estoque até a venda para o cliente. Os dados mais importantes que o sistema vai gerenciar:

* + **Informações dos Produtos:** Cada produto terá um código exclusivo e informações contendo nome, descrição, preço, custo e data de validade. Manter esses dados atualizados evita problemas como produtos vencidos ou garantir a rotação de estoque.
  + **Informações dos Fornecedores**: É preciso reabastecer constantemente o estoque.Por isso, a ferramenta inclui registros do nome, endereço, contato dos fornecedores.
  + **Dados de Compra:** Toda compra que o supermercado realizar a partir de um fornecedor é adicionada com a informação do número total de produtos e seus preços.
  + **Dados de Venda:** as vendas também são monitoradas para garantir reposição de estoque. Isso inclui a data, a quantidade vendida e o preço geral da venda.

1. **Modelo conceitual UML inicial:**



1. **Integração da inteligência artificial:**

Foi utilizado o ChatGPT como ferramenta IA para a integração. O Prompt utilizado foi: *“O que podemos fazer para melhorar o projeto de acordo com a proposta: o projeto terá como tema a gestão de estoque de um mercado. O banco de dados irá gerenciar as informações de produtos, como código, nome, preço e quantidade disponível em estoque. Além disso, será possível registrar entradas e saídas de produtos, atualizando o estoque em tempo real. As entidades principais incluem “Produto”, “Fornecedor”, “Venda” e “Compra”, e as relações entre elas ajudarão a rastrear o fluxo de mercadorias no mercado. O sistema será capaz de gerar relatórios sobre produtos em baixa, otimizando a reposição de estoque e o controle financeiro. Está permitido constraints, association classes, generalization, composition, aggregation, etc.*”. A resposta referente ao prompt foi:” *Vamos discutir quando e por que seria útil incluir:*

*1. Associação de Classes (Association Class)*

*Uma associação de classe é utilizada quando uma relação entre duas classes precisa de atributos próprios ou comportamentos específicos que não pertencem diretamente a uma das classes envolvidas, mas à própria relação entre elas.*

*Exemplos no seu projeto:*

*Venda - Produto (ItemVenda): A classe ItemVenda já está modelada como uma classe intermediária entre Venda e Produto, o que é um exemplo clássico de associação de classe. Ela representa a associação entre uma venda e os produtos vendidos, com seus atributos específicos (como preço unitário e quantidade). Esta modelagem está correta, mas vale reforçar que esta é uma associação de classe que gerencia os detalhes da relação.*

*Compra - Produto (ItemCompra): Seria útil introduzir uma classe de associação entre Compra e Produto, similar ao que foi feito entre Venda e Produto. Isso permitiria registrar atributos específicos para cada item na compra, como a quantidade de produtos comprados, o preço unitário no momento da compra e até mesmo a data prevista de entrega. Vamos chamá-la de ItemCompra. Assim, o sistema pode rastrear o preço pago por cada unidade de produto ao longo do tempo e a quantidade comprada.*

1. *Qualificadores*

*Os qualificadores são usados para refinar uma associação, permitindo que uma coleção de objetos seja indexada por um valor, tornando o acesso aos objetos mais eficiente ou lógico.*

*Onde usar qualificadores no seu sistema?*

*Fornecedor - Produto: Um bom caso para usar um qualificador seria a relação entre Fornecedor e Produto. Suponha que cada fornecedor tenha um código de identificação próprio para cada produto que ele fornece (ou outro valor único). Esse código poderia ser utilizado como um qualificador para a relação. Isso refina a associação, permitindo que o sistema acesse os produtos fornecidos por um fornecedor específico com base nesse código.*

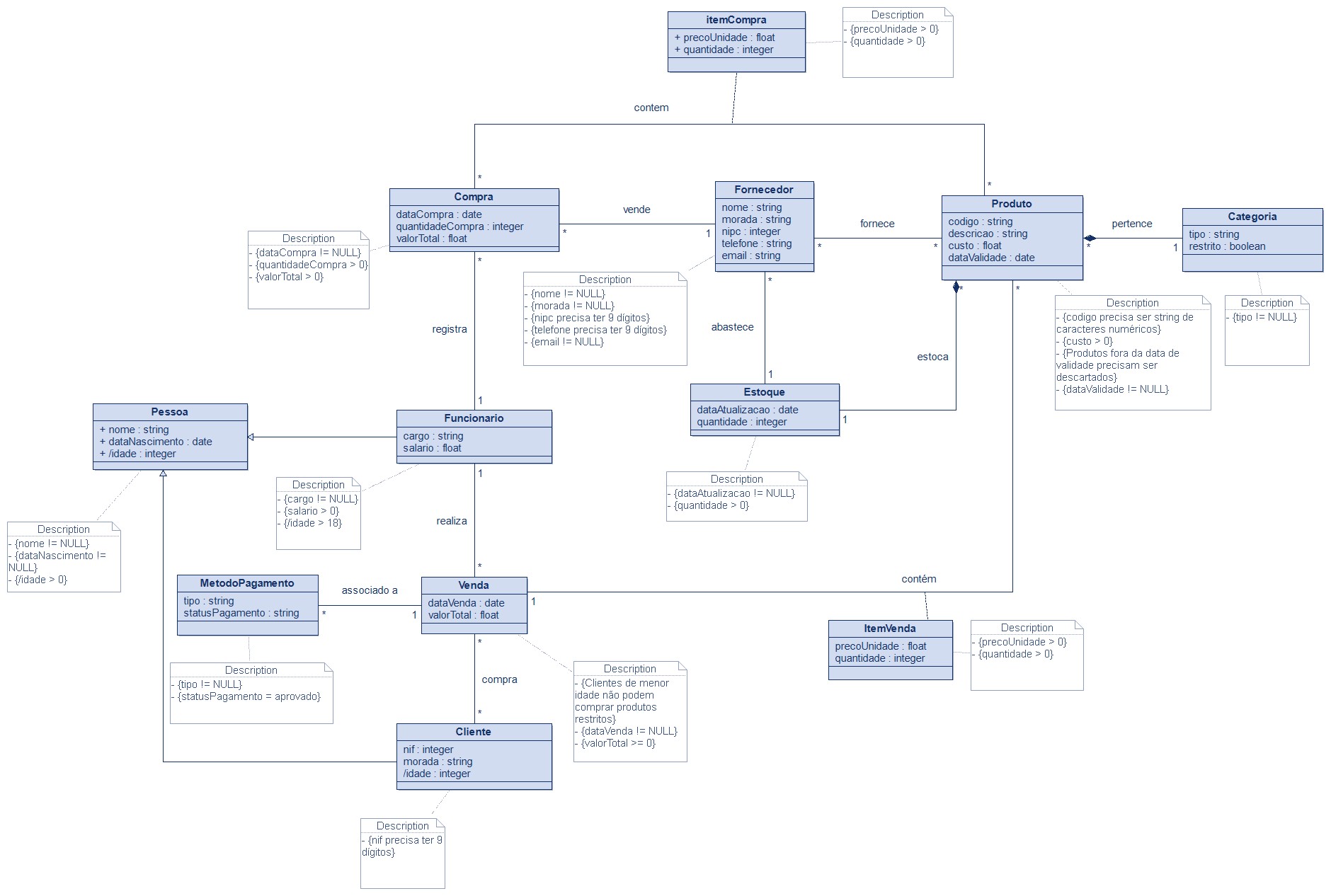
1. *Combinando Associações de Classe e Qualificadores*

*É possível combinar qualificadores com associações de classe para criar um modelo ainda mais detalhado. Por exemplo, no caso de Venda - ItemVenda - Produto, você poderia usar o qualificador dataVenda para refinar a consulta por data, mas ainda manter a associação de classe ItemVenda para gerenciar os detalhes da transação.".*

1. *Uso de Composição*

*Composição entre Produto e Categoria: Atualmente, um Produto pertence a uma Categoria. Seria mais lógico usar composição, onde um produto sempre pertence a uma única categoria, e se a categoria for deletada, o produto associado também deixaria de existir. Assim, o relacionamento ficaria como uma composição.*

6. **Modelo UML conceitual final:**



Após discussões e análises do modelo da inteligência artificial generativa, concluímos como pertinentes as seguintes alterações:

* Adicionar relações de composição entre Produto e Categoria, assim como, em Produto e Estoque. Desta forma garantimos uma relação mais assertiva, onde se uma categoria ou o estoque for deletado, o produto é excluído também.
* Adicionar as associações de classes “itemVenda”, entre Venda e Produto, e “itemCompra”, entre Compra e Produto. Porque as associações de classes criam uma melhor relação e uma clareza nas ações de venda e compra, tornando mais fácil o rastreamento.
* Optamos por não usar os qualificadores, pois acreditamos que o atributo NIPC e a lista dos produtos que o fornecedor dispõe são suficientes para suprir a necessidade do qualificador, citado pelo chat GPT, e consequentemente o tópico 3 não seria necessário.

7**. Modelo Conceitual Corrigido**

Depois da correção e feedback do modelo conceitual, realizamos algumas alterações de forma a refinar o mesmo.

Diagrama

Descrição gerada automaticamente

* A classe estoque não se mostrava mais necessária por conta de ter propósito único e atributo único da quantidade de cada produto (afinal, só havia apenas um estoque). Dessa forma reformulamos adicionando o atributo “Quantidade” à classe Produto e retiramos o atributo “DataValidade”.
* Corrigimos multiplicidades destacadas e revisamos as mesmas.
* Retirada de atributos redundantes e nomeação devida das relações de generalização.
* Criamos uma associação de classes com o intuito de implementar o atributo “Prazo de Entrega”.

8. **Esquema Relacional Inicial**

Na conjuntura atual do modelo conceitual, o esquema relacional discorrido inicialmente, sem a integração com Inteligência Articial, é:

* PessoaFuncionario(NIF, Nome, DataNascimento, Cargo, Salario)
* PessoaCliente(NIF, Nome, DataNascimento, Morada)
* Fornecedor(NIPC, Nome, Morada, Telefone, Email)
* Fornece(NIPC -> Fornecedor.NIPC, Codigo -> Produto.Codigo, PrazoEntrega)
* Produto(Codigo, Descricao, Custo, Categoria -> Categoria.Tipo, Fornecedor -> Fornecedor.NIPC, Quantidade)
* Compra(IDCompra, DataCompra, QuantidadeCompra, ValorTotal, Fornecedor -> Fornecedor.NIPC)
* ItemCompra(CodigoItem -> Produto.Codigo, IDCompra ->Compra.IDCompra, PrecoUnidade, Quantidade)
* Venda(IDVenda, IDCliente -> PessoaCliente.NIF, Vendedor ->PessoaFuncionario.NIF, DataVenda, ValorTotal, MetodoPagamento -> MetodoPagamento.Tipo)
* MetodoPagamento(Tipo, Status)
* ItemVenda(CodigoItem -> Produto.Codigo, IDVenda -> Venda.IDVenda, PrecoUnidade, Quantidade)

9.1. **Integração com Inteligência Artificial**

A principal sugestão de alteração no esquema relacional foi de adicionar um IDItemCompra e um IDItemVenda nas respectivas Classes ItemCompra e ItemVenda, mas após discussão e análise, desconsideramos a hipótese pois para nosso propósito não faz sentido a criação de um ID por item quando temos o próprio código que a classe Produto fornece, bem como o ID do momento da venda. Outras sugestões que a Inteligência Artificial ChatGPT apontou foram mais relacionadas ao escopo do modelo conceitual, sugerindo alterações na proposta que decidimos, o que também nos fez divergir das idéias apresentadas da mesma. Portanto, mantivemos nosso esquema relacional sem intervenções, resultando no esquema final o mesmo mostrado acima.

10. **Análise de Dependências Funcionais e suas Formas Normais**

Abaixo estão as dependências funcionais determinadas pelo grupo após análise inicial com a respectiva conclusão acerca da sua forma normal:

* PessoaFuncionario:
  + NIF ->Nome,DataNasc,Morada,Cargo
  + Cargo -> Salário
  + Justificativa de Normalização: Não está em Boyce Codd Normal Form, portanto não está em Third Normal Form pois a chave primária NIF é chave a partir de uma transitividade do atributo Cargo e, assim, viola os conceitos das formas supracitadas.
  + Conclusão: Para entrar nas Formas Normais requeridas, é necessário realizar uma decomposição em duas relações diferentes:
    - PessoaFuncionario(NIF,DataNasc,Morada,Cargo)
    - CargoSalario(Cargo,Salario)
* PessoaCliente:
  + NIF -> Nome, DataNasc, Morada
  + Justificativa de Normalização: Está na Terceira Forma Normal, e portanto, na Boyce Codd, pois na única Dependência Funcional, o NIF é Super Chave.
* Fornecedor:
  + NIPC -> Nome, Morada, Email, Telefone
  + Justificativa de Normalização: Está na Terceira Forma Normal, e portanto, na Boyce Codd pois na única Dependência Funcional, o NIPC é a única Super Chave.
* Fornece:
  + NIPC,Codigo ->PrazoEntrega
  + Justificativa de Normalização: Está na Terceira Forma Normal e, portanto, na Boyce Codd, pois a chave primária composta(NIPC,Codigo) é a única super chave.
* Categoria:
  + Tipo -> Restrito
  + Justificativa de Normalização: Está na Terceira Forma Normal, e portanto, na Boyce Codd pois na única Dependência Funcional, o Tipo é a única Super Chave.
* Produto:
  + Codigo -> Descrição, Custo, DataValidade, Categoria, Fornecedor, Quantidade
  + Justificativa de Normalização: Está na Terceira Forma Normal, e portanto, na Boyce Codd pois na única Dependência Funcional, o Codigo é a única Super Chave.
* Compra:
  + IDCompra -> DataCompra, QuantidadeCompra, ValorTotal, Fornecedor
  + Justificativa de Normalização: Está na Terceira Forma Normal, e portanto, na Boyce Codd pois na única Dependência Funcional, o NIPC é a única Super Chave.
* ItemCompra:
  + CodigoItem, IDCompra -> PrecoUnidade, Quantidade
  + Justificativa de Normalização: Está na Terceira Forma Normal pois a chave primária composta (CodigoItem,IDCompra) é a única super chave.
* Venda:
  + IDVenda ->IDCliente, IDVendedor, DataVenda, ValorTotal, MetodoPagamento
  + Justificativa de Normalização: Está na Terceira Forma Normal, e portanto, na Boyce Codd pois na única Dependência Funcional, o IDVenda é a única Super Chave.
* MetodoPagamento:
  + Tipo -> Status
  + Justificativa de Normalização: Está na Terceira Forma Normal, e portanto, na Boyce Codd pois na única Dependência Funcional, o Tipo é a única Super Chave.
* ItemVenda:
  + CodigoItem, IDVenda -> PrecoUnidade, Quantidade
  + Justificativa de Normalização: Está em Terceira Forma Normal e, portanto, na Forma Normal Boyce Codd, pois a chave primária composta (CodigoItem,IDVenda) é a única Super Chave.

10.1 **Integração com Inteligência Artificial**

Nossos comandos de prompt na Inteligência Artificial do chatGPT se mostrou ser mais com o intuito de correção do que acreditávamos estar certos. E após análise da mesma, a inteligência artificial apontou uma pequena correção na justificação de normalização pois a mesma se mostrou levemente incompleta. A frase abaixo mostra a correção da mesma, o que nos clarificou acerca do nosso equívoco anterior:

“O problema não está na chave primária NIF, mas sim no fato de que existe uma dependência transitiva envolvendo Cargo. A justificativa pode ser reescrita assim: A relação não está na Boyce-Codd Normal Form (BCNF) porque o atributo Cargo não é uma superchave, mas é determinante na dependência funcional Cargo -> Salário. Além disso, a relação não está na 3ª Forma Normal (3FN) devido à dependência transitiva NIF -> Cargo -> Salário.“

1. **Criação da Base de Dados em SQL**

Com o resultado da análise de dependências funcionais e conclusões acerca das justificativas das Formas Normais, implementamos no sqlite3 uma base de dados, se atentando acima de tudo às constraints e dando prioridade à manutenção da integridade dos dados por meio de update cascade e delete set null, considere o arquivo “create1.sql”.

11.1 **Integração com Inteligência Artificial**

Ao integrarmos com o ChatGPT, o mesmo sugeriu algumas mudanças, que foram ou não implementadas após discussões. Uma das mudanças que não aderimos foi a de ter um limite para a quantidade de caracteres em texto por meio de uma constraint. Acreditamos não ser o ideal pois não apenas temos tipos variados de informação de texto como também pensamos em aceitar nomes completos, o que traria uma indeterminação para o tamanho do nome em questão. Agora abaixo, encontra-se algumas mudanças sugeridas e implementadas pela equipe no arquivo “create2.sql”:

* + Uma correção de NIF que acabamos nos equivocando na correção e escrevemos errado de NIF TEXT para NIF INTEGER.
  + Alteramos alguns comandos de DELETE SET NULL para DELETE ON CASCADE em ITEMVENDA, ITEMCOMPRA e FORNECE devido a alta dependência das tabelas referenciadas. Se um produto, compra ou venda for deletado, não faz sentido manter itens ou associações órfãs.
  + Um problema que foi reparado e solucionado estava em MetodoPagamento. O Tipo ser chave primária única fazia com que não pudéssemos ter múltiplas entradas para o mesmo tipo de pagamento. Então para corrigir essa situação incluímos STATUS na chave primária, tornando assim, a tabela mais flexível e garantindo mais integridade relacional.

1. **População da Base de Dados**

O processo de população da base de dados foi realizado em duas etapas. Inicialmente, foram inseridos exemplos para cada tabela, utilizando ferramentas especializadas disponíveis online para gerar NIFs, nomes e códigos de forma aleatória. Este processo está documentado no arquivo *populate1.sql*. Posteriormente, empregou-se a ferramenta ChatGPT para ampliar a quantidade de registros inseridos na base de dados. Solicitamos à ferramenta a geração de até 10 novos elementos para as tabelas que apresentavam contexto apropriado. Como resultado, obteve-se uma base de dados consistente, uma vez que os atributos gerados foram devidamente adequados ao esquema da base, conforme ilustrado no arquivo “populate2.sql”.

1. **Integração com Inteligência Artificial**

A aplicação de ferramentas de inteligência artificial desempenhou um papel relevante em várias etapas do projeto, especialmente na geração e revisão de código. Contudo, no âmbito de modelagem relacional, o suporte da IA mostrou-se limitado, resultando em uma baixa adoção de suas sugestões nessa área. A contribuição mais significativa da ferramenta foi observada na etapa de população da base de dados, proporcionando registros bem estruturados e coerentes com o esquema do banco de dados. Por outro lado, sua utilidade foi menor na elaboração do diagrama UML, onde as orientações fornecidas não foram amplamente utilizadas.