**Uma imagem com Gráficos, Tipo de letra, captura de ecrã, design

Descrição gerada automaticamente**

**Franquia de Lojas de Penhores**

**Base de Dados**

**Licenciatura em Engenharia Informática e Computação**

**Docentes:**

Prof. Carla Lopes, [ctl@fe.up.pt](mailto:ctl@fe.up.pt)

Prof. Michel Ferreira, [mpferrei@fc.up.pt](mailto:mpferrei@fc.up.pt)

Prof. Mariana Dias, [mfdias@fe.up.pt](mailto:mfdias@fe.up.pt)

**Discentes:**

Carolina Mosqueiro, [up202303637@fe.up.pt](mailto:up202303637@fe.up.pt)

Mafalda Dias, [up202307365@fe.up.pt](mailto:up202307365@fe.up.pt)

Maria Catarina Moreira, [up202306603@fe.up.pt](mailto:up202306603@fe.up.pt)

Turma 5, Grupo 2

Porto, 20 de outubro de 2024

**Índice**

[Descrição do Tema 3](#_Toc180250315)

[Modelo Inicial 3](#_Toc180250316)

[Descrição 3](#_Toc180250317)

[Modelo Conceptual 5](#_Toc180250318)

[Integração de IA 6](#_Toc180250319)

[Modelo Final 8](#_Toc180250320)

[Descrição 8](#_Toc180250321)

[Modelo Conceptual 9](#_Toc180250322)

# **Descrição do Tema**

A Base de Dados expressa neste relatório pelos membros do grupo 502 pretende simular um sistema de franchising aplicado ao típico funcionamento de lojas de penhores. Como tal, todos os estabelecimentos associados a esta franquia estão estruturados de maneira similar e, no que diz respeito ao seu funcionamento, seguem um conjunto fixo de regras e padrões, nomeadamente em relação aos empréstimos que fornecem aos seus clientes, às taxas de juros associadas e aos tipos de artigos que podem penhorar.

No que diz respeito à definição básica de lojas de penhores, estas são estabelecimentos que não só vendem certos tipos de artigos – tais como eletrodomésticos, joias, antiguidades, entre outros –, mas também os adquirem de clientes que, em troca, recebem um empréstimo – geralmente em dinheiro e que não lhes foi possível obter através de meios tradicionais. Estes artigos penhorados não ficam imediatamente na posse da loja, pois é concedido um intervalo de tempo aos clientes para devolverem o valor emprestado – agora com juros adicionados – de forma a recuperarem os seus itens. Caso não consigam fazê-lo até ao fim do prazo outrora estabelecido, estes passam a pertencer à loja e podem ser vendidos normalmente a entidades interessadas. Os detalhes inerentes aos itens são estabelecidos pelos funcionários das lojas.

O sistema de franchising aplicado a este conceito tem, como vantagem, a partilha de dados entre as diversas lojas associadas, particularmente no que diz respeito aos seus clientes e artigos.

# **Modelo Inicial**

## **Descrição**

**Loja:** A classe "Loja" pretende representar os diversos estabelecimentos pertencentes à franquia. Todos possuem um ID, nome, endereço e contacto telefónico. Cada loja tem dois inventários, da mesma forma que cada um desses inventários só pertence a uma loja.

**Inventário:** A classe “Inventário” procura distinguir os itens penhorados daqueles que se encontram à venda. Como tal, define todos os artigos da loja, os quais, pelo contrário, recebem um estado através desta classe.

**Item:** A classe “Item” define quer os artigos penhorados quer aqueles a serem vendidos. Estes possuem um ID, um estado e a data em que se integraram na loja. Um item pode ou não estar à venda e pode ou não estar penhorado, que, por conseguinte, se transparece num empréstimo. Cada artigo só pertence a um cliente, mas um cliente pode penhorar mais do que um artigo. Cada item possui uma descrição.

**Descrição:** A classe “Descrição” está diretamente associada a um item, na medida em que inclui o seu nome, a categoria a que pertence, as suas dimensões, o material de que é feito e o seu valor estimado. Está incluído, ainda, um ID da descrição.

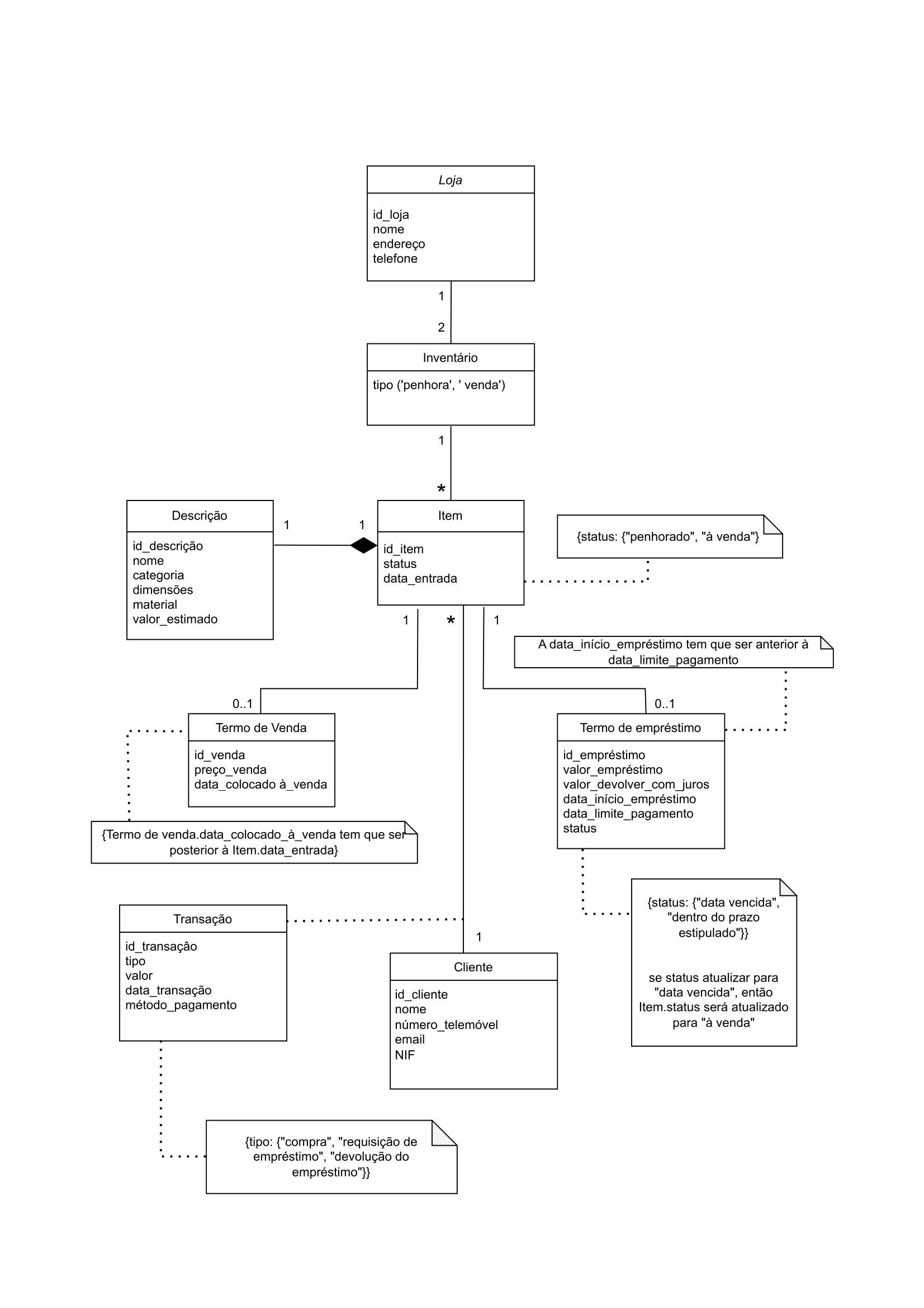
**Termo de Empréstimo:** A classe “Termo de Empréstimo” diz respeito ao empréstimo fornecido ao cliente de acordo com os itens penhorados. Inclui um ID, a quantia do empréstimo, os juros associados, a data em que foi realizado, a data limite para a retorna do empréstimo e respetivos juros e um estado, o qual define se o prazo para o seu pagamento foi ultrapassado ou não.

**Termo de Venda:** A classe “Termo de Venda” traduz a venda de um artigo a um cliente. Possui, como atributos, um ID, o preço do artigo e a data em que este foi vendido.

**Cliente:** A classe “Cliente” representa aquele que pretende penhorar ou comprar um item. Inclui um ID associado à pessoa, o seu nome, contacto telefónico, e-mail e NIF. Um cliente e o(s) seu(s) respetivo(s) artigo(s) desencadeiam uma transação.

**Transação:** A classe “Transação” simboliza, de um modo geral, que tipo de transferência entre a loja e o cliente está a ser realizada: comprar um artigo, requisitar um empréstimo ou devolver a quantia outrora requisitada. Inclui um ID associado à transação, o seu tipo, valor transacionado, data da ocorrência e o método de pagamento utilizado.

## **Modelo Conceptual**



# **Integração de IA**

Escolheu-se, entre os vários modelos LLM, usar Claude.ai (versão 3.5 Sonnet). Esta decisão baseou-se na capacidade do Claude 3.5 Sonnet de compreender e gerar respostas detalhadas sobre tópicos complexos, como sistemas de bases de dados e modelos UML. Não só isso, como é capaz de ultrapassar algumas limitações de outras IA’s, nomeadamente a de criar e desenhar esquemas importantes de maneira avançada e minunciosa, bem como a de anexar ficheiros pdf. Utilizamos estas funcionalidades do Claude para gerar modelos conceituais de uma maneira mais visual, detalhada, iterativa e de fácil leitura, como também para conseguirmos fornecer os modelos que criamos inicialmente, já que se tornaria limitante ter de os descrever por texto, em vez de apenas anexar os ficheiros exportados para pdf com os esquemas que criamos no draw.io.

Além disso, a capacidade do Claude de formular perguntas pertinentes sobre tópicos relevantes, a fim de contextualizar o problema, e de utilizar nossas respostas para fornecer respostas precisas, sem fazer suposições, foi, por último, mas não menos importante, um dos fatores decisivos para a nossa escolha.

Antes de abordar a IA, foram definidas algumas falhas do modelo inicial que se pretendia corrigir:

* Desenvolver mais o modelo de modo a que a sua complexidade corresponda integralmente à do tópico escolhido;
* Simplificar a distinção entre itens penhorados e itens à venda.

Primeira Interação: Análise Geral do Modelo

“O esquema UML anexado retrata uma possível base de dados para um franchising de lojas de penhores. Os clientes podem dirigir-se à loja para penhorar ou comprar itens. Os itens à venda são itens penhorados cujos empréstimos não foram saldados no prazo estipulado. O esquema tem obrigatoriamente de ter entre 10 a 15 relações. O que mudarias neste esquema?”

A IA esquematizou um modelo relacional melhorando, que propunha várias alterações:

Nova entidade: Avaliação

Criação de uma nova entidade denominada "Funcionário”.

Atributos da entidade sugeridos: atributos como id\_avaliacão, valor\_estimado, data\_avaliacao”, entre outros

* Nova entidade: Funcionários

Criação de uma nova entidade denominada "Funcionário”.

Atributos da entidade sugeridos: id\_funcionario, nome, cargo, entre outros

Objetivo: permite criar uma ligação entre cada item e a sua avaliação formal, realizada por um funcionário;

Novos atributos na entidade: Termo de Empréstimo

separação de "valor\_devolver\_com\_juros" em "valor\_empréstimo" e "valor\_juros"

* Remoção da entidade: Descrição

Remoção da entidade “Descrição” e integração dos atributos da mesma (nome, categoria, dimensões e material) diretamente na entidade Item.

Análise e Decisão:

Sugestões Aceites:

1. Aceitou-se a sugestão de criar a entidade “Avaliação” porque acreditamos ser pertinente segregar os atributos que realmente pertencem objetivamente ao Item na entidade “Descrição” (como por exemplo o nome e dimensões do item), daqueles que necessitam de uma avaliação mais subjetiva, cuidada e formal (como por exemplo o estado de conservação do item e o valor estimado para o mesmo). Por isso, ajustou-se a restrição do Termo de Empréstimo, para que a data do início do empréstimo fosse obrigatoriamente posterior à data de avaliação do mesmo, para que nenhuma penhora seja feita sem o Item ter passado primeiro por uma avaliação, após dar entrada na loja de penhores (atributo guardado em Item.data\_entrada).
2. Aceitou-se a sugestão de criar a entidade “Funcionário” porque se relaciona de forma pertinente com a entidade “Avaliação”, agora implementada.
3. Aceitou-se a sugestão de expandir os atributos na entidade “Termo de Empréstimo”, porque num contexto de operações financeiras e de transações monetárias, é importante guardar diretamente o valor total que o cliente terá de devolver após pedir o empréstimo (o valor que lhe foi emprestado a somar com o valor de juros após ser aplicada determinada taxa), e não somente o valor de juros a serem aplicados em determinada penhora, para uma maior facilidade a utilizar, atualizar e manusear os dados.

Sugestões Rejeitadas:

1. Recusou-se a sugestão de remover a classe “Descrição”, porque continuamos a crer que não adicionava complexidade desnecessária ao modelo, mas antes uma normalização adequada ao tratamento de dados, pois nos traria uma organização e clareza no modelo.

Segunda Iteração: Simplificação das Relações

“Quanto às relações entre ‘Loja’, ‘Inventário’ e ‘Item’: o nosso objetivo era criar uma divisão entre os itens para venda e os itens penhorados, no entanto, a classe ‘Item’ já tem um atributo ‘status’ com um efeito parecido. Como simplificarias/modificavas estas relações?”

A IA identificou os seguintes pontos:

* **Redundância Estrutural**

A entidade 'Inventário' estava a ser utilizada como uma camada intermediária entre 'Loja' e 'Item';

O propósito principal era categorizar itens como "para venda" ou "penhorados";

Esta função já estava sendo cumprida pelo atributo 'status' na entidade 'Item'.

* **Proposta de Simplificação**

Sugeriu eliminar a camada intermediária desnecessária

Recomendou manter apenas o atributo 'status' como identificador do estado do item

Propôs uma ligação direta entre 'Loja' e 'Item' para maior eficiência

* **Justificação da IA**

A duplicação de funcionalidade aumentava a complexidade sem benefícios

A manutenção de duas formas de classificação poderia levar a inconsistências

Uma estrutura mais simples facilitaria a gestão dos dados

Decisão

Tal como proposto, foi decidido remover a entidade “Inventário”, ligando diretamente “Loja” e “Item”, já que a sua função principal – distinguir entre itens para venda e itens penhorados – já está a ser cumprida pelo atributo “status” na classe “Item”.

Assim, conseguimos eliminar a redundância e obteve-se um esquema mais claro, objetivo e simplificado. No entanto, ainda se mantinha um certo ceticismo em relação à pertinência ou não de manter o atributo “status” do item ou torná-lo numa classe separada, o que deu origem a uma nova interação com a IA.

Terceira Interação: Refinamento do Status

“Devido à sua importância, fará sentido incluir o ‘status’ do item como uma classe separada do esquema UML para facilitar a sua interpretação?”

A IA apresentou uma análise comparativa de prós e contras:

**Manter status como atributo:**

* Vantagens:
  + Esquema mais simples
* Desvantagens:
  + Menor visibilidade do status
  + Menor flexibilidade
  + Alterações estruturais necessárias para mudanças

**Criar classe Estado separada:**

* Vantagens:
  + Facilita adição de novos status
  + Destaca importância do estado no modelo
  + Permite rastreamento temporal de mudanças
* Desvantagens:
  + Adiciona complexidade ao modelo

Análise e Decisão Final

Decidiu-se remover a classe “Inventário”, mas também descartar a primeira proposta que mantinha “status” em “Item”.

Como o estado (“à venda”/“penhorado”) é dinâmico, ou seja, muda após um dado número de dias – após “data vencida” –, considerou-se importante retratar esta característica numa classe separada, denominada “Estado”, ainda que se tivesse que adicionar complexidade ao modelo.

Assim, criou-se uma nova entidade “Estado”, associada a “Item”, que visa mostrar, de forma clara, se o item foi penhorado e está armazenado até que seja pago o empréstimo a ele associado ou se foi retido pela loja e está disponível para venda.

Com esta nova entidade, a adição e transição para novos estados tornou-se mais flexível e de acesso mais direto, resultando num modelo mais robusto e adaptado às necessidades do negócio.

Quarta Interação: Restrições

“Quanto às restrições, sugeres alguma mudança?”

A IA fez as seguintes sugestões:

1. Em vez de remover o item quando vendido, sugeriu adicionar um estado "vendido", pois permitiria manter histórico de vendas e fazer análises de desempenho.
2. Para o Termo de Empréstimo, quando o status muda para "data vencida", sugeriu ter um período de graça antes de automaticamente mudar o estado para "à venda".
3. Sugeriu alterar o atributo “status” de Termo De Empréstimo para uma restrição, visto não fazer sentido poder ter um Termo De Empréstimo simultaneamente com um Termo de Venda, se um item também não pode estar simultaneamente à venda e penhorado.

Decisão

Sugestoes aceites:

Decidiu-se adicionar o tipo de estado “vendido” , em vez de retirar o item da base de dados quando fosse vendido, pois, de facto, é importante manter o histórico na base de dados dos itens que a loja já disponibilizou. Além disso, por motivos de coerência, decidiu-se adicionar também a restrição que o Termo de Venda seria finalizado quando a transação “compra” ocorresse e o tipo de estado fosse atualizado para “vendido”.

Aceitou-se a última sugestão, que propõe a inclusão de uma restrição que impeça que um mesmo item tenha simultaneamente um Termo de Venda e um Termo de Empréstimo. Em vez de implementar um atributo ‘status’, a decisão foi fundamentada na necessidade de evitar uma associação entre “Termo de Empréstimo” e um “Item” que não utiliza seus atributos, uma vez que está à venda e não mais penhorado. Essa incoerência foi resolvida ao garantir que os atributos mantenham uma multiplicidade de 0..1, o que permite que não haja um Termo de Empréstimo para um item que esteja à venda ou vendido.

Sugestões recusadas:

Não se aproveitou a sugestão de registar o motivo da mudança de estado, porque adicionaria complexidade desnecessária, visto que não se encontrou nenhuma vantagem significativa em não mudar o estado para “à venda” automaticamente.

# **Modelo Final**

## **Descrição**

As mudanças ao modelo podem ser resumidas de forma breve:

## **Modelo Conceptual**

