

**Faculdade de Engenharia da Universidade do Porto**



**Universidade do Porto**

**FEUP** Faculdade de Engenharia

# **Livraria**

## **Base de Dados**

### **Licenciatura Engenharia Informática e Computação**

Carla Alexandra Teixeira Lopes  
Michel Celestino Paiva Ferreira  
Mariana Ferreira Dias

#### **Estudantes e Autores**

##### **2LEIC08 - Grupo 806**

Ana Carolina Brandão, [up202004461@up.pt](mailto:up202004461@up.pt)  
Pedro Marques, [up202107525@up.pt](mailto:up202107525@up.pt)  
Marlon Belo, [up202209531@up.pt](mailto:up202209531@up.pt)

**Outubro 2023**  
**Porto, Portugal**

# Índice

<b>Índice</b>	<b>2</b>
<b>Introdução</b>	<b>3</b>
<b>Contexto</b>	<b>3</b>
<b>Diagrama UML</b>	<b>4</b>
<b>Integração de inteligência artificial generativa</b>	<b>5</b>
<b>Diagrama UML antes do chat gpt:</b>	<b>9</b>
<b>Considerações finais</b>	<b>10</b>
<b>Diagrama UML revisto</b>	<b>11</b>
<b>Alterações efetuadas - UML</b>	<b>12</b>
<b>Esquema relacional</b>	<b>13</b>
<b>Esquema Relacional - Integração AI</b>	<b>14</b>
<b>Dependências Funcionais</b>	<b>17</b>
<b>Análise de FNBC E 3ª Forma Normal</b>	<b>19</b>
<b>Dependências funcionais e Análise de FNBC - Integração AI</b>	<b>20</b>
<b>SQLite Database Creation - Integração AI</b>	<b>21</b>
<b>Data Loading - Integração AI</b>	<b>22</b>
<b>Conclusão</b>	<b>23</b>

# Introdução

O objetivo deste projeto é obter experiência prática no design e implementação de uma base de dados relacional em SQLite. Mais concretamente, pretende-se organizar e modelar as interações relativas a uma loja de livros, uma livraria.

## Contexto

A **Livraria** é identificada pelo seu nome e morada, possui também disponível um contacto telefónico. Funciona consoante dois **horários de atendimento** diferentes, um para os dias úteis da semana e outro para fins de semana e feriados.

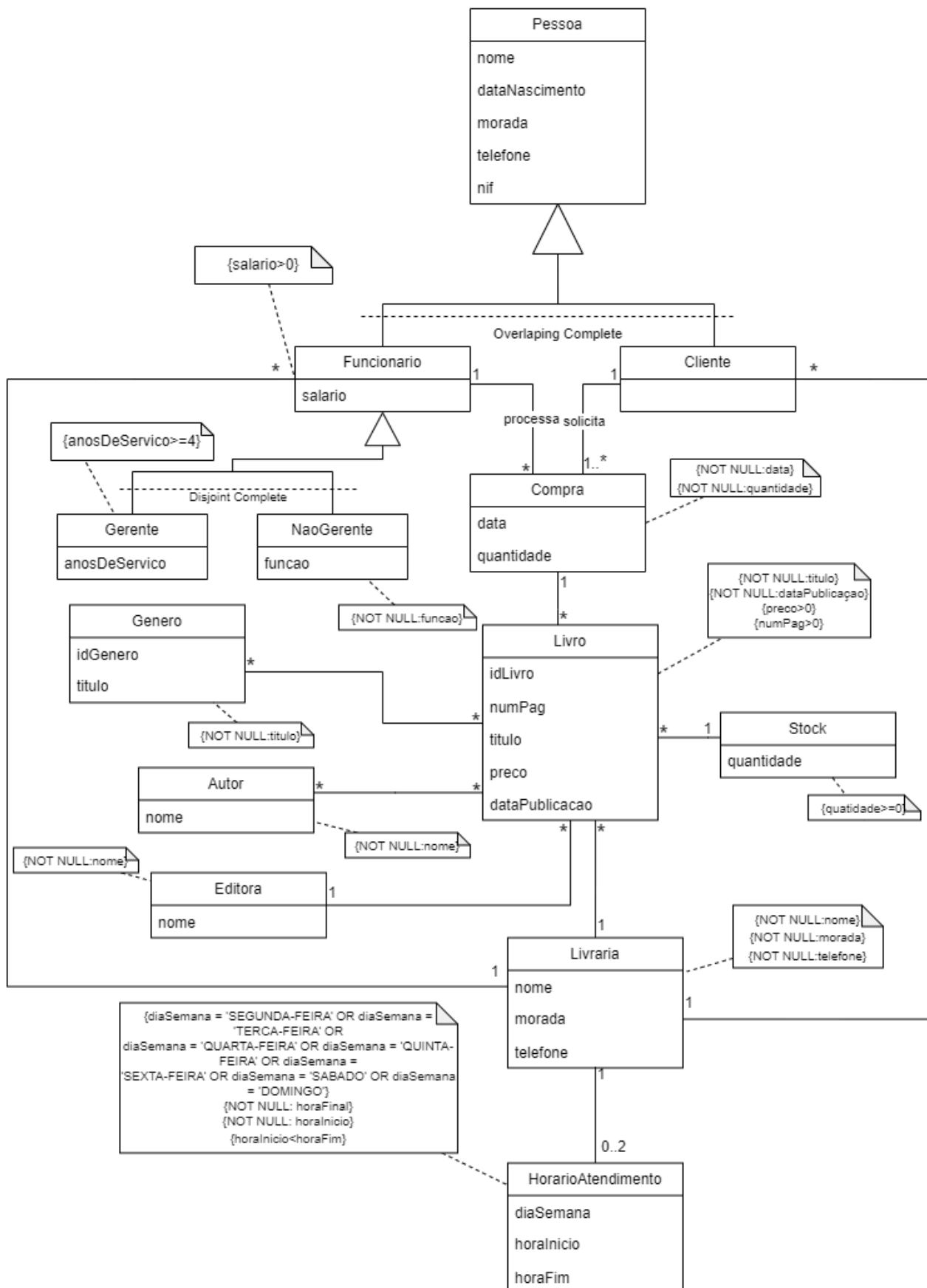
Tanto os **clientes** como os **funcionários** são identificados no sistema informático da livraria pelo seu nome, data de nascimento, morada, telefone e nif. Sendo os funcionários diferenciados dos clientes devido a possuírem um salário.

Os funcionários podem ou não ocupar o cargo de **gerência**, sendo que para tal aconteça, devem ser cumpridas certas condições. Apenas é possível a ocupação do cargo de gerente por um funcionário de cada vez. É necessário que o funcionário tenha no mínimo 4 anos de serviço.

Cada **compra** é processada por um funcionário e tem associada a quantidade de livros comprados e a data.

Um cliente pode comprar vários **livros**, desde que estejam presentes em **stock**. Livros estes de diversos géneros, autores e editoras. Um livro é identificado no sistema informático pelo seu id (idLivro) e possui as seguintes informações associadas: Título, número de páginas, preço e data de publicação.

# Diagrama UML



# Integração de inteligência artificial generativa

Para esta fase do Projeto foi utilizada primeiramente a ferramenta chat gpt-3.5 pertencente à empresa OpenAi [ChatGPT \(openai.com\)](https://openai.com).

Segue abaixo a nossa conversa com a Inteligência artificial, a **rosa** as prompts utilizadas e a **verde** as respostas obtidas juntamente com as nossas apreciações da mesma.

“Olá. Pretende-se organizar e modelar as interações relativas a uma loja de livros, uma livraria.

Poderias por favor fazer um esquema em UML para este efeito?

Aqui vai a descrição: \*Contexto”

Dado este input o ChatBot gerou as classes que considerou necessárias para a modelagem do nosso enunciado das quais:

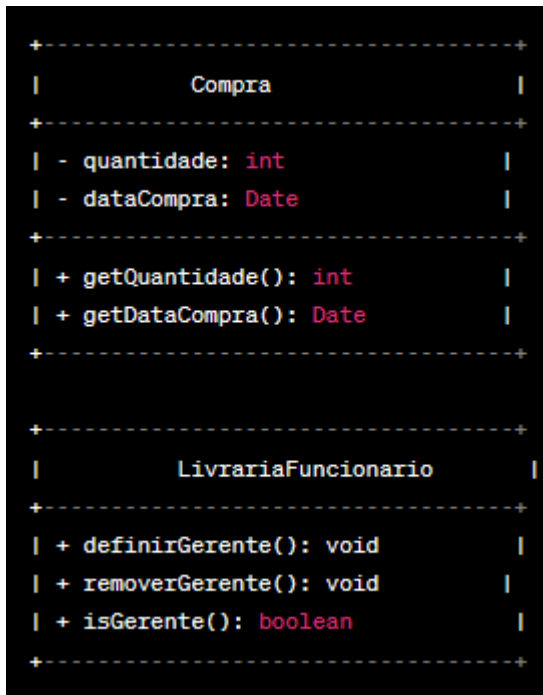
```
+-----+
|          Livraria          |
+-----+
| - nome: String             |
| - morada: String           |
| - contactoTelefonico: String |
| - horarioSemana: String     |
| - horarioFimDeSemana: String |
+-----+
| + getNome(): String        |
| + getMorada(): String      |
| + getContactoTelefonico(): String |
| + getHorarioSemana(): String |
| + getHorarioFimDeSemana(): String |
+-----+

+-----+
|          Pessoa           |
+-----+
| - nome: String             |
| - dataNascimento: Date     |
| - morada: String           |
| - telefone: String         |
| - nif: String              |
+-----+
| + getNome(): String        |
| + getDataNascimento(): Date |
| + getMorada(): String      |
| + getTelefone(): String    |
| + getNif(): String         |
+-----+
```

```
+-----+
|          Funcionario      |
+-----+
| - salario: double          |
| - cargo: String            |
| - anosServico: int         |
+-----+
| + getSalario(): double     |
| + getCargo(): String       |
| + getAnosServico(): int    |
+-----+

+-----+
|          Cliente          |
+-----+
| + comprarLivro(livro: Livro): void |
+-----+

+-----+
|          Livro            |
+-----+
| - idLivro: int             |
| - titulo: String           |
| - numPaginas: int          |
| - preco: double            |
| - dataPublicacao: Date     |
+-----+
| + getIdLivro(): int        |
| + getTitulo(): String      |
| + getNumPaginas(): int     |
| + getPreco(): double       |
| + getDataPublicacao(): Date |
+-----+
```



“Neste diagrama:

Livraria representa a livraria com seus atributos básicos, como nome, morada e horários de atendimento.

Pessoa é uma classe base que contém informações compartilhadas por funcionários e clientes.

Funcionario representa os funcionários da livraria com atributos adicionais, como salário, cargo e anos de serviço.

Cliente é uma classe que herda de Pessoa e possui um método para comprar livros.

Livro representa os livros disponíveis na livraria.

Compra contém informações sobre cada compra, como quantidade e data.

LivrariaFuncionario é uma classe auxiliar que permite definir um funcionário como gerente da livraria e verificar se um funcionário é gerente.

Lembre-se de que você pode adicionar mais detalhes e métodos conforme necessário para atender aos requisitos específicos da sua livraria.”

>>Podemos observar que de facto criou aproximadamente as mesmas classes e atributos que nós para a modelação do contexto fornecido. Este facto revela uma boa compreensão do problema e noção de contexto. É de salientar que não foi gerado um diagrama UML como pedido nem foi modelado como as classes se relacionam nem quais as suas multiplicidades (restrições em falta também). É provável que tal aconteça devido ao ChatBot estar restringido a fornecer respostas numa caixa de texto de tamanho e recursos limitados (sem suporte para esquemas). Vamos então tentar outra abordagem.<<

“Podes por favor descrever como se relacionam estas classes no contexto de um diagrama UML e quais as suas multiplicidades e restrições?”

“Vou descrever como as classes se relacionam no contexto de um diagrama UML, incluindo suas multiplicidades e restrições.

Relacionamento entre Livraria e Funcionario:

**Associação:** A classe Livraria tem uma associação com a classe Funcionario. Os funcionários trabalham na livraria.

**Multiplicidade:**

Livraria tem uma multiplicidade de 1 para N (ou seja, uma livraria pode ter vários funcionários).

Funcionario tem uma multiplicidade de 0 para N (um funcionário pode não estar associado a nenhuma livraria ou a várias).

**Restrições:** Nenhuma restrição específica.

**Relacionamento entre Funcionario e Cliente:**

**Associação:** A classe Funcionario e a classe Cliente são ambas subtipos da classe base Pessoa. Isso implica que funcionários e clientes compartilham informações pessoais em comum.

**Multiplicidade:** Não se aplica, pois é uma associação de herança.

**Relacionamento entre Funcionario e LivrariaFuncionario:**

**Associação:** A classe Funcionario tem uma associação com a classe LivrariaFuncionario para determinar se um funcionário é um gerente ou não.

**Multiplicidade:** A multiplicidade é de 0..1 de LivrariaFuncionario, o que significa que um funcionário pode ou não estar associado a um objeto LivrariaFuncionario (gerente).

**Restrições:**

Para que um funcionário seja considerado um gerente, ele deve estar associado a um objeto LivrariaFuncionario. Isso é definido pelo método definirGerente().

**Relacionamento entre Cliente e Compra:**

**Associação:** A classe Cliente tem uma associação com a classe Compra. Isso indica que um cliente pode realizar várias compras.

**Multiplicidade:**

Cliente tem uma multiplicidade de 0 para N (um cliente pode não fazer compras ou fazer várias).

Compra tem uma multiplicidade de 0 para N (uma compra pode ser feita por zero ou mais clientes).

**Relacionamento entre Compra e Livro:**

**Associação:** A classe Compra tem uma associação com a classe Livro. Isso indica que cada compra está relacionada a um ou mais livros.

**Multiplicidade:**

Compra tem uma multiplicidade de 1 para N (cada compra envolve pelo menos um livro).

Livro tem uma multiplicidade de 0 para N (um livro pode estar presente em zero ou mais compras).

**Relacionamento entre Livro e Livraria:**

**Associação:** A classe Livro tem uma associação com a classe Livraria para representar que a livraria possui uma coleção de livros.

**Multiplicidade:**

Livro tem uma multiplicidade de 1 para N (um livro pertence a uma ou mais livrarias).

Livraria tem uma multiplicidade de 0 para N (uma livraria pode ter zero ou mais livros).

Estas descrições representam como as classes estão relacionadas no contexto do diagrama UML e ajudam a entender as conexões entre as diferentes partes do sistema da

livraria. Lembre-se de que as restrições específicas podem variar dependendo dos detalhes da implementação do sistema.”

>>Podemos mais uma vez ver uma semelhança entre o output do ChatBot com o UML realizado pelo nosso grupo. Em geral, possui boas noções no que toca às relações entre classes e às suas respectivas multiplicidades, contudo, não concordamos com várias das multiplicidades que utilizou. Tal deve-se ao facto desta ser uma questão influenciada pela interpretação concreta do contexto e objetivo da base de dados. Ao analisar a resposta mais profundamente apercebemo-nos de que tínhamos uma inconsistência no que tocava às multiplicidades e que faltava a ligação de algumas classes, o que nos permitiu a sua correção.

Tome-se como exemplo os seguintes casos com os quais o nosso grupo não concordou: <<

“Funcionario tem uma multiplicidade de 0 para N (um funcionário pode não estar associado a nenhuma livraria ou a várias).”

>>Consideramos que um funcionário tem de estar associado a uma e uma só livraria, já que é aquilo que a nossa base de dados modela.<<

“A multiplicidade é de 0..1 de LivrariaFuncionario, o que significa que um funcionário pode ou não estar associado a um objeto LivrariaFuncionario (gerente).”

>>Na prompt fornecida em relação à existência de um gerente “Os funcionários podem ou não ocupar o cargo de gerência. sendo que para tal aconteça, devem ser cumpridas certas condições. Apenas é possível a ocupação do cargo de gerente por um funcionário de cada vez. É necessário que o funcionário tenha no mínimo 4 anos de serviço.”, o grupo achava que estava expresso que era necessária a existência de pelo menos um gerente contudo numa segunda leitura verificou-se que tal não era verdade e o erro foi corrigido<<

“Para que um funcionário seja considerado um gerente, ele deve estar associado a um objeto LivrariaFuncionario. Isso é definido pelo método definirGerente().”

>>Apesar de ser referido um método que supostamente define um gerente, não é em qualquer lado falada a condição da necessidade de pelo menos 4 anos de serviço por parte de um funcionário para que possa ocupar este cargo”<<

“Cliente tem uma multiplicidade de 0 para N (um cliente pode não fazer compras ou fazer várias).”

>> No nosso entendimento um cliente tem de fazer pelo menos uma compra para ser cliente.<<

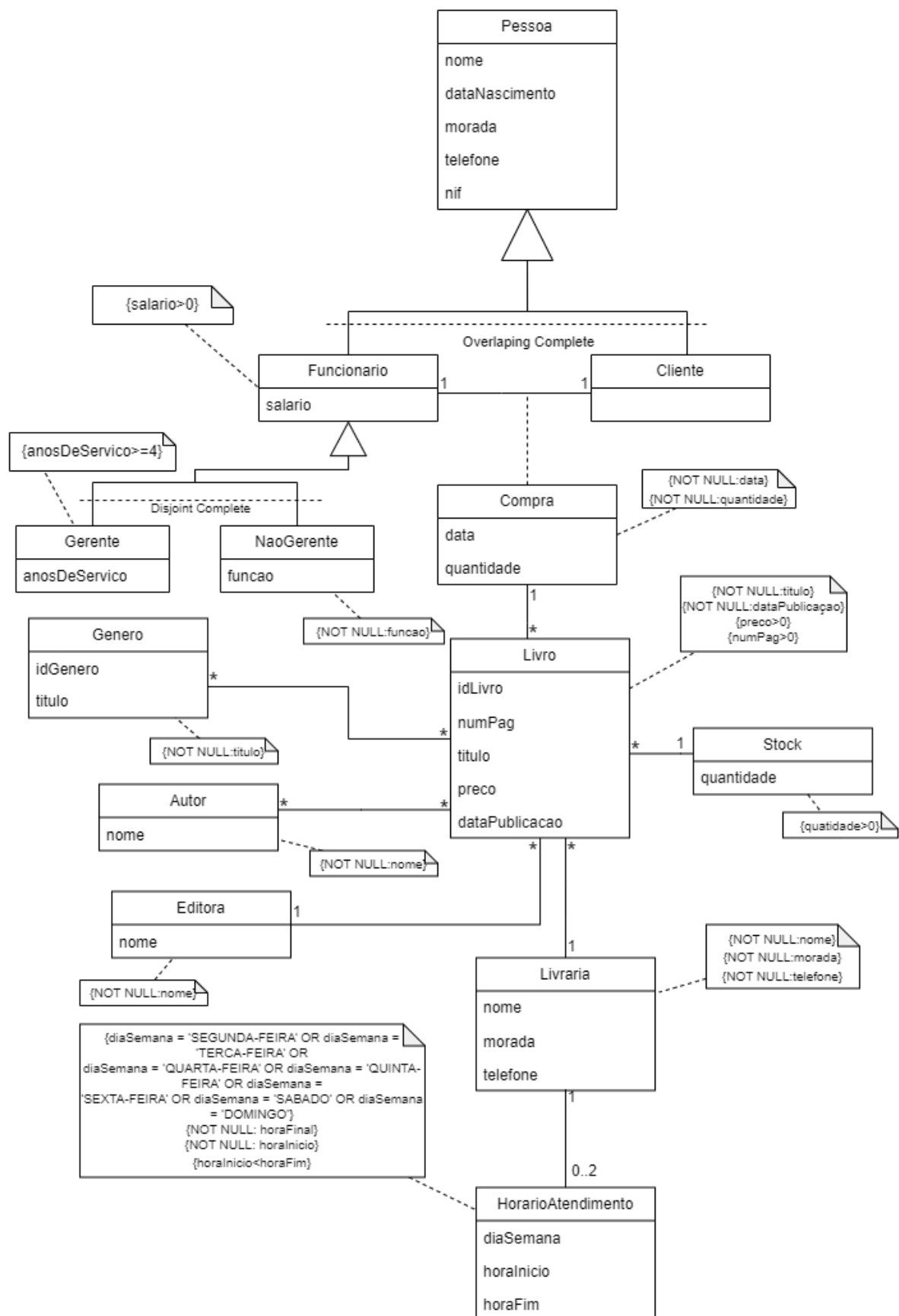
“Compra tem uma multiplicidade de 0 para N (uma compra pode ser feita por zero ou mais clientes).”

>> Uma compra tem de ser feita por 1 ou mais clientes <<

...



## Diagrama UML antes do chat gpt:

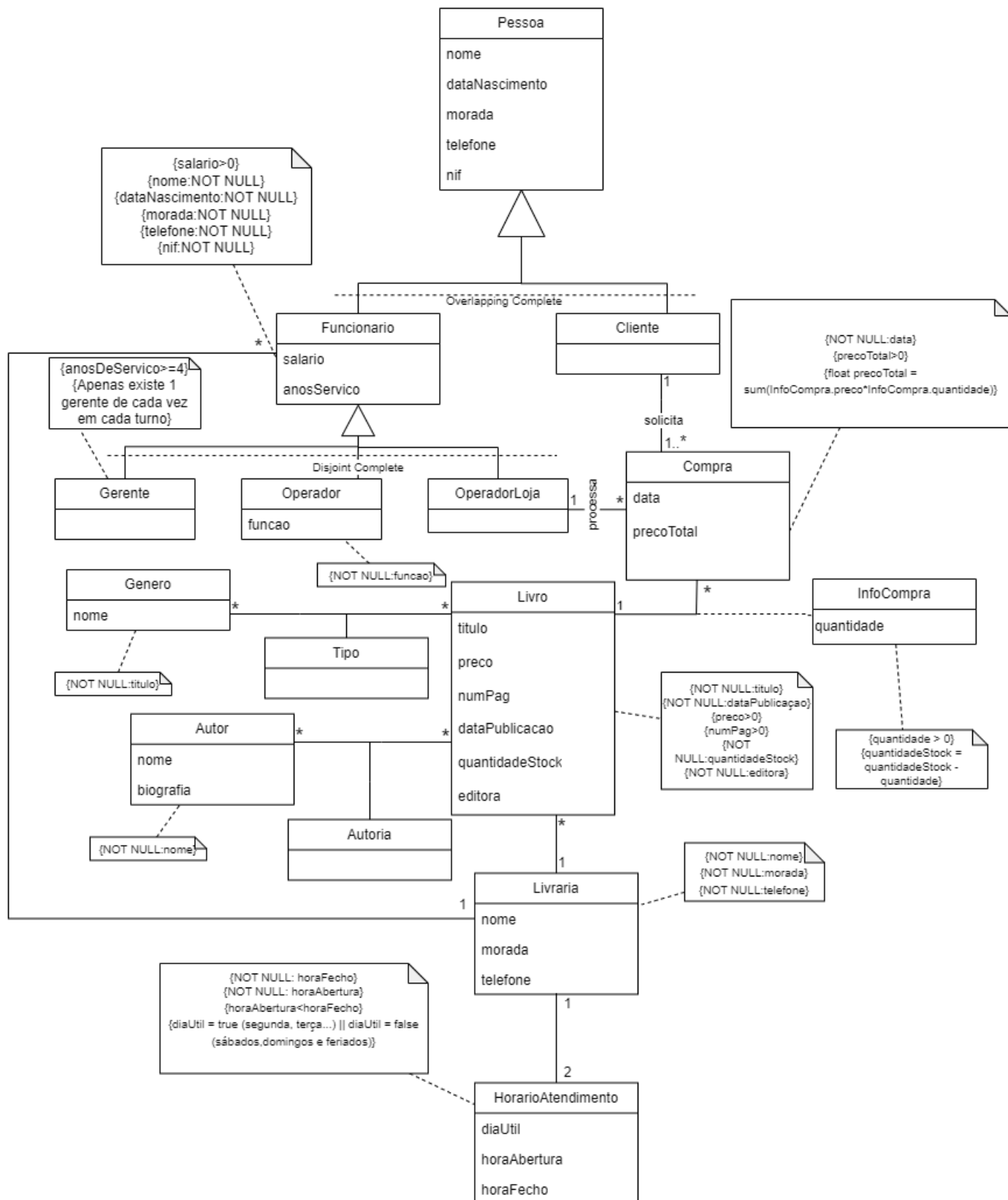


# Considerações finais

Na realização do nosso trabalho, se tivéssemos utilizado primeiramente o chatgpt, seria uma ajuda preciosa para o início do projeto como ponto de partida. Foi também muito útil numa fase mais avançada do trabalho como uma “segunda opinião” que nos alertou para erros e faltas de atenção. Especulamos também que se balizássemos melhor o problema, conseguiríamos alcançar resultados mais úteis, no entanto, até que ponto é que proceder desta forma seria assim tão útil? Uma vez que se for para utilizar o resultado que se espera como prompt, mais vale fazê-lo nós próprios.

Acreditamos que o uso das ferramentas de inteligência artificial atualmente ao nosso dispor é uma mais valia mas que ainda têm um longo caminho para andar. Já que possui problemas relativamente à especificidade e taxa de acerto no que toca a pedidos mais complexos feitos pelos utilizadores.

# Diagrama UML revisto



## Alterações efetuadas - UML

- Eliminação da classe **Editora**
- Criação de duas classes (**Tipo** e **Autoria**)
- Criação de uma Classe (**InfoCompra**)
- Alteração de restrições
- Mudança do nome de atributos
- Mudança do nome da Classe **NaoGerente** para **Operador**
- Criação de uma Classe **OperadorLoja**

# Esquema relacional

Cliente(idCliente, nome, dataNascimento, morada, telefone, nif)

Gerente(idGerente, nome, dataNascimento, morada, telefone, nif, salario, anosServico, idLivraria → Livraria)

Operador(idOperador, nome, dataNascimento, morada, telefone, nif, salario, anosServico, funcao, idLivraria → Livraria)

OperadorLoja(idOperadorLoja, nome, dataNascimento, morada, telefone, nif, salario, anosServico, idLivraria → Livraria)

Compra(idCompra, data, precoTotal, idOperadorLoja → OperadorLoja, idCliente → Cliente, idLivro → Livro)

Livro(idLivro, titulo, preco, numPag, dataPub, quantidadeStock, editora, idLivraria → Livraria)

InfoCompra(idLivro → Livro, idCompra → Compra, quantidade)

Genero(idGenero, nome)

Autor(idAutor, nome, biografia)

Tipo(idGenero → Genero, idLivro → Livro)

Autoria(idAutor → Autor, idLivro → Livro)

Livraria(idLivraria, nome, morada, telefone)

HorarioAtendimento(idHorarioAtendimento, diaUtil, horaAbertura, horaFecho, idLivraria → Livraria)

# Esquema Relacional - Integração AI

Vamos agora fazer uso das tecnologias atualmente à nossa disposição a fim de tentar melhorar o nosso produto final, neste caso, aplicando o poder da inteligência artificial para nos ajudar na nossa tentativa de melhorar o esquema relacional que estruturamos sem o seu auxílio.

A ferramenta utilizada desta vez foi o Bing-ChatGpt (Copilot), que faz uso do Gpt4. Escolhemo-lo dadas as suas capacidades de acesso ao contexto visual das páginas web em que nos encontramos, facilitando muito a nossa interação aquando da contextualização da inteligência artificial. Uma vez que não existe a necessidade de descrever de qualquer forma o nosso diagrama UML ou esquema relacional para que este pudesse tecer considerações sobre os mesmos.

Vamos usar também o mesmo esquema de cores para as nossas **prompts** e para as **respostas** da AI.

“Olá! Consegues ver o conteúdo da minha página atual?”

“Sim, eu consigo ver o conteúdo da sua página atual. Parece que você está usando o draw.io para criar um diagrama de entidade-relacionamento (ER) para um sistema de gestão de uma livraria. O diagrama mostra as entidades, os atributos, as relações e as restrições envolvidas no sistema. Você tem duas páginas no seu documento, cada uma com um diagrama diferente. A primeira página mostra um diagrama mais simples, com as entidades Livro, Autor, Editora, Funcionário, Cliente e Compra. A segunda página mostra um diagrama mais complexo, com as entidades Livro, Autor, Funcionário, Cliente, Compra, InfoCompra, OperadorLoja, Gerente, Operador e HorárioAtendimento. Você também tem algumas formas de texto, listas, etiquetas e setas na sua página ”

>> Neste momento a página web que se encontrava aberta era o draw.io. A AI, descreve com uma boa noção o conteúdo da página o que nos indica que consegue realmente ver o que nós estamos a ver (que é o diagrama UML antigo e atual do nosso projeto) , podendo usar o próprio conteúdo da página como base para as próximas prompts.

“Podes por favor sugerir um esquema relacional completo para o diagrama mais complexo presente na página atual?”

“Uma possível sugestão de esquema relacional completo para o diagrama mais complexo presente na página atual é:

- **Pessoa** (<u>nif</u>, nome, dataNascimento, morada, telefone)
- **Funcionario** (<u>nif</u>, salario, anosServico) <br> **Referência:** Funcionario.nif referencia Pessoa.nif
- **Gerente** (<u>nif</u>) <br> **Referência:** Gerente.nif referencia Funcionario.nif
- **Operador** (<u>nif</u>, funcao) <br> **Referência:** Operador.nif referencia Funcionario.nif
- **Cliente** (<u>nif</u>) <br> **Referência:** Cliente.nif referencia Pessoa.nif
- **Livro** (<u>titulo</u>, preco, numPag, dataPublicacao, quantidadeStock, editora)

- **Autor** (<u>nome</u>, biografia)
- **Escreve** (<u>nomeAutor, tituloLivro</u>) <br> **Referência:** Escreve.nomeAutor referencia Autor.nome, Escreve.tituloLivro referencia Livro.titulo
- **Compra** (<u>data, nifCliente, nifOperador</u>, precoTotal) <br> **Referência:** Compra.nifCliente referencia Cliente.nif, Compra.nifOperador referencia Operador.nif
- **InfoCompra** (<u>dataCompra, tituloLivro</u>, quantidade) <br> **Referência:** InfoCompra.dataCompra referencia Compra.data, InfoCompra.tituloLivro referencia Livro.titulo
- **HorarioAtendimento** (<u>diaUtil, nifGerente</u>, horaAbertura, horaFecho) <br> **Referência:** HorarioAtendimento.nifGerente referencia Gerente.nif”

“O que significa <u>, </u> e <br> na tua resposta?”

“O símbolo <u> indica que o atributo é uma chave primária, ou seja, um identificador único para cada tupla da entidade ou relação. O símbolo <br> indica uma quebra de linha, ou seja, uma mudança para uma nova linha no texto. ”

>> O chatbot parece ter usado o nif, que é um identificador único de uma entidade seja esta Pessoa, Compra, livro etc... da mesma maneira que nós utilizámos os identificadores (id, ex: idPessoa, idCompra, idLivro...). O que acontece é que para a classe **Livro**, possivelmente o título não será suficiente para termos segurança no facto de existir apenas um livro com esse título, não servindo como chave primária. Um caso semelhante acontece com a classe autor, se fosse para escolher um dos atributos como chave primária pensamos que seria mais seguro escolher biografia dado o seu carácter único, contudo o nosso grupo decidiu mesmo assim criar um id para esta classe a fim de simplificar.

>> Aparenta também ter utilizado a estratégia “E/R” para estruturar as relações entre generalizações, esta abordagem pode ser vista por exemplo relativamente às classes **Funcionario** e **Cliente** que em relação à sua superclasse Pessoa, possuem como chave primária uma chave estrangeira (Funcionario.nif e Cliente.nif respetivamente) que apontam para a chave primária da superclasse Pessoa (Pessoa.nif).

>> Foi criada uma classe **Escreve** que o nosso grupo não considera necessária para a modelação da nossa base de dados uma vez que colocámos no nosso diagrama UML uma relação de “muitos para muitos” entre Autor e livro já que um autor pode escrever vários livros e um livro pode ser escrito por vários autores. Assim, possuindo o idAutor e o idLivro não necessitamos de criar mais uma relação para relacionar a classe **Autor** e **Livro**

>> Por alguma razão que desconhecemos, utilizou como chave primária da classe **HorarioAtendimento** o atributo diaUtil e ainda mais questionavelmente, o atributo nifGerente que referencia Gerente.nif

>> Após a análise desta interação o grupo não fez qualquer alteração ao esquema relacional

>> Vamos então tentar uma abordagem diferente.

“Dado o diagrama UML mais complexo dos dois diagramas visíveis nesta página atua (\*contexto da página atual) , e o esquema relacional seguinte: (\*esquema relacional do grupo), farias alguma alteração ao esquema relacional? Se sim, quais e porquê?”

Uma possível alteração ao esquema relacional seria eliminar a entidade **OperadorLoja**, que parece ser redundante, e colocar o atributo **idOperadorLoja** na entidade **Compra**, referenciando diretamente o **idFuncionario** da entidade **Funcionario**. Assim, evita-se a criação de uma entidade extra que não acrescenta informação relevante ao modelo. Outra possível alteração seria eliminar o atributo **precoTotal** da entidade **Compra**, que pode ser calculado a partir dos atributos **quantidade** e **preco** da entidade **InfoCompra**. Assim, evita-se a redundância de dados e possíveis inconsistências.

>> O grupo não considerou a entidade **OperadorLoja** redundante uma vez que esta, sendo fundamental para modelar a interação entre a ação de compra (**Compra**) e a ação de processar a compra (**OperadorLoja**), numa situação prática teria atributos associados à classe **OperadorLoja** e a mais nenhuma classe. Não podendo por isso, o Operador de loja ser reduzido a um **idFuncionario** na classe **Compra**.

>> Não é fornecida uma razão válida em relação à necessidade da eliminação do atributo **precoTotal** da entidade compra. O atributo é **precoTotal** sendo derivado pode e deve na mesma ser modelado na entidade compra. Mais uma vez, nenhuma alteração foi efetuada ao modelo relacional.

>> Apesar da ferramenta de inteligência artificial não nos ter indicado diretamente nenhuma alteração significativa, esta alertou-nos para outros problemas mais sutis que possuímos no nosso diagrama UML e consequentemente no nosso esquema relacional. Estas mudanças foram tão mínimas que não merecem ser mencionadas.



# Dependências Funcionais

Cliente(idCliente, nome, dataNascimento, morada, telefone, nif)

- idCliente → idCliente, nome, dataNascimento, morada, telefone, nif
- nif → nif, idCliente, nome, dataNascimento, morada, telefone
- morada → morada, idCliente, nome, dataNascimento, telefone, nif
- telefone → telefone, idCliente, nome, dataNascimento, morada, nif

Gerente(idGerente, nome, dataNascimento, morada, telefone, nif, salario, anosServico, idLivraria → Livraria)

- idGerente → idGerente, nome, dataNascimento, morada, telefone, nif, salario, anosServico, idLivraria

Operador(idOperador, nome, dataNascimento, morada, telefone, nif, salario, anosServico, funcao, idLivraria → Livraria)

- idOperador → idOperador, nome, dataNascimento, morada, telefone, nif, salario, funcao, anosServico, idLivraria

OperadorLoja(idOperadorLoja, nome, dataNascimento, morada, telefone, nif, salario, anosServico, idLivraria → Livraria)

- idOperadorLoja → idOperadorLoja, nome, dataNascimento, morada, telefone, nif, salario, anosServico, idLivraria

Compra(idCompra, data, precoTotal, idOperadorLoja → OperadorLoja, idCliente → Cliente, idLivre → Livre)

- idCompra → idCompra, data, precoTotal, idOperadorLoja, idCliente, idLivre

Livre(idLivre, titulo, preco, numPag, dataPub, quantidadeStock, editora, idLivraria → Livraria)

- idLivre → idLivre, titulo, preco, numPag, dataPub, quantidadeStock, idLivraria

InfoCompra(idLivre → Livre, idCompra → Compra, quantidade)

- idLivre, idCompra → idLivre, idCompra, quantidade

Genero(idGenero, nome)

- idGenero → idGenero, nome
- nome → nome, idGenero

Autor(idAutor, nome, biografia)

- idAutor → idAutor, nome, biografia
- biografia → biografia, idAutor, nome

Autoria(idLivre → Livre, idAutor → Autor)

Tipo(idGenero → Genero, idLivre → Livre)

Livraria(idLivraria, nome, morada, telefone)

- idLivraria → idLivraria, nome, morada, telefone

- morada → morada, idLivraria , nome, telefone
- telefone → telefone, idLivraria, nome, morada

HorarioAtendimento(idHorarioAtendimento, diaUtil, horaAbertura, horaFecho, idLivraria\_→  
Livraria)

- idHorarioAtendimento →> diaUtil, horaAbertura, horaFecho, idLivraria
- horaFecho → horaFecho, diaUtil, horaAbertura, horaFecho, idLivraria

As relações **Autoria** e **GeneroLivro** são Classes associação many to many que ligam **Autor** a **Livro** e **Genero** a **Livro** respectivamente e por isso não existem dependências funcionais para além das triviais.

# Análise de FNBC E 3ª Forma Normal

Uma relação  $R$  encontra-se na Forma Normal de Boyce Codd se e só se: sempre que existe uma dependência funcional  $A_1 A_2 \dots A_n \rightarrow B_1 B_2 \dots B_m$  em  $R$ , dá-se o caso em que  $\{A_1, A_2, \dots, A_n\}$  é uma superchave de  $R$ . Isto é, o lado esquerdo de toda e qualquer dependência funcional não trivial deve ser uma superchave.

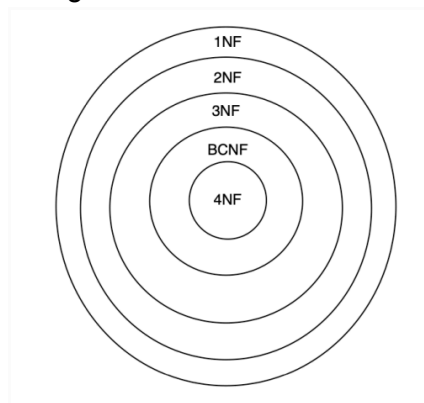
Sabemos que uma chave é um atributo (ou mais se estivermos a falar de uma chave composta) que é condição suficiente para identificar-se a si própria (dependência funcional trivial) e a todos os outros atributos dessa relação de forma inequívoca.

Sabemos também que o conceito de chave se encontra englobado pelo conceito de superchave, sendo uma chave uma superchave mínima .

Pudemos perceber na secção de dependências funcionais que todos os atributos que se encontravam do lado esquerdo de toda e qualquer dependência funcional, constituíam uma chave, sendo o seu fecho todos os atributos correspondentes à relação onde estão alocados.

→ Assim, conseguimos concluir que o nosso esquema relacional se encontra na Forma Normal de Boyce Codd.

Uma vez que tudo o que está na Forma Normal de Boyce Codd, se encontra também na 3ª Forma Normal (facto este que se deve à FNBC ser um subconjunto da 3ª Forma normal, como se pode ver pela imagem seguinte:



→ Podemos também concluir que o nosso esquema relacional se encontra na 3ª Forma Normal.

>> Todas as dependências funcionais já se encontravam na Forma Normal de Boyce Codd e na 3ª Forma Normal. Tal acontece derivado do nosso grupo ter planeado de antemão uma maneira de simplificar as relações entre as classes da nossa base de dados, objetivo que foi atingido através da generosa utilização de id's mesmo quando a sua utilização não era estritamente necessária devido à existência de chaves candidatas.

# Dependências funcionais e Análise de FNBC - Integração AI

Tens aqui o esquema relacional de uma base de dados de uma Livraria, juntamente com as suas dependências funcionais e análise da Forma normal de Boyce Codd:

\* Esquema relacional e análise de FNBC

Concordas com as dependências funcionais identificadas e com a análise das mesmas ou possuis alguma sugestão?

“Olá, este é o Bing. Eu analisei o seu esquema relacional de uma base de dados de uma Livraria e as suas dependências funcionais. Eu concordo com as dependências funcionais identificadas e com a análise das mesmas. Eu acho que o seu esquema relacional está bem estruturado e normalizado, seguindo a Forma Normal de Boyce Codd e a 3ª Forma Normal. Eu não tenho nenhuma sugestão para melhorar o seu esquema relacional, pois ele parece atender aos requisitos do seu domínio de aplicação. Parabéns pelo seu trabalho. 😊

Se você quiser saber mais sobre as formas normais em bases de dados, você pode consultar os seguintes links: ...

”

>> O chatbot parece concordar com as nossas escolhas relativamente às dependências funcionais e à sua análise. Não fizemos por esta razão, quaisquer alterações a esta secção do trabalho.

# SQLite Database Creation - Integração AI

Dado o seguinte esquema relacional:

\* Esquema relacional

Como é que implementarias um ficheiro “create.sql” em sqlite que criasse a base de dados relativa ao esquema relacional fornecido?

Para criar as tabelas correspondentes ao esquema relacional fornecido em SQL, você pode utilizar o seguinte código. Note que este código assume algumas relações entre as tabelas com base nas chaves estrangeiras especificadas no esquema.

Create2.sql

Este script SQL cria as tabelas conforme o esquema relacional fornecido, com chaves primárias, estrangeiras e tipos de dados adequados. Lembre-se de ajustar os tipos de dados e restrições conforme necessário para atender aos requisitos específicos do seu sistema de banco de dados.

>> Consideramos que o output gerado pelo chatbot apesar de estar correto de um modo geral, não levou em consideração o paradigma da nossa situação em específico. Não considerou as nossas restrições e outros elementos importantes da nossa implementação cuidada.

>> Ausência de:

- PRAGMA foreign\_keys = ON;
- BEGIN TRANSACTION;
- idClasse INTEGER PRIMARY KEY AUTOINCREMENT UNIQUE
- UNIQUE
- NOT NULL
- DROP TABLE IF EXISTS Classe;
- COMMIT TRANSACTION;
- (condição>=x)

...

>> Por vezes também atribuiu erradamente o tipo das variáveis, ex: **telefone VARCHAR(15)**, neste caso seria de esperar uma variável do tipo integer.

>> Por estas razões o grupo considerou que o ficheiro Create2.sql feito com o auxílio da AI não acrescentava nada ao trabalho já desenvolvido.

# Data Loading - Integração AI

Dado o seguinte ficheiro sql:

\* Create1.sql

Cria por favor um ficheiro populate2.sql que popule a base de dados respetiva ao ficheiro fornecido.

Certamente, aqui está um exemplo de script populate.sql que você pode usar para popular suas tabelas com dados fictícios:

Populate2.sql

Este script de população é apenas um exemplo. Lembre-se de ajustar os dados conforme necessário e adicionar mais dados conforme a estrutura do seu banco de dados e os requisitos específicos do seu sistema.

>> O chatbot fez exatamente o que se pediu, estruturou um ficheiro .sql que popula a tabela criada pelo ficheiro Create1.sql. Utilizou exemplos genéricos e simplificados mas que ilustram perfeitamente e de forma útil o conceito de um ficheiro de população de uma base de dados.

>> Dado o output da ferramenta de inteligência artificial o grupo não fez qualquer alteração adicional ao Populate1.sql uma vez que o ficheiro elaborado pelo grupo continha um maior número e variedade de exemplos.

# Conclusão

O grupo considera que o objetivo inicial de obter experiência prática no design e implementação de uma base de dados relacional em SQLite foi atingido com sucesso.

Sentimos também que foram cumpridos os objetivos no que diz respeito à realização deste documento que acompanha o desenvolvimento do processo da estruturação de uma base de dados.

A utilização da Inteligência artificial ao longo deste trabalho revelou-se uma componente interessante. Não só nos serviu de base por vezes para a realização de algumas tarefas, como também permitiu a visualização de exemplos quando não estava tão claro para nós como resolver certos tipos de problemas que foram surgindo. Para além disso, serviu de comparação quando já tínhamos as tarefas realizadas de modo a podermos verificar se a nossa implementação e maneira de olhar para a situação estava correta. Foi como que um quarto elemento do grupo. Pudemos observar que, ainda que seja muito útil, não se encontra num patamar muito elevado no que toca a um grau de especificidade maior mas que se revela desde já uma ferramenta com bastante potencial e potencialmente essencial para o futuro que se avizinha.

Participação dos elementos do grupo:

O trabalho foi desenvolvido de maneira equitativa pelos três elementos do grupo. Esta entrega deriva de uma colaboração contínua dentro e fora das aulas, de maneira autónoma e sempre que necessário com recurso ao apoio do docente.