U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Bases de Dados

Stand de Automóveis

2LEIC06 - Grupo 603

(19 de novembro de 2023)

David Carvalho up202208654@fe.up.pt

Leonardo Magalhães up202208726@fe.up.pt

Tiago Pinto up202206280@fe.up.pt

Descrição

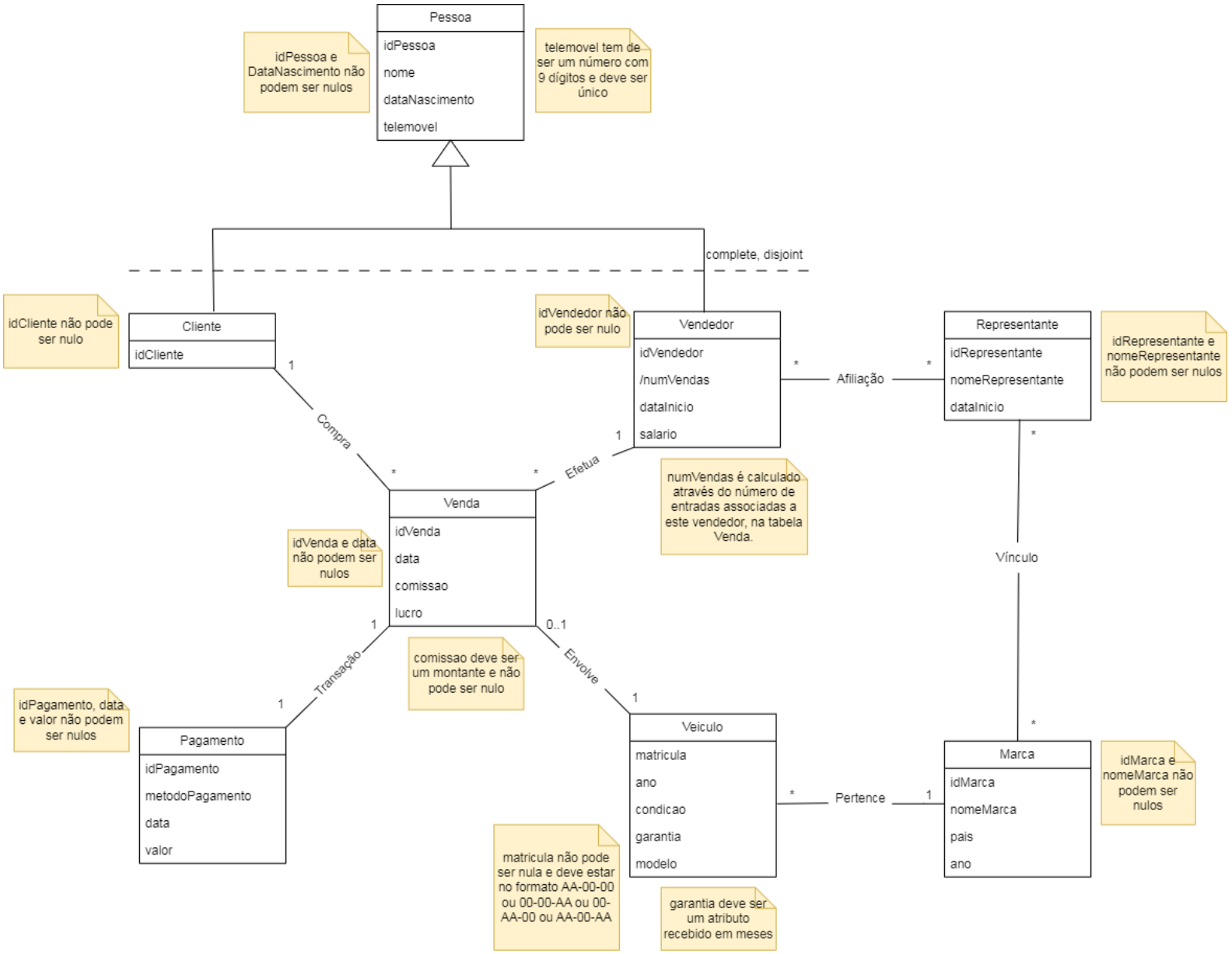
Um stand de automóveis pretende informatizar o seu serviço de **vendas**. É necessário armazenar informações de cada **pessoa**, o nome, a data de nascimento e o telemóvel, que tanto pode ser um **vendedor** do stand, onde se pretende guardar o número de vendas, o salário e data que começou a trabalhar no stand, como também um **cliente**. Interessa também saber a matrícula, o ano, a garantia, condição e modelo de cada **veículo**. De cada veículo é também fundamental saber a **marca** e o seu respetivo nome, país e ano de origem. As marcas são patenteadas por **representantes**, que usam os **vendedores** como intermediários, sendo estes últimos os responsáveis pelas **vendas** dos **veículos** de determinada **marca** ao **cliente**, sendo necessário guardar a data da **venda**, lucro e comissão correspondente. Por último, é igualmente relevante guardar informações sobre o **pagamento**, como o seu método, data e valor.

Esta informatização permite realizar diversas consultas importantes, tais como, por exemplo, calcular a média de vendas, determinar o lucro de cada venda, calcular a média de lucro por vendedor, identificar quais veículos foram vendidos, exibindo informações como data da venda, preço de venda e lucro, obter o valor do pagamento e o método utilizado em cada venda.

Também pode incidir sobre outras áreas, como verificar quantos carros de um modelo específico estão atualmente em stock, calcular o gasto mensal com salários, determinar quantos veículos um cliente específico comprou ao longo do tempo e verificar a garantia de um veículo.

Essas consultas contribuem significativamente para o funcionamento eficaz do stand de automóveis, ajudando a manter um controlo financeiro, permitindo uma gestão eficiente das vendas, dos custos e da satisfação do cliente.

Diagrama de Classes em UML - versão atualizada



Modelo Relacional

Pessoa(idPessoa, nome, dataNascimento, telemovel)

Cliente(idPessoa->Pessoa, idCliente)

Vendedor(idPessoa->Pessoa, idVendedor, numVendas, dataInicio, salario)

Representante(idRepresentante, nomeRepresentante, dataInicio)

VendedorRepresentante(idVendedor->Vendedor, idRepresentante->Representante)

Marca(idMarca, nomeMarca, pais, ano)

RepresentanteMarca(idRepresentante->Representante, idMarca->Marca)

Veiculo(matricula, ano, condicao, garantia, modelo, idMarca->Marca)

Pagamento(idPagamento, metodoPagamento, data, valor)

Venda(idVenda, data, comissao, lucro, idCliente->Cliente, idVendedor->Vendedor, idPagamento->Pagamento, matricula->Veiculo)

Modelo Relacional - integração de AI

Para nos auxiliar e otimizar o modelo relacional, usamos o ChatGPT, que é um software que utiliza inteligência artificial para processar e gerar texto com base em padrões e informações pré-existentes.

Para tal, fizemos a seguinte instrução:

> Pessoa(idPessoa, nome, dataNascimento, telemovel)
> Cliente(idPessoa->Pessoa, idCliente)
> Vendedor(idPessoa->Pessoa, idVendedor, numVendas, dataInicio, salario)
> Representante(idRepresentante, nomeRepresentante, dataInicio)
> VendedorRepresentante(idVendedor->Vendedor, idRepresentante->Representante)
> Marca(idMarca, nomeMarca, pais, ano)
> RepresentanteMarca(idRepresentante->Representante, idMarca->Marca)
> Veiculo(matricula, ano, condicao, garantia, modelo, idMarca->Marca)
> Pagamento(idPagamento, metodoPagamento, data, valor)
> Venda(idVenda, data, comissao, lucro, idCliente->Cliente, idVendedor->Vendedor, idPagamento->Pagamento, matricula->Veiculo)
>
> Analise o modelo relacional e corrija-o, caso necessário.

O ChatGPT deu a seguinte resposta:

ChatGPT

O modelo que você enviou está geralmente bem estruturado. Ele inclui entidades principais, chaves primárias, e relacionamentos aparentemente relevantes para representar as associações entre essas entidades. No entanto, sempre é bom revisar conforme as necessidades específicas do sistema e os requisitos do negócio.

Após pedir à AI uma análise mais detalhada, esta respondeu com uma revisão individual de cada tabela e reforçou a importância de manter chaves primárias exclusivas.

Em relação ao output da AI, achamos que este pode ser útil, dado que alerta, com eficiência, para erros relacionados com chaves estrangeiras e primárias.

Contudo, com base nas respostas, decidimos manter o nosso modelo inicial.

Dependências Funcionais e Análise da Forma Normal

Vamos analisar estas relações e identificar as dependências funcionais, violações da Forma Normal de Boyce-Codd (FNBC) e violações da Terceira Forma Normal (3FN) em cada uma das tabelas:

1. Pessoa(idPessoa, nome, dataNascimento, telemovel):

Dependências Funcionais: não há dependências funcionais além da chave primária.

- Não há violações da 3FN, pois cada atributo é diretamente dependente da chave primária.
- Não há violações da FNBC, pois a chave primária determina todos os outros atributos

2.Cliente(idPessoa->Pessoa, idCliente):

Dependências Funcionais:

idPessoa -> Pessoa

- Não há violações da 3FN, pois cada atributo é diretamente dependente da chave primária.
- Não há violações da FNBC, pois a chave primária é única e determina diretamente todos os outros atributos.

3. Vendedor(idPessoa->Pessoa, idVendedor, numVendas, dataInicio, salario):

Dependências Funcionais:

idPessoa -> Pessoa

- Não há violações da 3FN, pois cada atributo é diretamente dependente da chave primária.

- Não há violações da FNBC, pois a chave primária é única e determina diretamente todos os outros atributos.

4. Representante(idRepresentante, nomeRepresentante, dataInicio):

Dependências Funcionais: não há dependências funcionais explícitas além da chave primária.

- Não há violações da 3FN, pois cada atributo é diretamente dependente da chave primária.

- Não há violações da FNBC, pois a chave primária determina todos os outros atributos.

5. VendedorRepresentante(idVendedor->Vendedor, idRepresentante->Representante):

Dependências Funcionais:

idVendedor -> Vendedor

idRepresentante -> Representante

- Não há violações da 3FN, pois cada atributo é diretamente dependente das chaves primárias.

- Não há violações da FNBC, pois as chaves primárias são superchaves.

6. Marca(idMarca, nomeMarca, pais, ano):

Dependências Funcionais: não há dependências funcionais explícitas além da chave primária.

- Não há violações da 3FN, pois cada atributo é diretamente dependente da chave primária.

- Não há violações da FNBC, pois a chave primária determina todos os outros atributos.

7. RepresentanteMarca(idRepresentante->Representante, idMarca->Marca):

Dependências Funcionais:

idRepresentante -> Representante

idMarca -> Marca

- Não há violações da 3FN, pois cada atributo é diretamente dependente das chaves primárias.

- Não há violações da FNBC, pois as chaves primárias são superchaves.

8. Veiculo(matricula, ano, condicao, garantia, modelo, idMarca->Marca):

Dependência funcional:

idMarca -> Marca

- Não há violações da 3FN, pois cada atributo é diretamente dependente da chave primária.

- Não há violações da FNBC, pois a chave primária é única e determina diretamente todos os outros atributos.

9. Pagamento(idPagamento, metodoPagamento, data, valor):

Dependências Funcionais: não há dependências funcionais explícitas além da chave primária.

- Não há violações da 3FN, pois cada atributo é diretamente dependente da chave primária.

- Não há violações da FNBC, pois a chave primária é única e determina diretamente todos os outros atributos.

10. Venda(idVenda, data, comissao, lucro, idCliente->Cliente, idVendedor->Vendedor, idPagamento->Pagamento, matricula->Veiculo):

Dependências funcionais:

idCliente -> Cliente

idVendedor -> Vendedor

idPagamento -> Pagamento

matricula -> Veiculo

- Não há violações da 3FN, pois cada atributo é diretamente dependente da chave primária.

- Não há violações da FNBC, pois a chave primária é única e determina diretamente todos os outros atributos.

Com base na análise acima, todas as relações estão na Forma Normal de Boyce-Codd (FNBC) e na Terceira Forma Normal (3FN). Portanto, não é necessário decompor nenhuma relação.

Dependências Funcionais e Análise da Forma Normal - integração de AI

Apresentamos a seguinte instrução ao ChatGPT:

> Pessoa(idPessoa, nome, dataNascimento, telemovel)
> Cliente(idPessoa->Pessoa, idCliente)
> Vendedor(idPessoa->Pessoa, idVendedor, numVendas, dataInicio, salario)
> Representante(idRepresentante, nomeRepresentante, dataInicio)
> VendedorRepresentante(idVendedor->Vendedor, idRepresentante->Representante)
> Marca(idMarca, nomeMarca, pais, ano)
> RepresentanteMarca(idRepresentante->Representante, idMarca->Marca)
> Veiculo(matricula, ano, condicao, garantia, modelo, idMarca->Marca)
> Pagamento(idPagamento, metodoPagamento, data, valor)
> Venda(idVenda, data, comissao, lucro, idCliente->Cliente, idVendedor->Vendedor, idPagamento->Pagamento, matricula->Veiculo)
>
> Verifique a 3 Forma Normal e a Forma Normal Boyce-Codd.

A ferramenta AI confirmou que a 3ª Forma Normal e a Forma Normal Boyce-Codd estão verificadas. Contudo, reforçou que, sendo uma ferramenta de AI, apenas consegue fazer uma análise superficial, pelo que a análise das dependências podem necessitar intervenção humana. Desta forma, a utilização de AI foi útil para garantir que a implementação do modelo relacional não viola a forma normal.

Criação de código SQL- Integração de AI

Após a realização do modelo relacional, construímos o nosso código SQL e pedimos ao ChatGPT para o melhorar. Após a leitura do código e após aplicarmos um sentido crítico ao mesmo e compararmos com o código já feito pelo grupo, apercebemo-nos que nos faltavam alguns detalhes, como por exemplo:

- ON UPDATE/ON DELETE
- Alguns erros de sintaxe básica, como por exemplo, um “IdVenda” em vez de “idVenda”, embora seja um erro básico, estava a comprometer a nossa tabela e tivemos de alterar o código neste sentido.

Após todas estas alterações, pedimos ao ChatGPT para nos rever todo código sql e, para nossa surpresa, ainda não estava 100% funcional, pois ainda havia incongruências.

Primeiramente, tinha sido gerado código com “ON DELETE SET DEFAULT”, que, ao correrrmos o programa e o testarmos, percebemos que este comando não era suportado pelo Sqlite depois de confrontarmos o ChatGPT com o erro que era imprimido.

É preciso dizer que o ChatGPT não conseguia ler mentes e, por isso, após nos dar o código, foi preciso mudar alguns UPDATES/DELETES, já que não iam ao encontro do que queremos para a nossa base de dados.

Após isso, conseguimos corrigir todos os erros que o código estava a dar e foi-nos possível correr toda a base de dados, fazendo as alterações que queríamos.

Inserção de dados- Integração de AI

Após a definição final do código SQL da nossa base de dados, desempenhamos a inserção de múltiplos dados. Posteriormente, utilizamos o ChatGPT, que desempenhou um papel crucial ao nos ajudar a expandir, aumentar e desenvolver os conjuntos de dados que tínhamos inserido inicialmente. Este auxílio permitiu uma abordagem mais abrangente na representação e manipulação de dados, o que proporcionou melhorias significativas no nosso projeto.

Sendo assim, este auxílio refinou e influenciou positivamente o desenvolvimento de dados e a correção dos já existentes, o que resultou numa implementação mais robusta e eficaz do nosso sistema.

Participação no trabalho

David Carvalho (33.33%)

Leonardo Magalhães (33.33%)

Tiago Pinto (33.33%)