

**Agência de Viagens**

Relatório de projeto

Bases de Dados

Mestrado Integrado em Engenharia Informática e Computação (MIEIC)

2ºAno 2ºSemestre - 2020/2021

Turma 4 - Grupo 3:

Maria Leonor Beirão up201806798

Mónica Pereira up201905753

Nuno Castro up202003324

Índice

[1 – Introdução](#_heading=h.gjdgxs) **2**

[1.1 Apresentação do projeto](#_heading=h.30j0zll) 2

[1.2 Descrição do projeto](#_heading=h.1fob9te) 3

[2 – Modelo Conceptual](#_heading=h.ikspnh6z5dde) **4**

[3 – Esquema Relacional](#_heading=h.2et92p0) **5**

[4 – Análise de dependências funcionais e formas normais](#_heading=h.tyjcwt) **6**

[4.1 Dependências Funcionais](#_heading=h.yb3gbn4p0wcu) 6

[4.2 Formas Normais](#_heading=h.w0soeh4x1pbp) 8

[4.2.1 Forma Normal Boyce-Codd](#_heading=h.vbs9hbvqx5my) 8

[4.2.2 3ª Forma Normal](#_heading=h.gojxlbck5pge) 9

[5 – Implementação de restrições](#_heading=h.3ybm1lg214ab) **10**

[6 – Listagem de interrogações e gatilhos](#_heading=h.1t3h5sf) **16**

[6.1 Interrogações](#_heading=h.4d34og8) 16

[6.2 Gatilhos](#_heading=h.2s8eyo1) 16

# 1 – Introdução

## 1.1 Apresentação do projeto

Este projeto surge no âmbito da unidade curricular de Bases de Dados em que nos foi proposto a elaboração e interrogação de uma base de dados num determinado contexto.

O tema escolhido pelo grupo foi uma base de dados no contexto de uma agência de viagens e os requisitos iniciais para a realização deste projeto são os seguintes:

* definição do modelo conceptual em UML;
* definição do esquema relacional;
* análise de dependências funcionais e formas normais;
* criação da base de dados em SQLite e implementação de restrições;
* povoação da base de dados;
* interrogação da base de dados;
* adição de gatilhos.

## 1.2 Descrição do projeto

Uma agência de viagens tem como objetivo planear viagens dos seus clientes.

O cliente, após fornecer os seus dados, será auxiliado por um agente para proceder à programação da sua viagem.

A viagem terá um tipo, local de origem e destino, número de viajantes, data de início e fim, um custo e desconto aplicado.

À viagem podem ser associadas deslocações. Deslocações estas que poderão ser de ida ou de regresso e terão uma ordem. As deslocações terão como atributo duração de viagem, local e hora de partida. As deslocações podem ser de avião, barco, autocarro ou comboio, cada um com os seus lugares marcados e local de embarque.

Tanto à viagem como à deslocação está associado um país de partida e de chegada.

À viagem podem também ser associados quartos de hotéis para estadia. Os hotéis têm nome, morada, código postal, telefone e número de quartos. Os quartos atribuídos podem ter tipologias diferentes, número de identificação e andar do edifício.

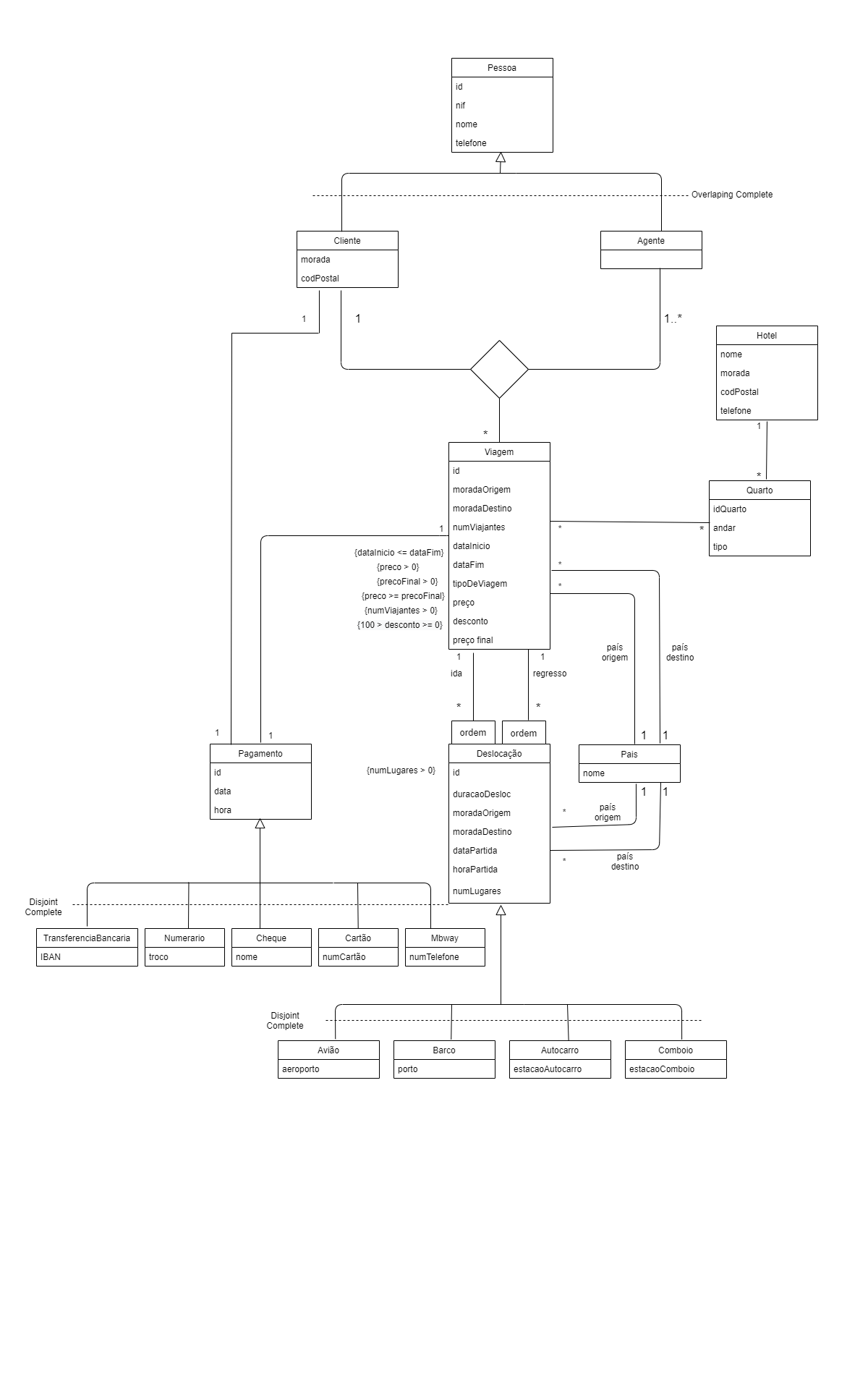
Ainda referente à viagem será também associado um pagamento, caracterizado com um identificador, data e hora em que foi efetuado. O pagamento pode ser efetuado através de numerário, cheque bancário, cartão de débito/crédito, transferência bancária ou MBway.

Caso o pagamento seja efetuado através de:

* Numerário, devolve-se o respectivo troco;
* Cheque bancário, guarda-se o nome de quem o passou;
* Cartão de débito/crédito, guarda-se o número de cartão;
* Transferência bancária guarda-se o IBAN da respectiva conta;
* MBway, guarda-se o número de telemóvel.

# 2 – Modelo Conceptual

O diagrama que se segue é representativo do modelo conceptual em UML para a base de dados que pretendemos criar.



# 3 – Esquema Relacional

Após o desenvolvimento do modelo conceptual procedeu-se à conversão do mesmo para o modelo relacional.

Pessoa(id, nif, nome, telefone)

Cliente(id -> Pessoa, morada, codPostal)

Agente(id -> Pessoa)

Viagem(id, moradaOrigem, moradaDestino, numViajantes, dataInicio, dataFim, tipoDeViagem, preço, desconto, preçoFinal, paisOrigem -> País, paisDestino -> País)

{dataFim > dataInicio}

{preco > 0}

{precoFinal > 0}

{preco >= preco final}

{100 > desconto > 0}

ClienteViagemAgente(idCliente -> Cliente, idViagem -> Viagem, idAgente -> Agente)

País(nome)

Deslocação(id, duracaoViagem, moradaOrigem, moradaDestino, dataPartida, horaPartida, numLugares, paisOrigem -> País, paisDestino -> País)

{duracaoViagem > 0}

{numLugares > 0}

OrdemIda(idViagem -> Viagem, idDeslocacao -> Deslocação, ordem)

OrdemRegresso(idViagem -> Viagem, idDeslocacao -> Deslocação, ordem)

Aviao(idDeslocacao -> Deslocação, aeroporto)

Barco(idDeslocacao -> Deslocação, porto)

Autocarro(idDeslocacao -> Deslocação, estacao)

Comboio(idDeslocacao -> Deslocação, estacao)

Hotel(nome, morada, codPostal, telefone)

Quarto(idQuarto, andar, tipo, [nomeHotel, moradaHotel] -> Hotel)

ViagemEstadia(idViagem -> Viagem, [idQuarto, nomeHotel, moradaHotel] -> Quarto)

Pagamento(id, data, hora, idCliente -> Cliente, [idViagem, preçoFinal] -> Viagem)

TransferênciaBancária(idPagamento -> Pagamento, IBAN)

Numerário(idPagamento -> Pagamento, troco)

Cheque(idPagamento -> Pagamento, nome)

CartãoDeCredito(idPagamento -> Pagamento, numCartao)

Mbway(idPagamento -> Pagamento, numTelefone)

# 4 – Análise de dependências funcionais e formas normais

## 4.1 Dependências Funcionais

Neste ponto são referidas as dependências funcionais para cada tabela.

Pessoa(id, nif, nome, telefone)

id -> nif, nome, telefone

nif -> nome

Cliente(id -> Pessoa, morada, codPostal)

id -> morada, codPostal

morada -> codPostal

Agente(id -> Pessoa)

Viagem(id, moradaOrigem, moradaDestino, numViajantes, dataInicio, dataFim, tipoDeViagem, preço, desconto, preçoFinal, paisOrigem -> País, paisDestino -> País)

id -> moradaOrigem, moradaDestino, numViajantes, dataInicio, dataFim, tipoDeViagem, preco, desconto, precoFinal, paisOrigem, paisDestino

moradaOrigem -> paisOrigem

moradaDestino -> paisDestino

preco, desconto -> precoFinal

ClienteViagemAgente(idCliente -> Cliente, idViagem -> Viagem, idAgente -> Agente)

País(nome)

Deslocação(id, duracaoViagem, moradaOrigem, moradaDestino, dataPartida, horaPartida, numLugares, paisOrigem -> País, paisDestino -> País)

id-> duracaoViagem, moradaOrigem, moradaDestino, dataPartida, horaPartida, numLugares, paisOrigem, paisDestino

moradaOrigem -> paisOrigem

moradaDestino -> paisDestino

OrdemIda(idViagem -> Viagem, idDeslocacao -> Deslocação, ordem)

idViagem, idDeslocacao -> ordem

OrdemRegresso(idViagem -> Viagem, idDeslocacao -> Deslocação, ordem)

idViagem, idDeslocacao -> ordem

Avião(idDeslocacao -> Deslocação, aeroporto)

idDeslocacao -> aeroporto

Barco(idDeslocacao -> Deslocação, porto)

idDeslocacao -> porto

Autocarro(idDeslocacao -> Deslocação, estacao)

idDeslocacao -> estacao

Comboio(idDeslocacao -> Deslocação, estacao)

idDeslocacao -> estacao

Hotel(nome, morada, codPostal, telefone)

nome, morada -> telefone, codPostal

morada -> codPostal

Quarto(idQuarto, [nomeHotel, moradaHotel] -> Hotel, andar, tipo)

idQuarto, nomeHotel, moradaHotel -> andar, tipo

ViagemEstadia(idViagem -> Viagem, [idQuarto, nomeHotel, moradaHotel] -> Quarto)

Pagamento(id, data, hora, idCliente -> Cliente, idViagem -> Viagem)

id -> data, hora, idCliente, idViagem

TransferênciaBancária(idPagamento -> Pagamento, IBAN)

idPagamento -> IBAN

Numerário(idPagamento -> Pagamento, troco)

idPagamento -> troco

Cheque(idPagamento -> Pagamento, nome)

idPagamento -> nome

CartãoDeCredito(idPagamento -> Pagamento, numCartao)

idPagamento -> numCartao

Mbway(idPagamento -> Pagamento, numTelefone)

idPagamento -> numTelefone

## 

## 4.2 Formas Normais

Neste ponto é feita uma análise referente a violações de formas normais para cada tabela do nosso modelo, nomeadamente a Forma Normal Boyce-Codd e a 3ª Forma Normal.

### 4.2.1 Forma Normal Boyce-Codd

Para cada dependência funcional A->B, A é uma chave.

Tabelas que apresentam violações à BCNF e respectivas violações:

Pessoa(id, nif, nome, telefone)

nif -> nome

Cliente(id -> Pessoa, morada, codPostal)

morada -> codPostal

Viagem(id, moradaOrigem, moradaDestino, numViajantes, dataInicio, dataFim, tipoDeViagem, preço, desconto, preçoFinal, paisOrigem -> País, paisDestino -> País)

moradaOrigem -> paisOrigem

moradaDestino -> paisDestino

preco, desconto -> precoFinal

Deslocação(id, duracaoViagem, moradaOrigem, moradaDestino, dataPartida, horaPartida, numLugares, paisOrigem -> País, paisDestino -> País)

moradaOrigem -> paisOrigem

moradaDestino -> paisDestino

Hotel(nome, morada, codPostal, telefone, idViagem -> viagem)

morada -> codPostal

### 4.2.2 3ª Forma Normal

Para cada dependência funcional A->B, A é uma chave ou então B consiste em apenas atributos primos)

Tabelas que apresentam violações à 3º Forma Normal e respectivas violações:

Pessoa(id, nif, nome, telefone)

nif -> nome

Cliente(id -> Pessoa, morada, codPostal)

morada -> codPostal

Viagem(id, moradaOrigem, moradaDestino, numViajantes, dataInicio, dataFim, tipoDeViagem, preço, desconto, preçoFinal, paisOrigem -> País, paisDestino -> País)

moradaOrigem -> paisOrigem

moradaDestino -> paisDestino

preco, desconto -> precoFinal

Deslocação(id, duracaoViagem, moradaOrigem, moradaDestino, dataPartida, horaPartida, numLugares, paisOrigem -> País, paisDestino -> País)

moradaOrigem -> paisOrigem

moradaDestino -> paisDestino

Hotel(nome, morada, codPostal, telefone, idViagem -> viagem)

morada -> codPostal

### 

# 5 – Implementação de restrições

Neste ponto são listadas as restrições definidas para cada tabela.

Pessoa:

* Não podem existir duas pessoas com o mesmo id ou nif.  
  id - PRIMARY KEY  
  nif - UNIQUE
* O nome e telefone de uma pessoa não podem ser nulos.  
  nome - NOT NULL  
  telefone - NOT NULL

Cliente:

* Não podem existir dois clientes com o mesmo id.  
  id - PRIMARY KEY, FOREIGN KEY
* A morada e código postal de uma pessoa não podem ser nulos.  
  morada - NOT NULL  
  codPostal - NOT NULL

Agente:

* Não podem existir dois agentes com o mesmo id.  
  id - PRIMARY KEY, FOREIGN KEY

Pais:

* Não podem existir dois países com o mesmo nome.  
  nome - PRIMARY KEY

Viagem:

* Não podem existir duas viagens com o mesmo id.  
  id - PRIMARY KEY
* O número de viajantes tem que ser superior a 0.  
  numViajantes - CHECK (numViajantes > 0)
* O preço e preço final têm que ser superiores a 0 e o preço final tem que ser inferior ou igual ao preço.  
  preco - CHECK (preco > 0)  
  precoFinal - CHECK (precoFinal > 0)  
  CHECK (preço >= preçoFinal)
* As moradas de origem e destino não podem ser nulas.  
  moradaOrigem - NOT NULL  
  moradaDestino - NOT NULL
* As datas de início e fim não podem ser nulas e a data de fim tem que ser posterior ou igual à de início.  
  dataFim - NOT NULL  
  dataInicio - NOT NULL  
  CHECK (dataFim >= dataInicio)
* O desconto está compreendido entre 0 (inclusive) e 100 (exclusive).  
  CHECK (desconto < 100 AND desconto >= 0)
* A viagem tem que ter um tipo não nulo.  
  tipoDeViagem - NOT NULL
* Os países de origem e destino não podem ser nulos.  
  paisOrigem - NOT NULL, FOREIGN KEY  
  paisDestino - NOT NULL, FOREIGN KEY

Cliente\_Viagem\_Agente:

* Os id’s do cliente, agente, e viagem não podem ser nulos.  
  idCliente - NOT NULL, FOREIGN KEY  
  idViagem - NOT NULL, FOREIGN KEY  
  idAgente - NOT NULL, FOREIGN KEY  
  idCliente, idViagem, idAgente - PRIMARY KEY

Deslocacao:

* Não podem existir duas deslocações com o mesmo id.  
  id - PRIMARY KEY
* As moradas de origem e destino não podem ser nulas.  
  moradaOrigem - NOT NULL  
  moradaDestino - NOT NULL
* A duração do deslocamento, data e hora de partida não podem ser nulas.

duracaoDesloc - NOT NULL  
dataPartida - NOT NULL  
horaPartida - NOT NULL

* O número de lugares reservados para a deslocação tem que ser superior a 0.

CHECK (numLugares > 0)

* Os países de origem e destino não podem ser nulos.  
  paisOrigem - NOT NULL, FOREIGN KEY  
  paisDestino - NOT NULL, FOREIGN KEY

Ordem\_Ida:

* Não podem existir duas ordens de ida definidas pelo mesmo conjunto id de viagem e id de deslocação.  
  idViagem - FOREIGN KEY  
  idDeslocacao - FOREIGN KEY  
  idViagem, idDeslocacao - PRIMARY KEY
* A ordem do deslocamento tem que ser superior a 0.  
  CHECK (ordem > 0)

Ordem\_Regresso:

* Não podem existir duas ordens de regresso definidas pelo mesmo conjunto id de viagem e id de deslocação.  
  idViagem - FOREIGN KEY  
  idDeslocacao - FOREIGN KEY  
  idViagem, idDeslocacao - PRIMARY KEY
* A ordem do deslocamento tem que ser superior a 0.  
  CHECK (ordem > 0)

Aviao:

* Não podem haver deslocamentos de avião que possuam o mesmo id de deslocação.  
  idDeslocacao - PRIMARY KEY, FOREIGN KEY
* O aeroporto de partida tem que ter um nome.  
  aeroporto - NOT NULL

Barco:

* Não podem haver deslocamentos de barco que possuam o mesmo id de deslocação.  
  idDeslocacao - PRIMARY KEY, FOREIGN KEY
* O porto de partida tem que ter um nome.  
  porto - NOT NULL

Autocarro:

* Não podem haver deslocamentos de autocarro que possuam o mesmo id de deslocação.  
  idDeslocacao - PRIMARY KEY, FOREIGN KEY
* A estação de partida tem que ter um nome.  
  estacao - NOT NULL

Comboio:

* Não podem haver deslocamentos de comboio que possuam o mesmo id de deslocação.  
  idDeslocacao - PRIMARY KEY, FOREIGN KEY
* A estação de partida tem que ter um nome.  
  estacao - NOT NULL

Hotel:

* Não podem existir hotéis com o mesmo nome e morada  
  nome - NOT NULL  
  morada - NOT NULL  
  nome, morada - PRIMARY KEY
* Um hotel tem que ter um número de telefone.  
  telefone - NOT NULL

Quarto:

* Não podem existir dois quartos com o mesmo id, nome e morada do hotel a que pertencem.  
  hotelNome - NOT NULL, FOREIGN KEY  
  hotelMorada - NOT NULL, FOREIGN KEY  
  idQuarto, hotelNome, hotelMorada - PRIMARY KEY
* Um quarto tem que ter uma tipologia.  
  tipo - NOT NULL

Viagem\_Estadia:

* Os id’s viagem e quarto, nome e morada do hotel não podem ser nulos.  
  idViagem - NOT NULL, FOREIGN KEY  
  idQuarto - NOT NULL, FOREIGN KEY  
  hotelNome - NOT NULL, FOREIGN KEY

hotelMorada - NOT NULL, FOREIGN KEY  
idCliente, idQuarto, hotelNome, hotelMorada - PRIMARY KEY

Pagamento:

* Não podem existir dois pagamentos com o mesmo id.  
  id - PRIMARY KEY
* A data e hora em que o pagamento foi efetuado não podem ser nulas.  
  data - NOT NULL  
  hora - NOT NULL
* Um pagamento tem que ter um cliente e uma viagem associados.  
  idCliente - NOT NULL, FOREIGN KEY  
  idViagem - NOT NULL, FOREIGN KEY

Transferencia\_Bancaria:

* Não podem haver dois pagamentos por transferência bancária com o mesmo id de pagamento.  
  idPagamento - PRIMARY KEY, FOREIGN KEY
* Uma transferência bancária tem um IBAN associado.  
  IBAN - NOT NULL

Numerario:

* Não podem haver dois pagamentos por numerário com o mesmo id de pagamento.  
  idPagamento - PRIMARY KEY, FOREIGN KEY

Cheque:

* Não podem haver dois pagamentos por cheque com o mesmo id de pagamento.  
  idPagamento - PRIMARY KEY, FOREIGN KEY
* Um pagamento com cheque tem o nome do assinante associado.  
  nome - NOT NULL

Cartao:

* Não podem haver dois pagamentos com cartão com o mesmo id de pagamento.  
  idPagamento - PRIMARY KEY, FOREIGN KEY
* Um pagamento com cartão tem um número de cartão associado.  
  numCartao - NOT NULL

Mbway:

* Não podem haver dois pagamentos por MBWay com o mesmo id de pagamento.  
  idPagamento - PRIMARY KEY, FOREIGN KEY
* Um pagamento por MBWay tem um número de telefone associado.  
  numTelefone - NOT NULL

# 6 – Listagem de interrogações e gatilhos

## 6.1 Interrogações

## 6.2 Gatilhos