# PROJETO DE BASE DE DADOS

## Gerência de Hotel


**Grupo 2MIEIC5_3 :**

Angela Cruz (up201806781)

Marina Dias (up201806787)

Raquel Sepúlveda (up201806664)

abril de 2020

# INTRODUÇÃO

Para o nosso projeto da unidade curricular de “Base de Dados” tivemos a oportunidade realizar uma de base de dados sobre um tema à nossa escolha. Decidimos então realizar uma base de dados sobre a gerência de um hotel em que este terá serviço de quartos, de bar e de spa.

Na nossa base de dados será guardada a informação de cada cliente, a época do ano da sua reserva e o tipo de quarto que reservou. Será ainda guardar a informação de todos os funcionários do hotel tal como os serviços realizados pelos mesmos.

# CONTEXTUALIZAÇÃO

O Hotel pretende guardar a informação dos seu clientes. Para cada **cliente** pretende-se armazenar o seu nome, o seu número de identificação fiscal, o seu email, a respetiva data de nascimento e as suas reservas. Um cliente pode reservar vários quartos.

Os **quartos** são diferenciados pelo seu número e caracterizam-se pelo tipo. Os tipos de quarto existentes no hotel são: quarto solteiro, quarto duplo solteiro, quarto casal e quarto familiar.

Cada **reserva** tem um número, data de início, data de fim e o preço total a pagar pela estadia. O preço varia consoante a **época do ano**, o tipo de quarto e a **taxa de ocupação** do hotel. Cada reserva tem que ter obrigatoriamente um cliente associado. Se um cliente pedir que o seu registo seja eliminado, as reservas associadas a este cliente deixam de existir.

O hotel possui vários **funcionários** com cargos diferentes que operam tanto no spa como no bar e também realizam serviços de quarto.

O hotel presta três tipos de **serviços** diferentes que só podem ser usufruídos por clientes que tenham uma reserva: o **serviço de quartos**, **tratamentos de spa** e serviço de **bebidas**. Os preços de cada um destes serviços variam consoante o que o cliente pedir.

O Bar do hotel oferece um serviço de acumulação de pontos por bebida. Consoante o preço de cada bebida acumula-se pontos diferentes num **cartão de pontos** único por estadia de cada cliente. Esses pontos serão somados e multiplicados por dez. A totalidade de pontos do cartão permite obter um desconto no valor total da fatura de serviços requisitados pelo cliente que este pagará no check-out do hotel. Cada ponto no cartão equivale a 1 cêntimo de desconto na fatura.

Na **fatura** é mencionado a data de emissão, os serviços requisitados e o valor total a pagar, que é calculado através da soma de todos os serviços requisitados menos o valor descontado através dos cartões de pontos. Cada fatura está ligada a uma reserva.

## CLASSES E RESPECTIVOS ATRIBUTOS

**Pessoa**
- nif
- nome
- email
- nr_tel

**Cliente**
- data_nascimento

**Funcionário**
- cargo

**Reserva**
- nr_reserva
- data_inicio
- data_fim

**Época do ano**
- data_inicio
- data_fim

**Taxa de Ocupação**
- percentagem

**Quarto**
- nr_quarto
- tipo_quarto

**Serviço**
- nome
- preco
- data

**Tipo de Bebida**

**Tratamento de Spa**

**Serviço de Quartos**

**Cartão de Pontos**
- nr_pontos

**Fatura**
- data
- total

# ESQUEMA RELACIONAL E ANÁLISE DE DEPENDÊNCIAS FUNCIONAIS E FORMAS NORMAIS

**Pessoa**(nif, nome, email, nr_tel)

- {nif -> nome, email, nr_tel}

**Cliente**(nif->Pessoa, data_nascimento)

- {nif -> data_nascimento}

**Funcionario**(nif->Pessoa, cargo)

- {nif -> cargo}

**Reserva**(nr_reserva, nr_quarto->Quarto,nif->Pessoa,  data_inicio, data_fim)

- {nr_reserva -> nr_quarto, nif, data_inicio, data_fim}

**Quarto**(nr_quarto, tipo_quarto)

- {nr_quarto -> tipo_quarto}

**EpocaDoAno**(data_inicio, data_fim)

**TaxaDeOcupacao**(percentagem)

**Servico**(id_servico, nome, preco, data)

- {id_servico -> nome, preco, data}

**ServicoDeQuartos**(id_servico->Servico, nome, preco, data)

- {id_servico -> nome, preco, data}

**TratamentoDeSpa**(id_servico->Servico, nome, preco, data)

- {id_servico -> nome, preco, data}

**TipodeBebida**(id_servico->Servico, nome, preco, data)

- {id_servico -> nome, preco, data}

**CartaoDePontos**(id_cartao, nif->Pessoa_, nrl_pontos)

- {id_cartao-> nif, nr_pontos}

**Fatura**(id_fatura, id_cartao->Cartao, data, total)

- {id_fatura->id_cartao, data, total}

**PrecoDeQuarto**(nr_quarto->Quarto, percentagem->TaxaDeOcupacao, data_inicio->EpocaDoAno, data_fim->EpocaDoAno, preco)

- {nr_quarto, percentagem, data_inicio, data_fim-> preco}

**QuantidadeDeServiçosPrestados**(id_fatura->Fatura, id_servico->Servico, quantidade);

- {id_fatura, id_servico->quantidade}

**QuantidadeDeBebidas**(id_cartao->Cartao, id_servico->Servico, quantidade)

- {id_cartao, id_servico->quantidade}

**DescontoAplicado**(id_cartao->Cartao, id_fatura->Fatura, desconto)

- {id_cartao, id_fatura->desconto}

Em todas as dependências funcionais da nossa base de dados, o lado esquerdo é sempre uma chave primária, logo nenhuma dependência viola a Forma Normal de Boyce-Codd e consequentemente também não violam a 3ª Forma Normal.

# RESTRIÇÕES

Neste projeto usamos restrições para assegurar a manutenção da base de dados.
Usou-se então restrições do tipo NOT NULL para referir a propriedades que são obrigatórias para a existência da classe, UNIQUE para distinguir atributos que não podem ser repetidos em certas classes daí serem denominada únicas e a restrição CHECK para restringir certos aspectos dos atributos. Usaram-se igualmente as restrições ON DELETE CASCADE e ON UPDATE CASCADE, para que atributos que fazem referência a outra classe sejam devidamente atualizados caso estes sejam removidos ou modificados na classe a que fazem referência.

**Pessoa**

- O atributo nif é chave primária e tem que obrigatoriamente conter nove dígitos, daí a restrição CHECK(nif>=100000000 AND nif<=999999999).
- O atributo nome não pode ser nulo, logo é usada a restrição NOT NULL
- Os atributo email e nr_tel são atributos que não podem ser repetido, pois é impossível que duas pessoas possuam o mesmo número de telemóvel ou o mesmo e-mail, daí usarmos a restrição UNIQUE.

**Cliente**

- O atributo nif é chave primária com referência para Pessoa, quando apagado ou modificado na classe Pessoa é igualmente apagado e modificado na classe Cliente, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- O atributo data_nascimento é um atributo necessário para a gerência do hotel para fins de serviços de maiores de idade como por exemplo o Bar do hotel, no caso de bebidas alcoólicas. A restrição usada é NOT NULL.

**Funcionario**

- O atributo nif é chave primária com referência para Pessoa, quando apagado ou modificado na classe Pessoa é igualmente apagado e modificado na classe Funcionario, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- O cargo não pode ser nulo. A um funcionário tem de ser atribuído pelo menos um cargo, daí a restrição NOT NULL.

**Reserva**

- O atributo nr_reserva é chave primária.
- Os atributos nr_quarto e nif fazem referência a Quarto e a Pessoa respetivamente, quando apagados ou modificado na classe Quarto e Pessoa são igualmente apagados e modificados na classe Reserva, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- O atributo data_inicio não pode ser nulo, logo é usada a restrição NOT NULL.
- A data de fim da reserva não pode ser nula e necessita de ser obrigatoriamente uma data depois da data de início, daí as restrições NOT NULL e CHECK(data_fim >data_inicio).

**Quarto**

- O atributo nr_quarto é chave primária.
- O atributo tipo_quarto não pode ser nulo pois todos os quartos têm um tipo, daí a um cliente não pode ser atribuído um quarto sem este atributo. Este atributo também tem de ser de um tipo aceite pelo hotel (solteiro, solteiro duplo, casal, familiar). Com isto são usadas as restrições NOT NULL e CHECK.

**EpocaDoAno**

- A chave primária é composta pelos atributos data_inicio e data_fim. Estes atributos não podem ser nulos e a data_fim tem que ser maior do que a data_inicio. As restrições usadas são NOT NULL e CHECK(data_inicio<data_fim).

**TaxaDeOcupacao**

- A percentagem é a chave primária. Esta não pode ser maior que 100% ou menor que 0%, logo é usada a restrição CHECK(percentagem>=0 AND percentagem<=100).

**Servico**

- O atributo id_servico é a chave primária.
- O atributo preco não pode ser nulo e tem que ser superior ou igual a 0, por esta razão as restrições são NOT NULL e CHECK(preco>=0).
- Os atributos nome e data não podem ser nulos, logo é usada a restrição NOT NULL.

**ServicoDeQuarto TratamentoDeSpa e TipoDeBebidas**

- O atributo id_servico é chave primária e faz referência a Servico, quando apagado ou modificado na classe Servico é igualmente apagado e modificado nas classes ServicoDeQuarto, TratamentoDeSpa e TipoDeBebida, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- Os atributos data, nome e preco não podem ser nulos, logo é usada a restrição NOT NULL. O atributo preco também não pode ser inferior a zero, daí a restrição CHECK(preco>=0).

**Cartao**

- O atributo id_cartao é a chave primária.
- O atributo nif faz referência a Pessoa, quando apagado ou modificado na classe Pessoa é igualmente apagado e modificado na classe Cartao, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- O atributo nr_pontos não pode ser nulo, mesmo no caso de o cliente não ter pontos, o nr_pontos deve estar atribuído a zero. São então usadas as restrições NOT NULL e CHECK(nr_pontos>=0).

**Fatura**

- O atributo id_fatura é chave primária.
- O atributo id_cartao faz referência a Cartao, quando apagado ou modificado na classe Cartao é igualmente apagado e modificado na classe Fatura, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- O atributo data não pode ser nulo, logo é usada a restrição NOT NULL.
- O atributo total não pode ser nulo e tem que ser maior ou igual a zero, são então usadas as restrições NOT NULL e CHECK(total>=0).

**PrecoDeQuarto**

- A chave primária é composta pelos atributos nr_quarto, que faz referência a Quarto, data_inicio e data_fim, que fazem referência a EpocaDoAno, e percentagem, que faz referência a TaxaDeOcupacao. Quando apagados ou modificados nas classes a que fazem referência são igualmente apagados e modificados na classe PrecoDeQuarto, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- O atributo preco não pode ser nulo e tem que ser superior ou igual a zero, logo são usadas as restrições NOT NULL e CHECK(preco>=0).

**QuantidadeDeServiçosPrestados**

- A chave primária é composta pelos atributos id_fatura e id_servico que fazem referência a Fatura e Servico respetivamente, quando apagados ou modificados na classe Fatura e Servico são igualmente apagados e modificados na classe QuantidadeDeServiçosPrestados, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- O atributo quantidade não poder ser nulo e tem que ter um valor maior ou igual a zero, logo são usadas as restrições NOT NULL e CHECK(quantidade>=0).

**QuantidadeDeBebidas**

- A chave primária é composta pelos atributos id_cartao e id_servico que fazem referência a Cartao e Servico respetivamente, quando apagados ou modificados nas classes Cartao e Servico são igualmente apagados e modificados na classe QuantidadeDeBebidas, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- O atributo quantidade não poder ser nulo e tem que ter um valor superior a zero, logo são usadas as restrições NOT NULL e CHECK(quantidade>=0).

**DescontoAplicado**

- A chave primária é composta pelos atributos id_cartao e id_fatura que fazem referência a Cartao e Fatura respetivamente, quando apagados ou modificados na classe Cartao e Fatura são igualmente apagados e modificados na classe DescontoAplicado, daí serem usadas as restrições ON DELETE CASCADE e ON UPDATE CASCADE.
- O atributo desconto não poder ser nulo e tem que ter um valor superior a zero, logo são usadas as restrições NOT NULL e CHECK(desconto>=0).

Diagrama UML HOTEL