Universidade do Minho
Licenciatura em Engenharia Informática

Unidade Curricular de Bases de Dados

Ano Letivo de 2022/2023

Hotel Éden

Ana Pimenta (A100830), Flávia Araújo (A96587), Miguel Pinto (A100815), Pedro Azevedo (A100557)

Março, 2023

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

Hotel Éden

Ana Pimenta (A100830), Flávia Araújo (A96587),

Miguel Pinto (A100815), Pedro Azevedo (A100557)

Março, 2023

Resumo

No seguimento dos métodos de avaliação da UC Bases de Dados, lecionada pelo regente, *Professor Orlando Belo*, visamos a realização de um projeto que consiste na implementação de uma base de dados.

Para efetuar a execução deste plano iremos abordar o tema da hotelaria, construindo um caso prático que ambiciona melhorar o sistema de reservas do hotel da Dona Arlete, tal como abordaremos de forma mais detalhada já de seguida. Tendo em conta este fator, pretendemos criar a base de dados de maneira que esta seja organizada, eficiente e de fácil acesso.

Com o intuito de tornar este projeto o mais viável possível dividimo-lo em três partes que consistem em: Levantamento de Requisitos e posterior construção do Modelo Conceptual, construção do Modelo Lógico e, por fim, a implementação física através do *MySQL*. Tal como revelamos mais adiante, todas estas fases foram sujeitas a métodos de validação das mesmas.

De modo a concretizar este trabalho iremos seguir a metodologia abordada durante as aulas teóricas e práticas.

Área de Aplicação: Bases de Dados

Palavras-Chave: Bases de Dados, Levantamento de Requisitos, Modelo Conceptual, Modelo Lógico, Implementação Física, *SQL*, Povoamento da Base de Dados, *Python*.

Índice

Anexos

Índice de Figuras

Índice de Tabelas

1. Introdução

Com o crescimento do turismo na encantadora cidade do Porto, alguns hotéis enfrentaram desafios ao lidar com o registo de hóspedes e das suas informações nos sistemas de reserva. Neste projeto, pretendemos aprimorar o sistema do simpático Hotel da Dona Arlete, garantindo uma gestão eficiente do estabelecimento e dos dados dos seus estimados clientes.
Com isso, proporcionaremos uma experiência fascinante aos visitantes, ajudando a construir uma reputação excecional para o hotel.

1.1. Apresentação do caso de estudo

Contextualização

O "Hotel Éden" é um hotel sediado no centro do Porto, criado em 2000 por Arlete Gonçalves. Originalmente, Arlete manteve o seu hotel um negócio bastante familiar – um ou outro turista no meio de vários hóspedes locais. Porém, com o avançar dos anos e do turismo em Portugal, Arlete reparou que desde 2019 que o seu hotel não possuía capacidades para albergar a nova enchente de turistas que preenchia a sua cidade. Posto isto, ela decidiu que os dias de gerência individual ficariam para trás, e convidou os seus filhos, Manuel e Susana, para se juntarem ao grupo de administração, e, deste modo, impulsionar o negócio até se tornar um hotel de referência na cidade do Porto.

Juntos, Arlete e os seus filhos dividiram as funções entre os trabalhos de hotelaria "frente da casa" e "fundos da casa". Por outras palavras, Arlete ficaria responsável pela organização da equipa de receção, das reservas, tudo aquilo que interagisse com os clientes, manter-se-ia a cara do negócio para o hotel não perder a sua imagem de marca, a sua essência. Enquanto isso, os filhos focar-se-iam na busca por um novo sistema de organização – no passado, a sua mãe guardara todos os dados em registos físicos ou agendas, e ela planeava dar o salto para o mundo digital através da implementação de um sistema de base de dados, permitindo ser possível, deste modo, a gerência de um negócio de maior dimensão.

Fundamentação

Apesar de não termos acesso à base de dados de um hotel, temos uma ideia das informações cruciais ao funcionamento da mesma. No nosso caso, os gerentes do Hotel Éden, após terem um maior fluxo de reservas e clientes, sentiram necessidade de melhorar o seu sistema de organização para permitir uma melhor gestão dos seus dados de hóspedes, reservas, quartos, serviços e funcionários, bem como o gerenciamento eficiente das reservas e geração de relatórios para melhorar as falhas no hotel. Posto isto, apostaram na criação de um sistema de base de dados para garantir uma maior eficiência, organização e segurança na gestão de informações, impedindo que problemas de gestão e atrasos ocorram, problemas estes que causaram muitos contratempos a Arlete no passado.

Viabilidade do projeto

Através da implementação deste projeto, pretendemos simplificar e aprimorar o registo e acesso aos dados do Hotel da Dona Arlete e, com uma base de dados eficiente, tornamos o processo mais fácil e eficaz.

Considerando agora a viabilidade do projeto, acreditamos que sua implementação é totalmente viável e fazível na prática, dado que um sistema de base de dados permite a realização de várias operações que promovem a gestão dos dados. Assim sendo, o seu uso traz inúmeras vantagens e benefícios, bem como uma melhor organização das informações e uma maior rapidez no acesso aos dados.

Desta forma, acreditamos que temos um sistema funcional que trará melhorias significativas ao Hotel.

1.2. Motivação e Objetivos

Tal como referido anteriormente, foi feita uma análise dos problemas que têm causado maiores atrasos e contratempos no hotel de Arlete, e Manuel e Susana chegaram à conclusão de que a base de dados seria útil para resolver os seguintes problemas:

1. ***Overbooking:*** Isto acontece quando uns números excessivos de reservas são feitos para um período específico e não há quartos suficientes disponíveis para atender a todas as reservas. Com um sistema de base de dados (SBD), é possível gerenciar melhor as reservas e evitar que o número de reservas seja superior ao número de quartos disponíveis.
2. **Falta de informações precisas*:*** Quando se recorria ao gerenciamento manual de informações, a falta de precisão dos dados é um problema recorrente, o que pode levar a erros e inconsistências. Com um SBD, as informações são armazenadas e gerenciadas eletronicamente, o que pode ajudar a garantir a integridade dos dados.

3. **Processos de *check-in* e *check-out* demorados:** Agora que Arlete tinha um grande número de hóspedes, o processo de *check-in e check-out* tornou-se lento e burocrático. Com um SBD, é possível acelerar esses processos, permitindo que os hóspedes sejam atendidos com mais eficiência.
4. **Dificuldade na gestão de reservas:** À medida que o número de reservas aumenta, a gestão manual das mesmas tornou-se uma tarefa demasiado complexa para Arlete. Com um SBD, é possível gerenciar as reservas de forma centralizada, facilitando o acompanhamento das mesmas.
5. **Falta de informações sobre ocupação dos quartos:** Com o sistema ultrapassado de Arlete, era muito difícil saber que quartos estavam ocupados e quais estavam disponíveis. Implementando este novo sistema, é possível obter informações precisas sobre a ocupação dos quartos, permitindo melhor gestão da disponibilidade e otimização do uso dos mesmos.

1.3 Análise da viabilidade do processo

Consideremos que o nosso projeto seja viável uma vez que este retrata um problema real e, com o acesso às aulas e com a devida disponibilização de material de trabalho, acreditamos ser possível implementar uma Base de Dados funcional e eficaz.

1.4 Recursos e Equipa de Trabalho

No que toca aos recursos, utilizaremos o material disponibilizados pelos docentes da UC. Com base nestes materiais, como grupo, iremos lê-los, revê-los e discuti-los de maneira a adotar a melhor abordagem e método para resolver o problema em causa.

1.5 Plano de Execução do Projeto

2. Levantamento e Análise de Requisitos

2.1. Métodos de levantamento e de análise de requisitos adotados

De modo a compreender o funcionamento e logística do gerenciamento de um Hotel, recorremos a diversos meios de informação de forma a tornar o nosso caso o mais próximo possível de uma situação e problema real.

Através desta pesquisa e reunião de ideias torna-se possível efetuar um levantamento de requisitos mais completo, o que posteriormente também nos cede uma maior facilidade na correção e deteção de erros.

O nosso método de levantamento de requisitos baseou-se então em:

- **Experiências:** Com base nas experiências dos elementos do grupo e de pessoas a si próximas foi-nos possível obter informações sobre o funcionamento de um hotel bem como alguns dos seus problemas de gerenciamento.

- **Análise de métodos de gestão:** Através de pesquisas sobre os métodos de gestão de informação dos hotéis, foi-nos possível compreender alguns dos seus problemas, como por exemplo, o overbooking, tal como abordamos anteriormente. É a partir da apreciação crítica deste e de outros problemas que se torna possível efetuar uma melhoria tanto no próprio gerenciamento como no tratamento de informação.

- **Análise de documentação:** Outro meio que utilizamos foi a procura de informação sobre este domínio na internet. Este método tornou-se útil no sentido em que acedemos a diversos sites de hotéis o que nos permitiu ter uma visão mais ampla daquilo em que nos devemos focar, investigar, aplicar e trabalhar.

2.2. Vista “Reservas”

• Requisitos de descrição

- RD1. Dona Arlete e funcionários: Tanto a dona Arlete e os seus filhos bem como alguns funcionários serão os utilizadores primários do sistema e, por isso, serão inseridos pelos administradores da base de dados. A informação a estes associados recai sobre o seu nome, email, local e horário de trabalho.

- RD2. Horário de funcionamento da receção: A dona Arlete e outros funcionários do hotel trabalham na sua administração das 9h-13h e das 14h-19h, intercalando entre si os dias de trabalho.

- RD3. Hóspedes: Os hóspedes representam no sistema os clientes do Hotel Éden. A informação a eles associada inclui o seu nome, a sua data de nascimento, o seu email, o seu contacto telefónico e a sua condição de saúde, isto é, se precisa de algum tipo de medicação ou necessidade especiais. Para além disso, a um cliente do Hotel está também associado o seu NIB, caso este pretende fatura com número de contribuinte, e também o feedback relativo à sua estadia.

- RD4. Reserva de quartos: No que toca às reservas de quartos, desde que haja disponibilidade e desde que os requisitos do Hotel sejam cumpridos, o cliente tem a liberdade de escolha face às suas necessidades e condições. Assim sendo, a informação de uma reserva de quarto inclui a data de check-in e de check-out, o tipo de quarto reservado, isto é, se pretende executivo, suíte, quarto normal ou se requer cama extra, estando esta última associada ao acréscimo de uma taxa. Para além disso é também necessário registar o número de reserva e de pessoas, se o hóspede pretende ou não pequeno-almoço incluído (inclui um acréscimo no preço da reserva), o preço total e o método de pagamento utilizado.

• Requisitos de manipulação

Inserção

- RM1. Inserção dos detalhes de um novo hóspede (por exemplo, o hóspede Miguel Azevedo, nascido a 24/10/1994, com email miguelazevedo@gmail.com, e contacto telefónico 951876345, que faz check-in no hotel no dia 12/03/2023 e o check-out no dia 20/03/2023. O seu NIF é 251266012 e o seu NIB é 2134 5671 65231789015 73).

- RM2. Inserção dos detalhes de uma reserva de quarto, sabendo-se a data e hora do check-in e check-out, o hóspede, o tipo de quarto e/ou se quer cama extra (o que inclui o acréscimo de taxa), o número de pessoas, se este pretende ou não pequeno-almoço incluído (inclui também um acréscimo no preço da reserva), o preço total e o método de pagamento utilizado.

- RM3. Inserção dos detalhes de uma nova reserva.

- RM4. Inserção dos detalhes do serviço de quarto que já se encontra incluído na reserva.

Atualização/Remoção

- RM5. Atualização/remoção de número de hóspedes associados a uma reserva.
- RM6. Atualização/remoção do pedido de uma cama extra.

- ## Requisitos de controlo

- RC1. Apenas os engenheiros/administradores da base de dados poderão adicionar/alterar/remover serviços relacionados com o hotel.

- RC2. Apenas a Dona Arlete e determinados funcionários podem adicionar/alterar/remover reservas.

- RC3. A dona Arlete e determinados funcionários podem aceder a toda a informação das reservas, bem como a informação sobre os seus hóspedes.

- RC4. Os hóspedes não podem aceder à informação de outros hóspedes.

2.3. Vista “Quarto”

- ## Requisitos de descrição

- RD1. Hóspede: A um hóspede pode ser associado um número ilimitado de quartos no Hotel, pelo que se encontra a si associado o número de quartos que o hóspede reservou bem como o respetivo número associado a estes.

- RD2. Estado do quarto: No que toca ao estado do quarto, o sistema fornece informação sobre se o quarto se encontra ou não limpo e se é ou não necessária algum tipo de manutenção.

- RD3. Disponibilidade do quarto: Relativamente á disponibilidade do quarto, o sistema fornece informações sobre a disponibilidade do quarto para as datas inseridas no processo da reserva.

- RD4. Características: O sistema guarda informações sobre as características do quarto, como por exemplo o andar em que este se encontra e a vista.

- RD5. Tipo de quarto: A cada quarto está associado o seu tipo, pelo que o sistema guarda informações sobre o tipo de quartos existentes, estando dentro das possibilidades o quarto executivo, a suíte e o quarto normal.

- **Requisitos de manipulação**

Inserção

- RM1. Inserção de novos detalhes do quarto de um hóspede (por exemplo, adicionar uma cama extra no quarto, pedir fornecimento de mais toalhas ou personalizações específicas ao quarto).

Atualização/Remoção

- RM4. Atualização da disponibilidade de um quarto.
- RM5. Atualização do pedido de uma cama extra.

- **Requisitos de controlo**

- RC1. O quarto de um hóspede deve ser inacessível aos outros hóspedes.

2.4. Vista “Hóspedes”

- **Requisitos de descrição**

- RD1. Cliente: Os clientes do Hotel são os hóspedes. Os hóspedes podem efetuar as suas reservas como bem o entendem e podem também reservar um número ilimitado de quartos no Hotel.

- RD3. Método de pagamento: São guardadas as informações sobre a forma de pagamento do hóspede e informações de cartão de crédito tanto para garantir a reserva como para cobrar outros serviços extras. Estas informações incluem o número de cartão de crédito e o NIB do hóspede (sinceramente não sei se é só isto).

- ### Requisitos de manipulação

Inserção

- RM1. Inserção de um novo método de pagamento.
- RM2. Inserção ou comunicação dos dados de uma reserva, seja através do site do hotel, telefone ou pessoalmente.

Atualização/Remoção

- RM5. Atualização dos dados pessoais de um hóspede.
- RM6. Atualização do método de pagamento.

Requisitos de controlo

- RC1. A reserva de um cliente deve ser inacessível aos outros clientes.

2.5. Vista “Funcionários”

- ### Requisitos de Descrição

- RD1. Identificação: Os funcionários têm de estar identificados com o seu número e o seu nome. Para além disso o sistema guarda informações como o seu local de trabalho, o email, o contacto telefónico, a data de nascimento, a fotografia e o cargo que exerce. São também armazenadas outras informações pessoais que são sobretudo necessárias para a entidade

empregadora, como por exemplo, o currículo, uma cópia do cartão de cidadão, a sua morada de residência, o salário, o NIF e o NIB.

- ## Requisitos de Manipulação

- RM1. Os funcionários podem manipular uma base de dados introduzindo novas informações de clientes, reservas ou transações financeiras.

- RM2. Estes também podem atualizar os dados existentes sobre os quartos, reservas como o tipo de quarto, o tempo em que vai ficar no quarto e preços.

- ## Requisitos de Controlo

- RC1. Apenas os engenheiros e administradores ou funcionários autorizados podem efetuar alterações na base de dados.

2.6 Método de levantamento de requisitos

De modo a compreender o funcionamento e logística do gerenciamento de um Hotel, recorremos a diversos meios de informação de forma a tornar o nosso caso o mais próximo possível de uma situação e problema real.

Através desta pesquisa e reunião de ideias torna-se possível efetuar um levantamento de requisitos mais completo, o que posteriormente também nos cede uma maior facilidade na correção e deteção de erros.

O nosso método de levantamento de requisitos baseou-se então em:

- **Experiências:** Com base nas experiências dos elementos do grupo e de pessoas a si próximas foi-nos possível obter informações sobre o funcionamento de um hotel bem como alguns dos seus problemas de gerenciamento.

- **Análise de métodos de gestão:** Através de pesquisas sobre os métodos de gestão de informação dos hotéis, foi-nos possível compreender alguns dos seus problemas, como por exemplo, o *overbooking,* tal como abordamos anteriormente. É a partir da apreciação crítica deste e de outros problemas que se torna possível efetuar uma melhoria tanto no próprio gerenciamento como no tratamento de informação.

- **Análise de documentação:** Outro meio que utilizamos foi a procura de informação sobre este domínio na *internet*. Este método tornou-se útil no sentido em que acedemos a diversos *sites* de hotéis o que nos permitiu ter uma visão mais ampla daquilo em que nos devemos focar, investigar, aplicar e trabalhar.

2.7 Análise e validação geral dos requisitos

De maneira a validar os requisitos, como grupo, optámos por individualmente reler e rever tudo aquilo que tínhamos definido. Após essa revisão reunimos e discutimos pontos a alterar, remover ou melhorar.

Uma das nossas dúvidas iniciais resistia sobre a relevância de determinados requisitos, não sabíamos quão específicos deveríamos ser. No entanto, após analisarmos e debatermos os pontos cruciais, chegamos a um consenso, definindo assim os requisitos finais.

3. Modelação Conceptual

3.1 Apresentação do método de modelação utilizado

De modo a construir um modelo de estrutura para a nossa Base de Dados, decidimos recorrer ao Diagrama Entidade Relacionamento (Diagrama ER).

Este tipo de diagrama é utilizado para criar relacionamentos entre as entidades de um problema, sendo que, consequente, é feita a identificação dos seus respetivos atributos.

A aplicação deste *design* torna-se bastante útil no que toca à construção de uma Base de Dados uma vez que é de fácil compreensão e simples. Para além disso, este modelo permite uma representação visual, o que promove uma compreensão rápida e eficaz da relação estabelecida entre diferentes entidades.

3.2 Identificação e caracterização das entidades

Tendo em conta o funcionamento e gerenciamento de um Hotel decidimos considerar seis entidades, sendo elas: Serviço de Receção, Reserva, Hóspede, Quarto, Serviço de Limpeza e Funcionários.

- **Serviço de Receção:** Esta entidade representa os funcionários que trabalham com o processo das reservas e os seus derivados. Os atributos que lhe dizem respeito são os seus contactos, o nacional e o estrangeiro, o seu email e o seu horário de funcionamento.

- **Reserva:** A entidade “Reserva” representa o conjunto de informações relativas a uma reserva de quarto efetuada por um hóspede. Nestas informações constam atributos como o número da reserva, o número de pessoas a si associadas, o preço, as datas de *check-in* e *check-out*, o tipo de quarto, o método de pagamento e o acesso a serviços extra como o pequeno-almoço ou cama extra.

- **Hóspede:** Por sua vez, a entidade “Hóspede” representa o conjunto de informações relativas a um hóspede do Hotel. Os seus atributos recaem sobre o seu nome, o seu contacto telefónico, o email, a data de nascimento, de modo a calcular a sua idade, as suas condições de saúde, o seu NIB e o seu NIF e o *feedback* da estadia.

- **Quarto:** Nesta entidade são guardadas informações sobre os quartos do Hotel. Os seus atributos correspondem ao número do quarto, às

suas características, como a vista e o andar em que se encontra, acessibilidade, disponibilidade e o seu estado.

- **Funcionários:** Nesta entidade são guardadas informações sobre os funcionários do Hotel. Os atributos a si associados consistem no seu número de funcionário, nome, email, contacto telefónico, fotografia, cargo bem como outras informações pessoais necessárias para a entidade empregadora.

3.3 Identificação e caracterização dos relacionamentos

No nosso modelo recorremos a relacionamentos binários como a relacionamentos ternários e iremos de seguida analisar cada um deles individualmente.

- **Relacionamento "Serviço de Receção cria Reserva"**

  Decidimos criar este relacionamento de modo a facilitar a criação de reservas ou até mesmo a verificação de outras já existentes. Nos atributos da entidade "Reserva" é armazenada informação importante que torna o seu acesso mais rápido e mais fácil através de, por exemplo, atributos chave como o número da reserva. Desta forma não é necessário percorrer todas as reservas e podemos aceder a variadas informações como as datas e número de pessoas associados à reserva.

  Relativamente à sua **multiplicidade** temos que:

  Serviço de Receção (1,1) – Reserva (0, n)

  Esta multiplicidade deve-se ao facto de que o serviço de receção pode criar várias reservas, no entanto uma reserva está associada a um serviço de receção.
  Porém temos também de ter em conta que o serviço de receção nem sempre realiza reservas (0), enquanto uma reserva está sempre associada a um serviço de receção (1).

- **Relacionamento "Reserva associada a Quarto"**

  A criação deste relacionamento permite-nos estabelecer uma ligação importante entre a reserva e o quarto. Através dele é possível verificar detalhes relevantes como por exemplo a sua disponibilidade e características.

  Relativamente à sua **multiplicidade** temos que:

Reserva (0, n) – Quarto (1, n)

A multiplicidade deste relacionamento está associada ao facto de que uma reserva pode estar associada a vários quartos, assim como um quarto pode estar associado a várias reservas, por exemplo, no caso de se efetuarem reservas para um mesmo quarto em datas diferentes.
É também importante ter em consideração que uma reserva tem de estar sempre associada a um quarto (1), por sua vez um quarto não tem de estar associado a uma reserva (0) uma vez que podem existir quartos não ocupados.

- **Relacionamento "Reserva associada a Hóspede"**

  Este relacionamento, tal como o anterior, estabelece uma importante relação entre a reserva e uma outra entidade, os "Hóspedes". Através desta ligação é possível aceder a informações sobre a reserva de um determinado hóspede ou vice-versa.

  Relativamente à sua **multiplicidade** temos que:

  Reserva (1, n) - Hóspede (1, n)

  A multiplicidade deste relacionamento tem a ver com o facto de que uma reserva está associada a um hóspede, no entanto um hóspede pode estar associado a várias reservas.
  Porém é importante ter em atenção que uma reserva tem de estar necessariamente associada a um hóspede (1), no entanto um hóspede não tem de estar necessariamente associado a uma reserva (0) como é o caso de uma criança.

- **Relacionamento "Serviço de Receção ocupado por Funcionários"**

  Este relacionamento permite obter conhecimentos sobre os funcionários que ocupam o serviço de receção. Algumas destas informações são, por exemplo, o seu nome e o número de funcionário.

  Relativamente à sua **multiplicidade** temos que:

  Serviço de Receção (1, n) – Funcionários (1, n)

  A multiplicidade do relacionamento tem a ver com o facto de que o serviço de receção pode ser realizado por vários funcionários e vários funcionários podem realizar esse serviço.

É importante referir que tanto o serviço de receção tem de ser realizado por um funcionário (1), como pelo menos um funcionário tem de realizar o serviço de receção (1).

- **Relacionamento "Hóspede <u>associado</u> a Quarto"**

  Este relacionamento permite obter informações sobre os hóspedes que estão associados a um quarto. Algumas destas informações são, por exemplo, o seu nome e o contacto telefónico.

  Relativamente à sua **multiplicidade** temos que:

  Hóspede (0, n) – Quarto (1, n)

  A multiplicidade do relacionamento tem a ver com o facto de que o hóspede pode estar associado a vários quartos e vários quartos podem estar associados a um hóspede.
  É importante mencionar que o hóspede tem de estar necessariamente associado a um quarto (1), no entanto um quarto não tem de estar necessariamente associado a um hóspede (0).

3.4 Identificação e caracterização da associação dos atributos com as entidades e relacionamentos

ENTIDADE: FUNCIONÁRIOS

Atributos	Tipos de dados	Nulo	Composto	Multivalorado	Derivado	Candidato
Nr_funcionario	INT	NÃO	NÃO	NÃO	NÃO	SIM
Email	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	NÃO
Nome	VARCHAR(100)	NÃO	NÃO	NÃO	NÃO	NÃO
Contacto_telefonico	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	SIM
NIB	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	NÃO
NIF	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	SIM
Salario	DOUBLE	NÃO	NÃO	NÃO	NÃO	NÃO
Nr_cartao_cidadao	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	SIM
Rua	VARCHAR(100)	NÃO	NÃO	NÃO	NÃO	NÃO
Localidade	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	NÃO
Código_postal	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	NÃO

ENTIDADE: RESERVA

Atributos	Tipos de dados	Nulo	Composto	Multivalorado	Derivado	Candidato
Nr_reserva	INT	NÃO	NÃO	NÃO	NÃO	SIM
Nr_pessoas	INT	NÃO	NÃO	NÃO	NÃO	NÃO
Nr_Quartos	INT	NÃO	NÃO	NÃO	NÃO	NÃO
Check_in	DATETIME	NÃO	NÃO	NÃO	NÃO	NÃO
Check_out	DATETIME	NÃO	NÃO	NÃO	NÃO	NÃO
Metodo_pagamento	VARCHAR(45)	NÃO	SIM	NÃO	NÃO	NÃO
Preco_quarto	INT	NÃO	NÃO	NÃO	NÃO	NÃO
Preco_total	INT	NÃO	NÃO	NÃO	SIM	NÃO

ENTIDADE: SERVIÇO DE RECEÇÃO

Atributos	Tipos de dados	Nulo	Composto	Multivalorado	Derivado	Candidato
Nr_funcionario	INT	NÃO	NÃO	NÃO	NÃO	SIM
Email_geral	VARCHAR(50)	NÃO	NÃO	NÃO	NÃO	NÃO
Contacto_telefonico	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	NÃO
Inicio_turno	TIME	NÃO	NÃO	NÃO	NÃO	NÃO
Fim_turno	TIME	NÃO	NÃO	NÃO	NÃO	NÃO

ENTIDADE: HÓSPEDE

Atributos	Tipos de dados	Nulo	Composto	Multivalorado	Derivado	Candidato
Id_hospede	INT	NÃO	NÃO	NÃO	NÃO	SIM
Nome	VARCHAR(100)	NÃO	NÃO	NÃO	NÃO	NÃO
Email	VARCHAR(100)	NÃO	NÃO	NÃO	NÃO	NÃO
NIF	VARCHAR(100)	NÃO	NÃO	NÃO	NÃO	SIM
NIB	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	NÃO
Contacto_telefonico	VARCHAR(45)	NÃO	NÃO	NÃO	NÃO	NÃO
Data_nascimento	DATETIME	NÃO	NÃO	NÃO	SIM	NÃO

ENTIDADE: QUARTO

Atributos	Tipos de dados	Nulo	Composto	Multivalorado	Derivado	Candidato
Nr_quarto	INT	NÃO	NÃO	NÃO	NÃO	SIM
Cama_extra	TINYINT	NÃO	NÃO	NÃO	NÃO	NÃO
Disponibilidade	TINYINT	NÃO	NÃO	NÃO	NÃO	NÃO
Tipo_quarto	VARCHAR(45)	NÃO	SIM	NÃO	NÃO	NÃO

3.5 Apresentação e explicação do diagrama ER produzido

Tendo em conta a prévia caracterização das entidades e explicação de todos os relacionamentos presentes no nosso esquema, achamos agora necessário apresentar uma imagem geral do nosso modelo conceptual.

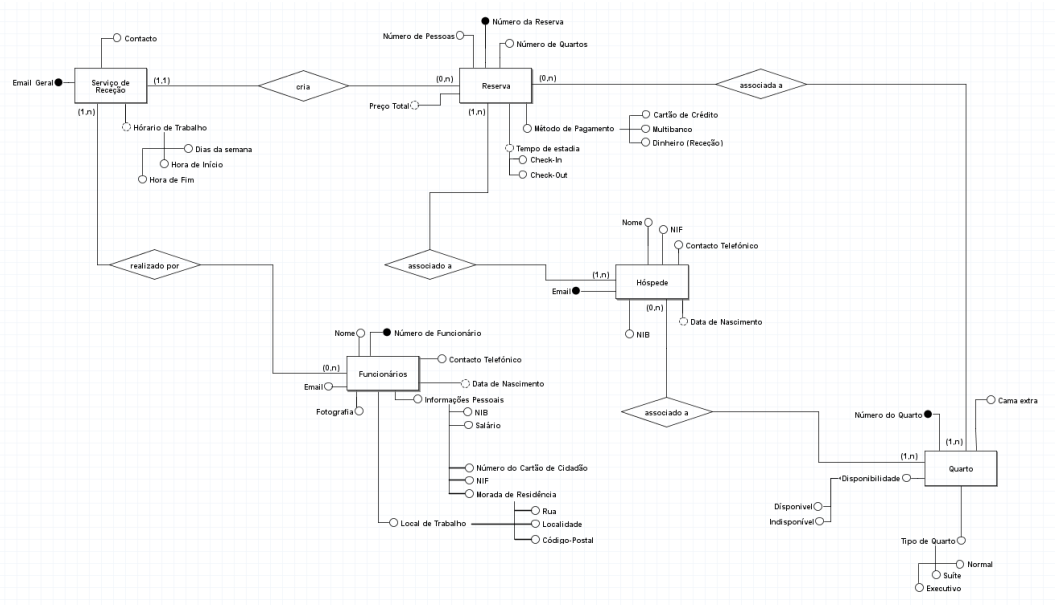


Figura 1 - Modelo Conceptual

Tendo agora em consideração esta modelagem e visando que este apresenta uma análise de viabilidade favorável, torna-se mais fácil a construção e implementação no nosso no nosso modelo, o modelo lógico.

3.6 Validação do Modelo Conceptual

De modo a comprovar a viabilidade do nosso modelo, podemos começar por analisar que os requisitos estão associados às respetivas entidades. Podemos comprovar e demonstrar, verificando a imagem que disponibilizámos no ponto anterior, que todos os atributos que representamos nas nossas entidades justificam todos os requisitos que enumerámos previamente.

Sendo assim, tendo os requisitos validados e, agora, tendo a validação do modelo conceptual, podemos então passar à próxima fase do projeto.

4 Modelação Lógica

4.1 Construção e validação do modelo de dados lógico

Tal como referimos anteriormente, a construção do Modelo Lógico foi feita com base no Modelo Conceptual projetado.

O que difere do conceptual para o lógico é que neste as entidades são representadas em tabelas que acomodam os seus atributos. Os relacionamentos também são definidos de acordo com o definido na modelação conceptual.

Os atributos chave das entidades passam aqui a ter um novo nome, designando-se por *Primary Key*. Esta chave é também responsável por estabelecer relação entre tabelas, pelo que, neste caso, passa como uma *Foreign Key* para a tabela com a qual se pretende relacionar.

Assim sendo, vamos de seguida apresentar a conversão para o modelo lógico, começando por representar as tabelas que surgiram.

Tabelas relativas às entidades:

Serviço de Receção

Primary Key: Email Geral VARCHAR(45);

Atributos: Contacto VARCHAR(45), Inicio e Fim de Turno DATETIME;

Foreign Key: Não apresenta chave estrangeira;

Reserva

Primary Key: Número da Reserva INT;

Atributos: Número de Pessoas INT, Número de Quartos INT, Check-In e Check-Out DATETIME, Método de Pagamento VARCHAR(45), Preço INT;

Foreign Key: Email Geral VARCHAR(45);

Quarto

***Primary Key*:** Número do Quarto INT;

Atributos: Cama Extra TINYINT, Disponibilidade TINYINT, Tipo de Quarto VARCHAR(45);

Foreign Key: Não apresenta chave estrangeira;

Hóspede

***Primary Key*:** ID do Hóspede INT;

Atributos: Email VARCHAR(45), NIB VARCHAR(45), NIF VARCHAR(45), Contacto VARCHAR(45), Nome VARCHAR(45), Data Nacimento DATETIME;

Foreign Key: Não apresenta chave estrangeira;

Funcionários

***Primary Key*:** Número de Funcionário INT;

Atributos: Nome VARCHAR(45), NIB VARCHAR(45), NIF VARCHAR(45), Contacto VARCHAR(45), Email VARCHAR(45), Salário DOUBLE, Número Cartão de Cidadão VARCHAR(45), Rua VARCHAR(45), Localidade VARCHAR(45), Código Postal VARCHAR(45);

Foreign Key: Não apresenta chave estrangeira;

Tabelas relativas às relações:

Serviço_Funcionários: Apresenta duas *Foreign Keys*, sendo cada uma delas as *Primary Keys* das tabelas Serviço de Receção e Funcionários.

Quarto_Reserva: Apresenta duas *Foreign Keys*, sendo cada uma delas as *Primary Keys* das tabelas Quarto e Reserva.

Quarto_Hóspede: Apresenta duas *Foreign Keys*, sendo cada uma delas as *Primary Keys* das tabelas Quarto e Hóspede.

Hóspede_Reserva: Apresenta duas *Foreign Keys*, sendo cada uma delas as *Primary Keys* das tabelas Hóspede e Reserva.

4.2 Apresentação e explicação do modelo lógico produzido

Neste ponto pretendemos representar visualmente o Modelo Lógico que implementamos.

Esta representação permite ter uma visão mais ampla e completa de como se estabelecem as relações e como ficam representadas cada uma das entidades neste esquema.

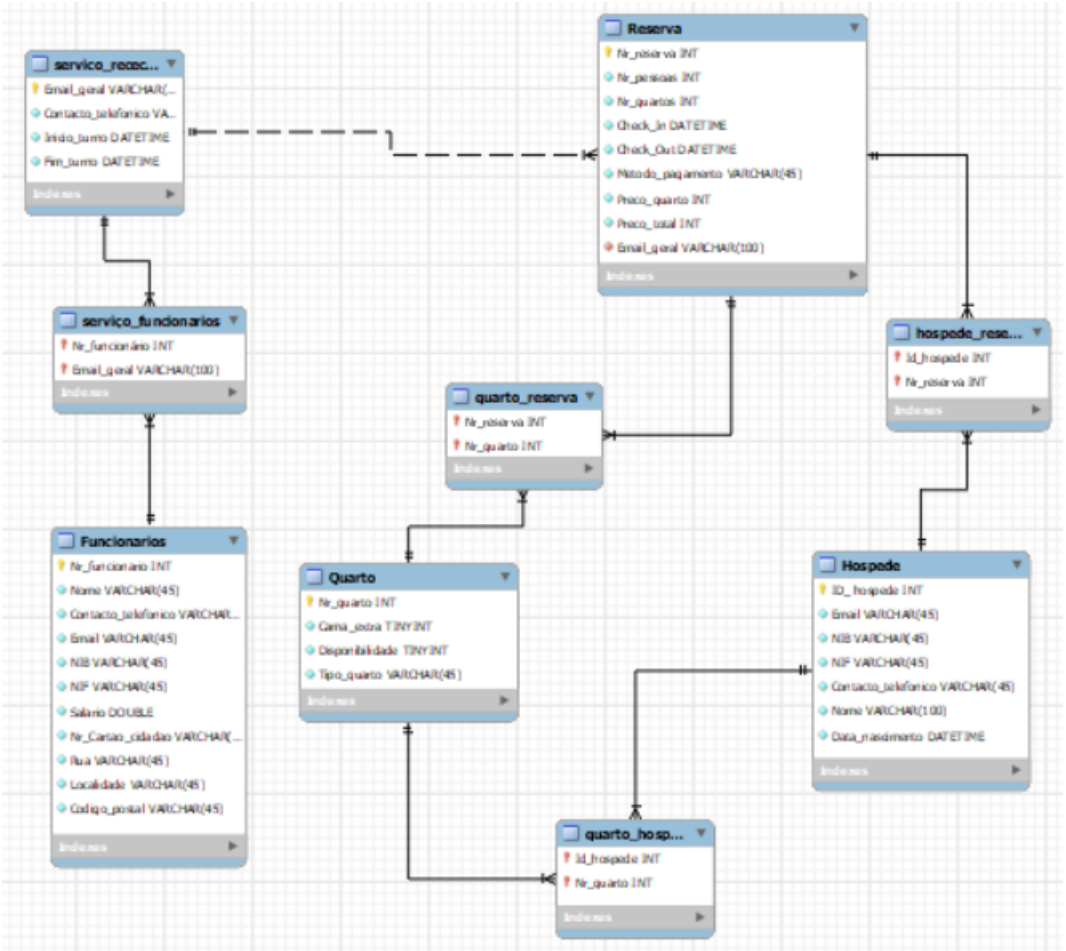


Figura 2 - Modelo Lógico

4.3 Normalização de Dados

A normalização é um processo importante para evitar a redundância de dados e melhorar o desempenho de um modelo, garantindo assim consistência e, evitando anomalias durante operações de inclusão, exclusão e alteração de registos.

Para avaliar a normalização de um modelo, devemos verificar se ele atende às três formas normais.

- **1ª Forma Normal (1FN):** Os atributos devem ser atômicos, sem repetições ou atributos multivalorados. O modelo cumpre essa condição.

- **2ª Forma Normal (2FN):** Os atributos não-chave devem depender completamente da chave primária. O modelo atende a essa regra.

- **3ª Forma Normal (3FN):** Não deve haver dependência entre atributos. O modelo não possui atributos dependentes, cumprindo assim a 3FN.

Concluímos assim que o modelo é válido em termos de normalização, pois satisfaz as três formas normais.

4.4 Validação do modelo com interrogações do utilizador

De modo a realizar a validação do modelo lógico, aquilo que fizemos foi aplicar as técnicas de normalização.
Através desse processo foi-nos possível confirmar que o esboço do modelo era válido, ou seja, atendia aos critérios de integridade e eficiência necessários para o bom funcionamento da base de dados.

5 Implementação Física

5.1 Tradução do esquema lógico para SQL

A tradução do esquema lógico para o *MySQL* foi feita manualmente.
Para além disso adicionámos comandos para criar um *schema* e usá-lo por defeito. Ao correr este ficheiro fomos capazes de criar a nossa base de dados em *SQL*.

5.2 *Tradução das interrogações do utilizador para SQL*

➔ Atualização do número de hóspedes associados a uma reserva.

```
DELIMITER //
CREATE PROCEDURE atualizar_num_hospedes_reserva(IN n_reserva INT, IN novo_numero_hospedes INT)
BEGIN
    UPDATE `Hotel`.`reserva`
    SET nr_pessoas = novo_numero_hospedes
    WHERE nr_reserva = n_reserva;
END //
DELIMITER ;
```

Figura 3 – Código SQL que atualiza número de hóspedes

➔ Atualização das datas de Check-In e de Check-Out.

```
-- Função para atualizar o número de hóspedes associados a uma reserva
DELIMITER //
CREATE PROCEDURE atualizar_datas_checkin_checkout(IN n_reserva INT, IN novo_checkin DATE, IN novo_checkout DATE)
BEGIN
    UPDATE `Hotel`.`reserva`
    SET check_in = novo_checkin, check_out = novo_checkout
    WHERE nr_reserva = n_reserva;
END //
DELIMITER ;
```

Figura 4 - Código SQL que atualiza Check-In e Check-Out

➔ Atualizar o pedido de uma cama extra num quarto.

```
-- Função para atualizar o pedido de uma cama extra em um quarto
DELIMITER //
CREATE PROCEDURE atualizar_pedido_cama_extra(IN n_reserva INT, IN novo_pedido_cama_extra TINYINT)
BEGIN
    UPDATE `Hotel`.`reserva`
    SET cama_extra = novo_pedido_cama_extra
    WHERE nr_reserva = n_reserva;
END //
DELIMITER ;
```

Figura 5 - Código SQL que atualiza pedido de cama extra

➔ Atualização do tipo de quarto selecionado para reserva.

```
-- Função para atualizar o tipo de quarto
DELIMITER //
CREATE PROCEDURE atualizar_tipo_quarto(IN n_quarto INT, IN novo_tipo_quarto VARCHAR(45))
BEGIN
    UPDATE `Hotel`.`quarto`
    SET tipo_quarto = novo_tipo_quarto
    WHERE nr_quarto = n_quarto;
END //
DELIMITER ;
```

Figura 6 - Código SQL que atualiza o tipo de quarto

5.3 Definição e caracterização das vistas de utilização em SQL

A criação de vistas num projeto de base de dados usando *MySQL* oferece uma série de benefícios e funcionalidades importantes. Uma vista é uma tabela virtual que é criada com base nos resultados de uma consulta específica. Em vez de armazenar os dados fisicamente, a vista armazena a definição da consulta usada para criar a tabela virtual.

➔ Criação de vista que mostra os detalhes de um hóspede e do seu quarto.

```
--
-- vista que mostra os detalhes de um hóspede e do seu quarto.
CREATE VIEW vw_quarto_hospede AS
SELECT q.nr_quarto, q.cama_extra, q.disponibilidade, q.tipo_quarto, h.id_hospede, h.nome,
 h.email, h.NIF, h.NIB, h.contacto_telefonico, h.data_nascimento
FROM quarto q
INNER JOIN quarto_hospede qh ON q.nr_quarto = qh.nr_quarto
INNER JOIN hospede h ON qh.id_hospede = h.id_hospede;

SELECT * FROM vw_quarto_hospede;
```

Figura 7 – Código SQL que implementa vista

➔ Criação de uma vista que mostra os detalhes dos funcionários que trabalham na receção.

```
--
-- vista que mostra os detalhes dos funcionários que trabalham na receção.
CREATE VIEW vw_funcionario_servico AS
SELECT f.nr_funcionario, f.email, f.nome, f.contacto_telefonico, f.NIB, f.NIF, f.salario,
 f.nr_cartao_cidadao, f.rua, f.localidade, f.codigo_postal,
sr.email_geral, sr.inicio_turno, sr.fim_turno
FROM funcionario f
INNER JOIN servico_rececao sr ON f.nr_funcionario = sr.nr_funcionario;

SELECT * FROM vw_funcionario_servico;
```

Figura 8 – Código SQL que implementa vista

➔ Criação de vista que mostra os detalhes dos hóspedes e as suas respetivas reservas.

```
--
-- vista que mostra os detalhes dos hóspedes e as suas respetivas reservas.
CREATE VIEW vw_hospede_reserva AS
SELECT h.id_hospede, h.nome, h.email, h.NIF, h.NIB, h.contacto_telefonico, h.data_nascimento,
       r.nr_reserva, r.nr_pessoas, r.nr_quartos, r.check_in, r.check_out, r.metodo_pagamento,
       r.preco_quarto, r.preco_total
FROM hospede h
INNER JOIN hospede_reserva hr ON h.id_hospede = hr.nr_hospede
INNER JOIN reserva r ON hr.nr_reserva = r.nr_reserva;

SELECT * FROM vw_hospede_reserva;
```

Figura 9 - Código SQL que implenta vista

5.4 Cálculo do espaço da bases de dados (inicial e taxa de crescimento anual)

Sabendo o tamanho ocupado pelos *datatypes* associados a cada atributo é-nos possível calcular o espaço ocupado em disco.

Deste modo, é possível obter uma estimativa do espaço em disco da nossa base de dados bem como a sua taxa de crescimento anual.

Serviço de Receção

Tabela 1 - Cálculo de bytes associados à entidade Serviço de Receção

Atributos	Tipo	Tamanho
Email Geral	VARCHAR(45)	**47**
Contacto	VARCHAR(45)	**47**
Início de Turno	DATETIME	8
Fim de Turno	DATETIME	8
Total	**----------**	**110**

Reserva

Tabela 2 - Cálculo de bytes associados à entidade Reserva

Atributos	Tipo	Tamanho
Número de Reserva	INT	4
Número de Quartos	INT	4
Número de Pessoas	INT	4
Check-In	DATETIME	8
Check-Out	DATETIME	8
Método de Pagamento	VARCHAR(45)	47
Preço	INT	4
Email Geral	VARCHAR(100)	**102**
Total	**----------**	

Hóspede

Tabela 3 - Cálculo de bytes associados à entidade Hóspede

Atributos	Tipo	Tamanho
ID do Hóspede	INT	4
Email	VARCHAR(45)	47
NIB	VARCHAR(45)	47
NIF	VARCHAR(45)	47
Contacto Telefónico	VARCHAR(45)	47
Nome	VARCHAR(100)	102

Data de Nascimento	DATETIME	8
Total	----------	294

Quarto

Tabela 4 - Cálculo de bytes associados à entidade Quarto

Atributos	**Tipo**	**Tamanho**
Número do Quarto	INT	4
Cama Extra	TINYINT	
Disponibilidade	TINYINT	
Tipo de Quarto	VARCHAR(45)	
Total	**----------**	

Funcionários

Tabela 5 - Cálculo de bytes associados à entidade Funcionários

Atributos	**Tipo**	**Tamanho**
Número de Funcionário	INT	4
Nome	VARCHAR(45)	47
Contacto Telefónico	VARCHAR(45)	47
NIB	VARCHAR(45)	47
NIF	VARCHAR(45)	47
Salário	DOUBLE	16
Número Cartão Cidadão	VARCHAR(45)	47
Rua	VARCHAR(45)	47
Localidade	VARCHAR(45)	47
Código-Postal	VARCHAR(45)	47
Email	VARCHAR(45)	47
Total	**----------**	**443**

Hóspede_Reserva

Tabela 6 - Cálculo de bytes associados ao relacionamento Hóspede_Reserva

Atributos	Tipo	Tamanho
ID do Hóspede	INT	4
Número da Reserva	INT	4
Total	**----------**	8

Quarto_Hóspede

Tabela 7 - Cálculo de bytes associados ao relacionamento Quarto_Hóspede

Atributos	Tipo	Tamanho
ID do Hóspede	INT	4
Número do Quarto	INT	4
Total	**----------**	8

Quarto_Reserva

Tabela 8 - Cálculo de bytes associados ao relacionamento Quarto_Reserva

Atributos	Tipo	Tamanho
Número da Reserva	INT	4
Número do Quarto	INT	4
Total	**----------**	8

Serviço_Funcionários

Tabela 9 - Cálculo de bytes associados ao relacionamento Serviço_Funcionários

Atributos	Tipo	Tamanho
Número de Funcionário	INT	4
Email Geral	VARCHAR(100)	102
Total	**----------**	**106**

5.5 Procedimentos implementados

Tendo em conta o sistema implementado, verifica-se que este respeita todos os requisitos estabelecidos.

Em termos de espaço, este sistema ocupa um espaço em disco satisfatório e efetua uma gerência eficiente dos dados.

Já a implementação física do sistema, que é expressa em linguagem *SQL*, reflete as ideias representadas no modelo lógico, permitindo que os dados sejam armazenados, manipulados e consultados de maneira eficaz e precisa.

Por fim, a validação da implementação confirma que esta foi realizada corretamente e está de acordo com as especificações definidas.

5.6 Plano de segurança e recuperação de dados

De modo a recuperar os dados efetuámos um backup.

Para isso executamos os seguintes comandos:

cd "C:\\Program Files\\MySQL\\MySQL Server 8.0\\bin"

Create a dump of your MySQL database. mysqldump.exe --user=root--password=*--host=localhost --port=3306 --result-**

6 Implementação do Sistema de Recolha de Dados

6.1 Apresentação e modelo do sistema

Chegámos a uma parte da base de dados em que precisamos de começar a povoar todas as tabelas. E para este processo criamos um programa na linguagem de programação *Python* que está conectado às tabelas criadas nos nossos ficheiros em *MySQL*. E para fazer a povoação temos a ajuda de um ficheiro de texto "*csv*" para cada tabela que contêm as informações sobre todas as tabelas e que através do programa em *Python* vai fazer a separação dos diferentes dados para as respetivas tabelas e consequentemente povoá-las.

6.2 Implementação do sistema de recolha

Temos os dois únicos imports necessários para esta execução em que o "*import csv"* é o import que é preciso para que o programa *python* consigo ler de um ficheiro "*csv*" para que possa povoar as tabelas. E de seguida temos o "*import mysql.connector*" que é usado para estabelecer contacto com a base de dados, em análise, para as tabelas serem povoadas no *MySQL*.

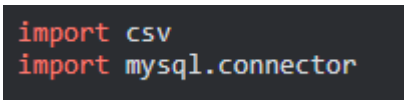


Figura 10 - Imports

Estas duas linhas são as que vão dar uso ao *"import mysql.connector*" que irão estabelecer a conexão com a base de dados. Para esta conexão ser estabelecida precisamos de fornecer o host, o user, a password e o nome da base de dados.

```
db = mysql.connector.connect(host='localhost', user='root', password='*********', database='Hotel')
cursor = db.cursor()
```

Figura 11 - Conexão com a Base de Dados

Esta função que é a que vai dar o nosso objetivo vai receber como argumentos o nome da tabela e o ficheiro texto "*csv*" e esta função irá ler o ficheiro "*csv*" e para cada linha do arquivo fará a execução do comando *MySQL "INSERT"* na tabela que esteja a ser povoado no momento. Para obtermos uma confirmação que tudo correu bem decidimos meter um print no final do programa que diz "Dados inseridos na tabela {nome_tabela} com sucesso!" para saber se tudo correu bem com cada tabela, caso contrário a linguagem *Python* daria algum erro ou *warning*.

```
def povoar_tabela(nome_tabela, nome_arquivo_csv):
    # Caminho para o arquivo CSV
    csv_file = nome_arquivo_csv

    # Ler o arquivo CSV e inserir os dados na tabela correspondente
    with open(csv_file, 'r') as file:
        csv_data = csv.reader(file)
        next(csv_data)  # Pular o cabeçalho do CSV

        for row in csv_data:
            # Escapar os valores antes de inserir na consulta
            values = [f"'{value}'" if isinstance(value, str) else str(value) for value in row]

            # Inserir os dados na tabela correspondente
            query = f"INSERT INTO {nome_tabela} VALUES ({', '.join(values)})"
            cursor.execute(query)

    print(f'Dados inseridos na tabela {nome_tabela} com sucesso!')
```

Figura 12 - Povoação das tabelas

Estas linhas de código são simplesmente chamadas da função para povoar todas as tabelas que sejam precisas inserir os valores.

```
# Povoar tabela "funcionario"
povoar_tabela('funcionario', 'funcionarios.csv')

# Povoar tabela "servico_rececao"
povoar_tabela('servico_rececao', 'servico_rececao.csv')

# Povoar tabela "servico_funcionario"
povoar_tabela('servico_funcionario', 'servico_funcionario.csv')

# Povoar tabela "hospede"
povoar_tabela('hospede', 'hospede.csv')

# Povoar tabela "reserva"
povoar_tabela('reserva', 'reserva.csv')

# Povoar tabela "quarto"
povoar_tabela('quarto', 'quarto.csv')

# Povoar tabela "hospede_reserva"
povoar_tabela('hospede_reserva', 'hospede_reserva.csv')

# Povoar tabela "quarto_reserva"
povoar_tabela('quarto_reserva', 'quarto_reserva.csv')

# Povoar tabela "quarto_hospede"
povoar_tabela('quarto_hospede', 'quarto_hospede.csv')
```

Figura 13 - Povoação de cada uma das tabelas

Estas duas linhas confirmam a transação dos dados e consequentemente fecha a conexão com a base de dados.

```
db.commit()
db.close()
```

Figura 14 - Transação de dados e fecha conexão

Esta linha é só uma confirmação para saber se todo o seu programa correu como planeado.

```
print('Povoamento de tabelas concluído com sucesso!')
```

Figura 15 - Confirmação de povoamento bem sucedido

6.3 Funcionamento do sistema

Ao executar este programa segundo as tabelas da nossa base de dados obtemos um *input* como a imagem abaixo.

```
PS C:\Users\pedro\OneDrive\Ambiente de Trabalho\Base de Dados> python funcio.py
Dados inseridos na tabela funcionario com sucesso!
Dados inseridos na tabela servico_rececao com sucesso!
Dados inseridos na tabela servico_funcionario com sucesso!
Dados inseridos na tabela hospede com sucesso!
Dados inseridos na tabela reserva com sucesso!
Dados inseridos na tabela quarto com sucesso!
Dados inseridos na tabela hospede_reserva com sucesso!
Dados inseridos na tabela quarto_reserva com sucesso!
Dados inseridos na tabela quarto_hospede com sucesso!
Povoamento de tabelas concluído com sucesso!
```

Figura 16 - Sucesso na povoação

Através disto as tabelas foram povoadas com sucesso.

7 Implementação do Sistema de Painéis de Análise

7.1 Definição e caracterização da vista de dados para análise

A definição e caracterização da vista de dados para análise é uma de suas principais características e vantagens do *PowerBI*. Deste modo, recorremos a esta para nos auxiliar na construção deste tópico.

Esta plataforma oferece variados recursos que permitem aos seus usuários criar estruturas personalizadas para análise de dados.

A caracterização da vista de dados no *PowerBI* refere-se à personalização e formatação dos dados tendo em consideração as necessidades e preferências dos usuários.

7.2 Povoamento das estruturas de dados para análise

O povoamento das estruturas de dados para análise no *PowerBI* envolve o processo de importar e carregar os dados relevantes para lá.

O carregamento dos dados foi feito automaticamente a partir da Base de Dados *MySQL*, tendo em conta que esta já se encontrava povoada.

7.3 Apresentação e caracterização dos *dashboards* implementados

Neste tópico iremos revelar os *dashboards* que criámos com base nos requisitos do nosso trabalho.

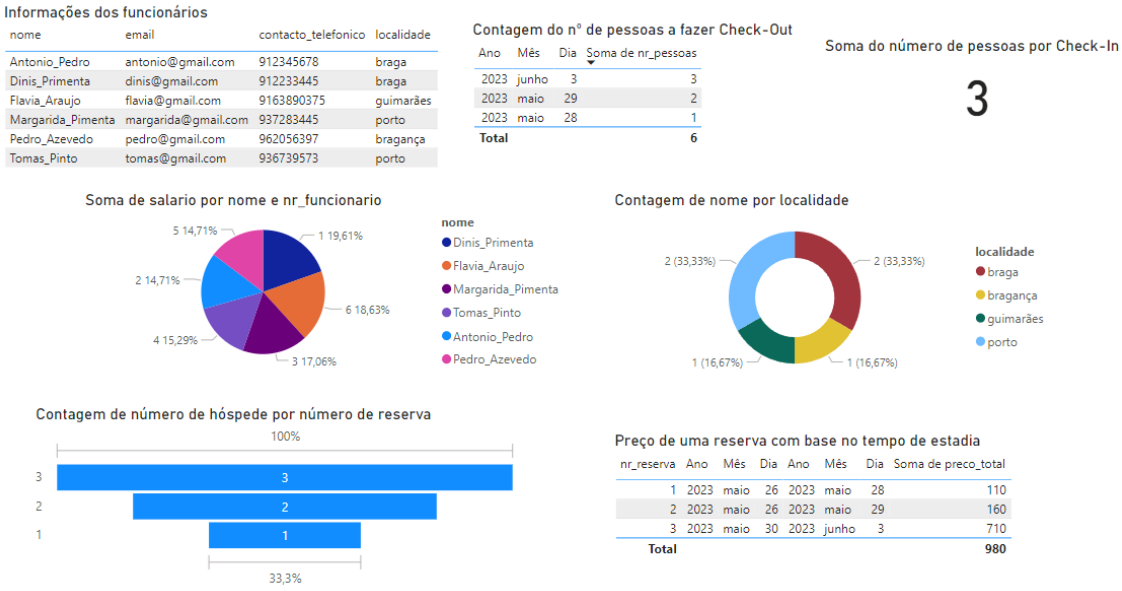


Figura 17 - Dashboards criados no PowerBI

Tal como é possível verificar, através das informações importadas da nossa base de dados para o *PowerBi*, foi possível criar *dashboards* com dados e estatísticas úteis relativamente aos funcionários e hóspedes do Hotel de Dona Arlete.

Um dos *dashboards* criados é uma tabela que guarda informação dos funcionários do hotel, registando esta os seus respetivos nomes, *emails*, contactos e localidades. Ainda relacionado com eles, temos também dois gráficos circulares que guardam igualmente informações sobre os funcionários tendo em conta o seu salário e localidade.

Outro exemplo que implementamos foi uma tabela que calcula o preço total de uma estadia tendo em conta o número de dias e, implicitamente, o tipo de quarto.

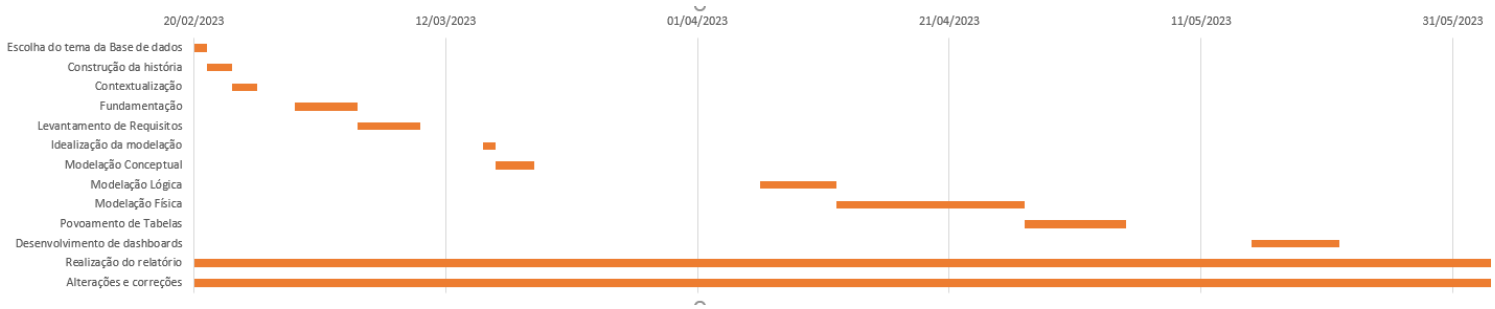
8 Conclusões e Trabalho Futuro

Em suma, é possível afirmar que a realização deste trabalho se tornou vantajosa e enriquecedora tanto a nível de adquirição de conhecimentos, bem como proporcionou benefícios no que toca ao trabalho em equipa.

Este foi um projeto que exigiu de nós bastante tempo e compreensão de conceitos relativos a Bases de Dados.

Durante a sua execução tivemos alguns impasses, surgiram dúvidas e problemas, no entanto ambos foram discutidos pelo grupo e, de seguida, comunicado aos professores de modo a esclarecer todas as dúvidas.

No que toca a trabalho futuro, pretendemos ainda melhorar a nossa implementação. Este foi um trabalho interessante e temos sem dúvida intenções de nos informarmos melhor sobre a área.


20/02/2023
12/03/2023
01/04/2023
21/04/2023
11/05/2023
31/05/2023
Escolha do tema da Base de dados
Construção da história
Contextualização
Fundamentação
Levantamento de Requisitos
Idealização da modelação
Modelação Conceptual
Modelação Lógica
Modelação Física
Povoamento de Tabelas
Desenvolvimento de dashboards
Realização do relatório
Alterações e correções