Universidade do Minho
Mestrado em Engenharia de Sistemas

Unidade Curricular de Administração e Exploração Avançada de Bases de Dados

Ano Letivo de 2018/2019

Aluguer de Automóveis

Célia, Márcia, Junior, Mathieu

Janeiro,2019

Data de Recepção	
Responsável	
Avaliação	
Observações	

Aluguer de Automóveis

Célia, Márcia, Júnior, Mathieu

Janeiro,2019

Resumo

O documento contém toda a informação sobre o processo de elaboração de uma arquitetura de base de dados, que será a base para a informatização do processo de uma empresa que aluga veículos. Inicialmente é feita a contextualização do problema apresentado, seguindo-se da análise do caso de estudo em questão. Ainda numa parte introdutória é descrita a motivação que levou a criação da empresa, assim como os objetivos a atingir. Esta primeira parte do relatório do projeto incluirá os seguintes pontos: contextualização do projeto, motivação e objetivos, análise de requisitos e a estrutura do relatório. Durante a fase de modelação conceptual da base de dados serão identificados, com base nos requisitos, as entidades envolvidas no sistema e os seus relacionamentos. De seguida, serão identificados os atributos associados às entidades e/ou relacionamentos. Em cada entidade serão apresentadas as suas chaves candidatas e a escolha da sua chave primária. Após concluído o modelo conceptual e antes de prosseguir com o modelo lógico, será feita a eleição do motor de base de dados, onde futuramente será implementada toda a estrutura que derivará dos vários desenhos que serão calculados durante todo o processo. Com base na segunda fase da metodologia, será derivado o modelo lógico correspondente ao modelo conceptual. De seguida serão validadas as relações, a qual corresponde a um dos passos mais importantes da metodologia. Após verificar que o modelo está consistente serão validadas as relações através do uso de transações do utilizador.

Área de Aplicação: Desenho e arquitetura de Sistemas de Bases de Dados.

Palavras-Chave: Bases de dados relacionais, análise de requisitos, entidades, atributos, Notação Chen, relacionamentos, metodologia, modelo conceptual, modelo lógico, modelo físico, SGBD

Índice

Índice de Figuras

Índice de Tabelas

1. Definição do sistema

Este primeiro capítulo tem como objetivo apresentar uma breve introdução ao projeto a realizar. Sendo assim, é necessário definir o contexto no qual se desenvolve o caso de estudo, seguido da sua descrição. É também importante perceber os motivos que levaram à criação da *OnRoad,* assim como os objetivos pretendidos.

1.1. Contexto de aplicação do sistema

A *OnRoad* é uma empresa de aluguer de viaturas localizada no centro do Porto há 8 anos. O *corebusiness* da empresa são os alugueres de curta duração, apesar de gerar mais processos administrativos é o que gera mais lucro. O seu principal público alvo são os turistas e representantes de empresas em viagens de negócios.

A cidade do Porto foi eleita o melhor destino europeu pela *European Best Destinations* duas vezes consecutivas (2016 e 2017), isto levou a um aumento exponencial no número de turistas e atraiu ainda investimento de grandes empresas internacionais que decidiram construir instalações na cidade. Com isto, a carteira de clientes habituais e esporádicos da *OnRoad* aumentou de forma significativa.

A empresa *OnRoad* efetua todo o processo de aluguer de um veículo desde escolha do cliente em loja até à conclusão do contrato de aluguer e posterior assinatura do mesmo. Está ainda incluído um controlo minucioso sobre as datas previstas e efetivas de entrega do veículo. A empresa está dividida em vários sectores nomeadamente o sector financeiro, recursos humanos, compra de veículos a entidades externas e aluguer do mesmo ao consumidor final. Apenas serão alvo de estudo o sector de aluguer de veículos. Será também englobado no problema, a gestão dos funcionários que fazem parte da empresa. Assim sendo, os funcionários e o supervisor de secção de aluguer serão abrangidos, mas todos os outros trabalhadores de outras secções não farão parte do sistema a desenvolver. O processo de aluguer de veículos segue a sequência descrita de seguida: na primeira fase, um funcionário atende um determinado cliente, procurando satisfazer o pedido do mesmo. Um cliente seleciona um veículo que pretenda alugar. Após selecionar um dos veículos disponíveis na empresa cabe ao respetivo funcionário tratar de todo o processo de aluguer. O processo de aluguer requer que o mesmo tenha associado uma caução e um seguro, que para o caso são obrigatórios, isto é, todo o cliente que pretenda efetuar o aluguer de um veículo tem

obrigatoriamente que pagar uma caução e assinar e pagar um seguro. Cada seguro tem associado uma descrição associada a cada tipo de seguro. Seguidamente é entregue ao cliente o veículo e respetiva chave. Neste documento, todo este processo será alvo de um estudo profundo, documentado através de uma pormenorizada análise de requisitos, para de acordo com a metodologia traçada, ser então desenvolvido um sistema que cumpra todos os objetivos definidos.

1.2. Fundamentação da implementação da base de dados

Devido ao elevado número de clientes deixou de ser viável armazenar as informações em papel, não só por razões de responsabilidade social e ambiental, mas também devido à dificuldade inerente aos processos de consulta. Estas circunstâncias causaram uma queda de 15% na taxa de serviço da *OnRoad.* Em resposta a este cenário, o CEO da *OnRoad* decidiu investir na informatização da empresa. O primeiro passo, será elaborar uma base de dados relacional para armazenar todos os processos inerentes ao aluguer de viaturas, buscando assim, otimizar os seus processos e consequentemente aumentar a taxa de serviço. É também esperado reduzir em 70% a quantidade de papel utilizada e aumentar em cerca de 15% a eficiência dos colaboradores.

1.3. Análise da viabilidade do processo

A implementação de uma base de dados relacional conta com inúmeras vantagens que tornam este processo rentável. Destacamos como principais vantagens da implementação da base de dados:

1. Redução da pegada ecológica causada pela empresa;
2. Maior celeridade nos processos administrativos;
3. Aumento da taxa de serviço;
4. Integridade da informação;
5. Possibilidade de centralização da informação em caso de crescimento da empresa.

2. Levantamento e análise de requisitos

2.1. Método de levantamento e de análise de requisitos adotado

Para possibilitar a criação de uma base de dados realmente funcional, é crucial conhecer minuciosamente os requisitos do negócio em questão. Para isto além de consultar colegas nossos que já alugaram viaturas, visitamos também um stand para esclarecer algumas particularidades relevantes no desenvolvimento da base de dados.

2.2. Requisitos levantados

2.2.1 Requisitos de descrição

A empresa *On Road* é constituída por um funcionário que é um gerente e por outros funcionários subordinados. Todos os funcionários devem estar registados no sistema.
O sistema atribuirá automaticamente a cada funcionário que se registe, um número de identificação único entre todos os funcionários. Para se registar, um funcionário terá de fornecer os seus dados pessoais, nomeadamente o nome, o país e cidade onde reside, contato telefónico, email e data de nascimento. O funcionário terá uma data de contratação e um salário, que deverá ser no mínimo 580 euros. Para se registar, um funcionário deverá ter idade igual ou superior a 18 anos e inferior a 65 anos. Um funcionário efetua zero ou mais alugueres, consoante a procura do serviço.

Para cada cliente será gerado um número sequencial e único, identificando o cliente de forma inequívoca. Um cliente para alugar um automóvel terá de fornecer o nome, número de identificação fiscal, o país e a cidade onde reside, contato de e-mail e telemóvel e data de nascimento. Um cliente deverá ter idade igual ou superior a 18 anos e ser portador de carta de condução. Um cliente é atendido por um e somente um funcionário. Um cliente escolhe um automóvel por cada aluguer.

Cada veículo será identificando por um identificador sequencial e único. O veículo é caracterizado por uma matrícula, uma marca, um modelo, número de quilómetros e

preço em novo (p.v.p.), isto é, quanto custa o veículo à empresa. O preço em novo de uma viatura deverá ser superior a 0€. Um veículo apresenta também um ano de compra e uma taxa de desvalorização ao ano, que é variável em função da marca.

Para cada seguro será gerado um número sequencial e único, identificando o seguro de forma inequívoca. Um seguro terá uma data de validade e preço de seguro. Este preço será constante, pois a empresa *OnRoad* definiu com a seguradora externa um valor que é fixo para toda a frota automóvel.

Para cada aluguer será gerado um número sequencial e único, identificando o aluguer de forma inequívoca. Um aluguer é definido por uma data de aluguer, data prevista de levantamento, data prevista de entrega, data real de entrega e preço de aluguer. O aluguer terá associado uma caução e os quilómetros percorridos. Por cada aluguer está associado obrigatoriamente um seguro. O preço de aluguer de um automóvel deverá ser superior a 0€.
Vários automóveis podem ser alugados várias vezes.

2.2.2 Requisitos de exploração

As bases de dados são criadas com o propósito de nos dar a possibilidade de manipular dados e de guardá-los, dados estes que são informação sobre algo, neste caso a informação que vamos guardar e manipular é sobre todas as entidades que têm algum relacionamento com a nossa empresa de aluguer de automóveis *OnRoad*. Estas informações devem ser dadas com a maior eficiência e rapidez aqueles que necessitam dela para realizarem o seu trabalho, mas para que essa entrega seja eficiente deve ser dada apenas a informação pedida e nada mais. Por exemplo, se o diretor de *marketing* da empresa *OnRoad* pedir o nome e contacto dos clientes para que este possa realizar um estudo sobre a satisfação dos nossos consumidores e a partir dai conseguir realizar campanhas marketing eficazes, a informação que deve ser passada a este é apenas uma tabela com nome, contacto (telemóvel e e-mail). Mas este diretor de *marketing* também pode pedir mais informação sobre os clientes como por exemplo quais os clientes que mais alugueres realizaram no ano de 2018 o que implicaria a movimentação de muita informação ao mesmo tempo, logo a gestão dos dados deve ser feita pelo sistema de base de dados, podendo assim manipular e guardar toda a informação de forma mais eficiente porque este contem toda a informação.

De seguida apresentamos alguns requisitos necessários à manipulação da base de dados:

- Consulta de todos os funcionários por parte do gerente,
- Consulta de todos os carros por parte de qualquer utilizador,
- Mostrar quais clientes alugaram determinado veículo,
- Mostrar a lista de carros por ordem crescente relativamente ao número de quilómetros,
- Apresentar os últimos alugueres realizados,
- Exibir o nome do funcionário que fez mais alugueres.

2.2.3 Requisitos de controlo

Uma base de dados é uma ferramenta de recolha e organização de informações que permite guardar e gerir da forma eficiente um sistema informático. De realçar que, todos os utilizadores devem aceder aos seus próprios dados, sem que de alguma forma não seja possível que esses mesmos utilizadores acedam a dados de outros. Esta realidade é cada vez mais relevante na atualidade, na medida em que as leis de proteção de dados estão cada vez mais atentas a situações que possam comprometer questões relacionadas com a privacidade e proteção de dados pessoais. Como proteção de dados entende-se a possibilidade de cada utilizador determinar de forma autônoma a utilização que é feita dos seus próprios dados. Para tal, é de extrema importância garantir formas capazes de monitorizar e limitar o acesso a vários dados por parte do utilizador.
Foram identificadas três espécies de utilizadores: o administrador, o cliente e o funcionário.

- **Administrador**

  Profissional responsável por gerir, instalar, configurar, atualizar e monitorizar a base de dados ou sistemas de base de dados. O administrador tem as permissões necessárias para resolver questões relacionadas com informações guardadas na base de dados. Assim sendo, o administrador pode alterar, remover, inserir e consultar toda a informação da base de dados.

- **Cliente**

  O cliente pode fazer todas as operações que digam respeito aos seus dados, exceto a remoção completa do seu registo.

- **Funcionário**

O funcionário tem acesso a todos os dados que sejam necessários para o aluguer de um veículo. Desta forma, este utilizador pode consultar a informação disponível na base de dados relativa ao cliente que solicitou o aluguer e a todas as informações inerentes a tal processo.

2.3. Análise geral dos requisitos

Ao levantamento de requisitos estão relacionados os principais problemas durante o desenvolvimento de qualquer software.

Para que os objetivos sejam atingidos, deve ser tudo muito bem planeado, elaborado e desenvolvido. No entanto, a análise de requisitos é a base de qualquer projeto, identificando, quantificando e priorizando-os, pois será todo o alicerce para um projeto bem-sucedido.

3. Modelação Concetual

3.1. Apresentação da abordagem de modelação realizada

A modelação conceptual é uma metodologia que é composta por três fases principais compreendidas num processo de elaboração de um modelo de dados partindo de uma análise detalhada dos requisitos. Este modelo é independente dos detalhes de implementação, das linguagens de programação, das plataformas de hardware, etc. Apresentamos de seguida as etapas do modelo:

1.Identificar as entidades do sistema;
2.Identficar os relacionamentos;
3.Identificar e relacionar atributos com as entidades e relacionamentos;
4.Apresentação e explicação do diagrama ER.

3.2. Identificação e caracterização das entidades

A primeira fase da construção do modelo concetual é a identificação das entidades que constituem o problema, sendo estas os objetos com maior interesse para o utilizador. A classificação de entidade de um modelo é um processo metódico e cuidadoso, na medida em que um objeto do problema pode ter uma importância considerável, mas não implicar que seja classificado como entidade no modelo concetual. Inicialmente serão apresentadas as entidades importantes do problema e a sua respetiva justificação. Posteriormente serão documentadas todas as entidades num dicionário de dados com uma descrição, um sinónimo e a justificação pela ocorrência dessa mesma entidade.

3.2.1 Cliente

Entidade que descreve todos os clientes que estão registados na base de dados. Cada cliente possui atributos próprios, tem uma existência autónoma e desta forma, podem ser identificados de forma inequívoca. Tais factos fazem deles uma entidade fundamental do sistema.

3.2.2 Funcionário

Entidade que representa todos os funcionários da *OnRoad* registados na base de dados. A entidade Funcionário possui também atributos próprios e trata-se de uma entidade responsável por alugar veículos a clientes.

3.2.3 Aluguer

O aluguer é a entidade que serve para os funcionários registarem tudo o que é relativo ao aluguer de um veículo, como por exemplo: o tempo de aluguer e o preço, a caução e o os quilómetros percorridos.

3.2.4 Veículo

A entidade Veículo é criada porque todos os veículos que a empresa possui estão diferenciados com algumas características para poderem ser identificados facilmente, desta forma, cada veículo possuí atributos próprios e podem ser especificados de forma única.

3.2.5 Seguro

Esta identidade existe porque todos os alugueres têm um seguro obrigatório, tendo este uma validade e um preço (fixo).

3.2.6 País

A entidade país foi criada para facilitar o acesso à informação do país do cliente e do funcionário. Desta forma consegue-se evitar erros de escrita relacionados com o país onde residem.

3.2.7 Cidade

A entidade cidade foi igualmente criada para facilitar o acesso à informação da cidade onde tanto o cliente como o funcionário residem. O facto de se tratar de uma entidade torna o acesso aos dados mais eficiente e rápido. Possui atributos próprios e é identificada de forma única cada entrada da tabela.

3.2.8 Dicionário de dados

Nome da Entidade	Descrição	Sinónimos	Ocorrência
Funcionário	Entidade que representa as várias pessoas que trabalham na *OnRoad*	Colaborador, Empregado	O Funcionário pode ter vários cargos e são os funcionários que fazem a empresa e que a representam.
Cliente	Esta entidade representa as várias pessoas que alugam veículos na *OnRoad*.	Consumidor	Cliente é das entidades mais importantes e principais pois são quem alugam veículos.
Aluguer	Esta entidade representa o aluguer feito por um funcionário a pedido de um Cliente.	Arrendar	Aluguer é a entidade responsável por qual o preço a considerar por veículo e as datas relevantes do ato de alugar.
Veículo	Esta entidade representa o veículo que pertence à empresa e que é alugado por um cliente	Viatura, Meio de transporte	Veículo é outra das entidades mais relevantes do problema. É o produto que é alugado ao cliente.
Seguro	Representa um contrato que o cliente necessita de assinar para efetuar um aluguer de um veículo. Este seguro é contratado externamente, porém o cliente necessita de o adquirir para o aluguer do carro	Proteção, Acordo	O Seguro é obrigatório para o cliente que pretende alugar um veículo.
Cidade	Representa a cidade onde o cliente e o funcionário residem.	Município, Localidade	O cliente e o funcionário necessitam fornecer a sua cidade de residência.
País	Representa o país onde o cliente e o funcionário residem.	Nação	O cliente e o funcionário necessitam fornecer o seu país de residência.

Tabela 1 - Tabela de entidades

3.3. Identificação e caracterização dos relacionamentos

Após identificar as entidades é necessário identificar os respetivos relacionamentos. Em primeiro lugar será apresentado um diagrama ER do modelo para ser mais fácil perceber as entidades e a forma como elas se relacionam entre si. A acompanhar o modelo ER será apresentado o Dicionário de dados dos relacionamentos.

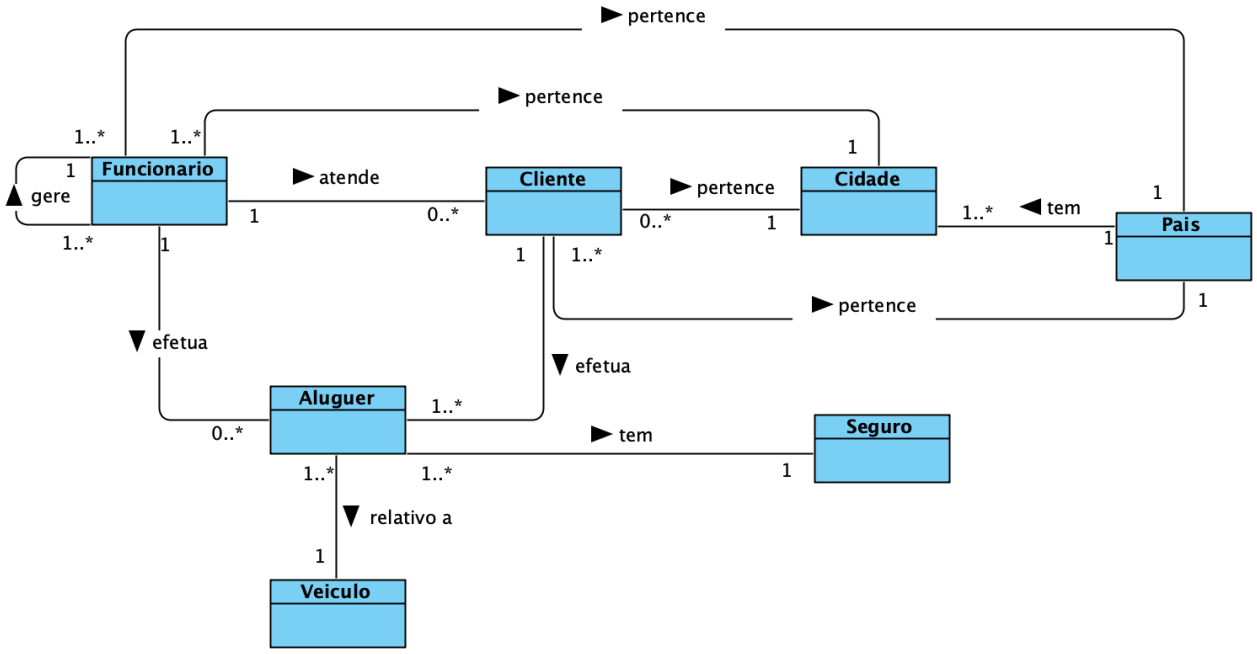

Figura 1 - Esquema de relacionamentos entre entidades

3.3.1 Dicionário de dados dos relacionamentos

Nome da Entidade	Multiplicidade	Relacionamento	Multiplicidade	Nome da Entidade
Funcionário	1	gere	0...N	Funcionário
Funcionário	1	atende	0...N	Cliente
Funcionário	1...N	pertence	1	Cidade
Funcionário	1...N	pertence	1	País
Funcionário	1	efetua	0...N	Aluguer
Cliente	1	efetua	1...N	Aluguer
Cliente	0...N	pertence	1	Cidade
Cliente	0...N	pertence	1	País
País	1	tem	1...N	Cidade
Aluguer	1...N	tem	1	Seguro
Veículo	1	relativo a	1...N	Aluguer

Tabela 2 - Tabela relacional

3.4. Identificação e caracterização das associações dos atributos com entidades e relacionamentos

Após ter-se concluído a fase da metodologia surge a fase de identificar os atributos de cada entidade e de cada relacionamento. Para tal, é necessário analisar os requisitos e perguntar "Qual a informação que preciso para guardar esta entidade ou este relacionamento?". Será também feita a atribuição dos domínios dos atributos, ou seja, atribuir o intervalo de valores que os atributos podem tomar. Apresentar-se-á então de seguida o dicionário de dados com os nomes das entidades, os seus atributos, a descrição de cada atributo, o domínio de valores, se o atributo pode ser nulo ou não e o tipo de atributo (simples, composto, derivado ou multi-valor).

3.4.1 Dicionário de dados dos Atributos das Entidades

Entidade	Atributo	Descrição	Tipo de dados e tamanho	Nulo	Multi-valor
Cliente	Id_Cliente	Código que identifica um cliente	INT	N	N
	Nome	Nome do Cliente	VARCHAR(80)	S	N

	Data de nascimento	Data de nascimento do cliente	DATE	N	N
	Telemovel	Número de telemóvel do cliente	INT(9)	N	N
	Email	Email do cliente	VARCHAR(45)	S	N
	Rua	Rua do cliente	VARCHAR(100)	S	N
	Carta de condução	Bolleano que esclarece se o cliente é ou não portador de carta de condução	BOLEAN	N	N
	Nif	Número de contribuinte do cliente	INT(5)	S	N
Funcionário	Id_funcionario	Código que identifica um Funcionário	INT	N	N
	nome	Nome do Cliente	VARCHAR(80)	N	N
	salario	Salário corresponde ao valor que o funcionário é remunerado.	DECIMAL(8,2)	N	N
	Data_contratação	Data correspondente ao dia em que o funcionário foi contratado.	DATE	N	N
	Data_nascimento	Data correspondente ao dia de nascimento do funcionário.	DATE	N	N
	email	Email do funcionário	VARCHAR(45)	N	N
	telémovel	Número de telemóvel do funcionário.	INT	N	N
	rua	Nome da rua onde o funcionário reside.	VARCHAR(75)	S	N
Aluguer	Id_aluguer	Código que identifica um Aluguer	INT	N	N
	caução	Valor que cliente paga como caução do aluguer.	DECIMAL(8,2)	N	N
	Data_real_entrega	Data e hora em que o cliente entregou o automóvel.	DATETIME	S	N
	Kms_percorridos	Número de quilómetros percorridos durante o aluguer.	DECIMAL(8,2)	S	N
	Data_aluguer	Data em que o cliente assinou contrato de aluguer.	DATE	S	N
	Data_prevista_entrega	Data estipulada para entrega do automóvel.	DATE	S	N
	Data_prevista_levantamento	Data estipulada para o cliente levantar o veículo.	DATE	S	N
Veiculo	Id_veiculo	Código que identifica um Veículo	INT	N	N

	Taxa_desvalorizaçao	Valor associado a cada veículo alugado.	DECIMAL(8,3)	S	N
	Nº_kms	Número de quilómetros que o veículo já possui na data de levantamento.	DECIMAL(8,2)	S	N
	matrícula	Matrícula que identifica o veículo.	VARCHAR(20)	N	N
	Preço_em_novo	Preço que o veículo custou à empresa *OnRoad*.	DECIMAL(8,2)	S	N
	Marca	Marca do veículo.	VARCHAR(50)	S	N
	Modelo	Modelo do veículo.	VARCHAR(50)	S	N
	Ano_compra	Data relativa ao ano em que o veículo foi comprado.	DATE	S	N
Seguro	Id_seguro	Código que identifica um Seguro	INT	N	N
	Preço_seguro	Valor que custa o seguro.	DECIMAL(8,2)	S	N
	Data_validade	Data de validade do seguro relativamente ao seguro.	DATE	S	N
Cidade	Id_cidade	Código que identifica uma Cidade	INT	N	N
	designação	Nome da cidade à qual pertence o cliente e o funcionário.	TEXT	S	N
País	Id_país	Código que identifica um País	INT	N	N
	designação	Nome do país à qual pertence o cliente e o funcionário.	TEXT	S	N

Tabela 3 - Tabela de entidades e atributos

3.5. Apresentação e explicação do diagrama ER

Apresenta-se de seguida na Figura o esquema concetual detalhado relativo ao projeto. Como não se trata de um projeto muito complexo, apenas utilizamos uma vista de desenvolvimento, usando a ferramenta *BrModel*.

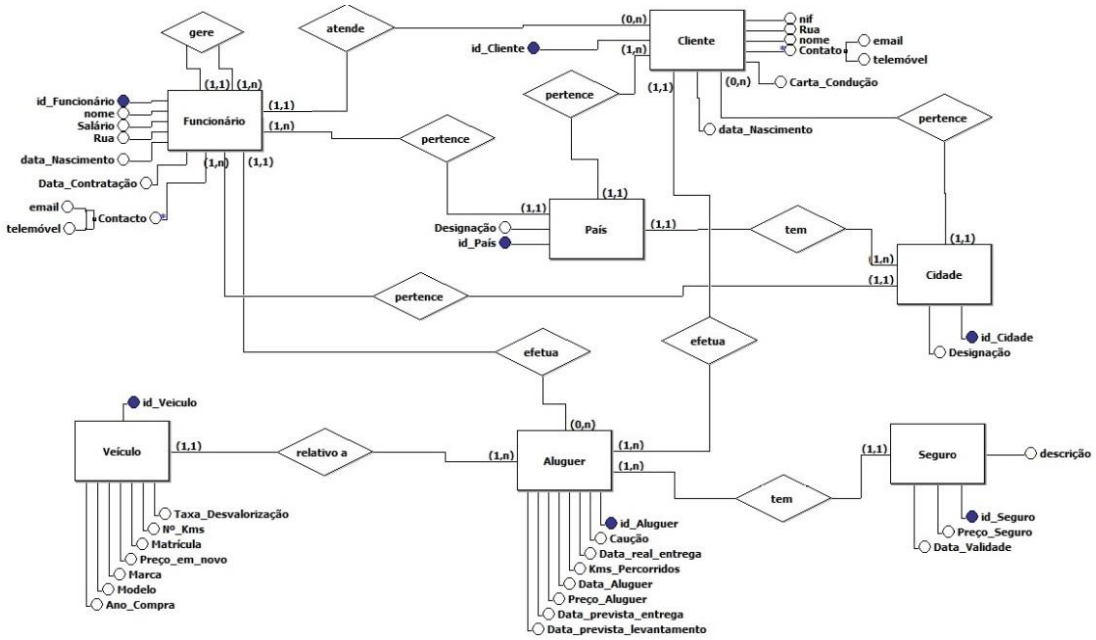

Figura 2- Modelo Concetual

4. Modelação Lógica

Terminada a modelação concetual, procedemos à fase de modelação lógica. Esta consiste na tradução do modelo conceptual anteriormente desenvolvido para um modelo lógico capaz de representar os requisitos definidos, assim como a validação do mesmo. Após a conclusão desta fase, deveremos obter um modelo lógico único representativo dos requisitos do sistema.

4.1. Construção e validação do modelo de dados lógico

4.1.1 Derivação para as relações do modelo lógico

Neste primeiro passo, é necessário derivar as entidades, relacionamentos e atributos considerados no modelo conceptual para as relações do modelo lógico. Foi usada a DDL (*Database Definition Language*), onde se descreve a composição de cada uma das relações, tendo por base as seguintes estruturas que se podem encontrar no modelo conceptual:

1 - Entidades Fortes
2- Relacionamentos um-para-muitos Recursivos
3- Relacionamentos um-para-muitos
4- Relacionamentos muitos-para-muitos

Para cada um dos pontos acima listados será descrito como foram abordados no contexto do problema de forma a derivar as relações e os relacionamentos alcançados no modelo lógico.

1 - Entidades Fortes

Uma entidade forte é uma entidade que não depende da existência de outra entidade e que pode por si só formar uma chave primária com os seus atributos. No modelo desenvolvido todas as nossas entidades são fortes.

- **Cliente** (idCliente, nome, nif, DataNascimento, pais, cidade, rua, cartaConducao, email, telemovel)
  Chave Primária: idCliente
  Chave Estrangeira: País, cidade

- **Funcionario** (idFuncionario, data_contrato, salario, telemovel, email, nome, FuncionarioSuperior, cidade, pais, rua, dataNascimento)
  Chave Primária: idFuncionario
  Chave Estrangeira: cidade, pais, FuncionarioSuperior

- **Veiculo** (idVeiculo, matricula, precoEmNovo, marca, modelo, nr_Kms, anoCompra, taxaDesvalorizacao)
  Chave Primária: idVeiculo

- **Aluguer** (idAluguer, dataAluguer, dataPrevistaLevantamento, dataPrevistaEntrega, dataRealEntrega, Cliente, Veiculo, precoAluguer, KmsPercorrido, Seguro, Funcionario, caucao)
  Chave Primária: idAluguer
  Chaves Estrangeiras: Cliente, Veiculo, Seguro, Funcionario

- **Seguro** (idSeguro, dataValidade, precoSeguro, descricao)
  Chave Primária: idSeguro

- **Pais** (idPais, designacao)
  Chave Primária: idPais

- **Cidade** (idCidade, designacao, pais)
  Chave Primária: idCidade
  Chave Estrangeira: pais

2- Relacionamentos um-para-muitos Recursivo

Neste tipo de relacionamentos, a entidade de multiplicidade “muitos” fica com um novo atributo (chave estrangeira) que é a chave primária representante da outra entidade. O modelo tem o relacionamento onde um funcionário gere outros funcionários. Existe um funcionário superior que é responsável por gerir todos os outos.

3- Relacionamentos um-para-muitos

Neste tipo de relacionamentos, a entidade de multiplicidade “muitos” fica com um novo atributo (chave estrangeira) que é a chave primária representante da outra entidade. O modelo apresenta vários relacionamentos deste tipo, entre outros, tem o exemplo que um funcionário atende vários clientes, ou ainda que um funcionário efetua muitos alugueres. Este terceiro tipo de relacionamento é o mais comum no modelo apresentado.

4- Relacionamentos muitos-para-muitos

Quando este tipo de relacionamentos acontece é criada uma nova relação que conterá todos os atributos que fazem parte do relacionamento. A nova relação terá uma cópia das chaves primárias das entidades que participam no relacionamento e funcionam como chaves estrangeiras. Uma dessas chaves ou mesmo as duas formarão a chave primária da nova relação ou ainda em combinações com outros atributos da nova relação. O modelo apresentado não contém nenhum relacionamento deste tipo devido ao facto de não existir argumentos para suportar tal decisão.

4.2. Desenho do modelo lógico

O modelo lógico apresentado na figura 3 foi construído de acordo com os pontos referidos nos pontos anteriores.

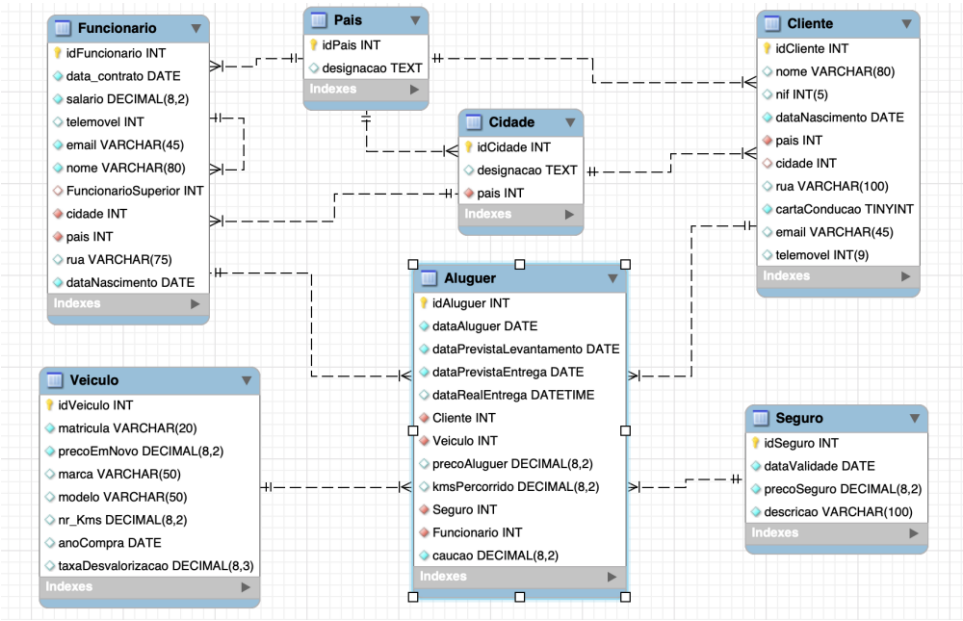

Figura 3 – Modelo Lógico

4.3. Validação do modelo com interrogações do utilizador

Um dos requisitos que é necessário para que o modelo possa ser considerado válido é responder a todas as perguntas que o utilizador possa fazer. Assim sendo foram selecionadas as consideradas pertinentes cuja viabilidade vai ser verificada através da comparação com o modelo lógico definido:

1. **Lista dos 5 clientes que mais fizeram alugueres em 2018**

Para ter acesso a esta informação são necessárias as tabelas Cliente e Aluguer. Após a junção da informação das tabelas filtrou-se os alugueres cuja data de aluguer tem o ano 2018. Para cada cliente é contado o número de alugueres que efetuou, esta informação é ordenada por ordem decrescente e são retiradas da tabela resultante as primeiras cinco linhas.

2. **Quantos alugueres se teve em 2018?**

Esta informação é acedida usando apenas a tabela Aluguer. Bastando contar o número de alugueres existentes com a data de aluguer com o ano 2018.

3. **Quantos clientes são de Portugal?**

Para a obtenção desta resposta temos duas possibilidades. A primeira poderá ser encontrada logo na tabela Cliente, pois possui um campo que é o país, porém é uma chave estrangeira o que implica ter que saber qual o número do identificador do país Portugal. Por outro lado, a informação poderá ser obtida da junção das tabelas Cliente e País, e filtrando pela designação do país.

4. **Quais os veículos que a sua entrega está atrasada?**

A informação dos veículos que tem a sua entrega atrasada pode ser calculada a partir da junção das tabelas veículo e aluguer. A informação será filtrada pelas datas previstas de entrega de veículos e a data atual e também com a data real da entrega do veículo, esta que não se encontra definida, pois o veículo ainda não foi entregue.

5. **Quais os clientes que alugaram carros e compraram o seguro do tipo A?**

Para a obtenção desta resposta é necessário o cruzamento de informação de três tabelas: a Cliente, Aluguer e Seguro. Filtrando o campo descrição do Seguro pela palavra A. Ou por outro lado poder-se-ia só aceder a duas tabelas a Cliente e Aluguer e filtrar pelo identificador do seguro, este previamente consultado.

6. **Quais os dois funcionários que efetuaram mais vendas em Dezembro de 2018**

Esta informação é obtida com a junção da tabela Funcionário e Aluguer. Efetuando a soma de preços de aluguer efetuados pelo funcionário, filtrado pelo ano 2018 e mês 12 e ordenado por ordem decrescente.

4.4. Validação do modelo com as transações estabelecidas

No modelo lógico é possível validar as transações através de mapas de transações, deste modo verifica-se visualmente as interações entre as diferentes entidades, atributos e relacionamentos com a finalidade de obter o resultado pretendido.

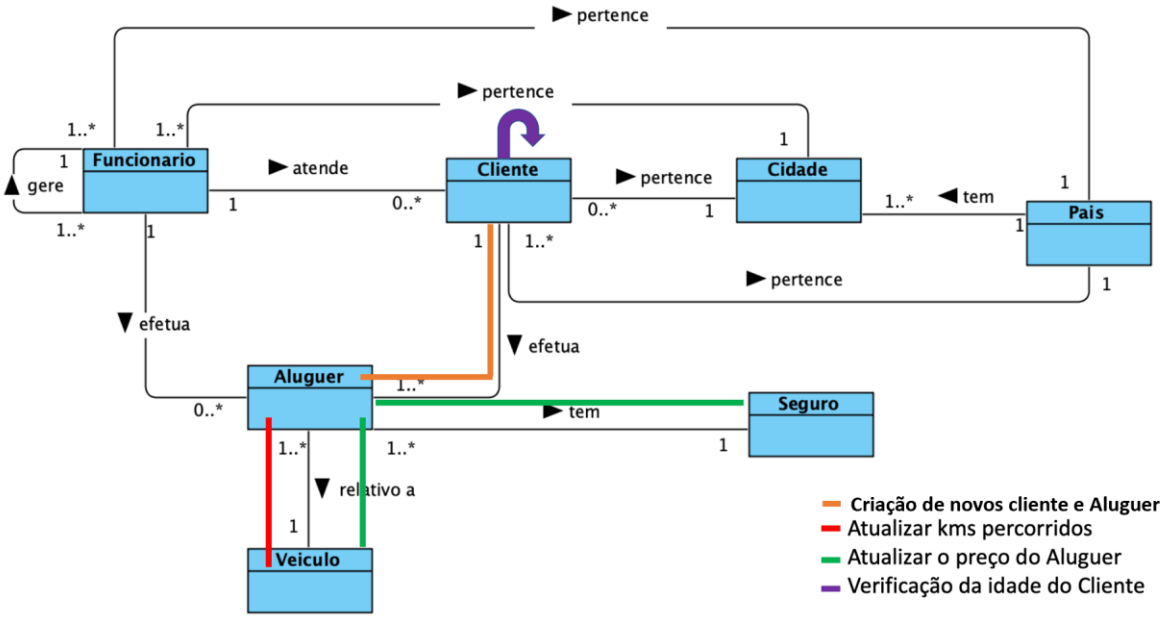

Figura 4 – Mapa de Transações

5. Implementação Física

5.1. Tradução do esquema lógico para o sistema de gestão de bases de dados escolhido em SQL

Para construirmos a Base de Dados proposta utilizamos como sistema de gestão de base de dados o *MySQL*. Esta decisão deve-se ao facto deste sistema ter sido o usado nesta Unidade Curricular, o que de certa forma veio facilitar a nossa implementação da Base de Dados.

A primeira tarefa para a modelação do modelo físico envolve a conversão das relações no modelo lógico num formato que pode ser implementado no SGDB. A primeira parte desse processo envolve agrupar as informações recolhidas durante o desenho do modelo lógico e documentadas no dicionário de dados, juntamente com as informações recolhidas durante a etapa de recolha e análise de requisitos. A segunda parte do processo usa essas informações para produzir o desenho das relações de base.

5.1.1 Relação País

Dominio id_Pais → Inteiro , NOT NULL

Dominio designação → texto padrao , NULL

```
CREATE TABLE IF NOT EXISTS `roadTrip`.`Pais` (
  `idPais` INT NOT NULL AUTO_INCREMENT,
  `designacao` TEXT NULL,
  PRIMARY KEY (`idPais`))
ENGINE = InnoDB;
```

5.1.2 Relação Cidade

Dominio id_Cidade → Inteiro , NOT NULL

Dominio designação → texto padrao , NULL

Dominio Pais → Inteiro, NOT NULL

```
CREATE TABLE IF NOT EXISTS `roadTrip`.`Cidade` (
  `idCidade` INT NOT NULL AUTO_INCREMENT,
  `designacao` TEXT NULL,
  `pais` INT NULL,
  PRIMARY KEY (`idCidade`),
  CONSTRAINT `fk_Cidade_Pais1`
    FOREIGN KEY (`pais`)
    REFERENCES `roadTrip`.`Pais` (`idPais`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

5.1.3 Relação Funcionário

Dominio id_Funcionario → Inteiro , NOT NULL

Dominio data_contrato → DATE, NOT NULL

Dominio salario → DECIMAL(8,2) , NOT NULL

Dominio telemovel → INT, NULL

Dominio email → string de tamanho variável com tamanho 45 , NOT NULL

Dominio nome → string de tamanho variável com tamanho 45 , NOT NULL

Dominio FuncionarioSuperior → Inteiro, NULL

Dominio Cidade → Inteiro, NOT NULL

Dominio Pais → Inteiro, NOT NULL

Dominio rua → string de tamanho variável com tamanho 75 , NULL

Dominio dataNascimenro → DATE, NOT NULL

```
CREATE TABLE IF NOT EXISTS `roadTrip`.`Funcionario` (
  `idFuncionario` INT NOT NULL AUTO_INCREMENT,
  `data_contrato` DATE NOT NULL,
  `salario` DECIMAL(8,2) NOT NULL CHECK (salario>=500),
  `telemovel` INT NULL,
  `email` VARCHAR(45) NULL,
  `nome` VARCHAR(45) NULL,
  `FuncionarioSuperior` INT NULL,
  `cidade` INT NULL,
  `pais` INT NULL,
  `rua` VARCHAR(75) NULL,
  `dataNascimento` DATE NULL,
  PRIMARY KEY (`idFuncionario`),
  CONSTRAINT `fk_FuncionarioSuperior`
    FOREIGN KEY (`FuncionarioSuperior`)
    REFERENCES `roadTrip`.`Funcionario` (`idFuncionario`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Funcionario_Cidade1`
    FOREIGN KEY (`cidade`)
    REFERENCES `roadTrip`.`Cidade` (`idCidade`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Funcionario_Pais1`
    FOREIGN KEY (`pais`)
    REFERENCES `roadTrip`.`Pais` (`idPais`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

5.1.4 Relação Veiculo

Dominio id_Veiculo → Inteiro , NOT NULL

Dominio matricula → string de tamanho variável com tamanho 20 , NOT NULL

Dominio precoEmNovo → DECIMAL(8,2) , NOT NULL

Dominio marca → string de tamanho variável com tamanho 50 , NULL

Dominio modelo → string de tamanho variável com tamanho 50 , NULL

Dominio nr_kms → DECIMAL(8,2) , NOT NULL

Dominio anoCompra → DATE, NULL

Dominio taxaDesvalorizalao → DECIMAL(8,3) , NULL

```
CREATE TABLE IF NOT EXISTS `roadTrip`.`Veiculo` (
  `idVeiculo` INT NOT NULL AUTO_INCREMENT,
  `matricula` VARCHAR(20) NOT NULL,
  `precoEmNovo` DECIMAL(8,2) NULL CHECK (precoEmNovo>=0),
  `marca` VARCHAR(50) NULL,
  `modelo` VARCHAR(50) NULL,
  `nr_Kms` DECIMAL(8,2) NULL,
  `anoCompra` DATE NULL,
  `taxaDesvalorizacao` DECIMAL(8,3) NULL,
  PRIMARY KEY (`idVeiculo`))
ENGINE = InnoDB;
```

5.1.5 Relação Cliente

Dominio id_Cliente → Inteiro , NOT NULL

Dominio nome → string de tamanho variável com tamanho 80 , NULL

Dominio nif → INT(5), NULL

Dominio dataNascimento → DATE, NOT NULL

Dominio Pais → Inteiro, NOT NULL

Dominio Cidade → Inteiro, NULL

Dominio rua → string de tamanho variável com tamanho 100 , NULL

Dominio cartaConducao → TINYINT, NOT NULL

Dominio email → string de tamanho variável com tamanho 45 , NOT NULL

Dominio telemovel → INT(9), NULL

```
CREATE TABLE IF NOT EXISTS `roadTrip`.`Cliente` (
  `idCliente` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(80) NULL,
  `nif` INT(5) NULL,
  `dataNascimento` DATE NOT NULL,
  `pais` INT NOT NULL,
  `cidade` INT NULL,
  `rua` VARCHAR(100) NULL,
  `cartaConducao` TINYINT NOT NULL,
  `email` VARCHAR(45) NULL,
  `telemovel` INT(9) NULL,
  PRIMARY KEY (`idCliente`),
  CONSTRAINT `fk_Cliente_Pais1`
    FOREIGN KEY (`pais`)
    REFERENCES `roadTrip`.`Pais` (`idPais`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Cliente_Cidade1`
    FOREIGN KEY (`cidade`)
    REFERENCES `roadTrip`.`Cidade` (`idCidade`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

5.1.6 Relação Seguro

Dominio id_Seguro → Inteiro , NOT NULL

Dominio dataValidade → DATE, NOT NULL

Dominio precoSeguro → DECIMAL(8,2) , NOT NULL

Dominio descricao → string de tamanho variável com tamanho 45 , NOT NULL

```
CREATE TABLE IF NOT EXISTS `roadTrip`.`Seguro` (
  `idSeguro` INT NOT NULL AUTO_INCREMENT,
  `dataValidade` DATE NULL,
  `precoSeguro` DECIMAL(8,2) NULL CHECK (precoSeguro>=0),
   `descricao` VARCHAR(100) NULL,
  PRIMARY KEY (`idSeguro`))
ENGINE = InnoDB;
```

5.1.7 Relação Aluguer

Dominio id_Aluguer → Inteiro , NOT NULL

Dominio dataAluguer → DATE, NOT NULL

Dominio dataPrevistaLevantamento → DATE, NULL

Dominio dataPrevistaEntrega → DATE, NULL

Dominio dataRealEntrega → DATE, NOT NULL

Dominio Cliente → Inteiro , NOT NULL

Dominio Veiculo → Inteiro , NOT NULL

Dominio precoAluguer → DECIMAL(8,2) , NOT NULL

Dominio kmsPercorrido→ DECIMAL(8,2) , NOT NULL

Dominio Seguro → Inteiro , NOT NULL

Dominio Funcionario → Inteiro , NOT NULL

Dominio caucao→ DECIMAL(8,2) , NOT NULL

```
CREATE TABLE IF NOT EXISTS `roadTrip`.`Aluguer` (
  `idAluguer` INT NOT NULL AUTO_INCREMENT,
  `dataAluguer` DATE NULL,
  `dataPrevistaLevantamento` DATE NULL,
  `dataPrevistaEntrega` DATE NULL,
  `dataRealEntrega` DATE NULL,
  `Cliente` INT NOT NULL,
  `Veiculo` INT NOT NULL,
  `precoAluguer` DECIMAL(8,2)  NOT NULL ,
  `kmsPercorrido` DECIMAL(8,2) NULL,
  `Seguro` INT NOT NULL,
  `Funcionario` INT NOT NULL,
  `caucao` DECIMAL(8,2) NOT NULL,
  PRIMARY KEY (`idAluguer`),
  CONSTRAINT `fk_Aluguer_Veiculo1`
    FOREIGN KEY (`Veiculo`)
    REFERENCES `roadTrip`.`Veiculo` (`idVeiculo`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Aluguer_Seguro1`
    FOREIGN KEY (`Seguro`)
    REFERENCES `roadTrip`.`Seguro` (`idSeguro`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Aluguer_Cliente1`
    FOREIGN KEY (`Cliente`)
    REFERENCES `roadTrip`.`Cliente` (`idCliente`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Aluguer_Funcionario1`
    FOREIGN KEY (`Funcionario`)
    REFERENCES `roadTrip`.`Funcionario` (`idFuncionario`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB:
```

5.2. Povoamento da base de dados criada

Para efetuar o povoamento da base de dados é necessário ter em atenção a ordem pela qual são inseridos os dados, pois existem dados que dependem de outros. Como tal em primeiro lugar fez-se a inserção dos dados relativos ao país, apresentam-se de seguida apenas excertos de exemplos dos dados inseridos:

```
INSERT INTO `Pais` (`idPais`, `designacao`)
VALUES (1, 'Portugal'),
```

De seguida optou-se pela tabela que usa a tabela de países que é a cidade:

```
INSERT INTO `Cidade` (`idCidade`, `designacao`, `pais`)
VALUES (1, ' Coimbra', 1), (idCidade, ' Leiria', 1), (idCidade, ' Guimaraes', 1),
```

A tabela a seguir a ter os dados inseridos foi a de Seguro, pois ela será utilizada noutra:

```
INSERT INTO `Seguro` (`idSeguro`, `dataValidade`, `precoSeguro`, `descricao`)
VALUES (1, '2019-12-16', '5.60', 'Seguro contra todos os riscos'),
```

Neste momento tem-se dados suficientes para criar as tabelas de Cliente e Funcionário :

```
INSERT INTO `Cliente` (`idCliente`, `nome`, `nif`, `dataNascimento`, `pais`, `cidade`, `rua`, `cartaConducao`, `email`, `telemovel`)
VALUES (1, 'Célia Figueiredo', '262646080', '1993-12-24', '1', '13', 'rua da Costa', 1,'a@gmail.com','912345678'),
```

```
INSERT INTO `Funcionario` (`idFuncionario`, `data_contrato`, `salario`, `telemovel`, `email`, `nome`, `cidade`,`pais`, `rua`,`dataNascimento`, `Funci
VALUES (1, '2008-12-24', '1000.00', '933337717', 'celianatalia@gmail.com', 'Natália Lemos', 13, 1,'rua da Costa','1994-12-24', null),
(idFuncionario, '2015-12-24', '1300.00', '933337717', 'celia@gmail.com', 'Celia Costa', 13, 1,'rua da Costa','1990-12-24', null),
```

Para a inserção de todos os dados na tabela Aluguer, é necessária informação do Veiculo em primeiro lugar:

```
INSERT INTO `Veiculo` (`idVeiculo`, `matricula`, `precoEmNovo`, `marca`, `modelo`, `nr_Kms`, `anoCompra`, `taxaDesvalorizacao`)
VALUES (1, '47-WT-55', '25000.00', 'Fiat', 'Punto', '1056.00', '2018-05-20', '0.050'),
```

E por fim, como esta tabela depende de quase todas as outras inseriu-se as informações relativas ao Aluguer:

```
INSERT INTO `Aluguer` (`idAluguer`, `dataAluguer`, `dataPrevistaLevantamento`, `dataPrevistaEntrega`,
`dataRealEntrega`, `Cliente`, `Veiculo`,`precoAluguer`, `kmsPercorrido`, `Seguro`,`Funcionario`, `caucao`)
VALUES (1, '2018-12-26', '2018-12-30', '2019-01-13', '2019-01-13', 11, 1,250.00, 900.00,1,2,500),
(idAluguer, '2018-06-24', '2018-06-30', '2018-07-24', '2018-07-24', 2, 2,250.00, 800.00,1,2,500),
```

5.3. Tradução das interrogações do utilizador para SQL

1. Lista dos 5 clientes que mais fizeram alugueres em 2018

```
SELECT nome, COUNT(a.idAluguer) as nAlugueres from Cliente as c
INNER JOIN Aluguer as a
   ON c.idCliente=a.Cliente
   WHERE YEAR (a.dataAluguer) = '2018'
   GROUP BY idCliente
   ORDER BY nAlugueres desc
   LIMIT 5;
```

2. Quantos alugueres tivemos em 2018?

```
-- Quantos alugueres tivemos em 2018?
SELECT COUNT(idAluguer) AS nrAlugueres FROM Aluguer
WHERE YEAR (dataAluguer)= '2018';
```

3. Quantos dos nossos clientes são de Portugal?

```
SELECT COUNT(idCliente) AS nrClientesPortugueses FROM Cliente
WHERE pais = 1;

-- ou

SELECT COUNT(idCliente) AS nrClientesPortugueses FROM Cliente as C
   INNER JOIN Pais as P
      ON C.Pais=P.idPais
WHERE P.designacao = 'Portugal';
```

4. Quais os veículos que a sua entrega está atrasada?

```
SELECT V.marca, V.matricula, a.dataPrevistaEntrega FROM Veiculo as V
INNER JOIN Aluguer AS a
   ON V.idVeiculo=a.Veiculo
   WHERE a.dataPrevistaEntrega <= current_date AND dataRealEntrega is null ;
```

5. Quais os clientes que alugaram carros e compraram o seguro do tipo A?

```
SELECT DISTINCT (C. idCliente), C.nome, S.descricao FROM Cliente AS C
INNER JOIN Aluguer AS A
   ON C.idCliente=A.Cliente
   INNER JOIN Seguro AS S
      ON A.Seguro=S.idSeguro
      WHERE S.descricao LIKE '%A - %';
-- ou

SELECT DISTINCT (C. idCliente), C.nome FROM Cliente AS C
INNER JOIN Aluguer AS A
   ON C.idCliente=A.Cliente
      WHERE a.Seguro = 1;
```

6. Quais os dois funcionários que efetuaram mais vendas em Dezembro de 2018

```
SELECT SUM(A.precoAluguer) AS vendas, F.nome FROM Funcionario as F
INNER JOIN Aluguer AS A
   ON F.idFuncionario=A.Funcionario
      WHERE YEAR (A.dataAluguer) = '2018' AND MONTH (A.dataAluguer) = '12'
      GROUP BY F.idFuncionario
      ORDER BY vendas DESC
      LIMIT 2;
```

5.4. Desenho das restrições/transações

Nesta fase são definidas as restrições gerais (ou regras de negócio) que servem para garantir a coerência no "mundo real" dos dados armazenados. Por exemplo, não faz sentido existir uma tarefa cuja data de início seja superior à data em que foi finalizada. Assim, é necessário impor regras que impeçam a ocorrência destes casos.

A idade de um funcionário deve ser igual ou superior a 18 anos e inferior a 65

```
delimiter $$
create trigger anoFuncionario
before insert on Funcionario
for each row
begin
DECLARE msg varchar(255);

if((datediff(now(),new.DataNascimento))/365<18) -- temos de /365 porque datediff devolve o número de dias
Then
 set msg= 'Não tem idade minima para exercer funções';
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = msg;

elseif ((datediff(now(),new.DataNascimento))/365>65)
Then
 set msg= 'EXCEDEU A IDADE PARA EXERCER FUNÇÕES';
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = msg;
end if;
end
$$
```

A idade de um Cliente que seja apto para conduzir também deverá ser igual ou superior a 18 anos.

```
delimiter $$
create trigger idadeInvalida
before insert on Cliente
for each row
begin
DECLARE msg varchar(255);

if((datediff(now(),new.DataNascimento))/365<18) -- temos de /365 porque datediff devolve o número de dias
Then
 set msg= 'Não tem idade minima para conduzir';
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = msg;

end if;
end
$$
```

Por outro lado, também é necessário atualizar o número de quilómetros do veículo utilizado após o aluguer, ou seja, no momento da entrega do veículo. O *trigge*r foi desenvolvido tendo em consideração a possibilidade de o funcionário inserir os dados erradamente. Caso seja a primeira atualização do número de quilómetros (número de km efetuados no aluguer iguais a nulo ou zero) ao número total de quilómetros percorridos pelo veículo é adicionado o número de quilómetros percorridos no aluguer. Se o número de quilómetros do aluguer for atualizado uma segunda vez ou mais (número de quilómetros do aluguer maior que zero) significa que o funcionário se enganou a introduzir os dados da primeira vez. Neste caso ao número total de quilómetros percorridos pelo veículo deverão ser retirados os valores inseridos erradamente e adicionados os valores agora adicionados. Além de corrigir o erro anterior o *trigger* valida também que o número de quilómetros inserido tem de ser obrigatoriamente positivo. Tal é atualizado com recurso ao seguinte gatilho:

```
delimiter $$
create trigger atualizarNrKm
before update on Aluguer
for each row
begin
DECLARE msg varchar(255);
    if New.kmsPercorrido <> old.kmsPercorrido && (old.kmsPercorrido = 0 II old.kmsPercorrido is null) && new.kmsPercorrido>0  then
        update Veiculo
            set nr_Kms= nr_Kms + New.kmsPercorrido where idVeiculo=New.Veiculo; -- se é 1ª vez adiciona o novo valor
    elseif new.kmsPercorrido>0 then
        update Veiculo
            set nr_Kms= nr_Kms - old.kmsPercorrido + New.kmsPercorrido where idVeiculo=New.Veiculo;  -- caso aconteça algum erro de esc
    else
        set msg= 'Número de kms adicionado incorreto';
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = msg;
    end if;
end; $$
```

Outro aspeto considerado importante para a consistência de dados foi a atualização do preço de aluguer. Utilizou-se a seguinte fórmula para o cálculo da atualização do preço: $precoAluguer = precoPorDia * diasDeAluguer + seguro * diasDeAluguer$ este que é calculado de forma automática, impedindo assim, o possível erro humano. E por isso optou-se pela criação do gatilho a seguir demonstrado.

```
delimiter $$
create trigger atualizarValor
before insert on Aluguer
for each row
begin
    declare totalAluguer decimal (8,2);
    declare seguro decimal (8,2);
    declare precoDocarroNovo decimal (8,2);
    declare taxas decimal (8,2);
    declare diasDeAluguer decimal (8,2);
    declare anosDoCarro int;
    declare precoPorDia decimal (8,2);

    select precoSeguro into seguro from Seguro where idSeguro= New.Seguro;
    select precoEmNovo into precoDocarroNovo from Veiculo where idVeiculo=New.Veiculo;
    select taxaDesvalorizacao into taxas from Veiculo where idVeiculo=New.Veiculo;
    SELECT TIMESTAMPDIFF(YEAR, curdate(), anoCompra) into anosDoCarro from Veiculo where idVeiculo=New.Veiculo;

    set diasDeAluguer := datediff(New.dataPrevistaEntrega,New.dataPrevistaLevantamento) ;
    set precoPorDia :=(precoDocarroNovo - (precoDocarroNovo*taxas*anosDoCarro))/365;

    if New.precoAluguer is null then
        set New.precoAluguer = precoPorDia*diasDeAluguer+seguro*diasDeAluguer ;
    end if ;

end; $$
```

É necessário também verificar que o cliente que irá efetuar um aluguer tem a carta de condução e para tal foi criado um gatilho que é acionado antes de inserir um novo aluguer. Verificando se o atributo onde se guarda se o cliente tem carta de condução está com o valor 0, caso seja verdade não deixa fazer a inserção na tabela aluguer.

```
delimiter $$
create trigger verificaCartaConducao
before insert on Aluguer
for each row
begin
    DECLARE msg varchar(255);
    declare cartaCond tinyint;

    select cartaConducao into cartaCond from Cliente where idCliente=new.Cliente;
    if cartaCond = 0 then
        set msg= 'Aluguer rejeitado porque cliente não tem carta de conducao';
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = msg;
    end if;
end
$$
```

Criou-se também um procedimento que consiste em registar um aluguer para um cliente novo. Este procedimento vai inserir na tabela cliente o novo cliente, e depois usará esse cliente novo, no aluguer que se estaria a registar. Deverão ser passados como argumentos todos os valores cujo valor é não nulo.

```
delimiter $$
create procedure registarAluguerParaClienteNovo (IN NomeCliente VARCHAR(80), IN dataNasc DATE, IN Pais INT, in cartaConducao tinyint,
                                                 IN  Veiculo INT, IN Seguro INT,IN  Funcionario INT, IN caucao DECIMAL(8,2), IN dataAluguer DATE,
                                                 IN dataPrevistaLevantamento DATE, IN dataPrevistaEntrega DATE)

BEGIN
declare AluguerID INT;
declare ClienteID INT;
declare erro bool default 0;
declare continue handler for sqlexception set erro=1;

start transaction;

insert into Cliente (idCliente,nome,dataNascimento, pais, cartaConducao )
        values (idCliente,NomeCliente,dataNasc, Pais, cartaConducao );
select idCliente into ClienteID
from Cliente
order by idCliente DESC
LIMIT 1;

insert into Aluguer (idAluguer,dataAluguer,dataPrevistaLevantamento,dataPrevistaEntrega,Cliente,Veiculo,Seguro, Funcionario,caucao)
      values (idAluguer,dataAluguer,dataPrevistaLevantamento,dataPrevistaEntrega,ClienteID,Veiculo,Seguro, Funcionario,caucao);


IF ERRO
THEN rollback;
ELSE COMMIT;
END IF;
END $$
```

6. Conclusões e Trabalho Futuro

O desenvolvimento de uma base de dados é um processo complicado e moroso, portanto, sendo feito de forma incorreta pode provocar elevados custos a uma organização. Na tentativa de diminuir possíveis erros, deve ser seguida uma metodologia de forma a sistematizar todo o processo de desenvolvimento.

Apesar dos vários pontos positivos da utilização de uma metodologia, o processo de modelação torna-se bastante intensivo e demorado, podendo então, ser desnecessário para casos simples. Já para problemas complexos e com vários utilizadores, a sua utilização é fundamental para que se possa assegurar que o sistema desenvolvido solicitado pela organização

Criar e implementar um sistema de base de dados é um trabalho que deve ser feito cautelosamente em diferentes níveis e para isso a melhor forma de o fazer é seguindo uma linha de trabalho organizada. Essa linha de trabalhos deve começar com a análise dos requisitos com o cliente e a partir dai estabelecer quais as entidades e quais os seus relacionamentos entre si para se poder criar uma forma ótima de armazenar e gerir os dados do sistema.

A metodologia sugerida pelo livro *Database Systems* foi seguida à risca na tentativa de que o resultado satisfizesse com sucesso as necessidades da *OnRoad*. Durante a primeira etapa de desenvolvimento, os futuros utilizadores da base de dados da *OnRoad* foram consultados para que cada passo fosse validado e portanto considerado de acordo com os requisitos.

De seguida foi implementado o modelo lógico e o esquema físico da base de dados e um conjunto de testes, que serviu como ponto de partida para iniciar o funcionamento do sistema.

Futuramente também se pretende acompanhar a utilização da base de dados por parte da *OnRoad*, onde se poderá intervir sempre que necessário, fazendo a manutenção do sistema. No entanto, qualquer Base de Dados necessita de manutenção regular, de modo a garantir que funcione bem e sem anomalias ao longo do tempo, caso isso não seja satisfeito, será diminuída lentamente a sua performance, deixando de ser funcional e, tornando-se portanto, inútil.

Por fim, achamos desta forma que conseguimos cumprir os objetivos definidos no início do projeto.

Referências

Thomas M. Connolly, Carolyn E. Begg - Database Systems: A Practical Approach to Design,
Implementation and Management - 4th Edition.

Lista de Siglas e Acrónimos

BD - Base de Dados

DW - *Data Warehouse*

SGBD - Sistema de Gestão de Bases de Dados

P.V.P -Preço de Venda ao Público

ER - Entidade Relacionamento

DDL- *Database Definition Language*

Anexos

I. Script de criação da base de dados

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema roadTrip
-- -----------------------------------------------------
DROP SCHEMA IF EXISTS `roadTrip` ;

-- -----------------------------------------------------
-- Schema roadTrip
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `roadTrip` DEFAULT CHARACTER SET utf8 ;
USE `roadTrip` ;

-- -----------------------------------------------------
-- Table `roadTrip`.`Pais`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `roadTrip`.`Pais` ;

CREATE TABLE IF NOT EXISTS `roadTrip`.`Pais` (
  `idPais` INT NOT NULL AUTO_INCREMENT,
  `designacao` TEXT NULL,
  PRIMARY KEY (`idPais`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `roadTrip`.`Cidade`
```

```
-- -----------------------------------------------------
DROP TABLE IF EXISTS `roadTrip`.`Cidade` ;

CREATE TABLE IF NOT EXISTS `roadTrip`.`Cidade` (
  `idCidade` INT NOT NULL AUTO_INCREMENT,
  `designacao` TEXT NULL,
  `pais` INT NULL,
  PRIMARY KEY (`idCidade`),
  CONSTRAINT `fk_Cidade_Pais1`
    FOREIGN KEY (`pais`)
    REFERENCES `roadTrip`.`Pais` (`idPais`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

CREATE INDEX `fk_Cidade_Pais1_idx` ON `roadTrip`.`Cidade` (`pais` ASC) VISIBLE;


-- -----------------------------------------------------
-- Table `roadTrip`.`Funcionario`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `roadTrip`.`Funcionario` ;

CREATE TABLE IF NOT EXISTS `roadTrip`.`Funcionario` (
  `idFuncionario` INT NOT NULL AUTO_INCREMENT,
  `data_contrato` DATE NOT NULL,
  `salario` DECIMAL(8,2) NOT NULL CHECK (salario>=500),
  `telemovel` INT NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  `FuncionarioSuperior` INT NULL,
  `cidade` INT NOT NULL,
  `pais` INT NOT NULL,
  `rua` VARCHAR(75) NULL,
  `dataNascimento` DATE NOT NULL,
  PRIMARY KEY (`idFuncionario`),
  CONSTRAINT `fk_FuncionarioSuperior`
```

```
    FOREIGN KEY (`FuncionarioSuperior`)
    REFERENCES `roadTrip`.`Funcionario` (`idFuncionario`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Funcionario_Cidade1`
    FOREIGN KEY (`cidade`)
    REFERENCES `roadTrip`.`Cidade` (`idCidade`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Funcionario_Pais1`
    FOREIGN KEY (`pais`)
    REFERENCES `roadTrip`.`Pais` (`idPais`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

CREATE INDEX `fk_FuncionarioSuperior_idx` ON `roadTrip`.`Funcionario` (`FuncionarioSuperior` ASC) VISIBLE;

CREATE INDEX `fk_Funcionario_Cidade1_idx` ON `roadTrip`.`Funcionario` (`cidade` ASC) VISIBLE;

CREATE INDEX `fk_Funcionario_Pais1_idx` ON `roadTrip`.`Funcionario` (`pais` ASC) VISIBLE;


-- -----------------------------------------------------
-- Table `roadTrip`.`Veiculo`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `roadTrip`.`Veiculo` ;

CREATE TABLE IF NOT EXISTS `roadTrip`.`Veiculo` (
  `idVeiculo` INT NOT NULL AUTO_INCREMENT,
  `matricula` VARCHAR(20) NOT NULL,
  `precoEmNovo` DECIMAL(8,2) NULL CHECK (precoEmNovo>=0),
  `marca` VARCHAR(50) NULL,
  `modelo` VARCHAR(50) NULL,
```

```
  `nr_Kms` DECIMAL(8,2) NULL,
  `anoCompra` DATE NULL,
  `taxaDesvalorizacao` DECIMAL(8,3) NULL,
  PRIMARY KEY (`idVeiculo`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `roadTrip`.`Cliente`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `roadTrip`.`Cliente` ;

CREATE TABLE IF NOT EXISTS `roadTrip`.`Cliente` (
  `idCliente` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(80) NULL,
  `nif` INT(5) NULL,
  `dataNascimento` DATE NOT NULL,
  `pais` INT NOT NULL,
  `cidade` INT NULL,
  `rua` VARCHAR(100) NULL,
  `cartaConducao` TINYINT NOT NULL,
  `email` VARCHAR(45) NULL,
  `telemovel` INT(9) NULL,
  PRIMARY KEY (`idCliente`),
  CONSTRAINT `fk_Cliente_Pais1`
    FOREIGN KEY (`pais`)
    REFERENCES `roadTrip`.`Pais` (`idPais`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Cliente_Cidade1`
    FOREIGN KEY (`cidade`)
    REFERENCES `roadTrip`.`Cidade` (`idCidade`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

CREATE INDEX `fk_Cliente_Pais1_idx` ON `roadTrip`.`Cliente` (`pais` ASC) VISIBLE;
```

```
CREATE INDEX `fk_Cliente_Cidade1_idx` ON `roadTrip`.`Cliente` (`cidade` ASC) VISIBLE;


-- -----------------------------------------------------
-- Table `roadTrip`.`Seguro`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `roadTrip`.`Seguro` ;

CREATE TABLE IF NOT EXISTS `roadTrip`.`Seguro` (
  `idSeguro` INT NOT NULL AUTO_INCREMENT,
  `dataValidade` DATE NULL,
  `precoSeguro` DECIMAL(8,2) NULL CHECK (precoSeguro>=0),
   `descricao` VARCHAR(100) NULL,
  PRIMARY KEY (`idSeguro`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `roadTrip`.`Aluguer`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `roadTrip`.`Aluguer` ;

CREATE TABLE IF NOT EXISTS `roadTrip`.`Aluguer` (
  `idAluguer` INT NOT NULL AUTO_INCREMENT,
  `dataAluguer` DATE NOT NULL,
  `dataPrevistaLevantamento` DATE NOT NULL,
  `dataPrevistaEntrega` DATE NOT NULL,
  `dataRealEntrega` DATE NULL,
  `Cliente` INT NOT NULL,
  `Veiculo` INT NOT NULL,
  `precoAluguer` DECIMAL(8,2)  NOT NULL ,
  `kmsPercorrido` DECIMAL(8,2) NULL,
  `Seguro` INT NOT NULL,
  `Funcionario` INT NOT NULL,
  `caucao` DECIMAL(8,2) NOT NULL,
```

```
  PRIMARY KEY (`idAluguer`),
  CONSTRAINT `fk_Aluguer_Veiculo1`
    FOREIGN KEY (`Veiculo`)
    REFERENCES `roadTrip`.`Veiculo` (`idVeiculo`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Aluguer_Seguro1`
    FOREIGN KEY (`Seguro`)
    REFERENCES `roadTrip`.`Seguro` (`idSeguro`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Aluguer_Cliente1`
    FOREIGN KEY (`Cliente`)
    REFERENCES `roadTrip`.`Cliente` (`idCliente`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Aluguer_Funcionario1`
    FOREIGN KEY (`Funcionario`)
    REFERENCES `roadTrip`.`Funcionario` (`idFuncionario`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

CREATE INDEX `fk_Aluguer_Veiculo1_idx` ON `roadTrip`.`Aluguer` (`Veiculo` ASC)
VISIBLE;

CREATE INDEX `fk_Aluguer_Seguro1_idx` ON `roadTrip`.`Aluguer` (`Seguro` ASC)
VISIBLE;

CREATE INDEX `fk_Aluguer_Cliente1_idx` ON `roadTrip`.`Aluguer` (`Cliente` ASC)
VISIBLE;

CREATE INDEX `fk_Aluguer_Funcionario1_idx` ON `roadTrip`.`Aluguer` (`Funcionario`
ASC) VISIBLE;


SET SQL_MODE=@OLD_SQL_MODE;
```

```
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

II. Script de povoamento da base de dados

```
USE `roadTrip` ;

INSERT INTO `Pais` (`idPais`, `designacao`)
VALUES (1, 'Portugal'),
(idPais, 'China'), (idPais, 'Espanha'), (idPais, 'Argentina'), (idPais, 'Franca'),
(idPais, 'Canadá'), (idPais, 'Croácia'), (idPais, 'Suécia'), (idPais, 'Brasil'),
(idPais, 'Luxemburgo'), (idPais, 'Alemanha'),
(idPais, 'Inglaterra'), (idPais, 'Argélia'), (idPais, 'Austria'), (idPais, 'Belgica');

INSERT INTO `Seguro` (`idSeguro`, `dataValidade`, `precoSeguro`, `descricao`)
VALUES (1, '2019-12-16', '5.60', 'A - Seguro contra todos os riscos'),
(idSeguro, '2020-11-19', '3.30', 'B - seguro contra danos'), (idSeguro, '2018-11-05',
'2.80', 'C - seguro pobre');

INSERT INTO `Cidade` (`idCidade`, `designacao`, `pais`)
VALUES (1, ' Coimbra', 1), (idCidade, ' Leiria', 1), (idCidade, ' Guimaraes', 1),
(idCidade, ' Silves', 1), (idCidade, ' Tomar', 1), (idCidade, ' Beja', 1), (idCidade, ' Viana
do Castelo', 1),
(idCidade, ' Tavira', 1),(idCidade, ' Cascais', 1), (idCidade, ' Portalegre', 1), (idCidade, '
Lisboa', 1), (idCidade, ' Viseu', 1),
(idCidade, ' Barcelos', 1), (idCidade, ' Vila Nova de Famalicão', 1), (idCidade, ' Aveiro',
1), (idCidade, ' Faro', 1),
(idCidade, ' Sintra', 1), (idCidade, ' Elvas', 1), (idCidade, ' Albufeira', 1), (idCidade, '
Lagos', 1), (idCidade, ' Almada', 1),
(idCidade, ' Évora', 1), (idCidade, ' Guimaraes', 1), (idCidade, ' Porto', 1), (idCidade, '
Maia', 1), (idCidade, ' Paris', 5), (idCidade, ' Londres', 12);

INSERT INTO `Veiculo` (`idVeiculo`, `matricula`, `precoEmNovo`, `marca`, `modelo`,
`nr_Kms`, `anoCompra`, `taxaDesvalorizacao`)
VALUES (1, '47-WT-55', '25000.00', 'Fiat', 'Punto', '1056.00', '2018-05-20', '0.050'),
(idVeiculo, '47-RS-25', '30000.00', 'Mercedes', 'cla', '1000.00', '2017-04-19', '0.100'),
(idVeiculo, '47-RS-26', '50000.00', 'Mercedes', 'gla', '10000.00', '2018-02-10', '0.100'),
(idVeiculo, '50-20-RS', '27000.00', 'Seat', 'Ibiza', '0.00', '2017-01-01', '0.060'),
(idVeiculo, '47-rs-30', '18000.00', 'Kia', 'f', '0.00', '2016-04-20', '0.070'),
(idVeiculo, '47-QP-47', '26000.00', 'Hyundai', 'i20', '100.00', '2016-09-19', '0.500'),
```

(idVeiculo, '47-WE-80', '54000.00', 'Hyundai', 'i30', '2000.00', '2017-04-22', '0.500'),
(idVeiculo, '25-PZ-99', '25000.00', 'Renault', 'Captur', '2000.00', '2016-03-17', '0.300'),
(idVeiculo, '83-RJ-26', '30000.00', 'Renault', 'Megane', '200.00', '2016-05-20', '0.300'),
(idVeiculo, '04-WE-OP', '33000.00', 'Ford', 'Fiesta', '33600.00', '2016-02-02', '0.800'),
(idVeiculo, '85-RX-56', '28000.00', 'Tesla', 'Model S', '100.00', '2018-01-20', '0.0050'),
(idVeiculo, '25-99-PZ', '25000.00', 'Tesla', 'Model X', '200.00', '2018-01-20', '0.0050'),
(idVeiculo, '42-TW-39', '60000.00', 'Toyota', 'CHR', '0.00', '2018-05-10', '0.050'),
(idVeiculo, '99-XY-20', '90000.00', 'Toyota', 'CHR', '0.00', '2018-05-10', '0.050'),
(idVeiculo, '47-QX-19', '17000.00', 'Ford', 'Focus', '0.00', '2018-05-10', '0.040'),
(idVeiculo, '47-YY-30', '19500.00', 'Toyota', 'yaris', '289.00', '2018-05-10', '0.090'),
(idVeiculo, '47-ZZ-50', '28900.00', 'Renault', 'Zoe', '2223.00', '2018-05-10', '0.008');

INSERT INTO `Cliente` (`idCliente`, `nome`, `nif`, `dataNascimento`, `pais`, `cidade`, `rua`, `cartaConducao`, `email`, `telemovel`)
VALUES (1, 'Célia Figueiredo', '262646080', '1993-12-24', '1', '13', 'rua da Costa', 1,'a@gmail.com','912345678'),
(idCliente, 'Rita Pereira', '262646080', '1984-05-06', '1', '12', 'rua da rita', 1,'a@gmail.com','912345678'),
(idCliente, 'Manuela Ferreira Leite', '112646080', '1965-12-24', '1', '14', 'rua da Costa', '1','a@gmail.com','912345678'),
(idCliente, 'Márcia Figueiredo', '262646080', '1994-01-21', '1', '3', 'rua do pinheiro', 1,'a@gmail.com','912345678'),
(idCliente, 'Junior', '262646080', '1993-12-24', '1', '15', 'rua da Costa', 1,'a@gmail.com','912345678'),
(idCliente, 'Pedro', '262646080', '1993-12-24', '1', '16', 'rua da Costa', 1,'a@gmail.com','912345678'),
(idCliente, 'Marco', '262646080', '1993-12-24', '1', '17', 'rua da Costa', 1,'a@gmail.com',912345678),
(idCliente, 'Luis', '262646080', '1993-12-24', '1', '18', 'rua da Costa', 1,'a@gmail.com',912345678),
(idCliente, 'Ricardo', '262646080', '1993-12-24', '1', '19', 'rua da Costa', 1,'a@gmail.com',91234544),
(idCliente, 'Joana', '262646080', '1993-12-24', '1', '20', 'rua da Costa', 1,'a@gmail.com',934567098),
(idCliente, 'Margarida', '262646080', '1993-12-24', '1', '21', 'rua da Costa', 1,'a@gmail.com', 987654909),

(idCliente, 'Juliana', '262646080', '1993-12-24', '1', '22', 'rua da Costa', 1,'a@gmail.com',0000000),
(idCliente, 'Cristina', '262646080', '1993-12-24', '1', '23', 'rua da Costa', 1,'a@gmail.com',0000000),
(idCliente, 'Maria', '262646080', '1993-12-24', '1', '24', 'rua da Costa', 1,'a@gmail.com',00000000),
(idCliente, 'Josefina Catarro', '262646080', '1993-12-24', '1', '25', 'rua da Costa', 1,'a@gmail.com',0000000),
(idCliente, 'José Carlos Malato', '262246080', '1993-12-24', '1', '17', 'rua da Costa', 1,'a@gmail.com',923454444),
(idCliente, 'Catalina Pestana', '232646080', '1980-12-24', '1', '11', 'rua da Costa', 1,'a@gmail.com',912223345),
(idCliente, 'António Costa', '342645080', '1974-12-24', '1', '11', 'rua da Costa', 1,'a@gmail.com',912227777),
(idCliente, 'José Socrates', '782666080', '1970-12-24', '1', '11', 'rua da Costa', 1,'a@gmail.com',967778889),
(idCliente, 'Luis Montenegro', '197646080', '1971-01-13', '1', '11', 'rua da Costa', 0,'a@gmail.com',922222787) ;

INSERT INTO `Funcionario` (`idFuncionario`, `data_contrato`, `salario`, `telemovel`, `email`, `nome`, `cidade`,`pais`, `rua`,`dataNascimento`,
`FuncionarioSuperior`)
VALUES (1, '2008-12-24', '1000.00', '933337717', 'celianatalia@gmail.com', 'Natália Lemos', 13, 1,'rua da Costa','1994-12-24', null),
(idFuncionario, '2015-12-24', '1300.00', '933337717', 'celia@gmail.com', 'Celia Costa', 13, 1,'rua da Costa','1990-12-24', null),
(idFuncionario, '2017-01-24', '900.00', '963126799', 'socrates@gmail.com', 'Socrates Lemos', 11, 1,'rua da Penuria', '1991-12-24',1),
(idFuncionario, '2018-12-24', '800.00', '913336617', 'mendes@gmail.com', 'Fernando Mendes', 13, 1,'rua do preco certo','1987-12-24', 2),
(idFuncionario, '2018-12-24', '850.00', '924447717', 'pedro@gmail.com', 'Pedro Mexia', 13, 1,'rua do mexia', '1989-12-24',1);

INSERT INTO `Aluguer` (`idAluguer`, `dataAluguer`, `dataPrevistaLevantamento`, `dataPrevistaEntrega`,

`dataRealEntrega`, `Cliente`, `Veiculo`,`precoAluguer`, `kmsPercorrido`, `Seguro`,`Funcionario`, `caucao`)

VALUES (1, '2018-12-26', '2018-12-30', '2019-01-13', '2019-01-13', 11, 1,250.00, 900.00,1,2,500),

(idAluguer, '2018-06-24', '2018-06-30', '2018-07-24', NULL, 2, 2,250.00, 800.00,1,2,500),

(idAluguer, '2018-06-01', '2018-06-07', '2018-06-30', '2018-06-30', 12, 3,250.00, 600.00,1,1,500),

(idAluguer, '2018-01-24', '2018-02-24', '2018-03-24', '2018-03-24', 2, 4,250.00, 200.00,2,1,500),

(idAluguer, '2018-04-24', '2018-05-24', '2018-06-24', '2018-06-24', 13, 5,250.00, 200.00,2,1,500),

(idAluguer, '2018-12-24', '2018-12-24', '2019-03-24', '2019-03-24', 2, 6,250.00, 234.00,2,2,500),

(idAluguer, '2017-06-13', '2017-07-01', '2017-07-29', '2017-07-30', 14, 7,250.00, 3455.00,1,1,500),

(idAluguer, '2018-06-24', '2018-07-24', '2018-08-15', '2018-08-15', 3, 8,250.00, 345.00,3,1,500),

(idAluguer, '2018-08-24', '2018-09-24', '2018-09-24', '2018-09-30', 3, 9,250.00, 698.00,1,1,500),

(idAluguer, '2017-12-21', '2018-01-03', '2018-01-24', '2018-01-24', 3, 10,250.00, 300.00,3,2,500),

(idAluguer, '2018-05-23', '2018-12-24', '2018-12-24', '2018-12-29', 15, 11,250.00, 235.00,1,1,500),

(idAluguer, '2018-12-24', '2018-12-24', '2019-01-24', null, 2, 12,250.00, 457.00,1,1,500),

(idAluguer, '2018-12-25', '2018-12-24', '2018-12-24', '2018-12-24', 4, 13,250.00, 345.00,1,1,500),

(idAluguer, '2018-05-26', '2018-05-27', '2018-06-01', '2018-06-01', 4, 14,250.00, 678.00,3,1,500),

(idAluguer, '2018-06-27', '2018-06-28', '2018-07-05', '2018-07-05', 15, 15,250.00, 841.00,1,1,500),

(idAluguer, '2018-06-12', '2018-06-24', '2018-06-30', '2018-06-30', 14, 16,250.00, 356.00,1,1,500),

(idAluguer, '2018-07-14', '2018-07-24', '2018-08-01', '2018-08-01', 13, 17,250.00, 666.00,3,1,500),

(idAluguer, '2018-12-01', '2018-12-10', '2018-12-24', '2018-12-24', 12, 1,250.00, 777.00,3,2,1500),

(idAluguer, '2018-06-04', '2018-06-24', '2018-07-24', '2018-07-24', 11, 1,250.00, 990.00,3,1,1500),
(idAluguer, '2017-01-24', '2017-02-24', '2017-03-24', '2017-03-24', 10, 11,250.00, 346.00,1,2,500),
(idAluguer, '2017-12-24', '2017-12-24', '2017-12-24', '2017-12-24', 11, 12,250.00, 678.00,3,1,500),
(idAluguer, '2017-12-24', '2017-12-24', '2017-12-24', '2017-12-24', 18, 17,250.00, 67.00,1,2,500),
(idAluguer, '2017-08-04', '2017-09-24', '2017-10-24', '2017-12-24', 12, 2,250.00, 56.00,2,3,500),
(idAluguer, '2017-06-03', '2017-06-24', '2017-07-04', '2017-07-04', 19, 3,250.00, 78.00,2,3,500),
(idAluguer, '2018-02-03', '2018-02-24', '2018-03-24', '2018-03-24', 13, 4,250.00, 77.00,2,4,500),
(idAluguer, '2018-07-04', '2018-07-24', '2018-07-30', '2018-07-30', 18, 5,250.00, 89.00,2,2,500),
(idAluguer, '2018-09-04', '2018-10-24', '2018-11-24', '2018-11-24', 17, 6,250.00, 443.00,1,4,500),
(idAluguer, '2018-12-24', '2018-12-24', '2018-12-24', NULL , 16, 7,250.00, 4567.00,1,3,500);