Universidade do Minho
Mestrado Integrado em Engenharia Informática
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Lectivo de 2020/2021

Rede de Transportes Públicos

Sofia Santos (A89615), Ema Dias (A89518),

Tânia Teixeira (A89613), Sara Queirós (A89491)

Dezembro, 2020

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

Rede de Transportes Públicos

Sofia Santos (A89615), Ema Dias (A89518),

Tânia Teixeira (A89613), Sara Queirós (A89491)

Dezembro, 2020

Resumo

Este relatório foi realizado no âmbito da unidade curricular de Bases de Dados, que tinha como objetivo principal desenvolver um sistema de base de dados com particular ênfase na análise, planeamento, modelação, arquitetura e implementação deste tipo de sistemas.

O nosso sistema pretende fazer uma renovação de uma base de dados de uma empresa de redes de transportes públicos, devendo ser, por isso, capaz de suportar grandes quantidades de informação, organizada e de fácil acesso e compreensão. De modo a poder estruturar o nosso sistema da melhor forma, o processo foi feito em três etapas, em que todas elas foram alvo de validação antes de avançar para a etapa seguinte.

Primeiramente, estabelecemos os requisitos que a nossa base de dados teria de ser capaz de suportar. De seguida fizemos uma modelação conceptual, em que estabelecemos as entidades essenciais e os relacionamentos existentes entre si, de acordo com os requisitos definidos. Antes de avançar para a etapa seguinte, fizemos a validação do nosso modelo conceptual.

Após esta validação, montamos o modelo lógico partindo do conceptual, fizemos a sua análise e validação através da normalização e das interrogações do utilizador. Quando consideramos que a modelação lógica estava pronta passamos à terceira fase.

Nesta terceira fase, a partir do modelo lógico geramos um modelo físico no *MySQL Workbench* garantindo assim que as modelações feitas anteriormente convergiriam para a obtenção de uma base de dados correta e segura.

Após concluída esta fase dá-se por terminado o projeto.

Área de Aplicação: Desenvolvimento e implementação de Sistemas de Bases de Dados.

Palavras-Chave: Bases de Dados, Bases de Dados Relacionais, Análise de Requisitos, Entidades, Atributos, Relações, ,Modelo Conceptual, Modelo Lógico, Normalização, Vistas de Utilização, MySQL Workbench, SQL

Índice

Índice de Figuras

Índice de Tabelas

1. Definição do Sistema

1.1. Contexto da aplicação do sistema

A nossa aplicação irá incidir sobre uma rede de transportes públicos. Escolhemos este tema visto que se trata de uma rede que implica o registo de diversas informações de forma organizada e consistente para o seu bom funcionamento.

Cada vez mais as cidades investem nas redes de transportes públicos. A escassez de parques de estacionamento nas grandes cidades, os custos associados a ter um veículo próprio e a poluição provocada pelos mesmos são razões que levam atualmente as pessoas a optar pelos transportes públicos para a sua locomoção.

Dado isto, consideramos bastante importante que um serviço de transportes públicos possua uma base de dados eficiente, daí a nossa escolha de tema para o trabalho prático. Esta afirmação é especialmente aplicável na situação atual de pandemia que vivemos. É essencial que haja controlo da lotação nas viagens de autocarro, por exemplo, e é também muito importante ter informação detalhada dos passageiros, para no caso da infeção de um passageiro a empresa de transportes ser capaz de indicar às autoridades de saúde pública quais as pessoas que viajaram com o dito passageiro.

1.2. Fundamentação da implementação da base de dados

Apesar de não termos acesso à base de dados de uma rede de transportes públicos, sabemos quais são as informações que são necessárias ao funcionamento da mesma.

Este sistema implica guardar diversas informações de maneira a poder gerir o registo dos utilizadores, sejam eles as pessoas que viajam nos autocarros ou os motoristas que os conduzem, assim como o registo dos meios existentes para o bom funcionamento da rede. Dado isto, é necessário organizar dados para que o sistema de informação tenha conhecimento dos tais utilizadores, dos percursos dos autocarros, das paragens do mesmo, assim como, dos passes que os clientes da rede possuem, e até mesmo dos autocarros em si.

Para isto, o sistema deve recolher, armazenar, processar e transmitir informações, de forma a garantir que pesquisas posteriores às mesmas são feitas de forma eficaz e válida.

Como tal, para tornar isto possível é necessário implementar uma Base de Dados, pois esta é uma ferramenta de recolha e organização de informações, nada mais do que aquilo que procuramos para responder às necessidades da rede.

1.3. Análise da viabilidade do projeto

Como referido anteriormente, este projeto tem como objetivo relacionar os dados que o sistema de informação contém, assim como, mais tarde, obter dados de pesquisas. Por outras palavras, deve tornar possível o registo de elementos e acesso aos mesmos.

Uma Base de Dados deve registar e guardar informações que se considera importante e necessárias à organização que gere o sistema.

Portanto, sabendo que um DBMS permite realizar diversas operações de gestão de dados, consideramos que a Base de Dados tornará praticável as necessidades do projeto.

2. Levantamento e Análise de Requisitos

2.1. Métodos de levantamento e de análise de requisitos adotados

Com o intuito de recolher informações acerca do funcionamento, na íntegra, de uma rede de transportes públicos, decidimos recorrer a vários métodos de procura e obtenção de informação, de forma a abranger todas as perspetivas e casos possíveis.

Este levantamento de informação, permite-nos estabelecer previamente um conjunto completo de requisitos básicos, e não básicos, que pretendemos que seja

possível implementar na nossa base de dados, corrigir problemas e erros pré-existentes, de forma a torná-la fidedigna de utilização.

Entre esses métodos de recolha de informação estão:

- **Questionários**: foram feitas a vários tipos de utilizadores: utilizadores frequentes com passe, utilizadores frequentes sem passe e utilizadores casuais com, e sem passe. Deste modo foi possível perceber as necessidades que estas pessoas têm e como seria a sua interação com o novo sistema da nova rede.
- **Simulações**: Diversas pessoas, ao longo de diversos dias da semana, utilizaram os transportes públicos, nomeadamente autocarros, de diversas empresas concorrentes, apurando as funcionalidades e serviços prestados por estes, bem como as suas limitações e os seus problemas.
- **Análise de gestão de informações**: Com análise externa das informações que são recolhidas pela empresa sobre os seus utilizadores e as informações sobre a empresa que são fornecidas aos utilizadores, foi possível obter-se uma aproximação da forma como a gestão dos dados é interligada e armazenada, percebendo assim os tópicos importantes a considerar para uma melhoria no tratamento das informações necessárias a um bom funcionamento da empresa.

2.2. Requisitos levantados

2.2.1 Requisitos de descrição

1. Cada paragem registada no sistema tem de ter associado um id, um nome e as coordenadas da sua localização (latitude e longitude).
2. Um percurso contém várias paragens, sendo armazenada a informação das horas a que cada percurso passa em cada paragem.
3. Cada percurso introduzido no sistema é identificado por um número e pela sua respetiva duração.
4. Uma viagem é realizada num percurso.
5. Quando uma viagem é registada no sistema, é necessário efetuar o registo com um id de viagem e a data da mesma.
6. Vários autocarros podem fazer a mesma viagem, assim como clientes.

7. Quando um autocarro é adquirido deve conter informações relativas à sua matrícula, lotação, data de inspeção mais recente e o tipo (o tipo de combustível que consome, i.e., se usa gasolina, gasóleo, gás natural, eletricidade, etc.)
8. Um motorista faz várias viagens, podendo conduzir diferentes autocarros.
9. Quando um motorista é contratado regista-se no sistema o seu nome, NIF, data de nascimento, salário que irá receber, IBAN e morada completa(rua, código postal e localidade).
10. Os clientes podem fazer viagem, podendo possui um passe ou não, e todas as viagens são registadas no sistema através do seu NIF.
11. Cada viagem que o cliente efetua tem um custo associado.
12. Um passe é criado a pedido de um cliente, sendo por isso obrigatório registar as suas informações pessoais tais como data de nascimento e nome. Também é associada a informação do tipo de passe e do seu custo mensal.

2.2.2 Requisitos de exploração

O sistema deverá ser capaz de contabilizar o total de viagens realizadas por dia.

O sistema deverá ser capaz de contabilizar os clientes que realizam mais viagens.

O sistema deverá ser capaz de contabilizar o lucro obtido através da venda de bilhetes por percurso.

O sistema deverá ser capaz de contabilizar os clientes que realizaram uma determinada viagem.

O sistema deverá ser capaz de contabilizar os autocarros que realizaram viagens numa determinada data.

O sistema deverá ser capaz de contabilizar o total de clientes por viagem e as viagens com mais clientes.

O sistema deverá ser capaz de contabilizar o lucro obtido pela venda de passes.

O sistema deverá ser capaz de contabilizar as paragens por onde passam mais percursos.

O sistema deverá ser capaz de contabilizar o número de viagens realizadas por motorista.

O sistema deverá ser capaz de contabilizar o número médio das viagens realizadas num determinado intervalo de tempo.

O sistema deverá ser capaz de contabilizar os percursos com o maior número de viagens.

2.2.3 Requisitos de controlo

Cada viagem pode ser realizada por um conjunto de clientes, mas apenas por um motorista.

Cada motorista pode realizar um conjunto de viagens.

Cada cliente pode realizar um conjunto de viagens, e poderá ter um passe ou não.

Cada viagem está associada a um percurso.

Cada percurso pode ser percorrido num conjunto de viagens.

Cada viagem é realizada por um autocarro, e um autocarro pode realizar um conjunto de viagens.

2.3. Análise e validação geral dos requisitos

Após o levantamento de requisitos foi necessário realizar uma análise geral dos mesmos. Para isso, a nossa equipa reuniu-se com o intuito de discutir a sua viabilidade. Através desta discussão chegamos a um acordo quanto aos diferentes detalhes e requisitos nos quais nos deveríamos realmente focar e desenvolver, ou desvalorizar.

Um dos detalhes que gerou uma maior indecisão foi a necessidade de cada cliente utilizador da nossa rede de transportes públicos ter de dar o seu NIF. Apesar de não ser algo que acontece atualmente em Portugal, para efeitos de simplificação da base de dados, decidimos que uma empresa que utilize esta base de dados deve armazenar sempre o NIF dos seus passageiros.

Após esta fase, ajustamos diferentes interpretações que tínhamos e chegamos a um consenso sobre os nossos requisitos, e aquilo que pretendemos desenvolver.

3. Modelação Conceptual

3.1. Apresentação da abordagem de modelação realizada

Após a recolha de informação e formulação dos requisitos, inicia-se o processo de planeamento da estrutura e design da Base de Dados que pretendemos implementar.

Para facilitar o processo de construção da nossa base de dados, o mais adequado é começar com um Diagrama ER, uma vez que é um tipo de fluxograma que ilustra como é que entidades (pessoas, objetos, conceitos, etc) se relacionam entre si. A sua construção é simples, uma vez que o primeiro passo é identificar as entidades do problema, de seguida os seus relacionamentos e por fim os atributos, quer seja das entidades ou dos relacionamentos.

Assim, uma modelação conceptual deve representar um modelo de estrutura relativamente a entidades, às suas relações e atributos de acordo com as necessidades de gestão, descrevendo assim, um modelo comportamental. No nosso caso, com o intuito de gerir da melhor forma um sistema de informação relativo a uma rede de transportes, fizemos esta modelação de forma a permitir a aproximação a um sistema real.

3.2. Identificação e caracterização das entidades

Avaliando o funcionamento das redes de transportes consideramos que as entidades essenciais seriam: motorista, viagem, cliente, passe, percurso, paragem e autocarro.
Para cumprir os requisitos apresentados anteriormente, as entidades possuem os seguintes atributos que são dados elementares que caracterizam as mesmas:

- **Motorista** é a entidade que representa quem trabalha para a rede de transportes, conduzindo os autocarro. É identificado através do seu NIF e caracterizado pelo seu nome, data de nascimento, salário, IBAN e morada (constituída por rua, código postal e localidade, sendo por isso um atributo composto).
- **Cliente** representa um utilizador da rede que é identificado pelo seu NIF, uma vez que consideramos que no ato de uma compra, o cliente pede sempre

fatura com número de contribuinte. O cliente não possui mais atributos, já que um cliente pode não ter passe, efetuando a viagem através da compra do bilhete com o seu NIF, e desta forma não temos mais nenhuma informação acerca do cliente para armazenar.

- **Passe** é a licença de transporte livre mensal, sendo o id a chave identificadora desta entidade. Para além disso, o mesmo possui informações sobre o seu custo mensal (que varia com a idade) , data de nascimento, nome e morada do cliente. A morada é novamente um atributo composto, tal como na entidade Motorista, com as mesmas informações associadas.
- **Autocarro** representa o veículo essencial à rede de transportes. Este é identificado pela matrícula e é descrito pela lotação, tipo (se é elétrico, a gasolina, etc.) e data de inspeção.
- **Viagem** tem como chave um id e possui como atributo a data em que foi realizada.
- **Percurso** é um caminho que será percorrido numa viagem, sendo o seu número a chave primária. Este atributo também armazena a duração de cada percurso.
- **Paragem** representa um dos locais onde o autocarro para e onde entram e saem passageiros. Cada paragem tem um id associado, a sua chave primária. De forma a completar as informações, é possível ainda saber o seu nome e a sua localização através das coordenadas (atributo composto por latitude e longitude).

3.3. Identificação e caracterização dos relacionamentos

Nesta modelação englobamos vários tipos de relacionamentos entre as entidades. Assim, apresentamos uma análise e explicação individual feita a cada um desses relacionamentos:

- **Relacionamento Paragem - Percurso**

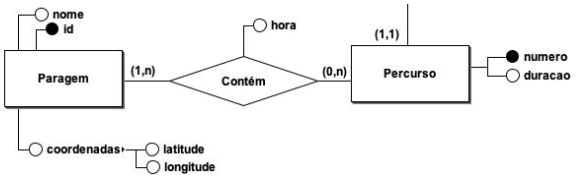

Figura 1 - Relacionamento Paragem-Percurso.

Relacionamento: Percurso contém Paragem

Descrição: Com o intuito de poder estabelecer vários percursos que passem na mesma paragem, é importante armazenar a informação desta forma para, aquando de uma extensão do alcance da empresa, ao criar novos percursos, poder aceder diretamente às paragens registadas no sistema e não ter de rever os percursos existentes para analisar as paragens já existentes.

Multiplicidade: Paragem(1,n) - Percurso (0,n)

Um certo percurso contém diferentes paragens. Uma paragem pode fazer parte de vários percursos.

Atributos: No relacionamento "contém" entre percurso e paragem existe o atributo "hora", pois o percurso passa a uma certa hora na paragem, sendo assim possível registar essa informação.

- **Relacionamento Percurso-Viagem**

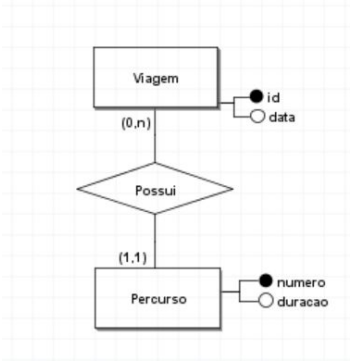

Figura 2 - Relacionamento Percurso-Viagem.

Relacionamento: Viagem possui Percurso

Descrição: Com o objetivo de poder estabelecer os diversos percursos que existam, quiçá adicionar novos, e atribuí-los por viagens, é importante armazenar a informação separadamente de modo a conseguir obter os percursos abrangidos numa determinada viagem. Assim também não é necessário definir percursos novos por cada viagem.

Multiplicidade: Viagem(0,n) - Percurso (1,1)

Várias viagens podem fazer o mesmo percurso. Uma viagem contém um e só um percurso.

Atributos: O relacionamento não tem atributos associados.

- **Relacionamento Autocarro-Viagem**

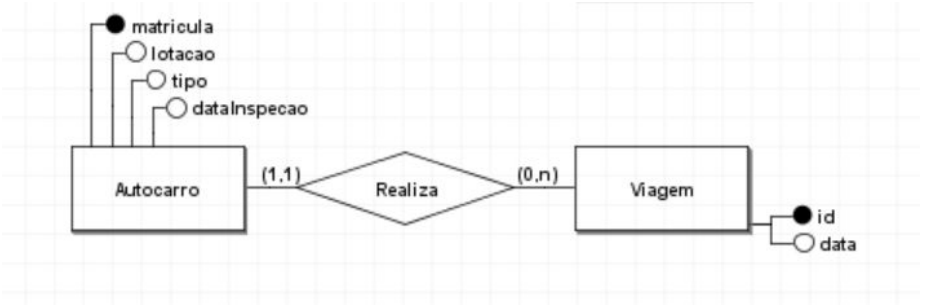

Figura 3 - Relacionamento Autocarro-Viagem.

Relacionamento: Autocarro realiza Viagem

Descrição: Uma vez que um autocarro pode realizar várias viagens, é importante relacionar estas entidades, de modo a poder armazenar as informações relativas às viagens realizadas por um autocarro.

Multiplicidade: Viagem(0,n) - Autocarro (1,1)

Um autocarro poderá realizar inúmeras viagens, no entanto, uma viagem apenas será realizada por um autocarro.

Atributos: O relacionamento não tem atributos associados.

- **Relacionamento Motorista-Viagem**

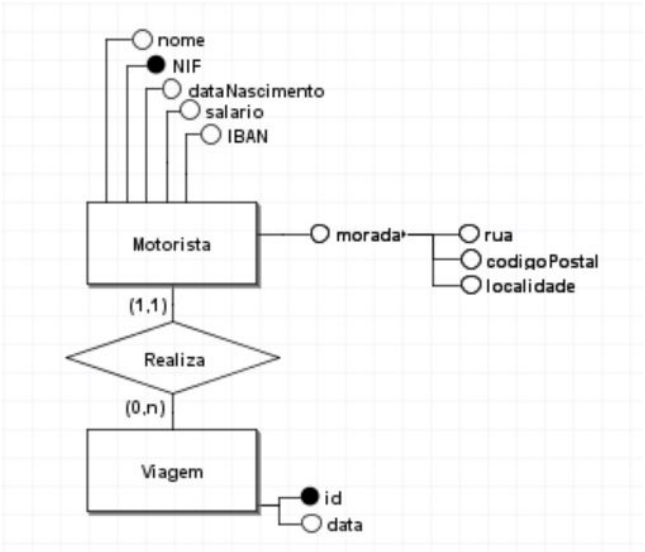

Figura 4 - Relacionamento Motorista-Viagem.

Relacionamento: Motorista realiza Viagem

Descrição: Cada viagem é realizada por um motorista, é importante armazenar as informações desta maneira para depois conseguirmos obter as viagens realizados por um determinado motorista.

Multiplicidade: Viagem(0,n) - Motorista (1,1)

Um motorista pode realizar várias viagens mas uma viagem só pode ser realizada por um único motorista.

Atributos: O relacionamento não tem atributos associados.

- **Relacionamento Cliente-Viagem**

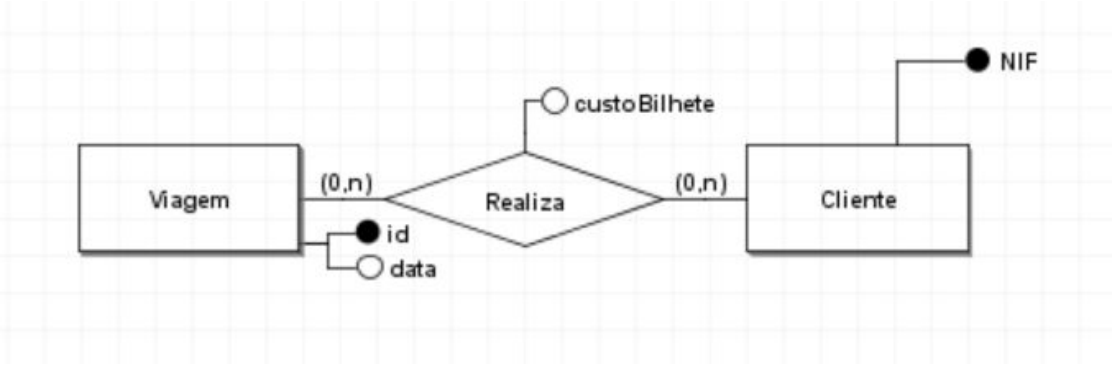

Figura 5 - Relacionamento Cliente-Viagem.

Relacionamento: Cliente realiza Viagem

Descrição: Com o intuito de ser possível registar a quantidade de clientes que realizam viagens e o preço que pagam por cada bilhete, consideramos que seria importante estabelecer este relacionamento,

Multiplicidade: Viagem(0,n) - Cliente (0,n)

Vários clientes podem realizar uma viagem, assim como várias viagens podem ser realizadas pelo mesmo cliente.

Atributos: No relacionamento "realiza" entre cliente e viagem existe um atributo associado que é "custoBilhete", uma vez que clientes sem passe, ou passes que não abrangem essas viagens deverão ter um custo associado.

- **Relacionamento Cliente-Passe**

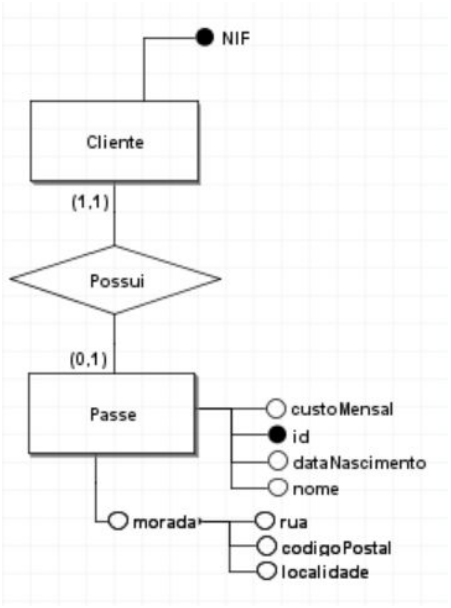

Figura 6 - Relacionamento Cliente-Passe.

Relacionamento: Cliente possui Passe

Descrição: Com o intuito de poder atribuir o respetivo passe a um determinado cliente, é importante manter estes registos para, de forma mensal, pagar a respetiva mensalidade e poder usufruir do passe nas suas viagens, armazenando assim as suas informações.

Multiplicidade: Cliente(1,1) - Passe(0,1)

Um passe obrigatoriamente terá que pertencer a um cliente, no entanto, um cliente poderá ou não ter um passe.

Atributos: O relacionamento não tem atributos associados.

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

ENTIDADE: Paragem

Atributos	Tipo de Dados	Nulo	Composto	Multivalorado	Derivado	Candidato
id	INT	Não	Não	Não	Não	SIM
nome	VARCHAR(30)	Não	Não	Não	Não	Não
latitude	DOUBLE	Não	Sim	Não	Não	Não
longitude	DOUBLE	Não	Sim	Não	Não	Não

Tabela 1 - Caracterização dos atributos de Paragem.

ENTIDADE: Percurso

Atributos	Tipo de Dados	Nulo	Composto	Multivalorado	Derivado	Candidato
numero	INT	Não	Não	Não	Não	SIM
duracao	VARCHAR(30)	Não	Não	Não	Não	Não

Tabela 2 - Caracterização dos atributos de Percurso.

ENTIDADE: Viagem

Atributos	Tipo de Dados	Nulo	Composto	Multivalorado	Derivado	Candidato
id	INT	Não	Não	Não	Não	SIM
data	DATE	Não	Não	Não	Não	Não

Tabela 3 - Caracterização dos atributos de Viagem.

ENTIDADE: Autocarro

Atributos	Tipo de Dados	Nulo	Composto	Multivalorado	Derivado	Candidato
matricula	VARCHAR(8)	Não	Não	Não	Não	SIM
lotacao	INT	Não	Não	Não	Não	Não
tipo	FLOAT	Não	Não	Não	Não	Não
dataInspecao	DATE	Não	Não	Não	Não	Não

Tabela 4 - Caracterização dos atributos de Autocarro.

ENTIDADE: Motorista

Atributos	Tipo de Dados	Nulo	Composto	Multivalorado	Derivado	Candidato
nome	VARCHAR(100)	Não	Não	Não	Não	Nao
nif	INT	Não	Não	Não	Não	SIM
dataNascimento	DATE	Não	Não	Não	Não	Não
salario	FLOAT	Não	Não	Não	Não	Não
iban	VARCHAR(50)	Não	Não	Não	Não	Não
rua	VARCHAR(100)	Não	Sim	Não	Não	Não
codigoPostal	VARCHAR(8)	Não	Sim	Não	Não	Não
localide	VARCHAR(30)	Não	Sim	Não	Não	Não

Tabela 5 - Caracterização dos atributos de Motorista.

ENTIDADE: Cliente

Atributos	Tipo de Dados	Nulo	Composto	Multivalorado	Derivado	Candidato
nif	INT	Não	Não	Não	Não	SIM

Tabela 6 - Caracterização dos atributos de Cliente.

ENTIDADE: Passe

Atributos	Tipo de Dados	Nulo	Composto	Multivalorado	Derivado	Candidato
custoMensal	DE	Não	Não	Não	Não	Não
id	INT	Não	Não	Não	Não	SIM
dataNascimento	DATE	Não	Não	Não	Não	Não
nome	VARCHAR(50)	Não	Não	Não	Não	Não
rua	VARCHAR(100)	Não	Sim	Não	Não	Não
codigoPostal	VARCHAR(8)	Não	Sim	Não	Não	Não
localidade	VARCHAR(30)	Não	Sim	Não	Não	Não

Tabela 7 - Caracterização dos atributos de Passe.

3.5. Detalhe ou generalização das entidades

Uma vez que as nossas entidades não partilham o mesmo tipo de atributos, não consideramos que fosse pertinente a criação de uma superclasse que os passasse por herança. Estas apenas partilham alguns atributos, tal como, por exemplo, entre passe e motoristas, cujos atributos caracterizadores que possuem em comum são morada, data de nascimento e nome. Por isso, não consideramos essencial a criação de uma superclasse para englobar estes atributos, mas poderíamos ter optado por fazê-lo, o que seria justificável.

3.6. Apresentação e explicação do diagrama ER

Após uma detalhada explicação de todas as entidades existentes no sistema, de cada um dos relacionamentos entre elas e os seus respetivos atributos, apresentamos, de uma forma visual, o seguinte modelo conceptual.

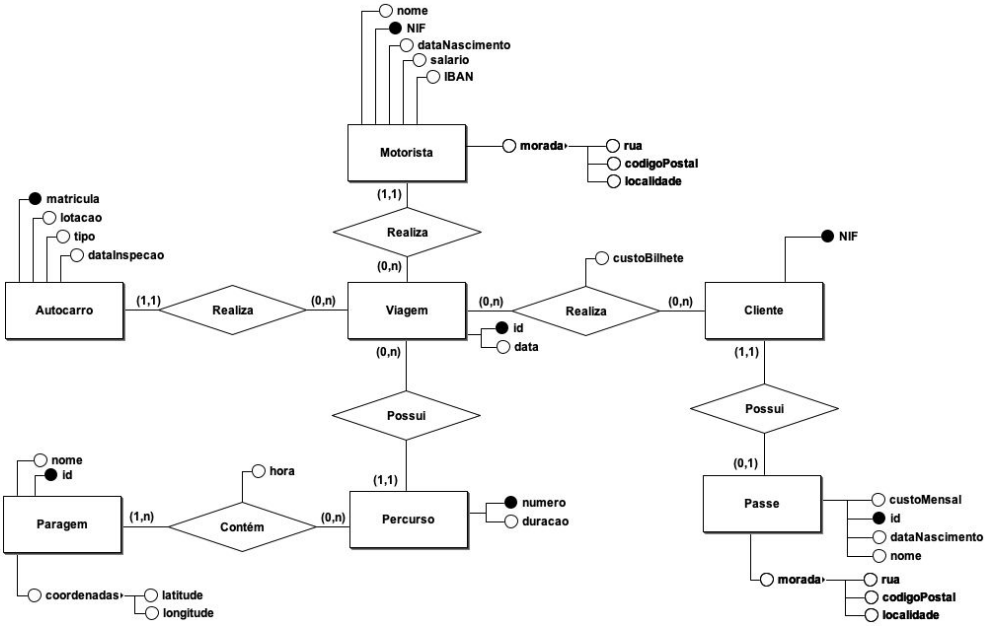

Figura 7 - Modelo Conceptual.

Desta forma, após ter uma melhor perceção do modelo que pretendemos implementar na nossa base de dados da rede de transportes públicos, é possível avançar para a conceção de uma nova modelação, após a validação deste modelo, rumo à sua implementação física.

3.7. Validação do modelo de dados produzido

Como forma de avaliar o nosso modelo conceptual para o poder considerar como sendo o modelo final, é necessário verificar se é possível responder aos requisitos previamente definidos.

1. Cada paragem registada no sistema tem de ter associado um id, um nome e as coordenadas da sua localização (latitude e longitude).

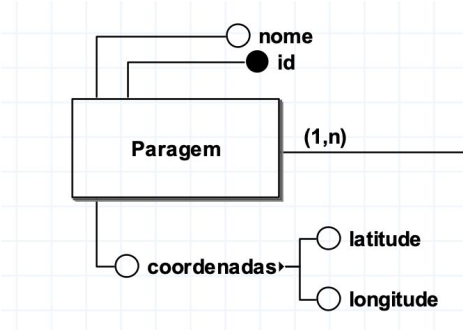

Figura 8 - Atributos de Paragem.

Todos os atributos mencionados no requisitos estão associados à entidade.

2. Um percurso contém várias paragens, sendo armazenada a informação das horas a que cada percurso passa em cada paragem.

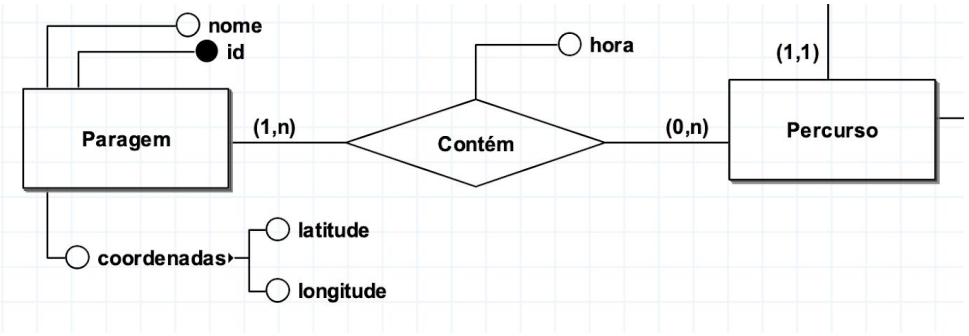

Figura 9 - Relacionamento Paragem-Percurso.

A hora a que cada percurso passa em cada paragem é registada através de um atributo no relacionamento entre as duas entidades.

3. Cada percurso introduzido no sistema é identificado por um número e pela sua respetiva duração.

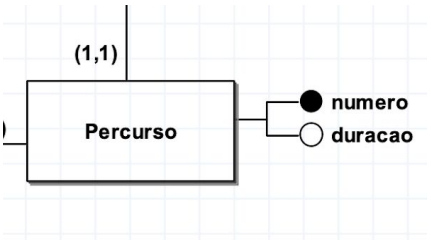

Figura 10 - Atributos de Percurso.

Todos os atributos mencionados no requisitos estão associados à entidade.

4. Uma viagem é realizada num percurso.

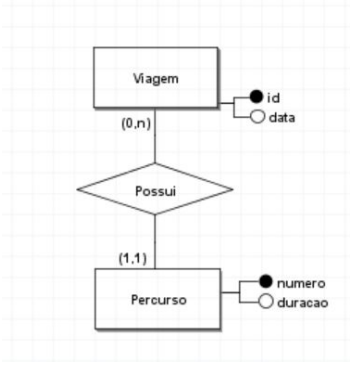

Figura 11 - Relacionamento Percurso-Viagem.

Cada percurso é registado através de um número que está relacionado com a viagem realizada.

5. Quando uma viagem é registada no sistema, é necessário efetuar o registo com um id de viagem e a data da mesma.

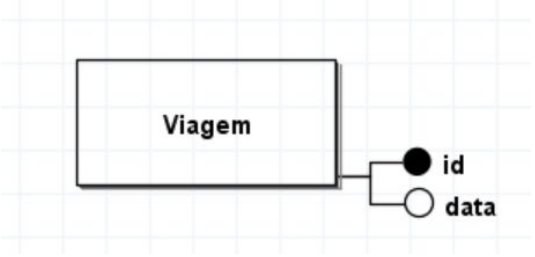

Figura 12 - Atributos de Viagem.

Todos os atributos mencionados no requisitos estão associados à entidade.

6. Vários autocarros podem fazer a mesma viagem, assim como clientes.

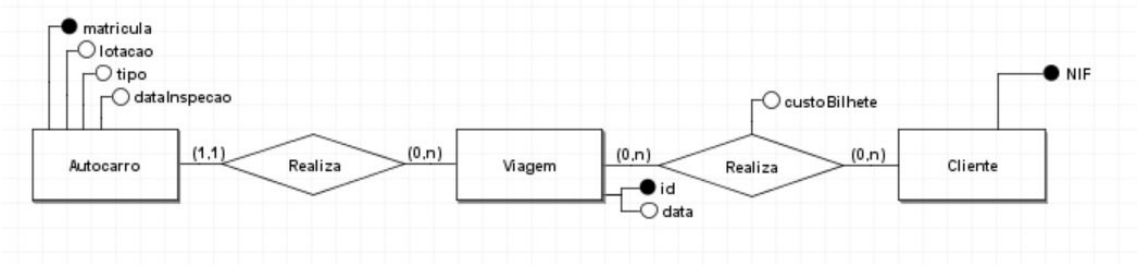

Figura 13 - Relacionamento Autocarro-Viagem, Cliente-Viagem.

Através da Cardinalidade das relações entre as 3 entidades, cumpre-se o requisito.

7. Quando um autocarro é adquirido deve conter informações relativas à sua matrícula, lotação, data de inspeção mais recente e o tipo (o tipo de combustível que consome, i.e., se usa gasolina, gasóleo, gás natural, eletricidade, etc.)

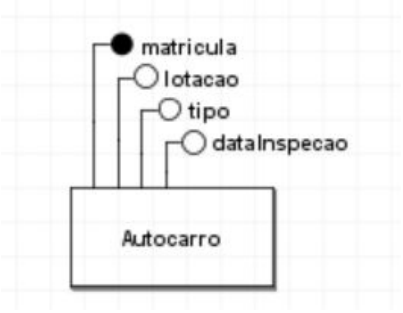

Figura 14 - Atributos de Autocarro.

Todos os atributos mencionados no requisitos estão associados à entidade.

8. Um motorista faz várias viagens, podendo conduzir diferentes autocarros.

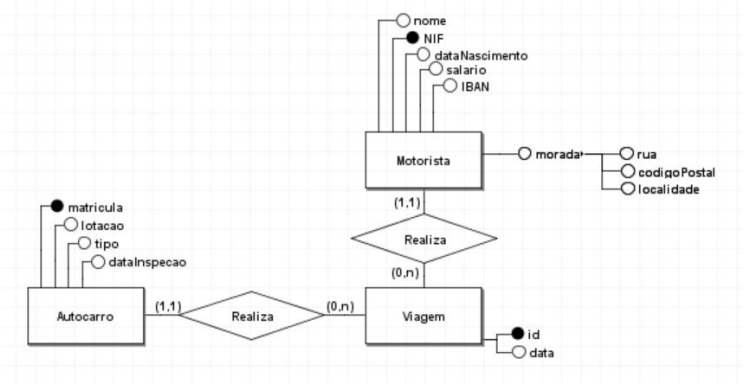

Figura 15 - Relacionamento Autocarro-Viagem, Motorista-Viagem.

9. Quando um motorista é contratado regista-se no sistema o seu nome, NIF, data de nascimento, salário que irá receber, IBAN e morada completa(rua, código postal e localidade).

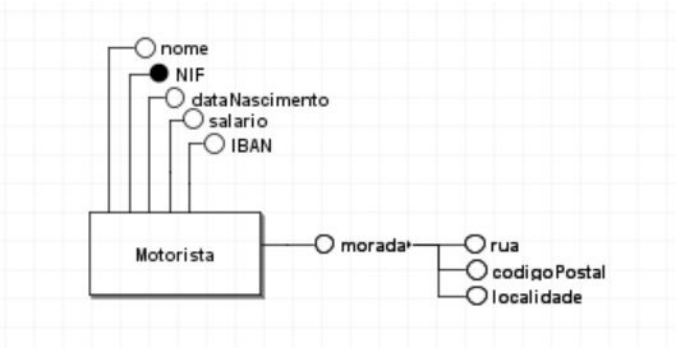

Figura 16 - Atributos de Motorista.

Todos os atributos mencionados no requisitos estão associados à entidade.

10. Os clientes podem fazer viagem, podendo possui um passe ou não, e todas as viagens são registadas no sistema através do seu NIF.

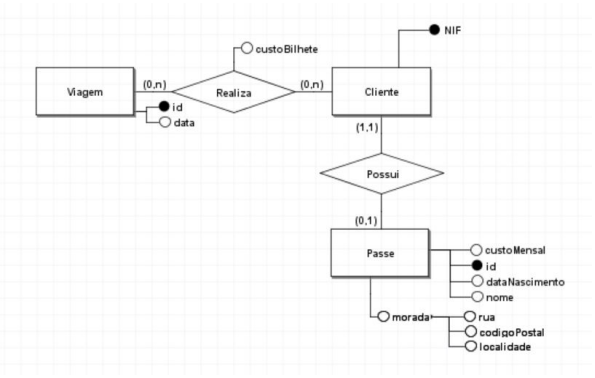

Figura 17 - Relacionamento Cliente-Viagem,Cliente-Passe.

Todos os atributos mencionados no requisito estão associados à entidade. Quando um cliente não possui passe, o preço do billhete é registado através de um atributo no relacionamento entre as duas entidades.

11. Cada viagem que o cliente efetua tem um custo associado.

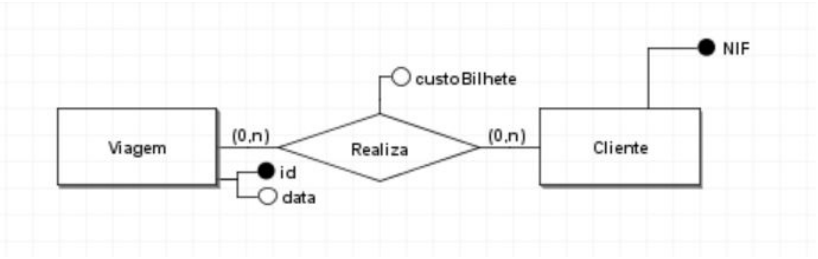

Figura 18 - Relacionamento Cliente-Viagem.

O preço do billhete é registado através de um atributo no relacionamento entre as duas entidades.

12. Um passe é criado a pedido de um cliente, sendo por isso obrigatório registar as suas informações pessoais tais como data de nascimento e nome. Também é associada a informação do tipo de passe e do seu custo mensal.

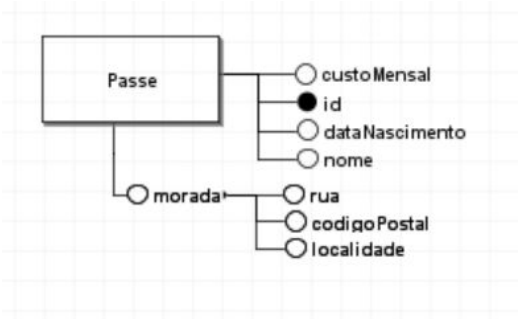

Figura 19 - Atributos de Motorista.

Todos os atributos mencionados no requisitos estão associados à entidade.

Tendo em conta que todos os requisitos são abrangidos pela nossa proposta de modelo conceptual, consideramos este como sendo o nosso modelo conceptual final, e podemos assim avançar para a fase seguinte.

4. Modelação Lógica

4.1. Construção e validação do modelo de dados lógico

O nosso modelo lógico foi construído a partir do nosso modelo conceptual. Aqui as entidades são convertidos em tabelas, assim como os relacionamentos que possuem requisitos associados. Os identificadores de uma entidade passa a designar-se uma *Primary Key* ou *Foreign Key* caso o identificador de uma entidade esteja presente noutra sob a forma de tabela.

Assim, após feita a conversão obtemos nove tabelas, das quais duas foram originadas a partir de relacionamentos. Seguem-se essas tabelas:

1. **Motorista:**
   - 1.1. ***Primary Key***: NIF como um INT
   - 1.2. **Atributos**: nome: VARCHAR(100), dataNascimento: DATE, salario: FLOAT, IBAN: VARCHAR(50), rua: VARCHAR(100), localidade: VARCHAR(30), codigoPostal: VARCHAR(8)
   - 1.3. ***Foreign Key:*** não possui

2. **Viagem:**
   - 2.1. ***Primary Key:*** id como INT
   - 2.2. **Atributo**: data: DATE
   - 2.3. ***Foreign Key:*** matriculaAutocarro: VARCHAR(8), NIFMotorista: INT, numeroPercurso: INT

3. **Autocarro:**
   - 3.1. ***Primary Key:*** matricula como VARCHAR(8)
   - 3.2. **Atributos:** lotacao: INT, tipo: VARCHAR(20), dataInspecao: DATE
   - *3.3.* ***Foreign Key:*** não possui

4. **Percurso:**
   - 4.1. ***Primary Key:*** numero como INT
   - 4.2. **Atributo:** duracao TIME
   - 4.3. ***Foreign Key:*** não possui

5. **Paragem:**
    5.1. ***Primary Key:*** id com INT
    5.2. **Atributos:** nome: VARCHAR(30), latitude: DOUBLE, longitude: DOUBLE
    5.3. ***Foreign Key:*** não possui

6. **Cliente**
    6.1. ***Primary Key:*** NIF como INT
    6.2. **Atributo:** não possui
    6.3. ***Foreign Key:*** não possui

7. **Passe**
    7.1. ***Primary Key:*** id como INT
    7.2. **Atributo:** custoMensal: FLOAT, dataNascimento: DATE, nome: VARCHAR(100), codigoPostal: VARCHAR(8), rua: VARCHAR(100), localidade: VARCHAR(30)
    7.3. ***Foreign Key:*** NIFCliente: INT

Relativamente aos dois relacionamentos:

8. **ParagemPercurso**
    8.1. ***Primary Key:*** não possui
    8.2. **Atributo:** hora: TIME
    8.3. ***Foreign Key:*** idParagem: INT, numeroPercurso: INT
9. ViagemCliente
    9.1. ***Primary Key:*** não possui
    9.2. **Atributo:** custoBilhete: FLOAT
    9.3. ***Foreign Key:*** idViagem: INT, NIFCliente: INT

4.2. Desenho do modelo lógico

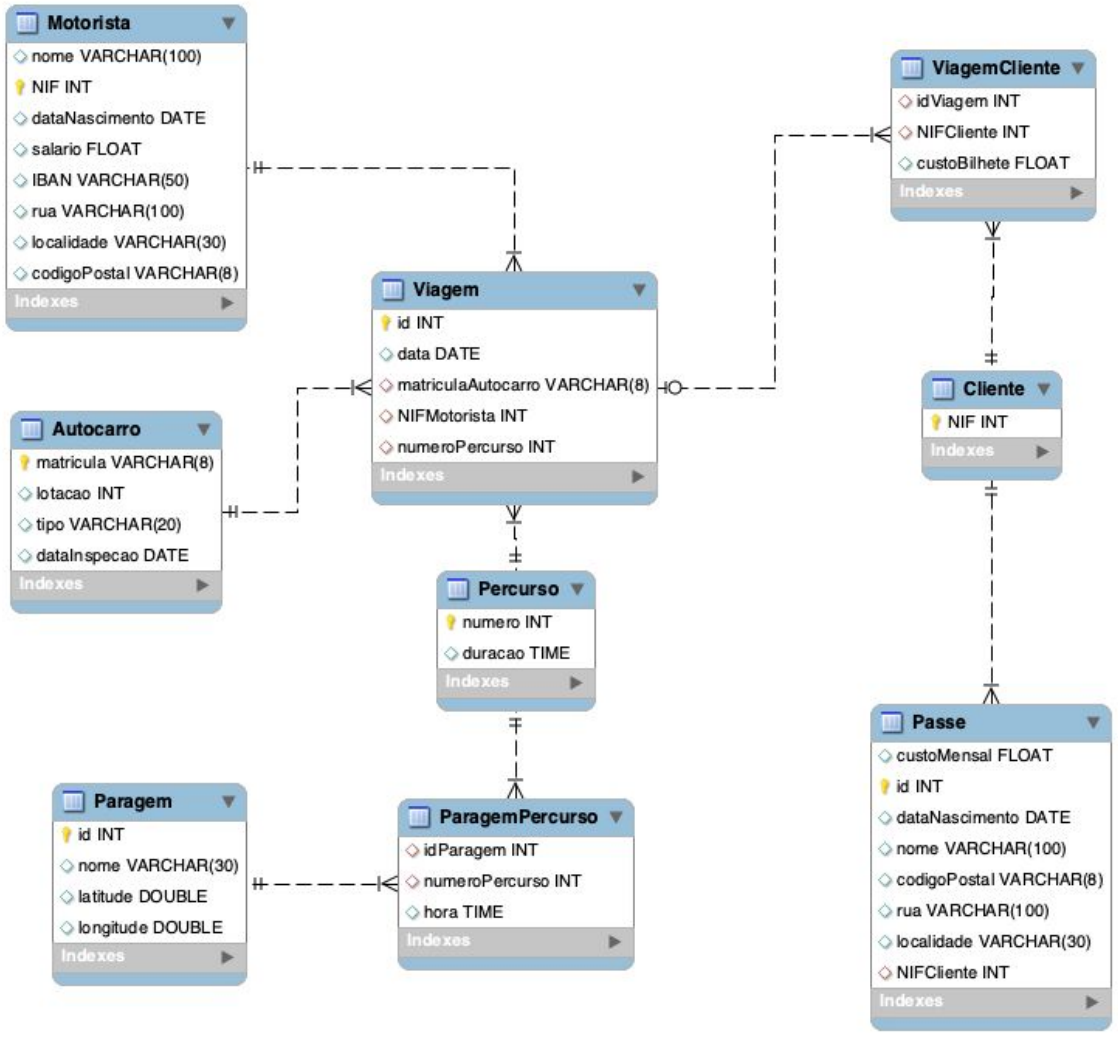

Figura 20 - Modelo Lógico.

4.3. Validação do modelo através da normalização

A normalização tem como principal objetivo evitar a redundância de dados, possibilitando um aumento do desempenho do modelo, já que se evita anomalias em inclusão, exclusão e alteração de registos.

Para avaliar a normalização do modelo temos que verificar se o mesmo satisfaz 3 regras, as fórmulas normais.

A **Primeira Fórmula Normal** (1FN) diz-nos que os atributos necessitam de ser atómicos, ou seja as tabelas não podem ter valores repetidos nem pode existir nenhum atributo multivalorado. Com uma simples análise às tabelas do nosso modelo, verificamos que tal não acontece, tornando válida a 1FN.

A **Segunda Fórmula Normal** (2FN), indica que só pode ser cumprida se a primeira for satisfeita, diz-nos que os atributos normais devem depender apenas da chave primária da tabela. Tomemos como exemplo a data de inspeção do autocarro, este atributo depende apenas da chave primária, matrícula. O mesmo acontece com o restante modelo, validando a 2FN.

Por fim, a **Terceira Fórmula Normal** (3FN) diz-nos que estando válidas as 2 anteriores fórmulas, devemos verificar se existem atributos dependentes de outros, ou seja se existe algum atributo que pode ser gerado a partir de outro. No caso do nosso modelo, nenhum atributo é proporcional ou possível de obter a partir de outro, o que cumpre a 3FN.

Desta forma, podemos afirmar que o modelo é válido em termos de normalização.

4.4. Validação do modelo com interrogações do utilizador

Para validar o nosso modelo lógico desenvolvido selecionamos algumas interrogações dos requisitos de exploração e, usando álgebra relacional, mostramos de que forma o modelo as satisfaz.

a. Contabilizar o total de viagens realizadas por dia.

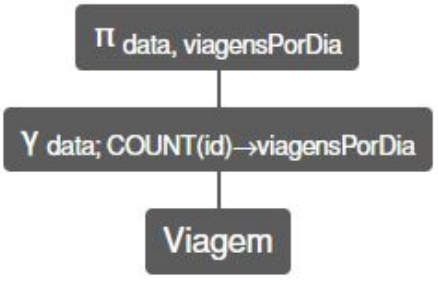

Figura 21 - Árvore correspondente à primeira interrogação.

Primeiro são selecionados os componentes da relação Viagem, depois agrupamos estes componentes pela data dos mesmos e por cada componente agregado contamos os ids a que essa data corresponde, dando a este novo componente o nome viagensPorDia. Por fim projetamos os componentes data e viagensPorDia da nova relação.

b. Contabilizar os clientes que realizaram mais viagens.

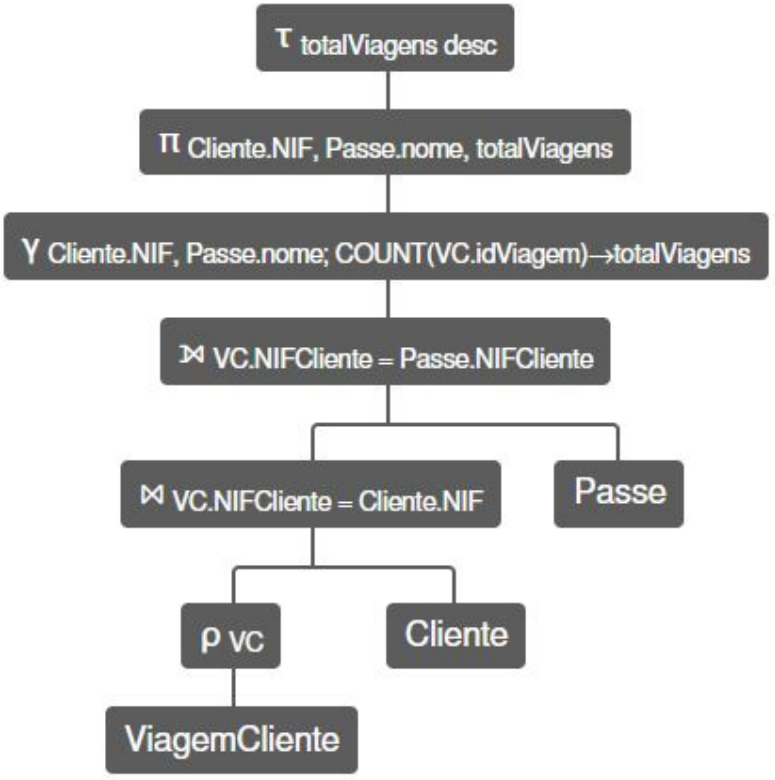

Figura 22 - Árvore correspondente à segunda interrogação.

Primeiro selecionamos a relação ViagemCliente, renomeando-a para VC, e a relação Cliente, unindo-as através do NIF do Cliente. Unimos depois esta nova relação à relação Passe também através do NIF do Cliente. Contudo, esta união é feita pela esquerda, ou seja, se houver um Cliente sem Passe associado terá componentes associados ao Passe nulos. Se a união fosse normal, Clientes sem Passe associado não seriam incluídos na nova relação. Tendo esta relação, podemos agora agrupá-la com base nos componentes NIFCliente e nome, agregando os componentes idViagem e retornando a sua quantidade, dando origem a um novo componente, totalViagens. Podemos assim projetar os componentes NIF, nome e totalViagens desta última relação, ordenando-os segundo o componente totalViagens de forma decrescente.

c. Contabilizar o lucro total obtido com a venda de bilhetes por percurso.

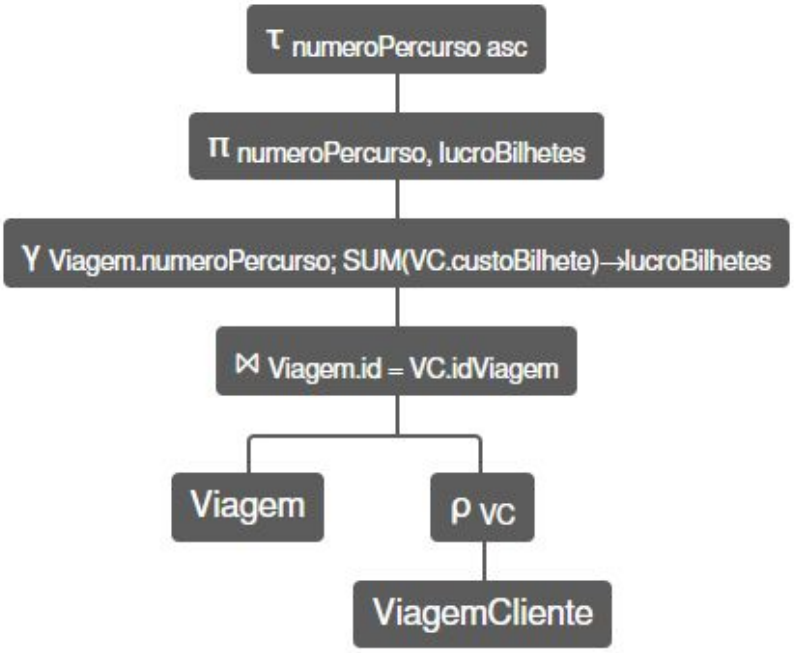

Figura 23 - Árvore correspondente à terceira interrogação.

Inicialmente, selecionamos as relações Viagem e ViagemCliente (renomeada para VC) e unimo-las através do componente idViagem. De seguida, agrupamos esta nova relação pelo componente numeroPercurso, agregando a soma dos componentes custoBilhete no novo componente lucroBilhetes. Por fim, projetamos os componentes numeroPercurso e lucroBilhetes, ordenados de acordo com lucroBilhetes de modo ascendente.

d. Contabilizar os clientes que realizaram uma dada viagem.

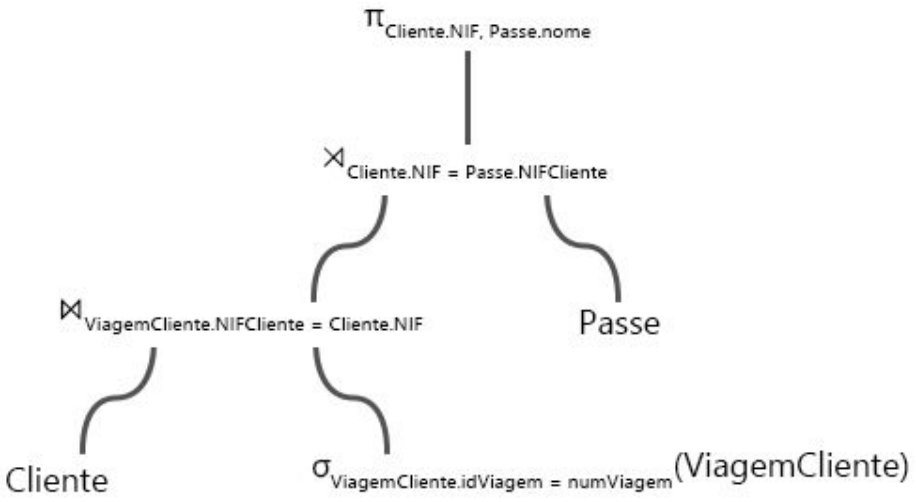

Figura 24 - Árvore correspondente à quarta interrogação

Primeiro selecionamos da relação ViagemCliente os elementos com o componente idViagem igual ao id da viagem pretendida. Depois unimos essa relação à relação Cliente pelo componente NIFCliente, e unimos esta relação unida à relação Passe pelo componente NIFCliente mais uma vez, desta vez pela esquerda. Por fim projetamos os componentes NIF e nome.

e. Contabilizar os autocarros que realizaram viagens num determinado dia.

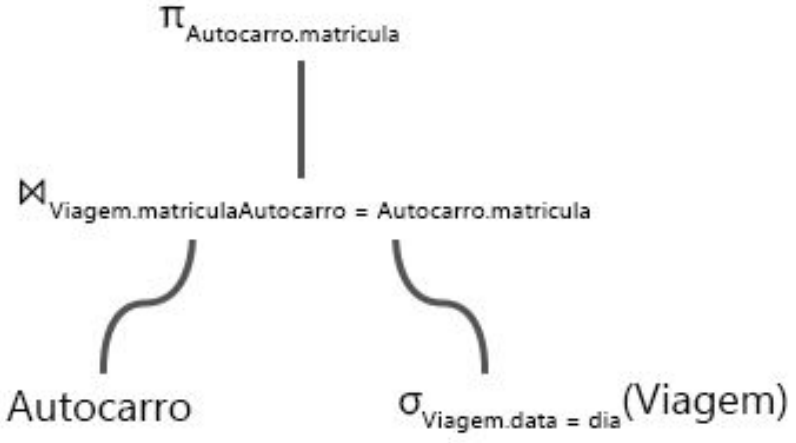

Figura 25 - Árvore correspondente à quinta interrogação.

Começamos por selecionar da relação Viagem os elementos com o componente data igual ao dia pretendido. Depois unimos essa relação à relação Autocarro pelo componente matriculaAutocarro. Finalmente, projetamos o componente matrícula.

4.5. Revisão do modelo lógico produzido

Depois de concluído o esboço inicial relativo às tabelas que iriam constituir o modelo lógico, onde identificamos previamente as chaves primárias e estrangeiras pertencentes a cada tabela, a sua representação gráfica foi instantânea, utilizando a mesma multiplicidade entre relacionamentos e entidades, nas tabelas.

Posteriormente, realizamos a validação do modelo lógico através da normalização, podendo assim concluir que o nosso modelo lógico esboçado inicialmente seria válido.

Para além disso, a validação com interrogações do utilizador foi o passo essencial que nos permitiu concluir e tornar o nosso esboço do modelo como sendo o modelo lógico final. Isto deve-se ao facto de termos podido verificar que os nossos requisitos de exploração podem ser obtidos através da implementação física da nossa base de dados de uma rede de transportes públicos.

Também numa perspetiva de antever a implementação física, verificamos que a qualquer momento poderia ser possível expandir a base de dados, e consequentemente a empresa, quer seja a contratar motoristas, a comprar autocarros, adicionar paragens, entre outros, pois, para isso, é apenas necessário criar cada uma destas entidades e adicioná-la ao nosso sistema.

5. Implementação Física

5.1. Seleção do sistema de gestão de bases de dados

Decidimos usar o MySQL como DBMS visto que foi este que nos foi introduzido na UC e, desta forma, podemos usar o MySQL Workbench para o desenvolvimento da base da dados, um programa que já sabemos como utilizar. Também poderíamos ter usado o MariaDB, um *fork* do MySQL desenvolvido por alguns dos criadores do MySQL, bastante utilizado em sistemas Linux e compatível com o MySQL Workbench, mas como elaborámos a base de dados em Windows não achámos necessário usar esse DBMS, sendo porém uma alternativa viável e que teremos em consideração no futuro.

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

Para traduzir o esquema lógico para o MySQL usamos o comando “converter para físico” do brModelo no nosso esquema lógico. Este comando gera o código necessário para criar a base de dados em SQL com base no esquema lógico que concebemos anteriormente. Depois apenas tivemos que copiar esse código para um ficheiro .sql, adicionar dois comandos para criar um *schema* e usá-lo por defeito e ao correr este ficheiro fomos capazes de criar a nossa base de dados em SQL.

5.3. Tradução das interrogações do utilizador para SQL (alguns exemplos)

A partir das árvores geradas na secção anterior fomos capazes de converter as interrogações do utilizador para SQL de uma forma rápida e eficaz.

a. **Contabilizar o total de viagens realizadas por dia.**

```
delimiter $$
CREATE PROCEDURE viagensPorDia()
BEGIN
    SELECT `data`, COUNT(id) AS viagensPorDia FROM Viagem
    GROUP BY `data`;
END $$
```

Figura 26 - Código SQL correspondente à primeira interrogação.

b. **Contabilizar os clientes que realizaram mais viagens.**

```
delimiter $$
CREATE PROCEDURE viagensPorCliente()
BEGIN
    SELECT Cliente.NIF, Passe.nome, count(VC.idViagem) AS totalViagens
    FROM ViagemCliente AS VC
    JOIN Cliente ON VC.NIFCliente = Cliente.NIF
    LEFT JOIN Passe ON VC.NIFCliente = Passe.NIFCliente
    GROUP BY Cliente.NIF, Passe.nome
    ORDER BY totalViagens desc;
END $$
```

Figura 27 - Código SQL correspondente à segunda interrogação.

c. **Contabilizar o lucro total obtido com a venda de bilhetes por percurso.**

```
delimiter $$
CREATE PROCEDURE lucroPorPercurso()
BEGIN
    SELECT numeroPercurso, sum(VC.custoBilhete) AS lucroBilhetes FROM Viagem
    JOIN ViagemCliente AS VC ON Viagem.id = VC.idViagem
    GROUP BY Viagem.numeroPercurso
    ORDER BY numeroPercurso;
END $$
```

Figura 28 - Código SQL correspondente à terceira interrogação.

d. **Contabilizar os clientes que realizaram uma dada viagem.**

```
delimiter $$
CREATE PROCEDURE clientesPorViagem(IN numViagem int)
BEGIN
    SELECT Cliente.NIF, Passe.nome FROM ViagemCliente
    JOIN Cliente on ViagemCliente.NIFCliente = Cliente.NIF
    LEFT JOIN Passe on Cliente.NIF = Passe.NIFCliente
    WHERE ViagemCliente.idViagem = numViagem;
END $$
```

Figura 29 - Código SQL correspondente à quarta interrogação.

e. **Contabilizar os autocarros que realizaram viagens num determinado dia.**

```
delimiter $$
CREATE PROCEDURE autocarrosQueViajaram(IN dia DATE)
BEGIN
    SELECT Autocarro.matricula FROM Autocarro
    JOIN Viagem on Viagem.matriculaAutocarro = Autocarro.matricula
    WHERE Viagem.data = dia;
END $$
```

Figura 30 - Código SQL correspondente à quinta interrogação.

5.4. Escolha, definição e caracterização de índices em SQL (alguns exemplos)

Ao criar um objeto, são automaticamente criados índices para as colunas com restrição PRIMARY KEY ou UNIQUE, chamados índices implícitos. Porém, nós também podemos definir os nossos próprios índices, ou seja, índices explícitos. A vantagem de criar índices é que estes aceleram operações de seleção e consulta (SELECT e WHERE, por exemplo), mas por outro lado abrandam operações de inserção e atualização (INSERT e UPDATE, por exemplo).

Sendo assim, devemos tentar apenas definir índices para colunas que são consultadas com frequência mas não são atualizadas com frequência. Os índices serão mais eficazes em tabelas com muitas entradas, por isso não vale a pena definir índices em tabelas com poucas entradas, como as tabelas Motorista e Autocarro.

Uma boa escolha de coluna para definir um índice é a coluna nome da tabela Passe, por exemplo, visto que, ao procurar por entradas na tabela dos passes, uma tabela que pode conter várias entradas, a primeira opção será procurar pelo nome, em vez de procurar pelo id do passe, a chave primária. Assim, para criar este índice, usamos o seguinte comando:

```
CREATE INDEX NomePasse ON Passe (nome);
```

Figura 31 - Comando para criar o índice NomePasse na coluna nome da tabela Passe.

Outro índice que poderá valer a pena criar é na coluna data da tabela Viagem, visto que esta tabela conterá imensas entradas, e pode ser útil poder encontrar entradas nesta com base na sua data.

```
CREATE INDEX DataViagem ON Viagem (data);
```

Figura 32 - Comando para criar o índice DataViagem na coluna data da tabela Viagem.

5.5. Estimativa do espaço em disco da base de dados e taxa de crescimento anual

Tendo em conta os conhecimentos prévios relativamente ao tamanho ocupado por *datatypes* como Int, String, Float, entre outros, calculamos o espaço ocupado em disco por cada atributo pertencente a cada tabela, de forma a gerar uma estimativa da ocupação total em disco da nossa base de dados.

Motorista

Atributos	Tipo	Tamanho (bytes)
NIF	INT	4
nome	VARCHAR(100)	102
dataNascimento	DATE	3
salario	FLOAT	8
IBAN	VARCHAR(50)	52
rua	VARCHAR(100)	102
localidade	VARCHAR(30)	32
codigoPostal	VARCHAR(8)	10
TOTAL	**-**	**313**

Tabela 8 - Cálculo dos bytes associados à entidade Motorista.

Viagem

Atributos	Tipo	Tamanho (bytes)
id	INT	4
data	DATE	3
matriculoAutocarro	VARCHAR(8)	10
NIFMotorista	INT	4
numeroPercurso	INT	4
TOTAL	**-**	**25**

Tabela 9 - Cálculo dos bytes associados à entidade Viagem.

Autocarro

Atributos	Tipo	Tamanho (bytes)
matricula	VARCHAR(8)	10
lotacao	INT	4
tipo	VARCHAR(20)	22
dataInspecao	DATE	3
TOTAL	**-**	**39**

Tabela 10 - Cálculo dos bytes associados à entidade Autocarro.

Percurso

Atributos	Tipo	Tamanho (bytes)
numero	INT	4
duracao	TIME	5
TOTAL	**-**	**9**

Tabela 11 - Cálculo dos bytes associados à entidade Percurso.

Paragem

Atributos	Tipo	Tamanho (bytes)
nome	VARCHAR(30)	32
id	INT	4
latitude	DOUBLE	16
longitude	DOUBLE	16
TOTAL	**-**	**68**

Tabela 12 - Cálculo dos bytes associados à entidade Paragem.

Cliente

Atributos	Tipo	Tamanho (bytes)
NIF	INT	4
TOTAL	**-**	**4**

Tabela 13 - Cálculo dos bytes associados à entidade Cliente.

Passe

Atributos	Tipo	Tamanho (bytes)
id	INT	4
custoMensal	FLOAT	8
dataNascimento	DATE	3
nome	VARCHAR(100)	102
codigoPostal	VARCHAR(8)	10
rua	VARCHAR(100)	102
localidade	VARCHAR(30)	32
NIFCliente	INT	4
TOTAL	**-**	**265**

Tabela 14 - Cálculo dos bytes associados à entidade Passe.

Paragem-Percurso

Atributos	Tipo	Tamanho (bytes)
hora	TIME	5
idParagem	INT	4
numeroPercurso	INT	4
TOTAL	**-**	**13**

Tabela 15 - Cálculo dos bytes associados ao relacionamento Paragem-Percurso.

Viagem-Cliente

Atributos	Tipo	Tamanho (bytes)
custoBilhete	FLOAT	8
idViagem	INT	4
nifCliente	INT	4
TOTAL	**-**	**16**

Tabela 16 - Cálculo dos bytes associados ao relacionamento Viagem-Cliente.

Desta forma, o tamanho total da nossa base de dados seria, sem povoamento, 752 bytes. No entanto, para uma estimativa real, tendo em conta o povoamento do nosso modelo faria um total de:

23*4 + 265*13 + 313*5 + 68*11 + 9*3 + 13*17 + 5* 39 + 62*25 + 304*16 = 12 707 bytes = 13 kB

É de notar que estes valores são uma mera estimativa sobre o primeiro mês de utilização desta base de dados. Seguindo este modelo de utilização, no final do primeiro ano teríamos 153 kB de espaço em disco na nossa base de dados.

Segundo um estudo do Instituto Nacional de Estatística, registou-se um aumento de 5,5% em 2018 no número de passageiros deste tipo de transporte. Tendo em conta que o crescimento de número de passageiros leva a uma consequente necessidade do reforço de autocarro, motoristas e viagens, bem como investimentos na expansão de percursos e paragens, este valor poderá ser ainda maior do que o estimado, apenas com o aumento de passageiros.

5.6. Definição e caracterização das vistas de utilização em SQL (alguns exemplos)

- **Vista sobre os passes**

```
CREATE VIEW viewPasses AS
    SELECT id, nome, dataNascimento, rua, codigoPostal,
        localidade, NIFCliente AS NIF, custoMensal FROM Passe;
```

Figura 33 - Código que implementa a vista sobre os passes.

- **Vista sobre as viagens**

```
CREATE VIEW viewViagens AS
    SELECT Viagem.id, data, matriculaAutocarro, Motorista.nome AS NomeMotorista,
        count(DISTINCT VC.NIFCliente) AS Passageiros,
        group_concat(DISTINCT Paragem.nome ORDER BY PP.hora asc) AS Paragens, duracao FROM Viagem
    JOIN ViagemCliente AS VC ON VC.idViagem = Viagem.id
    JOIN Motorista ON Viagem.NIFMotorista = Motorista.NIF
    JOIN Percurso on Viagem.numeroPercurso = Percurso.numero
    JOIN ParagemPercurso AS PP on PP.numeroPercurso = Percurso.numero
    JOIN Paragem on PP.idParagem = Paragem.id
    GROUP BY Viagem.id, Percurso.numero;
```

Figura 34 - Código que implementa a vista sobre os passes.

Nesta vista, as paragens aparecem como uma lista de nomes, pela ordem na qual são visitadas na viagem.

5.7. Revisão do sistema implementado

Após a implementação do sistema, este é capaz de responder a todos os nossos requisitos, ocupando um espaço em disco aceitável para o tipo de dados que estamos a tratar, gerindo eficientemente os mesmos.

O nosso sistema consegue responder às interrogações do utilizador e fá-lo cumprindo as métricas de normalização.

A sua implementação física traduz em linguagem SQL todas as ideias representadas no modelo lógico, e, portanto, é nada mais que a descrição de como estes dados se armazenam. Esta implementação é também ela validada, o que nos revela que foi corretamente concretizada.

6. Conclusões e Trabalho Futuro

Inicialmente, começamos por analisar detalhadamente as necessidades que o sistema teria que ser capaz de suportar, ou seja aquilo que era importante para garantir que estaríamos a criar um sistema de informação completo, coeso e responsável. Seria, além disso, indispensável certificar-nos que a Base de Dados é capaz de responder às obrigações que o sistema necessita.

Ao longo do projeto, modelamos este sistema através do modelo conceptual, aquele que está com o maior nível de abstração, desenvolvido a partir do levantamento dos requisitos, revelando as entidades, atributos e relacionamentos através do modelo lógico onde se descrevem as tabelas de dados, as suas ligações, chaves privadas, chaves estrangeiras e atributos, e, por fim, por meio do modelo físico onde é descrito, em linguagem SQL, como será feito o armazenamento de dados.

No decorrer do mesmo, fomos efetuando validações do modelo, de maneira a nos certificarmos que aquilo que realizamos cumpria todos os requisitos e normas de gestão de Base de Dados. Chegamos à conclusão que os modelos eram válidos, tanto para a normalização como para as interrogações do utilizador, revelando que as métricas comuns de avaliação estariam a ser satisfeitas.

Por último, é importante referir a gestão eficiente de dados. Além da preocupação de satisfazer os requisitos levantados, este facto também foi considerado por nós, como por exemplo na escolha rigorosa e cuidada que fizemos para os atributos das entidades, onde estudamos quais eram as melhores opções para as mesmas.

Relativamente a um trabalho futuro, consideramos que existem aspetos a melhorar, nomeadamente, otimizar a organização do registo das informações necessárias ao funcionamento da nossa rede de transportes, como forma de diminuir o espaço necessário ao armazenamento em disco. Certamente que, com o crescimento gradual da nossa base de dados, iríamos sentir a necessidade de rentabilizar o modo de gestão das informações que são guardadas, adicionando e/ou removendo atributos a determinadas entidades.

Em suma, temos confiança que o nosso sistema de informação estará apto a sustentar uma rede de transportes públicos cumprindo todas as exigências propostas, o que torna a nossa Base de Dados competente para os objetivos inicialmente traçados.

Referências

- Rlsystem.com.br. n.d. *Tipos De Dados (Data Types) No SQL Server - RL System*. [online] Available at: <https://www.rlsystem.com.br/tipos-dados-sql-server/> [Accessed 4 December 2020].
- 2020. *Estatísticas Dos Transportes E Comunicações 2018*. 2nd ed. Lisbon: INE.
- Olofsson, A., n.d. *SQL Server Data Types Reference*. [online] connectionstrings.com. Available at: <https://www.connectionstrings.com/sql-server-data-types-reference/> [Accessed 4 December 2020].

Lista de Siglas e Acrónimos

DBMS Sistema de Gestão de Base de Dados (*DataBase Management System*)

NIF Número de Identificação Fiscal

SQL *Structured Query Language*

ER Entidade Relação